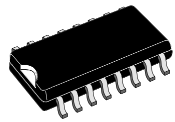
ST7L05, ST7L09

8-bit MCU for automotive with single voltage Flash memory, data EEPROM, ADC, timers, SPI

Features

- **Memories**
  - 1.5 Kbytes program memory: Single voltage extended Flash (XFlash) with read-out protection capability, In-Application Programming (IAP) and In-Circuit Programming (ICP) for XFlash devices
  - 128 bytes RAM
  - 128 bytes data EEPROM with read-out protection, 300K Write/Erase cycles guaranteed
  - XFlash and EEPROM data retention: 20 years at 55°C
- **Clock, Reset and Supply Management**
  - Clock sources: High precision internal RC oscillator or external clock
  - PLL x8 for 8 MHz internal clock
  - 4 Power Saving Modes: Halt, Active Halt, Wait and Slow
- **Interrupt Management**
  - 10 interrupt vectors plus TRAP and RESET
  - 4 external interrupt lines (on four vectors)
- **I/O Ports**
  - 13 multifunctional bidirectional I/O lines
  - 9 alternate function lines
  - 6 high sink outputs
- **2 Timers**
  - One 8-bit Lite Timer (LT) with prescaler including: Watchdog, one realtime base and one input capture
  - One 12-bit Autoreload Timer (AT) with output compare function and PWM



SO16
150mil

- **1 Communication Interface**
  - SPI synchronous serial interface
- **A/D Converter**
  - 8-bit resolution for 0 to V_{DD}
  - 5 input channels
- **Instruction Set**
  - 8-bit data manipulation
  - 63 basic instructions with illegal opcode detection
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instruction
- **Development Tools**
  - Full hardware/software development package

Table 1. Device Summary

Features	ST7L05	ST7L09
Program Memory	1.5 Kbytes Flash	
RAM (stack)	128 bytes (64 bytes)	
Data EEPROM	-	128 bytes
Peripherals	Lite Timer with Watchdog, Autoreload Timer with 1 PWM, SPI, 8-bit ADC	
Operating Supply	3.0 to 5.5V	
CPU Frequency	Up to 8 MHz (with external resonator/clock or internal RC oscillator)	
Operating Temperature	Up to -40 to +85°C, -40 to +105°C	
Packages	SO16 150mil	

Rev. 3

December 2006

Table of Contents

Table of Contents

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

Please also pay special attention to the Section "KNOWN LIMITATIONS" on page 102.

1 DESCRIPTION

The ST7L0x is a member of the ST7 microcontroller family suitable for automotive applications. All ST7 devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

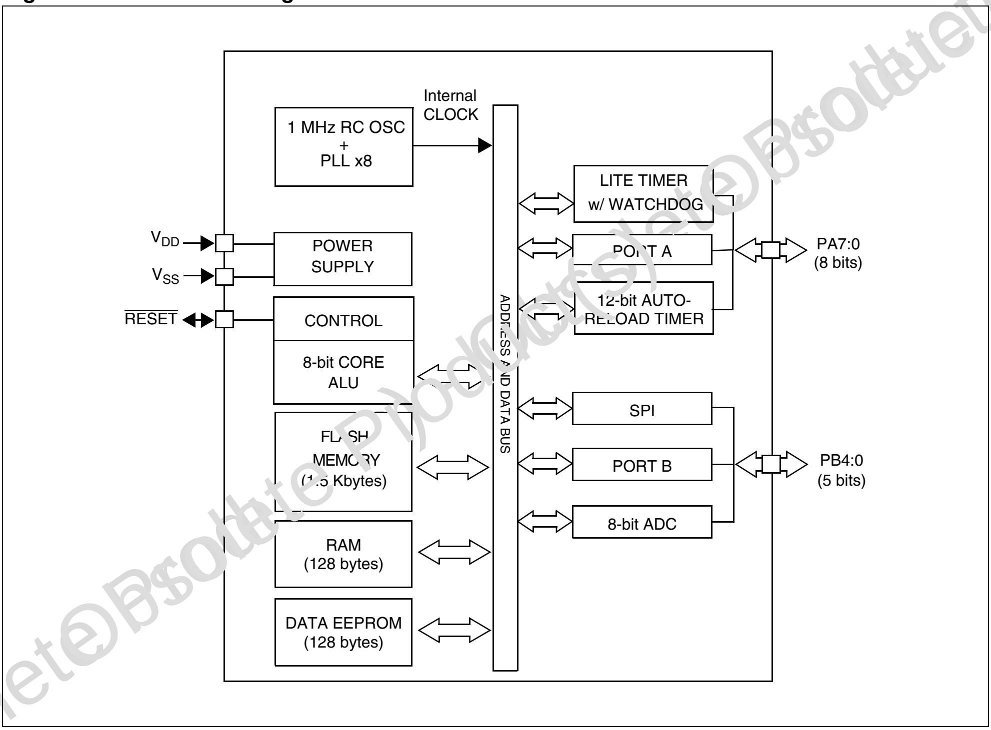
The ST7L0x features Flash memory with byte-by-byte In-Circuit Programming (ICP) and In-Application Programming (IAP) capability.

Under software control, the ST7L0x devices can be placed in WAIT, SLOW, or HALT mode, reducing power consumption when the application is in idle or standby state.

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

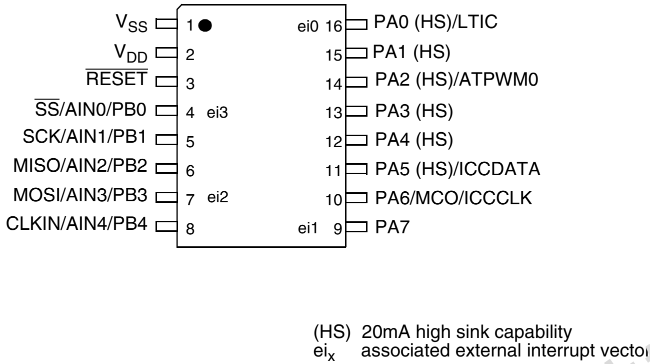
For easy reference, all parametric data is found in Section 13 on page 73.

Figure 1. General Block Diagram



2 PIN DESCRIPTION

Figure 2. 16-Pin Package Pinout (150mil)



PIN DESCRIPTION (Cont'd)

Legend / Abbreviations for Table 2:

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

In/Output level: C= CMOS $0.15V_{DD}/0.85V_{DD}$ with input trigger

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[1)], ana = analog

– Output: OD = open drain, PP = push-pull

Table 2. 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	$\overline{RESET}$	I/O	C_T			X			X		Top priority non-maskable interrupt (active low)	
4	PB0/AIN0/$\overline{SS}$	I/O	C_T		**X**	ei3		X	X	X	**Port B0**	ADC Analog Input 0 or SPI Slave Select (active low)
5	PB1/AIN1/SCK	I/O	C_T		**X**	X		X	X	X	**Port B1**	ADC Analog Input 1 or SPI Clock **Caution:** No negative current injection allowed on this pin. For details, refer to Section 13.2.2 on page 74.
6	PB2/AIN2/MISO	I/O	C_T		**X**	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	PA7	I/O	C_T		**X**	ei1			X	X	**Port A7**	
10	PA6 /MCO/ICCCLK	I/O	C_T		X	**X**			X	X	**Port A6**	Main Clock Output/In-Circuit Communication Clock. **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.
11	PA5/ ICCDATA	I/O	C_T	HS	**X**	X			X	X	**Port A5**	In-Circuit Communication Data
12	PA4	I/O	C_T	HS	**X**	X			X	X	**Port A4**	
13	PA3	I/O	C_T	HS	**X**	X			X	X	**Port A3**	
14	PA2/ATPWM0	I/O	C_T	HS	**X**	X			X	X	**Port A2**	Autoreload Timer PWM0

Pin No.	Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
					Input				Output			
			Input	Output	float	wpu	int	ana	OD	PP		
15	PA1	I/O	C_T	HS	**X**	X			X	X	**Port A1**	
16	PA0/LTIC	I/O	C_T	HS	**X**	ei0			X	X	**Port A0**	Lite Timer Input Capture

Note: In the interrupt input column, “eix” defines the associated external interrupt vector. If the weak pull-up column (wpu) is merged with the interrupt column (int), then the I/O configuration is pull-up interrupt input, else the configuration is floating interrupt input.

3 REGISTER AND MEMORY MAP

As shown in Figure 3, the MCU is capable of addressing 64 Kbytes of memories and I/O registers.

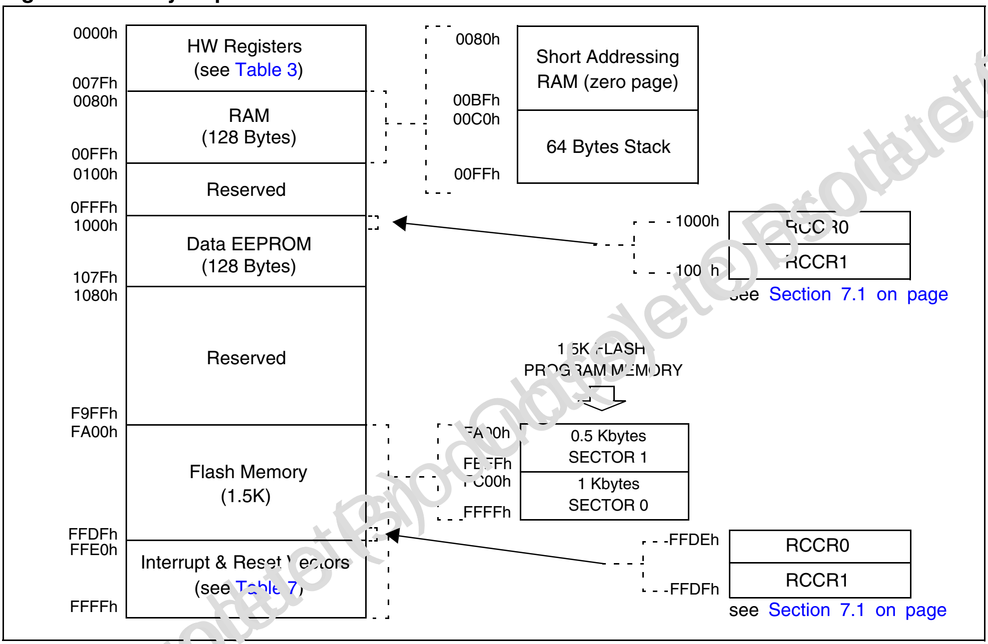
The available memory locations consist of up to 128 bytes of register locations, 128 bytes of RAM, 128 bytes of data EEPROM and up to 1.5 Kbytes of user program memory. The RAM space includes up to 64 bytes for the stack from 0C0h to 0FFh.

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

The size of Flash Sector 0 is configurable by Option byte.

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

Figure 3. Memory Map



REGISTER AND MEMORY MAP (Cont'd)

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

Table 3. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h 0001h 0002h	Port A	PADR PADDR PAOR	Port A Data Register Port A Data Direction Register Port A Option Register	00h 1) 00h 40h	R/W R/W R/W
0003h 0004h 0005h	Port B	PBDR PBDDR PBOR	Port B Data Register Port B Data Direction Register Port B Option Register	E0h 1) 00h 00h	R/W R/W R/W 2)
0006h to 000Ah	Reserved area (5 bytes)				
000Bh 000Ch	LITE TIMER	LTCSR LTICR	Lite Timer Control/Status Register Lite Timer Input Capture Register	xxh xxh	R/W Read Only
000Dh 000Eh 000Fh 0010h 0011h 0012h 0013h	AUTORELOAD TIMER	ATCSR CNTRH CNTRL ATRH ATRL PWMCR PWM0CSR	Timer Control/Status Register Counter Register High Counter Register Low Autoreload Register High Autoreload Register Low PWM Output Control Register PWM 0 Control/Status Register	00h 00h 00h 00h 00h 00h 00h	R/W Read Only Read Only R/W R/W R/W R/W
0014h to 0016h	Reserved area (3 bytes)				
0017h 0018h	AUTORELOAD TIMER	DCR0H DCR0L	PWM 0 Duty Cycle Register High PWM 0 Duty Cycle Register Low	00h 00h	R/W R/W
0019h to 002Eh	Reserved area (22 bytes)				
0002Fh	FLASH	FCSR	Flash Control/Status Register	00h	R/W
00030h	EEPROM	EECSR	Data EEPROM Control/Status Register	00h	R/W
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 ADCDR ADCCSR2	A/D Control Status Register A/D Data Register Control Status Register 2	00h 00h 00h	R/W Read Only R/W
0037h	ITC	EICR	External Interrupt Control Register	00h	R/W
0038h 0039h	CLOCKS	MCCSR RCCR	Main Clock Control/Status Register RC oscillator Control Register	00h FFh	R/W R/W
003Ah	SI	SICSR	System Integrity Control/Status Register	0xh	R/W
003Bh to 007Fh	Reserved area (69 bytes)				

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.

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 organization 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 can be programmed or erased.
- In-Circuit Programming. In this mode, Flash sectors 0 and 1, option byte row and data EEPROM can be programmed or erased without removing the device from the application board.
- In-Application Programming. In this mode, sector 1 and data EEPROM 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:

1. 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.
2. Download ICP Driver code in RAM from the ICCDATA pin
3. 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, allowing it to adapt to the user application, (such as user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored).

IAP mode is 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 four and up to six pins to be connected to the programming tool. These pins are:

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

Notes:

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

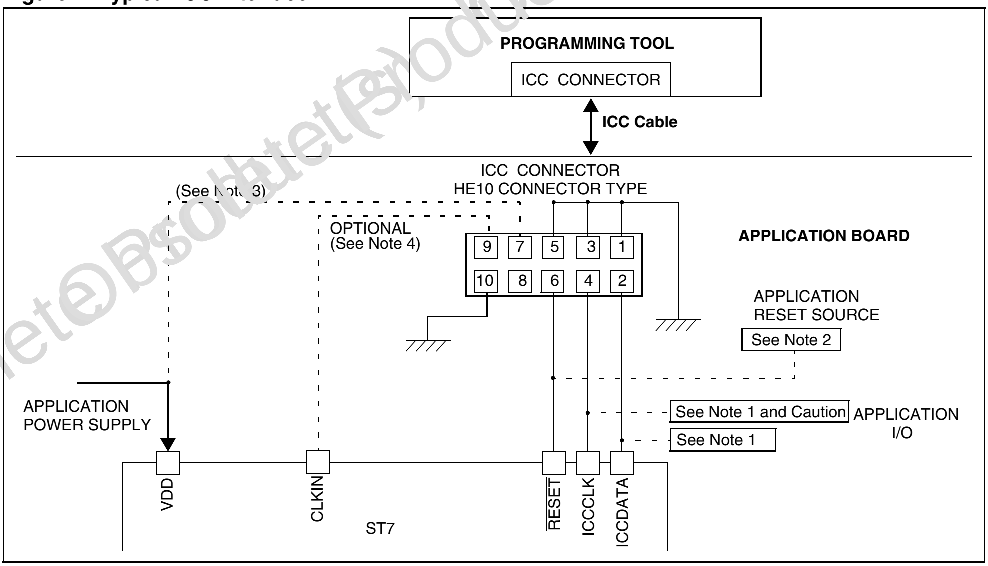
2. During the ICP session, the programming tool must control the $\overline{RESET}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor <1K). A Schottky diode can be used to isolate 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 must be connected to the CLKIN pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte.

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

Figure 4. Typical ICC Interface



FLASH PROGRAM MEMORY (Cont'd)

4.5 MEMORY PROTECTION

Two different types of memory protection exist: Read-out Protection and Write/Erase Protection, which are applied individually.

4.5.1 Read-out Protection

Read-out protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. Even if no protection is 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 is reprogrammed.

Read-out protection selection depends on the device type:

– In Flash devices it is enabled and removed through the FMP_R bit in the option byte.

– In ROM devices it is enabled by mask option specified in the Option List.

4.5.2 Flash Write/Erase Protection

Write/Erase protection, when set, makes it impossible to both overwrite and erase program memory. It does not apply to 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*.

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.

Table 4. Flash Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
002Fh	**FCSR** **Reset Value**	0	0	0	0	0	OPT 0	LAT 0	PGM 0

5 DATA EEPROM

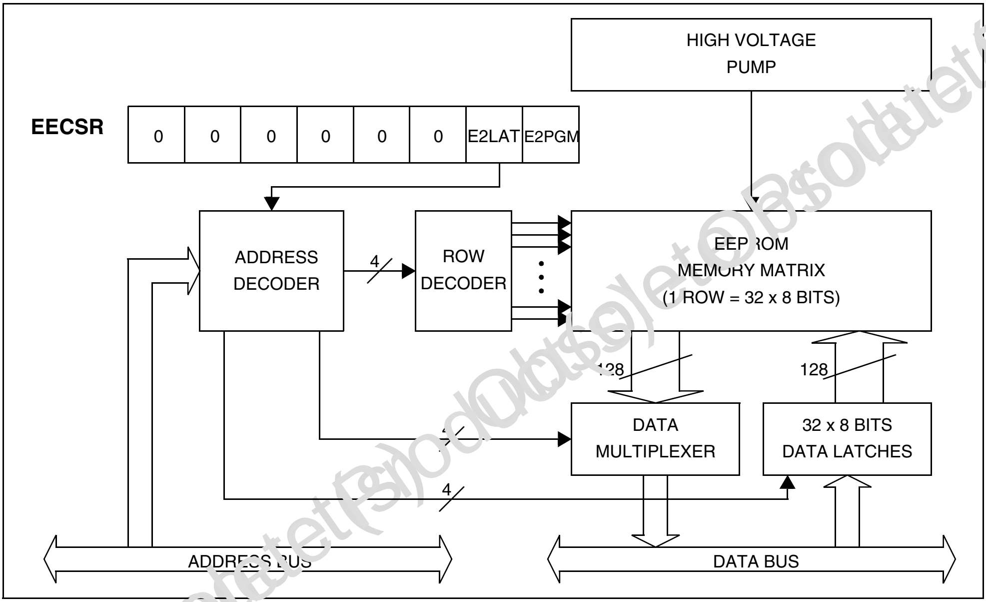
5.1 INTRODUCTION

The Electrically Erasable Programmable Read Only Memory can be used as a non-volatile back-up for storing data. Using the EEPROM requires a basic access protocol described in this chapter.

5.2 MAIN FEATURES

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

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

Write Operation (E2LAT = 1)

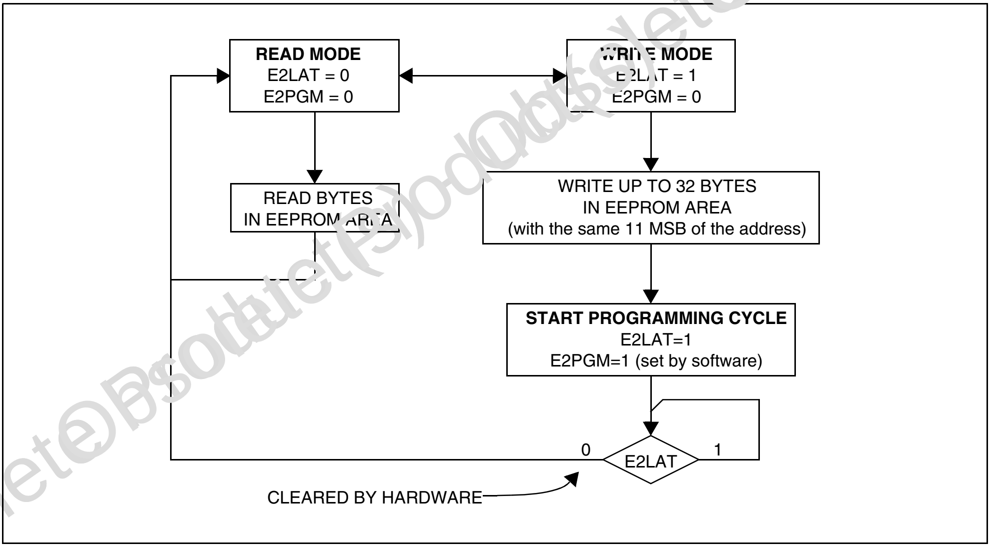
To access the write mode, the E2LAT bit has to be set by software (the E2PGM bit remains cleared). When a write access to the EEPROM area occurs, the value is latched inside the 32 data latches according to its address.

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

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

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

Figure 6. Data EEPROM Programming Flowchart

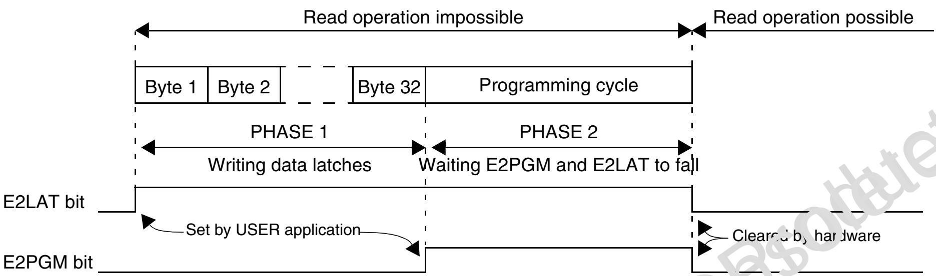


DATA EEPROM (Cont'd)

Figure 7. Data E^{2}PROM Write Operation

ROW DEFINITION

⇓ 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 RESET action), the integrity of the data in memory will not be guaranteed.

