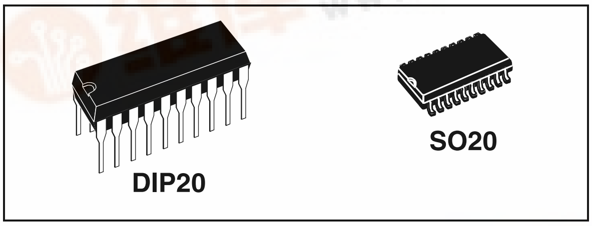
ST7LITE3

8-BIT MCU WITH SINGLE VOLTAGE FLASH, DATA EEPROM, ADC, TIMERS, SPI, LINSCI™

- **Memories**
  - 8 Kbytes program memory: single voltage extended Flash (XFlash) Program memory with read-out protection, In-Circuit Programming and In-Application programming (ICP and IAP), data retention: 20 years at 55°C.
  - 384 bytes RAM
  - 256 bytes data EEPROM with read-out protection. 300K write/erase cycles guaranteed, 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 RC1% oscillator, crystal/ceramic resonator or external clock
  - Optional x4 or x8 PLL for 4 or 8 MHz internal clock
  - Five Power Saving Modes: Halt, Active-Halt, Wait and Slow, Auto Wake Up From Halt
- **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 realtime base and 1 input capture
  - Two 12-bit Auto-reload Timers with 4 PWM outputs, input capture and output compare functions



- **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)

Device Summary

Features	ST7LITE30	ST7LITE35	ST7LITE39
Program memory - bytes	8K		
RAM (stack) - bytes	384 (128)		
Data EEPROM - bytes	-	-	256
Peripherals	Lite Timer, Autoreload Timer, SPI, LINSCI, 10-bit ADC		
Operating Supply	2.7V to 5.5V		
CPU Frequency	Up to 8Mhz (w/ ext OSC up to 16MHz)	Up to 8Mhz (w/ ext OSC up to 16MHz and int 1MHz RC 1% PLLx8/4MHz)	
Operating Temperature	-40°C to +85°C		
Packages	SO20 300”, DIP20		

Rev. 4.0

Table of Contents

Table of Contents

1 INTRODUCTION

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

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

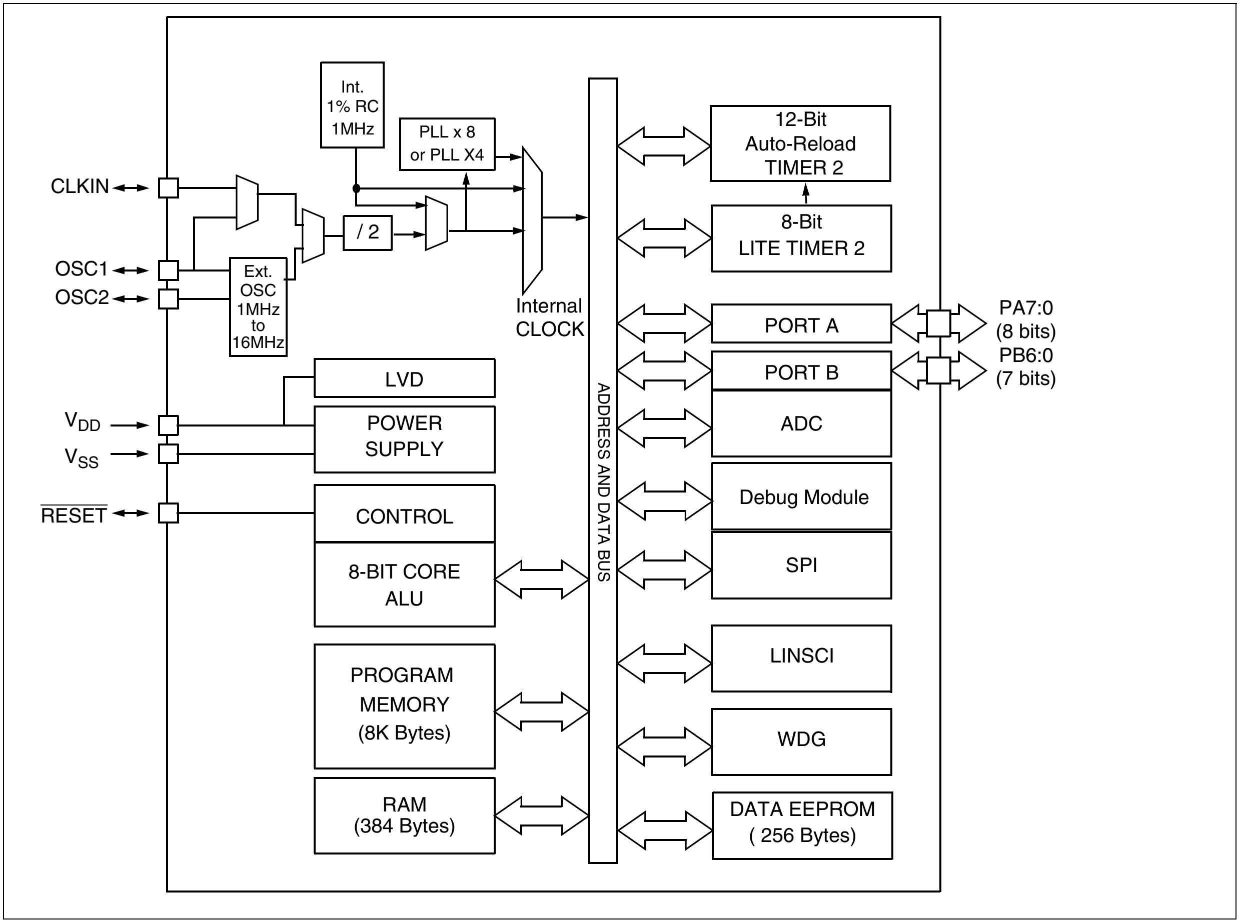
Under software control, the ST7LITE3 device can be placed in WAIT, SLOW, or HALT mode, reducing power consumption when the application is in idle or standby state.

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

For easy reference, all parametric data are located in section 13 on page 129.

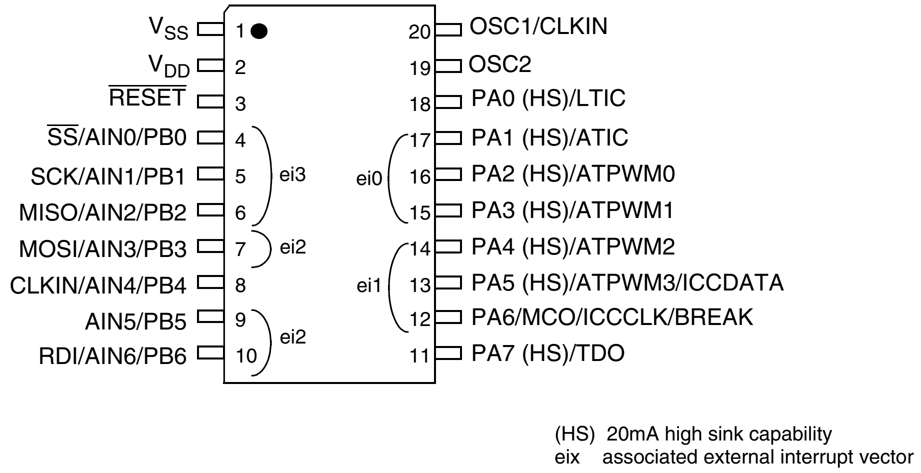
The devices feature an on-chip Debug Module (DM) to support in-circuit debugging (ICD). For a description of the DM registers, refer to the ST7 ICC Protocol Reference Manual.

Figure 1. General Block Diagram



2 PIN DESCRIPTION

Figure 2. 20-Pin SO and DIP Package Pinout



PIN DESCRIPTION (Cont'd)

Legend / Abbreviations for Table 1:

Type: I = input, O = output, S = supply

In/Output level: C_T= CMOS 0.3V_{DD}/0.7V_{DD} with input trigger

Output level: HS = 20mA high sink (on N-buffer only)

Port and control configuration:

- Input: float = floating, wpu = weak pull-up, int = interrupt, ana = analog
- Output: OD = open drain, PP = push-pull

The RESET configuration of each pin is shown in bold which is valid as long as the device is in reset state.

Table 1. Device Pin Description

Pin No.	Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
					Input				Output			
			Input	Output	float	wpu	int	ana	OD	PP		
1	V_{SS}	S									Ground	
2	V_{DD}	S									Main power supply	
3	$\overline{\text{RESET}}$	I/O	C_T			X			X		Top priority non maskable interrupt (active low)	
4	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) **Caution:** No negative current injection allowed on this pin. For details, refer to section 13.2.2 on page 130
5	PB1/AIN1/SCK	I/O	C_T		**X**			X	X	X	**Port B1**	ADC Analog Input 1 or SPI Serial Clock **Caution:** No negative current injection allowed on this pin. For details, refer to section 13.2.2 on page 130
6	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	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	PB4/AIN4/ CLKIN**	I/O	C_T		**X**	X		X	X	X	**Port B4**	ADC Analog Input 4 or External clock input
9	PB5/AIN5	I/O	C_T		**X**	ei2		X	X	X	**Port B5**	ADC Analog Input 5
10	PB6/AIN6/RDI	I/O	C_T		**X**			X	X	X	**Port B6**	ADC Analog Input 6 or LINSCI Input
11	PA7/TDO	I/O	C_T	HS	**X**	X			X	X	**Port A7**	LINSCI Output

Pin No.	Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
			Input	Output	Input				Output			
					float	wpu	int	ana	OD	PP		
12	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 10k mandatory in noisy environment). This is to avoid entering ICC mode unexpectedly during a reset. In the application, even if the pin is configured as output, any reset will put it back in input pull-up.
13	PA5 /ATPWM3/ ICCDATA	I/O	C_T	HS	**X**				X	X	**Port A5**	Auto-Reload Timer PWM3 or In Circuit Communication Data
14	PA4/ATPWM2	I/O	C_T	HS	**X**				X	X	**Port A4**	Auto-Reload Timer PWM2
15	PA3/ATPWM1	I/O	C_T	HS	**X**	ei0			X	X	**Port A3**	Auto-Reload Timer PWM1
16	PA2/ATPWM0	I/O	C_T	HS	**X**				X	X	**Port A2**	Auto-Reload Timer PWM0
17	PA1/ATIC	I/O	C_T	HS	**X**				X	X	**Port A1**	Auto-Reload Timer Input Capture
18	PA0/LTIC	I/O	C_T	HS	**X**	X			X	X	**Port A0**	Lite Timer Input Capture
19	OSC2	O									Resonator oscillator inverter output	
20	OSC1/CLKIN	I									Resonator oscillator inverter input or External clock input	

Note:

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 & MEMORY MAP

As shown in Figure 3, the MCU is capable of addressing 64K bytes 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 8 Kbytes of user program memory. The RAM space includes up to 128 bytes for the stack from 180h to 1FFh.

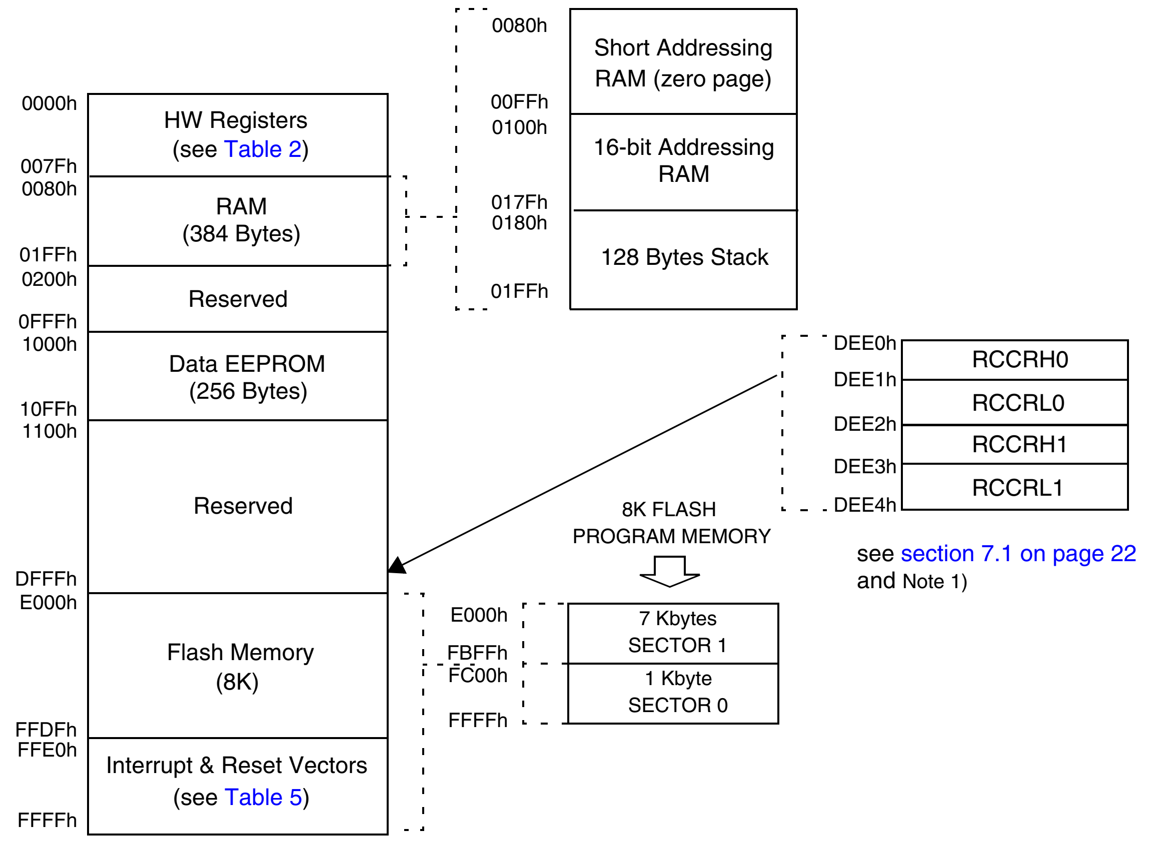
The highest address bytes contain the user reset and interrupt vectors.

The Flash memory contains two sectors (see Figure 3) 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.

IMPORTANT: Memory locations marked as “Reserved” must never be accessed. Accessing a reseved area can have unpredictable effects on the device.

Figure 3. 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 read out protection removal), then the RC calibration values can still be obtained through these addresses.

Table 2. 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 Auto-reload Register	00h	R/W
000Ah		LTCNTR	Lite Timer Counter Register	00h	Read Only
000Bh		LTCSR1	Lite Timer Control/Status Register 1	0X00 0000h	R/W
000Ch		LTICR	Lite Timer Input Capture Register	xxh	Read Only
000Dh	AUTO-RELOAD TIMER 3	ATCSR	Timer Control/Status Register	0X00 0000h	R/W
000Eh		CNTR1H	Counter Register 1 High	00h	Read Only
000Fh		CNTR1L	Counter Register 1 Low	00h	Read Only
0010h		ATR1H	Auto-Reload Register 1 High	00h	R/W
0011h		ATR1L	Auto-Reload 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	Read Only
0020h		ATICRL	Input Capture Register Low	00h	Read Only
0021h		ATCSR2	Timer Control/Status Register 2	03h	R/W
0022h		BREAKCR	Break Control Register	00h	R/W
0023h		ATR2H	Auto-Reload Register 2 High	00h	R/W
0024h		ATR2L	Auto-Reload 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

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 Read Only 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 0000 0XX0h	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	Read Only 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)				

Legend: x=undefined, R/W=read/write

Notes:

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 DM registers, see the ST7 ICC 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 or In-Application Programming.

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

4.2 Main Features

- ICP (In-Circuit Programming)
- IAP (In-Application Programming)
- ICT (In-Circuit Testing) for downloading and executing user application test patterns in RAM
- Sector 0 size configurable by option byte
- Read-out 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 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 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).

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored etc.)

IAP 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.

FLASH PROGRAM MEMORY (Cont'd)

4.4 ICC interface

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

- $\overline{\text{RESET}}$: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- PB4/OSC1: main clock input for external source (not required on devices without OSC1/OSC2 pins)
- V_{DD}: application board power supply (optional, see Note 3)

Notes:

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

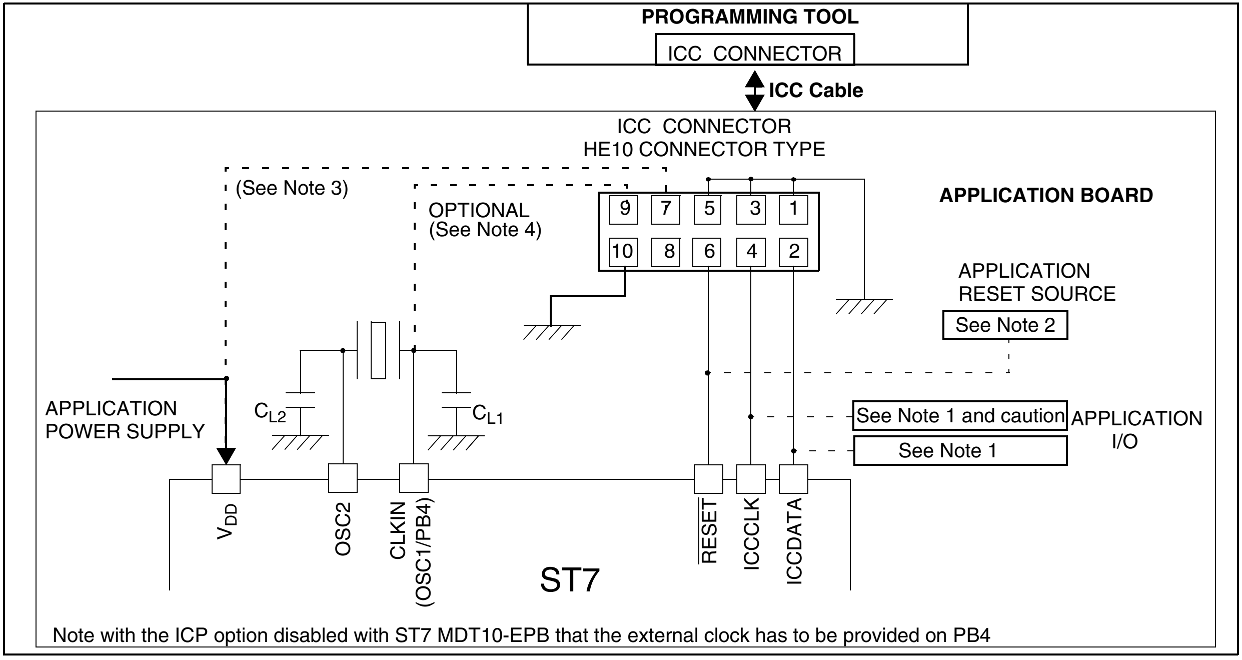
2. During the ICP session, the programming tool must control the $\overline{\text{RESET}}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor<1K). A schottky diode can be used to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

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

4. Pin 9 has to be connected to the 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 need to have OSC2 grounded in this case.

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

Figure 4. Typical ICC Interface



Note with the ICP option disabled with ST7 MDT10-EPB that the external clock has to be provided on PB4

FLASH PROGRAM MEMORY (Cont'd)

4.5 Memory Protection

There are two different types of memory protection: Read Out Protection and Write/Erase Protection which can be applied individually.

4.5.1 Read out Protection

Readout protection, when selected provides a protection 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 E^2 memory are protected.

In flash devices, this protection is removed by reprogramming the option. In this case, both program and data E^2 memory are automatically erased and the device can be reprogrammed.

– Read-out protection selection is enabled and removed through the FMP_R bit in the option byte.

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 E^2 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)
Read/Write
Reset Value: 000 0000 (00h)
1st RASS Key: 0101 0110 (56h)
2nd RASS Key: 1010 1110 (AEh)

7							0
0	0	0	0	0	OPT	LAT	PGM

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

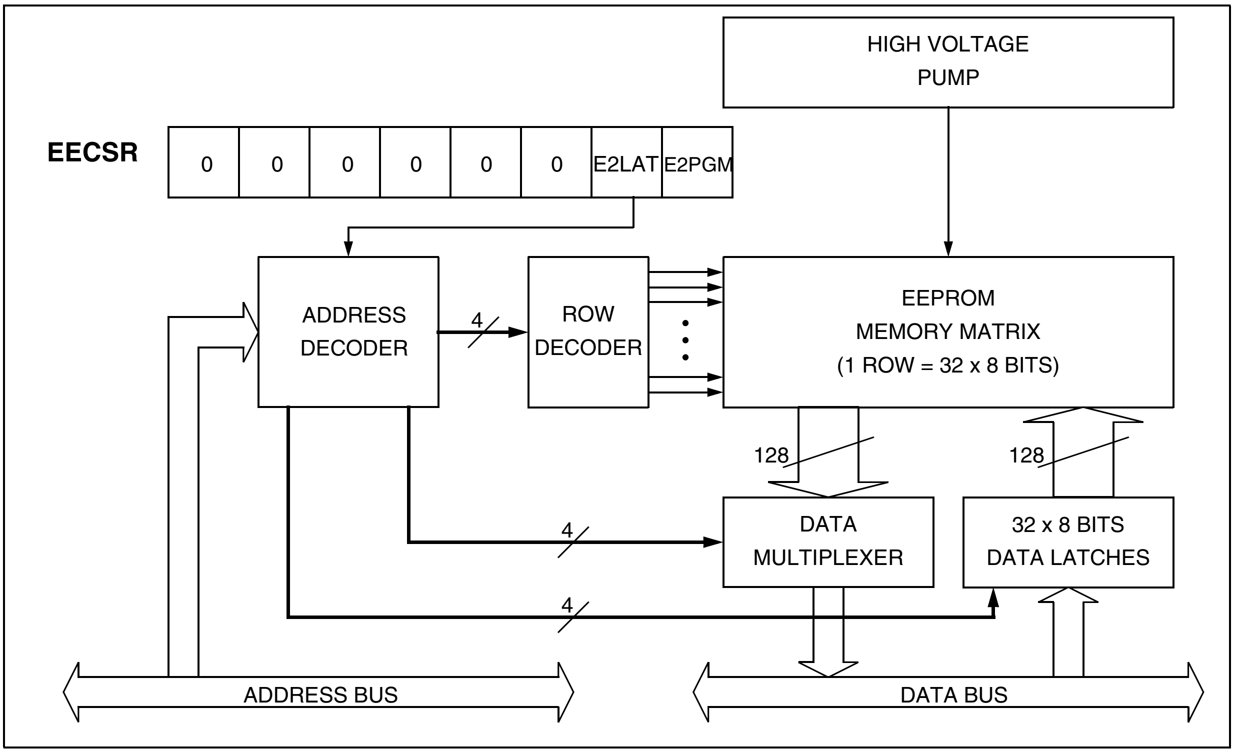
5.1 INTRODUCTION

The Electrically Erasable Programmable Read Only Memory can be used as a non volatile backup 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 5. EEPROM Block Diagram



DATA EEPROM (Cont'd)

5.3 MEMORY ACCESS

The Data EEPROM memory read/write access modes are controlled by the E2LAT bit of the EEPROM Control/Status register (EECSR). The flowchart in Figure 6 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. In a read cycle, the byte to be accessed is put on the data bus in less than 1 CPU clock cycle. This means that reading data from EEPROM takes the same time as reading data from EPROM, but this memory cannot be used to execute machine code.

Write Operation (E2LAT=1)

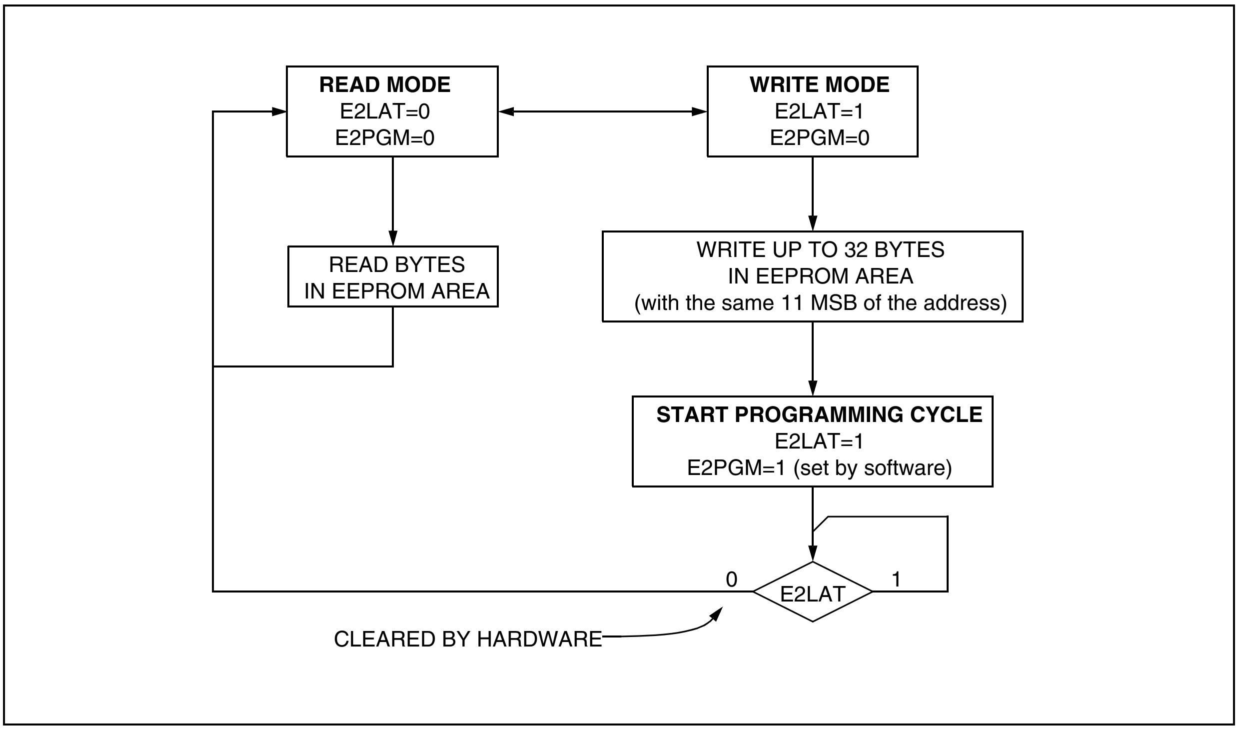
To access the write mode, the E2LAT bit has to be set by software (the E2PGM bit remains cleared). When a write access to the EEPROM area occurs, the value is latched inside the 32 data latches according to its address.

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

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

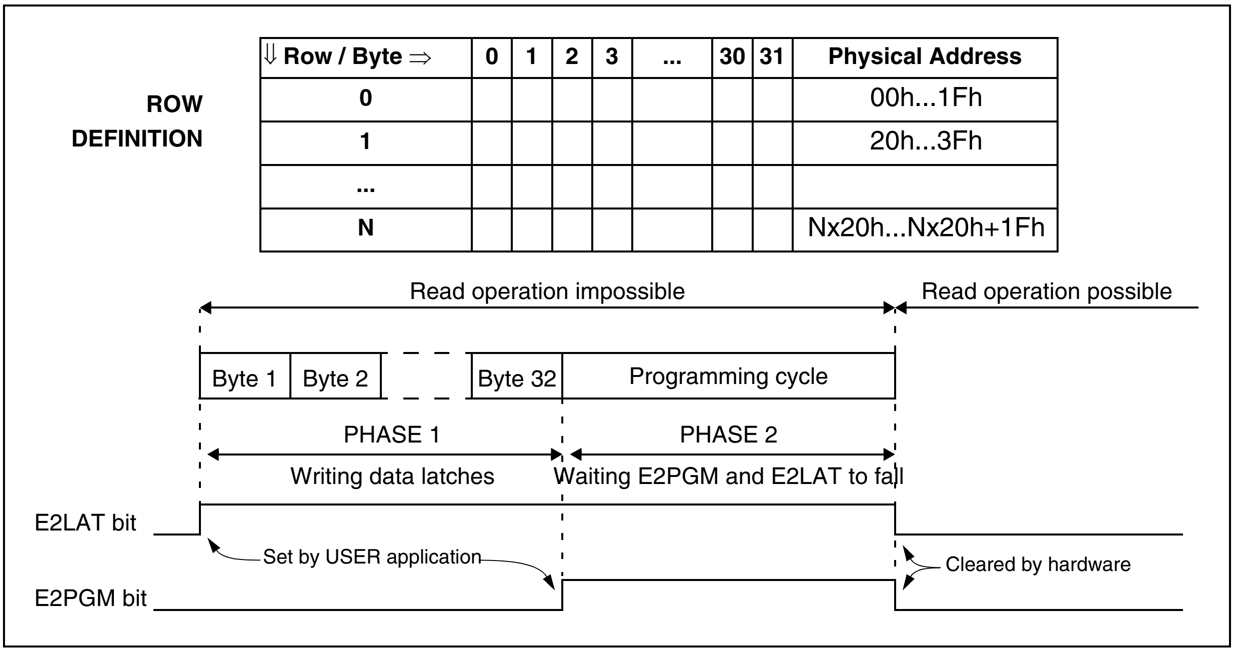
Note: Care should be taken during the programming cycle. Writing to the same memory location will over-program the memory (logical AND between the two write access data result) because the data latches are only cleared at the end of the programming cycle and by the falling edge of the E2LAT bit.
It is not possible to read the latched data.
This note is ilustrated by the Figure 8.

Figure 6. Data EEPROM Programming Flowchart



DATA EEPROM (Cont'd)

Figure 7. Data E^2PROM Write Operation



⇓ Row / Byte ⇒	0	1	2	3	...	30	31	Physical Address
0								00h...1Fh
1								20h...3Fh
...								
N								Nx20h...Nx20h+1Fh

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

DATA EEPROM (Cont'd)

5.4 POWER SAVING MODES

Wait mode

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

Active-Halt mode

Refer to Wait mode.

Halt mode

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

5.5 ACCESS ERROR HANDLING

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

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

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

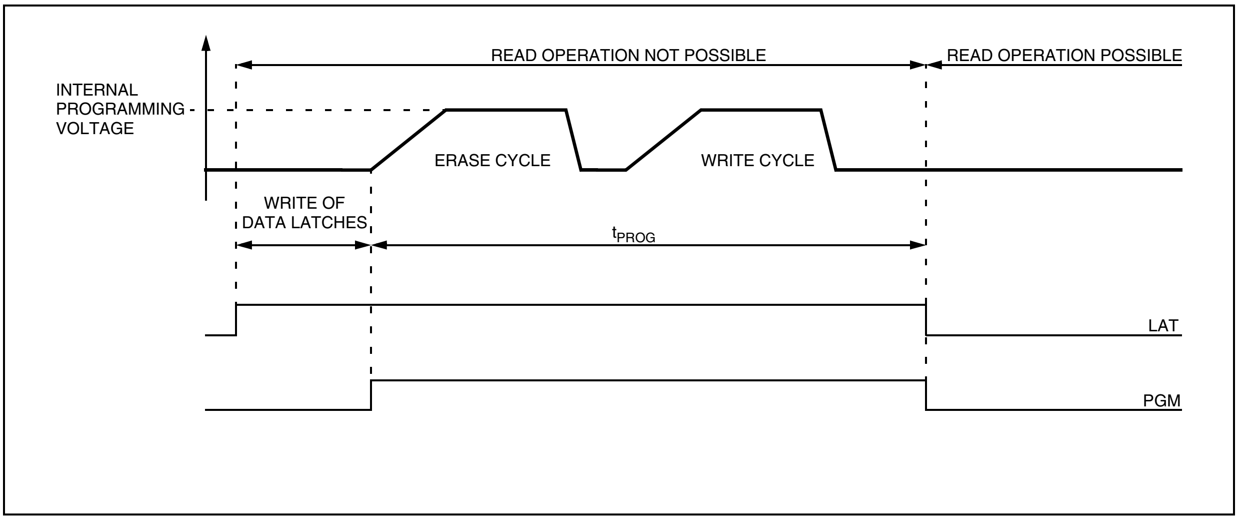
5.6 Data EEPROM Read-out Protection

The read-out protection is enabled through an option bit (see section 15.1 on page 158).

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

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

Figure 8. Data EEPROM Programming Cycle



DATA EEPROM (Cont'd)

5.7 REGISTER DESCRIPTION

EEPROM CONTROL/STATUS REGISTER (EECSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	E2LAT	E2PGM

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

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

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

Note: if the E2PGM bit is cleared during the programming cycle, the memory data is not guaranteed

Table 3. 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 6 CPU registers shown in Figure 9 are not present in the memory mapping and are accessed by specific instructions.

Accumulator (A)

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

Index Registers (X and Y)

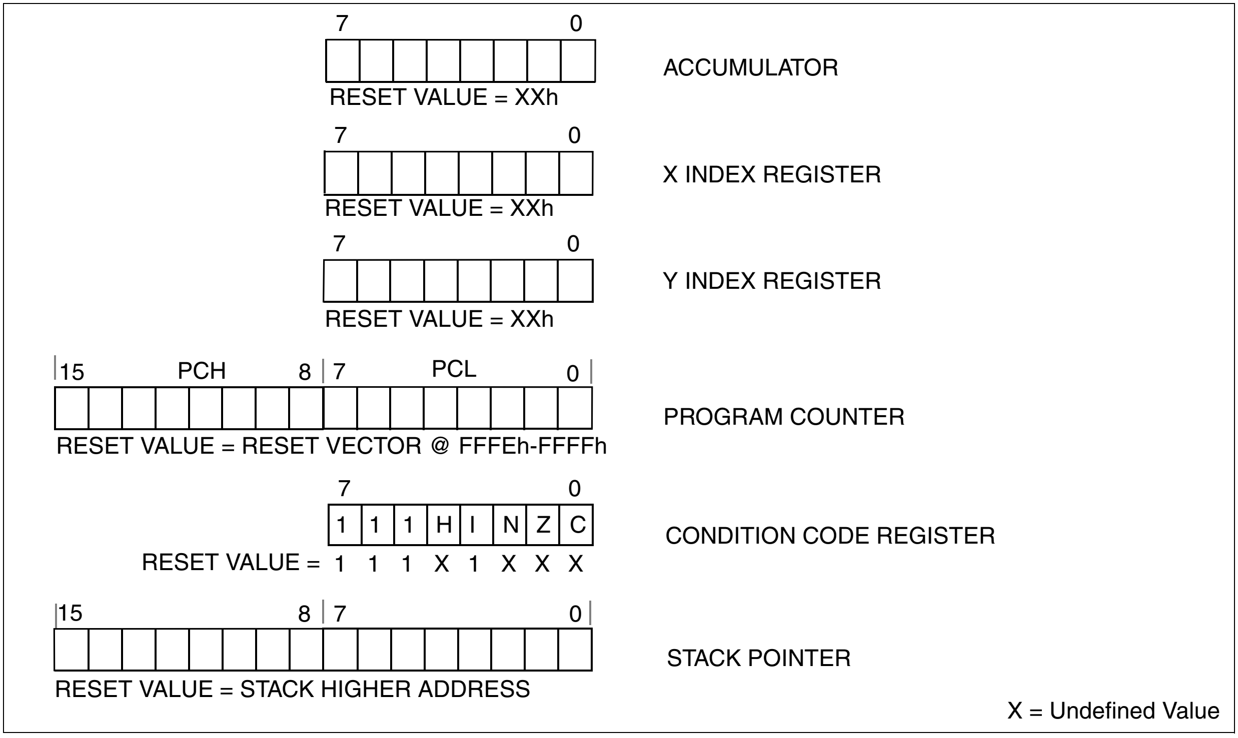
In indexed addressing modes, these 8-bit registers are used to create either effective addresses or temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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

Program Counter (PC)

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

Figure 9. CPU Registers



CPU REGISTERS (Cont'd)

CONDITION CODE REGISTER (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	1	H	I	N	Z	C

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

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

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

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

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

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

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

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

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

Bit 2 = **N** *Negative*.

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

This bit is accessed by the JRMI and JRPL instructions.

Bit 1 = **Z** *Zero*.

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

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

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

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

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

CPU REGISTERS (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 01FFh

15							8
0	0	0	0	0	0	0	1

7							0
1	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see Figure 10).

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.

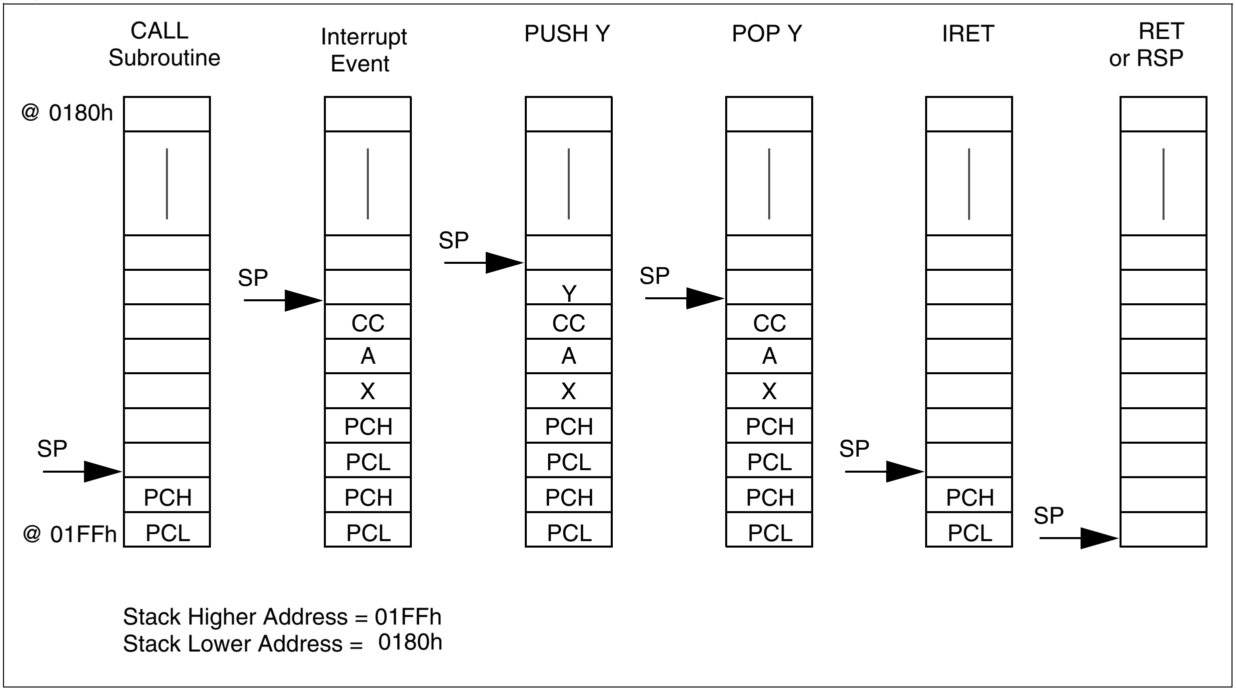
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 10.

– When an interrupt is received, the SP is decremented and the context is pushed on the stack.

– On return from interrupt, the SP is incremented and the context is popped from the stack.

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

Figure 10. Stack Manipulation Example



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 32kHz External crystal/ceramic resonator (selected by option byte)
  - External Clock Input (enabled by option byte)
  - PLL for multiplying the frequency by 8 or 4 (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 an accuracy of 1% for a given device, temperature and voltage range (4.5V-5.5V). It must be calibrated to obtain the frequency required in the application. This is done by software writing a 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), i.e. each time the device is reset, the calibration value must be loaded in the RCCR. Predefined calibration values are stored in EEPROM for 3V and 5V V_{DD} supply voltages at 25°C, as shown in the following table.

RCCR	Conditions	ST7LITE3 Addresses
RCCRH0	V_{DD}=5V T_A=25°C f_{RC}=1MHz	DEE0h [1] (CR[9:2] bits)
RCCRL0		DEE1h [1] (CR[1:0] bits)
RCCRH1	V_{DD}=3.3V T_A=25°C f_{RC}=1MHz	DEE2h [1] (CR[9:2] bits)
RCCRL1		DEE3h [1] (CR[1:0] bits)

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 read out 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 5th and 6th position of DEE1 and DEE3 addresses.

Note:

- See “ELECTRICAL CHARACTERISTICS” on page 129. for more information on the frequency and accuracy of the RC oscillator.
- To improve clock stability, it is recommended to place a decoupling capacitor between the V_{DD} and V_{SS} pins.
- These bytes are systematically programmed by ST, including on FASTROM devices. Consequently, customers intending to use FASTROM service must not use these bytes.
- RCCR0 and RCCR1 calibration values will not be erased if the read-out protection bit is reset after it has been set . See “Read out Protection” on page 13.

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 1MHz frequency from the RC oscillator or the external clock by 4 or 8 to obtain f_{OSC} of 4 or 8 MHz. The PLL is enabled and the multiplication factor of 4 or 8 is selected by 2 option bits.

- The x4 PLL is intended for operation with V_{DD} in the 2.7V to 3.3V range
- The x8 PLL is intended for operation with V_{DD} in the 3.3V to 5.5V range

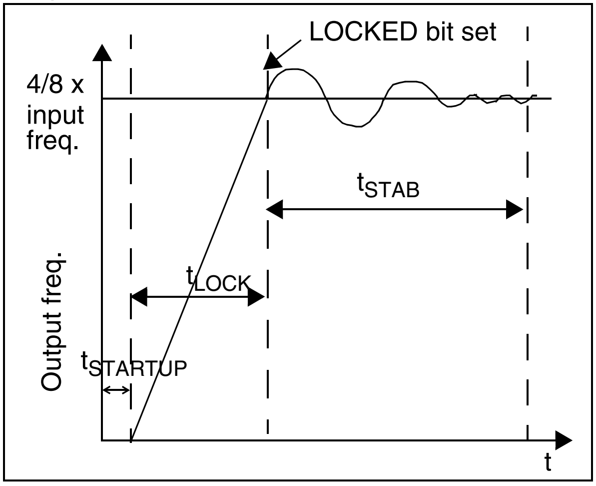
Refer to Section 15.1 for the option byte description.

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

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

PHASE LOCKED LOOP (Cont'd)

Figure 11. 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 11 and 13.3.4Internal RC Oscillator and PLL)

Refer to section 7.6.4 on page 32 for a description of the LOCKED bit in the SICSR register.

7.3 REGISTER DESCRIPTION

MAIN CLOCK CONTROL/STATUS REGISTER (MCCSR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	MCO	SMS

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

Bit 1 = **MCO** *Main Clock Out enable*
This bit is read/write by software and cleared by hardware after a reset. This bit allows to enable the MCO output clock.
0: MCO clock disabled, I/O port free for general purpose I/O.
1: MCO clock enabled.

Bit 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)

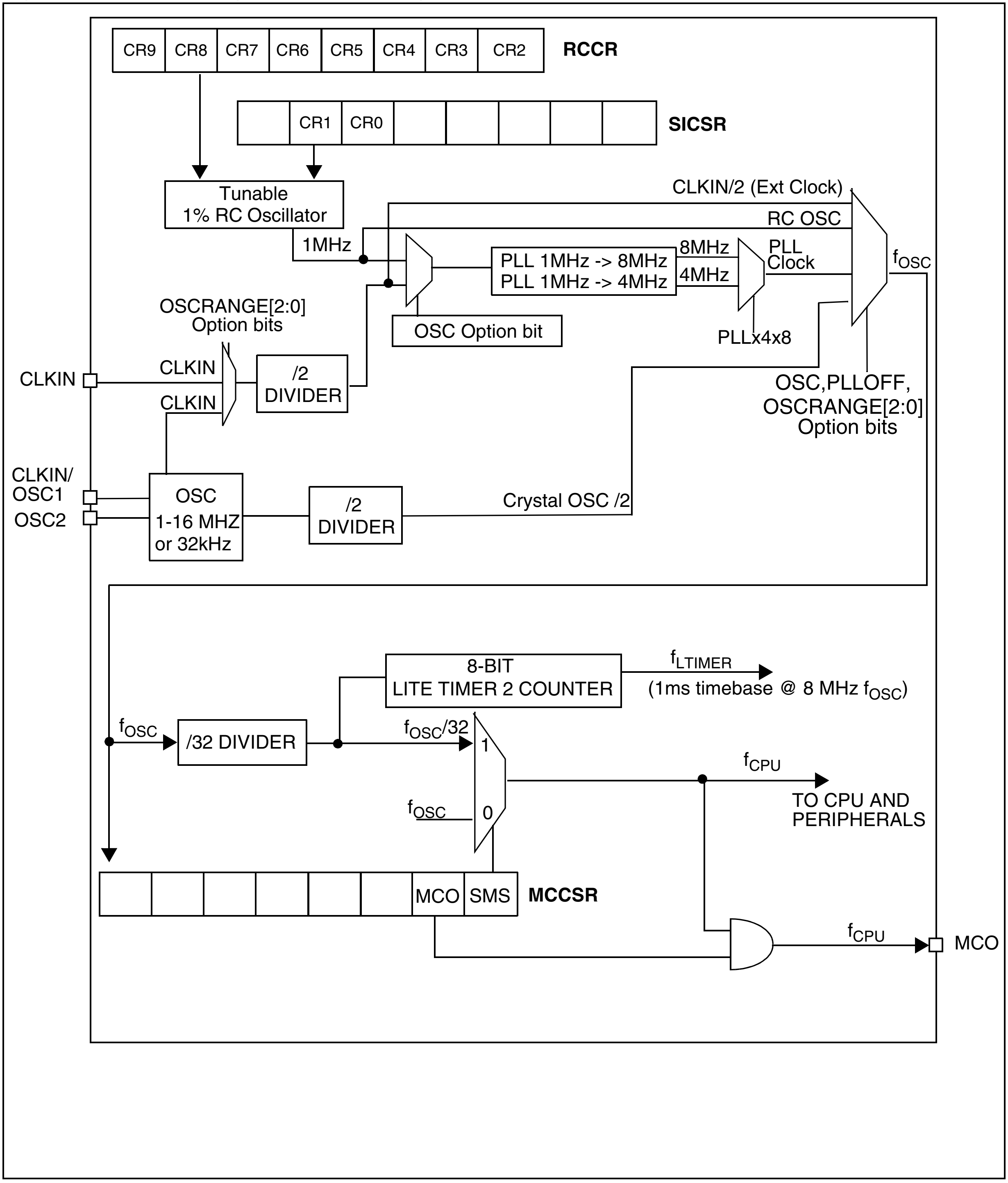
Read / Write

Reset Value: 1111 1111 (FFh)

7							0
CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2

Bits 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 start-up.
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 on page 32
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 12. 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 16MHz or 32kHz):

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

Each oscillator is optimized for a given frequency range in terms of consumption and is selectable through the option byte. The associated hardware configurations are shown in Table 4. Refer to the electrical characteristics section for more details.

External Clock Source

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

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

Crystal/Ceramic Oscillators

This family of oscillators has the advantage of producing a very accurate rate on the main clock of the ST7. The selection within a list of 4 oscillators with different frequency ranges has to be done by option byte in order to reduce consumption (refer to section 15.1 on page 158 for more details on the frequency ranges). In this mode of the multi-oscillator, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

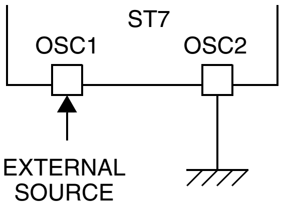
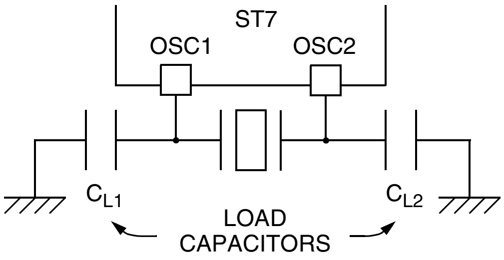
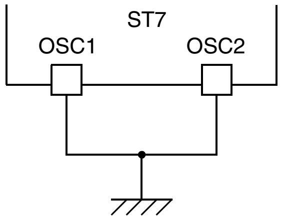
These oscillators are not stopped during the RESET phase to avoid losing time in the oscillator start-up phase.

Internal RC Oscillator

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

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

Table 4. ST7 Clock Sources

	Hardware Configuration
External Clock	
Crystal/Ceramic Resonators	
Internal RC Oscillator	

7.5 RESET SEQUENCE MANAGER (RSM)

7.5.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 14:

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

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

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

The basic RESET sequence consists of 3 phases as shown in Figure 13:

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

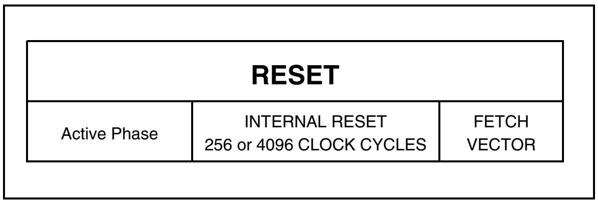
The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay is automatically selected depending on the clock source chosen by option byte:

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

The RESET vector fetch phase duration is 2 clock cycles.

If the PLL is enabled by option byte, it outputs the clock after an additional delay of $t_{STARTUP}$ (see Figure 11).

Figure 13. RESET Sequence Phases

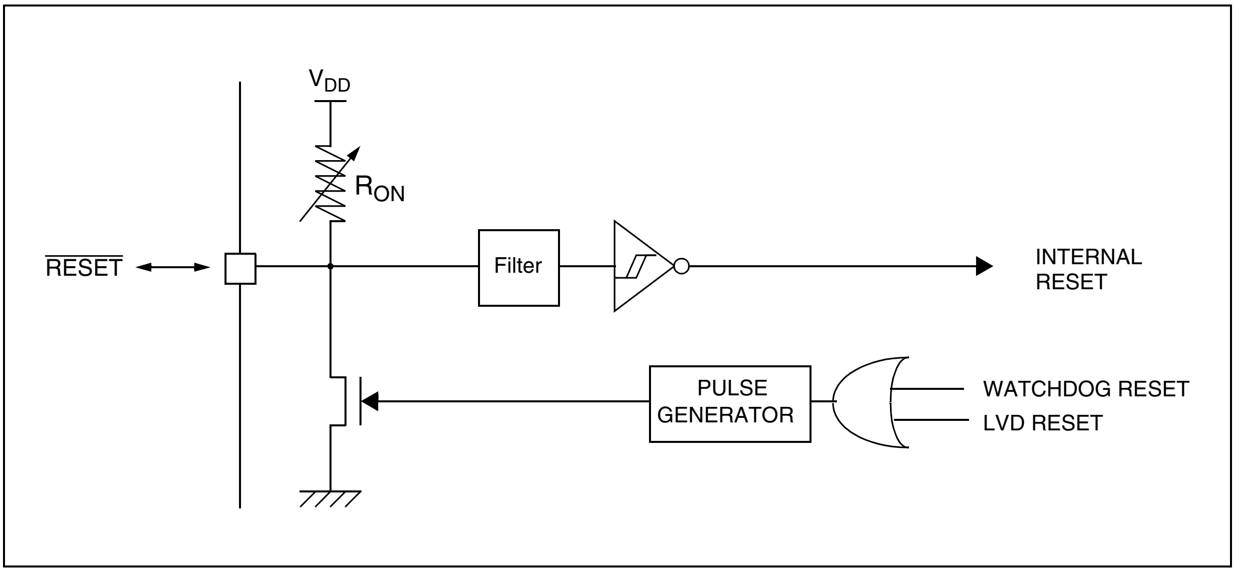


7.5.2 Asynchronous External $\overline{\text{RESET}}$ pin

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

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see Figure 15). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

Figure 14. Reset Block Diagram



RESET SEQUENCE MANAGER (Cont'd)

The RESET pin is an asynchronous signal which plays a major role in EMS performance. In a noisy environment, it is recommended to follow the guidelines mentioned in the electrical characteristics section.

7.5.3 External Power-On RESET

If the LVD is disabled by option byte, to start up the microcontroller correctly, the user must ensure by means of an external reset circuit that the reset signal is held low until V_{DD} is over the minimum level specified for the selected f_{OSC} frequency.

A proper reset signal for a slow rising V_{DD} supply can generally be provided by an external RC network connected to the 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 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 15.

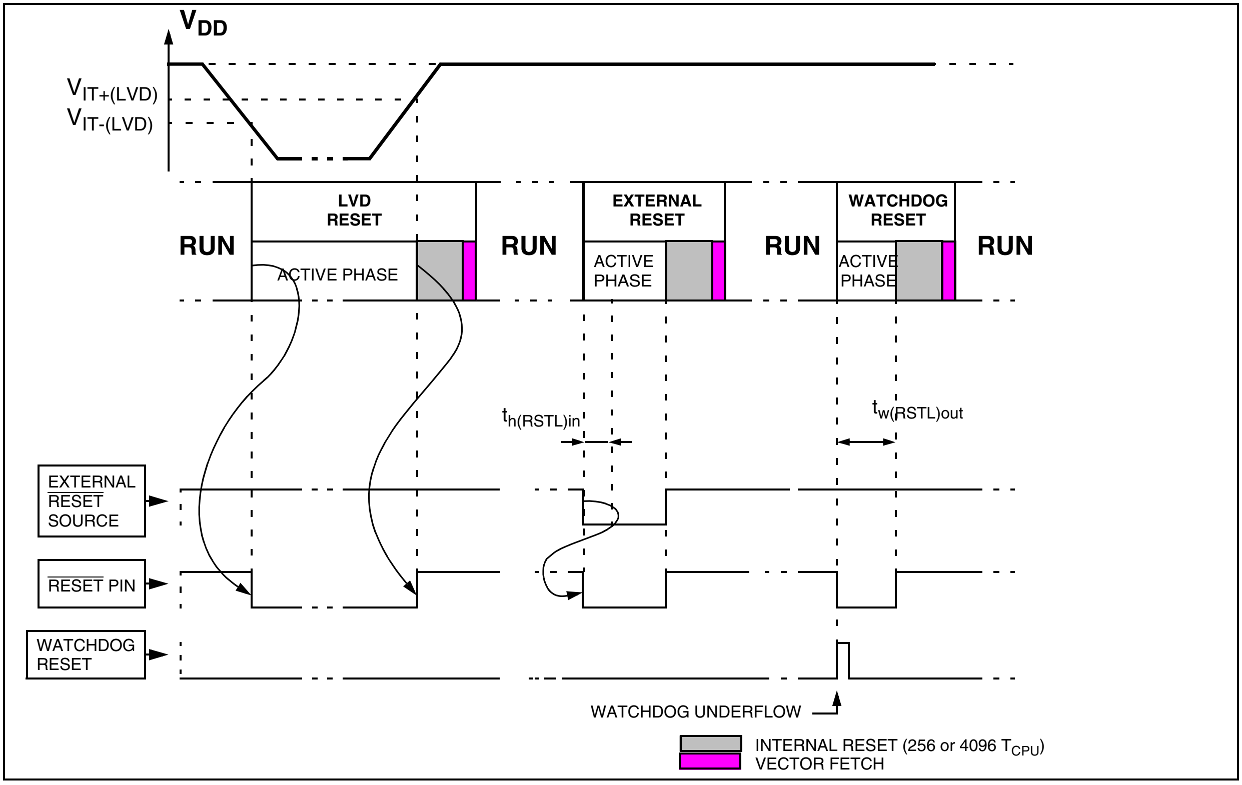
The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

7.5.5 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 15.

Starting from the Watchdog counter underflow, the device RESET pin acts as an output that is pulled low during at least $t_{w(RSTL)out}$.

Figure 15. 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 12.2.1 on page 126 for further details.

7.6.1 Low Voltage Detector (LVD)

The Low Voltage Detector function (LVD) 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 in order 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 16.

The voltage threshold can be configured by option byte to be low, medium or high.

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.

Notes:

The LVD allows the device to be used without any external RESET circuitry.

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

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 16. Low Voltage Detector vs Reset

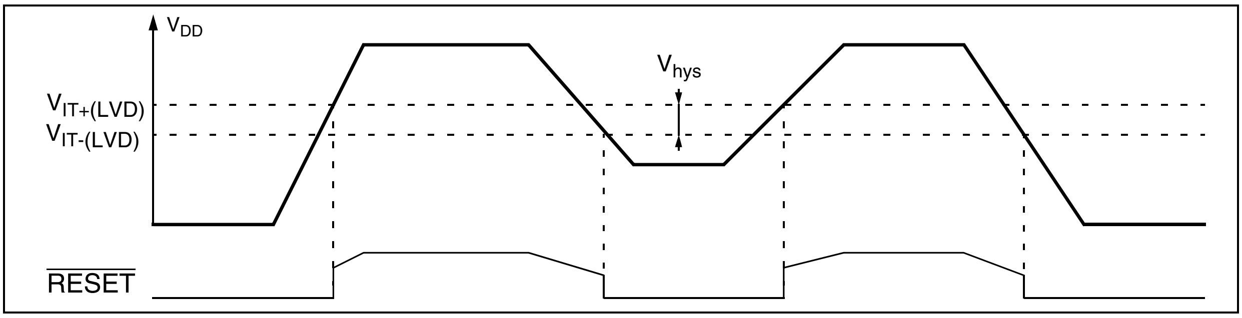
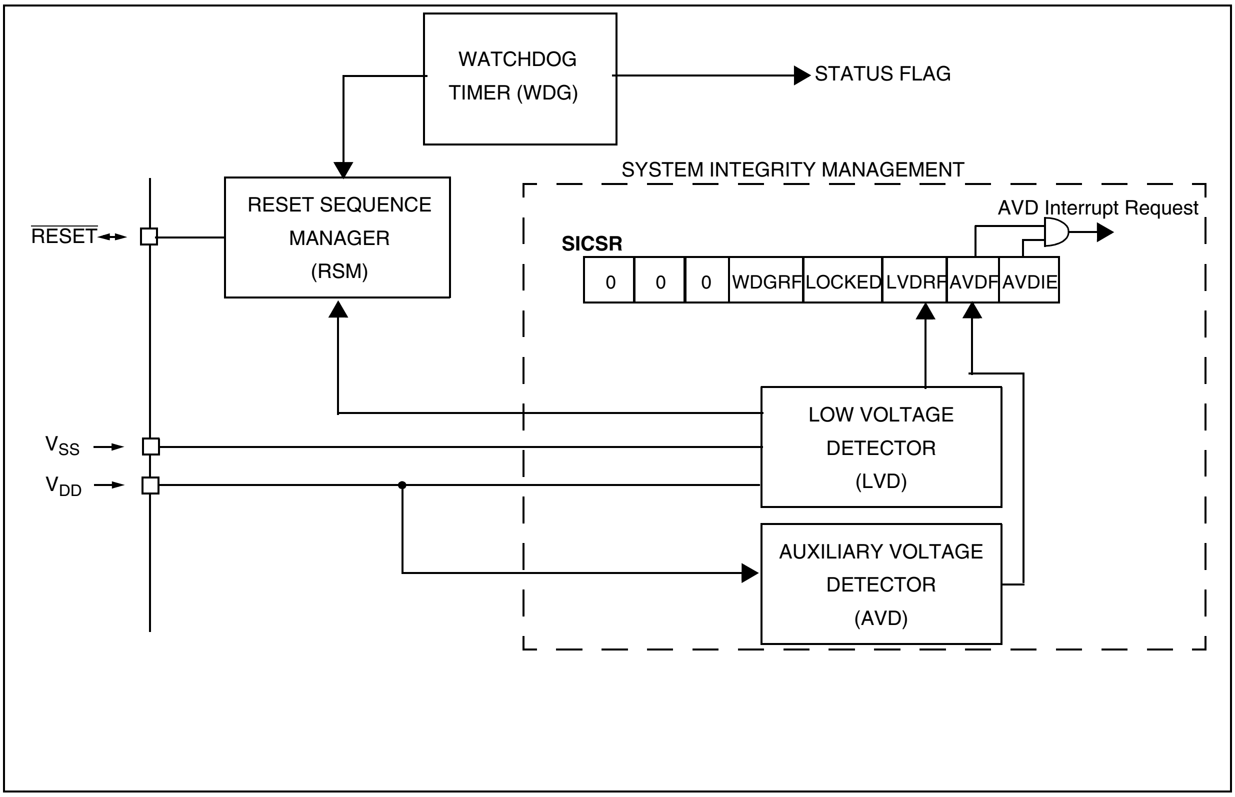


Figure 17. Reset and Supply Management Block Diagram



SYSTEM INTEGRITY MANAGEMENT (Cont'd)

7.6.2 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.

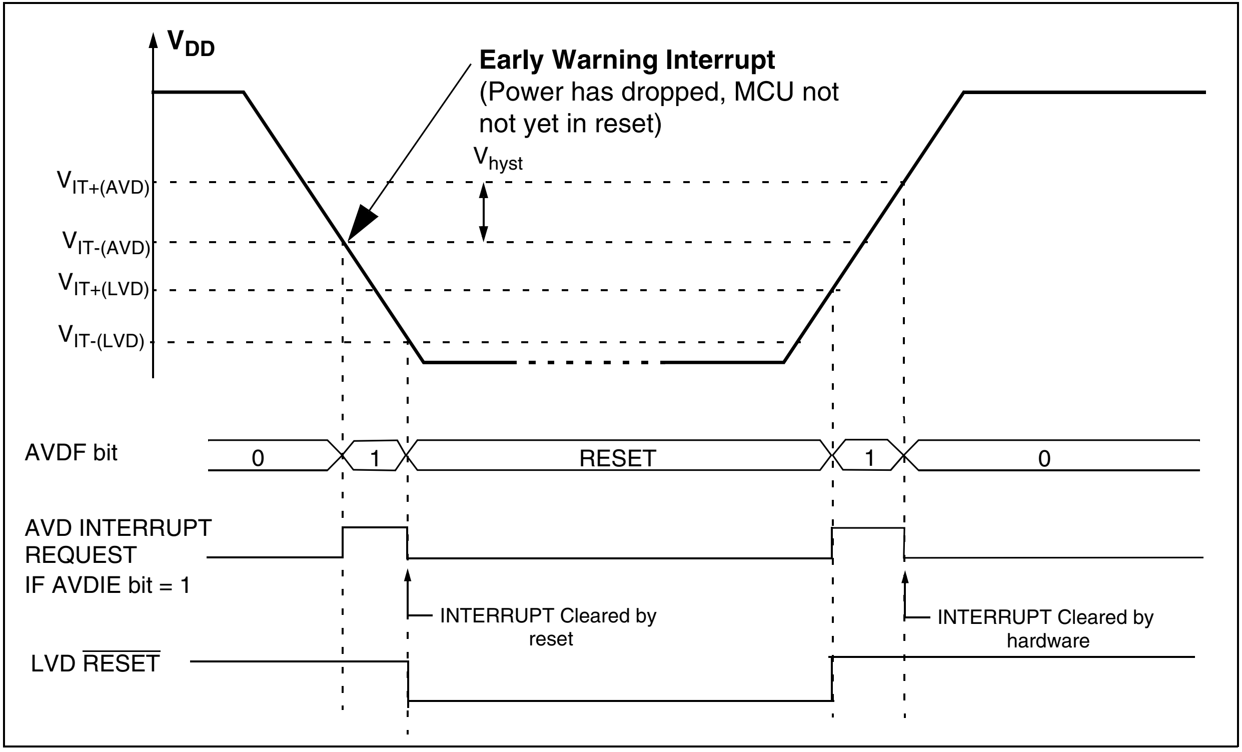
7.6.2.1 Monitoring the V_{DD} Main Supply

The AVD voltage threshold value is relative to the selected LVD threshold configured by option byte (see section 15.1 on page 158).

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 18.

Figure 18. Using the AVD to Monitor V_{DD}



SYSTEM INTEGRITY MANAGEMENT (Cont'd)

7.6.3 Low Power Modes

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.

7.6.3.1 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).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
AVD event	AVDF	AVDIE	Yes	No

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

7.6.4 Register Description

SYSTEM INTEGRITY (SI) CONTROL/STATUS REGISTER (SICSR)

Read/Write

Reset Value: 0000 0xx0 (0xh)

7							0
0	CR1	CR0	WDG RF	LOCKED	LVDRF	AVDF	AVDIE

Bit 7 = Reserved, must be kept cleared.

Bits 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 on page 23

Bit 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 by the following table.

RESET Sources	LVDRF	WDGRF
External $\overline{\text{RESET}}$ pin	0	0
Watchdog	0	1
LVD	1	X

Bit 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

Bit 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.

Bit 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 18 and to Section 7.6.2.1 for additional details.
0: V_{DD} over AVD threshold
1: V_{DD} under AVD threshold

Bit 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

Application notes

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.

8 INTERRUPTS

The ST7 core may be interrupted by one of two different methods: maskable hardware interrupts as listed in the Interrupt Mapping Table and a non-maskable software interrupt (TRAP). The Interrupt processing flowchart is shown in Figure 19.
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 the Interrupt Mapping Table 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 will be cleared and the main program will resume.

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 the Interrupt Mapping Table).

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 the Interrupt Mapping Table).

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 will be serviced according to the flowchart on Figure 19.

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.

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 I bit of the CC register is cleared.

– The corresponding enable bit is set in the control register.

If any 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 (i.e. waiting for being enabled) will therefore be lost if the clear sequence is executed.

INTERRUPTS (Cont'd)

Figure 19. Interrupt Processing Flowchart

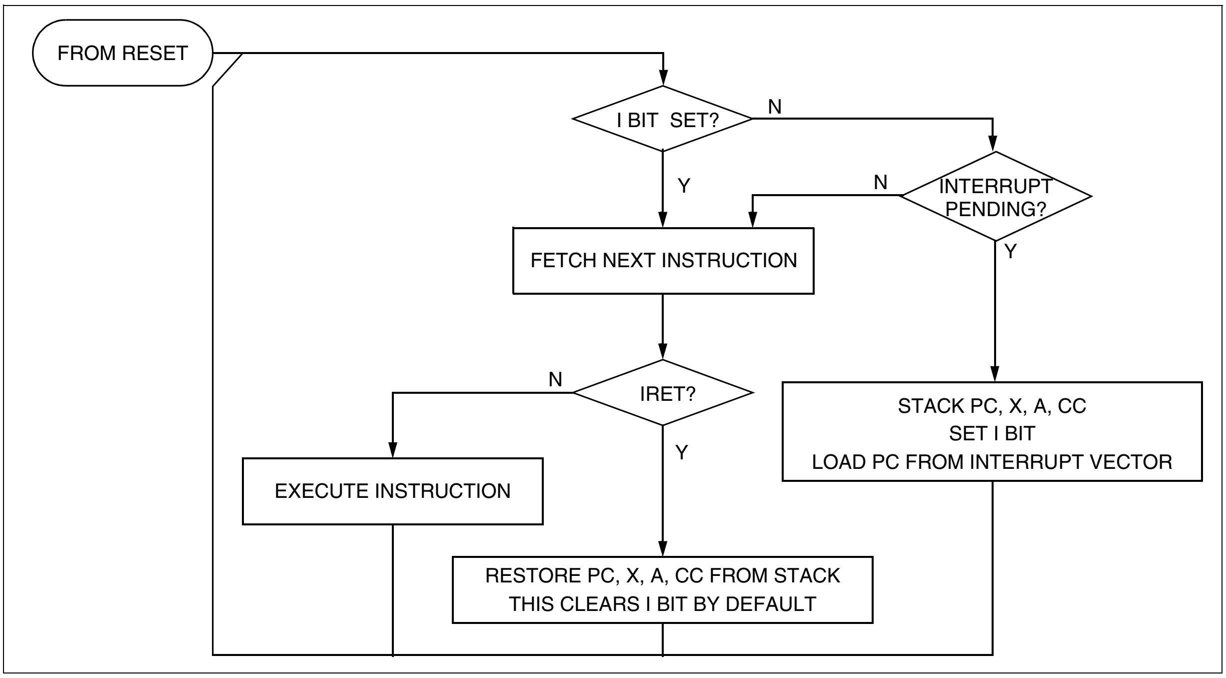


Table 5. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Address Vector
	RESET	Reset	N/A	Highest Priority	yes	FFFEh-FFFFh
	TRAP	Software Interrupt			no	FFFCh-FFFDh
0	AWU	7 Interrupt	AWUCSR		yes1)	FFFAh-FFFBh
1	ei0	External Interrupt 0	N/A		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		yes2)	FFE8h-FFE9h
10	LITE TIMER	LITE TIMER Input Capture Interrupt	LTCSR		no	FFE6h-FFE7h
11		LITE TIMER RTC1 Interrupt	LTCSR		yes2)	FFE4h-FFE5h
12	SPI	SPI Peripheral Interrupts	SPICSR	Lowest Priority	yes	FFE2h-FFE3h
13	AT TIMER	AT TIMER Overflow Interrupt 2	ATCSR2		no	FFE0h-FFE1h

Note 1: This interrupt exits the MCU from "Auto Wake-up from Halt" mode only.

Note 2: These interrupts exit the MCU from "ACTIVE-HALT" mode only.

INTERRUPTS (Cont'd)

EXTERNAL INTERRUPT CONTROL REGISTER (EICR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
IS31	IS30	IS21	IS20	IS11	IS10	IS01	IS00

Bit 7:6 = **IS3[1:0]** *ei3 sensitivity*
These bits define the interrupt sensitivity for ei3 (Port B0) according to Table 6.

Bit 5:4 = **IS2[1:0]** *ei2 sensitivity*
These bits define the interrupt sensitivity for ei2 (Port B3) according to Table 6.

Bit 3:2 = **IS1[1:0]** *ei1 sensitivity*
These bits define the interrupt sensitivity for ei1 (Port A7) according to Table 6.

Bit 1:0 = **IS0[1:0]** *ei0 sensitivity*
These bits define the interrupt sensitivity for ei0 (Port A0) according to Table 6.

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

Table 6. Interrupt Sensitivity Bits

ISx1	ISx0	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

.

EXTERNAL INTERRUPT SELECTION REGISTER (EISR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
ei31	ei30	ei21	ei20	ei11	ei10	ei01	ei00

Bit 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 according to the table below.

External Interrupt I/O pin selection

ei31	ei30	I/O Pin
0	0	No interrupt *
0	1	PB0
1	0	PB1
1	1	PB2

* Reset State

Bit 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 according to the table below.

External Interrupt I/O pin selection

ei21	ei20	I/O Pin
0	0	No interrupt *
0	1	PB3
1	0	PB5
1	1	PB6

* Reset State

INTERRUPTS (Cont'd)

Bit 3:2 = **ei1[1:0]** *ei1 pin selection*
These bits are written by software. They select the Port A I/O pin used for the ei1 external interrupt according to the table below.

External Interrupt I/O pin selection

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

* Reset State

Bit 1:0 = **ei0[1:0]** *ei0 pin selection*
These bits are written by software. They select the Port A I/O pin used for the ei0 external interrupt according to the table below.

External Interrupt I/O pin selection

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

* Reset State

Bits 1:0 = Reserved.

9 POWER SAVING MODES

9.1 INTRODUCTION

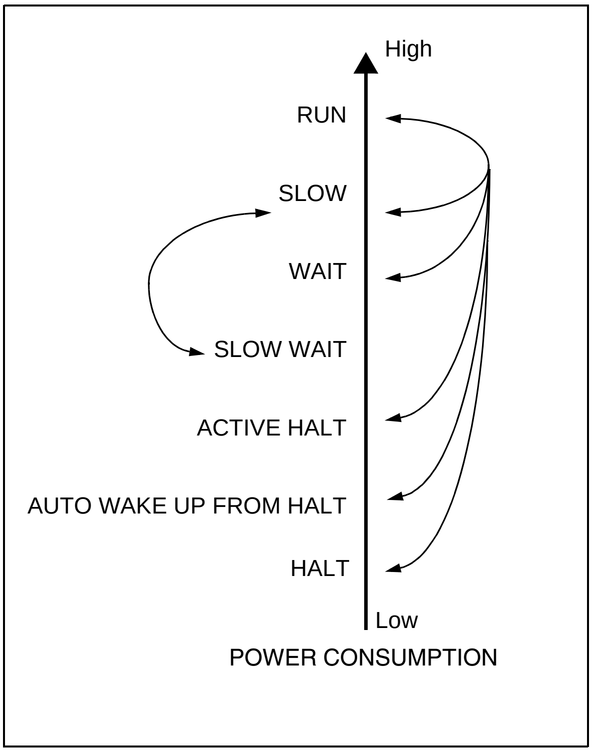
To give a large measure of flexibility to the application in terms of power consumption, five main power saving modes are implemented in the ST7 (see Figure 20):

- Slow
- Wait (and Slow-Wait)
- Active Halt
- Auto Wake up 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 may be selected by setting the relevant register bits or by calling the specific ST7 software instruction whose action depends on the oscillator status.

Figure 20. Power Saving Mode Transitions



9.2 SLOW MODE

This mode has two targets:

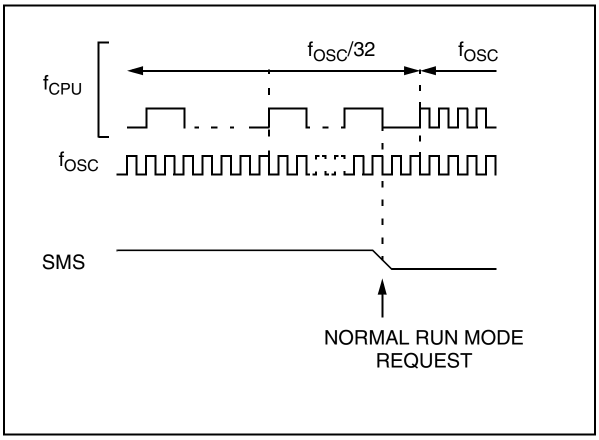
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

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 21. SLOW Mode Clock Transition



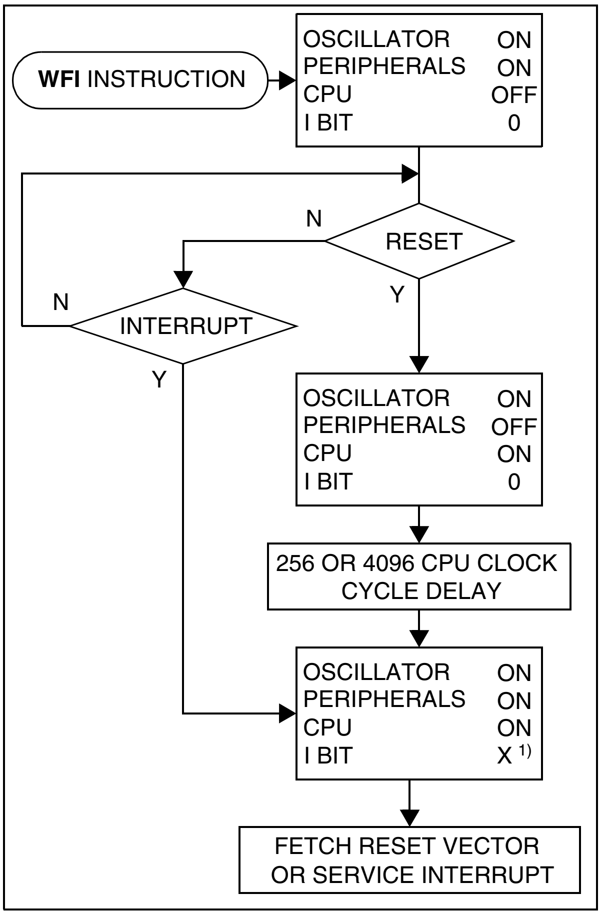
POWER SAVING MODES (Cont'd)

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 an interrupt or RESET occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine.
The MCU will remain in WAIT mode until a Reset or an Interrupt occurs, causing it to wake up.

Refer to Figure 22.

Figure 22. WAIT Mode Flow-chart



Note:

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.

POWER SAVING MODES (Cont'd)

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 on page 40 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 5, "Interrupt Mapping," on page 34) or a RESET. When exiting HALT mode by means of a RESET or an interrupt, the oscillator is immediately turned on and the 256 CPU cycle delay is used to stabilize the oscillator. After the start up delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 24).

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 causing all internal processing to be stopped, including the operation of the on-chip peripherals. All peripherals are not clocked except the ones which get 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 15.1 on page 158 for more details).