DATA EEPROM (Cont'd)

5.4 POWER SAVING MODES

Wait mode

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

Active Halt mode

Refer to Wait mode.

Halt mode

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

5.5 ACCESS ERROR HANDLING

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

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

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

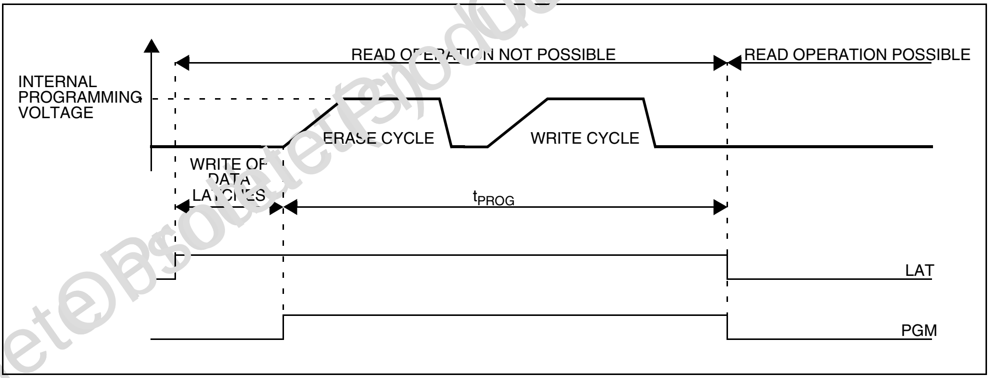
5.6 DATA EEPROM READ-OUT PROTECTION

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

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

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

Figure 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 5. Data EEPROM Register Map and Reset Values

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

6 CENTRAL PROCESSING UNIT

6.1 INTRODUCTION

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

6.2 MAIN FEATURES

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

6.3 CPU REGISTERS

The six CPU registers shown in Figure 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)

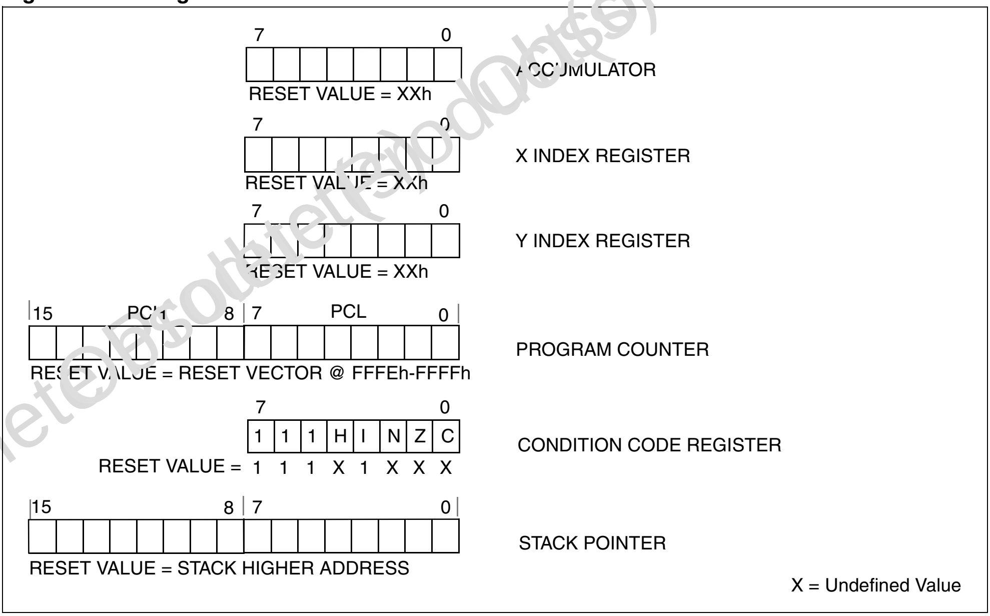
In indexed addressing modes, these 8-bit registers are used to create either effective addresses or temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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 interruptible because the I bit is set by hardware at the start of the routine and reset by the IRET instruction at the end of the routine. If the I bit is cleared by software in the interrupt routine, pending interrupts are serviced regardless of the priority level of the current interrupt routine.

Bit 2 = **N** *Negative*

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

This bit is accessed by the JRMI and JRPL instructions.

Bit 1 = **Z** *Zero*

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

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

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

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

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

CPU REGISTERS (Cont'd)

Stack Pointer (SP)

Read/Write

Reset Value: 00 FFh

15							8
0	0	0	0	0	0	0	0

7							0
1	1	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 64 bytes deep, the 10 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 SP5 to SP0 bits are set) which is the stack higher address.

The least significant byte of the Stack Pointer (called S) is directly accessed by an 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.

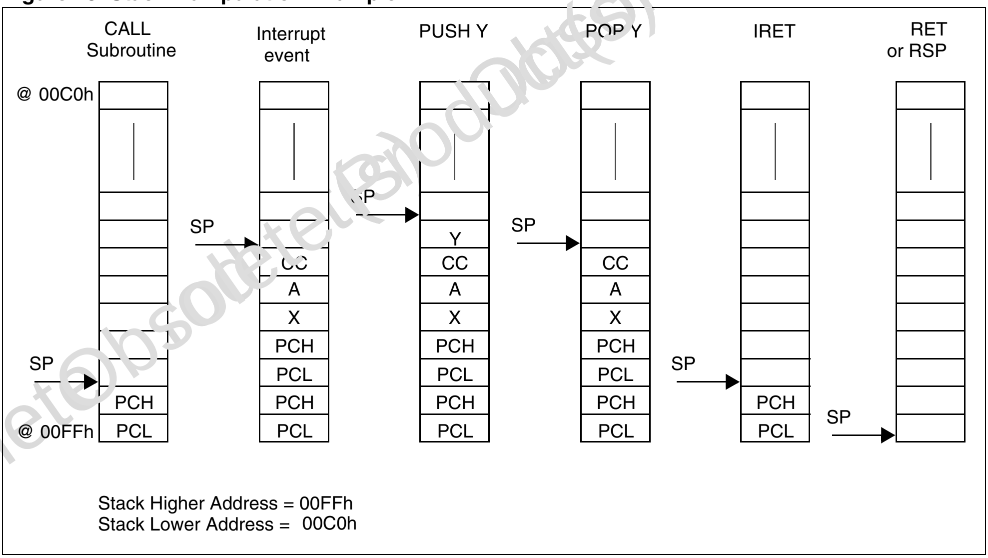
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the 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)
  - External Clock Input (enabled by option byte)
  - PLL for multiplying the frequency by 8
- Reset Sequence Manager (RSM)

7.1 INTERNAL RC OSCILLATOR ADJUSTMENT

The ST7 contains an internal RC oscillator with an accuracy of 1% for a given device, temperature and voltage. It must be calibrated to obtain the frequency required in the application. This is done by software writing a calibration value in the RCCR (RC Control Register).

Whenever the microcontroller is reset, the RCCR returns to its default value (FFh), that is, each time the device is reset, the calibration value must be loaded in the RCCR. The predefined calibration value is stored in EEPROM for 5V V_{DD} supply voltage at 25°C, as shown in the following table.

RCCR	Conditions	ST7FL09 Address	ST7FL05 Address
RCCR0	V_{DD} = 5V T_A = 25°C f_{RC} = 1 MHz	1000h and FFDEh	FFDEh
RCCR1	V_{DD} = 3.3V T_A = 25°C f_{RC} = 700 kHz	1001h and FFDFh	FFDFh

Notes:

- See "ELECTRICAL CHARACTERISTICS" on page 73 for more information on the frequency and accuracy of the RC oscillator.
- To improve clock stability and frequency accuracy, it is recommended to place a decoupling capacitor, typically 100nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device.
- This byte is systematically programmed by ST, including on FASTROM devices. Consequently, customers intending to use FASTROM service must not use this byte.
- RCCR0 and RCCR1 calibration values are erased if the read-out protection bit is reset after it has been set. See "Read-out Protection" on page 14.

Caution: If the voltage or temperature conditions change in the application, the frequency may require recalibration.

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

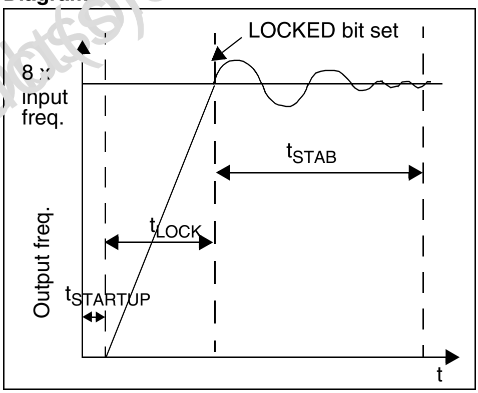
The x8 PLL is intended for operation with V_{DD} in the 3.6 to 5.5V range

Refer to Section 15.2 for the option byte description.

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

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

Figure 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 section 13.3.2 Internal RC Oscillator and PLL).

SUPPLY, RESET AND CLOCK MANAGEMENT (Cont'd)

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
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0

Bits 7:0 = **CR[7:0]** *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

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.

SYSTEM INTEGRITY (SI) CONTROL/STATUS REGISTER (SICSR)
Read/Write
Reset Value: 0000 0x00 (0xh)

7							0
0	0	0	0	LOCKED	0	0	0

Bits 7:4 = Reserved, must be kept cleared.

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

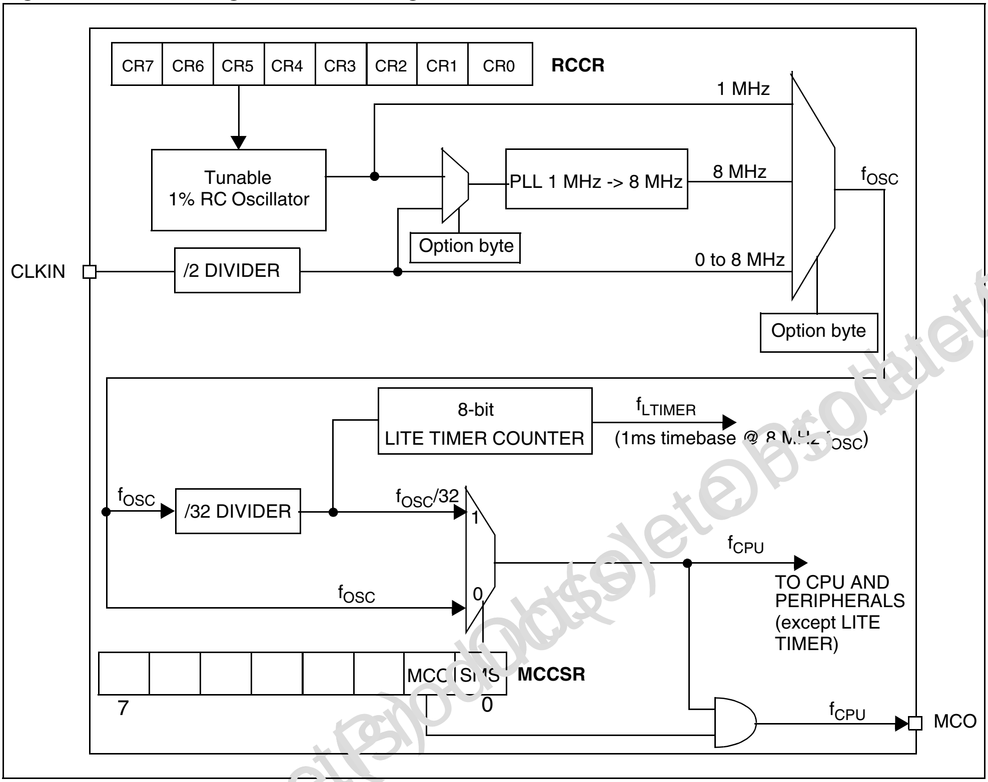
Bits 2:0 = Reserved, must be kept cleared.

Table 5. Clock Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0038h	**MCCSR** Reset Value	0	0	0	0	0	0	MCO 0	SMS 0
0039h	**RCCR** Reset Value	CR7 1	CR6 1	CR5 1	CR4 1	CR3 1	CR2 1	CR1 1	CR0 1
003Ah	**SICSR** Reset Value	0	0	0	0	LOCKED 0	0	0	0

SUPPLY, RESET AND CLOCK MANAGEMENT (Cont'd)

Figure 12. Clock Management Block Diagram



SUPPLY, RESET AND CLOCK MANAGEMENT (Cont'd)

7.4 RESET SEQUENCE MANAGER (RSM)

7.4.1 Introduction

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

- External $\overline{\text{RESET}}$ source pulse
- Internal WATCHDOG RESET

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to Section 12.2.1 on page 70 for further details.

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

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

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

- Active Phase depending on the RESET source
- 256 CPU clock cycle delay
- RESET vector fetch

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

The 256 CPU clock cycle delay allows the oscillator to stabilize and ensures that recovery has taken place from the Reset state.

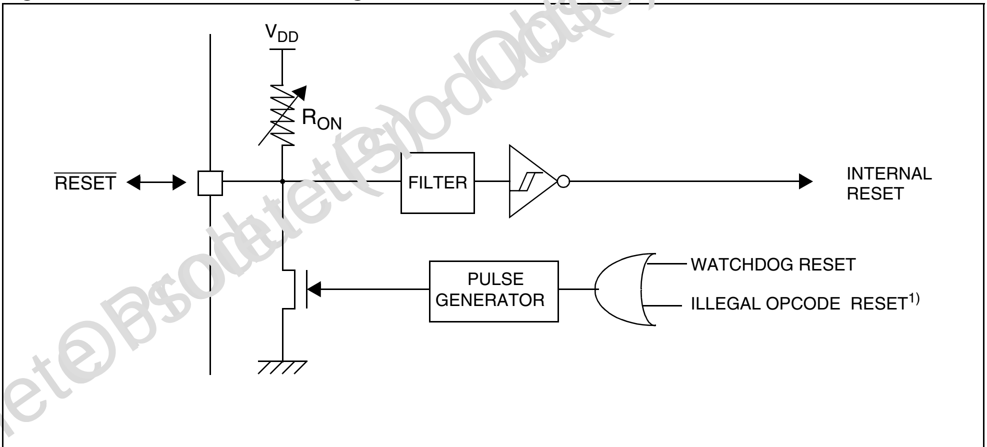
The RESET vector fetch phase duration is two clock cycles.

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

RESET		
Active Phase	INTERNAL RESET 256 CLOCK CYCLES	FETCH VECTOR

Figure 14. ST7L0x Reset Block Diagram



Note 1: See "Illegal Opcode Reset" on page 70. for more details on illegal opcode reset conditions.

SUPPLY, RESET AND CLOCK MANAGEMENT (Cont'd)

7.4.2 Asynchronous External $\overline{\text{RESET}}$ Pin

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It is pulled low by external circuitry to reset the device. See Electrical Characteristics 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. This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

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

7.4.3 External Power-On RESET

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

A proper reset signal for a slow rising V_{DD} supply can generally be provided by an external RC network connected to the $\overline{\text{RESET}}$ pin.

7.4.4 Internal Watchdog RESET

A RESET sequence is generated by a internal Watchdog counter overflow.

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

8 INTERRUPTS

The ST7 core may be interrupted by one of two different methods: Maskable hardware interrupts as listed in Table 7, "Interrupt Mapping," on page 29 and a non-maskable software interrupt (TRAP). The Interrupt processing flowchart is shown in Figure 15.

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 is cleared and the main program resumes.

Priority Management

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

In the case when several interrupts are simultaneously pending, an hardware priority defines which one will be serviced first (see 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 is serviced according to the flowchart in Figure 15.

8.2 EXTERNAL INTERRUPTS

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

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

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

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

8.3 PERIPHERAL INTERRUPTS

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

– The 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 (that is, waiting for being enabled) will therefore be lost if the clear sequence is executed.

INTERRUPTS (Cont'd)

Figure 15. Interrupt Processing Flowchart

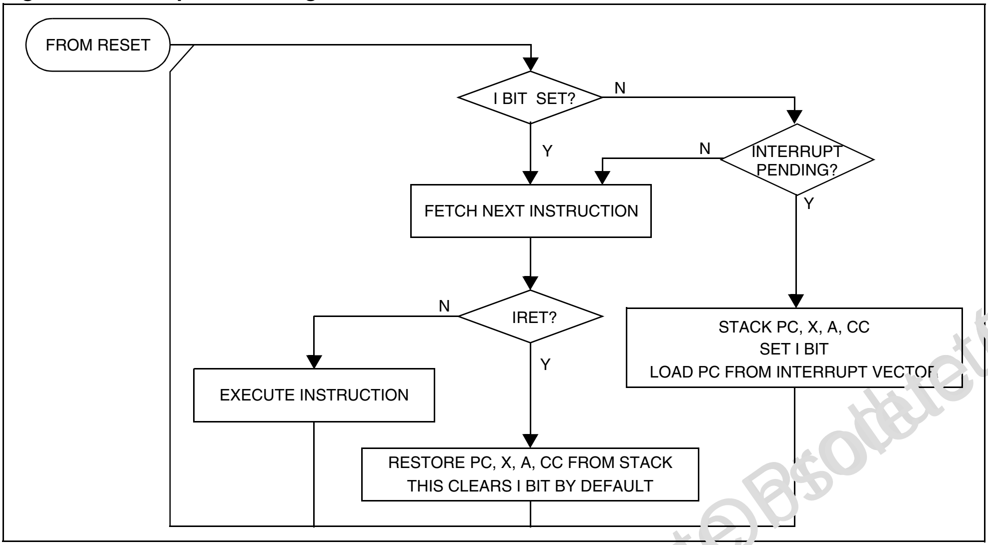


Table 7. Interrupt Mapping

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		Not used				FFFAh-FFFBh
1	ei0	External Interrupt 0			yes	FFF8h-FFF9h
2	ei1	External Interrupt 1				FFF6h-FFF7h
3	ei2	External Interrupt 2				FFF4h-FFF5h
4	ei3	External Interrupt 3				FFF2h-FFF3h
5		Not used				FFF0h-FFF1h
6		Not used				FFEEh-FFEFh
7		Not used				FFECh-FFEDh
8	AT TIMER	AT TIMER Output Compare Interrupt	PWM0CSR		no	FFEAh-FFEBh
9		AT TIMER Overflow Interrupt	ATCSR		yes	FFE8h-FFE9h
10	LITE TIMER	LITE TIMER Input Capture Interrupt	LTCSR		no	FFE6h-FFE7h
11		LITE TIMER RTC Interrupt	LTCSR		yes	FFE4h-FFE5h
12	SPI	SPI Peripheral Interrupts	SPICSR		yes	FFE2h-FFE3h
13		Not used		Lowest Priority		FFE0h-FFE1h

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

Bits 7:6 = **IS3[1:0]** *ei3 sensitivity*
These bits define the interrupt sensitivity for ei3 (Port B0) according to Table 8.

Bits 5:4 = **IS2[1:0]** *ei2 sensitivity*
These bits define the interrupt sensitivity for ei2 (Port B3) according to Table 8.

Bits 3:2 = **IS1[1:0]** *ei1 sensitivity*
These bits define the interrupt sensitivity for ei1 (Port A7) according to Table 8.

Bits 1:0 = **IS0[1:0]** *ei0 sensitivity*
These bits define the interrupt sensitivity for ei0 (Port A0) according to Table 8.

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

Table 8. 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

9 POWER SAVING MODES

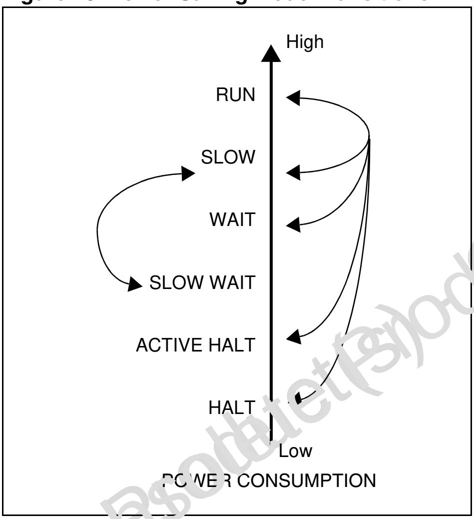
9.1 INTRODUCTION

To give a large measure of flexibility to the application in terms of power consumption, four main power saving modes are implemented in the ST7 (see Figure 16): SLOW, WAIT (SLOW WAIT), ACTIVE HALT and 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 (f_{OSC}).

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 16. 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.

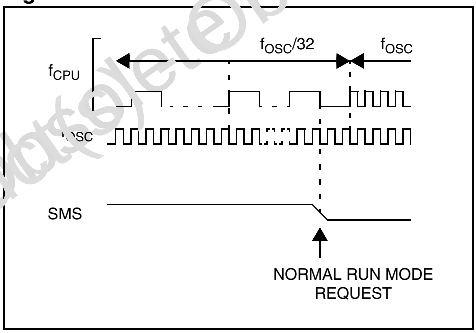
In this mode, the oscillator frequency is divided by 32. The CPU and peripherals are clocked at this lower frequency.