Figure 23. HALT Timing Overview

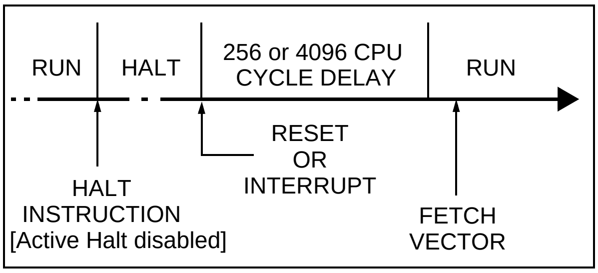
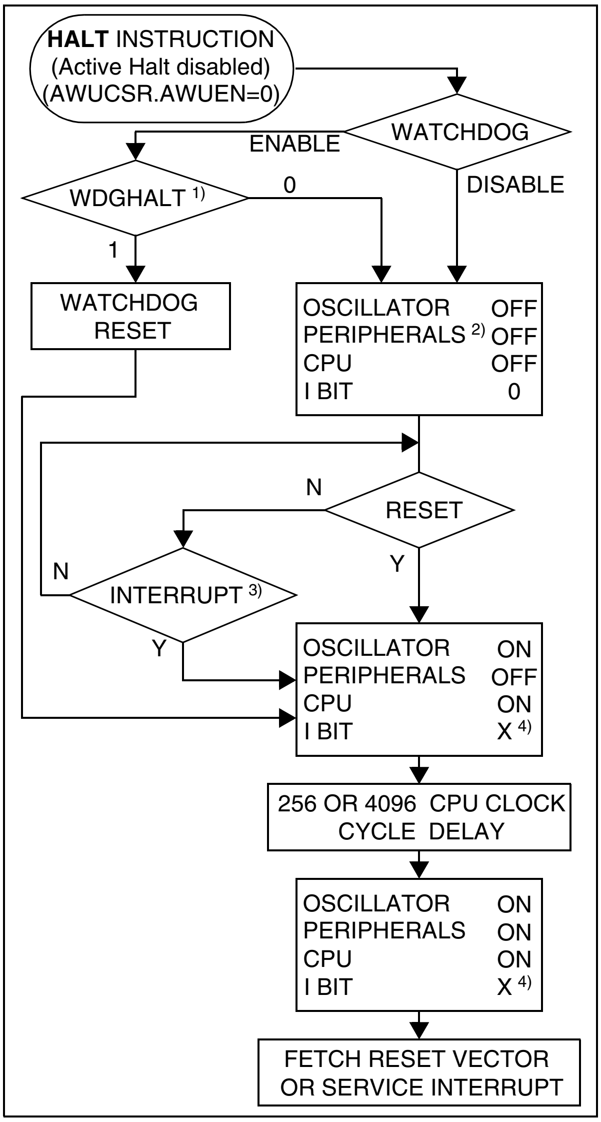


Figure 24. HALT Mode Flow-chart



Notes:

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

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

3. Only some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to Table 5, "Interrupt Mapping," on page 34 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.

POWER SAVING MODES (Cont'd)

9.4.0.1 Halt Mode Recommendations

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

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

– For the same reason, reinitialize the level sensitiveness of each external interrupt as a precautionary measure.

– The opcode for the HALT instruction is 0x8E. To avoid an unexpected HALT instruction due to a program counter failure, it is advised to clear all occurrences of the data value 0x8E from memory. For example, avoid defining a constant in 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 wake-up 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:.

LTCSR1 TB1IE bit	ATCSR OVFIE1 bit	ATCSR CK1 bit	ATCSR CK0 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 can exit ACTIVE-HALT mode on reception of a specific interrupt (see Table 5, "Interrupt Mapping," on page 34) or a RESET.

– When exiting ACTIVE-HALT mode by means of a RESET, a 256 CPU cycle delay occurs. After the start up delay, the CPU resumes operation by fetching the reset vector which woke it up (see Figure 26).

– 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 26).

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 Note 3).

In ACTIVE-HALT mode, only the main oscillator and the selected timer counter (LT/AT) are running to keep a wake-up time base. All other peripherals are not clocked except those which get 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 spend more than a defined delay in this power saving mode.

POWER SAVING MODES (Cont'd)

Figure 25. ACTIVE-HALT Timing Overview

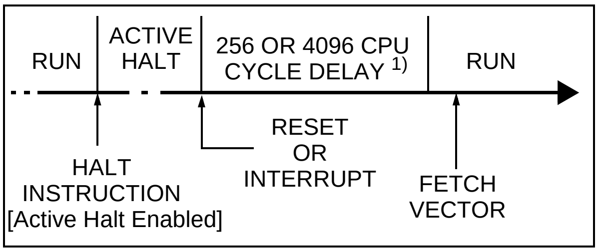
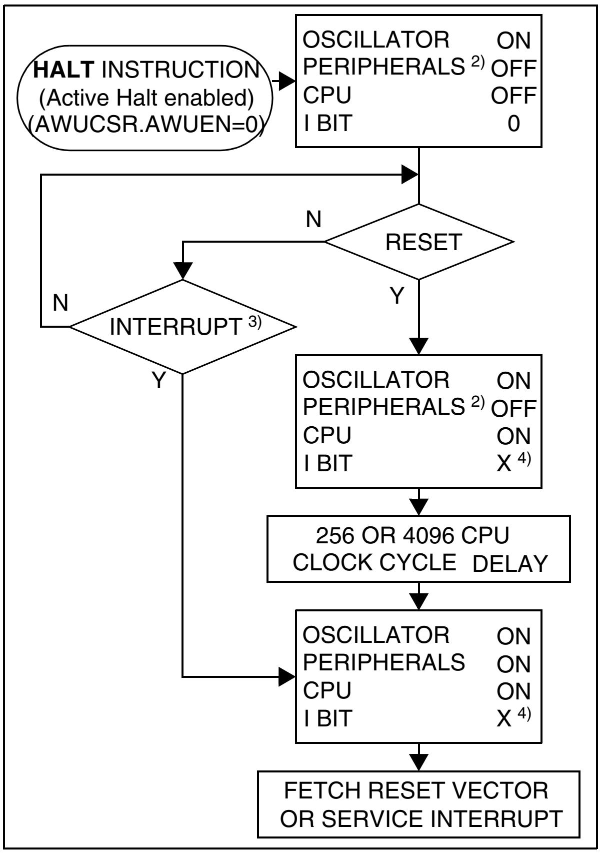


Figure 26. ACTIVE-HALT Mode Flow-chart



Notes:

1. This delay occurs only if the MCU exits ACTIVE-HALT mode by means of a RESET.

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

3. Only the RTC1 interrupt and some specific interrupts can exit the MCU from ACTIVE-HALT mode. Refer to Table 5, “Interrupt Mapping,” on page 34 for more details.

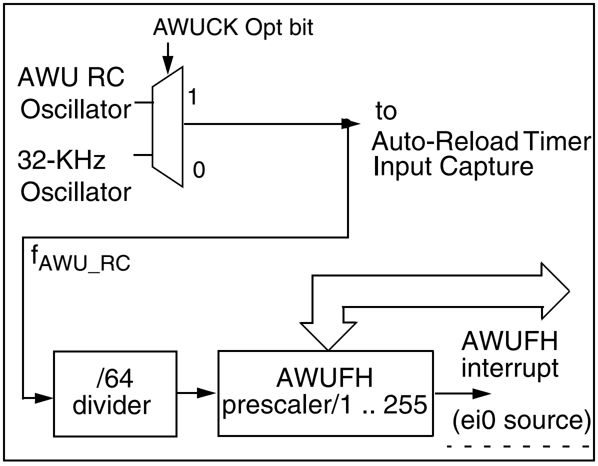
4. Before servicing an interrupt, the CC register is pushed on the stack. The I bit of the CC register is set during the interrupt routine and cleared when the CC register is popped.

9.6 AUTO WAKE UP FROM HALT MODE

Auto Wake Up From Halt (AWUFH) mode is similar to Halt mode with the additional of an internal RC oscillator for wake-up. Compared to ACTIVE-HALT mode, AWUFH has lower power consumption (the main clock is not kept running), but there is no accurate realtime clock available.

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

Figure 27. 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 start-up 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 Auto-Reload timer, allowing the f_{AWU_RC} to be measured using the main oscillator clock as a reference timebase.

POWER SAVING MODES (Cont'd)

Similarities with Halt mode

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

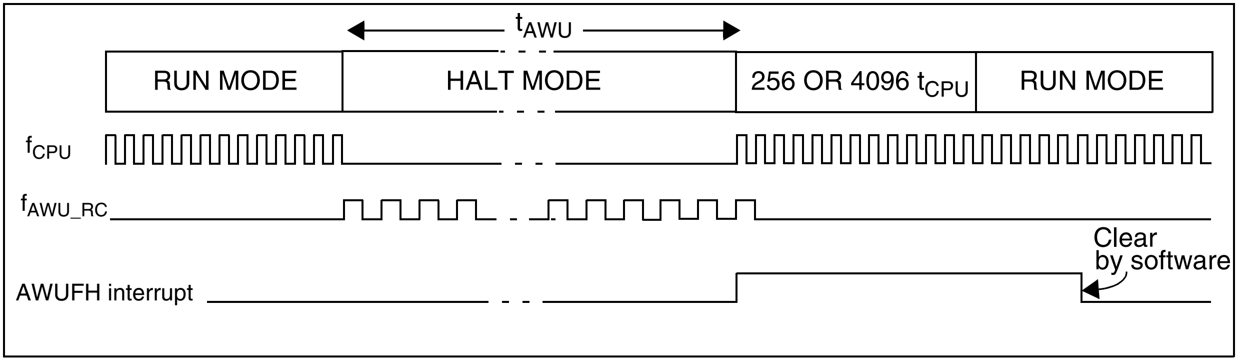
– The MCU can exit AWUFH mode by means of any interrupt with exit from Halt capability or a reset (see Section 9.4 HALT MODE).

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

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

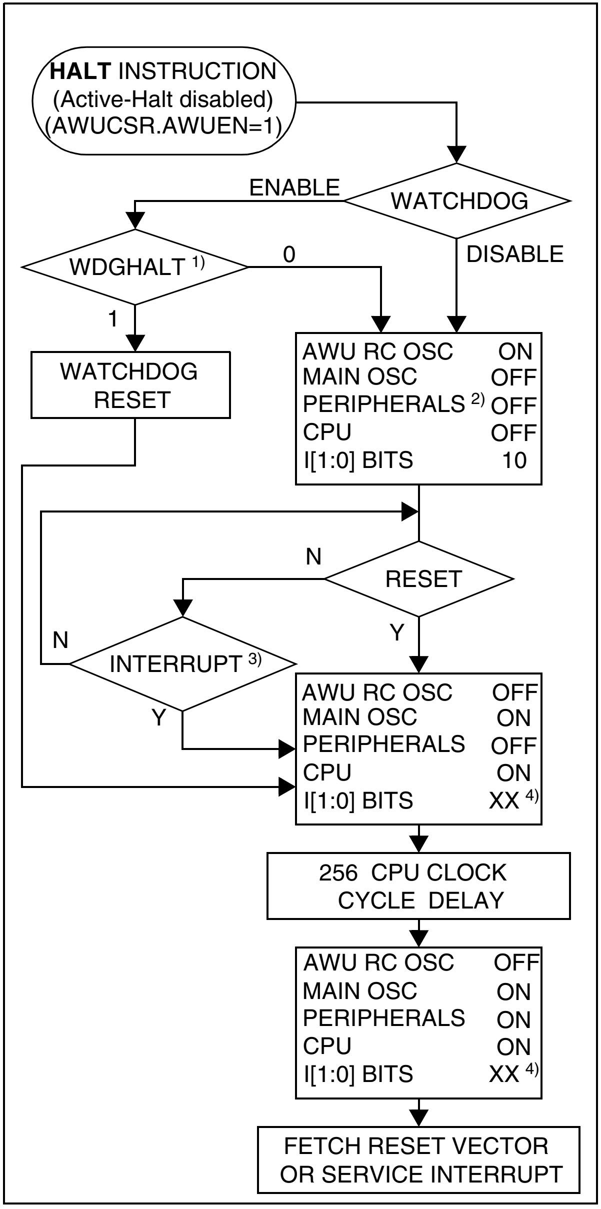
– The compatibility of Watchdog operation with AWUFH mode is configured by the WDGHALT option bit in the option byte. Depending on this setting, the HALT instruction when executed while the Watchdog system is enabled, can generate a Watchdog RESET.

Figure 28. AWUF Halt Timing Diagram



POWER SAVING MODES (Cont'd)

Figure 29. AWUFH Mode Flow-chart



Notes:

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

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

3. Only an AWUFH interrupt and some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to Table 5, "Interrupt Mapping," on page 34 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.

POWER SAVING MODES (Cont'd)

9.6.0.1 Register Description

AWUFH CONTROL/STATUS REGISTER (AWUCSR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	AWUF	AWUM	AWUEN

Bits 7:3 = Reserved.

Bit 1= **AWUF** *Auto Wake Up 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

Bit 2= **AWUM** *Auto Wake Up Measurement*
This bit enables the AWU RC oscillator and connects its output to the input capture of the 12-bit auto-reload timer. This allows the timer to be used 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

Bit 0 = **AWUEN** *Auto Wake Up From Halt Enabled*
This bit enables the Auto Wake Up 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 Wake Up From Halt) mode disabled
1: AWUFH (Auto Wake Up From Halt) mode enabled

AWUFH PRESCALER REGISTER (AWUPR)
Read/Write
Reset Value: 1111 1111 (FFh)

7							0
AWU PR7	AWU PR6	AWU PR5	AWU PR4	AWU PR3	AWU PR2	AWU PR1	AWU PR0

Bits 7:0= **AWUPR[7:0]** *Auto Wake Up Prescaler*
These 8 bits define the AWUPR Dividing factor (as explained below

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 28 on page 42) 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 7. 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 can contain up to 8 pins. Each pin can be programmed independently either as a digital input or digital output. In addition, specific pins may have several other functions. These functions can include external interrupt, alternate signal input/output for on-chip peripherals or analog input.

10.2 FUNCTIONAL DESCRIPTION

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

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

Figure 30 shows the generic I/O block diagram.

10.2.1 Input Modes

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

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

Notes:

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

External Interrupt Function

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

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

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

10.2.2 Output Modes

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

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

DR Value and Output Pin Status

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

10.2.3 Alternate Functions

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

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

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

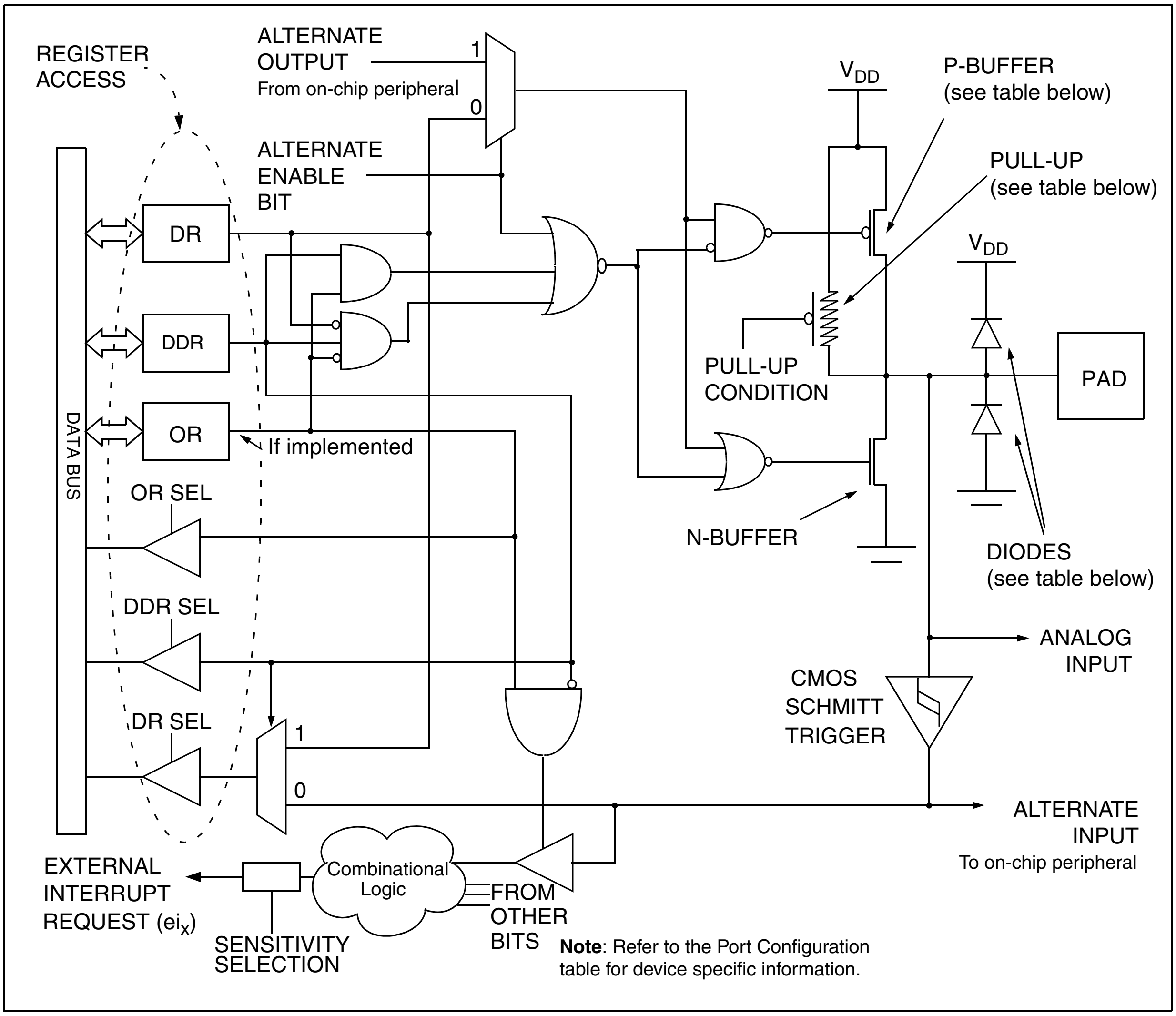
Configure an I/O as input floating for an on-chip peripheral signal which can be input and output.

Caution:

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

I/O PORTS (Cont'd)

Figure 30. I/O Port General Block Diagram



Note: Refer to the Port Configuration table for device specific information.

Table 8. I/O Port Mode Options

Configuration Mode		Pull-Up	P-Buffer	Diodes	
				to V_{DD}	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 (see note 1)	

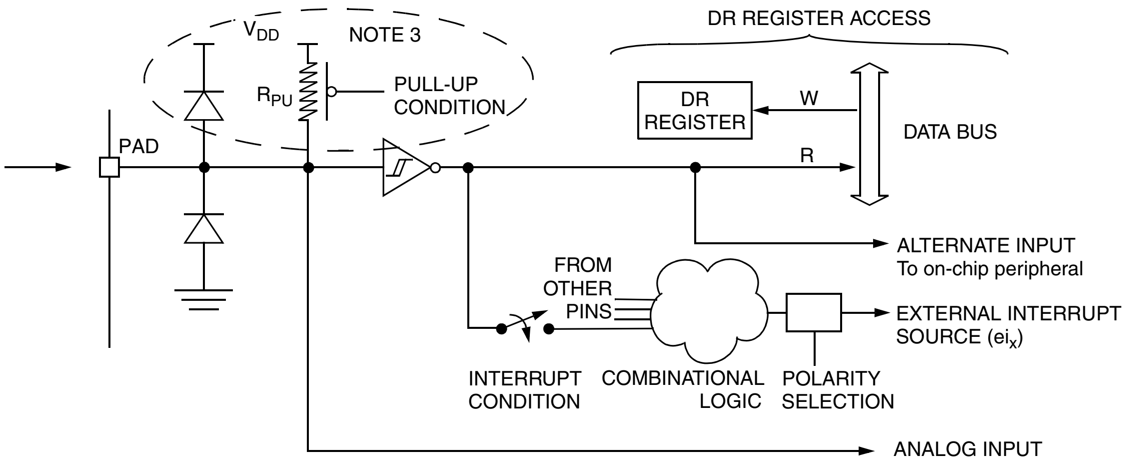
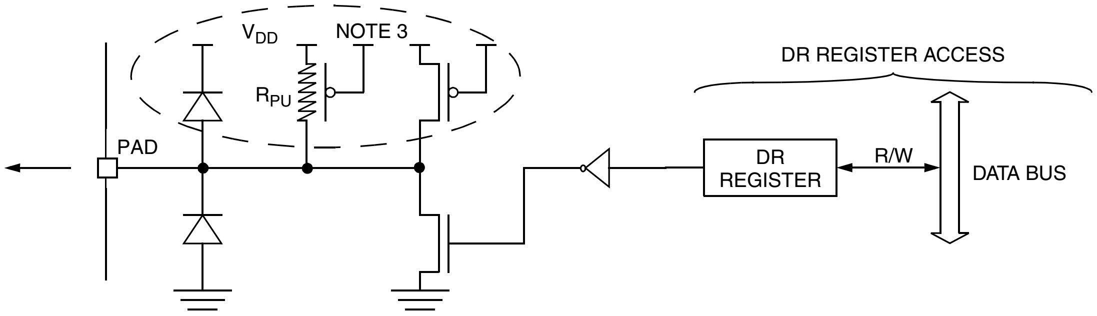
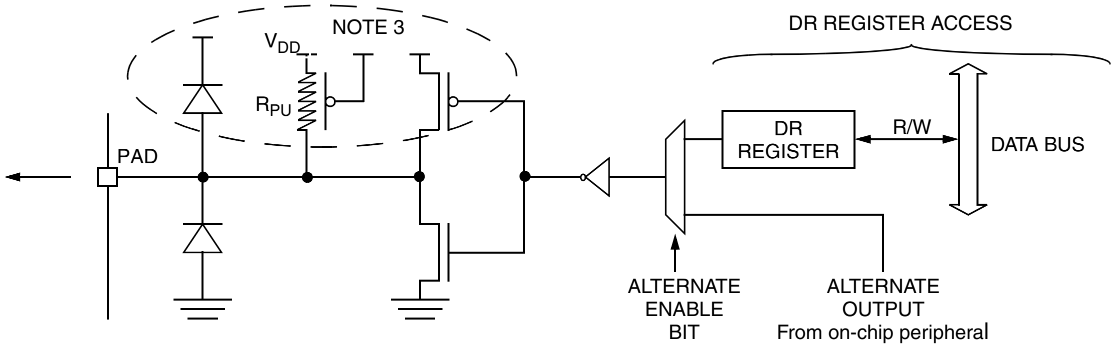
Legend: NI - not implemented
Off - implemented not activated
On - implemented and activated

Note 1: 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.

Note 2: For further details on port configuration, please refer to Table 10 and Table 11 on page 49.

I/O PORTS (Cont'd)

Table 9. I/O Configurations

	Hardware Configuration
INPUT 1)	
OPEN-DRAIN OUTPUT 2)	
PUSH-PULL OUTPUT 2)	

Notes:

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

I/O PORTS (Cont'd)

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.

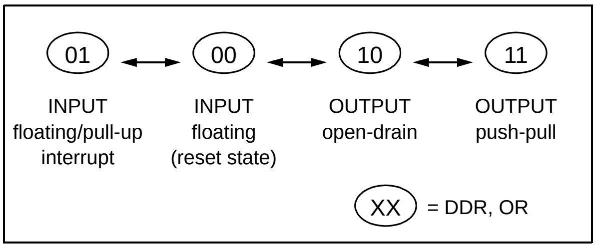
WARNING: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

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

Figure 31. 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.

10.5 LOW POWER MODES

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

10.6 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).

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

AN 970: SPI Communication between ST7 and EEPROM

AN1045: S/W implementation of I2C bus master

AN1048: Software LCD driver

I/O PORTS (Cont'd)

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

Standard Ports

PA7:0, PB6:0

MODE	DDR	OR
floating input	0	0
pull-up input	0	1
open drain output	1	0
push-pull output	1	1

Interrupt Ports

Ports where the external interrupt capability is selected using the EISR register

MODE	DDR	OR
floating input	0	0
pull-up interrupt input	0	1

Table 10. 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	floating	pull-up	open drain	push-pull

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

Table 11. 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	floating	pull-up

Table 12. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0000h	**PADR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	1	1
0001h	**PADDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	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 on expiry of a programmed time period, unless the program refreshes the counter's contents before the T6 bit becomes cleared.

11.1.2 Main Features

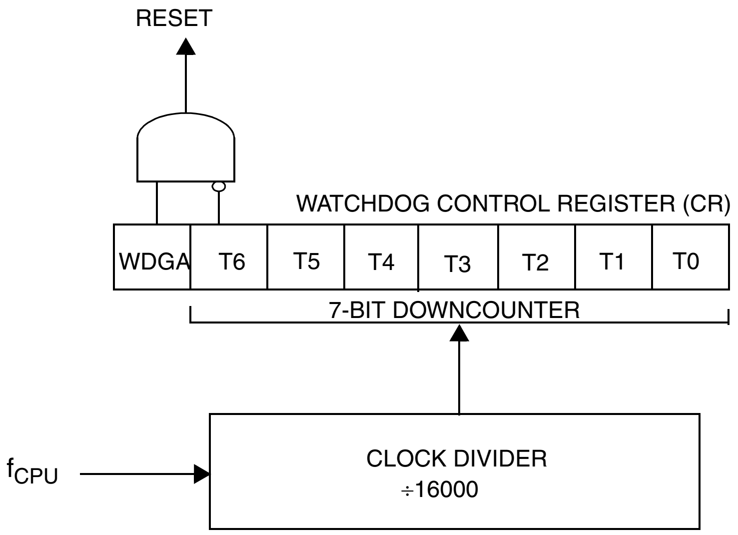
- Programmable free-running downcounter (64 increments of 16000 CPU cycles)
- Programmable reset
- Reset (if watchdog activated) when the T6 bit reaches zero
- Optional reset on HALT instruction (configurable by option byte)
- Hardware Watchdog selectable by option byte

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 30µs.

Figure 32. Watchdog Block Diagram



WATCHDOG TIMER (Cont'd)

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 13 .Watchdog Timing):

- 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 cannot be disabled, except by a reset.

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

If the watchdog is activated, the HALT instruction will generate a Reset.

Table 13.Watchdog Timing

f_{CPU} = 8MHz		
WDG Counter Code	**min [ms]**	**max [ms]**
C0h	1	2
FFh	127	128

Notes: The timing variation shown in Table 13 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 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 15.1 on page 158.

11.1.4.1 Using Halt Mode with the WDG (WDGHALT option)

If Halt mode with Watchdog is enabled by 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.

WATCHDOG TIMER (Cont'd)

11.1.5 Interrupts

None.

11.1.6 Register Description

CONTROL REGISTER (CR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

Bit 7 = **WDGA** *Activation bit.*
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

Note: This bit is not used if the hardware watchdog option is enabled by option byte.

Bit 6:0 = **T[6:0]** *7-bit timer (MSB to LSB).*
These bits contain the decremented value. A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

WATCHDOG TIMER (Cont'd)

Table 14. 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 6 external pins:

– Four PWM outputs

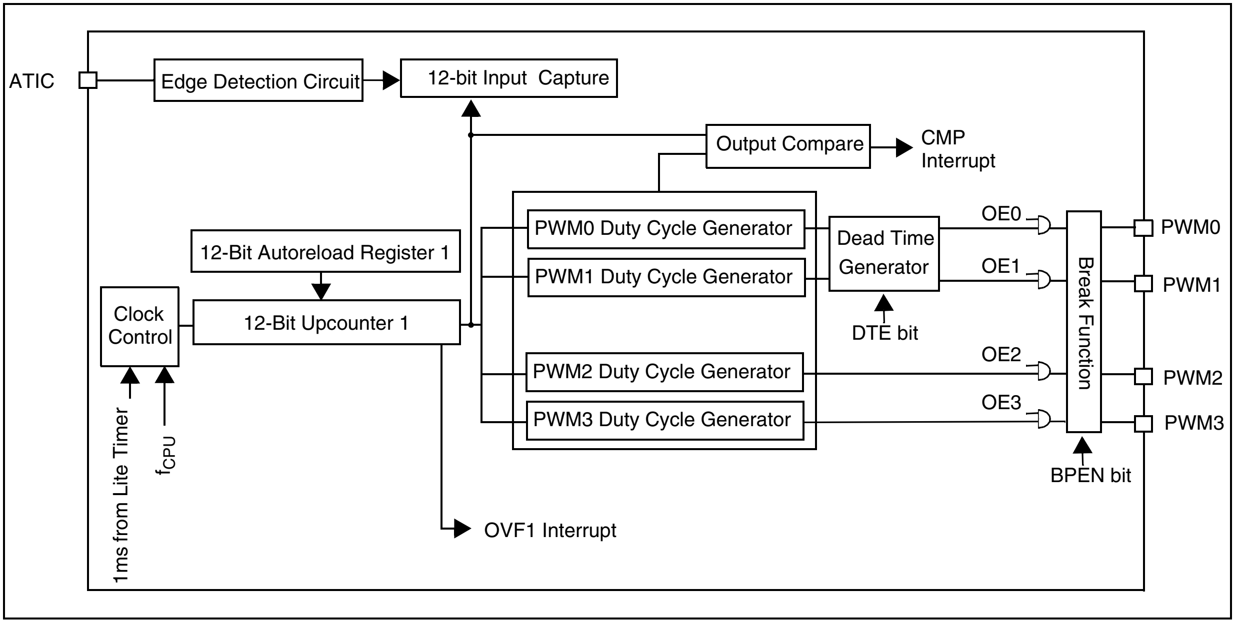
– ATIC/LTIC pin for the Input Capture function

– BREAK pin for forcing a break condition on the PWM outputs

11.2.2 Main Features

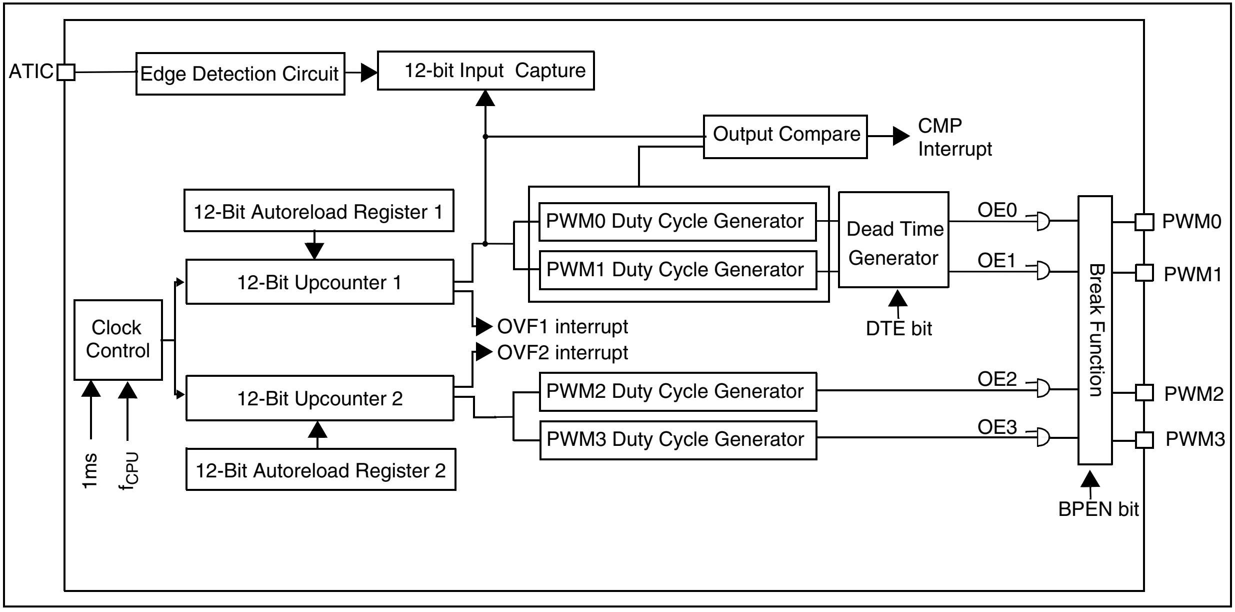
- Single Timer or Dual Timer mode with two 12-bit upcounters (CNTR1/CNTR2) and two 12-bit autoreload registers (ATR1/ATR2)
- Maskable overflow interrupts
- PWM mode
  - – Generation of four independent PWMx signals
  - – Dead time generation for Half bridge driving mode with programmable dead time
  - – Frequency 2KHz-4MHz (@ 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 33. Single Timer Mode (ENCNTR2=0)



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

Figure 34. Dual Timer Mode (ENCNTR2=1)



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

11.2.3 Functional Description

11.2.3.1 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 33 and Figure 34).

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:

$$\text{Resolution} = 1 / (4096 - ATR)$$

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.

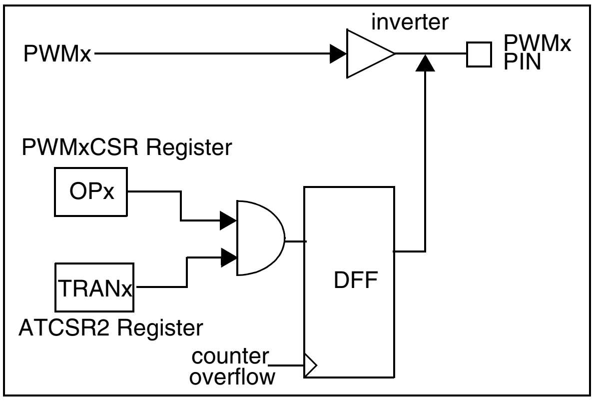
When a upcounter overflow occurs (OVF event), the preloaded Duty cycle values are transferred to the active Duty Cycle registers and the PWMx signals are set to a high level. When the upcounter matches the active DCRx value the PWMx signals are set to a low level. To obtain a signal on a PWMx pin, the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

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

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 35.

Figure 35. 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.

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

Figure 36. PWM Function

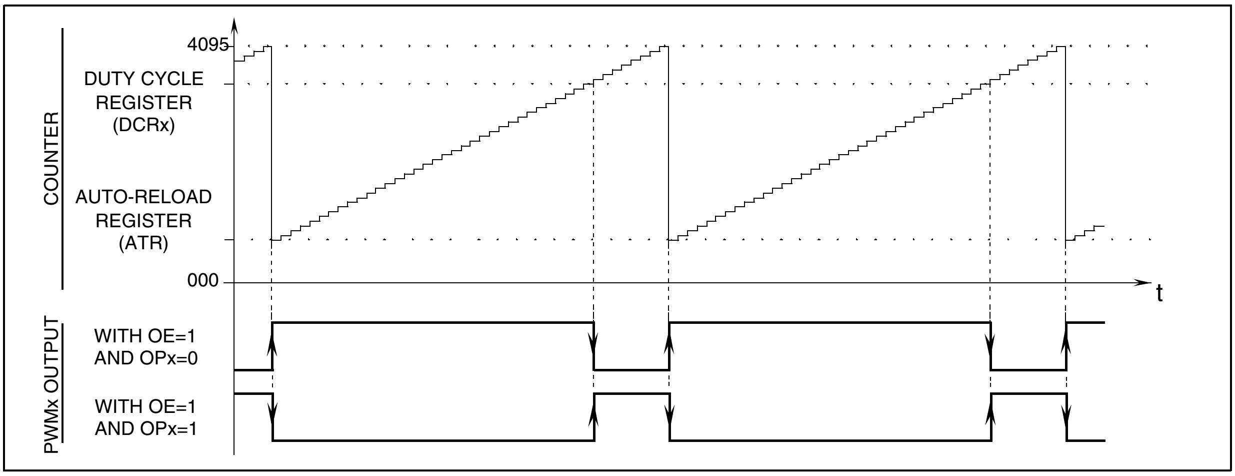
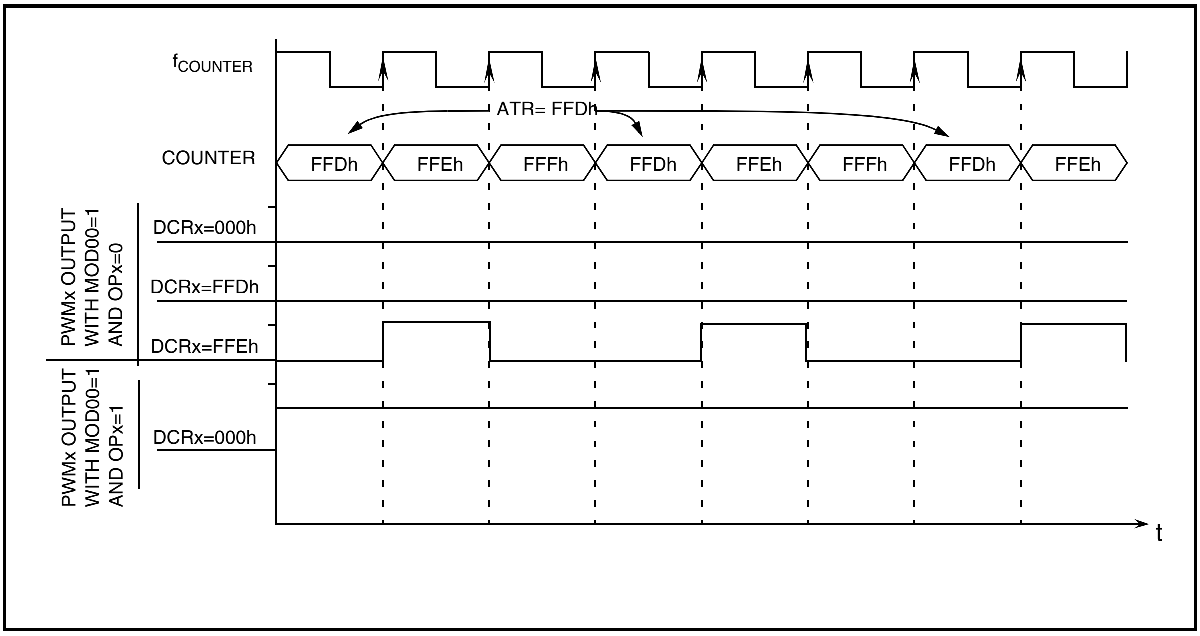


Figure 37. PWM Signal from 0% to 100% Duty Cycle



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

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

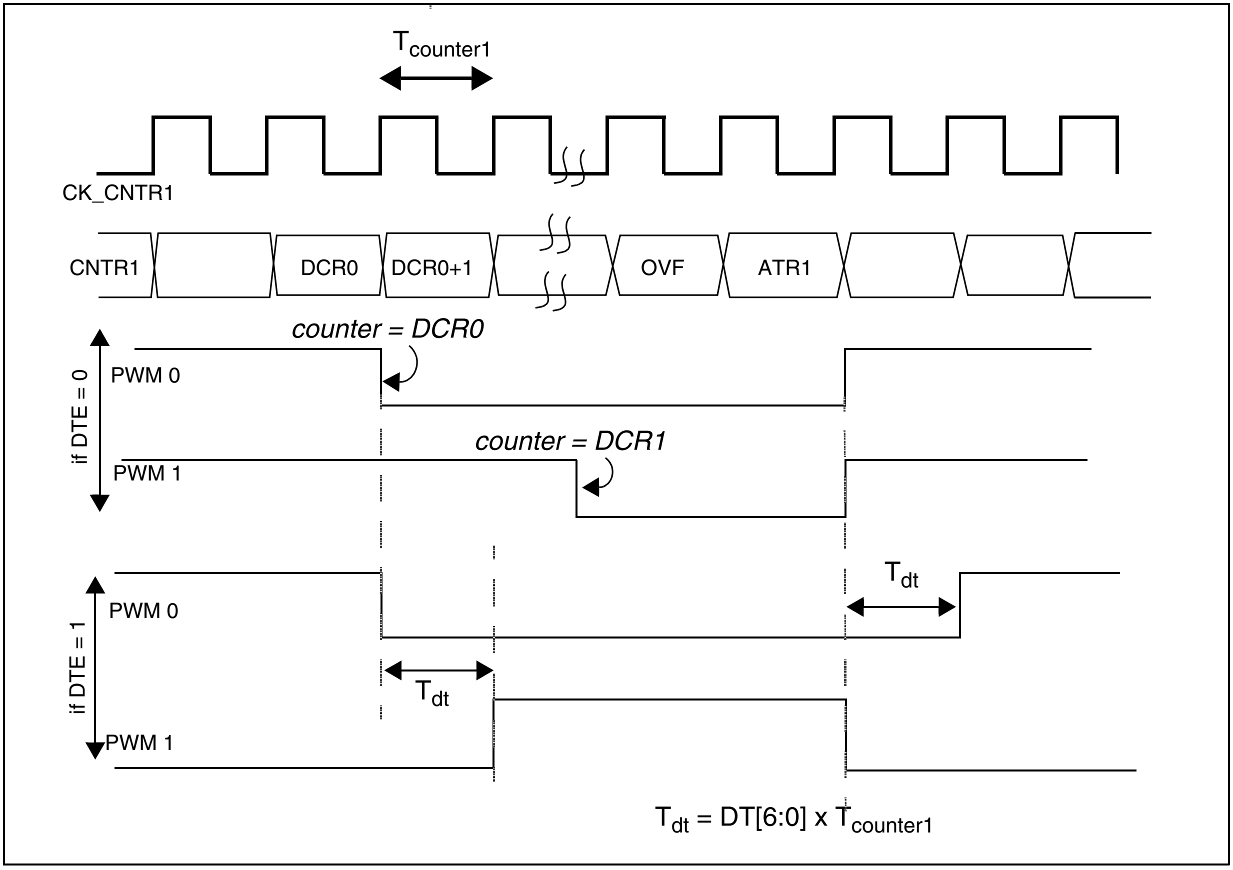
DTGR[7:0] is buffered inside so as to avoid deforming the current PWM cycle. The DTGR effect will take place only after an overflow.