Notes:

SLOW-WAIT mode is activated when entering WAIT mode while the device is already in SLOW mode.

SLOW mode has no effect on the Lite Timer which is already clocked at $f_{OSC/32}$.

Figure 17. SLOW Mode Clock Transition



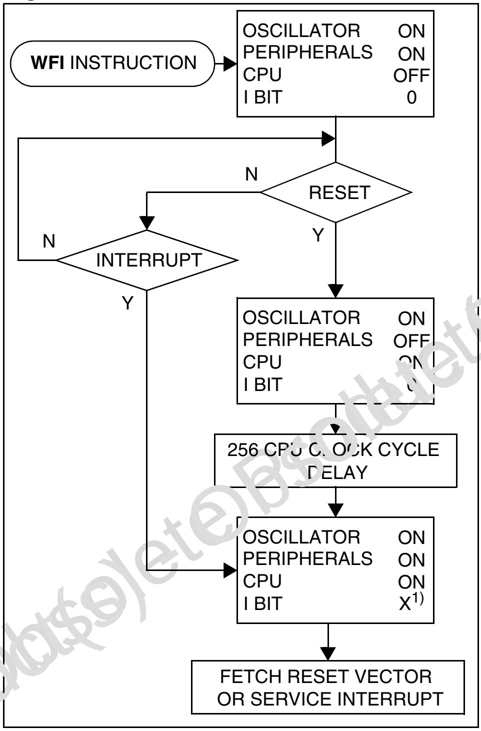
POWER SAVING MODES (Cont'd)

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 18.

Figure 18. WAIT Mode Flowchart



Notes:

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 ACTIVE HALT AND HALT MODES

ACTIVE HALT and HALT modes are the two lowest power consumption modes of the MCU. They are both 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:

LTCSR TBIE bit	ATCSR OVFIE bit	ATCSR CK1 bit	ATCSR CK0 bit	Meaning
0	x	x	0	ACTIVE HALT mode disabled
0	0	x	x	
0	1	1	1	
1	x	x	x	ACTIVE HALT mode enabled
x	1	0	1	

9.4.1 ACTIVE HALT Mode

ACTIVE HALT mode is the lowest power consumption mode of the MCU with a real time clock available. It is entered by executing the 'HALT' instruction when active halt mode is enabled.

The MCU can exit ACTIVE HALT mode on reception of a Lite Timer / AT Timer interrupt 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 20).

– 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 20).

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.

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).

Caution: As soon as ACTIVE HALT is enabled, executing a HALT instruction while the Watchdog is active does not generate a RESET if the WDGHALT bit is reset.
This means that the device cannot spend more than a defined delay in this power saving mode.

Figure 19. ACTIVE HALT Timing Overview

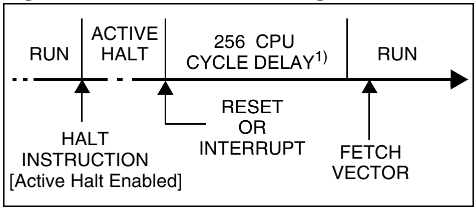
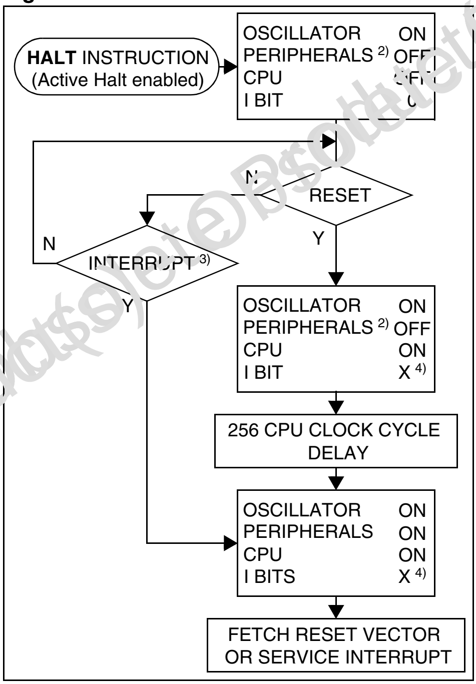


Figure 20. ACTIVE HALT Mode Flowchart



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 Lite Timer RTC and AT Timer interrupts can exit the MCU from ACTIVE HALT mode.
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)

9.4.2 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 mode is disabled.

The MCU can exit HALT mode on reception of either a specific interrupt (see Table 7, "Interrupt Mapping," on page 29) 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 22).

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 immediately.

In HALT mode, the main oscillator is turned off causing all internal processing to stop, 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 15.2 on page 96 for more details).

Figure 21. HALT Timing Overview

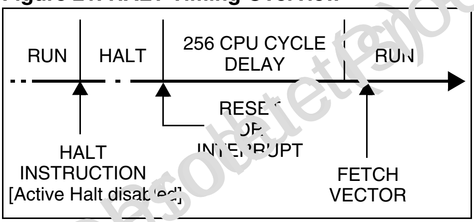
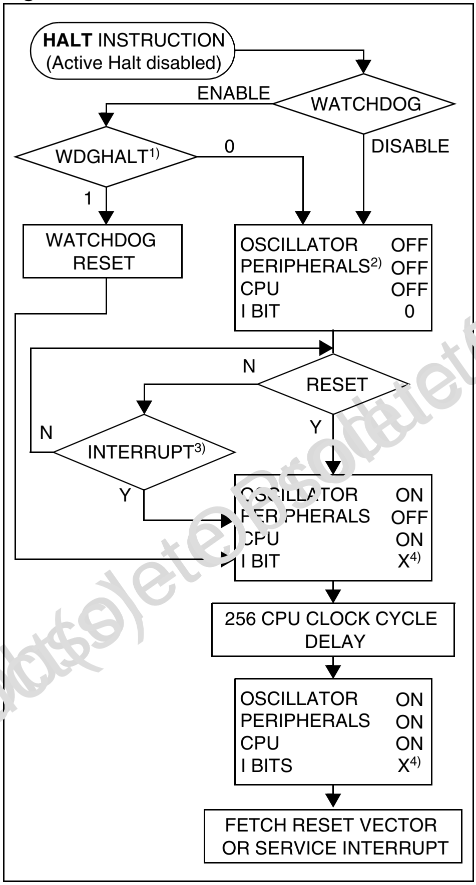


Figure 22. HALT Mode Flowchart



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 7, "Interrupt Mapping," on page 29 for more details.

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

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

POWER SAVING MODES (Cont'd)

9.4.2.1 HALT Mode Recommendations

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

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

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

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

– As the HALT instruction clears the I bit in the CC register to allow interrupts, the user may choose to clear all pending interrupt bits before executing the HALT instruction. This avoids entering other peripheral interrupt routines after executing the external interrupt routine corresponding to the wake-up event (reset or external interrupt).

10 I/O PORTS

10.1 INTRODUCTION

The I/O ports offer different functional modes:
– transfer of data through digital inputs and outputs

and for specific pins:
– external interrupt generation
– alternate signal input/output for the on-chip peripherals.

An I/O port contains up to eight pins. Each pin can be programmed independently as digital input (with or without interrupt generation) or digital output.

10.2 FUNCTIONAL DESCRIPTION

Each port has two main registers:

– Data Register (DR)

– Data Direction Register (DDR)

and one optional register:

– Option Register (OR)

Each I/O pin may be programmed using the corresponding register bits in the DDR and OR registers: bit X corresponding to pin X of the port. The same correspondence is used for the DR register.

The following description takes into account the OR register, (for specific ports which do not provide this register refer to the I/O Port Implementation section). The generic I/O block diagram is shown in Figure 23

10.2.1 Input Modes

The input configuration is selected by clearing the corresponding DDR register bit.

In this case, reading the DR register returns the digital value applied to the external I/O pin.

Different input modes can be selected by software through the OR register.

Note: Writing the DR register modifies the latch value but does not affect the pin status.

External interrupt function

When an I/O is configured as Input with Interrupt, an event on this I/O can generate an external interrupt request to the CPU.

Each pin can independently generate an interrupt request. The interrupt sensitivity is independently programmable using the sensitivity bits in the EICR register.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see pinout description and interrupt section). If several input pins are selected simultaneously as interrupt source, these are logically ANDed. For this reason if one of the interrupt pins is tied low, it may mask the others.

External interrupts are hardware interrupts. Fetching the corresponding interrupt vector automatically clears the request latch. Changing the sensitivity of a particular external interrupt clears this pending interrupt. This can be used to clear unwanted pending interrupts.

Spurious interrupts

When enabling/disabling an external interrupt by setting/resetting the related OR register bit, a spurious interrupt is generated if the pin level is low and its edge sensitivity includes falling/rising edge. This is due to the edge detector input which is switched to '1' when the external interrupt is disabled by the OR register.

To avoid this unwanted interrupt, a "safe" edge sensitivity (rising edge for enabling and falling edge for disabling) has to be selected before changing the OR register bit and configuring the appropriate sensitivity again.

Caution: In case a pin level change occurs during these operations (asynchronous signal input), as interrupts are generated according to the current sensitivity, it is advised to disable all interrupts before and to re-enable them after the complete previous sequence in order to avoid an external interrupt occurring on the unwanted edge.

This corresponds to the following steps:

1. To enable an external interrupt:
   - set the interrupt mask with the SIM instruction (in cases where a pin level change could occur)
   - select rising edge
   - enable the external interrupt through the OR register
   - select the desired sensitivity if different from rising edge
   - reset the interrupt mask with the RIM instruction (in cases where a pin level change could occur)
2. To disable an external interrupt:
   - set the interrupt mask with the SIM instruction SIM (in cases where a pin level change could occur)
   - select falling edge
   - disable the external interrupt through the OR

register

– select rising edge

reset the interrupt mask with the RIM instruction (in cases where a pin level change could occur)

10.2.2 Output Modes

The output configuration is selected by setting the corresponding DDR register bit. In this case, writing the DR register applies this digital value to the I/O pin through the latch. Then reading the DR register returns the previously stored value.

Two different output modes can be selected by software through the OR register: Output push-pull and open-drain.

DR register value and output pin status:

DR	Push-pull	Open-drain
0	V_{SS}	Vss
1	V_{DD}	Floating

Note: When switching from input to output mode, the DR register must be written first to drive the correct level on the pin as soon as the port is configured as an output.

10.2.3 Alternate Functions

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over the standard I/O programming under the following conditions:

– When the signal is coming from an on-chip peripheral, the I/O pin is automatically configured in output mode (push-pull or open drain according to the peripheral).

– When the signal is going to an on-chip peripheral, the I/O pin must be configured in floating input mode. In this case, the pin state is also digitally readable by addressing the DR register.

Notes:

– Input pull-up configuration can cause unexpected value at the input of the alternate peripheral input.

– When an on-chip peripheral uses a pin as input and output, this pin must be configured in input floating mode.

I/O PORTS (Cont'd)

Figure 23. I/O Port General Block Diagram

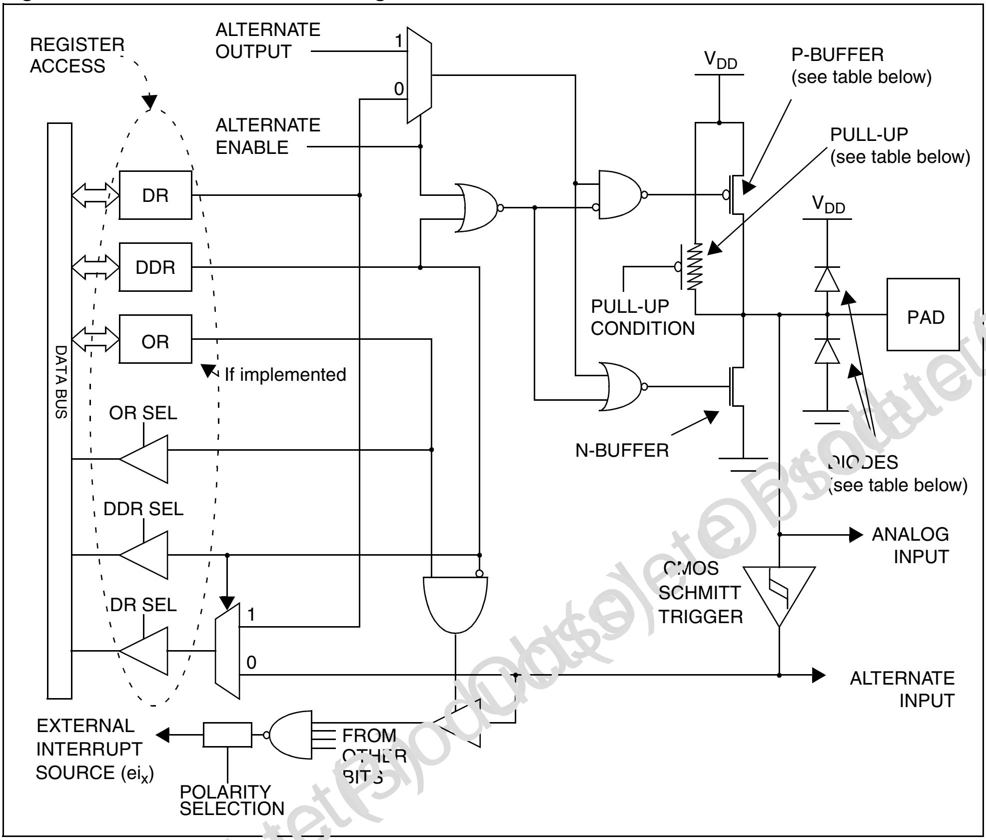


Table 9. I/O Port Mode Options

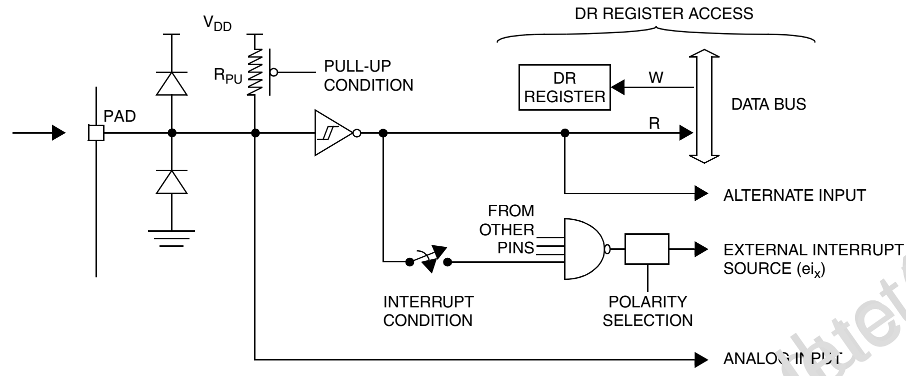
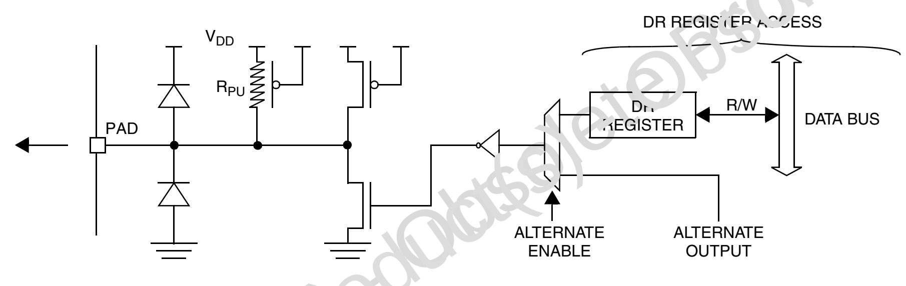
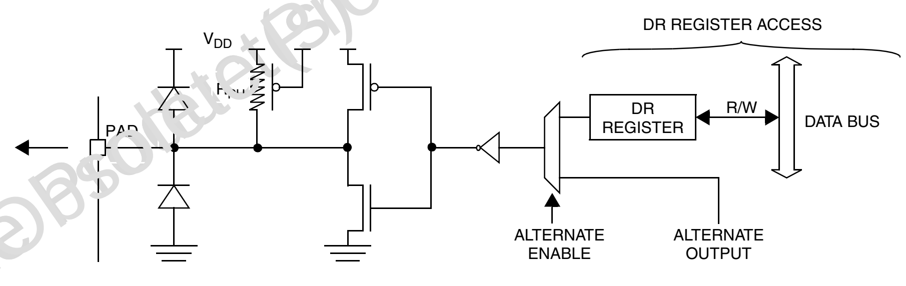
Configuration Mode		Pull-Up	P-Buffer	Diodes	
				to V_{DD}	to V_{SS}
Input	Floating with/without Interrupt	Off	Off	On	On
	Pull-up with/without Interrupt	On			
Output	Push-pull	Off	On		
	Open Drain (logic level)		Off		

Legend:

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

I/O PORTS (Cont'd)

Table 10. I/O Port 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.

I/O PORTS (Cont'd)

Caution: The alternate function must not be activated as long as the pin is configured as input with interrupt, in order to avoid generating spurious interrupts.

Analog alternate function

When the pin is used as an ADC input, the I/O must be configured as floating input. The analog multiplexer (controlled by the ADC registers) switches the analog voltage present on the selected pin to the common analog rail which is connected to the ADC input.

It is recommended not to change the voltage level or loading on any port pin while conversion is in progress. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

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

10.3 UNUSED I/O PINS

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

10.4 LOW POWER MODES

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

10.5 INTERRUPTS

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers and the interrupt mask in the CC register is not active (RIM instruction).

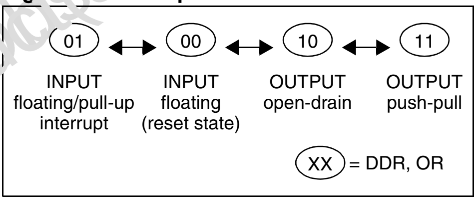
Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
External interrupt on selected external event	-	DDRx ORx	Yes	Yes

10.6 I/O PORT IMPLEMENTATION

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific feature of the I/O port such as ADC input.

Switching these I/O ports from one state to another must be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in Figure 24 Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 24. Interrupt I/O Port State Transitions



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

Table 11. Port Configuration

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

I/O PORTS (Cont'd)

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

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0000h	**PADR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0
0001h	**PADDR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0
0002h	**PAOR** Reset Value	MSB 0	1	0	0	0	0	0	LSB 0
0003h	**PBDR** Reset Value	MSB 1	1	1	0	0	0	0	LSB 0
0004h	**PBDDR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0
0005h	**PBOR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0

11 ON-CHIP PERIPHERALS

11.1 LITE TIMER (LT)

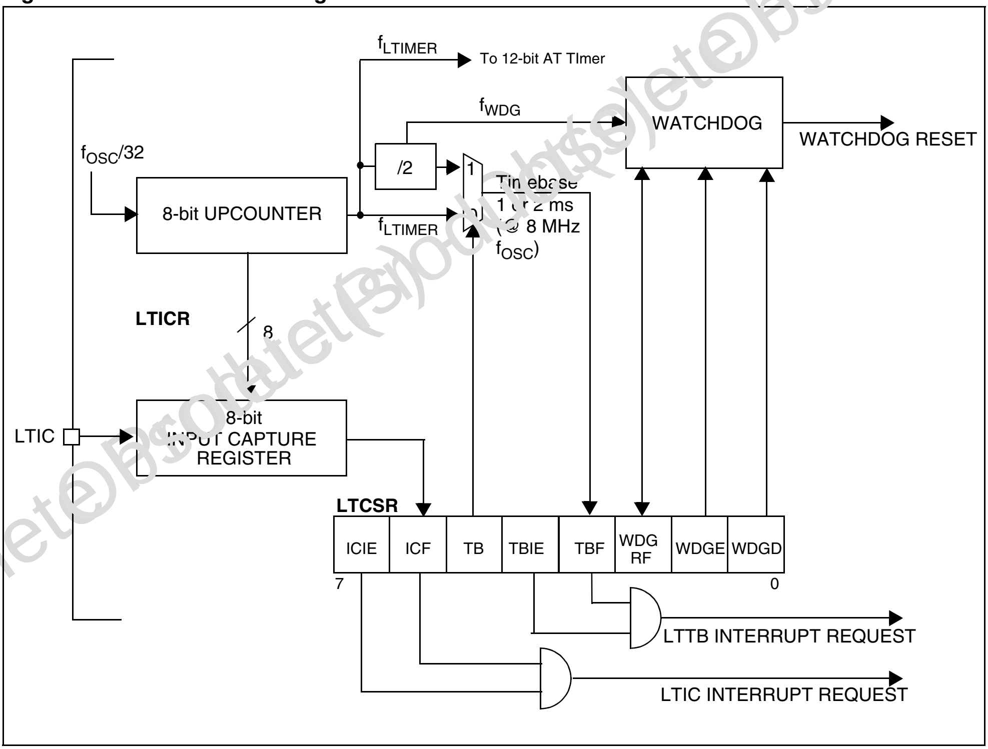
11.1.1 Introduction

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

11.1.2 Main Features

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

Figure 25. Lite Timer Block Diagram



LITE TIMER (Cont'd)

11.1.3 Functional Description

The value of the 8-bit counter cannot be read or written by software. After an MCU reset, it starts incrementing from 0 at a frequency of $f_{OSC}/32$. A counter 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 LTCSR register.

When the timer overflows, the TBF bit is set by hardware and an interrupt request is generated if the TBIE is set. The TBF bit is cleared by software reading the LTCSR register.

11.1.3.1 Watchdog

The watchdog is enabled using the WDGE bit. The normal Watchdog timeout is 2ms (@ = 8 MHz f_{OSC}), after which it then generates a reset.

To prevent this watchdog reset occuring, software must set the WDGD bit. The WDGD bit is cleared by hardware after t_{WDG}. This means that software must write to the WDGD bit at regular intervals to prevent a watchdog reset occurring. Refer to Figure 26.

If the watchdog is not enabled immediately after reset, the first watchdog timeout will be shorter than 2ms, because this period is counted starting from reset. Moreover, if a 2ms period has already elapsed after the last MCU reset, the watchdog reset will take place as soon as the WDGE bit is set. For these reasons, it is recommended to enable the Watchdog immediately after reset or else to set the WDGD bit before the WGDE bit so a watchdog reset will not occur for at least 2ms.

Note: Software can use the timebase feature to set the WDGD bit at 1 or 2 ms intervals.

A Watchdog reset can be forced at any time by setting the WDGRF bit. To generate a forced watchdog reset, first watchdog has to be activated by setting the WDGE bit and then the WDGRF bit has to be set.

The WDGRF bit also acts as a flag, indicating that the Watchdog was the source of the reset. It is automatically cleared after it has been read.

Caution: When the WDGRF bit is set, software must clear it, otherwise the next time the watchdog is enabled (by hardware or software), the microcontroller will be immediately reset.

Hardware Watchdog Option

If Hardware Watchdog is selected by option byte, the watchdog is always active and the WDGE bit in the LTCSR is not used.

Refer to the Option Byte description in the device configuration and ordering information section.

Using Halt Mode with the Watchdog (option)

If the Watchdog reset on HALT option is not selected by option byte, the Halt mode can be used when the watchdog is enabled.

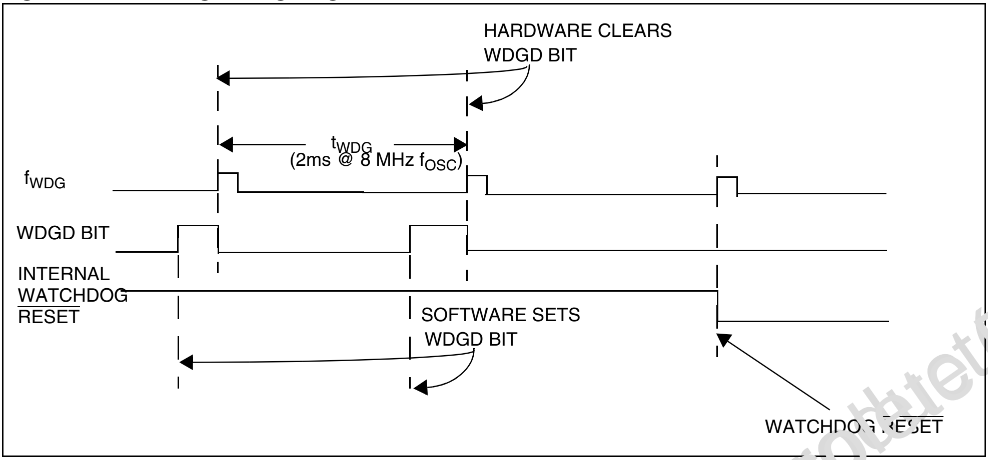
In this case, the HALT instruction stops the oscillator. When the oscillator is stopped, the Lite Timer stops counting and is no longer able to generate a Watchdog reset until the microcontroller receives an external interrupt or a reset.

If an external interrupt is received, the WDG restarts counting after 256 CPU clocks. If a reset is generated, the Watchdog is disabled (reset state).

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.

LITE TIMER (Cont'd)

Figure 26. Watchdog Timing Diagram



LITE TIMER (Cont'd)

Input Capture

The 8-bit input capture register is used to latch the free-running upcounter 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 the free-running upcounter. 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.

11.1.4 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

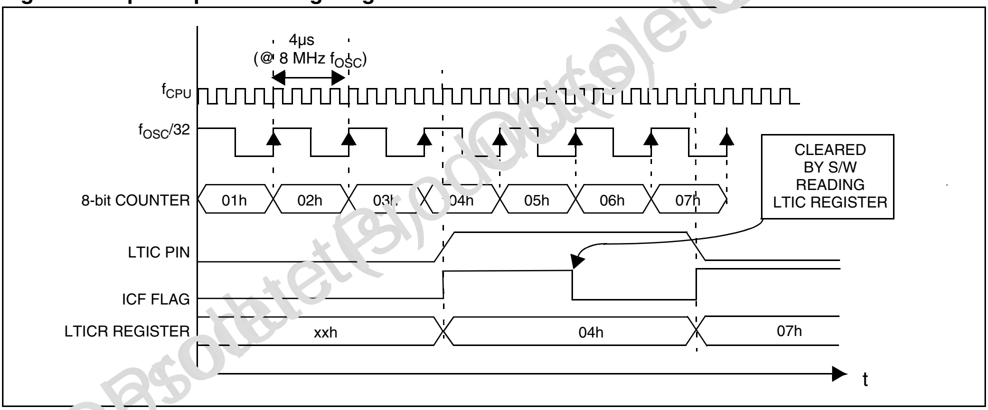
11.1.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt	Exit from Active-Halt
Timebase Event	TBF	TBIE	Yes	No	Yes
IC Event	ICF	ICIE			No

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

Timebase and IC events generate an interrupt if the enable bit is set in the LTCSR register and the interrupt mask in the CC register is reset (RIM instruction).

Figure 27. Input Capture Timing Diagram



LITE TIMER (Cont'd)

11.1.6 Register Description

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

7							0
ICIE	ICF	TB	TBIE	TBF	WDGR	WDGE	WDGD

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 = **TBIE** *Timebase Interrupt enable*
This bit is set and cleared by software.
0: Timebase (TB) interrupt disabled
1: Timebase (TB) interrupt enabled

Bit 3 = **TBF** *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

Bit 2 = **WDGRF** *Force Reset/ Reset Status Flag*
This bit is used in two ways: it is set by software to force a watchdog reset. It is set by hardware when a watchdog reset occurs and cleared by hardware or by software. It is cleared by hardware only when an LVD reset occurs. It can be cleared by software after a read access to the LTCSR register.
0: No watchdog reset occurred.
1: Force a watchdog reset (write), or, a watchdog reset occurred (read).

Bit 1 = **WDGE** *Watchdog Enable*
This bit is set and cleared by software.
0: Watchdog disabled
1: Watchdog enabled

Bit 0 = **WDGD** *Watchdog Reset Delay*
This bit is set by software. It is cleared by hardware at the end of each t_{WDG} period.
0: Watchdog reset not delayed
1: Watchdog reset delayed

LITE TIMER INPUT CAPTURE REGISTER (LTICR)
Read only
Reset Value: 0000 0000 (00h)

7							0
ICR7	ICR6	ICR5	ICR4	ICR3	ICR2	ICR1	ICR0

Bit 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.

Table 13. Lite Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0B	**LTCSR** Reset Value	ICIE 0	ICF x	TB 0	TBIE 0	TBF 0	WDGRF 0	WDGE 0	WDGD 0
0C	**LTICR** Reset Value	ICR7 0	ICR6 0	ICR5 0	ICR4 0	ICR3 0	ICR2 0	ICR1 0	ICR0 0

11.2 12-BIT AUTORELOAD TIMER (AT)

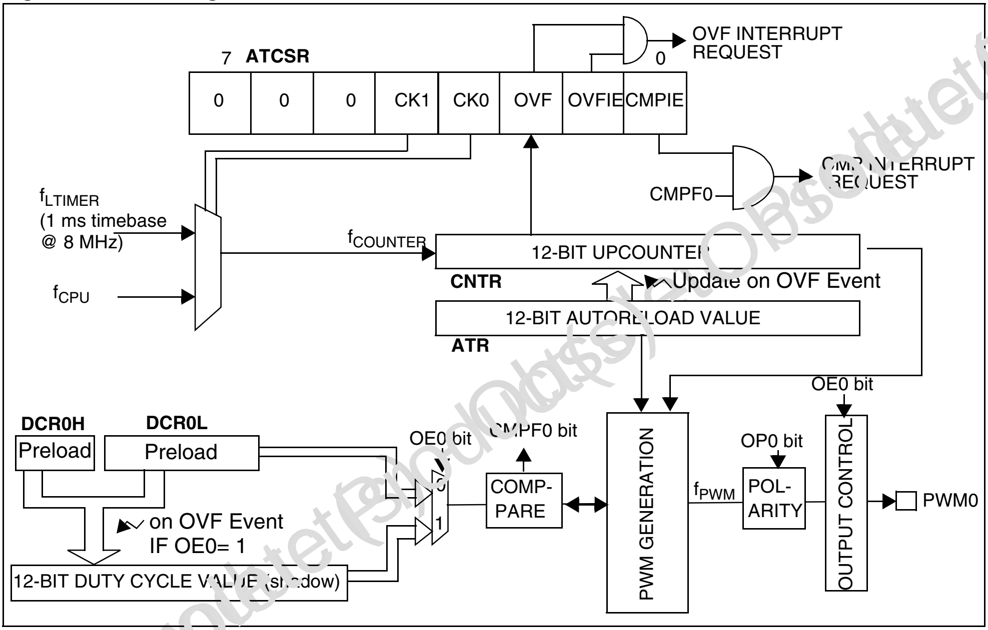
11.2.1 Introduction

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

11.2.2 Main Features

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

Figure 28. Block Diagram



12-BIT AUTORELOAD TIMER (Cont'd)

11.2.3 Functional Description

PWM Mode

This mode allows a Pulse Width Modulated signals generated on the PWM0 output pin with minimum core processing overhead. The PWM0 output signal can be enabled or disabled using the OE0 bit in the PWMCR register. When this bit is set the PWM I/O pin is configured as output push-pull alternate function.

Note: CMPF0 is available in PWM mode (see PWM0CSR description on page 51).

PWM Frequency and Duty Cycle

The PWM signal frequency (f_{PWM}) is controlled by the counter period and the ATR register value.

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

Following the above formula, if f_{CPU} is 8 MHz, the maximum value of f_{PWM} is 4 MHz (ATR register value = 4094), and the minimum value is 2 kHz (ATR register value = 0).

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

At reset, the counter starts counting from 0.

Software must write the duty cycle value in the DCR0H and DCR0L preload registers. The DCR0H register must be written first. See caution below.

When a upcounter overflow occurs (OVF event), the ATR value is loaded in the upcounter, the preloaded Duty cycle value is transferred to the Duty Cycle register and the PWM0 signal is set to a high level. When the upcounter matches the DCRx value the PWM0 signals is set to a low level. To obtain a signal on the PWM0 pin, the contents of the DCR0 register must be greater than the contents of the ATR register.

The polarity bit can be used to invert the output signal.

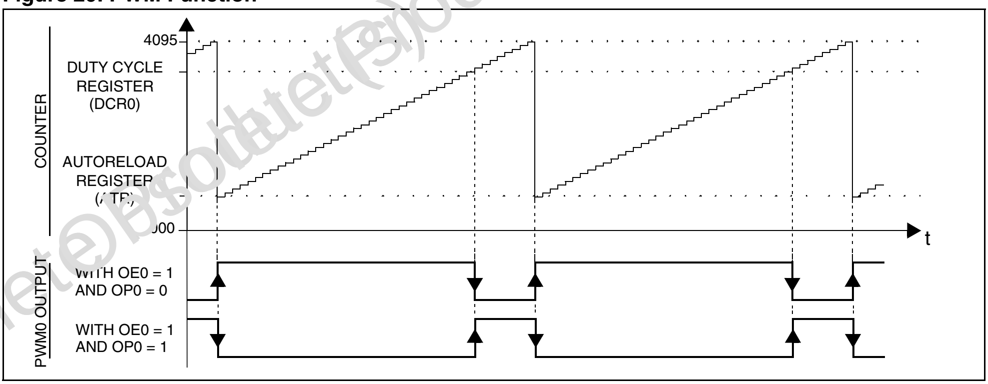
The maximum available resolution for the PWM0 duty cycle is:

Resolution = 1 / (4096 - ATR)

Note: To get the maximum resolution (1/4096), the ATR register must be 0. With this maximum resolution and assuming that DCR = ATR, a 0% or 100% duty cycle can be obtained by changing the polarity.

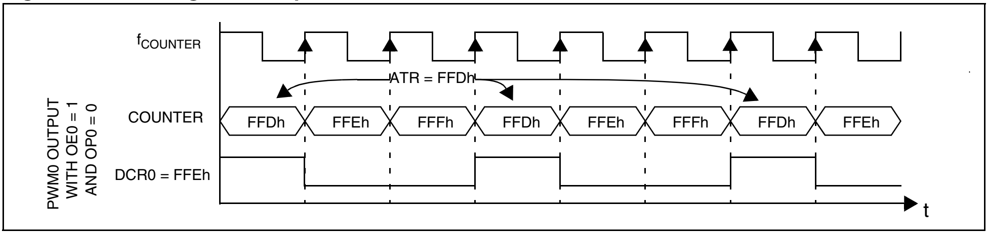
Caution: As soon as the DCR0H is written, the compare function is disabled and will start only when the DCR0L value is written. If the DCR0H write occurs just before the compare event, the signal on the PWM output may not be set to a low level. In this case, the DCRx register should be updated just after an OVF event. If the DCR and ATR values are close, then the DCRx register should be updated just before an OVF event, to avoid missing a compare event and to have the right signal applied on the PWM output.

Figure 29. PWM Function



12-BIT AUTORELOAD TIMER (Cont'd)

Figure 30. PWM Signal Example



Output Compare Mode

To use this function, the OE bit must be 0, otherwise the compare is done with the shadow register instead of the DCRx register. Software must then write a 12-bit value in the DCR0H and DCR0L registers. This value is loaded immediately (without waiting for an OVF event).

The DCR0H must be written first, the output compare function starts only when the DCR0L value is written.

When the 12-bit upcounter (CNTR) reaches the value stored in the DCR0H and DCR0L registers, the CMPF0 bit in the PWM0CSR register is set and an interrupt request is generated if the CMPIE bit is set.

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

Caution: At each OVF event, the DCRx value is written in a shadow register, even if the DCR0L value has not yet been written (in this case, the shadow register will contain the new DCR0H value and the old DCR0L value), then:

– If OE = 1 (PWM mode): The compare is done between the timer counter and the shadow register (and not DCRx).

– if OE = 0 (OCMP mode): The compare is done between the timer counter and DCRx. There is no PWM signal.

The compare between DCRx or the shadow register and the timer counter is locked until DCR0L is written.

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	OVF	OVFIE	Yes	No	Yes[2]
CMP Event	CMPFx	CMPIE	Yes	No	No

Notes:

1. The interrupt events are connected to separate interrupt vectors (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).

2. Only if CK0 = 1 and CK1 = 0.

12-BIT AUTORELOAD TIMER (Cont'd)

11.2.6 Register Description

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

7							0
0	0	0	CK1	CK0	OVF	OVFIE	CMPIE

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

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

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

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

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

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

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

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

15							8
0	0	0	0	CN11	CN10	CN9	CN8

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

7							0
CN7	CN6	CN5	CN4	CN3	CN2	CN1	CN0

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

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

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

12-BIT AUTORELOAD TIMER (Cont'd)

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

15							8
0	0	0	0	ATR11	ATR10	ATR9	ATR8

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

7							0
ATR7	ATR6	ATR5	ATR4	ATR3	ATR2	ATR1	ATR0

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

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

PWM0 DUTY CYCLE REGISTER HIGH (DCR0H)
Read / Write
Reset Value: 0000 0000 (00h)

15							8
0	0	0	0	DCR11	DCR10	DCR9	DCR8

PWM0 DUTY CYCLE REGISTER LOW (DCR0L)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
DCR7	DCR6	DCR5	DCR4	DCR3	DCR2	DCR1	DCR0

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

Bits 11:0 = **DCR[11:0]** *PWMx Duty Cycle Value*
This 12-bit value is written by software. The high register must be written first.

In PWM mode (OE0 = 1 in the PWMCR register) the DCR[11:0] bits define the duty cycle of the PWM0 output signal (see Figure 29). In Output Compare mode, (OE0 = 0 in the PWMCR register) they define the value to be compared with the 12-bit upcounter value.

PWM0 CONTROL/STATUS REGISTER (PWM0CSR)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	OP0	CMPF0

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

Bit 1 = **OP0** *PWM0 Output Polarity*
This bit is read/write by software and cleared by hardware after a reset. This bit selects the polarity of the PWM0 signal.
0: The PWM0 signal is not inverted.
1: The PWM0 signal is inverted.

Bit 0 = **CMPF0** PWM0 *Compare Flag.*
This bit is set by hardware and cleared by software by reading the PWM0CSR register. It indicates that the upcounter value matches the DCR0 register value.
0: Upcounter value does not match DCR value.
1: Upcounter value matches DCR value.

12-BIT AUTORELOAD TIMER (Cont'd)

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

7							0
0	0	0	0	0	0	0	OE0

Bits 7:1 = Reserved, must be kept cleared.

Bit 0 = **OE0** *PWM0 Output enable*
This bit is set and cleared by software.
0: PWM0 output Alternate Function disabled (I/O pin free for general purpose I/O)
1: PWM0 output enabled

Table 14. Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0D	**ATCSR** Reset Value	0	0	0	CK1 0	CK0 0	OVF 0	OVFIE 0	CMPIE 0
0E	**CNTRH** Reset Value	0	0	0	0	CN11 0	CN10 0	CN9 0	CN8 0
0F	**CNTRL** Reset Value	CN7 0	CN6 0	CN5 0	CN4 0	CN3 0	CN2 0	CN1 0	CN0 0
10	**ATRH** Reset Value	0	0	0	0	ATR11 0	ATR10 0	ATR9 0	ATR8 0
11	**ATRL** Reset Value	ATR7 0	ATR6 0	ATR5 0	ATR4 0	ATR3 0	ATR2 0	ATR1 0	ATR0 0
12	**PWMCR** Reset Value	0	0	0	0	0	0	0	OE0 0
13	**PWM0CSR** Reset Value	0	0	0	0	0	0	OP 0	CMPF0 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

11.3 SERIAL PERIPHERAL INTERFACE (SPI)

11.3.1 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 however the SPI interface can not be a master in a multi-master system.

11.3.2 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.

11.3.3 General Description

Figure 31 shows the serial peripheral interface (SPI) block diagram for three registers:

– SPI Control Register (SPICR)

– SPI Control/Status Register (SPICSR)

– SPI Data Register (SPIDR)

The SPI is connected to external devices through three pins:

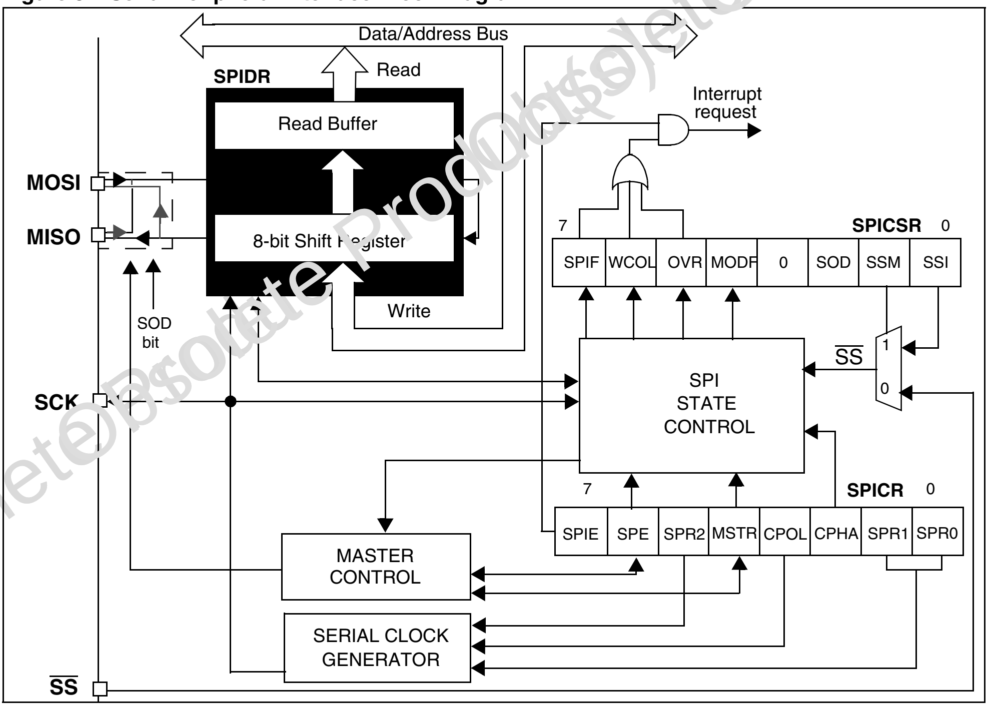
– MISO: Master In / Slave Out data

– 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 MCU.