Notes:

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, i.e. if OP0 and OP1 are not set. If polarity is inverted, overlapping PWM0/PWM1 signals will be generated.

Figure 38. Dead Time Generation



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

- PWM goes low at DCR0 match and goes high at ATR1+Tdt
- PWM1 goes high at DCR0+Tdt and goes low at ATR match.

With this programmable delay (Tdt), the PWM0 and PWM1 signals which are generated are not overlapped.

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

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 4 PWM signals are stopped.

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

When the break function is activated (BA bit =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, i.e. 00h.

– The 12-bit PWM counter CNTR2 is put to its reset value,i.e. 00h.

– ATR1, ATR2, Preload and Active DCRx are put to their reset values.

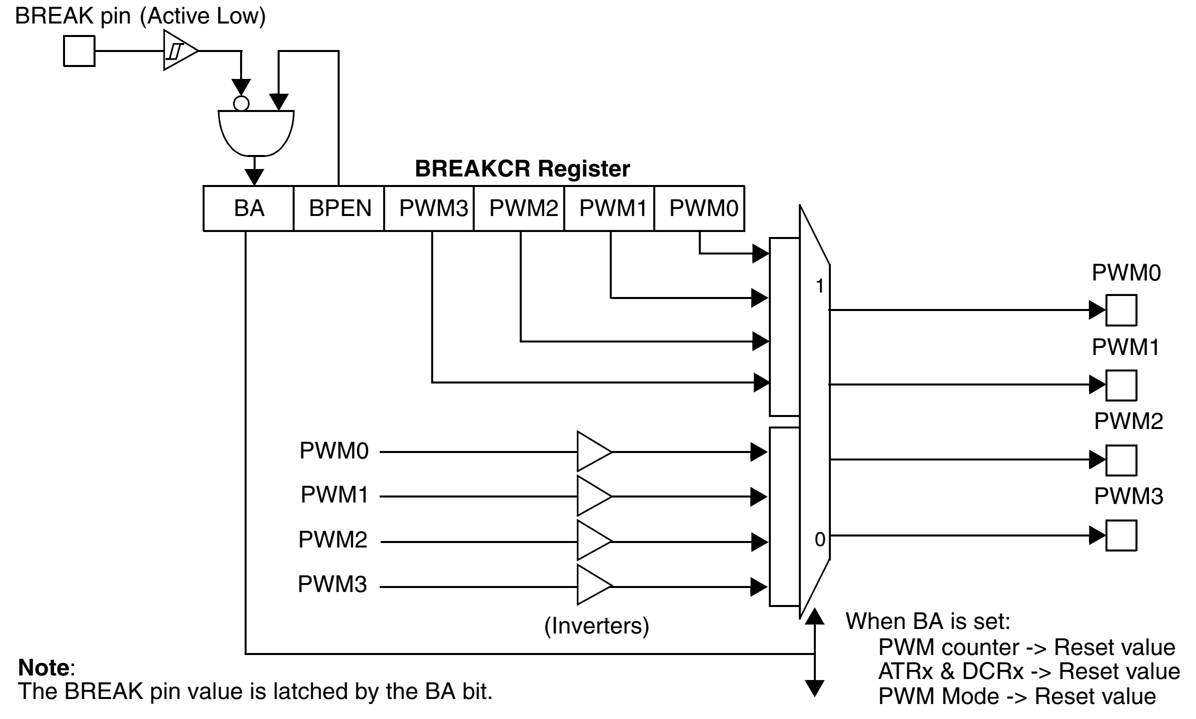
– The PWMCR register is reset.

– 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 4 PWM outputs is transferred to the port registers.

Figure 39. Block Diagram of Break Function



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

11.2.3.2 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.

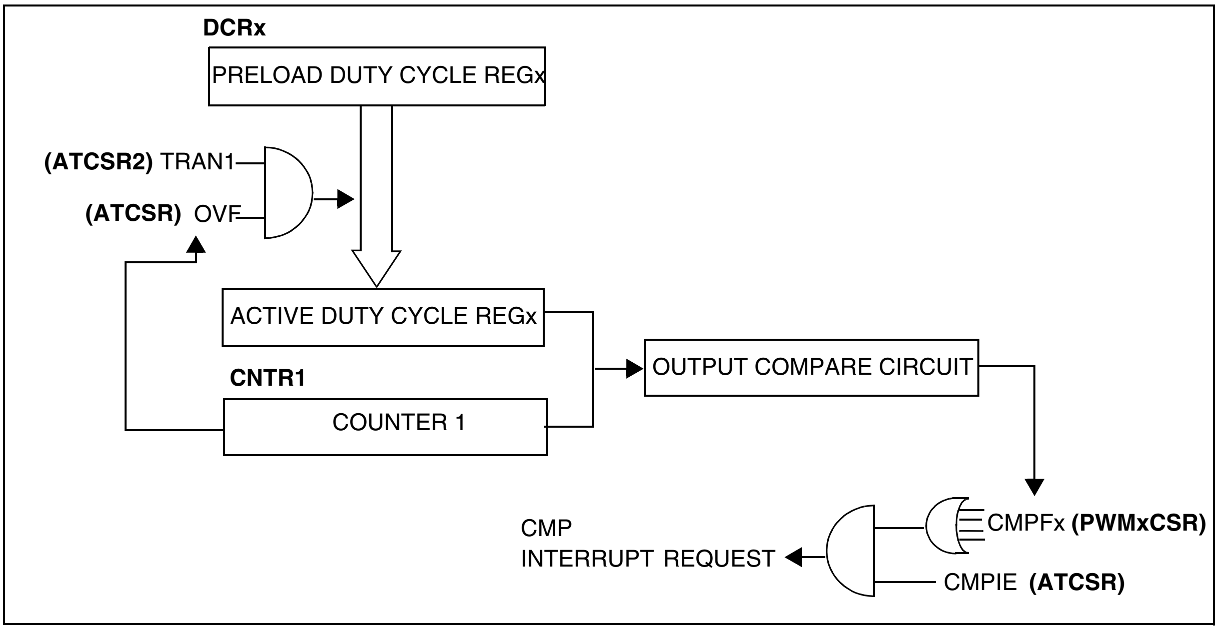
The output compare function is always performed on CNTR1 in both Single Timer mode and Dual Timer mode, and never on CNTR2. The difference is that in Single Timer mode the counter 1 can be compared with any of the four DCR registers, and in Dual Timer mode, counter 1 is compared with DCR0 or DCR1.

Notes:

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 in 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 40. Block Diagram of Output Compare Mode (single timer)



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

11.2.3.3 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 41. Block Diagram of Input Capture Mode

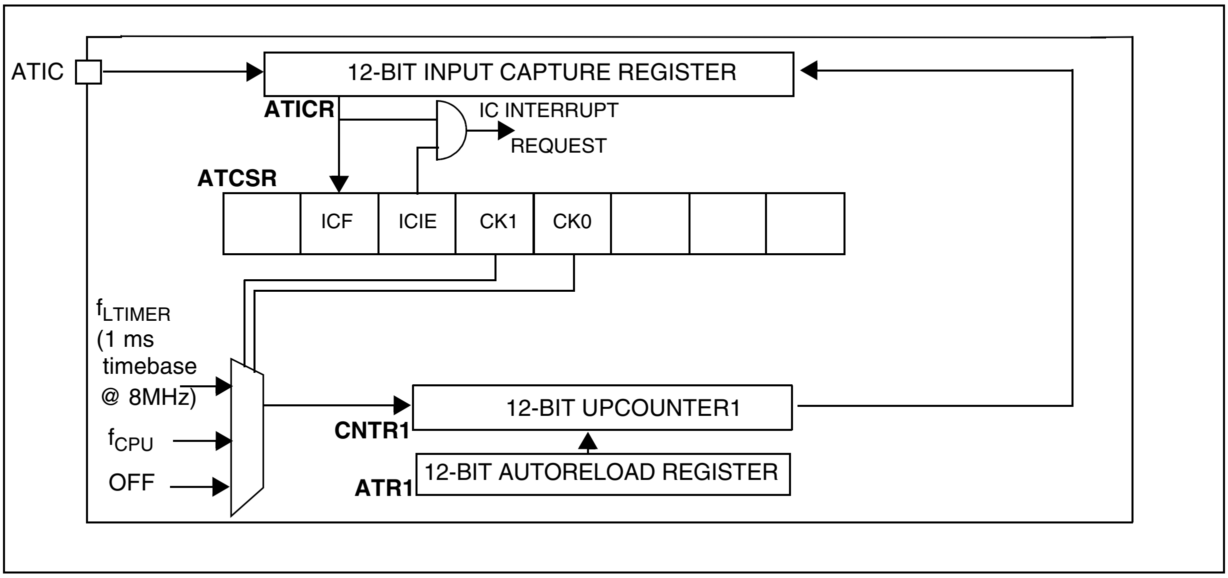
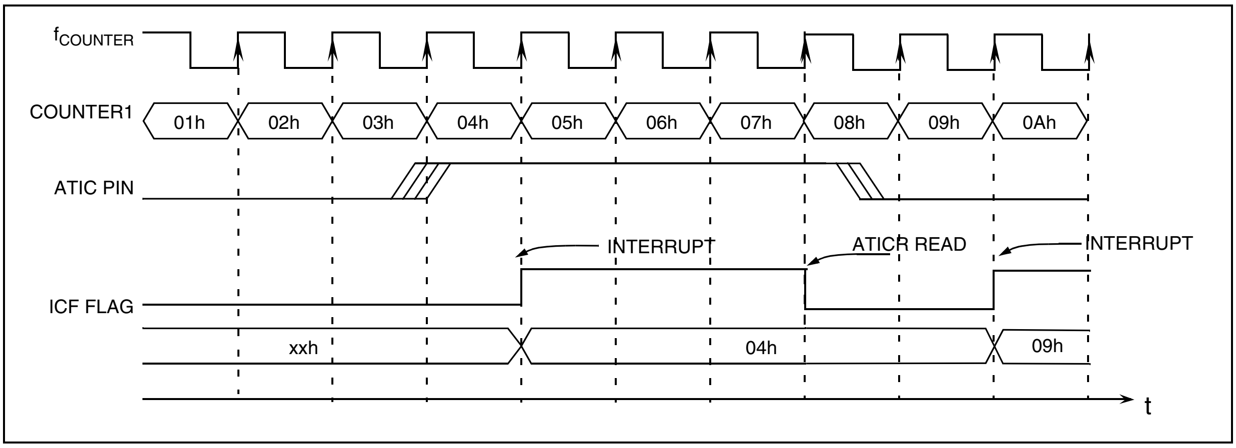


Figure 42. Input Capture timing diagram



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

- **Long input capture**

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

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

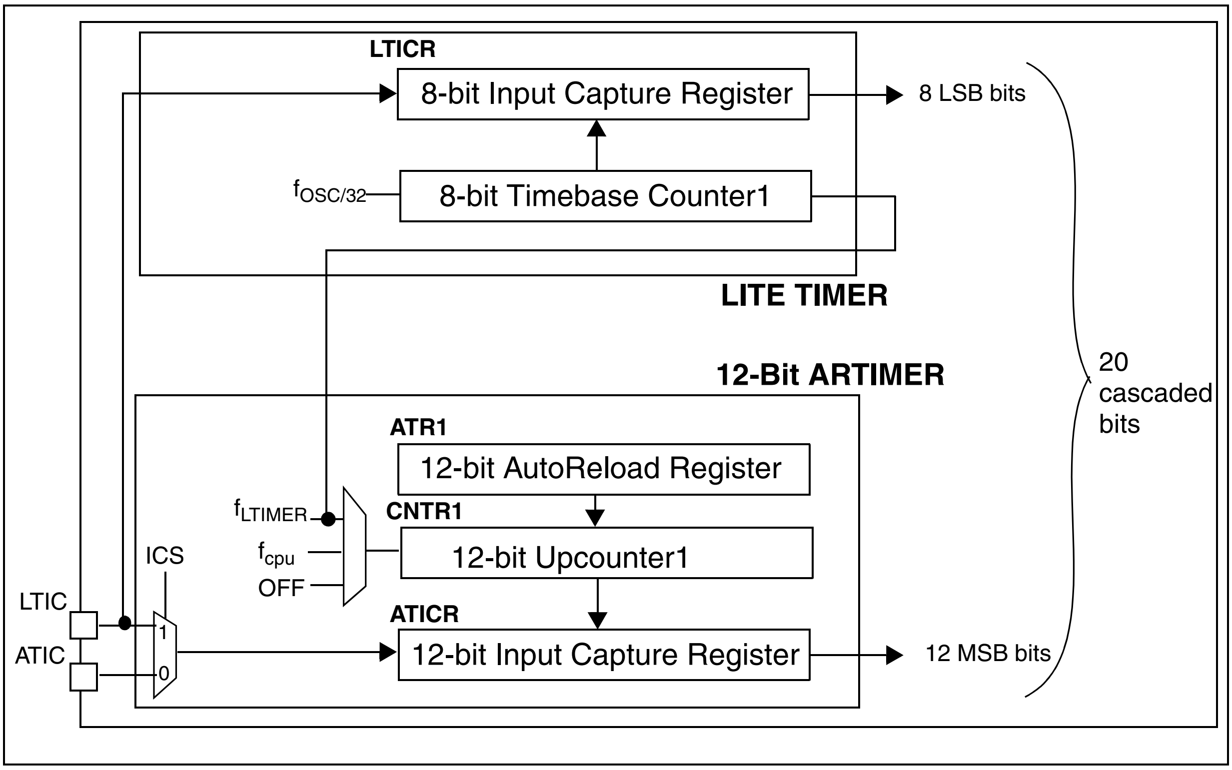
– The ICS bit in the ATCSR2 register is set so that the LTIC pin is used to trigger the AT3 Timer capture.

– The signal to be captured is connected to LTIC pin

– Input Capture registers LTICR, ATICRH and ATICRL are read

This configuration allows to cascade the Lite Timer and the 12-bit AT3 Timer to get a 20-bit input capture value. Refer to Figure 43.

Figure 43. Long Range Input Capture Block Diagram



Notes:

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

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

– First, reset both ICIE bits.

– Then set the ICS bit.

– Reset both ICF bits.

– And then set the ICIE bit of desired interrupt.

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

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

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

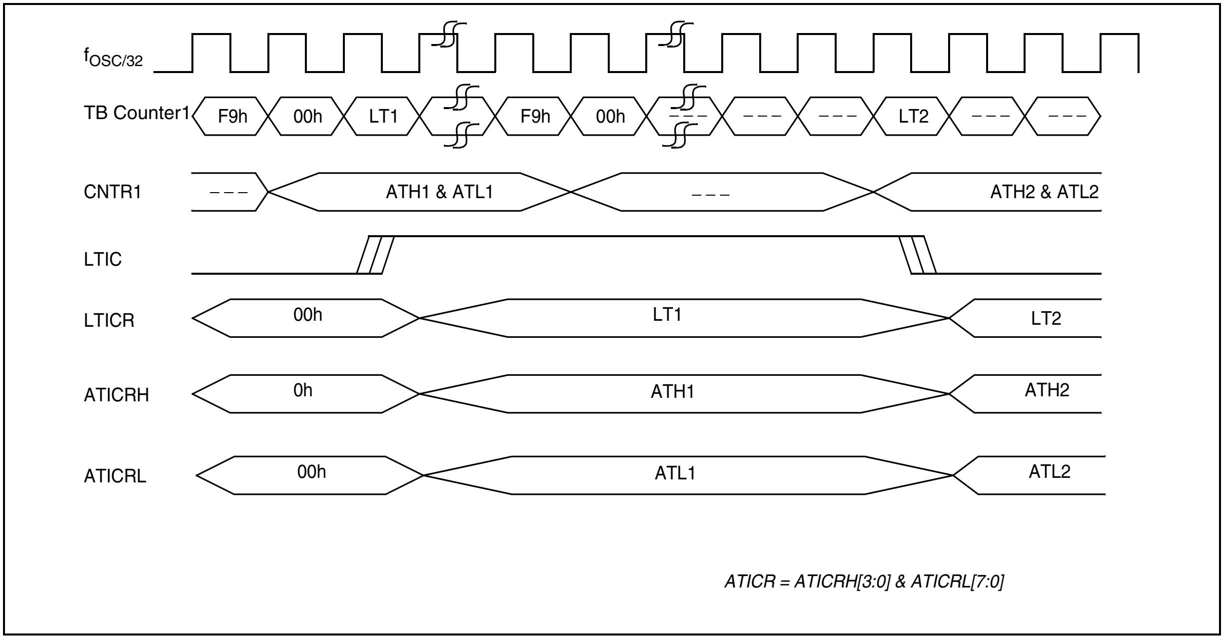
Refer to Figure 44 on page 63.

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

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

Figure 44. Long Range Input Capture Timing Diagram



ATICR = ATICRH[3:0] & ATICRL[7:0]

11.2.4 Low Power Modes

Mode	Description
SLOW	The input frequency is divided by 32
WAIT	No effect on AT timer
ACTIVE-HALT	AT timer halted except if CK0=1, CK1=0 and OVFIE=1
HALT	AT timer halted.

11.2.5 Interrupts

Interrupt Event [1]	Event Flag	Enable Control Bit	Exit from WAIT	Exit from HALT	Exit from ACTIVE -HALT
Overflow Event	OVF1	OVIE1	Yes	No	Yes[2]
AT3 IC Event	ICF	ICIE	Yes	No	No
CMP Event	CMPFx	CMPIE	Yes	No	No

Note 1: The CMP and AT3 IC events are connected to the same interrupt vector.
The OVF event is mapped on a separate vector (see Interrupts chapter).
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).

Note 2: Only if CK0=1 and CK1=0 ($f_{COUNTER} = f_{LTIMER}$)

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

11.2.6 Register Description

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

7	6						0
0	ICF	ICIE	CK1	CK0	OVF1	OVFIE1	CMPIE

Bit 7 = Reserved.

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

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

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

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

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

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

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

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

15							8
0	0	0	0	CNTR1_ 11	CNTR1_ 10	CNTR1_ 9	CNTR1_ 8

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

7							0
CNTR1_ 7	CNTR1_ 6	CNTR1_ 5	CNTR1_ 4	CNTR1_ 3	CNTR1_ 2	CNTR1_ 1	CNTR1_ 0

Bits 15:12 = Reserved.

Bits 11:0 = **CNTR1[11:0]** *Counter Value*.
This 12-bit register is read by software and cleared by hardware after a reset. The counter CNTR1 is incremented continuously as soon as a counter clock is selected. To obtain the 12-bit CNTR1 value, software should read the counter value in two consecutive read operations, LSB first. When a counter overflow occurs, the counter restarts from the value specified in the ATR1 register.

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

AUTORELOAD REGISTER (ATR1H)

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

15							8
0	0	0	0	ATR11	ATR10	ATR9	ATR8

AUTORELOAD REGISTER (ATR1L)

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

7							0
ATR7	ATR6	ATR5	ATR4	ATR3	ATR2	ATR1	ATR0

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

PWM OUTPUT CONTROL REGISTER (PWMCR)

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

7							0
0	OE3	0	OE2	0	OE1	0	OE0

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

PWMx CONTROL STATUS REGISTER (PWMxCSR)

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

7	6						0
0	0	0	0	0	0	OPx	CMPFx

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

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

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

BREAK CONTROL REGISTER (BREAKCR)

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

7							0
0	0	BA	BPEN	PWM3	PWM2	PWM1	PWM0

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

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

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

Bit 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

Bit 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.

PWMx DUTY CYCLE REGISTER HIGH (DCRxH)

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

15							8
0	0	0	0	DCR11	DCR10	DCR9	DCR8

PWMx DUTY CYCLE REGISTER LOW (DCRxL)

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

7							0
DCR7	DCR6	DCR5	DCR4	DCR3	DCR2	DCR1	DCR0

Bits 15:12 = Reserved.

Bits 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 36).

In PWM mode (OEx=1 in the PWMCR register) the DCR[11:0] bits define the duty cycle of the PWMx output signal (see Figure 36). In Output Compare mode, they define the value to be compared with the 12-bit upcounter value.

INPUT CAPTURE REGISTER HIGH (ATICRH)

Read only
Reset Value: 0000 0000 (00h)

15							8
0	0	0	0	ICR11	ICR10	ICR9	ICR8

INPUT CAPTURE REGISTER LOW (ATICRL)

Read only
Reset Value: 0000 0000 (00h)

7							0
ICR7	ICR6	ICR5	ICR4	ICR3	ICR2	ICR1	ICR0

Bits 15:12 = Reserved.

Bits 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 captured the 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)

Read/Write
Reset Value: 0000 0011 (03h)

7							0
0	0	ICS	OVFIE2	OVF2	ENCNTR2	TRAN2	TRAN1

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

Bit 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

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

Bit 4 = **OVFIE2** *Overflow interrupt 2 enable*
This bit is read/write by software and controls the overflow interrupt of counter2.
0: Overflow interrupt disabled.
1: Overflow interrupt enabled.

Bit 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 counter2 from FFFh to ATR2 value.
0: No counter overflow occurred
1: Counter overflow occurred

Bit 2 = **ENCNTR2** *Enable counter2*

This bit is read/write be software and switches the second counter CNTR2. If this bit is set, PWM2 and PWM3 will be generated using CNTR2.
0: CNTR2 stopped.
1: CNTR2 starts running.

Bit 1= **TRAN2** *Transfer enable2*
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.

(*Only DCR2/DCR3 can be controlled with this bit*)

Bit 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 (ATR2H)
Read / Write
Reset Value: 0000 0000 (00h)

15							8
0	0	0	0	ATR11	ATR10	ATR9	ATR8

AUTORELOAD REGISTER (ATR2L)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
ATR7	ATR6	ATR5	ATR4	ATR3	ATR2	ATR1	ATR0

Bits 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)

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

7							0
DTE	DT6	DT5	DT4	DT3	DT2	DT1	DT0

Bits 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.

Bit 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

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

Table 15. Register Map and Reset Values

Address (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

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
21	**ATCSR2** Reset Value	0	0	ICS 0	OVFIE2 0	OVF2 0	ENCNTR2 0	TRAN2 1	TRAN1 1
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)

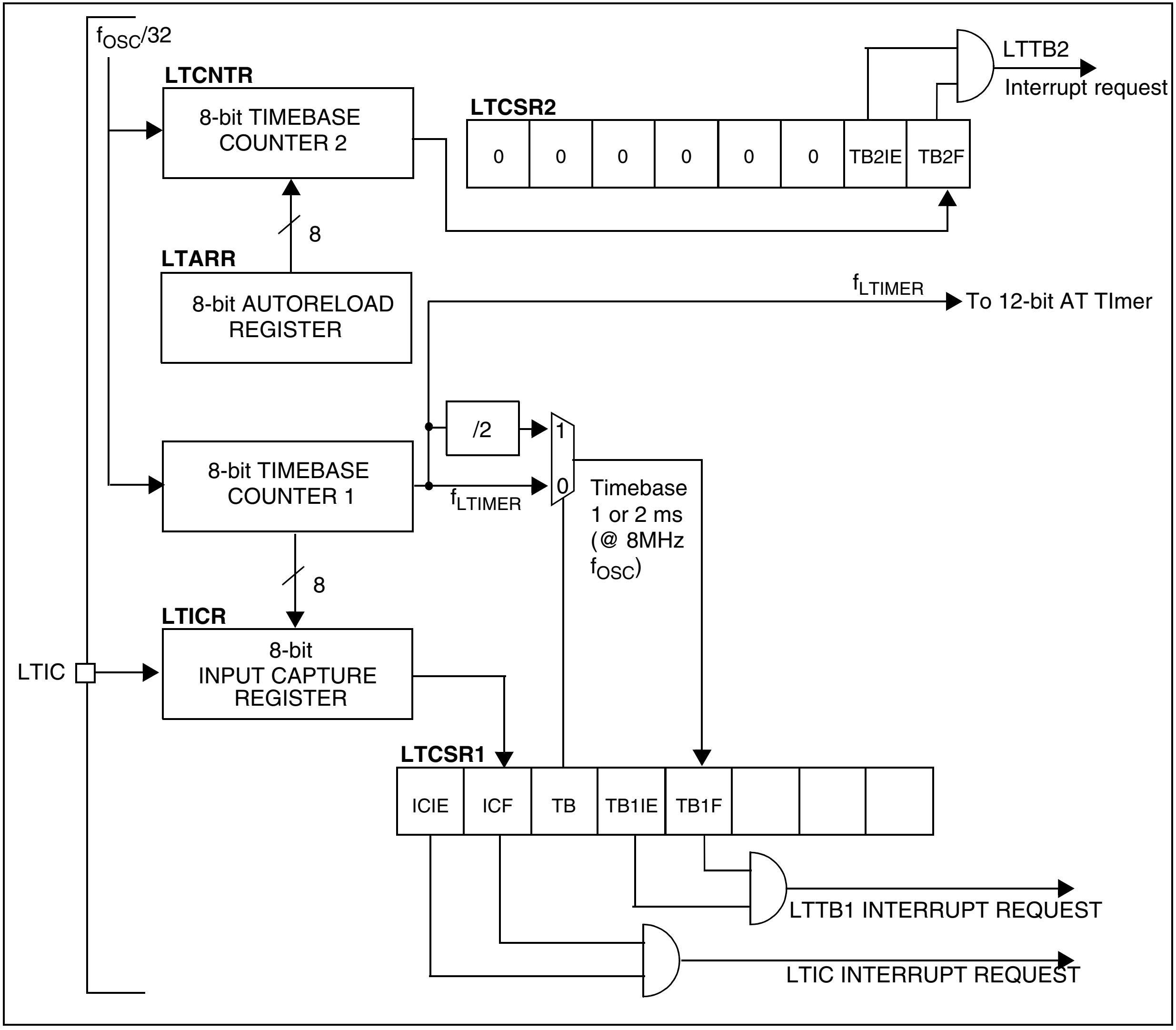
11.3.1 Introduction

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

11.3.2 Main Features

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

Figure 45. Lite Timer 2 Block Diagram



LITE TIMER (Cont'd)

11.3.3 Functional Description

11.3.3.1 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.

11.3.3.2 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.

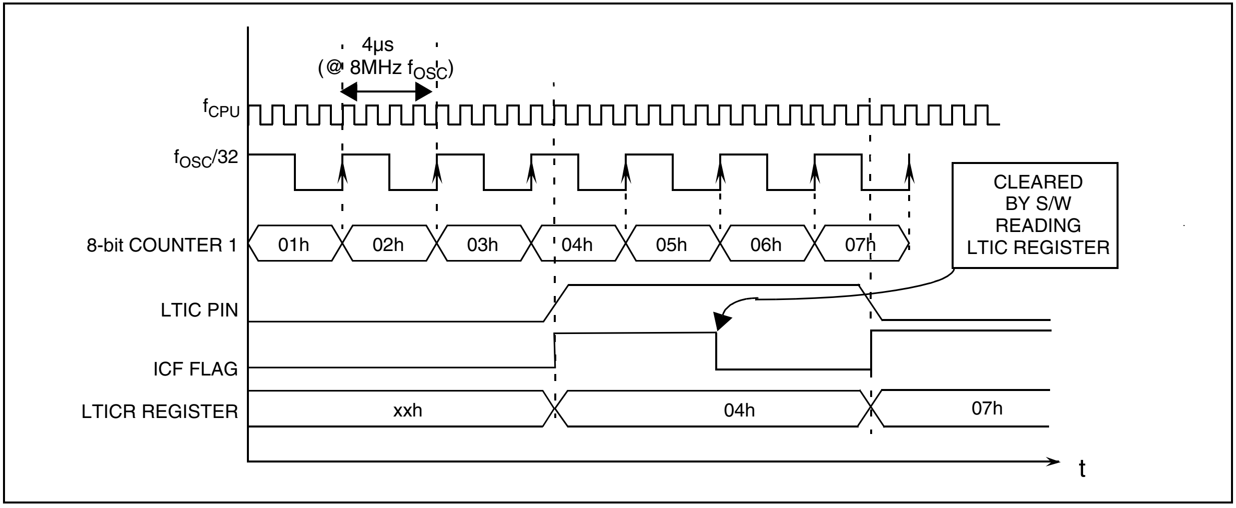
When Counter 2 overflows, the TB2F bit in the LTCSR2 register is set by hardware and an interrupt request is generated if the TB2IE bit is set. The TB2F bit is cleared by software reading the LTCSR2 register.

11.3.3.3 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 46. Input Capture Timing Diagram.



LITE TIMER (Cont'd)

11.3.4 Low Power Modes

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	No effect on Lite timer
HALT	Lite timer stops counting

11.3.5 Interrupts

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	Yes	No	No
IC Event	ICF	ICIE	Yes	No	No

Note: The TBxF and ICF interrupt events are connected to separate interrupt vectors (see Interrupts chapter).

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).

11.3.6 Register Description

LITE TIMER CONTROL/STATUS REGISTER 2 (LTCSR2)
Read / Write
Reset Value: 0x00 0000 (x0h)

7							0
0	0	0	0	0	0	TB2IE	TB2F

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

Bit 1 = **TB2IE** *Timebase 2 Interrupt enable*.
This bit is set and cleared by software.
0: Timebase (TB2) interrupt disabled
1: Timebase (TB2) interrupt enabled

Bit 0 = **TB2F** *Timebase 2 Interrupt Flag*.
This bit is set by hardware and cleared by software reading the LTCSR2 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)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
AR7	AR7	AR7	AR7	AR3	AR2	AR1	AR0

Bits 7:0 = **AR[7:0]** *Counter 2 Reload Value.*
These bits register is read/write by software. The LTARR value is automatically loaded into Counter 2 (LTCNTR) when an overflow occurs.

LITE TIMER (Cont'd)

LITE TIMER COUNTER 2 (LTCNTR)

Read only
Reset Value: 0000 0000 (00h)

7							0
CNT7	CNT7	CNT7	CNT7	CNT3	CNT2	CNT1	CNT0

Bits 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)

Read / Write
Reset Value: 0x00 0000 (x0h)

7							0
ICIE	ICF	TB	TB1IE	TB1F	-	-	-

Bit 7 = **ICIE** *Interrupt Enable.*
This bit is set and cleared by software.
0: Input Capture (IC) interrupt disabled
1: Input Capture (IC) interrupt enabled

Bit 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 initialise the ICF bit by reading the LTICR register

Bit 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)

Bit 4 = **TB1IE** *Timebase Interrupt enable*.
This bit is set and cleared by software.
0: Timebase (TB1) interrupt disabled
1: Timebase (TB1) interrupt enabled

Bit 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

Bits 2:0 = Reserved

LITE TIMER INPUT CAPTURE REGISTER (LTICR)

Read only
Reset Value: 0000 0000 (00h)

7							0
ICR7	ICR6	ICR5	ICR4	ICR3	ICR2	ICR1	ICR0

Bits 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 will be captured when a rising or falling edge occurs on the LTIC pin.

LITE TIMER (Cont'd)

Table 16. 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 3 lines)
- Simplex synchronous transfers (on 2 lines)
- Master or slave operation
- Six master mode frequencies (f_{CPU}/4 max.)
- f_{CPU}/2 max. slave mode frequency (see note)
- $\overline{SS}$ 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 47 shows the serial peripheral interface (SPI) block diagram. There are 3 registers:

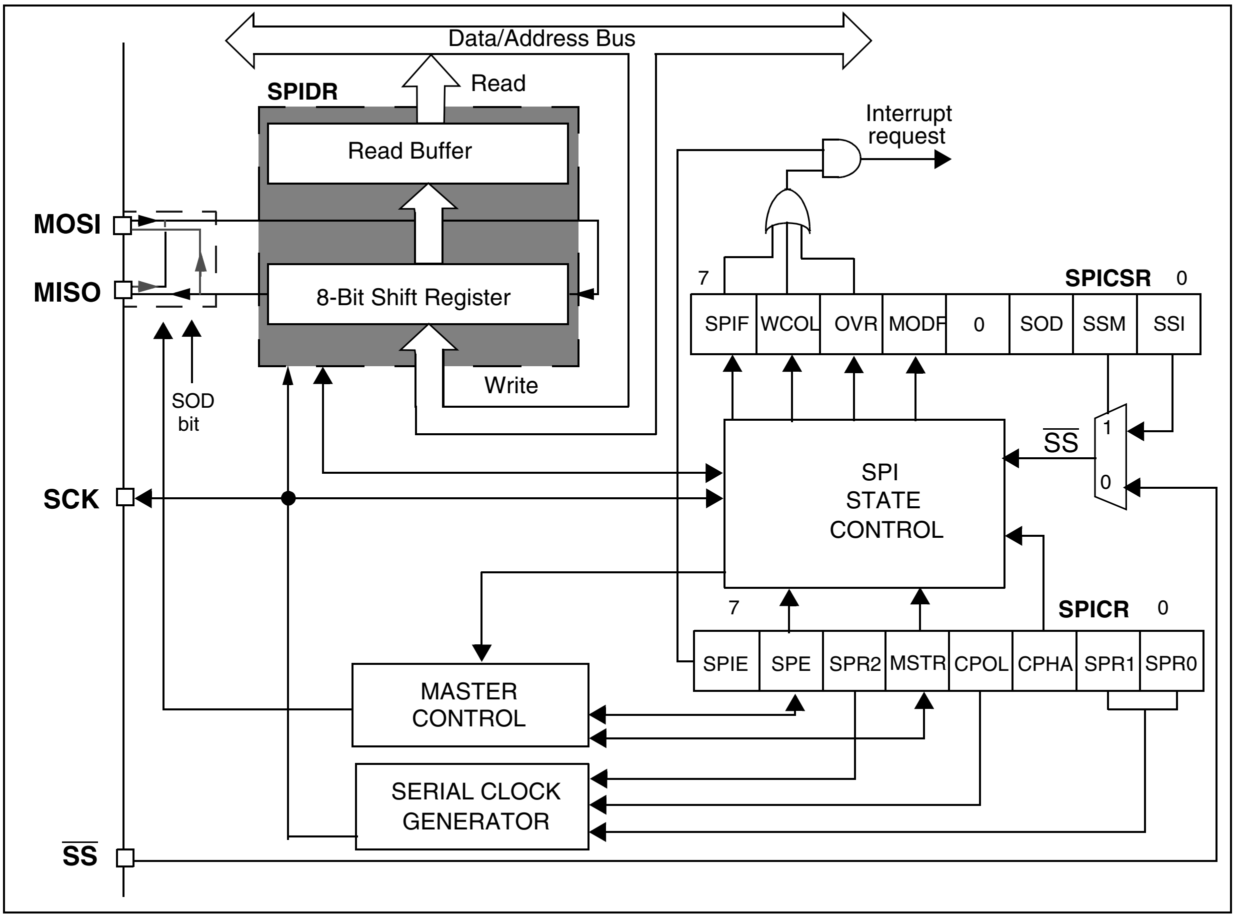
- SPI Control Register (SPICR)
- SPI Control/Status Register (SPICSR)
- SPI Data Register (SPIDR)

The SPI is connected to external devices through 4 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.

SERIAL PERIPHERAL INTERFACE (Cont’d)

Figure 47. Serial Peripheral Interface Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.3.1 Functional Description

A basic example of interconnections between a single master and a single slave is illustrated in Figure 48.