Figure 31. Serial Peripheral Interface Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.3.1 Functional Description

A basic example of interconnections between a single master and a single slave is illustrated in Figure 32.

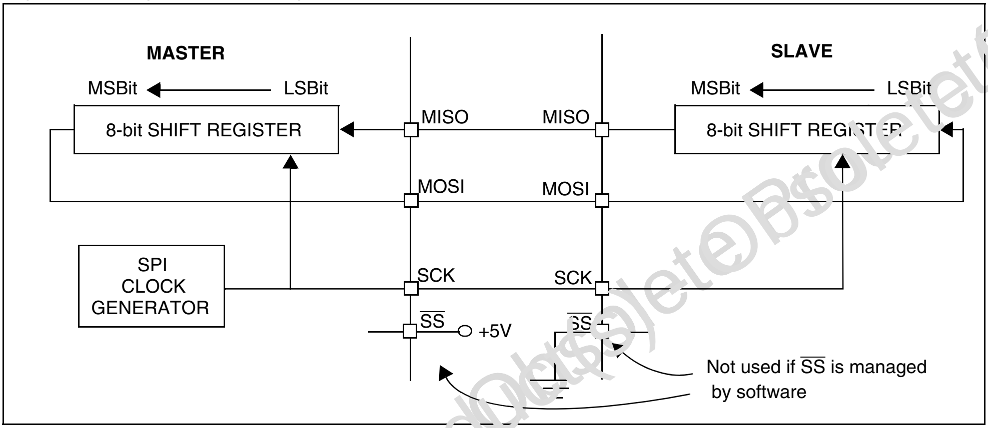
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

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

Figure 32. Single Master/ Single Slave Application



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.3.2 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 34)

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:

Two cases depend on the data/clock timing relationship (see Figure 33):

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 11.3.5.3).

Figure 33. Generic $\overline{SS}$ Timing Diagram

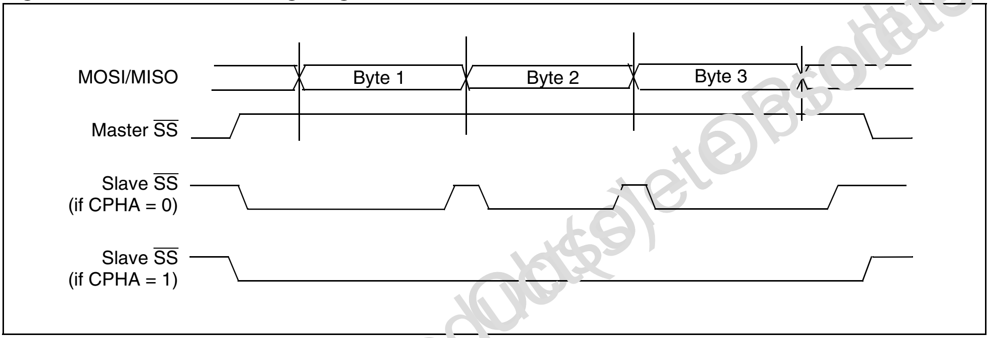
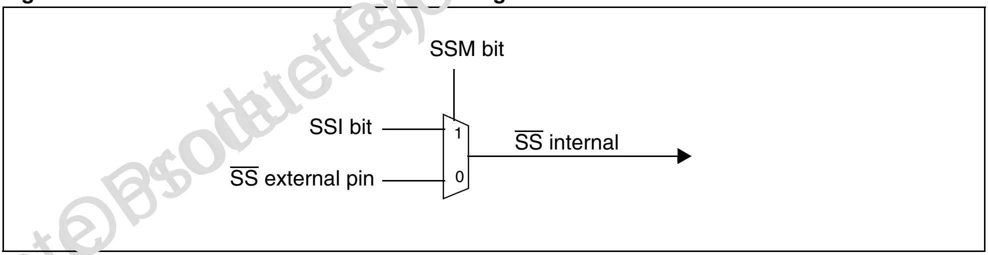


Figure 34. Hardware/Software Slave Select Management



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.3.3 Master Mode Operation

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

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

How to operate the SPI in master mode

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

1. Write to the SPICR register:
   – Select the clock frequency by configuring the SPR[2:0] bits.
   – Select the clock polarity and clock phase by configuring the CPOL and CPHA bits. Figure 35 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.

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

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

11.3.3.4 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.

11.3.3.5 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 35).
   Note: The slave must have the same CPOL and CPHA settings as the master.
   – Manage the $\overline{SS}$ pin as described in Section 11.3.3.2 and Figure 33. If CPHA = 1 $\overline{SS}$ must be held low continuously. If CPHA = 0 $\overline{SS}$ must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

11.3.3.6 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 11.3.5.2).

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.4 Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 35).

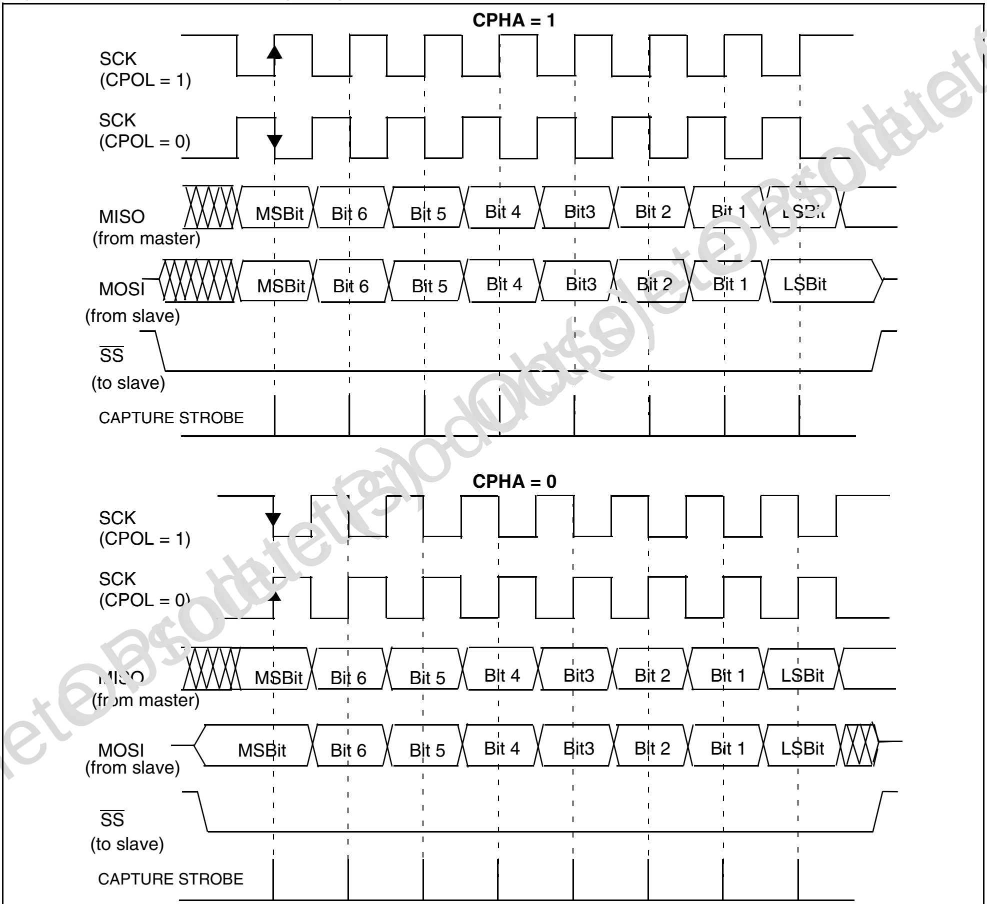
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL = 1 or pulling down SCK if CPOL = 0).

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

Figure 35, 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 when 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 35. 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)

11.3.5 Error Flags

11.3.5.1 Master Mode Fault (MODF)

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

11.3.5.2 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.

11.3.5.3 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 is unsuccessful.

Write collisions can occur both in master and slave mode. See also Section 11.3.3.2 Slave Select Management.

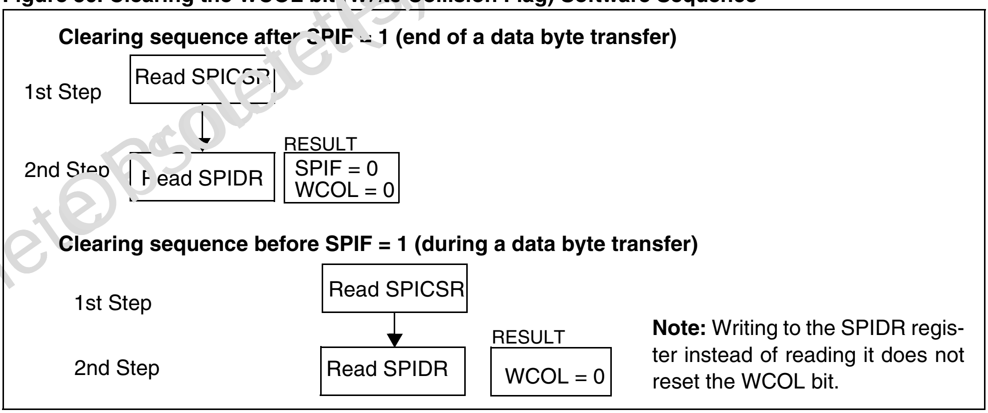
Note: A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation.

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 36).

Figure 36. Clearing the WCOL bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.5.4 Single Master Systems

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see Figure 37).

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.

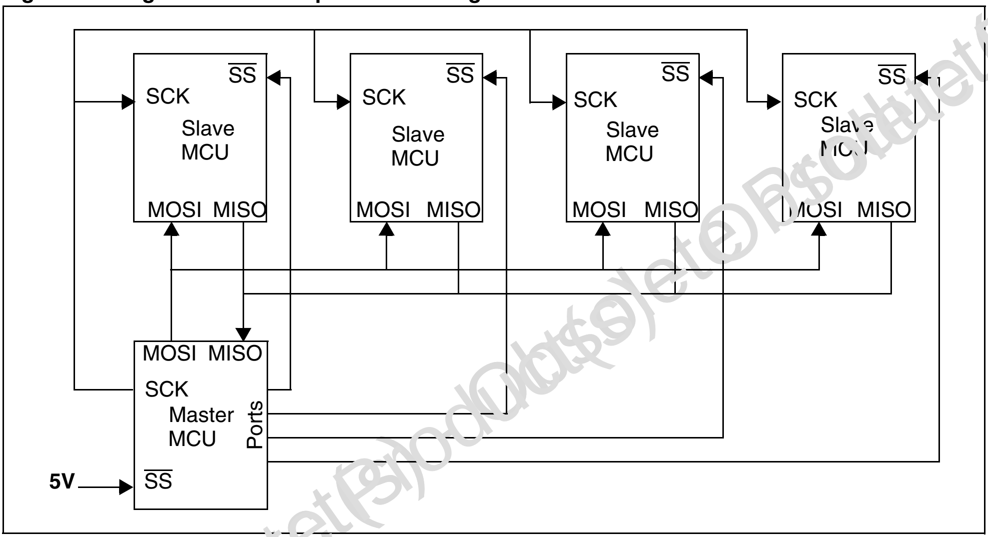
The $\overline{SS}$ pins are pulled high during reset since the master device ports are forced as inputs at that time, thus disabling the slave devices.

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.

Figure 37. Single Master / Multiple Slave Configuration



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.3.6 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 MCU 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 data is 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.

11.3.6.1 Using the SPI to wakeup the MCU from Halt mode

In slave configuration, the SPI is able to wakeup the ST7 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 ST7 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 ST7 enters Halt mode. So if Slave selection is configured as external (see Section 11.3.3.2), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters Halt mode.

11.3.7 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			No
Overrun Error	OVR			

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)

11.3.8 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 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 11.3.5.1 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 15 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 11.3.5.1 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 15. 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 36).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = **OVR** *SPI Overrun error (Read only)*
This bit is set by hardware when the byte currently being received in the shift register is ready for transfer into the SPIDR register while SPIF = 1 (See Section 11.3.5.2). An interrupt is generated if SPIE = 1 in the SPICR register. The OVR bit is cleared by software reading the SPICSR register.
0: No overrun error
1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only)*
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 11.3.5.1 Master Mode Fault (MODF)). An SPI interrupt can be generated if SPIE = 1 in the SPICR register. This bit is cleared by a software sequence (An access to the SPICSR register while MODF = 1 followed by a write to the SPICR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

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

Bit 1 = **SSM** *$\overline{SS}$ Management*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI $\overline{SS}$ pin and uses the SSI bit value instead. See Section 11.3.3.2 Slave Select Management.
0: Hardware management ($\overline{SS}$ managed by external pin)
1: Software management (internal $\overline{SS}$ signal controlled by SSI bit. External $\overline{SS}$ pin free for general-purpose I/O)

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

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 31).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 16. SPI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
31	**SPIDR** Reset Value	MSB x	 x	 x	 x	 x	 x	 x	LSB x
32	**SPICR** Reset Value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
33	**SPICSR** Reset Value	SPIF 0	WCOL 0	OVR 0	MODF 0	 0	SOD 0	SSM 0	SSI 0

11.4 8-BIT A/D CONVERTER (ADC)

11.4.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 8-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to five multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to five different sources.

The result of the conversion is stored in a 8-bit Data Register. The A/D converter is controlled through a Control/Status Register.

11.4.2 Main Features

– 8-bit conversion

– Up to 5 channels with multiplexed input

– Linear successive approximation

– Data register (DR) which contains the results

– Conversion complete status flag

– On/off bit (to reduce consumption)

11.4.3 Functional Description

11.4.3.1 Analog Power Supply

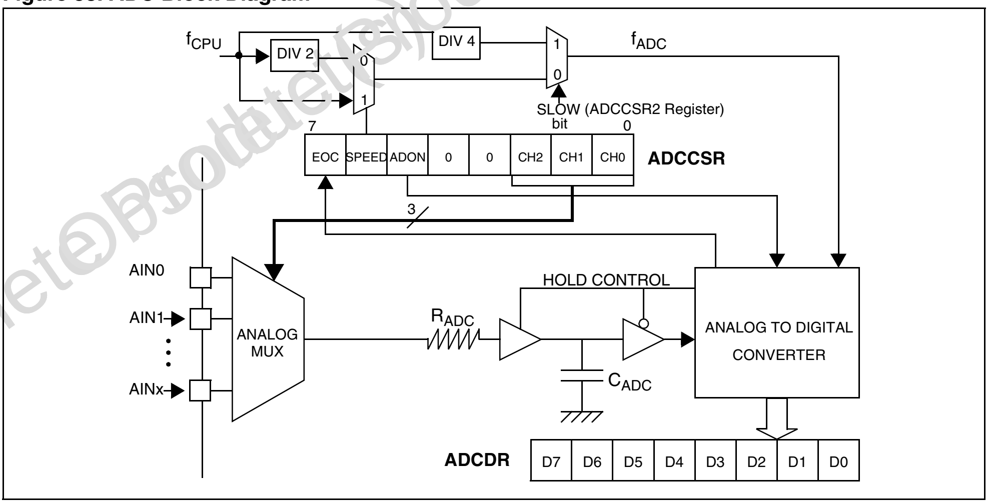
The block diagram is shown in Figure 38.

V_{DD} and V_{SS} are the high and low level reference voltage pins.

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

For more details, refer to Section 13 ELECTRICAL CHARACTERISTICS.

Figure 38. ADC Block Diagram



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

11.4.3.2 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 or equal to V_{DDA} (high-level voltage reference) then the conversion result in the DR register is FFh (full scale) without overflow indication.

If input voltage (V_{AIN}) is lower than or equal to V_{SSA} (low-level voltage reference) then the conversion result in the DR register is 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDR register. The accuracy of the conversion is described in the parametric 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 allotted time.

11.4.3.3 A/D Conversion Phases

The A/D conversion is based on two conversion phases as shown in Figure 39:

- Sample capacitor loading [duration: t_{SAMPLE}]
  During this phase, the V_{AIN} input voltage to be measured is loaded into the C_{ADC} sample capacitor.
- A/D conversion [duration: t_{HOLD}]
  During this phase, the A/D conversion is computed (8 successive approximations cycles) and the C_{ADC} sample capacitor is disconnected from the analog input pin to get the optimum analog to digital conversion accuracy.
- The total conversion time:
  $t_{CONV} = t_{SAMPLE} + t_{HOLD}$

While the ADC is on, these two phases are continuously repeated.

At the end of each conversion, the sample capacitor is kept loaded with the previous measurement load. The advantage of this behavior is that it minimizes the current consumption on the analog pin in case of single input channel measurement.

11.4.3.4 Software Procedure

Refer to the control/status register (CSR) and data register (DR) in Section 11.4.6 for the bit definitions and to Figure 39 for the timings.

ADC Configuration

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 CSR register:

– Select the CH[2:0] bits to assign the analog channel to be converted.

ADC Conversion

In the CSR register:

– Set the ADON bit to enable the A/D converter and to start the first conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

When a conversion is complete

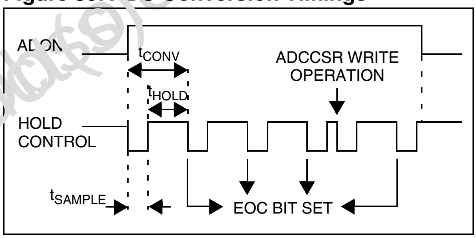
– The EOC bit is set by hardware.

– No interrupt is generated.

– The result is in the DR register and remains valid until the next conversion has ended.

A write to the ADCCSR register (with ADON set) aborts the current conversion, resets the EOC bit and starts a new conversion.

Figure 39. ADC Conversion Timings



11.4.4 Low Power Modes

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 before accurate conversions can be performed.

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.

11.4.5 Interrupts

None

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

11.4.6 Register Description

CONTROL/STATUS REGISTER (ADCCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
EOC	SPEED	ADON	0	0	CH2	CH1	CH0

Bit 7 = **EOC** *Conversion Complete*
This bit is set by hardware. It is cleared by software reading the result in the DR register or writing to the CSR register.
0: Conversion is not complete
1: Conversion can be read from the DR register

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 always be cleared.*

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

Channel Pin[1)]	CH2	CH1	CH0
AIN0	0	0	0
AIN1	0	0	1
AIN2	0	1	0
AIN3	0	1	1
AIN4	1	0	0

Notes:
1. The number of pins AND the channel selection varies according to the device. Refer to the device pinout.
2. A write to the ADCCSR register (with ADON set) aborts the current conversion, resets the EOC bit and starts a new conversion.

DATA REGISTER (ADCDR)

Read Only

Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0 = **D[7:0]** *Analog Converted Value*
This register contains the converted analog value in the range 00h to FFh.

Note: Reading this register resets the EOC flag.

CONTROL/STATUS REGISTER 2 (ADCCSR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	SLOW	0	0	0

Bit 7:4 = 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 2:0 = Reserved. Forced by hardware to 0.

Note: If ADC settings are changed by writing the ADCCSR2 register while the ADC is running, a dummy conversion is needed before obtaining results with the new settings.

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

Table 17. ADC Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
34h	**ADCCSR** Reset Value	EOC 0	SPEED 0	ADON 0	 0	 0	CH2 0	CH1 0	CH0 0
35h	**ADCDR** Reset Value	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0	D1 0	D0 0
36h	**ADCCSR2** Reset Value	0	0	0	0	SLOW 0	0	0	0

12 INSTRUCTION SET

12.1 ST7 ADDRESSING MODES

The ST7 Core features 17 different addressing modes which can be classified in seven 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 submodes 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 18. 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	Subroutine Return
IRET	Interrupt Subroutine Return
SIM	Set Interrupt Mask
RIM	Reset Interrupt Mask
SCF	Set Carry Flag
RCF	Reset Carry Flag
RSP	Reset Stack Pointer
LD	Load
CLR	Clear
PUSH/POP	Push/Pop to/from the stack
INC/DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
MUL	Byte Multiplication
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles

12.1.2 Immediate

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

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 1 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 submodes:

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 1 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 submodes:

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 submodes:

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 19. 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 submodes:

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 prebyte

The instructions are described with 1 to 4 bytes.

In order to extend the number of available opcodes for an 8-bit CPU (256 opcodes), three different prebyte opcodes are defined. These prebytes modify the meaning of the instruction they precede.

The whole instruction becomes:

PC-2 End of previous instruction

PC-1 Prebyte

PC Opcode

PC+1 Additional word (0 to 2) according to the number of bytes required to compute the effective address

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

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

PIX 92 Replace an instruction using direct, direct bit or direct relative addressing mode to an instruction using the corresponding indirect addressing mode.
It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.

PIY 91 Replace an instruction using X indirect indexed addressing mode by a Y one.

12.2.1 Illegal Opcode Reset

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

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

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 is based on $T_A = 25°C$, $V_{DD} = 5V$ (for the $4.5V \le V_{DD} \le 5.5V$ voltage range), $V_{DD} = 3.3V$ (for the $3V \le V_{DD} \le 3.6V$ voltage range). They are given only as design guidelines and are not tested.

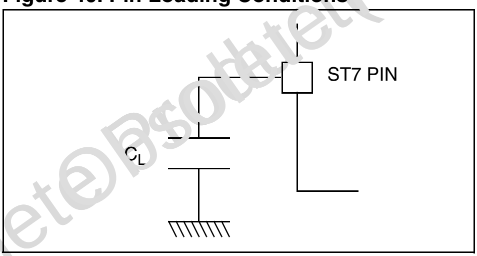
13.1.3 Typical Curves

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 40.

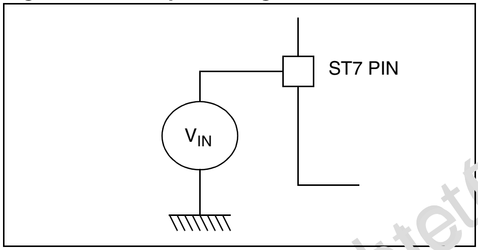
Figure 40. Pin Loading Conditions



13.1.5 Pin Input Voltage

The input voltage measurement on a pin of the device is described in Figure 41.

Figure 41. 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 13.7.2 on page 82	
$V_{ESD(MM)}$	Electrostatic discharge voltage (Machine Model)	see Section 13.7.2 on page 82	

13.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source)[3)]	75	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	20	
	Output current sunk by any high sink I/O pin	40	
	Output current source by any I/Os and control pin	25	
$I_{INJ(PIN)}$[2)4)]	Injected current on $\overline{RESET}$ pin	±5	
	Injected current on PB1 pin[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 THERMAL CHARACTERISTICS)		

Notes:

1. Directly connecting the I/O pins to V_{DD} or V_{SS} could damage the device if an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection must be done through a pull-up or pull-down resistor (typical: 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. For reset pin, please refer to Figure 64.
2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN} > V_{DD}$ while a negative injection is induced by $V_{IN} < V_{SS}$.
3. 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 0.8mA (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 PB1 pin.
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 characterization with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

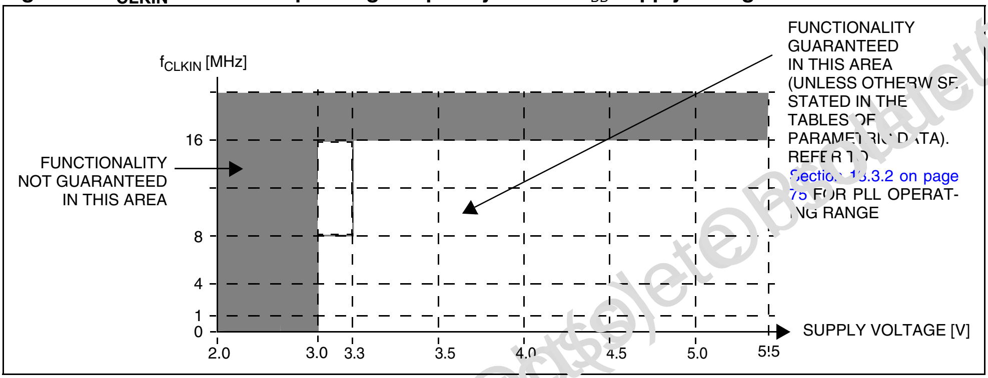
13.3 OPERATING CONDITIONS

13.3.1 General Operating Conditions

T_A = -40 to +105°C, unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	Supply voltage	f_{OSC} = 16 MHz max T_A = -40°C to T_A max	3.0	5.5	V
f_{CLKIN}	External clock frequency on CLKIN pin	$V_{DD} \geq 3V$	0	16	MHz
T_A	Ambient temperature range	A Suffix version	-40	+85	°C
		B Suffix version		+105	

Figure 42. f_{CLKIN} Maximum Operating Frequency Versus V_{DD} Supply Voltage



Note: For further information on clock management and f_{CLKIN} description, refer to Figure 12 in Section 7 on page 23.

13.3.2 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		3.0		5.5	V
$V_{DD(x8PLL)}$	x8 PLL operating voltage		3.6			
$t_{STARTUP}$	PLL startup time			60		PLL input clock (f_{PLL}) cycles

OPERATING CONDITIONS (Cont'd)

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

Operating conditions (tested for T_A = -40 to +105°C) @ V_{DD} = 4.5 to 5.5V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC} [1]	Internal RC oscillator frequency	RCCR = FF (reset value), T_A = 25°C, V_{DD} = 5V		760		kHz
		RCCR = RCCR0 [2], T_A = 25°C, V_{DD} = 5V	995	1000	1005	
ACC_{RC}	Accuracy of internal RC oscillator when calibrated with RCCR = RCCR0 [2][3]	T_A = 25°C, V_{DD} = 5V	-0.5		+0.5	%
		T_A = 25°C, V_{DD} = 4.5 to 5.5V [4]	-1		+1	
		T_A = -40 to +105°C, V_{DD} = 4.5 to 5.5V [4]	-5		+2	
$I_{DD(RC)}$	RC oscillator current consumption	T_A = 25°C, V_{DD} = 5V		970 [4][5]		µA
$t_{su(RC)}$	RC oscillator setup time	T_A = 25°C, V_{DD} = 5V			10 [2]	µs
f_{PLL}	x8 PLL input clock			1		MHz
t_{LOCK}	PLL lock time [8]			2		ms
t_{STAB}	PLL stabilization time [8]			4		
ACC_{PLL}	x8 PLL accuracy	f_{RC} = 1 MHz @ T_A = -40 to +105°C		0.1 [7]		%
JIT_{PLL}	PLL jitter ($\Delta f_{CPU}/f_{CPU}$)			1 [6]		
$I_{DD(PLL)}$	PLL current consumption	T_A = 25°C		600 [4]		µA

Notes:

1. If the RC oscillator clock is selected, to improve clock stability and frequency accuracy, it is recommended to place a decoupling capacitor, typically 100nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device.
2. See "INTERNAL RC OSCILLATOR ADJUSTMENT" on page 23
3. Minimum value is obtained for hot temperature and maximum value is obtained for cold temperature.
4. Data based on characterization results, not tested in production.
5. Measurement made with RC calibrated at 1 MHz.
6. Guaranteed by design.
7. Averaged over a 4ms period. After the LOCKED bit is set, a period of t_{STAB} is required to reach ACC_{PLL} accuracy.
8. After the LOCKED bit is set ACC_{PLL} is max. 10% until t_{STAB} has elapsed. See Figure 11 on page 23.

Operating conditions (tested for T_A = -40 to +105°C) @ V_{DD} = 3.0 to 3.6V [1]

Symbol	Parameter [1]	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal RC oscillator frequency	RCCR = FF (reset value), T_A = 25°C, V_{DD} = 3.3V		560		kHz
		RCCR = RCCR1 [3], T_A = 25°C, V_{DD} = 3.3V		700		
ACC_{RC}	Accuracy of internal RC oscillator when calibrated with RCCR = RCCR1 [3][4]	T_A = -40 to +105°C	-15		+15	%
$I_{DD(RC)}$	RC oscillator current consumption	T_A = 25°C, V_{DD} = 3.3V		700 [5]		µA
$t_{su(RC)}$	RC oscillator setup time				10 [3]	µs

Notes:

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

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

3. See "INTERNAL RC OSCILLATOR ADJUSTMENT" on page 23.
4. Minimum value is obtained for hot temperature and maximum value is obtained for cold temperature.
5. Measurement made with RC calibration at 1 MHz.
6. Guaranteed by design.
7. Averaged over a 4ms period. After the LOCKED bit is set, a period of t_{STAB} is required to reach ACC_{PLL} accuracy.
8. After the LOCKED bit is set ACC_{PLL} is max. 10% until t_{STAB} has elapsed. See Figure 11 on page 23.

OPERATING CONDITIONS (Cont'd)

Figure 43. RC Osc Freq vs V_{DD} (Calibrated with RCCR0: 5V@ 25°C)

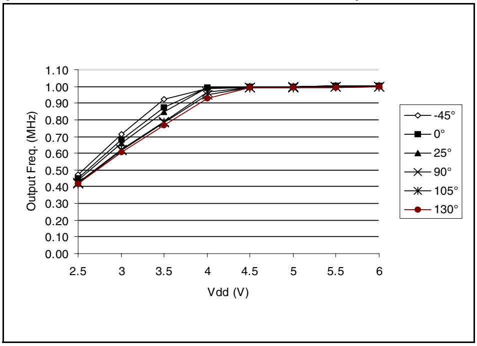


Figure 45. RC Osc Freq vs V_{DD} and RCCR Value

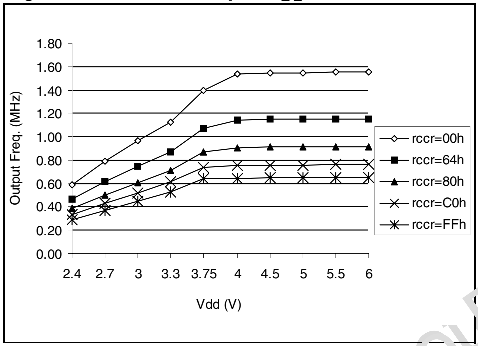


Figure 44. Typical RC oscillator Accuracy vs temperature @ V_{DD} = 5V (Calibrated with RCCR0: 5V @ 25°C

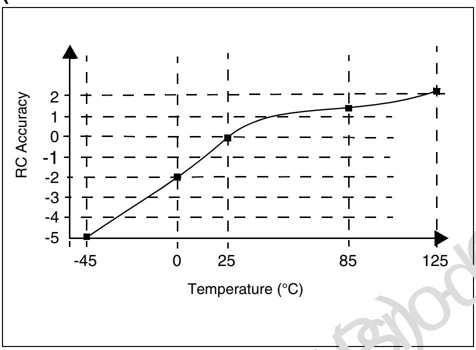
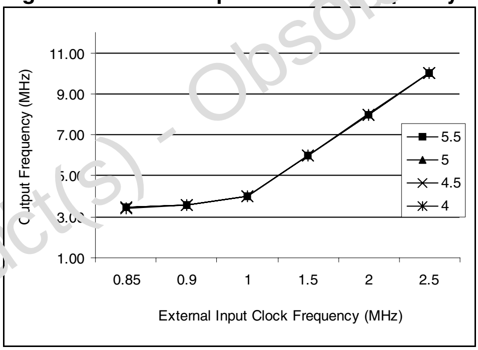


Figure 46. PLLx8 Output vs CLKIN frequency



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

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 +105°C, unless otherwise specified

Symbol	Parameter	Conditions		Typ	Max	Unit
I_{DD}	Supply current in RUN mode	V_{DD} = 5.5V	f_{CPU} = 8 MHz[1)]	5.0	7.0	mA
	Supply current in WAIT mode		f_{CPU} = 8 MHz[2)]	1.7	2.70	
	Supply current in SLOW mode		f_{CPU} = 250 kHz[3)]	0.6	1.0	
	Supply current in SLOW WAIT mode		f_{CPU} = 250 kHz[4)]	0.5	0.9	
	Supply current in HALT mode[5)]		-40°C ≤ T_A ≤ +105°C	0.5	10	µA
	Supply current in ACTIVE HALT mode			600	1000	

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.

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.

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.

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.

5. All I/O pins in output mode with a static value at V_{SS} (no load). Data based on characterization results, tested in production at V_{DD} max and f_{CPU} max.

Figure 47. Typical I_{DD} in RUN vs f_{CPU}

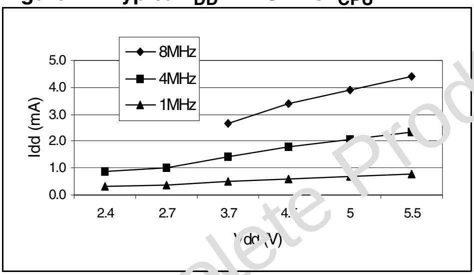


Figure 49. Typical I_{DD} in WAIT vs f_{CPU}

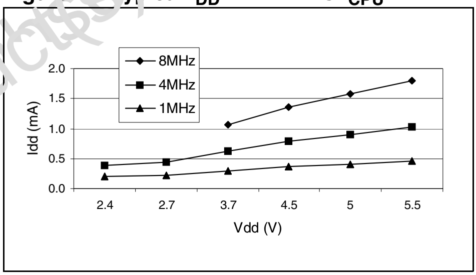


Figure 48. Typical I_{DD} in SLOW vs f_{CPU}

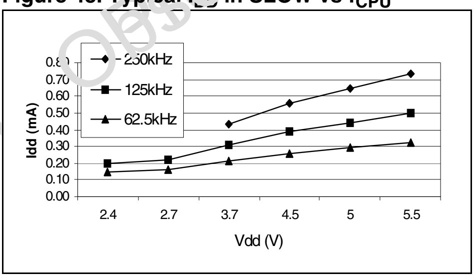
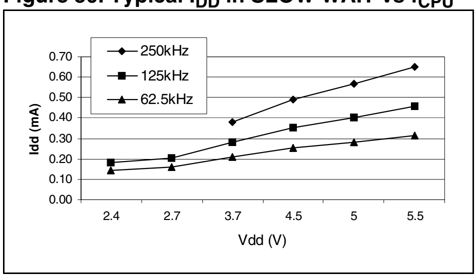
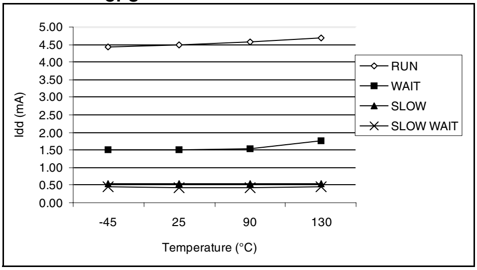


Figure 50. Typical I_{DD} in SLOW WAIT vs f_{CPU}



SUPPLY CURRENT CHARACTERISTICS (cont'd)

Figure 51. Typical I_{DD} vs Temperature at V_{DD} = 5V and f_{CPU} = 8 MHz



13.4.2 On-chip peripherals

Symbol	Parameter	Conditions		Typ	Unit
$I_{DD(AT)}$	12-bit Autoreload Timer supply current[1)]	f_{CPU} = 4 MHz	V_{DD} = 3.0V	150	µA
		f_{CPU} = 8 MHz	V_{DD} = 5.0V	250	
$I_{DD(SPI)}$	SPI supply current[2)]	f_{CPU} = 4 MHz	V_{DD} = 3.0V	50	
		f_{CPU} = 8 MHz	V_{DD} = 5.0V	300	
$I_{DD(ADC)}$	ADC supply current when converting[3)]	f_{ADC} = 4 MHz	V_{DD} = 3.0V	780	
			V_{DD} = 5.0V	1100	

Notes:

1. Data based on a differential I_{DD} measurement between reset configuration (timer stopped) and a timer running in PWM mode at f_{CPU} = 8 MHz
2. Data based on a differential I_{DD} measurement between reset configuration and a permanent SPI master communication (data sent equal to 55h)
3. Data based on a differential I_{DD} measurement between reset configuration and continuous A/D conversions

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} = 8 MHz	2	3	12	t_{CPU}
			250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time[3)] $t_{v(IT)} = \Delta t_{c(INST)} + 10$	f_{CPU} = 8 MHz	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 to +105°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, Table 13.3.1, "General Operating Conditions," on page 75	3.0		5.5	V
t_{PROG}	Programming time for 1~32 bytes[2]	T_A = −40 to +105°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°C[3]	20			years
N_{RW}	Write erase cycles	T_A = 25°C	1K			cycles
		T_A = 105°C	300			
I_{DD}	Supply current	Read / Write / Erase modes f_{CPU} = 8 MHz, V_{DD} = 5.5V			2.6[5]	mA
		No Read/No Write Mode			100	µA
		Power down mode / HALT		0	0.1	

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, Table 13.3.1, "General Operating Conditions," on page 75	3.0		5.5	V
t_{PROG}	Programming time for 1~32 bytes	T_A = −40 to +105°C		5	10	ms
t_{RET}[4]	Data retention with 1K cycling (T_{PROG} = −40 to +105°C)	T_A = 55°C[3]	20			years
	Data retention with 10K cycling (T_{PROG} = −40 to +105°C)		10			
	Data retention with 100K cycling (T_{PROG} = −40 to +105°C)		1			

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. 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 resume. 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 behavior 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} = 8 MHz conforms to IEC 1000-4-2	2B
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} = 8 MHz conforms to IEC 1000-4-4	3B

13.7.2 Electro Magnetic Interference (EMI)

Based on a simple application running on the product (toggling two 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
				1/4 MHz	1/8 MHz	
S_{EMI}	Peak level[1)]	V_{DD} = 5V, T_A = 25°C, SO16 package, conforming to SAE J 1752/3	0.1 MHz to 30 MHz	8	14	dBµV
			30 MHz to 130 MHz	27	32	
			130 MHz to 1 GHz	26	28	
			SAE EMI Level	3.5	4	-

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). Three models can be simulated: Human Body Model, Machine Model and Charge Device 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)	$T_A = 25°C$	2000	V
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine Model)		200	
$V_{ESD(CDM)}$	Electro-static discharge voltage (Charge Device Model)	Pins 1, 8, 9 and 16 ($T_A = 25°C$)	750	
		All other pins ($T_A = 25°C$)	500	

Notes:

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

13.7.3.2 Static and Dynamic Latch-Up

- **LU**: Three complementary static tests are required on 10 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard. For more details, refer to the application note AN1181.
- **DLU**: Electro-Static Discharges (one positive then one negative test) are applied to each pin of three samples when the micro is running to assess the latch-up performance in dynamic mode. Power supplies are set to the typical values, the oscillator is connected as near as possible to the pins of the micro and the component is put in reset mode. This test conforms to the IEC1000-4-2 and SAEJ1752/3 standards. For more details, refer to the application note AN1181.

Electrical Sensitivities

Symbol	Parameter	Conditions	Class[1)]
LU	Static latch-up class	$T_A = 25°C$, $T_A = 105°C$	A
DLU	Dynamic latch-up class	$V_{DD} = 5.5V$, $f_{OSC} = 4$ MHz, $T_A = 25°C$	

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 (-40 to +105°C), unless otherwise specified.

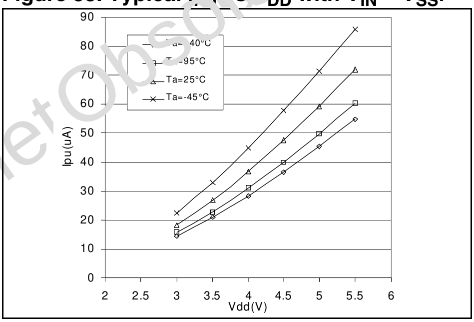
Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage			V_{SS} - 0.3		0.3 x V_{DD}	V
V_{IH}	Input high level voltage			0.7xV_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis[1)]				400		mV
I_L	Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$				± 1	µA
I_S	Static current consumption induced by each floating input pin[2)]	Floating input mode			400		
R_{PU}	Weak pull-up equivalent resistor[3)]	$V_{IN} = V_{SS}$	$V_{DD} = 5V$	50	120	250	kΩ
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)]						
$t_{w(IT)in}$	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 57). Static peak current value taken at a fixed V_{IN} value, based on design simulation and technology characteristics, not tested in production. This value depends on V_{DD} and temperature values.
3. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in Figure 53).
4. To generate an external interrupt, a minimum pulse width must be applied on an I/O port pin configured as an external interrupt source.