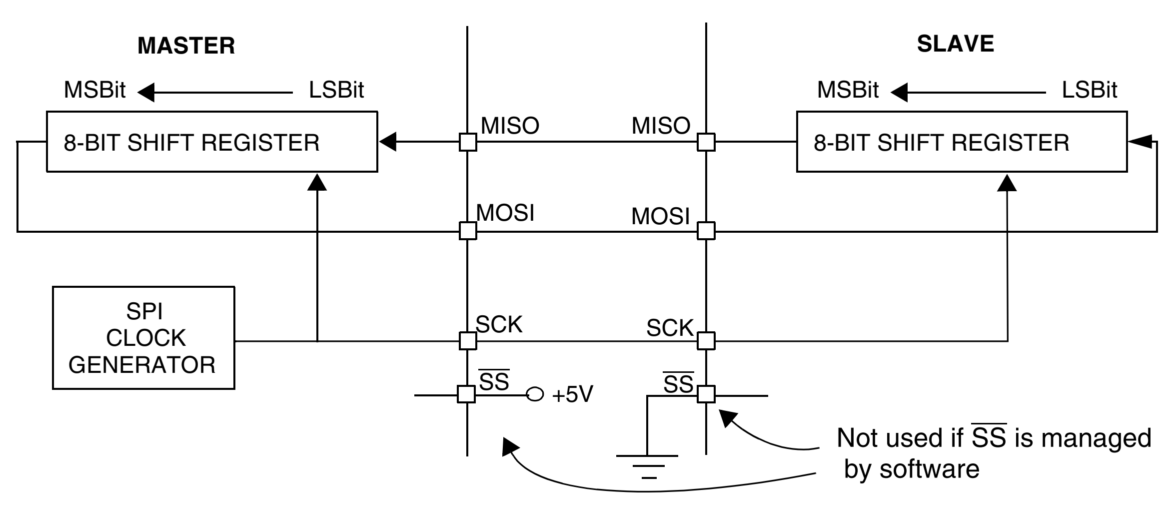
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

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 51) but master and slave must be programmed with the same timing mode.

Figure 48. Single Master/ Single Slave Application



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.3.2 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 50)

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 49):

If CPHA=1 (data latched on 2nd 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 1st 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 will occur when the slave writes to the shift register (see Section 11.4.5.3).

Figure 49. Generic $\overline{SS}$ Timing Diagram

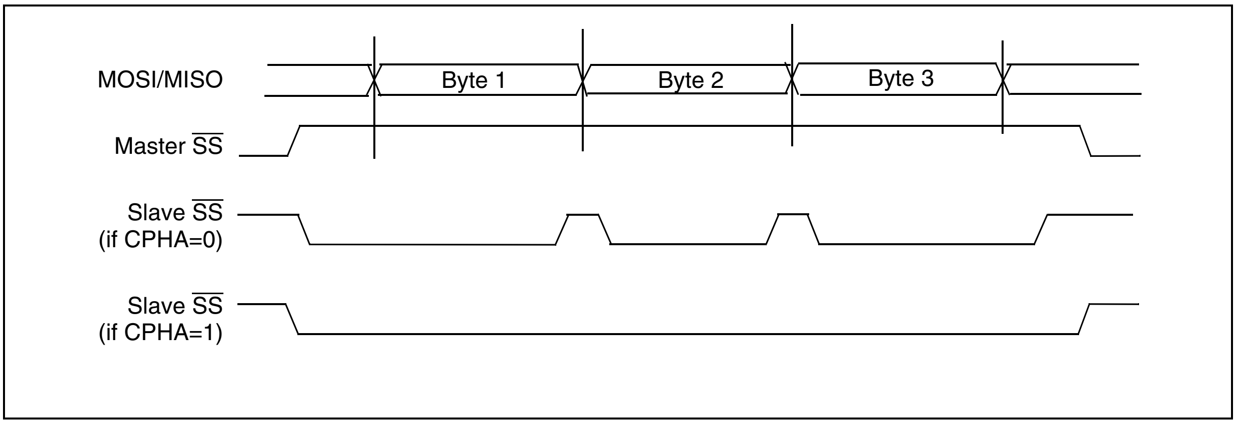
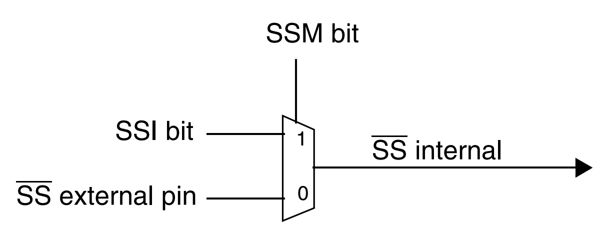


Figure 50. Hardware/Software Slave Select Management



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.3.3 Master Mode Operation

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

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

To operate the SPI in master mode, perform the following steps in order (if the SPICSR register is not written first, the SPICR register setting (MSTR bit) may be not taken into account):

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

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

11.4.3.4 Master Mode Transmit Sequence

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

When data transfer is complete:

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

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

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

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

11.4.3.5 Slave Mode Operation

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

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see Figure 51).
     Note: The slave must have the same CPOL and CPHA settings as the master.
   - Manage the $\overline{SS}$ pin as described in Section 11.4.3.2 and Figure 49. If CPHA=1 $\overline{SS}$ must be held low continuously. If CPHA=0 SS must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

11.4.3.6 Slave Mode Transmit Sequence

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

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

When data transfer is complete:

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

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

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

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

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

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.4 Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 51).

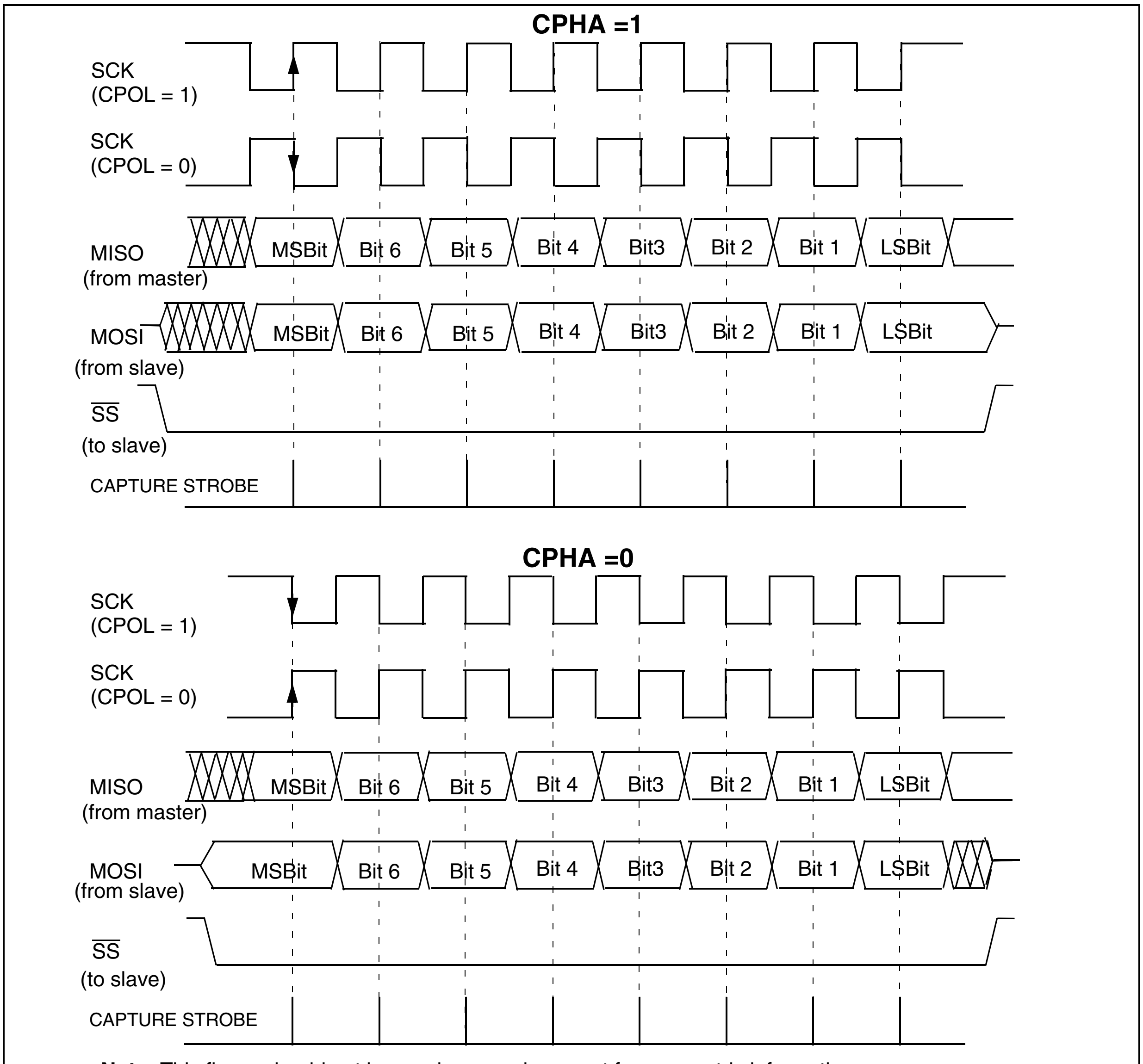
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

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

Figure 51, 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, 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 51. Data Clock Timing Diagram



Note: This figure should not be used as a replacement for parametric information.
Refer to the Electrical Characteristics chapter.

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.5 Error Flags

11.4.5.1 Master Mode Fault (MODF)

Master mode fault occurs when the master device has its $\overline{\text{SS}}$ pin pulled low.

When a Master mode fault occurs:

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

Clearing the MODF bit is done through a software sequence:

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

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

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

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

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

11.4.5.2 Overrun Condition (OVR)

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

When an Overrun occurs:

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

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

The OVR bit is cleared by reading the SPICSR register.

11.4.5.3 Write Collision Error (WCOL)

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

Write collisions can occur both in master and slave mode. See also Section 11.4.3.2 Slave Select Management.

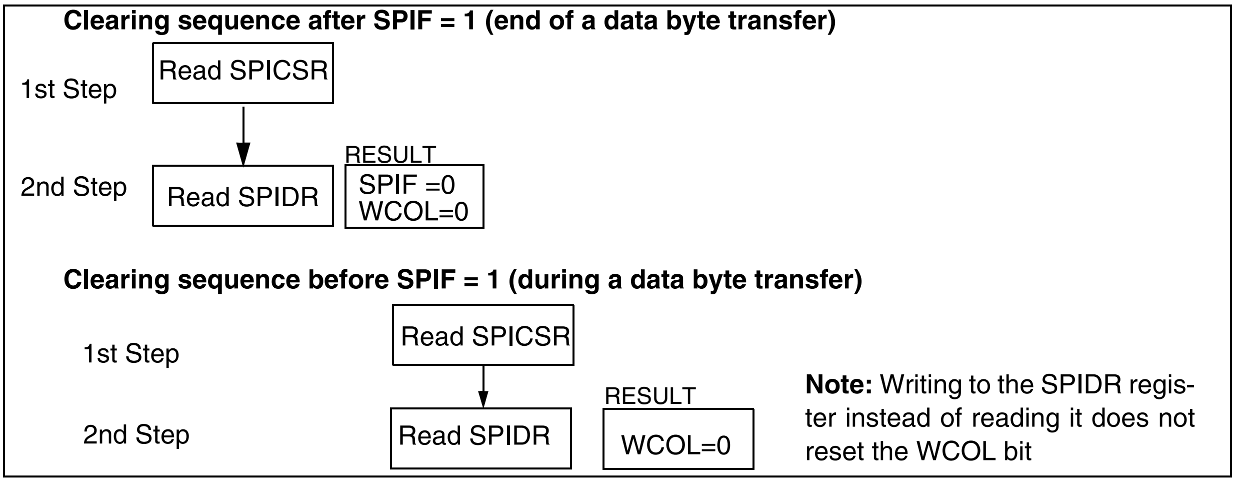
Note: a "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

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

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

Clearing the WCOL bit is done through a software sequence (see Figure 52).

Figure 52. Clearing the WCOL bit (Write Collision Flag) Software Sequence



Note: Writing to the SPIDR register instead of reading it does not reset the WCOL bit

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.5.4 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 53).

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.

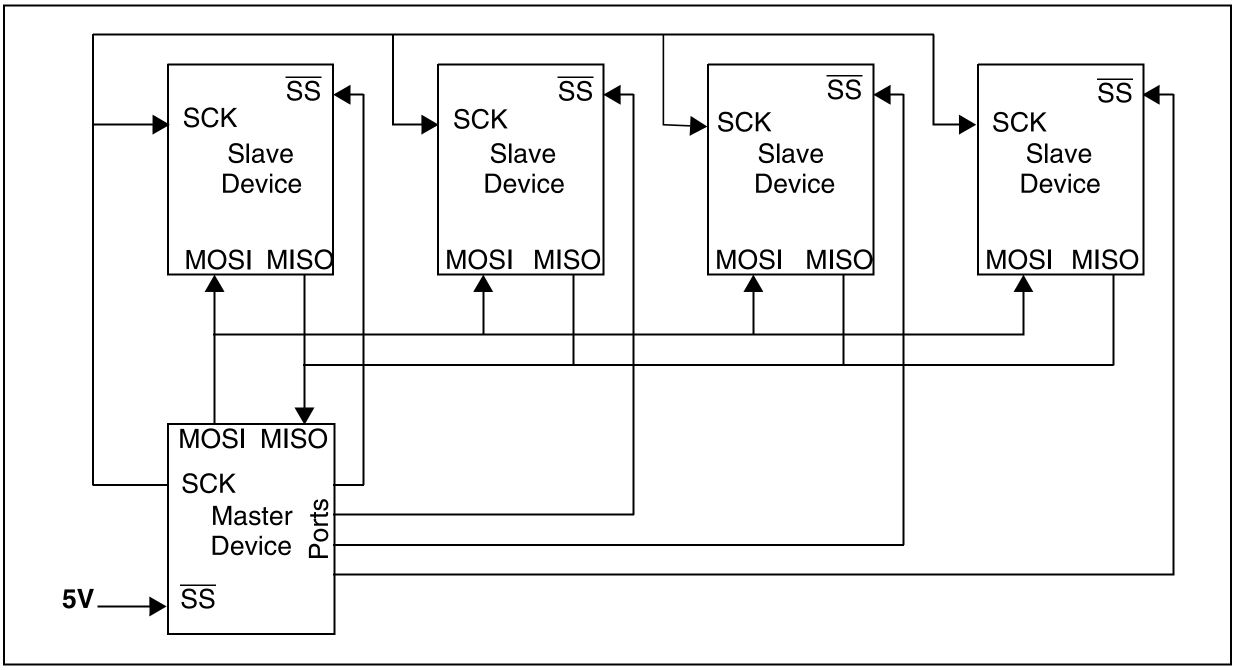
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Multi-Master System

A multi-master 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 multi-master system is principally handled by the MSTR bit in the SPICR register and the MODF bit in the SPICSR register.

Figure 53. Single Master / Multiple Slave Configuration



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.6 Low Power Modes

Mode	Description
WAIT	No effect on SPI. SPI interrupt events cause the Device to exit from WAIT mode.
HALT	SPI registers are frozen. In HALT mode, the SPI is inactive. SPI operation resumes when the 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 wake-up event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the Device.

11.4.6.1 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 Section 11.4.3.2), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters Halt mode.

11.4.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
SPI End of Transfer Event	SPIF	SPIE	Yes	Yes
Master Mode Fault Event	MODF		Yes	No
Overrun Error	OVR		Yes	No

Note: The SPI interrupt events are connected to the same interrupt vector (see Interrupts chapter). They generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.8 Register Description

CONTROL REGISTER (SPICR)

Read/Write

Reset Value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

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

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

Bit 5 = **SPR2** *Divider Enable.*
This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 17 SPI Master mode SCK Frequency.
0: Divider by 2 enabled
1: Divider by 2 disabled

Note: This bit has no effect in slave mode.

Bit 4 = **MSTR** *Master Mode.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$=0 (see Section 11.4.5.1 Master Mode Fault (MODF)).
0: Slave mode
1: Master mode. The function of the SCK pin changes from an input to an output and the functions of the MISO and MOSI pins are reversed.

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

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

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

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

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

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

Table 17. SPI Master mode SCK Frequency

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

SERIAL PERIPHERAL INTERFACE (Cont'd)

CONTROL/STATUS REGISTER (SPICSR)

Read/Write (some bits Read Only)
Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI

Bit 7 = **SPIF** *Serial Peripheral Data Transfer Flag (Read only).*
This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE=1 in the SPICR register. It is cleared by a software sequence (an access to the SPICSR register followed by a write or a read to the SPIDR register).
0: Data transfer is in progress or the flag has been cleared.
1: Data transfer between the Device and an external device has been completed.

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

Bit 6 = **WCOL** *Write Collision status (Read only).*
This bit is set by hardware when a write to the SPIDR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 52).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = **OVR** S*PI Overrun error (Read only).*
This bit is set by hardware when the byte currently being received in the shift register is ready to be transferred into the SPIDR register while SPIF = 1 (See Section 11.4.5.2). An interrupt is generated if SPIE = 1 in the SPICR register. The OVR bit is cleared by software reading the SPICSR register.
0: No overrun error
1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only).*
This bit is set by hardware when the $\overline{\text{SS}}$ pin is pulled low in master mode (see Section 11.4.5.1 Master Mode Fault (MODF)). An SPI interrupt can be generated if SPIE=1 in the SPICR register. This bit is cleared by a software sequence (An access to the SPICSR register while MODF=1 followed by a write to the SPICR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **SOD** *SPI Output Disable.*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode / MISO in slave mode)
0: SPI output enabled (if SPE=1)
1: SPI output disabled

Bit 1 = **SSM** *$\overline{\text{SS}}$ Management.*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI $\overline{\text{SS}}$ pin and uses the SSI bit value instead. See Section 11.4.3.2 Slave Select Management.
0: Hardware management ($\overline{\text{SS}}$ managed by external pin)
1: Software management (internal $\overline{\text{SS}}$ signal controlled by SSI bit. External $\overline{\text{SS}}$ pin free for general-purpose I/O)

Bit 0 = **SSI** *$\overline{\text{SS}}$ Internal Mode.*
This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the $\overline{\text{SS}}$ slave select signal when the SSM bit is set.
0 : Slave selected
1 : Slave deselected

DATA I/O REGISTER (SPIDR)

Read/Write
Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

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

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

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

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

A read to the SPIDR register returns the value located in the buffer and not the content of the shift register (see Figure 47).

Table 18. 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 500K baud.
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- Two 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
- Six 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 re-synchronization 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

LINSCI™ SERIAL COMMUNICATION INTERFACE (Cont'd)

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:

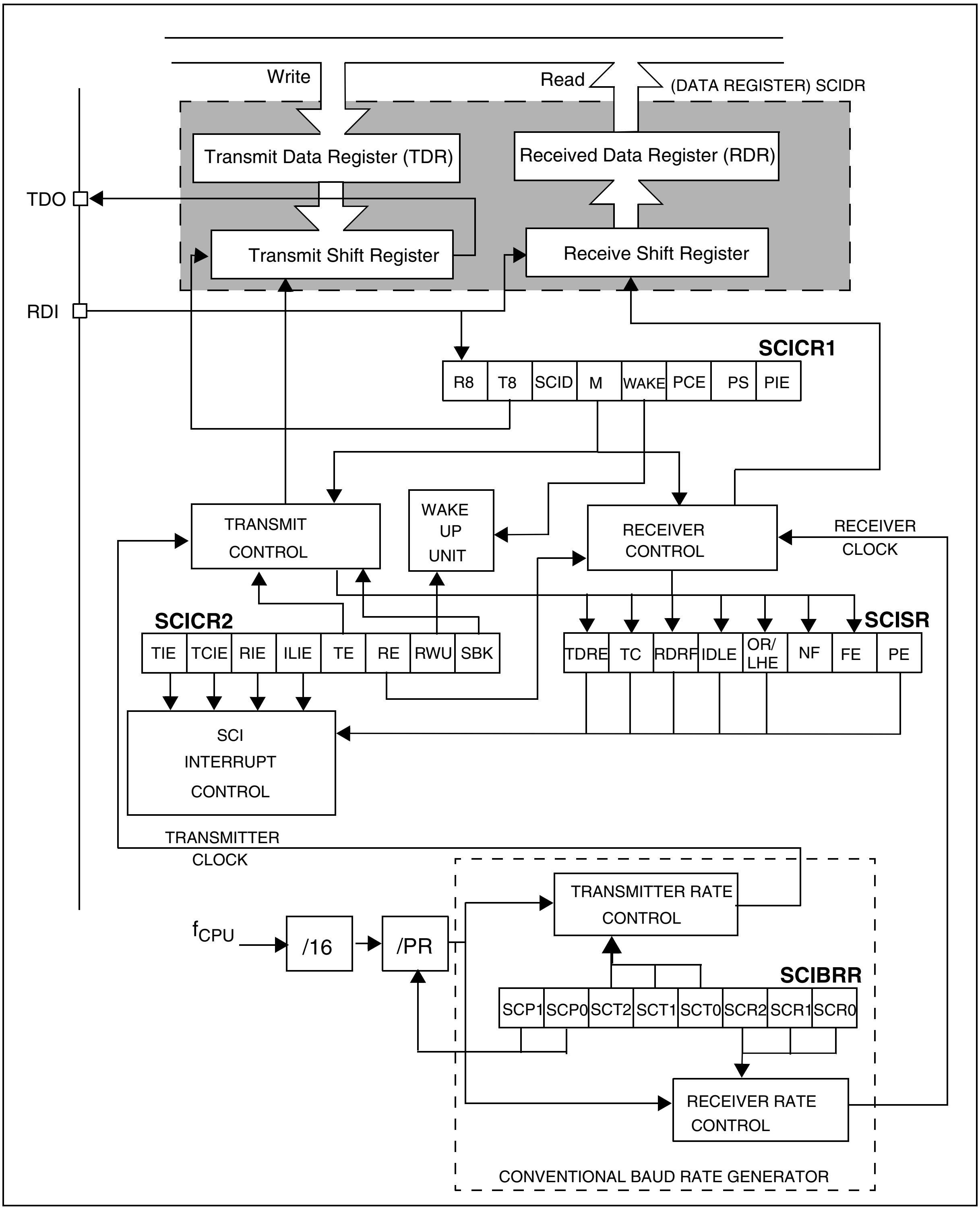
- An Idle Line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) least significant bit first
- A Stop bit indicating that the character is complete.

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.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

Figure 54. SCI Block Diagram (in Conventional Baud Rate Generator Mode)



LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

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 54.

It uses 4 registers:

- Two 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 56.

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

11.5.5.1 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 55).

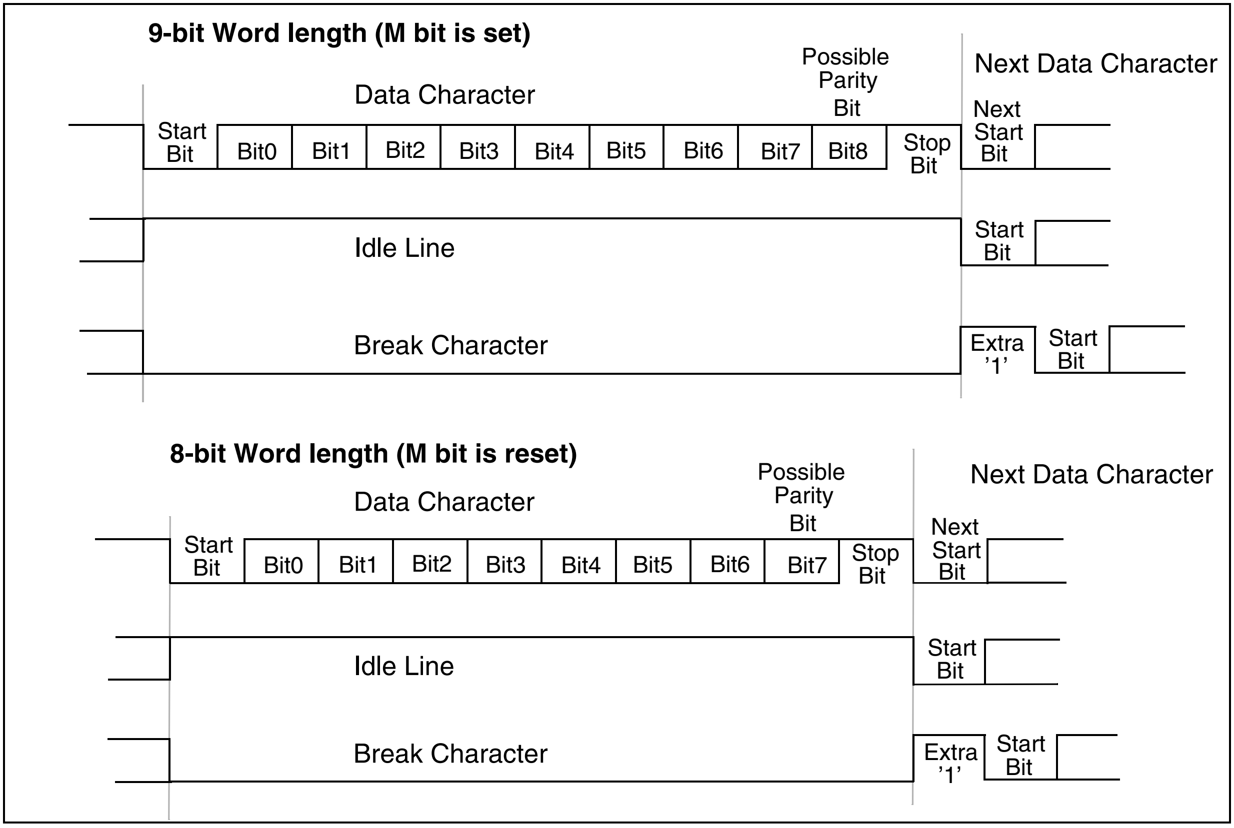
The TDO pin is in low state during the start bit.

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 55. Word length programming



LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

11.5.5.2 Transmitter

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

Character Transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 54).

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 or after the break character) 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 55)

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 i.e. before writing the next byte in the SCIDR.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

11.5.5.3 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 54).

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[|1:0] 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 de-synchronization 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.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

11.5.5.4 Conventional Baud Rate Generation

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

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

with:

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 SCIBRR register.

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.

11.5.5.5 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 56.

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:

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

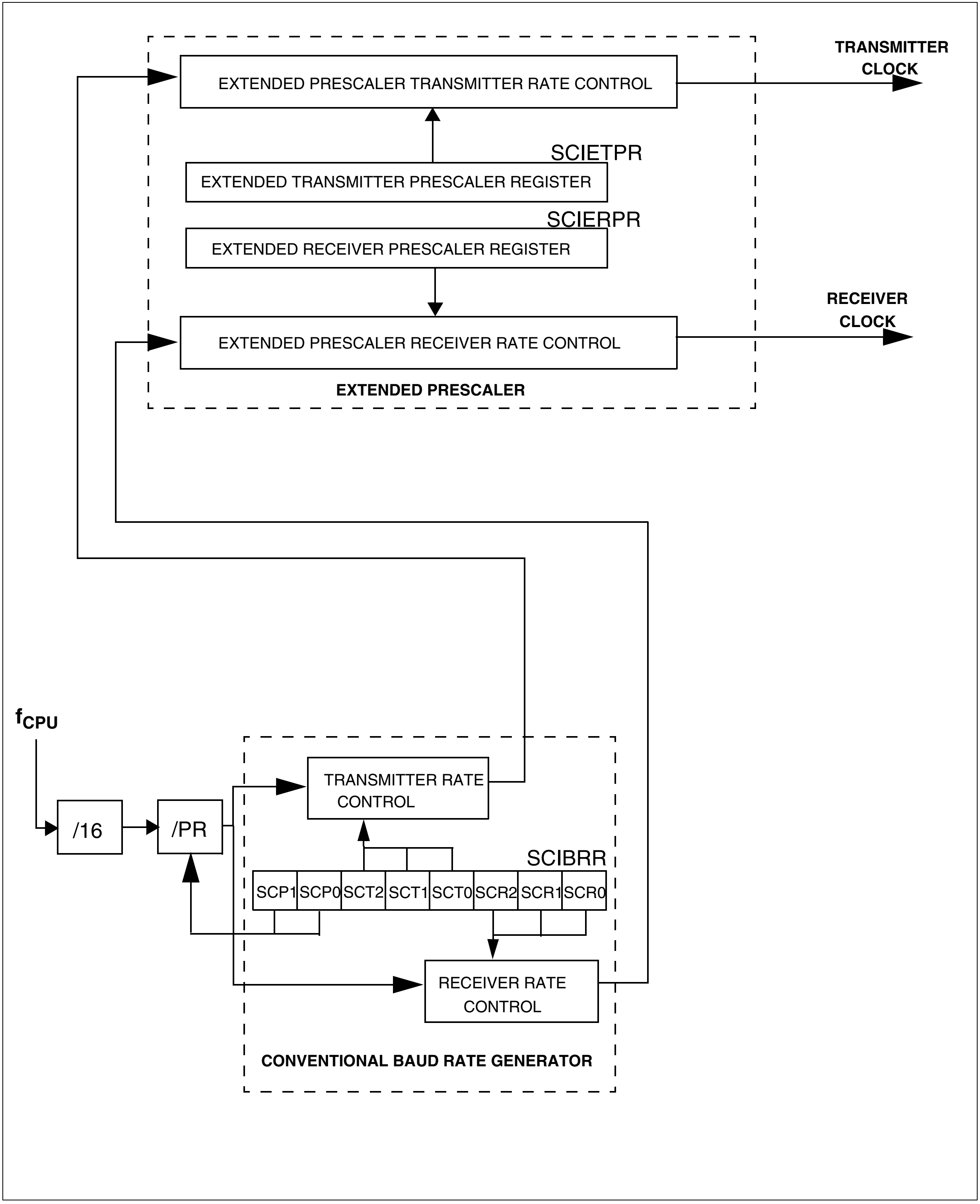
with:

ETPR = 1,..,255 (see SCIETPR register)

ERPR = 1,.. 255 (see SCIERPR register)

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

Figure 56. SCI Baud Rate and Extended Prescaler Block Diagram



LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

11.5.5.6 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 can not 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

Receiver wakes-up by Idle Line detection when the Receive line has recognised 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 receivers is waken up and analyse the first characters of the message which indicates the addressed receiver. 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. At the end of the message, an idle line is sent by the transmitter: this wakes up every receivers 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

Receiver wakes-up by Address Mark detection when it received 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 received an address character (most significant bit ='1'), the receivers are waken 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.

11.5.5.7 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 19.

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 19. Character Formats

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

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.

Ex: 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.

Ex: 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.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

11.5.6 Low Power Modes

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

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	Yes	No
Received Data Ready to be Read	RDRF	RIE	Yes	No
Overrun Error or LIN Synch Error Detected	OR/ LHE		Yes	No
Idle Line Detected	IDLE	ILIE	Yes	No
Parity Error	PE	PIE	Yes	No
LIN Header Detection	LHDF	LHIE	Yes	No

The SCI interrupt events are connected to the same interrupt vector (see Interrupts chapter).

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).

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

11.5.8 SCI Mode Register Description

STATUS REGISTER (SCISR)
Read Only
Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR[1)]	NF[1)]	FE[1)]	PE[1)]

Bit 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

Bit 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.

Bit 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 is ready to be read

Bit 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 will not be set again until the RDRF bit has been set itself (i.e. a new idle line occurs).

Bit 3 = **OR** *Overrun error*

The OR bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register whereas RDRF is still set. 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 detected

Note: When this bit is set, RDR register contents will not be lost but the shift register will be overwritten.

Bit 2 = **NF** *Character 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
1: Noise is detected

Note: This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.

Bit 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
1: Framing error or break character detected

Notes:

– This 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 a frame error and an overrun error, it will be transferred and only the OR bit will be set.

Bit 0 = **PE** *Parity error.*
This bit is set by hardware when a byte 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 detected

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

CONTROL REGISTER 1 (SCICR1)

Read/Write

Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE[1]	PS	PIE

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

Bit 7 = **R8** *Receive data bit 8.*
This bit is used to store the 9th bit of the received word when M=1.

Bit 6 = **T8** *Transmit data bit 8.*
This bit is used to store the 9th bit of the transmitted word when M=1.

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

Bit 4 = **M** *Word length.*
This bit determines the word length. It is set or cleared by software.
0: 1 Start bit, 8 Data bits, 1 Stop bit
1: 1 Start bit, 9 Data bits, 1 Stop bit

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

Bit 3 = **WAKE** *Wake-Up method.*
This bit determines the SCI Wake-Up method, it is set or cleared by software.
0: Idle Line
1: Address Mark

Note: If the LINE bit is set, the WAKE bit is de-activated and replaced by the LHDM bit

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

Bit 1 = **PS** *Parity selection.*
This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity will be selected after the current byte.
0: Even parity
1: Odd parity

Bit 0 = **PIE** *Parity interrupt enable.*
This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). 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

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont’d)

CONTROL REGISTER 2 (SCICR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU[1]	SBK[1]

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

Bit 7 = **TIE** *Transmitter interrupt enable.*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: In SCI interrupt is generated whenever TDRE=1 in the SCISR register

Bit 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

Bit 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

Bit 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.

Bit 3 = **TE** *Transmitter enable.*
This bit enables the transmitter. It is set and cleared by software.
0: Transmitter is disabled
1: Transmitter is enabled

Notes:

– 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.

Bit 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

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

Notes:

– 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.

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

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

DATA REGISTER (SCIDR)

Read/Write

Reset Value: Undefined

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

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

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 54).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 54).

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

BAUD RATE REGISTER (SCIBRR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

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

Bit 7:6= **SCP[1:0]** *First SCI Prescaler*
These 2 prescaling bits allow several standard clock division ranges:

PR Prescaling factor	SCP1	SCP0
1	0	0
3	0	1
4	1	0
13	1	1

Bit 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*
These 3 bits, in conjunction with the SCP1 & SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional Baud Rate Generator mode.

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Bit 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.

RR dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (Cont'd)

EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERPR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
ERPR 7	ERPR 6	ERPR 5	ERPR 4	ERPR 3	ERPR 2	ERPR 1	ERPR 0

Bit 7:0 = **ERPR[7:0]** *8-bit Extended Receive Prescaler Register.*

The extended Baud Rate Generator is activated when a value other than 00h is stored in this register. The clock frequency from the 16 divider (see Figure 56) 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)

Read/Write

Reset Value:0000 0000 (00h)

7							0
ETPR 7	ETPR 6	ETPR 5	ETPR 4	ETPR 3	ETPR 2	ETPR 1	ETPR 0

Bit 7:0 = **ETPR[7:0]** *8-bit Extended Transmit Prescaler Register.*

The extended Baud Rate Generator is activated when a value other than 00h is stored in this register. The clock frequency from the 16 divider (see Figure 56) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255).

The extended baud rate generator is not active 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 58.

It uses 6 registers:

– Three control registers: SCICR1, SCICR2 and SCICR3

– Two status registers: the SCISR register and the 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 11.5.10for the definitions of each bit.

11.5.9.1 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.

11.5.9.2 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

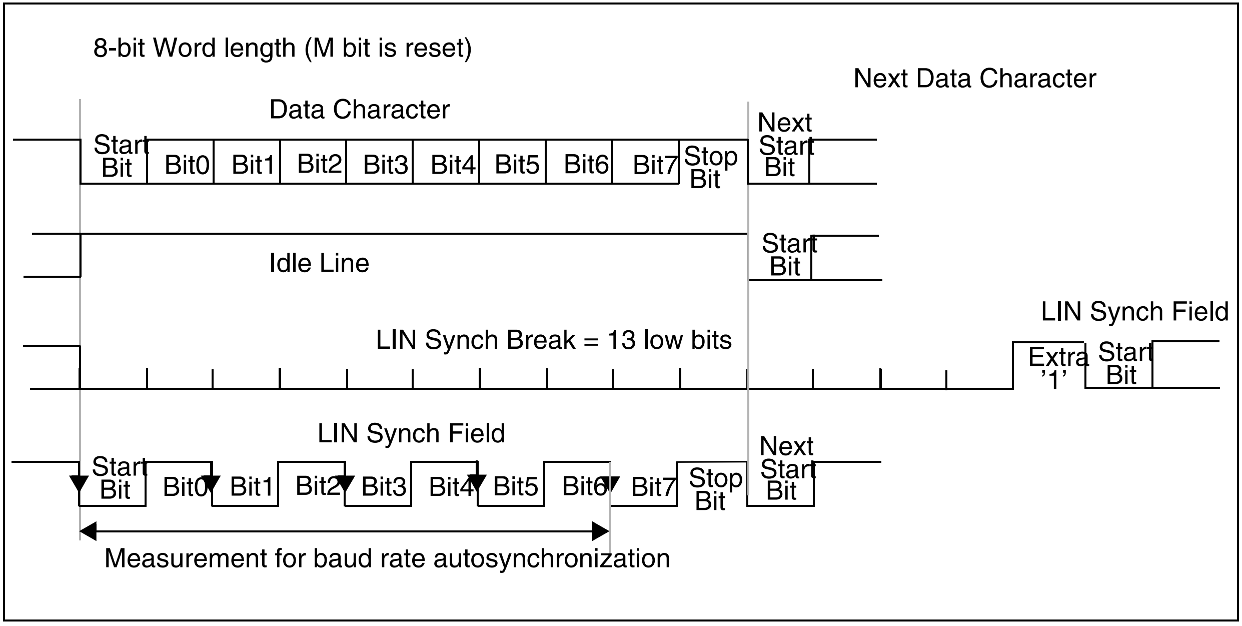
– Load the LIN Synch Field (0x55) in the SCIDR register to request Synch Field transmission

– 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.