Figure 52. Two Typical Applications with Unused I/O Pin Configured as Input

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 53. 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_{OSC} and T_A (-40 to +105°C), 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 55)	V_{DD} = 5V	I_{IO} = +5mA		0.65	1.0	V
			I_{IO} = +2mA		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 57)		I_{IO} = +20mA		1.05	1.4	
			I_{IO} = +8mA		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 60)		I_{IO} = -5mA	V_{DD} - 1.5	4.30		
			I_{IO} = -2mA	V_{DD} - 1.0	4.70		

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 54. Typical V_{OL} at V_{DD} = 3.3V (standard)

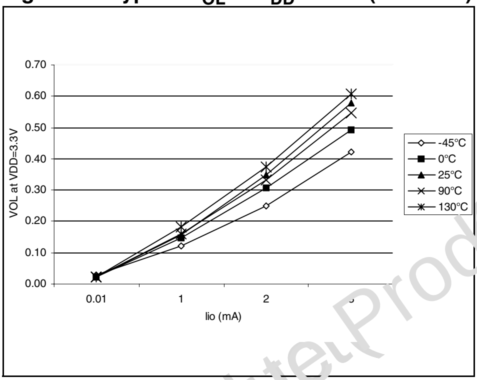


Figure 56. Typical V_{OL} at V_{DD} = 5V (high-sink)

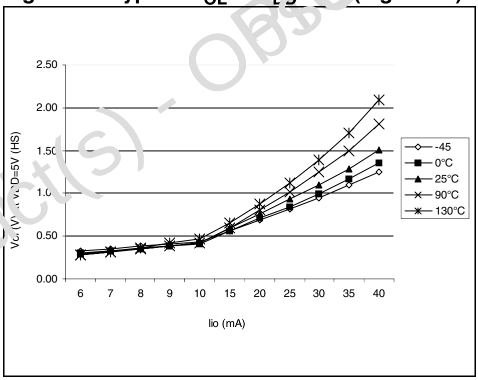


Figure 55. Typical V_{OL} at V_{DD} = 5V (standard)

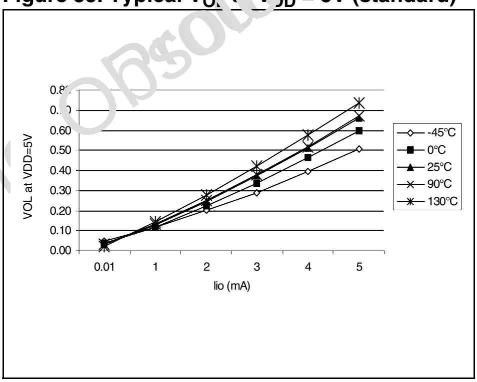
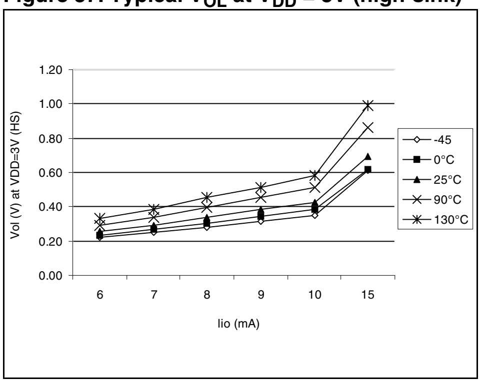


Figure 57. Typical V_{OL} at V_{DD} = 3V (high-sink)



I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 58. Typical V_{DD}-V_{OH} at V_{DD} = 3V

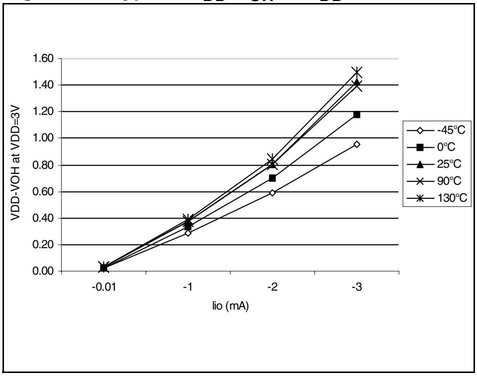


Figure 59. Typical V_{DD}-V_{OH} at V_{DD} = 4V

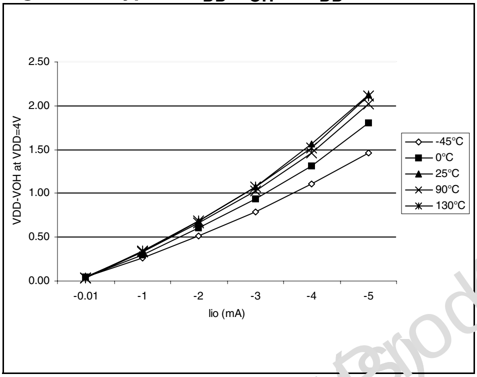
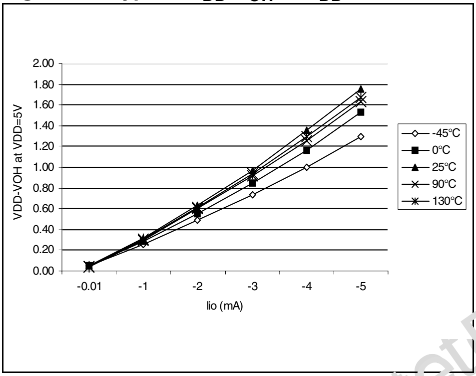


Figure 60. Typical V_{DD}-V_{OH} at V_{DD} = 5V



I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 61. Typical V_{OL} vs V_{DD} (standard I/Os)

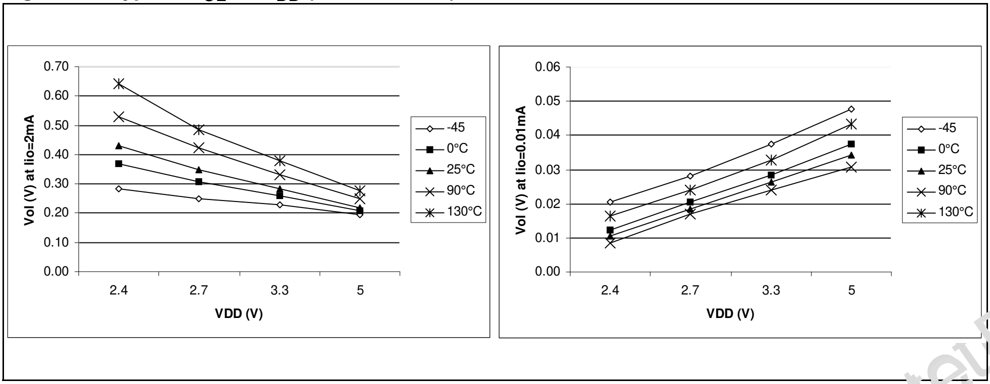


Figure 62. Typical V_{OL} vs V_{DD} (high-sink I/Os)

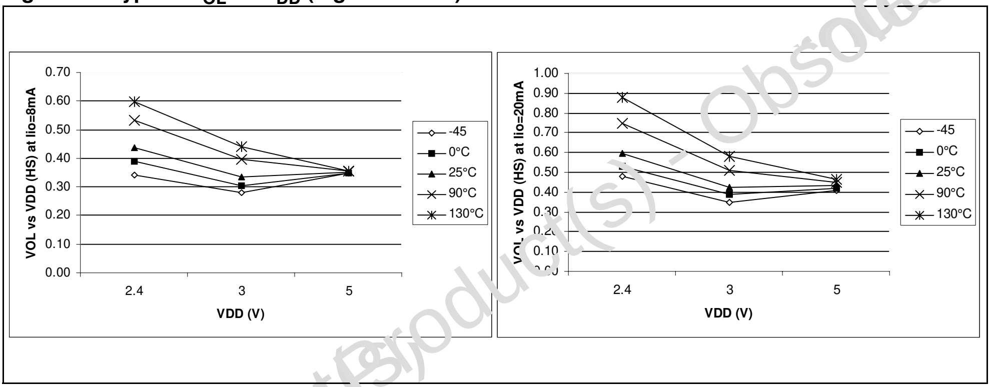
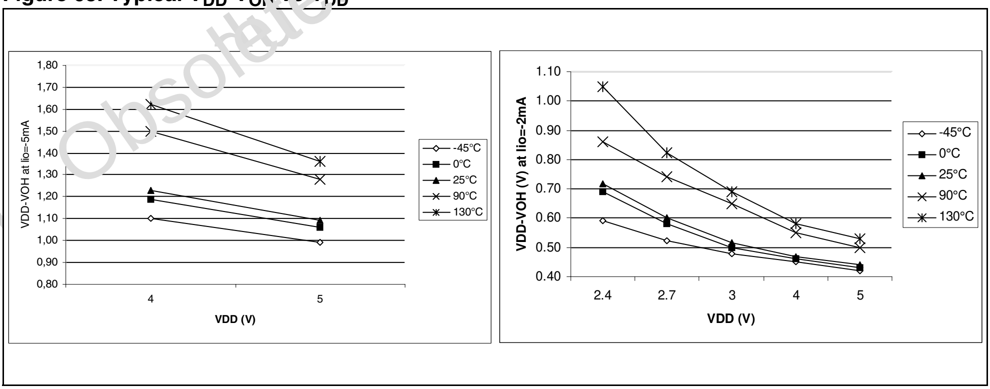


Figure 63. Typical V_{DD}-V_{OH} vs V_{DD}



13.9 CONTROL PIN CHARACTERISTICS

13.9.1 Asynchronous $\overline{RESET}$ Pin

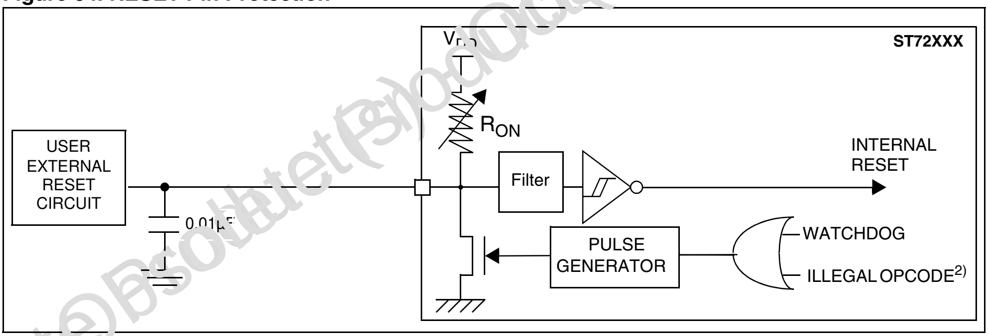
T_A = -40 to +105°C, unless otherwise specified

Symbol	Parameter	Conditions			Min	Typ	Max	Unit
V_{IL}	Input low level voltage				V_{SS} - 0.3		0.3 x V_{DD}	V
V_{IH}	Input high level voltage				0.7 x V_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis[1)]					2		
V_{OL}	Output low level voltage	V_{DD} = 5V	I_{IO} = +5mA	$T_A \le 85°C$ $T_A \le 105°C$		0.5	1.0[5)] 1.2[5)]	
			I_{IO} = +2mA	$T_A \le 85°C$ $T_A \le 105°C$		0.2	0.4[5)] 0.5[5)]	
R_{ON}	Pull-up equivalent resistor[3)1)]	V_{DD} = 5V			20	40	80	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Internal reset sources				30		µs
$t_{h(RSTL)in}$	External reset pulse hold time[4)]				20			
$t_{g(RSTL)in}$	Filtered glitch duration					200		ns

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 74 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 $\overline{RESET}$ pin between V_{ILmax} and V_{DD}
4. To guarantee the reset of the device, a minimum pulse must be applied to the $\overline{RESET}$ pin. All short pulses applied on $\overline{RESET}$ pin with a duration below $t_{h(RSTL)in}$ can be ignored.
5. Guaranteed by design, not tested in production

Figure 64. $\overline{RESET}$ Pin Protection[1)]



Notes:

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 (watchdog).

- Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{RESET}$ pin can go below the V_{IL} max. level specified in Section 13.9.1 on page 88. 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 $\overline{RESET}$ pin, the user must ensure that the current sunk on the $\overline{RESET}$ pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in Section 13.2.2 on page 74.

2. Please refer to "Illegal Opcode Reset" on page 70 for more details on illegal opcode reset conditions

13.10 COMMUNICATION INTERFACE CHARACTERISTICS

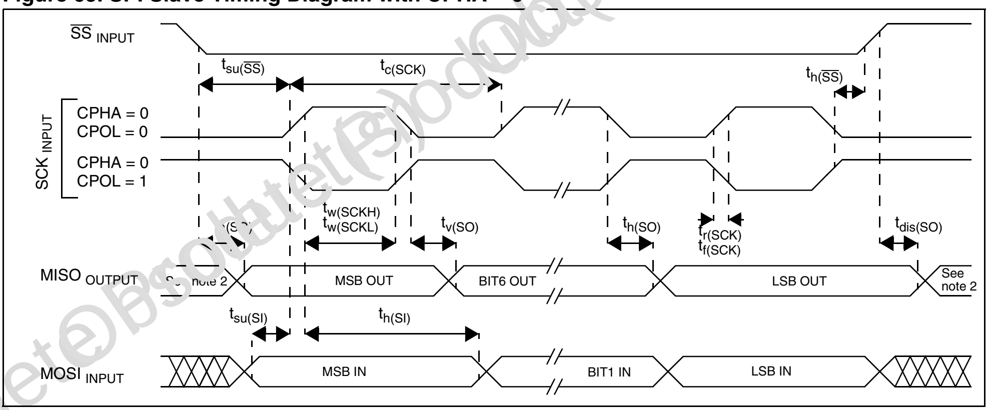
13.10.1 SPI - Serial Peripheral Interface

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

Figure 65. SPI Slave Timing Diagram with CPHA = 0 3)



Notes:

1. Data based on design simulation and/or characterization 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: 0.3 x V_{DD} and 0.7 x V_{DD}.

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

Figure 66. SPI Slave Timing Diagram with CPHA = 1[1)]

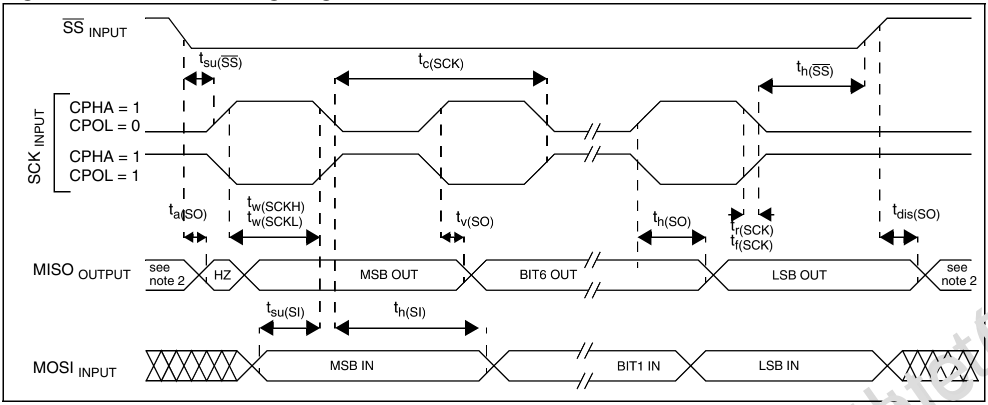
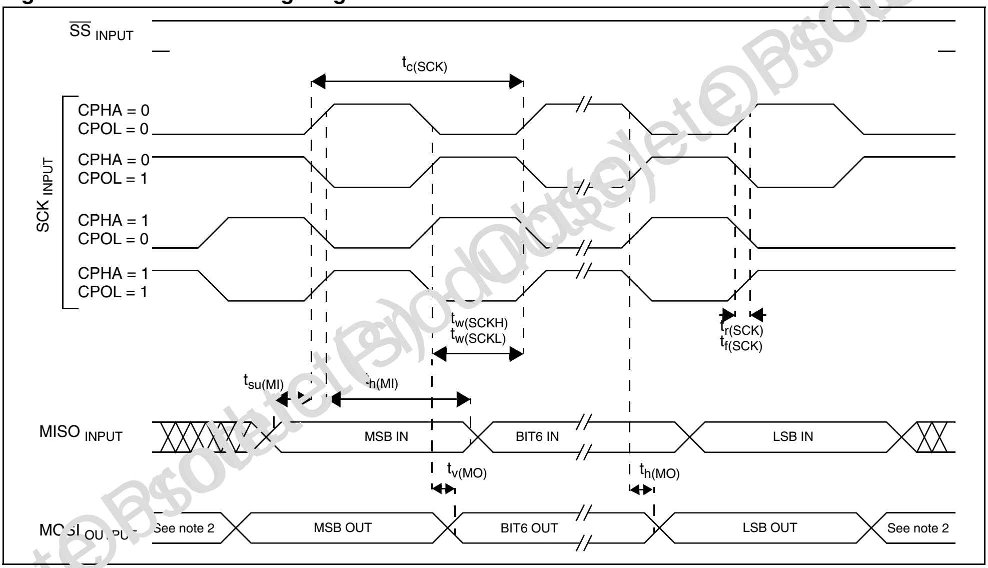


Figure 67. SPI Master Timing Diagram[1)]



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.

ADC CHARACTERISTICS (Cont'd)

13.11 8-BIT ADC CHARACTERISTICS

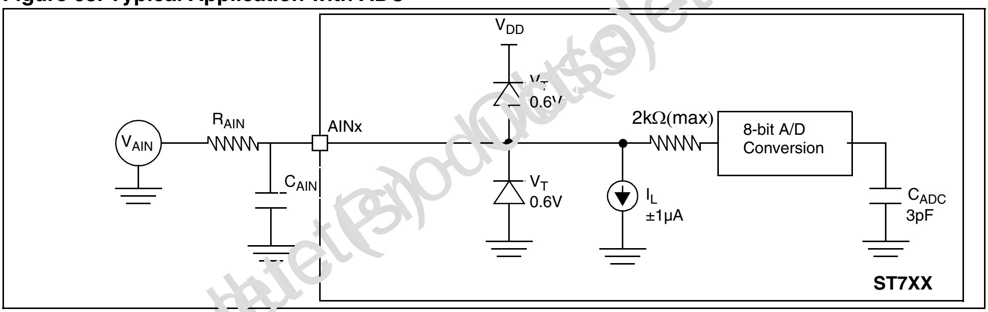
Subject to general operating conditions for V_{DD}, f_{OSC} and T_A (-40 to +105°C), unless otherwise specified.

Symbol	Parameter	Conditions	Min[2)]	Typ[1)]	Max[2)]	Unit
f_{ADC}	ADC clock frequency				4	MHz
V_{AIN}	Conversion voltage range		V_{SS}		V_{DD}	V
R_{AIN}	External input resistor				10[3)]	kΩ
C_{ADC}	Internal sample and hold capacitor	V_{DD} = 5V		3		pF
t_{STAB}	Stabilization time after ADC enable	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz		0[4)]		µs
t_{CONV}	Conversion time ($t_{SAMPLE} + t_{HOLD}$)	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz		3		µs
t_{SAMPLE}	Sample capacitor loading time	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz		4		$1/f_{ADC}$
t_{HOLD}	Hold conversion time	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz		8		$1/f_{ADC}$

Notes:

1. Unless otherwise specified, typical data is based on T_A = 25°C and $V_{DD} - V_{SS}$ = 5V. They are given only as design guidelines and are not tested.
2. Data based on characterization results, not tested in production.
3. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10kΩ). 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 68. Typical Application with ADC



ADC CHARACTERISTICS (Cont'd)

Figure 69. R_{AIN} max. vs f_{ADC} with C_{AIN} = 0pF[1)]

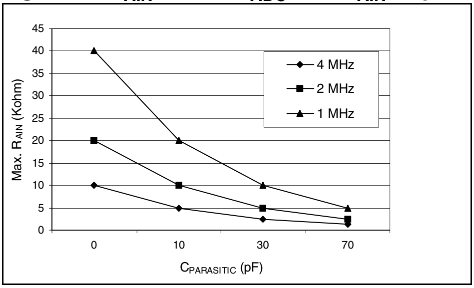
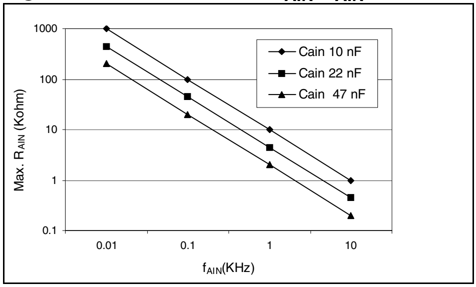


Figure 70. Recommended C_{AIN}/R_{AIN} values[2)]



Notes:

1.$C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high $C_{PARASITIC}$ value will downgrade conversion accuracy. To remedy this, f_{ADC} must be reduced.

2. This graph shows that depending on the input signal variation (f_{AIN}), C_{AIN} can be increased for stabilization and to allow the use of a larger serial resistor (R_{AIN}). It is valid for all f_{ADC} frequencies ≤ 4 MHz.

13.11.0.1 General PCB Design Guidelines

To obtain best results, some general design and layout rules must be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

– Properly place components and route the signal traces on the PCB to shield the analog inputs. Analog signals paths should run over the analog ground plane and be as short as possible. Isolate analog signals from digital signals that may switch while the analog inputs are being sampled by the A/D converter. Do not toggle digital outputs on the same I/O port as the A/D input being converted.