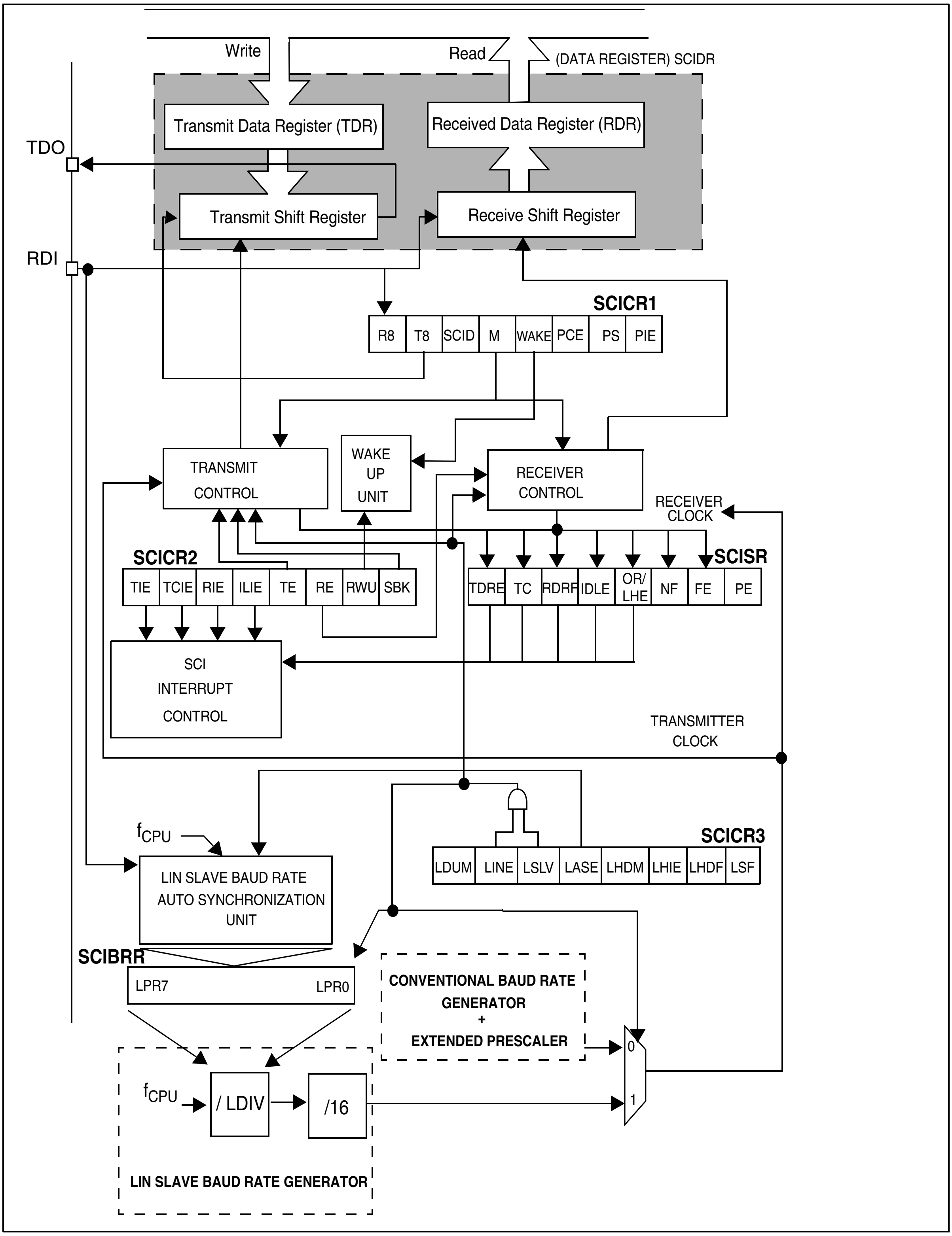
LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (Cont'd)

Figure 57. LIN characters



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

Figure 58. SCI Block Diagram in LIN Slave Mode



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

11.5.9.3 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 semiautomatically (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 Section 11.5.9.5 LIN Baudrate.

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:

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.

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.

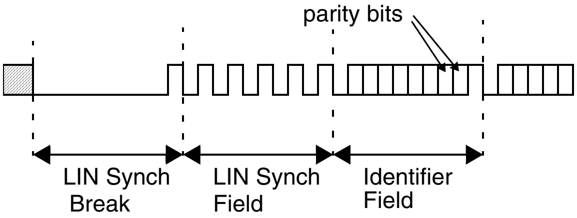
Notes:

During LIN Synch Field measurement, the SCI state machine is switched off: no characters are transferred to the data register.

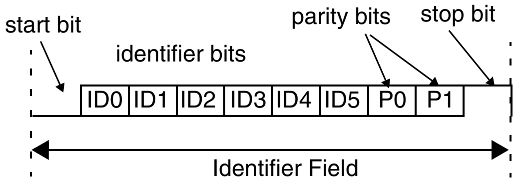
LIN Slave parity

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 recognised as the 3rd received character after a break character (included):



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 by the LIN specification:



$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$
$$P1 = \overline{ID1 \oplus ID3 \oplus ID4 \oplus ID5}$$

M=0

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

11.5.9.4 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 59. (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 checking 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's 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 $15.625\% \leq D < 16.40625\%$, 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.

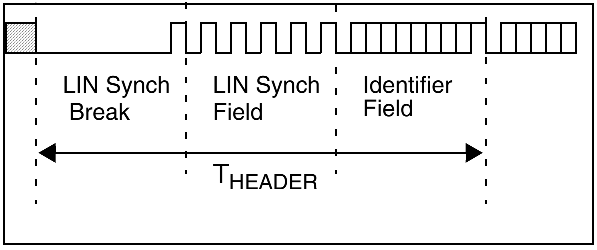
When LHE is set due to this error then the SCI goes into a blocked state (LSF bit is set).

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 59. LIN Header Reception Timeout



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

- A LIN Identifier Field has been received
- An LHE error occurred (other than a timeout error).
- A software reset of LSF bit (transition from high to low) occurred during the analysis of the LIN Synch Field or

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).

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

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 last 49-10 = 39T_{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 doesn't 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:

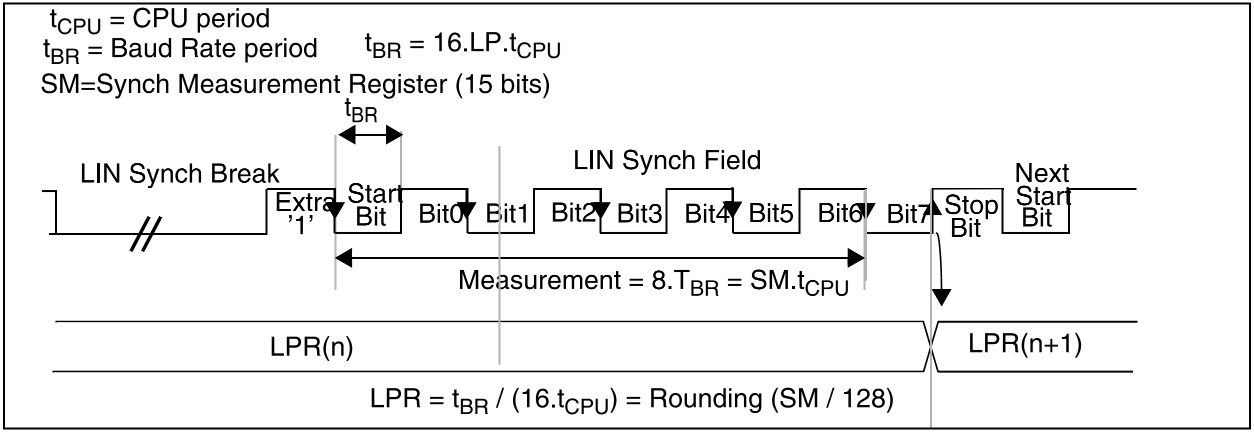
- A valid LIN Break Field (at least 11 dominant bits followed by a recessive bit)

- 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 60. LIN Synch Field Measurement



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

11.5.9.5 LIN Baudrate

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 5 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 60). 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.

11.5.9.6 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:

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

with:

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 is explained in Figure 61 and Figure 62. It depends on the LDUM bit setting (LIN Divider Update Method)

Note:

As explained in Figure 61 and Figure 62, 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.

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

Figure 61. LDIV Read / Write operations when LDUM=0

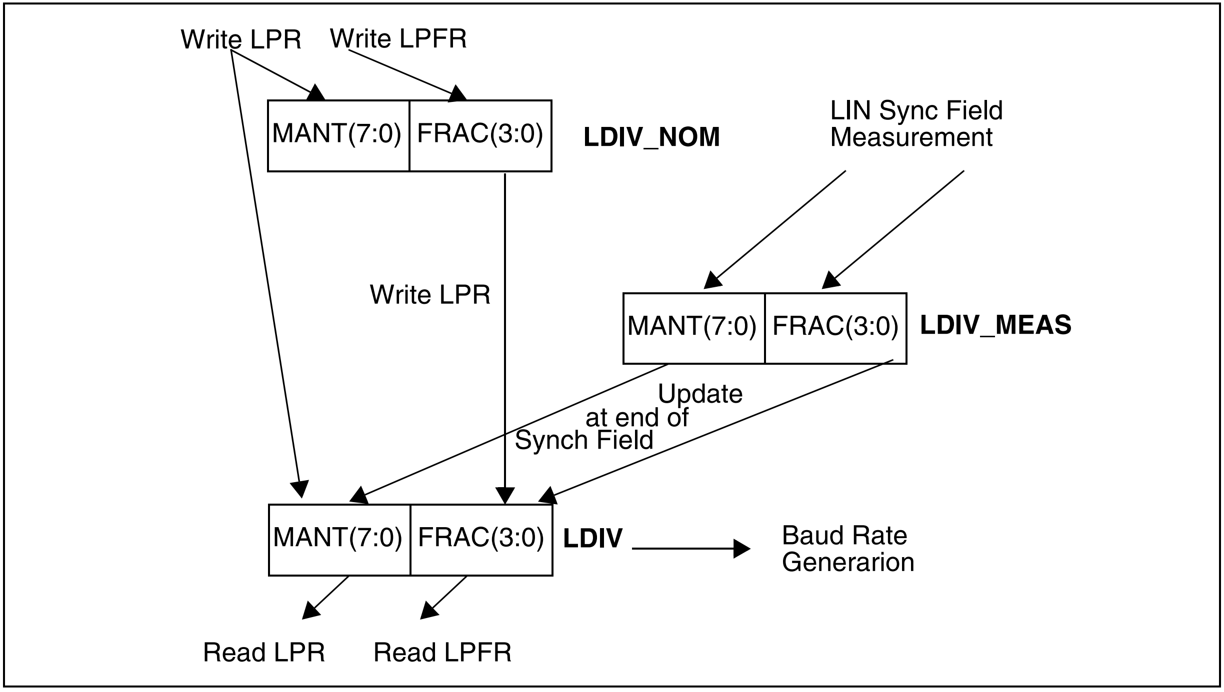
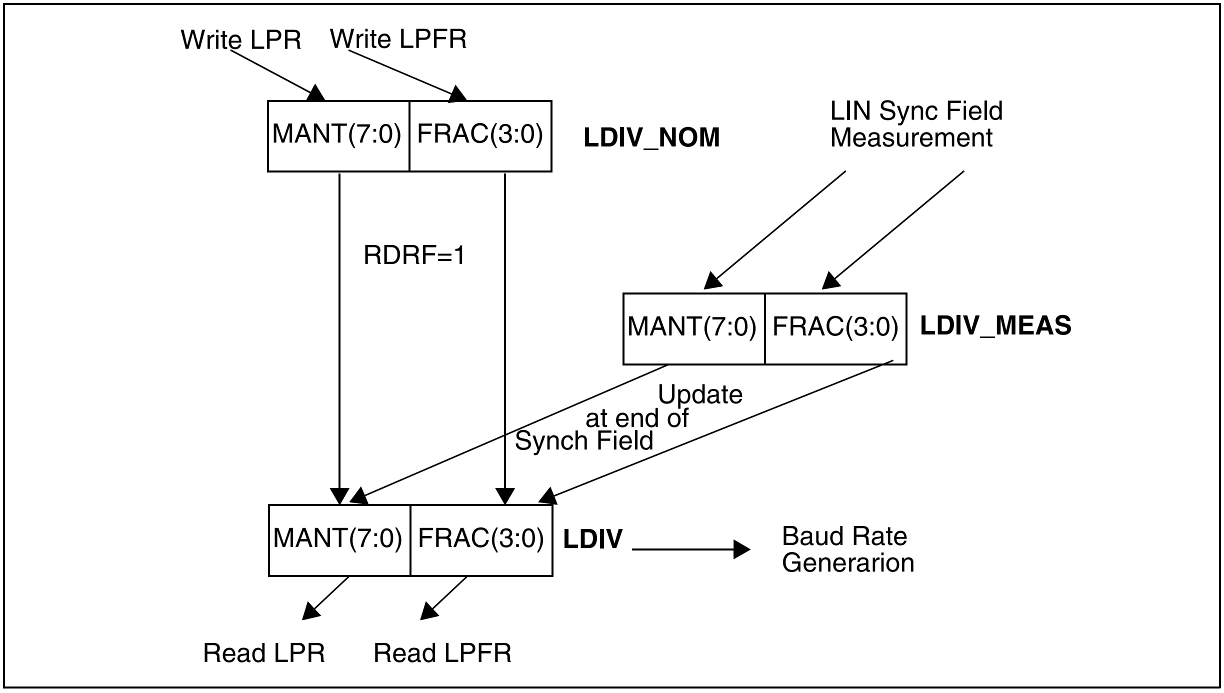


Figure 62. LDIV Read / Write operations when LDUM=1



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

11.5.9.7 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%.

11.5.9.8 Clock Deviation Causes

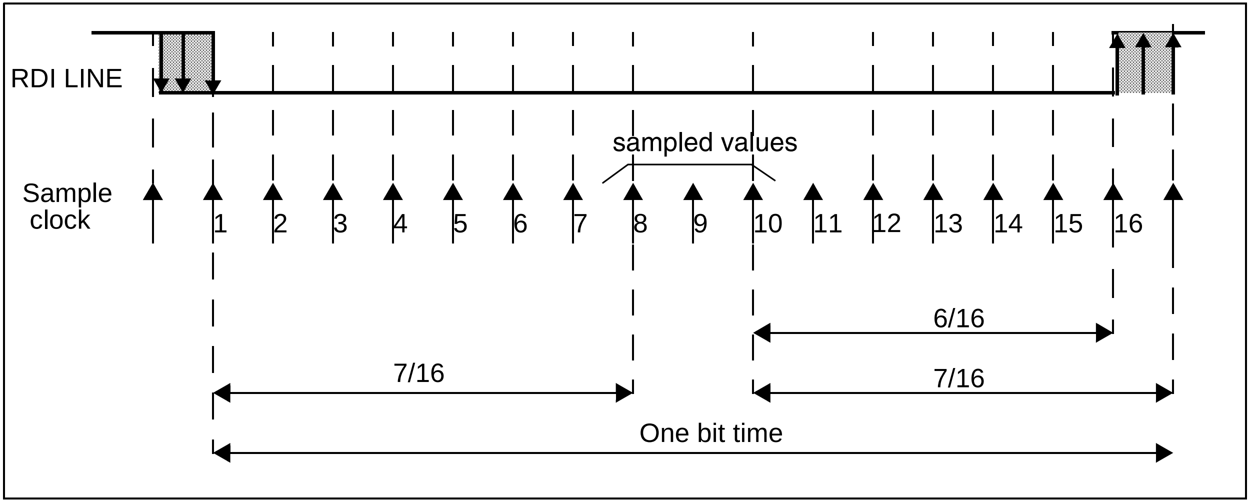
The causes which contribute to the total deviation are:

- 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 quantisation 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 63. Bit Sampling in Reception Mode



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

11.5.9.9 Error due to LIN Synch measurement

The LIN Synch Field is measured over eight bit times.

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

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

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

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

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

11.5.9.10 Error due to Baud Rate Quantisation

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

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

11.5.9.11 Impact of Clock Deviation on Maximum Baud Rate

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

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

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

Example:

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

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

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

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

LIN Slave systems

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

Hot Plugging Feature for LIN Slave Nodes

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

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

11.5.10 LIN Mode Register Description

STATUS REGISTER (SCISR)
Read Only
Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	LHE	NF	FE	PE

Bits 7:4 = Same function as in SCI mode, please refer to Section 11.5.8 SCI Mode Register Description.

Bit 3 = **LHE** *LIN Header Error.*
During LIN Header this bit signals three error types:

– The LIN Synch Field is corrupted and the SCI is blocked in LIN Synch State (LSF bit=1).

– A timeout occurred during LIN Header reception

– An overrun error was detected on one of the header field (see OR bit description in Section 11.5.8 SCI Mode Register Description)).

An interrupt is generated if RIE=1 in the SCICR2 register. If blocked in the LIN Synch State, the LSF bit must first be reset (to exit LIN Synch Field state and then to be able to clear LHE flag). Then it is cleared by the following software sequence : an access to the SCISR register followed by a read to the SCIDR register.
0: No LIN Header error
1: LIN Header error detected

Note:

Apart from the LIN Header this bit signals an Overrun Error as in SCI mode, (see description in Section 11.5.8 SCI Mode Register Description)

Bit 2 = **NF** *Noise flag*

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

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

Bit 1 = Bit 1 = **FE** *Framing error.*
In LIN slave mode, this bit is set only when a real framing error is detected (if the stop bit is dominant (0) and at least one of the other bits is recessive (1). It is not set when a break occurs, the LHDF bit is used instead as a break flag (if the LHDM bit=0). It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Framing error
1: Framing error detected

Bit 0 = **PE** *Parity error.*
This bit is set by hardware when a LIN parity error occurs (if the PCE bit is set) in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE=1 in the SCICR1 register.
0: No LIN parity error
1: LIN Parity error detected

CONTROL REGISTER 1 (SCICR1)
Read/Write
Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE

Bits 7:3 = Same function as in SCI mode, please refer to Section 11.5.8 SCI Mode Register Description.

Bit 2 = **PCE** *Parity control enable.*
This bit is set and cleared by software. It selects the hardware parity control for LIN identifier parity check.
0: Parity control disabled
1: Parity control enabled

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

Bit 1 = Reserved

Bit 0 = Same function as in SCI mode, please refer to Section 11.5.8 SCI Mode Register Description.

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

CONTROL REGISTER 2 (SCICR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

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

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

Notes:

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

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

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

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

CONTROL REGISTER 3 (SCICR3)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
LDUM	LINE	LSLV	LASE	LHDM	LHIE	LHDF	LSF

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

Notes:

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

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

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

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

The LIN Master configuration enables:

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

The LIN Slave configuration enables:

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

– Management of LIN Headers.

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

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

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

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

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

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

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

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

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

Bit 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.

Bit 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.

Notes: The header detection method depends on the LHDM bit:

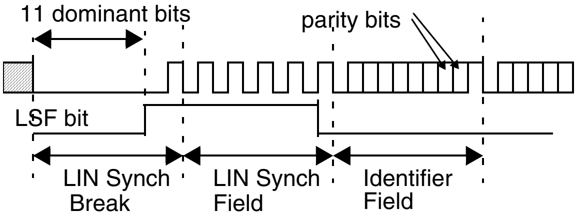
- If LHDM=0, a header is detected as a LIN Synch Break.
- If LHDM=1, a header is detected as a LIN Identifier, meaning that a LIN Synch Break Field + a LIN Synch Field + a LIN Identifier Field have been consecutively received.

Bit 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 64). 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 64. 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)

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

7							0
LPR7	LPR6	LPR5	LPR4	LPR3	LPR2	LPR1	LPR0

LPR[7:0] *LIN Prescaler (mantissa of LDIV)*

These 8 bits define the value of the mantissa of the LIN Divider (LDIV):

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.

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

LIN PRESCALER FRACTION REGISTER (LPFR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	LPFR 3	LPFR 2	LPFR 1	LPFR 0

Bits 7:4= Reserved.

Bits 3:0 = **LPFR[3:0]** *Fraction of LDIV*

These 4 bits define the fraction of the LIN Divider (LDIV):

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

1. When initializing LDIV, the LPFR register must be written first. Then, the write to the LPR register will effectively update LDIV and so the clock generation.

2. In LIN Slave mode, if the LPR[7:0] register is equal to 00h, the transceiver and receiver input clocks are switched off.

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

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

LIN HEADER LENGTH REGISTER (LHLR)

Read Only
Reset Value: 0000 0000 (00h).

7							0
LHL7	LHL6	LHL5	LHL4	LHL3	LHL2	LHL1	LHL0

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.

Bit 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 than 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}). The coding of this value is explained below:

LHL Coding:

$T_{HEADER\_MAX} = 57$

LHL(7:2) represents the mantissa of (57 - T_{HEADER})

LHL(1:0) represents the fraction (57 - T_{HEADER})

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

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

Example 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

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

Table 20. LINSCI1 Register Map and Reset Values

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

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 7 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 7 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 65.

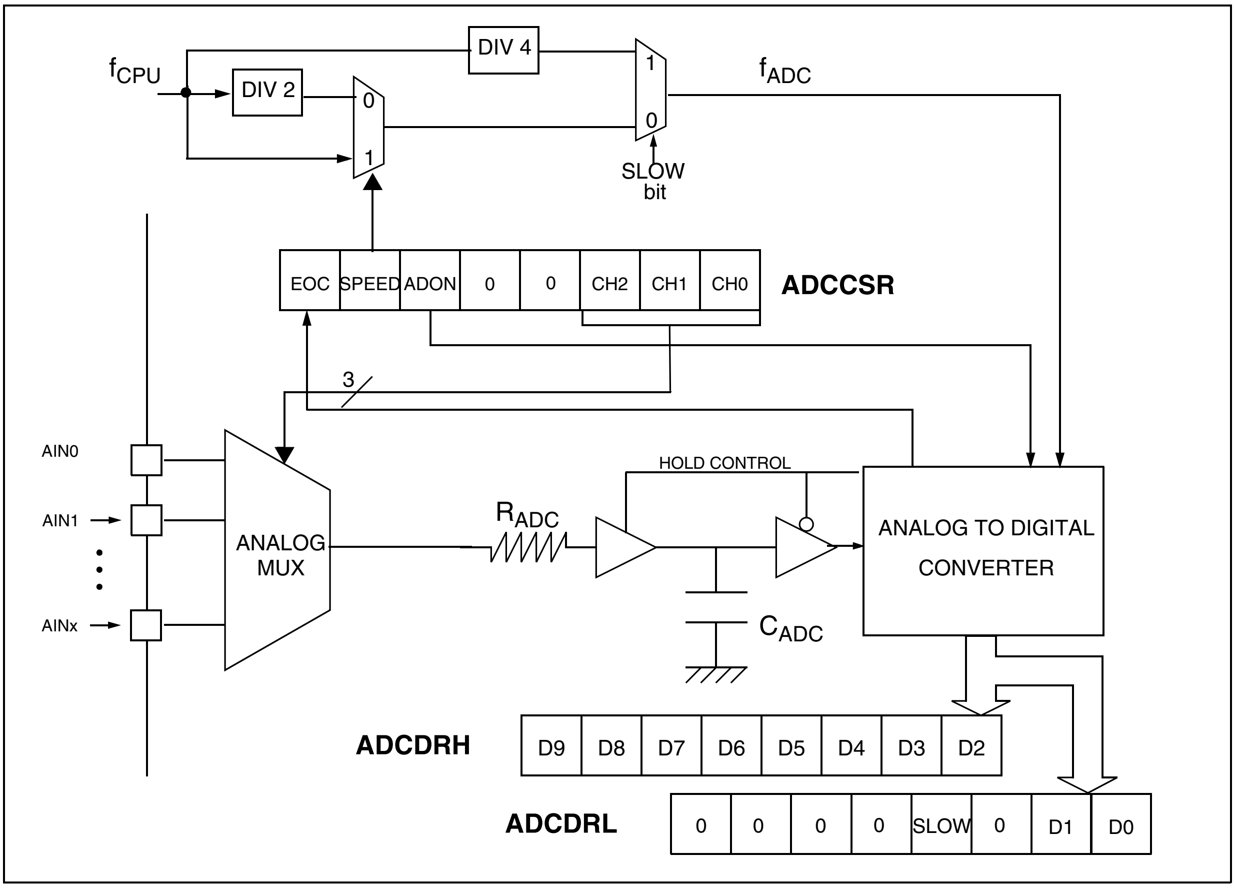
11.6.3 Functional Description

11.6.3.1 Analog Power Supply

V_{DDA} and V_{SSA} are the high and low level reference voltage pins. In some devices (refer to device pin out 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 65. ADC Block Diagram



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

11.6.3.2 Digital A/D Conversion Result

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

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 Electrical Characteristics Section.

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

11.6.3.3 A/D Conversion

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the «I/O ports» chapter. 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 CS[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 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.

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} (see Electrical Characteristics) before accurate conversions can be performed.

11.6.5 Interrupts

None.

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

11.6.6 Register Description

CONTROL/STATUS REGISTER (ADCCSR)

Read/Write (Except bit 7 read only)

Reset Value: 0000 0000 (00h)

7							0
EOC	SPEED	ADON	0	0	CH2	CH1	CH0

Bit 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

Bit 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 the table in the SLOW bit description.

Bit 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

Bit 4:3 = **Reserved.** Must be kept cleared.

Bit 2:0 = **CH[2:0]** *Channel Selection*
These bits are set and cleared by software. They select the analog input to convert.

Channel Pin*	CH2	CH1	CH0
AIN0	0	0	0
AIN1	0	0	1
AIN2	0	1	0
AIN3	0	1	1
AIN4	1	0	0
AIN5	1	0	1
AIN6	1	1	0

*The number of channels is device dependent. Refer to the device pinout description.

DATA REGISTER HIGH (ADCDRH)

Read Only

Reset Value: 0000 0000 (00h)

7							0
D9	D8	D7	D6	D5	D4	D3	D2

Bit 7:0 = **D[9:2]** *MSB of Analog Converted Value*

CONTROL AND DATA REGISTER LOW (ADCDRL)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	SLOW	0	D1	D0

Bit 7:5 = Reserved. Forced by hardware to 0.

Bit 4 = Reserved. Forced by hardware to 0.

Bit 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 on the table below.

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

Bit 2 = Reserved. Forced by hardware to 0.

Bit 1:0 = **D[1:0]** *LSB of Analog Converted Value*

Table 21. 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 0	D0 0

12 INSTRUCTION SET

12.1 ST7 ADDRESSING MODES

The ST7 Core features 17 different addressing modes which can be classified in 7 main 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 ST7 Instruction 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 22. ST7 Addressing Mode Overview

Mode			Syntax	Destination/ Source	Pointer Address (Hex.)	Pointer Size (Hex.)	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

Note 1. At the time the instruction is executed, the Program Counter (PC) points to the instruction following JRxx.

ST7 ADDRESSING MODES (Cont'd)

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.

Inherent Instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait For Interrupt (Low Power Mode)
HALT	Halt Oscillator (Lowest Power Mode)
RET	Sub-routine Return
IRET	Interrupt Sub-routine 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 two bytes, the first byte contains the opcode, the second byte contains the operand value.

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 submodes:

Direct (short)

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

Direct (long)

The address is a word, thus allowing 64 Kbyte addressing space, but requires 2 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 2 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.

ST7 ADDRESSING MODES (Cont'd)

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 23. 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

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.

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 the following table:

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

Using a pre-byte

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

PDY 90	Replace an X based instruction using immediate, direct, indexed, or inherent addressing mode by a Y one.
PIX 92	Replace an instruction using direct, direct bit, or direct relative addressing mode to an instruction using the corresponding indirect addressing mode. It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.
PIY 91	Replace an instruction using X indirect indexed addressing mode by a Y one.

12.2.1 Illegal Opcode Reset

In order to provide enhanced robustness to the device against unexpected behaviour, 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.

INSTRUCTION GROUPS (Cont'd)

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 >=							
JRUGT	Jump if (C + Z = 0)	Unsigned >							

INSTRUCTION GROUPS (Cont'd)

Mnemo	Description	Function/Example	Dst	Src	H	I	N	Z	C
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}=5V (for the $4.5V \le V_{DD} \le 5.5V$ voltage range) and V_{DD}=3.3V (for the $3V \le V_{DD} \le 4V$ voltage range). They are given only as design guidelines and are not tested.

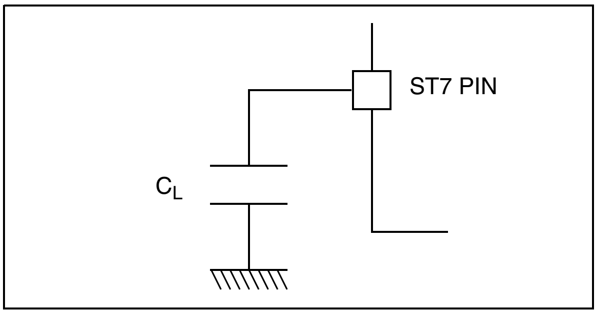
13.1.3 Typical curves

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

13.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in Figure 66.

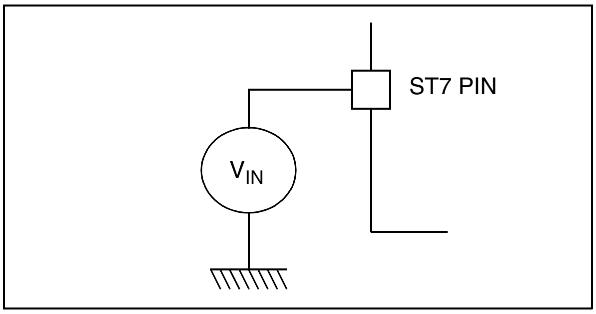
Figure 66. Pin loading conditions



13.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 67.

Figure 67. 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

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 on page 143	

13.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) 3)	150	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink) 3)	150	
I_{IO}	Output current sunk by any standard I/O and control pin	25	
	Output current sunk by any high sink I/O pin	50	
	Output current source by any I/Os and control pin	-25	
$I_{INJ(PIN)}$ 2) & 4)	Injected current on $\overline{\text{RESET}}$ pin	±5	
	Injected current on OSC1 and OSC2 pins	±5	
	Injected current on PB0 and PB1 pins 5)	+5	
	Injected current on any other pin 6)	±5	
$\Sigma I_{INJ(PIN)}$ 2)	Total injected current (sum of all I/O and control pins) 6)	±20	

13.2.3 Thermal Characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature (see section 14.2 on page 156)		

Notes:

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 has to be done through a pull-up or pull-down resistor (typical: 4.7kΩ for $\overline{\text{RESET}}$, 10kΩ 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
3. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
4. 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 1.6mA. In addition, it is recommended to inject the current as far as possible from the analog input pins.
5. No negative current injection allowed on PB0 and PB1 pins.
6. 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 characterisation with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

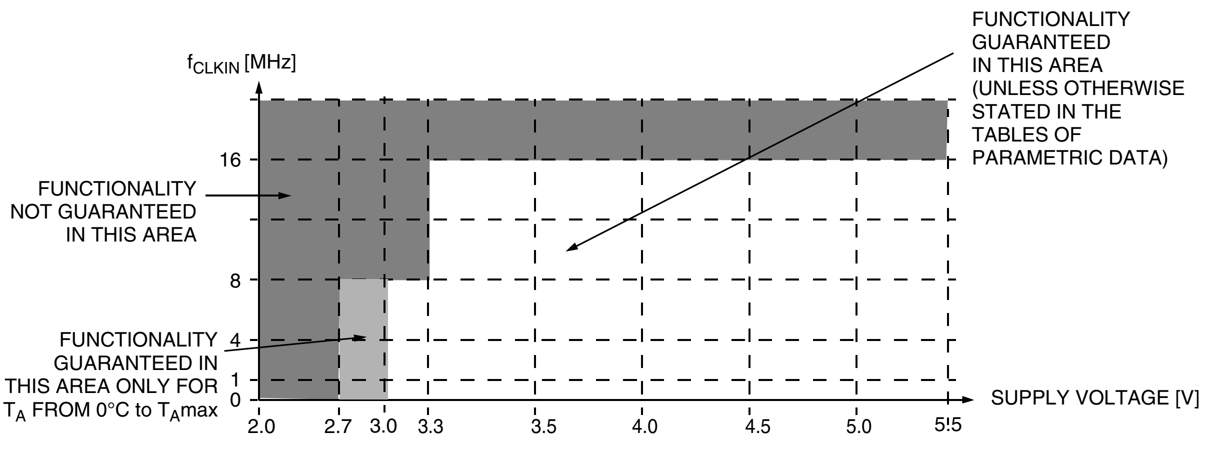
13.3 OPERATING CONDITIONS

13.3.1 General Operating Conditions: Suffix 6 Devices

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

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	Supply voltage	f_{OSC} = 8 MHz. max., T_A = 0 to 85°C	2.7	5.5	V
		f_{OSC} = 8 MHz. max., T_A = - 40 to 85°C	3.0	5.5	
		f_{OSC} = 16 MHz. max.	3.3	5.5	
f_{CLKIN}	External clock frequency on CLKIN pin	V_{DD}≥3.3V	up to 16		MHz
		V_{DD}≥3.0V	up to 8		

Figure 68. f_{CLKIN} Maximum Operating Frequency Versus V_{DD} Supply Voltage



OPERATING CONDITIONS (Cont'd)

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

13.3.1.1 Devices with '"6" order code suffix (tested for T_A = -40 to +85°C) @ V_{DD} = 4.5 to 5.5V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal RC oscillator frequency	RCCR = FF (reset value), T_A=25°C,V_{DD}=5V		1600		kHz
		RCCR = RCCR0[1],T_A=25°C,V_{DD}=5V	995	1000	1005	
ACC_{RC}	Accuracy of Internal RC oscillator with RCCR=RCCR0[1]	T_A=25°C, V_{DD}=4.5 to 5.5V	-1		+1	%
		T_A=-40 to +85°C, V_{DD}=4.5 to 5.5V	-2[2]		+5[2]	%
		T_A=0 to +85°C, V_{DD}=4.5 to 5.5V	-2		+2	%
$I_{DD(RC)}$	RC oscillator current consumption	T_A=25°C,V_{DD}=5V		600[3][4]		µA
$t_{su(RC)}$	RC oscillator setup time	T_A=25°C,V_{DD}=5V			10[1]	µs
f_{PLL}	x8 PLL input clock		1[3]			MHz
t_{LOCK}	PLL Lock time[7]			2		ms
t_{STAB}	PLL Stabilization time[7]			4		ms
ACC_{PLL}	x8 PLL Accuracy	f_{RC} = 1MHz@T_A=25°C,V_{DD}=4.5 to 5.5V		0.1[6]		%
		f_{RC} = 1MHz@T_A=-40 to +85°C,V_{DD}=5V		0.1[6]		%
$t_{w(JIT)}$	PLL jitter period	f_{RC} = 1MHz		8[5]		kHz
JIT_{PLL}	PLL jitter ($\Delta f_{CPU}/f_{CPU}$)			1[5]		%
$I_{DD(PLL)}$	PLL current consumption	T_A=25°C		550[3]		µA

Notes:

1. RCCR0 is a factory-calibrated setting for 1000kHz with ±0.2 accuracy @ T_A =25°C, V_{DD}=5V. See "INTERNAL RC OSCILLATOR ADJUSTMENT" on page 22
2. Min value is obtained for hot temperature and max value is obtained for cold temperature.
3. Data based on characterization results, not tested in production
4. Measurement made with RC calibrated at 1MHz.
5. Guaranteed by design.
6. Averaged over a 4ms period. After the LOCKED bit is set, a period of t_{STAB} is required to reach ACC_{PLL} accuracy.
7. After the LOCKED bit is set ACC_{PLL} is max. 10% until t_{STAB} has elapsed. See Figure 11 on page 23.