ADC CHARACTERISTICS (Cont'd)

T_A = -40 to +105°C unless otherwise specified

Table 20. ADC Accuracy with 3.0V $\leq V_{DD} \leq$ 3.6V

Symbol	Parameter	Conditions	Typ	Max[3]	Unit
$\lvert E_T \rvert$	Total unadjusted error	f_{CPU} = 4 MHz, f_{ADC} = 2 MHz[1)2)]	0.7	1.8	LSB
$\lvert E_O \rvert$	Offset error		0.3	0.9	
$\lvert E_G \rvert$	Gain error		0.4	1.4	
$\lvert E_D \rvert$	Differential linearity error		0.5	0.8	
$\lvert E_L \rvert$	Integral linearity error		0.4	0.8	

Table 21. ADC Accuracy with 4.5V $\leq V_{DD} \leq$ 5.5V

Symbol	Parameter	Conditions	Typ	Max[3]	Unit
$\lvert E_T \rvert$	Total unadjusted error	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz[1)2)]	0.9	2.1	LSB
$\lvert E_O \rvert$	Offset error		0.3	0.9	
$\lvert E_G \rvert$	Gain error		0.6	1.5	
$\lvert E_D \rvert$	Differential linearity error		0.5	0.9	
$\lvert E_L \rvert$	Integral linearity error		0.5	0.8	

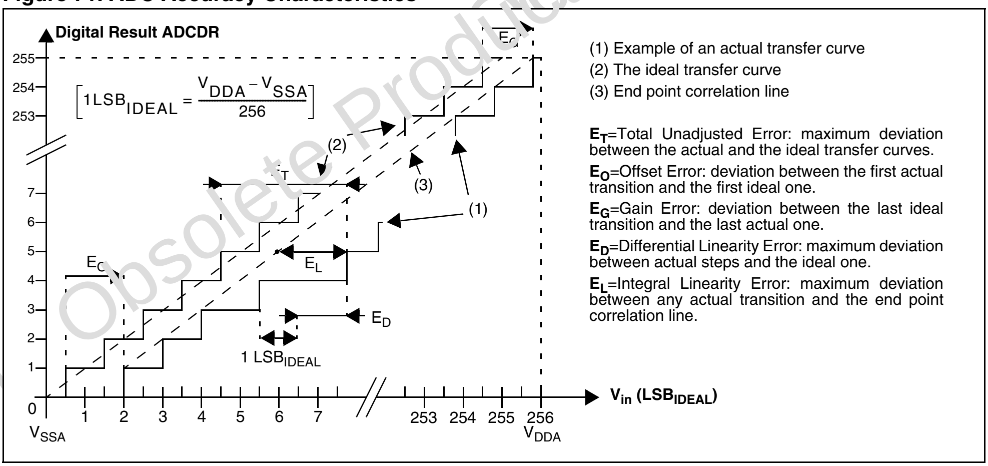
Notes:

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

2. ADC accuracy versus negative injection current: Injecting negative current on any of the analog input pins may reduce the accuracy of the conversion being performed on another analog input.
The effect of negative injection current on robust pins is specified in Section 13.11 on page 91. Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in Section 13.8 does not affect the ADC accuracy.

3. Data based on characterization results, monitored in production to guarantee 99.73% within ± max value from -40° to +105°C (± 3σ distribution limits).

Figure 71. ADC Accuracy Characteristics



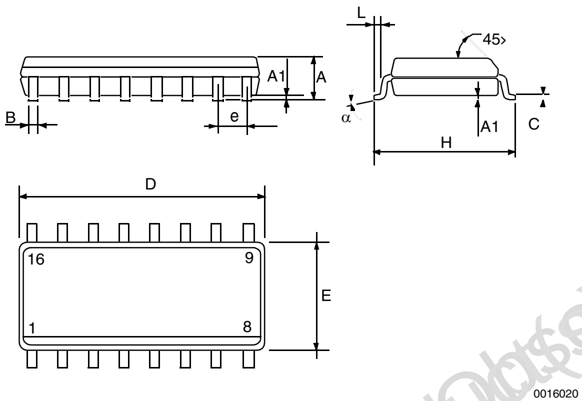
14 PACKAGE CHARACTERISTICS

In order to meet environmental requirements, ST offers these devices in ECOPACK® packages. These packages have a lead-free second level interconnect. The category of second Level Interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label.

ECOPACK is an ST trademark. ECOPACK specifications are available at www.st.com.

14.1 PACKAGE MECHANICAL DATA

Figure 72. 16-Pin Plastic Small Outline Package, 150-mil Width



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	1.35		1.75	0.053		0.069
A1	0.10		0.25	0.004		0.010
B	0.33		0.51	0.013		0.020
C	0.19		0.25	0.007		0.010
D	9.80		10.00	0.386		0.394
E	3.80		4.00	0.150		0.157
e		1.27			0.050	
H	5.80		6.20	0.228		0.244
α	0°		8°	0°		8°
L	0.40		1.27	0.016		0.050
	Number of Pins					
N	16					

14.2 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient)	95	°C/W
T_{Jmax}	Maximum junction temperature[1)]	150	°C
P_{Dmax}	Power dissipation[2)]	500	mW

Notes:
1. The maximum chip-junction temperature is based on technology characteristics.
2. The maximum power dissipation is obtained from the formula $P_D = (T_J - T_A) / R_{thJA}$.
The power dissipation of an application can be defined by the user with the formula: $P_D = P_{INT} + P_{PORT}$ where P_{INT} is the chip internal power ($I_{DD} \times V_{DD}$) and P_{PORT} is the port power dissipation depending on the ports used in the application.

PACKAGE CHARACTERISTICS (Cont'd)

14.3 SOLDERING INFORMATION

In accordance with the RoHS European directive, all STMicroelectronics packages have been converted to lead-free technology named ECOPACK™.

- ECOPACK™ packages are qualified according to the JEDEC STD-020C compliant soldering profile.
- Detailed information on the STMicroelectronics ECOPACK™ transition program is available on www.st.com/stonline/leadfree/, with specific technical application notes covering the main technical aspects related to lead-free conversion (AN2033, AN2034, AN2035, and AN2036).

Forward compatibility

ECOPACK™ SO packages are fully compatible with a lead (Pb) containing soldering process (see application note AN2034).

Table 22. Soldering Compatibility (Wave and Reflow Soldering Process)

Package	Plating material	Pb solder paste	Pb-free solder paste
SO	NiPdAu (Nickel-Palladium-Gold)	Yes	Yes

15 DEVICE CONFIGURATION AND ORDERING INFORMATION

15.1 INTRODUCTION

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

The ST7PL0x device is a Factory Advanced Service Technique ROM (FASTROM) version: It is a factory-programmed XFlash device.

The ST7FL0x XFlash device is shipped to customers with a default program memory content (FFh). The OSC option bit is programmed to 0 by default.

The FASTROM factory coded parts contain the code supplied by the customer. This implies that Flash devices have to be configured by the customer using the Option Bytes while the FASTROM devices are factory-configured.

15.2 OPTION BYTES

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

The option bytes can be accessed only in programming mode (for example, using a standard ST7 programming tool).

OPTION BYTE 0

Bits 7:4 = Reserved, must always be 1

Bits 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
1.5k	1	x

Bit 1 = **FMP_R** *Read-out protection*
Read-out 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 Section 4.5 and the *ST7 Flash Programming Reference Manual* for more details.
0: Read-out protection off
1: Read-out protection on

Bit 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

DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

OPTION BYTE 1

Bit 7 = Reserved, must always be 1

Bit 6 = **PLLOFF** *PLL disabled*
0: PLL enabled
1: PLL disabled (by-passed)

Bit 5 = Reserved, must always be 1

Bit 4 = **OSC** *RC Oscillator selection*
0: RC oscillator on
1: RC oscillator off

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

Table 23. List of Valid Option Combinations

Operating conditions				Option Bits	
V_{DD} range	**Clock Source**	**PLL**	**Typ f_{CPU}**	**OSC**	**PLLOFF**
3.6 to 5.5V	Internal RC 1%	off	1 MHz @ 5V	0	1
		x8	8 MHz @ 5V		0
	External clock	off	0 to 8 MHz	1	1
		x8	8 MHz		0

Note: See Clock Management Block diagram in Figure 12.

Bits 3:2 = Reserved

Bit 1 = **WDG SW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware (watchdog always enabled)
1: Software (watchdog to be enabled by software)

Bit 0 = **WDG HALT** *Watchdog Reset on Halt*
This option bit determines if a RESET is generated when entering HALT mode while the Watchdog is active.
0: No Reset generation when entering Halt mode
1: Reset generation when entering Halt mode

	OPTION BYTE 0 (7 … 0)								OPTION BYTE 1 (7 … 0)							
	Reserved				SEC1	SEC0	FMP R	FMP W	Res.	PLL OFF	Res.	OSC	Res.	Res.	WDG SW	WDG HALT
Default Value	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	1

DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

15.3 FLASH DEVICE ORDERING INFORMATiON

Figure 73. Flash Device Types

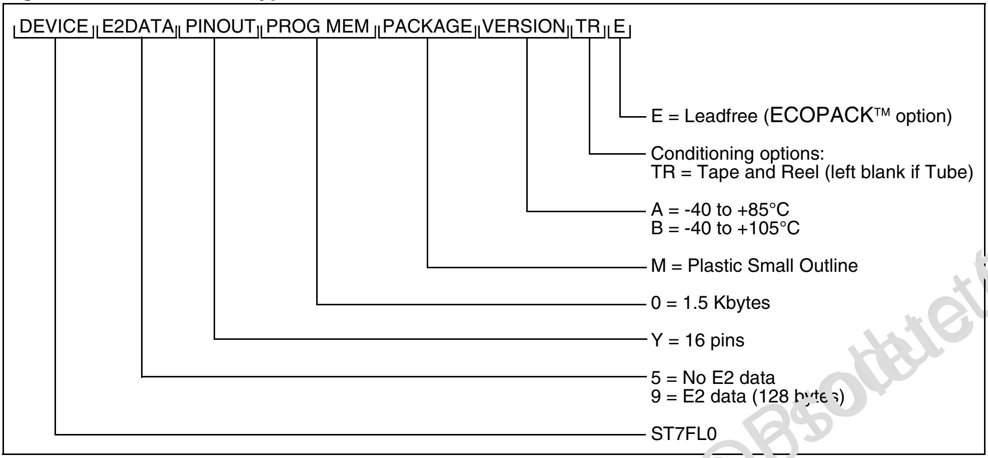


Table 24. Flash User Programmable Device Types

Part Number	Program Memory (Bytes)	Data EEPROM (Bytes)	RAM (Bytes)	Temp. Range	Package
ST7FL05Y0MA	1.5K Flash	-	128	-40 to +85°C	SO16
ST7FL09Y0MA		128			
ST7FL05Y0MB				-40 to +105°C	
ST7FL09Y0MB		128			

15.4 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.

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.

DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

Figure 74. FASTROM Commercial Product Code Structure

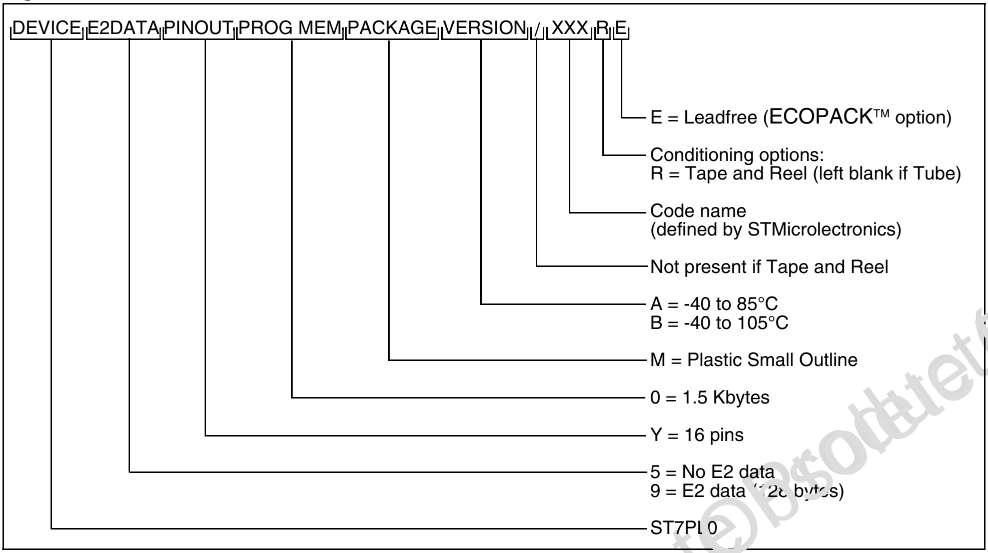


Table 25. FASTROM Factory Coded Device Types

Part Number	Program Memory (Bytes)	Data EEPROM (Bytes)	RAM (Bytes)	Temp. Range	Package
ST7PL05Y0MA	1.5K FASTROM	-	128	-40 to +85°C	SO16
ST7PL09Y0MA		128			
ST7PL05Y0MB		-		-40 to +105°C	
ST7PL09Y0MB		128			

ST7L0x FASTROM MICROCONTROLLER OPTION LIST
(Last update: December 2006)

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	1.5K
SO16:	[ ] ST7PL05 [ ] ST7PL09

Warning: Addresses 1000h, 1001h, FFDEh and FFDFh are reserved areas for ST to program RCCR0 and RCCR1 (see Section 7.1 on page 23).

Conditioning (check only one option):	[ ] Tape & Reel	[ ] Tube	
Special marking:	[ ] No	[ ] Yes "_ _ _ _ _ _ _ _ _"	
Authorized characters are letters, digits, '.', '-', '/' and spaces only.			
Maximum character count:			
SO16 (11 char. max): _ _ _ _ _ _ _ _ _ _ _			
Temperature range:	[ ] A (-40 to +85°C)	[ ] B (-40 to +105°C)	
Clock Source Selection:	[ ] External Clock	[ ] Internal RC Oscillator	
Sector 0 size:	[ ] 0.5K	[ ] 1K	[ ] 1.5K
Read-out Protection:	[ ] Disabled	[ ] Enabled	
Flash Write Protection:	[ ] Disabled	[ ] Enabled	
PLL:	[ ] Disabled	[ ] Enabled	
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 15.2 on page 96 for authorized option byte combinations.

16 DEVELOPMENT TOOLS

16.1 INTRODUCTION

Development tools for the ST7 microcontrollers include a complete range of hardware systems and software tools from STMicroelectronics and third-party tool suppliers. The range of tools includes solutions to help you evaluate microcontroller peripherals, develop and debug your application, and program your microcontrollers.

16.2 DEVELOPMENT AND DEBUGGING TOOLS

Application development for ST7 is supported by fully optimizing **C Compilers** and the **ST7 Assembler-Linker** toolchain, which are all seamlessly integrated in the ST7 integrated development environments in order to facilitate the debugging and fine-tuning of your application. The Cosmic C Compiler is available in a free version that outputs up to 16 Kbytes of code.

The range of hardware tools includes full-featured **ST7-EMU3** and cost effective **ST7-DVP3 series emulators**. These tools are supported by the **ST7 Toolset** from STMicroelectronics, which includes the STVD7 integrated development environment (IDE) with high-level language debugger, editor, project manager and integrated programming interface.

16.3 PROGRAMMING TOOLS

During the development cycle, the **ST7-DVP3** and **ST7-EMU3 series emulators** and the **RLink** provide in-circuit programming capability for programming the Flash microcontroller on your application board.

ST also provides a low-cost dedicated in-circuit programmer, the **ST7-STICK**, as well as **ST7 Socket Boards** which provide all the sockets required for programming any of the devices in a specific ST7 sub-family on a platform that can be used with any tool with in-circuit programming capability for ST7.

For production programming of ST7 devices, ST's third-party tool partners also provide a complete range of gang and automated programming solutions, which are ready to integrate into your production environment.

16.4 ORDER CODES FOR DEVELOPMENT AND PROGRAMMING TOOLS

Table 26 below lists the ordering codes for the ST7L0x development and programming tools. For additional ordering codes for spare parts and accessories, refer to the online product selector at www.st.com/mcu.

16.4.1 Order Codes for ST7L0x Development Tools

Table 26. Development Tool Order Codes for the ST7L0 Family

MCU	Emulator		Programming Tool	
	DVP Series	EMU Series	In-circuit Programmer	ST Socket Boards[5]
ST7FL05, ST7FL09	ST7MDT10-DVP3[1]	ST7MDT10-EMU3	ST7-STICK[2)3)] STX-RLINK[4]	ST7SB10-SU0[2]

Notes:

1. Includes connection kit for DIP16/SO16 only. See "How to order an EMU or DVP" in ST product and tool selection guide for connection kit ordering information
2. Add suffix /EU, /UK or /US for the power supply for your region
3. Parallel port connection to PC
4. USB connection to PC
5. Socket boards complement any tool with ICC capabilities (ST7-STICK, InDART, RLINK, DVP3, EMU3, etc.)

16.5 ST7 APPLICATION NOTES

All relevant ST7 application notes can be found on www.st.com.

17 KNOWN LIMITATIONS

17.1 EXECUTION OF BTJX INSTRUCTION

Description

Executing a BTJx instruction jumps to a random address in the following conditions: The jump goes to a lower address (jump backward) and the test is performed on data located at the address 00FFh.

17.2 IN-CIRCUIT PROGRAMMING OF DEVICES PREVIOUSLY PROGRAMMED WITH HARDWARE WATCHDOG OPTION

Description

In-Circuit Programming of devices configured with Hardware Watchdog (WDGSW bit in option byte 1 programmed to 0) requires certain precautions (see below).

In-Circuit Programming uses ICC mode. In this mode, the Hardware Watchdog is not automatically deactivated as one might expect. As a consequence, internal resets are generated every 2 ms by the watchdog, thus preventing programming.

The device factory configuration is Software Watchdog so this issue is not seen with devices that are programmed for the first time. For the same reason, devices programmed by the user with the Software Watchdog option are not impacted.

The only devices impacted are those that have previously been programmed with the Hardware Watchdog option.

Workaround

Devices configured with Hardware Watchdog must be programmed using a specific programming mode that ignores the option byte settings. In this mode, an external clock, normally provided by the programming tool, has to be used. In ST tools, this mode is called "ICP OPTIONS DISABLED".

Sockets on ST programming tools (such as ST7MDT10-EPB) are controlled using "ICP OPTIONS DISABLED" mode. Devices can therefore be reprogrammed by plugging them in the ST Programming Board socket, whatever the watchdog configuration.

When using third-party tools, please refer the manufacturer's documentation to check how to access specific programming modes. If a tool does not have a mode that ignores the option byte settings, devices programmed with the Hardware watchdog option cannot be reprogrammed using this tool.

17.3 IN-CIRCUIT DEBUGGING WITH HARDWARE WATCHDOG

In-Circuit Debugging is impacted in the same way as In-Circuit Programming by the activation of the hardware watchdog in ICC mode. Please refer to Section 17.2.

17.4 CLEARING ACTIVE INTERRUPTS OUTSIDE INTERRUPT ROUTINE

When an active interrupt request occurs at the same time as the related flag or interrupt mask is being cleared, the CC register may be corrupted.

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, that is, when:

– The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

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

Example:

SIM

rReset flag or interrupt mask

RIM

18 REVISION HISTORY

Table 27. Revision History

Date	Revision	Description of changes
27-Mar-06	1	Initial Release
09-Oct-06	2	Changed "Memories" on page 1 Changed last paragraph of Section 5.5 on page 18 Changed description of bits 11:0 of CNTR register in Section 11.2.6 on page 50 Changed I_{VSS}, I_{VDD} and I_{IO} maximum values in Section 13.2.2 on page 74 Changed ACC_{RC} parameter for RC oscillator operating conditions in Section 13.3.2 on page 75 Removed note "tested in production" from Figure 44 on page 77 Changed values for $tv_{(MO)}$ and $th_{(MO)}$ in Section 13.10.1 on page 89 Replaced "CPHA = 0" with "CPHA = 1" in Figure 66 on page 90 Repositioned $tv_{(MO)}$ and $th_{(MO)}$ in Figure 67 on page 90 Changed values and note 3 in Table 20 on page 93 and Table 21 on page 93 Removed figure 72 "16-Pin Plastic Dual In-Line Package, 300-mil Width" from Section 14.1 on page 94 Changed Section 14.3 on page 95 Changed Table 22 on page 95 Changed Section 15.3 on page 98 by adding Figure 73. Flash Device Types and Table 24, "Flash User Programmable Device Types," on page 98 Changed Section 15.4 on page 98 by adding Figure 74. FASTROM Commercial Product Code Structure and Table 25, "FASTROM Factory Coded Device Types," on page 99 Updated "ST7L0x FASTROM MICROCONTROLLER OPTION LIST (Last update: October 2006)" on page 100 Changed Section 16 DEVELOPMENT TOOLS Removed section STARTER KITS from Section 16 DEVELOPMENT TOOLS Removed Starter Kits from Table 26, "Development Tool Order Codes for the ST7L0 Family," on page 101 Added a statement to indicate that application notes can be found on the ST website and removed Table 26, ST7 Application Notes from Section 16.5 on page 101 Updated disclaimer (last page) to include a mention about the use of ST products in automotive applications
04-Dec-06	3	Replaced "ST7L0" with "ST7L05, ST7L09" in document name on page 1 Added "Features" heading above list of features on page 1 Added table number to "Device Summary" on page 1 Changed title of Section 1 on page 5 Table 2 on page 7: Removed caution about PB0 negative current injection restriction Section 7.4.1 on page 26: Added caution about avoiding unwanted behavior during Reset sequence Section 11.2.6 on page 50: Changed description of bits 11:0 of CNTR register (last sentence which had been inadvertantly removed in Rev. 2 has now been restored) Section 13.2.2 on page 74: - Replaced "Injected current on PB0 and PB1 pins" with "Injected current on PB1 pin" in ratings - Changed note 5 to remove PB0 negative current restriction Section 13.3.2 on page 75: Changed ACC_{RC} parameter for RC oscillator operating conditions Section 14.3 on page 95: Removed text concerning Pb-containing packages Table 22 on page 95: - Changed title of Plating Material column - Added Pb solder paste - Removed note 1 Removed link to st.com from bottom of "ST7L0x FASTROM MICROCONTROLLER OPTION LIST" on page 100