OPERATING CONDITIONS (Cont'd)

Figure 69. Typical accuracy with RCCR=RCCR0 vs V_{DD}= 4.5 to 5.5V and Temperature

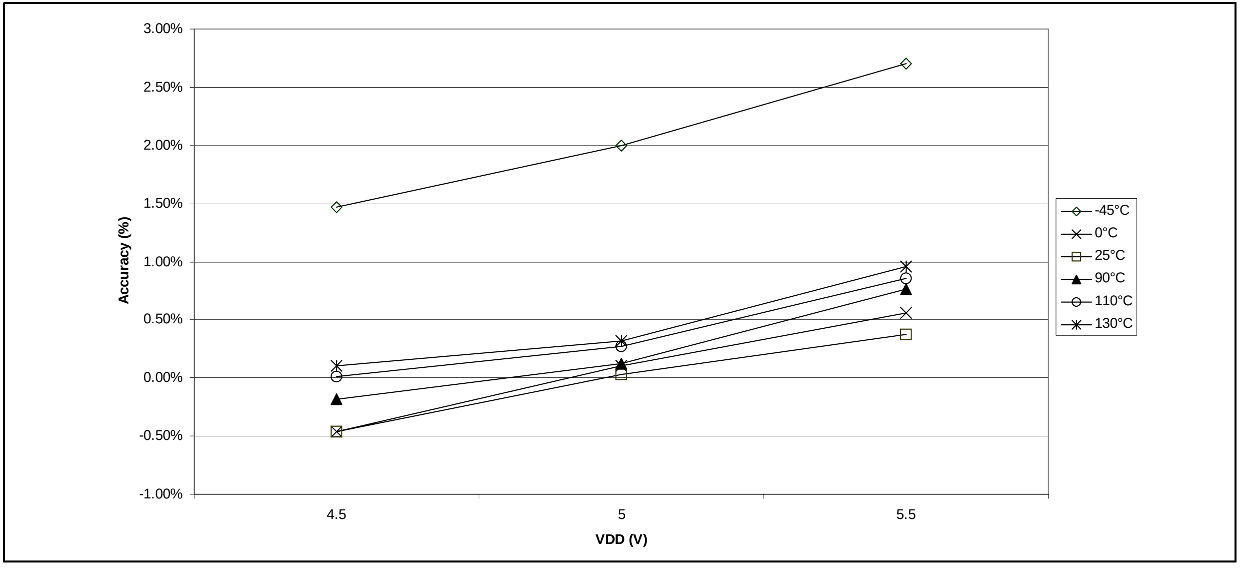
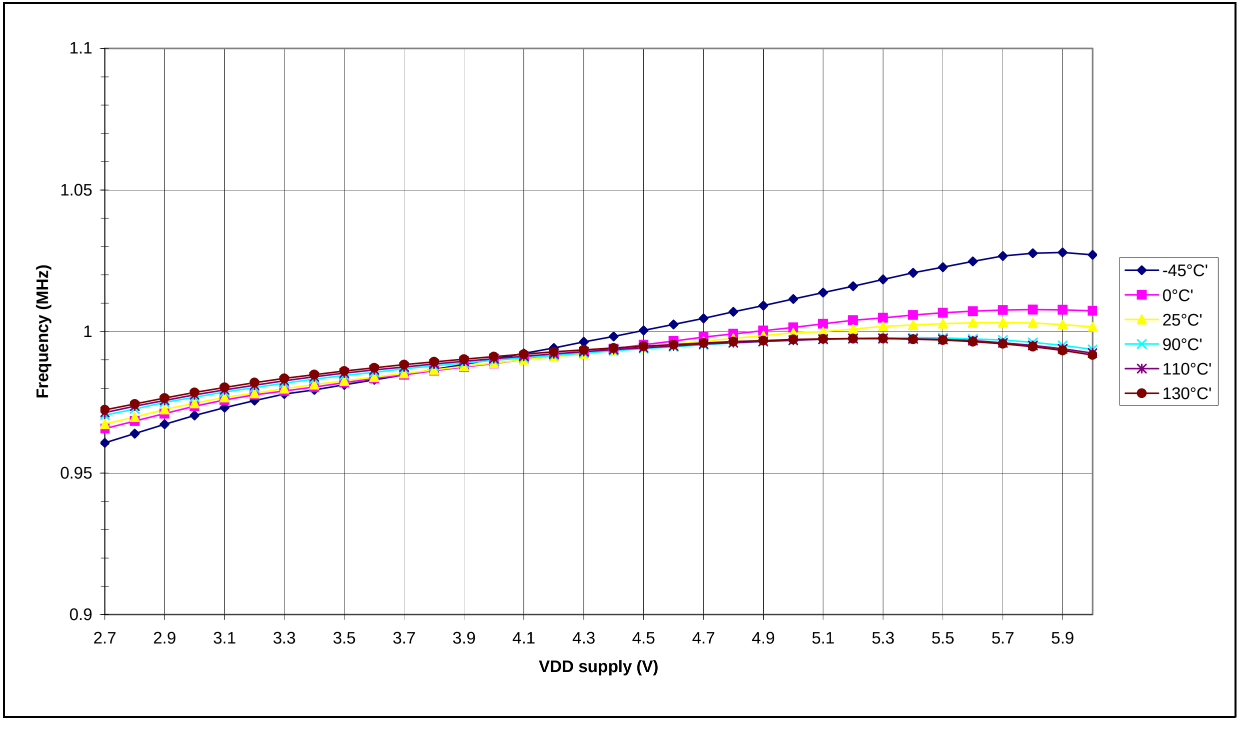


Figure 70. Typical RCCR0 vs V_{DD} and Temperature



OPERATING CONDITIONS (Cont'd)

13.3.1.2 Devices with '"6" order code suffix (tested for T_A = -40 to +85°C) @ V_{DD} = 3.0 to 3.6V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal RC oscillator frequency	RCCR = FF (reset value), T_A=25°C, V_{DD}= 3.3V		1600		kHz
		RCCR=RCCR1 1) ,T_A=25°C,V_{DD}= 3.3V	995	1000	1005	
ACC_{RC}	Accuracy of Internal RC oscillator when calibrated with RCCR=RCCR1 1)2)	T_A=25°C, V_{DD}=3.0 to 3.6V	-1		+1	%
		T_A=-40 to +85°C, V_{DD}=3.0 to 3.6V	-3		+3	
		T_A=0 to +85°C, V_{DD}=3.0 to 3.6V	-3		+3	
$I_{DD(RC)}$	RC oscillator current consumption	T_A=25°C,V_{DD}=3.3V		500 2)3)		µA
$t_{su(RC)}$	RC oscillator setup time	T_A=25°C,V_{DD}=3.3V			10 1)	µs
f_{PLL}	x4 PLL input clock		1 2)			MHz
t_{LOCK}	PLL Lock time 6)			2		ms
t_{STAB}	PLL Stabilization time 6)			4		ms
ACC_{PLL}	x4 PLL Accuracy	f_{RC} = 1MHz@T_A=25°C, V_{DD}=2.7 to 3.3V		0.1 5)		%
		f_{RC} = 1MHz@T_A=-40 to +85°C, V_{DD}= 3.3V		0.1 5)		%
$t_{w(JIT)}$	PLL jitter period	f_{RC} = 1MHz		8 4)		kHz
JIT_{PLL}	PLL jitter ($\Delta f_{CPU}/f_{CPU}$)			1 4)		%
$I_{DD(PLL)}$	PLL current consumption	T_A=25°C		450 2)		µA

Notes:

1.RCCR1 is a factory-calibrated setting for 1000kHz with ±0.2 accuracy @ T_A =25°C, V_{DD}=3.3V. See "INTERNAL RC OSCILLATOR ADJUSTMENT" on page 22.

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

3. Measurement made with RC calibrated at 1MHz.

4. Guaranteed by design.

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

6.After the LOCKED bit is set ACC_{PLL} is max. 10% until t_{STAB} has elapsed. See Figure 11 on page 23.

OPERATING CONDITIONS (Cont'd)

Figure 71. Typical accuracy with RCCR=RCCR1 vs V_{DD}= 3-3.6V and Temperature

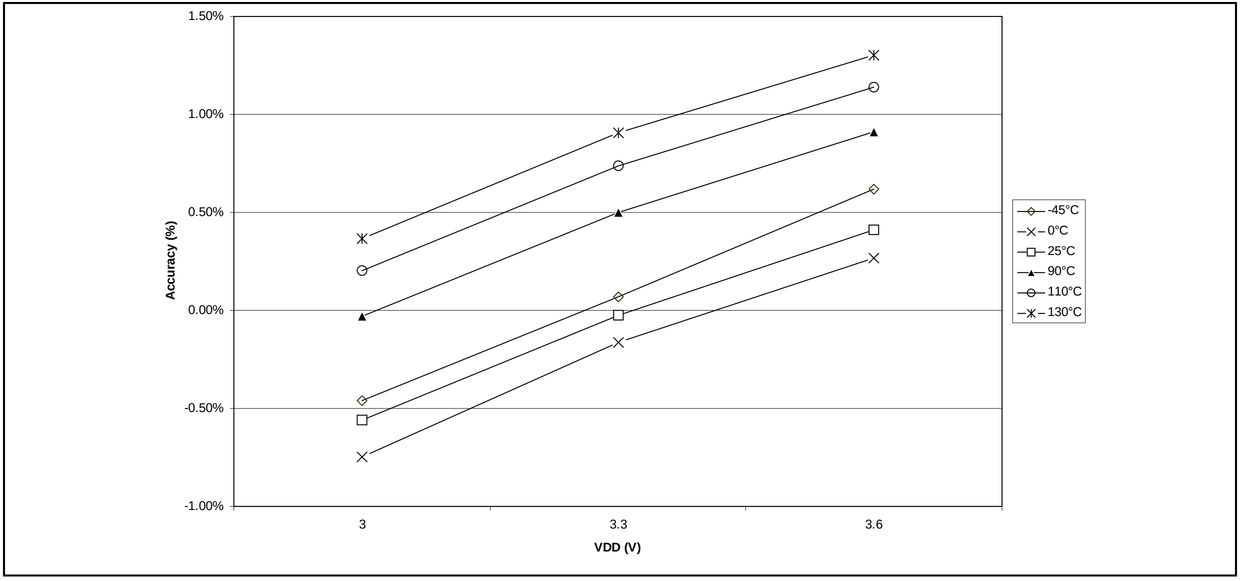
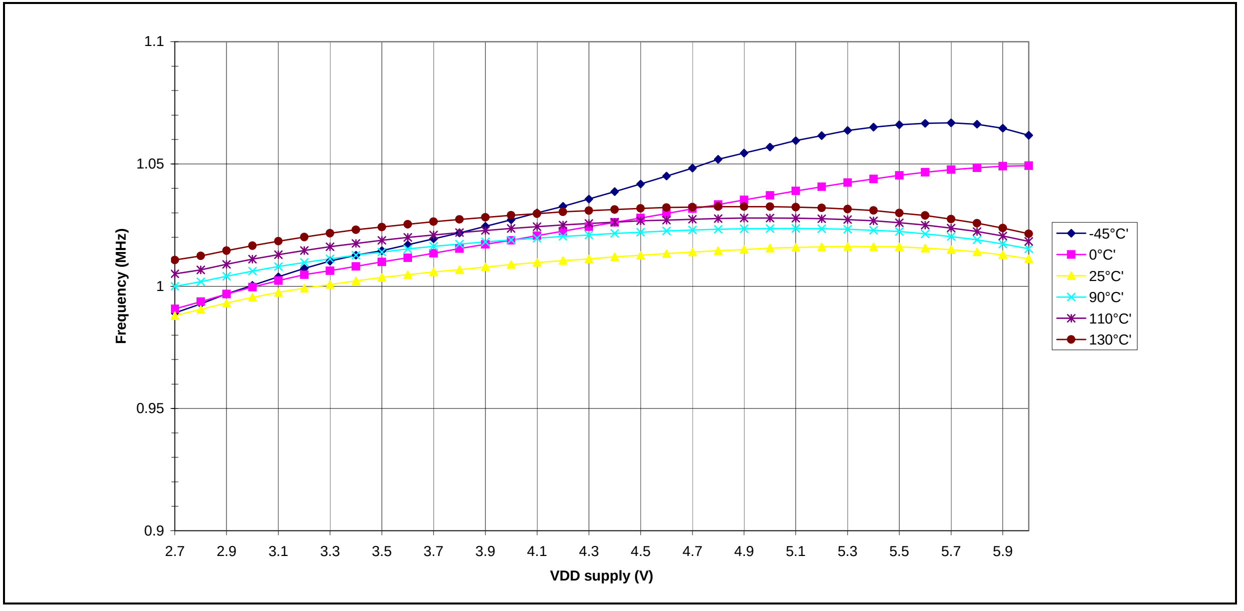


Figure 72. Typical RCCR1 vs V_{DD} and Temperature



OPERATING CONDITIONS (Cont'd)

Figure 73. PLL $\Delta f_{CPU}/f_{CPU}$ versus time

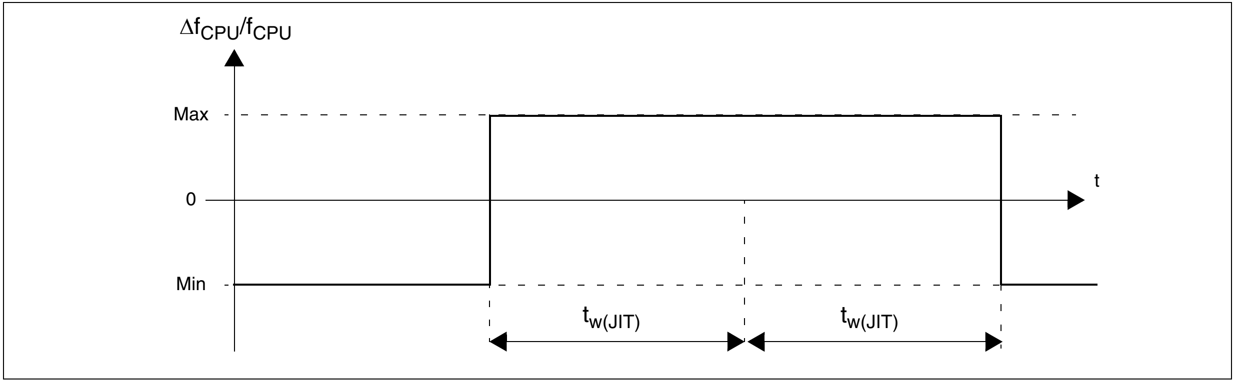
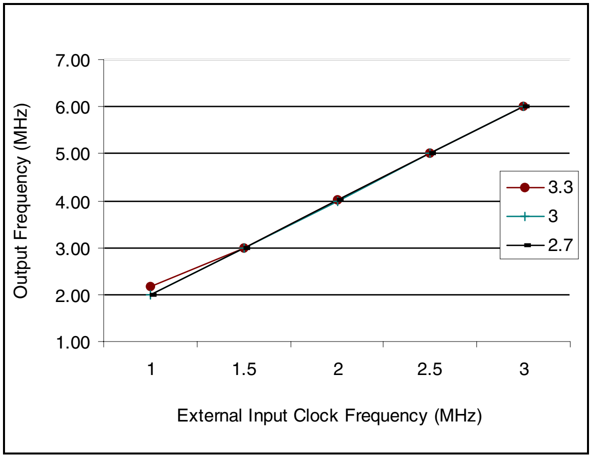
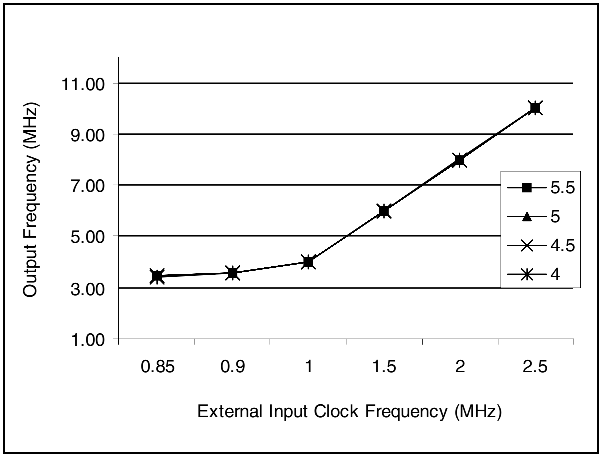


Figure 74. PLLx4 Output vs CLKIN frequency



Note: $f_{OSC} = f_{CLKIN}/2*PLL4$

Figure 75. PLLx8 Output vs CLKIN frequency



Note: $f_{OSC} = f_{CLKIN}/2*PLL8$

13.3.2 Operating Conditions with Low Voltage Detector (LVD)

T_A = -40 to 85°C, unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(LVD)}$	Reset release threshold (V_{DD} rise)	High Threshold Med. Threshold Low Threshold	3.60 1) 3.05 1) 2.45 1)	4.15 3.45 2.85	4.60 3.90 3.20	V
$V_{IT-(LVD)}$	Reset generation threshold (V_{DD} fall)	High Threshold Med. Threshold Low Threshold	3.40 2.90 2.30	3.95 3.30 2.70	4.35 1) 3.70 1) 3.00 1)	
V_{hys}	LVD voltage threshold hysteresis	$V_{IT+(LVD)}-V_{IT-(LVD)}$		200		mV
Vt_{POR}	V_{DD} rise time rate 2)		20 1)		100000 1)	µs/V
$t_{g(VDD)}$	Filtered glitch delay on V_{DD}	Not detected by the LVD			150	ns
$I_{DD(LVD)}$	LVD/AVD current consumption			220		µA

Notes:

1. Not tested in production.
2. 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.

13.3.3 Auxiliary Voltage Detector (AVD) Thresholds,

T_A = -40 to 85°C, unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(AVD)}$	1=>0 AVDF flag toggle threshold (V_{DD} rise)	High Threshold Med. Threshold Low Threshold	3.90 1) 3.45 1) 2.90 1)	4.45 3.90 3.30	4.85 4.30 3.65	V
$V_{IT-(AVD)}$	0=>1 AVDF flag toggle threshold (V_{DD} fall)	High Threshold Med. Threshold Low Threshold	3.85 3.35 2.75	4.40 3.85 3.15	4.80 1) 4.20 1) 3.50 1)	
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

Note:

1. 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).

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, section 13.3.1 on page 131	2.7		5.5	V
$V_{DD(x4PLL)}$	x4 PLL operating voltage		2.7		3.3	
$V_{DD(x8PLL)}$	x8 PLL operating voltage		3.3		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 get 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 +85°C unless otherwise specified

Symbol	Parameter	Conditions		Typ	Max	Unit
I_{DD}	Supply current in RUN mode	V_{DD}=5.5V	f_{CPU}=8MHz [1]	6.0	9.0	mA
			f_{CPU}=4MHz	2.6	5.6	
			f_{CPU}=1MHz	0.8	2.5	
	Supply current in WAIT mode		f_{CPU}=8MHz [2]	2.4	4.0	
	Supply current in SLOW mode		f_{CPU}=250kHz [3]	0.7	1.1	
	Supply current in SLOW WAIT mode		f_{CPU}=250kHz [4]	0.6	1.0	
	Supply current in HALT mode[5]		$-40°C \leq T_A \leq +85°C$	0.5	10	μA
	Supply current in AWUFH mode [6][7]		T_A= +25°C	20	50	

Notes:

1. 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.
2. 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. 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.
4. 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.
5. 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.
6. 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.
7. This consumption refers to the Halt period only and not the associated run period which is software dependent.

Figure 76. Typical I_{DD} in RUN vs. f_{CPU}

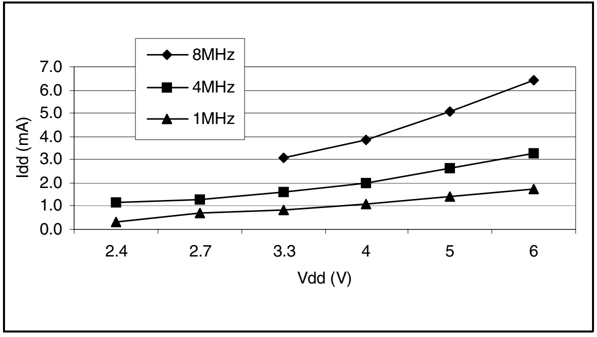
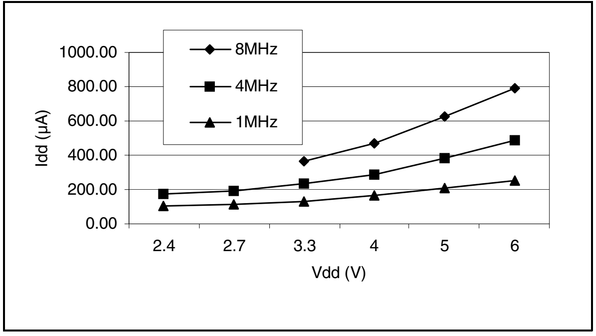


Figure 77. Typical I_{DD} in SLOW vs. f_{CPU}



SUPPLY CURRENT CHARACTERISTICS (Cont'd)

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

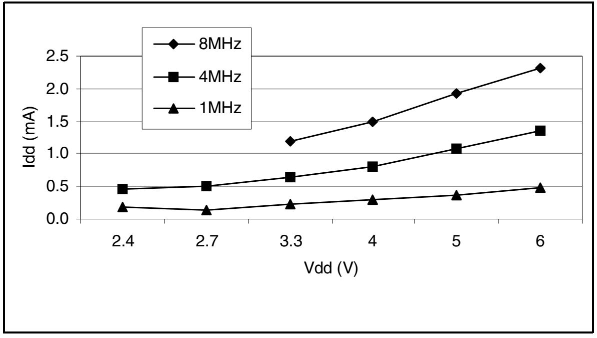


Figure 80. Typical I_{DD} vs. Temperature at V_{DD} = 5V and f_{OSC} = 16MHz

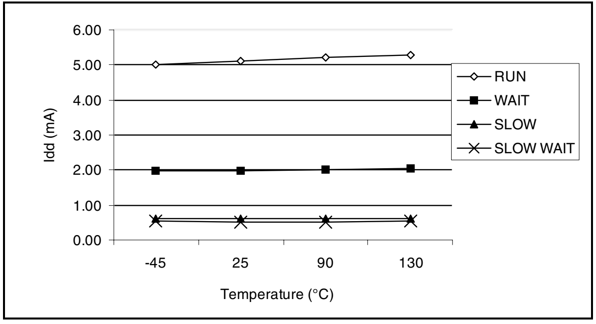


Figure 79. Typical I_{DD} in SLOW-WAIT vs. f_{CPU}

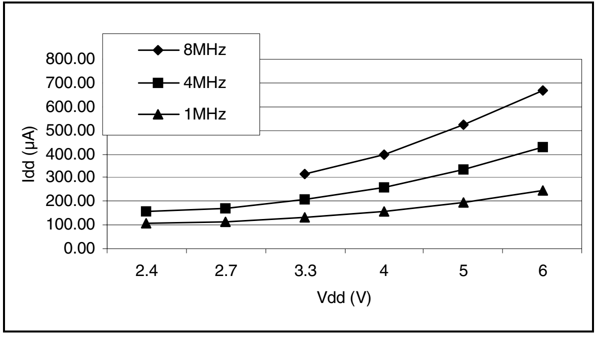
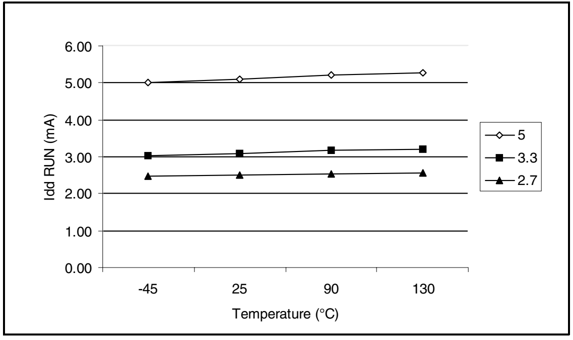


Figure 81. Typical I_{DD} vs. Temperature and V_{DD} at f_{OSC} = 16MHz



13.4.2 On-chip peripherals

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

1. Data based on a differential I_{DD} measurement between reset configuration (timer stopped) and a timer running in PWM mode at f_{cpu}=8MHz.

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

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

Notes:

1. Guaranteed by Design. Not tested in production.
2. Data based on typical application software.
3. Time measured between interrupt event and interrupt vector fetch. $Dt_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.

13.6 MEMORY CHARACTERISTICS

T_A = -40°C to 85°C, unless otherwise specified

13.6.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode 1)	HALT mode (or RESET)	1.6			V

13.6.2 FLASH Program 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 on page 131	2.7		5.5	V
t_{prog}	Programming time for 1~32 bytes 2)	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}	Data retention 4)	T_A=+55°C3)	20			years
N_{RW}	Write erase cycles	T_A=+25°C	10K			cycles
I_{DD}	Supply current	Read / Write / Erase modes f_{CPU} = 8MHz, V_{DD} = 5.5V			2.6 6)	mA
		No Read/No Write Mode			100	µA
		Power down mode / HALT		0	0.1	µA

13.6.3 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 on page 131	2.7		5.5	V
t_{prog}	Programming time for 1~32 bytes	T_A=–40 to +85°C		5	10	ms
t_{ret}	Data retention 4)	T_A=+55°C 3)	20			years
N_{RW}	Write erase cycles	T_A=+25°C	300K			cycles

Notes:

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). Guaranteed by construction, not tested in production.
2. Up to 32 bytes can be programmed at a time.
3. The data retention time increases when the T_A decreases.
4. Data based on reliability test results and monitored in production.
5. Data based on characterization results, not tested in production.
6. Guaranteed by Design. Not tested in production.

13.7 EMC CHARACTERISTICS

Susceptibility tests are performed on a sample basis during product characterization.

13.7.1 Functional EMS (Electro Magnetic Susceptibility)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electro magnetic events until a failure occurs (indicated by the LEDs).

- **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.

13.7.1.1 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 behaviour is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Symbol	Parameter	Conditions	Level/ Class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz conforms to IEC 1000-4-2	3B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100pF on V_{DD} and V_{DD} pins to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz conforms to IEC 1000-4-4	3B

13.7.2 Electro Magnetic 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.

Symbol	Parameter	Conditions	Monitored Frequency Band	Max vs. [f_{OSC}/f_{CPU}]		Unit
				8/4MHz	16/8MHz	
S_{EMI}	Peak level	V_{DD}=5V, T_A=+25°C, SO20 package, conforming to SAE J 1752/3	0.1MHz to 30MHz	16	17	dBμV
			30MHz to 130MHz	20	25	
			130MHz to 1GHz	15	16	
			SAE EMI Level	3	3.5	-

Notes:

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

EMC CHARACTERISTICS (Cont'd)

13.7.3 Absolute Maximum Ratings (Electrical Sensitivity)

Based on three different tests (ESD, LU and DLU) 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.

13.7.3.1 Electro-Static Discharge (ESD)

Electro-Static 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: Human Body Model and Machine Model. This test conforms to the JESD22-A114A/A115A standard.

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

Notes:

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

13.7.3.2 Static and Dynamic Latch-Up

- **LU**: 3 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 the application note AN1181.
- **DLU**: Electro-Static Discharges (one positive then one negative test) are applied to each pin of 3 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

Symbol	Parameter	Conditions	Class [1]
LU	Static latch-up class	T_A=+25°C	A
DLU	Dynamic latch-up class	V_{DD}=5.5V, f_{OSC}=4MHz, T_A=+25°C	A

Notes:

1. Class description: A Class is an STMicroelectronics internal specification. All its limits are higher than the JEDEC specifications, that means when a device belongs to Class A it exceeds the JEDEC standard. B Class 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 unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IL}	Input low level voltage		V_{SS} - 0.3		0.3xV_{DD}	V
V_{IH}	Input high level voltage		0.7xV_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis 1)			400		mV
I_L	Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			±1	µA
I_S	Static current consumption 2)	Floating input mode			200	
R_{PU}	Weak pull-up equivalent resistor3)	$V_{IN}=V_{SS}$, V_{DD}=5V, $T_A \leq 85°C$	50	100	170	kΩ
		V_{DD}=3V		200		
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time 1)	C_L=50pF Between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time 1)			25		
$t_{w(IT)in}$	External interrupt pulse time 4)		1			t_{CPU}

Notes:

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

3. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in Figure 83).

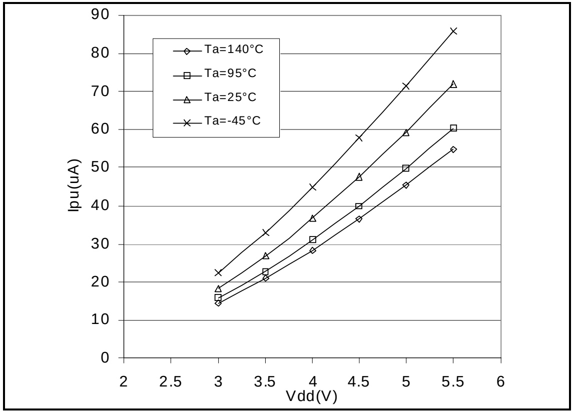
4. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

Figure 82. Two typical Applications with unused I/O Pin

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.

Note: 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.

Figure 83. Typical I_{PU} vs. V_{DD} with V_{IN}=V_{SS}



I/O PORT PIN CHARACTERISTICS (Cont'd)

13.8.2 Output Driving Current

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

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{OL} 1)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time (see Figure 86)	$V_{DD}=5V$	$I_{IO}=+5mA$ $T_A \leq 85°C$		0.65	1.0	V
			$I_{IO}=+2mA$ $T_A \leq 85°C$		0.25	0.4	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see Figure 89)		$I_{IO}=+20mA, T_A \leq 85°C$		1.05	1.3	
			$I_{IO}=+8mA$ $T_A \leq 85°C$		0.4	0.75	
V_{OH} 2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 92)		$I_{IO}= -5mA,$ $T_A \leq 85°C$	$V_{DD}-1.5$	4.30		
			$I_{IO}= -2mA$ $T_A \leq 85°C$	$V_{DD}-0.8$	4.70		
V_{OL} 1)3)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time (see Figure 85)	$V_{DD}=4V$	$I_{IO}=+2mA$ $T_A \leq 85°C$		0.25		
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see Figure 88)		$I_{IO}=+8mA$ $T_A \leq 85°C$		0.35		
V_{OH} 2)3)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 91)		$I_{IO}= -2mA$ $T_A \leq 85°C$		3.70		
V_{OL} 1)3)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time (see Figure 84)	$V_{DD}=3V$	$I_{IO}=+2mA$ $T_A \leq 85°C$		0.30		
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see Figure 87)		$I_{IO}=+8mA$ $T_A \leq 85°C$		0.40		
V_{OH} 2)3)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 90)		$I_{IO}= -2mA$ $T_A \leq 85°C$		2.60		

Notes:

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 13.2.2 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 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}.
3. Not tested in production, based on characterization results.

Figure 84. Typical V_{OL} at V_{DD}=3V

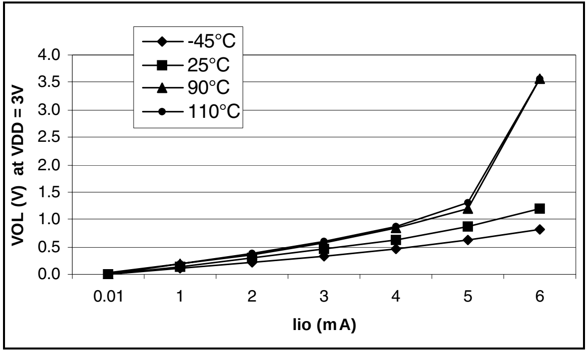


Figure 85. Typical V_{OL} at V_{DD}=4V

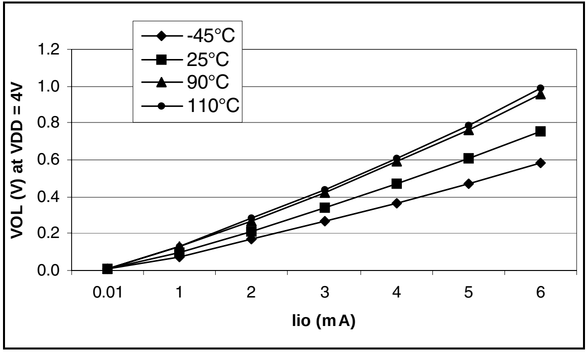


Figure 86. Typical V_{OL} at V_{DD}=5V

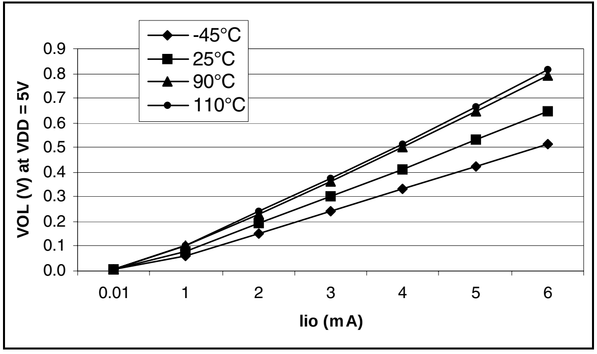


Figure 87. Typical V_{OL} at V_{DD}=3V (high-sink)

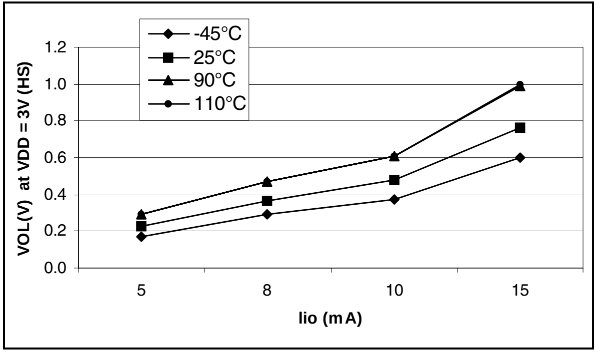


Figure 88. Typical V_{OL} at V_{DD}=4V (high-sink)

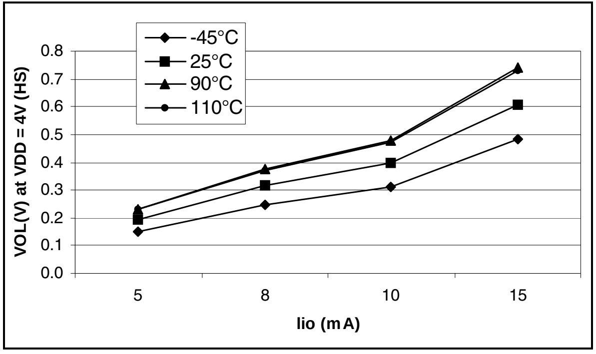


Figure 89. Typical V_{OL} at V_{DD}=5V (high-sink)

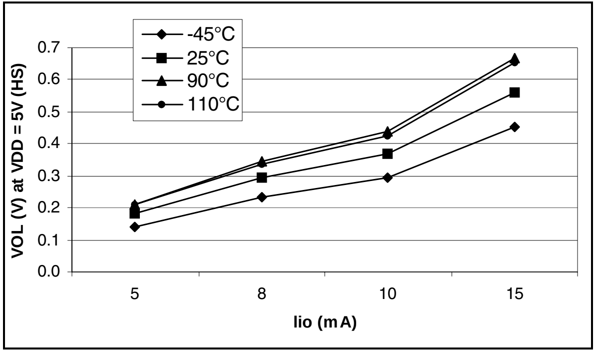


Figure 90. Typical V_{DD}-V_{OH} at V_{DD}=3.0V

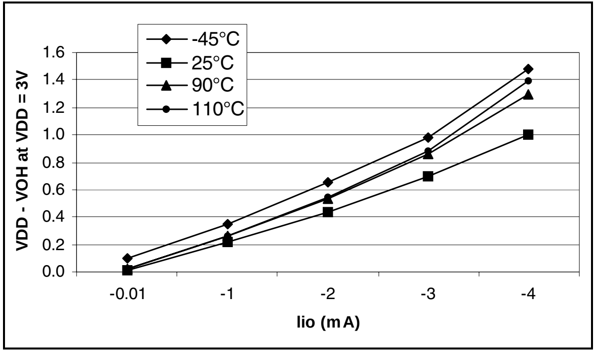
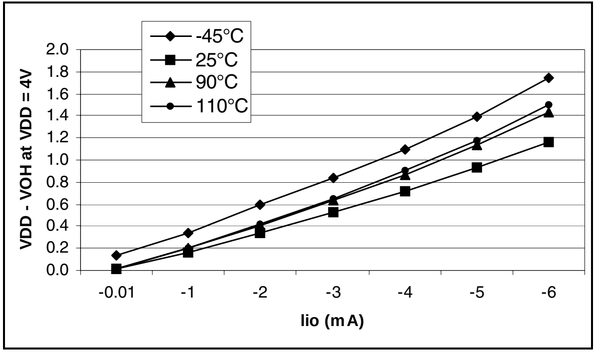


Figure 91. Typical V_{DD}-V_{OH} at V_{DD}=4.0V



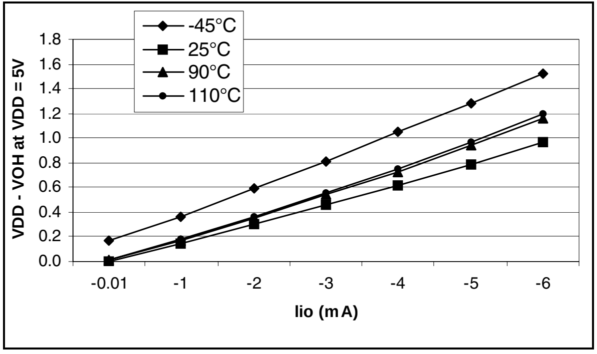


Figure 92. Typical V_{DD}-V_{OH} at V_{DD}=5V

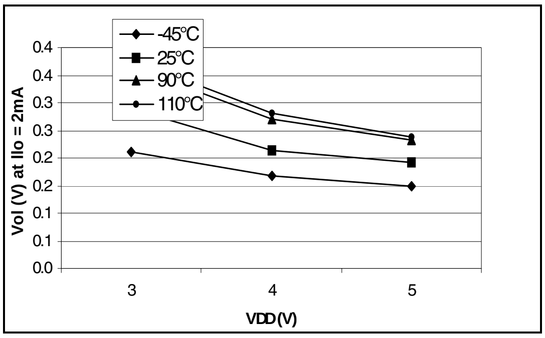


Figure 93. Typical V_{OL} vs. V_{DD} (standard I/Os)

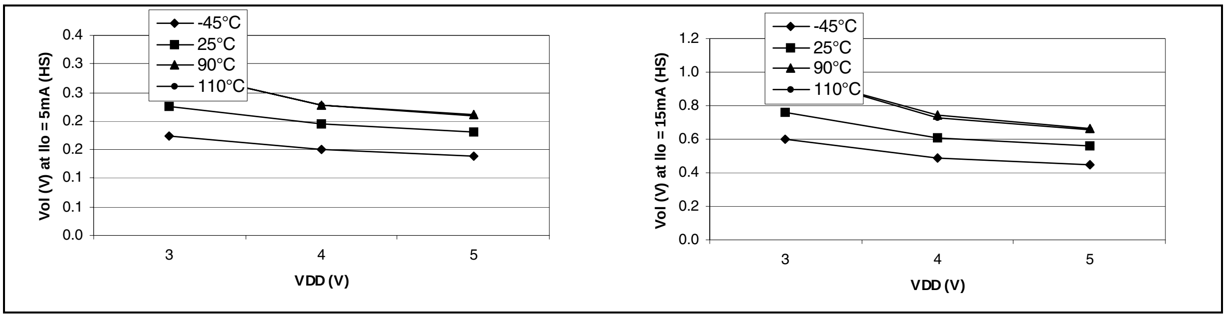


Figure 94. Typical V_{OL} vs. V_{DD} (high-sink I/Os)

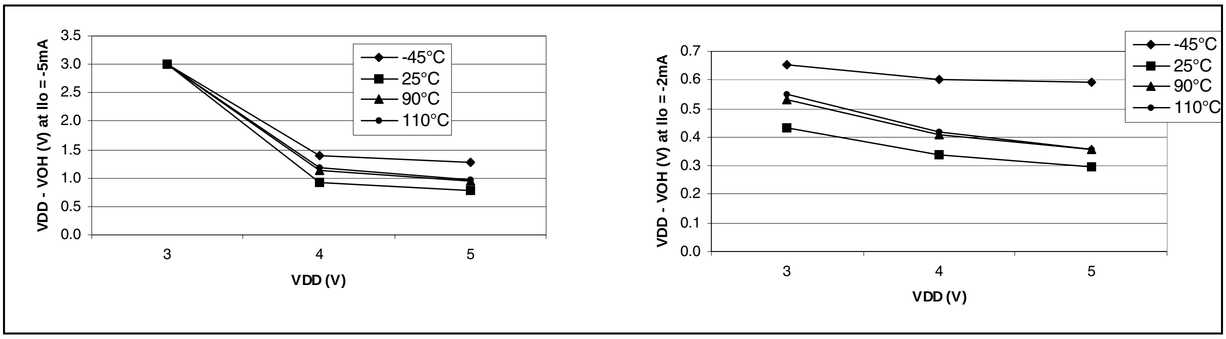


Figure 95. Typical V_{DD}-V_{OH} vs. V_{DD}

13.9 CONTROL PIN CHARACTERISTICS

13.9.1 Asynchronous $\overline{\text{RESET}}$ Pin

T_A = -40°C to 85°C, unless otherwise specified

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage			V_{SS} - 0.3		0.3xV_{DD}	V
V_{IH}	Input high level voltage			0.7xV_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis 1)				1		V
V_{OL}	Output low level voltage 2)	V_{DD}=5V	I_{IO}=+5mA T_A≤85°C		0.5	1.0	V
			T_A≥85°C		-	1.2	
			I_{IO}=+2mA T_A≤85°C		0.45	0.7	
			T_A≥85°C		-	0.9	
R_{ON}	Pull-up equivalent resistor 3) 1)	V_{DD}=5V	T_A≤85°C	10	46	70	kΩ
		V_{DD}=3V			91		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			µs
$t_{g(RSTL)in}$	Filtered glitch duration				200		ns

Notes:

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

2. The I_{IO} current sunk must always respect the absolute maximum rating specified in section 13.2.2 on page 130 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.

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 has to 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.

CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 96. $\overline{\text{RESET}}$ pin protection when LVD is enabled.[1)2)3)4)]

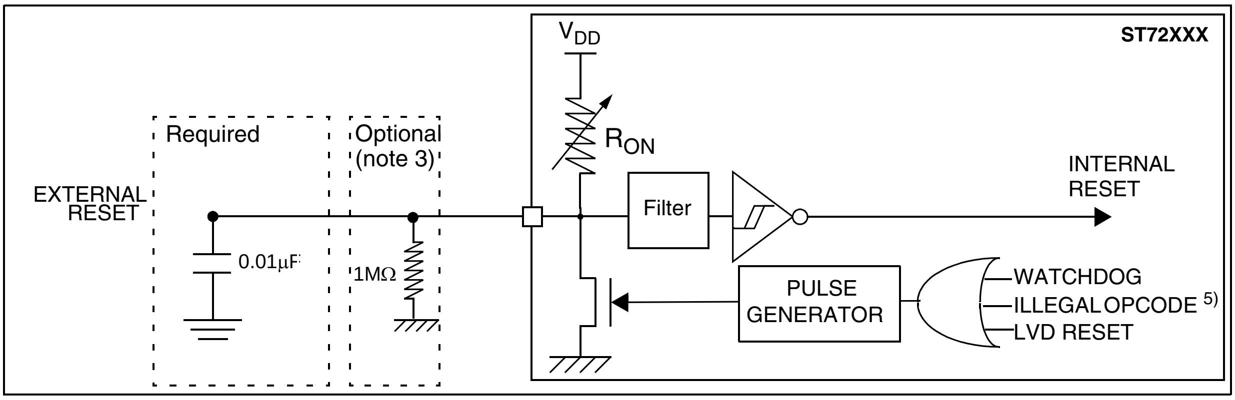
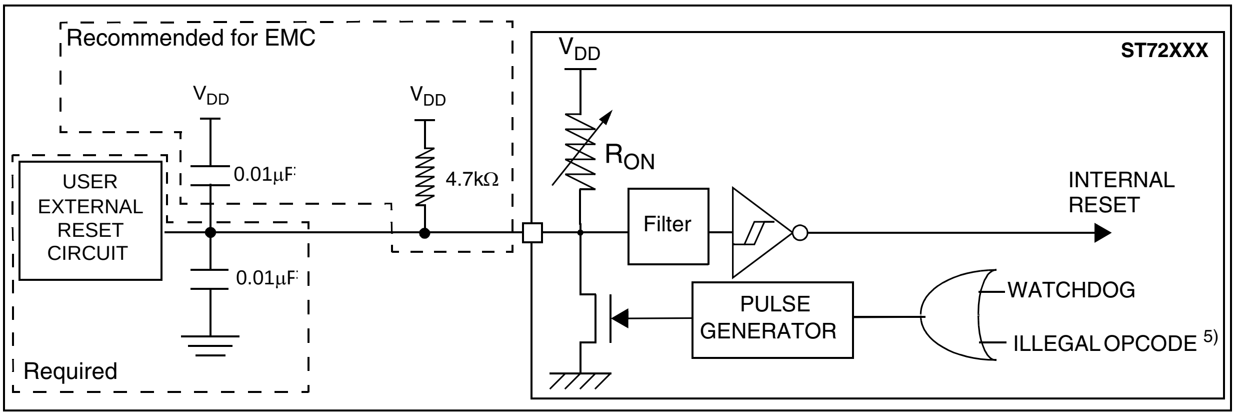


Figure 97. $\overline{\text{RESET}}$ pin protection when LVD is disabled.[1)]



Note 1:

- The reset network protects the device against parasitic resets.
- The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).
- Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{\text{RESET}}$ pin can go below the V_{IL} max. level specified in section 13.9.1 on page 149. Otherwise the reset will not be taken into account internally.
- Because the reset circuit is designed to allow the internal RESET to be output in the $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in section 13.2.2 on page 130.

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

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

Note 4: Tips when using the LVD:

- 1. Check that all recommendations related to ICCCLK and reset circuit have been applied (see caution in Table 1 on page 6 and notes above)
- 2. Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100nF + 1MΩ pull-down on the RESET pin.
- 3. The capacitors connected on the RESET pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the RESET pin with a 5µF to 20µF capacitor."

Note 5: Please refer to "Illegal Opcode Reset" on page 126 for more details on illegal opcode reset conditions

13.10 COMMUNICATION INTERFACE CHARACTERISTICS

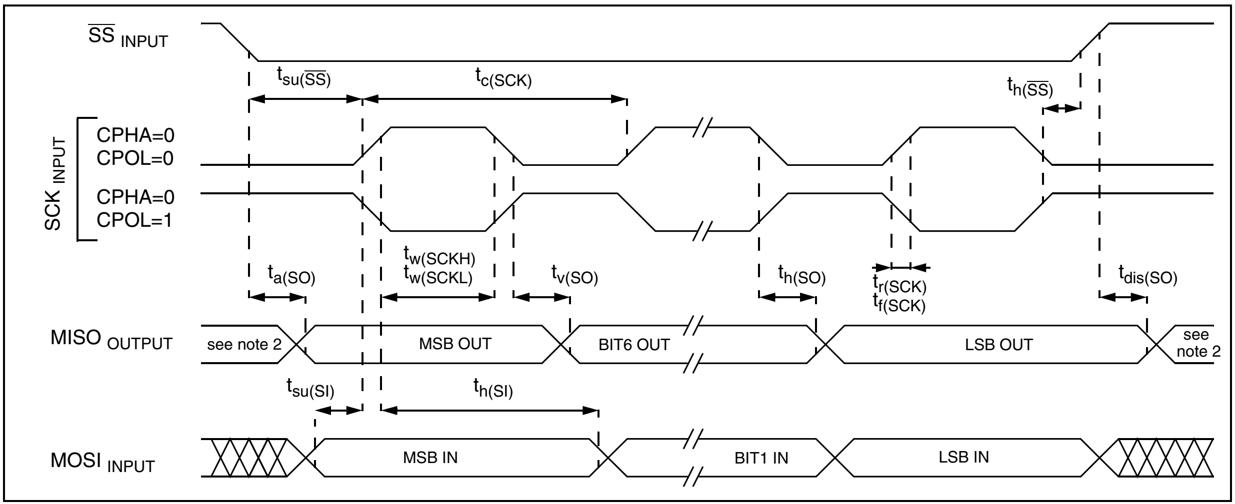
13.10.1 SPI - Serial Peripheral Interface

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

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

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCK} = $1/t_{c(SCK)}$	SPI clock frequency	Master f_{CPU}=8MHz	f_{CPU}/128 = 0.0625	f_{CPU}/4 = 2	MHz
		Slave f_{CPU}=8MHz	0	f_{CPU}/2 = 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see I/O port pin description		
$t_{su(\overline{SS})}$	$\overline{SS}$ setup time	Slave	120		ns
$t_{h(\overline{SS})}$	$\overline{SS}$ hold time	Slave	120		
$t_{w(SCKH)}$ $t_{w(SCKL)}$	SCK high and low time	Master Slave	100 90		
$t_{su(MI)}$ $t_{su(SI)}$	Data input setup time	Master Slave	100 100		
$t_{h(MI)}$ $t_{h(SI)}$	Data input hold time	Master Slave	100 100		
$t_{a(SO)}$	Data output access time	Slave	0	120	
$t_{dis(SO)}$	Data output disable time	Slave		240	
$t_{v(SO)}$	Data output valid time	Slave (after enable edge)		120	
$t_{h(SO)}$	Data output hold time		0		
$t_{v(MO)}$	Data output valid time	Master (before capture edge)	0.25		t_{CPU}
$t_{h(MO)}$	Data output hold time		0.25		

Figure 98. SPI Slave Timing Diagram with CPHA=0 3)



Notes:

1. Data based on design simulation and/or characterisation results, not tested in production.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.
3. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

Figure 99. SPI Slave Timing Diagram with CPHA=1[1)]

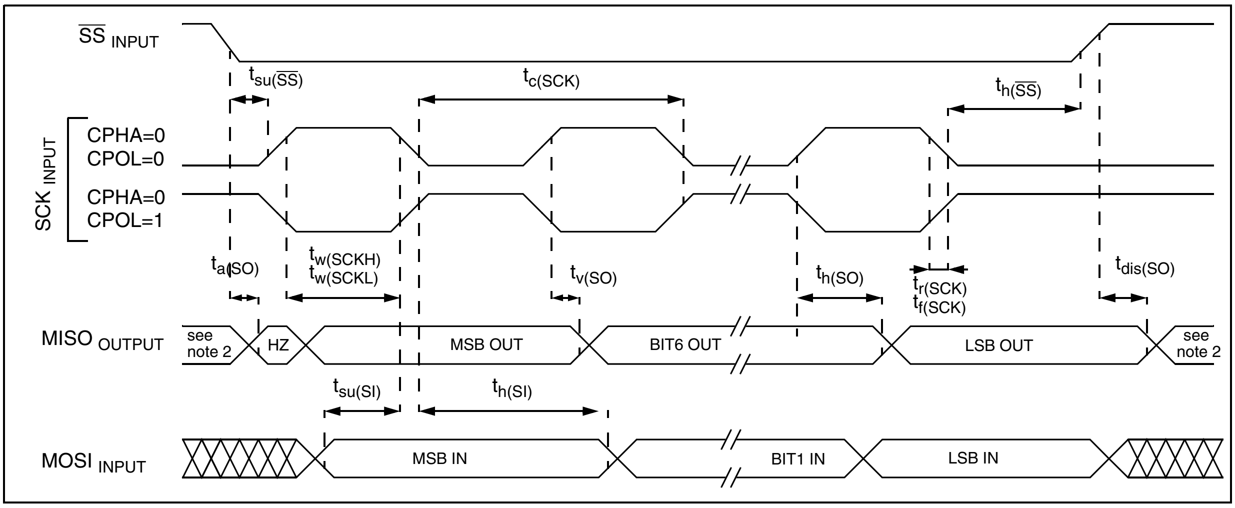
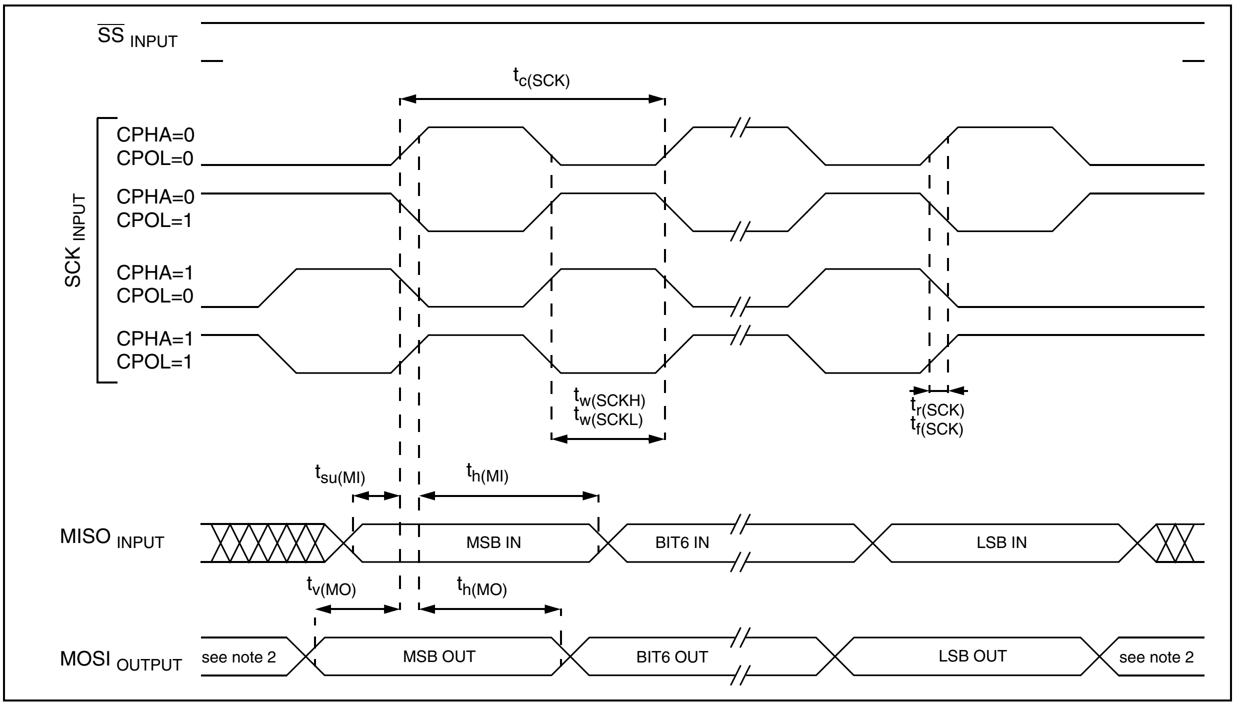


Figure 100. SPI Master Timing Diagram [1)]



Notes:

1. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.
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 condition for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
f_{ADC}	ADC clock frequency		0.5		4	MHz
V_{AIN}	Conversion voltage range [2]		V_{SSA}		V_{DDA}	V
R_{AIN}	External input resistor				10 [3]	kΩ
C_{ADC}	Internal sample and hold capacitor			6		pF
t_{STAB}	Stabilization time after ADC enable	f_{CPU}=8MHz, f_{ADC}=4MHz		0 [4]		μs
t_{ADC}	Conversion time (Sample+Hold)			3.5		
	- Sample capacitor loading time - Hold conversion time			4 10		$1/f_{ADC}$

Notes:

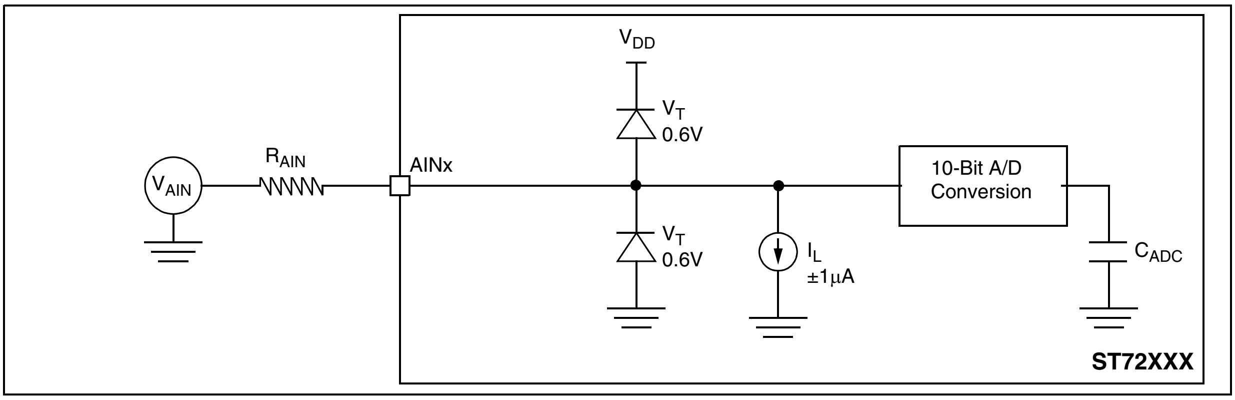
1. Unless otherwise specified, typical data are based on T_A=25°C and V_{DD}-V_{SS}=5V. They are given only as design guidelines and are not tested.

2. When V_{DDA} and V_{SSA} pins are not available on the pinout, the ADC refers to V_{DD} and V_{SS}.

3. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance g(reater than10kΩ). Data based on characterization results, not tested in production.

4. 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 101. Typical Application with ADC



ADC CHARACTERISTICS (Cont'd)

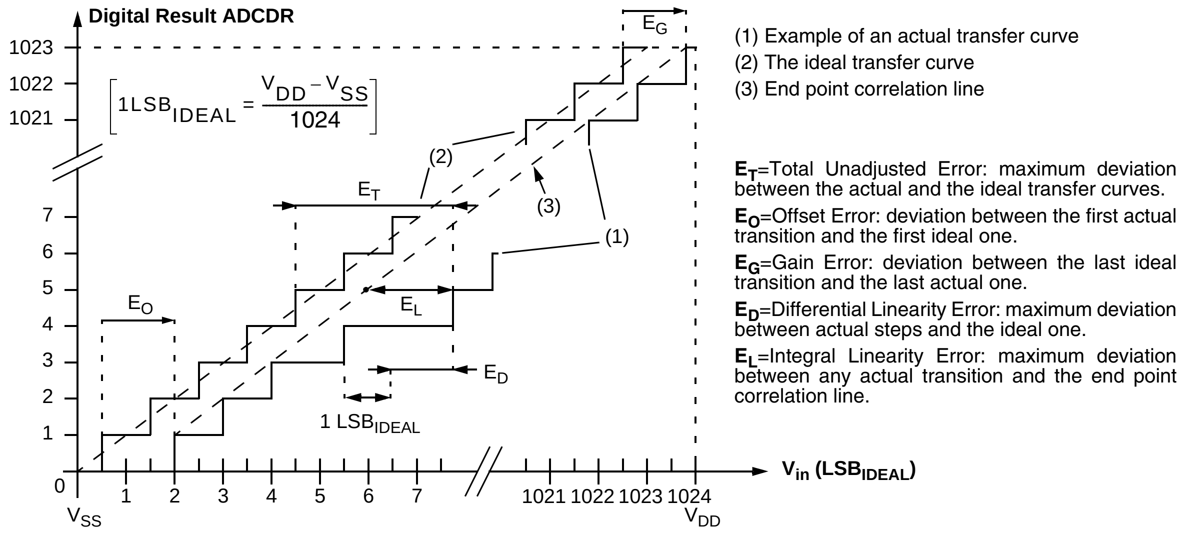
ADC Accuracy with $3V \le V_{DD} \le 5.5V$

Symbol	Parameter	Conditions	Typ	Max	Unit
$\lvert E_T \rvert$	Total unadjusted error	f_{CPU}=8MHz, f_{ADC}=4MHz 1)	1.5	4	LSB
$\lvert E_O \rvert$	Offset error		0.5	1.5	
$\lvert E_G \rvert$	Gain Error		1	1.5	
$\lvert E_D \rvert$	Differential linearity error		1.5	3	
$\lvert E_L \rvert$	Integral linearity error		1.5	3	

Notes:

1) Data based on characterization results over the whole temperature range, monitored in production.

Figure 102. ADC Accuracy Characteristics



(1) Example of an actual transfer curve
(2) The ideal transfer curve
(3) End point correlation line

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.

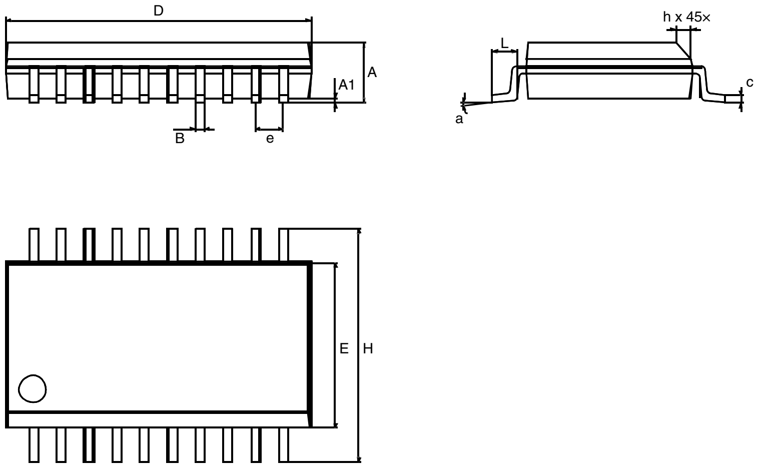
E_D=Differential Linearity Error: maximum deviation between actual steps and the ideal one.

E_L=Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

14 PACKAGE CHARACTERISTICS

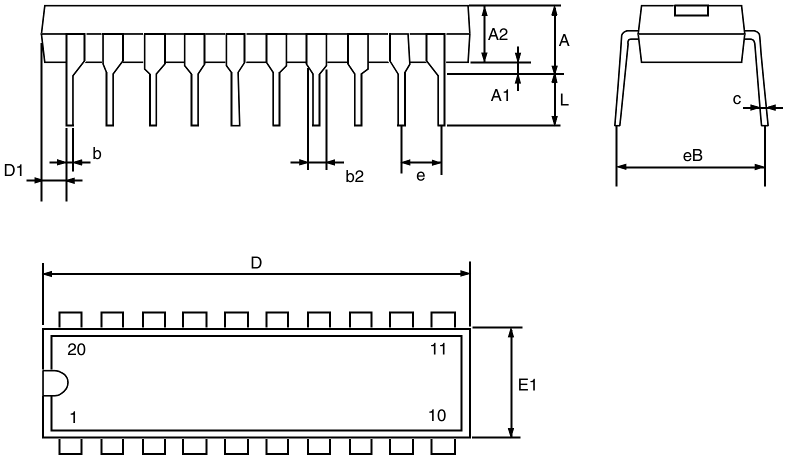
14.1 PACKAGE MECHANICAL DATA

Figure 103. 20-Pin Plastic Small Outline Package, 300-mil Width



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
	Number of Pins					
N	20					

Figure 104. 20-Pin Plastic Dual In-Line Package, 300-mil Width



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			5.33			0.210
A1	0.38			0.015		
A2	2.92	3.30	4.95	0.115	0.130	0.195
b	0.36	0.46	0.56	0.014	0.018	0.022
b2	1.14	1.52	1.78	0.045	0.060	0.070
c	0.20	0.25	0.36	0.008	0.010	0.014
D	24.89	26.16	26.92	0.980	1.030	1.060
D1	0.13			0.005		
e		2.54			0.100	
eB			10.92			0.430
E1	6.10	6.35	7.11	0.240	0.250	0.280
L	2.92	3.30	3.81	0.115	0.130	0.150
	Number of Pins					
N	20					

14.2 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient)	TBD	°C/W
P_D	Power dissipation [1]	500	mW
T_{Jmax}	Maximum junction temperature [2]	150	°C

Notes:

1. The power dissipation is obtained from the formula $P_D=P_{INT}+P_{PORT}$ where P_{INT} is the chip internal power ($I_{DD}xV_{DD}$) and P_{PORT} is the port power dissipation determined by the user.
2. The average chip-junction temperature can be obtained from the formula $T_J = T_A + P_D \times R_{thJA}$.

14.3 SOLDERING AND GLUEABILITY INFORMATION

Recommended soldering information given only as design guidelines.

Figure 105. Recommended Wave Soldering Profile (with 37% Sn and 63% Pb)

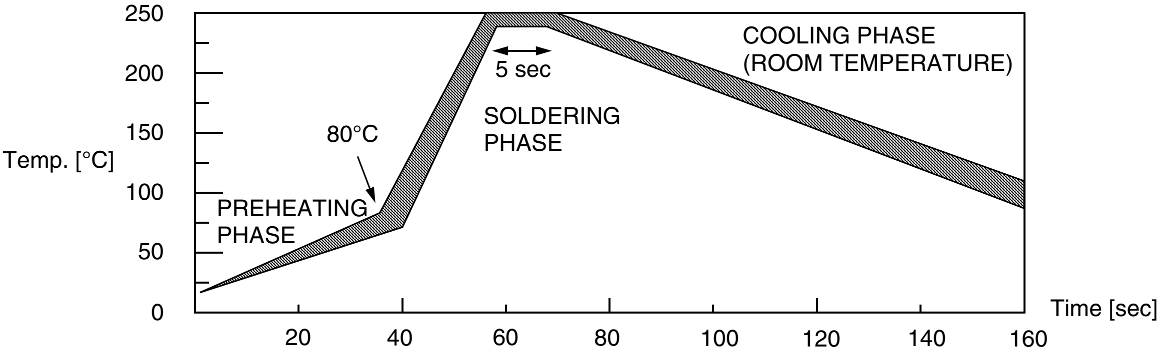
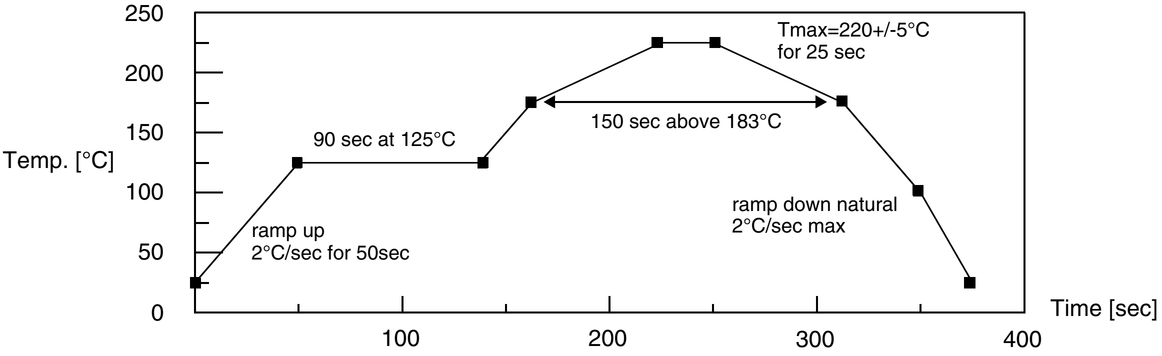


Figure 106. Recommended Reflow Soldering Oven Profile (MID JEDEC)



Recommended glue for SMD plastic packages:

- Heraeus: PD945, PD955
- Loctite: 3615, 3298

15 DEVICE CONFIGURATION

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

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

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

15.1 FLASH OPTION BYTES

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

OPTION BYTE 0

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

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

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

OPT6:4 = **OSCRANGE[2:0]** *Oscillator Range*
When the internal RC oscillator is not selected (Option OSC=1), these option bits select the range of the resonator oscillator current source or the external clock source.

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

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.

OPT 3:2 = **SEC[1:0]** *Sector 0 size definition*
These option bits indicate the size of sector 0 according to the following table.

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

OPT1 = **FMP_R** *Read-out protection*
Readout protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. Erasing the option bytes when the FMP_R option is selected will cause the whole memory to be erased first and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual and section 4.5 on page 13 for more details
0: Read-out protection off
1: Read-out protection on

OPT 0 = **FMP_W** *FLASH write protection*
This option indicates if the FLASH program memory is write protected.
Warning: When this option is selected, the program memory (and the option bit itself) can never be erased or programmed again.
0: Write protection off
1: Write protection on
The option bytes have no address in the memory map and can be accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the FLASH is fixed to FFh.

OPTION BYTES (Cont'd)

	OPTION BYTE 0								OPTION BYTE 1							
	7							0	7							0
	AWU CK	OSCRANGE 2:0			SEC1	SEC0	FMPR	FMPW	PLL x4x8	PLL OFF	Res.	OSC	LVD 1:0		WDG SW	WDG HALT
Default Value	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	1

OPTION BYTE 1

OPT 7 = **PLLx4x8** *PLL Factor Selection.*
0: PLLx4
1: PLLx8

OPT 6 = **PLLOFF** *PLL Disable*
This option bit enables or disables the PLL.
0: PLL enabled
1: PLL disabled (bypassed)

OPT 5 = Reserved. Must always be set to 1.

OPT 4 = **OSC** *RC Oscillator Selection*
This option bit enables to select the internal RC Oscillator.
0: RC Oscillator on
1: RC Oscillator off

OPT 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.

Configuration	VD1	VD0
LVD Off	1	1
Highest Voltage Threshold	1	0
Medium Voltage Threshold	0	1
Lowest Voltage Threshold	0	0

OPT 1 = **WDGSW** *Hardware or Software Watchdog*
0: Hardware (watchdog always enabled)
1: Software (watchdog to be enabled by software)

OPT 0 = **WDG HALT** *Watchdog Reset on Halt*
0: No reset generation when entering HALT mode
1: Reset generation when entering HALT mode

15.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

Customer code is made up of the FASTROM contents and the list of the selected options (if any). The FASTROM contents are to be sent on 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 161.

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.

Table 24. Supported part numbers

Part Number	Program Memory (Bytes)	Data EEPROM (Bytes)	RAM (Bytes)	INTERNAL RC	Temp. Range	Package
ST7FLITE30F2B6	8K FLASH	-	384	-	-40°C +85°C	DIP20
ST7FLITE30F2M6		-		-		SO20
ST7FLITE35F2B6		-		yes		DIP20
ST7FLITE35F2M6		-		yes		SO20
ST7FLITE39F2B6		256		yes		DIP20
ST7FLITE39F2M6		256		yes		SO20

░ Contact ST sales office for product availability

ST7LITE3 FASTROM MICROCONTROLLER OPTION LIST

(Last update: July 2005)

Customer . . . .
Address . . . .
. . . .
Contact . . . .
Phone No . . . .
Reference FASTROM Code*: . . . .
*FASTROM code name is assigned by STMicroelectronics.
FASTROM code must be sent in .S19 format. .Hex extension cannot be processed.

Device Type/Memory Size/Package (check only one option):

FASTROM DEVICE:	8K FASTROM
PDIP20:	[ ]
SO20:	[ ]

Conditioning (check only one option):

Packaged Product (do not specify for DIP package)	
[ ] Tape & Reel	[ ] Tube

Special Marking: [ ] No [ ] Yes "_ _ _ _ _ _ _ _ "
Authorized characters are letters, digits, '.', '-', '/' and spaces only.
Maximum character count:
PDIP20/SO20 (8 char. max) : _ _ _ _ _ _ _ _

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 OSC1		
	[ ] Internal RC Oscillator			
Sector 0 size:	[ ] 0.5K	[ ] 1K	[ ] 2K	[ ] 4K
Readout Protection:	[ ] Disabled	[ ] Enabled		
FLASH Write Protection	[ ] Disabled	[ ] Enabled		
PLL	[ ] Disabled	[ ] PLLx4	[ ] PLLx8	
LVD Reset	[ ] Disabled	[ ] Highest threshold [ ] Medium threshold [ ] Lowest threshold		
Watchdog Selection:	[ ] Software Activation		[ ] Hardware Activation	
Watchdog Reset on Halt:	[ ] Disabled		[ ] Enabled	

Comments : . . . .
Supply Operating Range in the application: . . . .
Notes . . . .
Date: . . . .
Signature: . . . .

Important note: Not all configurations are available. See section 15.1 on page 158 for authorized option byte combinations.

Please download the latest version of this option list from:
http://www.st.com/mcu > **downloads > ST7 microcontrollers > Option list**

15.3 DEVELOPMENT TOOLS

STMicroelectronics offers a range of hardware and software development tools for the ST7 microcontroller family. Full details of tools available for the ST7 from third party manufacturers can be obtained from the STMicroelectronics Internet site: http//www.st.com.

Tools from these manufacturers include C compliers, evaluation tools, in-circuit debuggers, emulators and programmers.

In-Circuit Debugging Tools

Two types of debuggers are available for the ST7LITE3 family:

- **ST7FLITE-SK/RAIS** Low-cost in-circuit debugging/programming tool from Raisonance.
- **STXF-INDART/USB** Low-cost in-circuit debugging tool from Softec Microsytem.

Emulators

Two types of emulators are available from ST for the ST7LITE3 family:

- **ST7 DVP3** entry-level emulator offers a flexible and modular debugging and programming solution.
- **ST7 EMU3** high-end emulator is delivered with everything (probes, TEB, adapters etc.) needed to start emulating the ST7LITE3. To configure it to emulate other ST7 subfamily devices, the active probe for the ST7EMU3 can be changed and the ST7EMU3 probe is designed for easy interchange of TEBs (Target Emulation Board). See Table 25.

Flash Programming tools

- ST7-STICK ST7 In-circuit Communication Kit, a complete software/hardware package for programming ST7 Flash devices. It connects to a host PC parallel port and to the target board or socket board via ST7 ICC connector.
- ICC Socket Boards provide an easy to use and flexible means of programming ST7 Flash devices. They can be connected to any tool that supports the ST7 ICC interface, such as ST7 EMU3, ST7-DVP3, inDART, Rlink, ST7-STICK, or many third-party development tools.

Table 25. STMicroelectronics Development Tools

Supported Products	Emulation				Programming
	ST7 DVP3 Series		ST7 EMU3 series		ICC Socket Board
	Emulator	Connection kit	Emulator	Active Probe & T.E.B.	
ST7FLITE30 ST7FLITE35 ST7FLITE39	ST7MDT10-DVP3	ST7MDT10-20/DVP	ST7MDT10-EMU3	ST7MDT10-TEB	ST7SB10-123[1]

Note 1: Add suffix /EU, /UK, /US for the power supply of your region.

16 KNOWN LIMITATIONS

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, i.e. 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 can be avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request

Ex:

```
        SIM
        reset flag or interrupt mask
        RIM
```

16.2 LINSCI LIMITATIONS

16.2.1 LINSCI wrong break duration

SCI Mode

A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 20 bits instead of 10 bits if M=0

- 22 bits instead of 11 bits if M=1.

In the same way, as long as the SBK bit is set, break characters are sent to the TDO pin. This may lead to generate one break more than expected.

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (f_{CPU}=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Workaround

If this wrong duration is not compliant with the communication protocol in the application, software can request that an Idle line be generated before the break character. In this case, the break duration is always correct assuming the application is not doing anything between the idle and the

IMPORTANT NOTES (Cont'd)

break. This can be ensured by temporarily disabling interrupts.

The exact sequence is:

- Disable interrupts
- Reset and Set TE (IDLE request)
- Set and Reset SBK (Break Request)
- Re-enable interrupts

LIN mode

If the LINE bit in the SCICR3 is set and the M bit in the SCICR1 register is reset, the LINSCI is in LIN master mode. A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 24 bits instead of 13 bits

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (fCPU=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Analysis

The LIN protocol specifies a minimum of 13 bits for the break duration, but there is no maximum value. Nevertheless, the maximum length of the header is specified as (14+10+10+1)x1.4=49 bits. This is composed of:

- the synch break field (14 bits),
- the synch field (10 bits),
- the identifier field (10 bits).

Every LIN frame starts with a break character. Adding an idle character increases the length of each header by 10 bits. When the problem occurs, the header length is increased by 11 bits and becomes ((14+11)+10+10+1)=45 bits.

To conclude, the problem is not always critical for LIN communication if the software keeps the time between the sync field and the ID smaller than 4 bits, i.e. 208us at 19200 baud.

The workaround is the same as for SCI mode but considering the low probability of occurrence (1%), it may be better to keep the break generation sequence as it is.

16.2.2 Header Time-out does not prevent wake-up from mute Mode

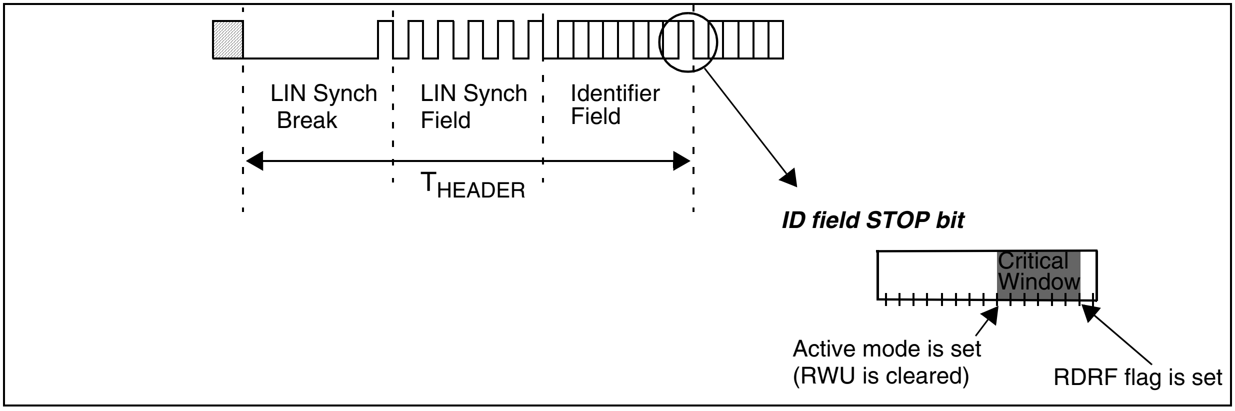
Normally, when LINSCI is configured in LIN slave mode, if a header time-out occurs during a LIN header reception (i.e. header length > 57 bits), the LIN Header Error bit (LHE) is set, an interrupt occurs to inform the application but the LINSCI should stay in mute mode, waiting for the next header reception.

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 107), 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 107. Header Reception Event Sequence



IMPORTANT NOTES (Cont'd)

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 will be generated on each data byte reception.

Workaround

The problem can be detected in the LINSCI interrupt routine. In case of timeout 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 108. Workaround is shown in bold characters.

Figure 108. 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

Date	Revision	Main changes
29-Jul-05	4.0	First release on Internet Main changes (versus rev. 3.0): – Changed status of the document: datasheet instead of preliminary data – Changed number of timers on first page – Changed $I_{DD(PLL)}$ and added note 4 to $I_{DD(RC)}$ in section 13.3.1.1 on page 132 and section 13.3.1.2 on page 134 – Removed section 13.3.2 (General Operating Conditions: Suffix 3 Devices) – Removed note 7 to section 13.6 on page 141 – Added V_{hys} typical value to section 13.8.1 on page 144 and changed R_{PU} typ value at V_{DD}=3V – Added Figure 69 on page 133 and Figure 71 on page 135 – Added Figure 70 on page 133 and Figure 72 on page 135 – Added Figure 73, Figure 74 and Figure 75 on page 136 – Added note to Figure 82 on page 144 – Added Figure 83 on page 145 – Removed min and max values for V_{DD}=3V and V_{DD}=4V in section 13.8.2 on page 146 – Added R_{ON} typ value for V_{DD}=3V in section 13.9.1 on page 149 and changed R_{ON} typ value for VDD=5V – In section 13.11 on page 153, added f_{ADC} min value, C_{ADC} and R_{AIN} values and removed I_{ADC} row – Changed section 15.3 on page 162 (removed note 1 to DVP3 and added in-circuit debugging tools)

STMicroelectronics group of companies
Australia – Belgium - Brazil - Canada - China – Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America
www.st.com