



Welcome to [E-XFL.COM](#)

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Active
Core Processor	ST7
Core Size	8-Bit
Speed	16MHz
Connectivity	SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	15
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.4V ~ 5.5V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	20-SOIC (0.295", 7.50mm Width)
Supplier Device Package	20-SO
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7flite15m6tr

Address	Block	Register Label	Register Name	Reset Status	Remarks
0037h	ITC	EICR	External Interrupt Control Register	00h	R/W
0038h	MCC	MCCSR	Main Clock Control/Status Register	00h	R/W
0039h 003Ah	Clock and Reset	RCCR SICSR	RC oscillator Control Register System Integrity Control/Status Register	FFh 0000 0XX0h	R/W R/W
003Bh	Reserved area (1 byte)				
003Ch	ITC	EISR	External Interrupt Selection Register	0Ch	R/W
003Dh to 0048h	Reserved area (12 bytes)				
0049h 004Ah	AWU	AWUPR AWUCSR	AWU Prescaler Register AWU Control/Status Register	FFh 00h	R/W R/W
004Bh 004Ch 004Dh 004Eh 004Fh 0050h	DM ³⁾	DMCR DMSR DMBK1H DMBK1L DMBK2H DMBK2L	DM Control Register DM Status Register DM Breakpoint Register 1 High DM Breakpoint Register 1 Low DM Breakpoint Register 2 High DM Breakpoint Register 2 Low	00h 00h 00h 00h 00h 00h	R/W R/W R/W R/W R/W R/W
0051h to 007Fh	Reserved area (47 bytes)				

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

Notes:

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

CPU REGISTERS (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 01FFh

15							8
0	0	0	0	0	0	0	1
7							0
1	SP6	SP5	SP4	SP3	SP2	SP1	SP0

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

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

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

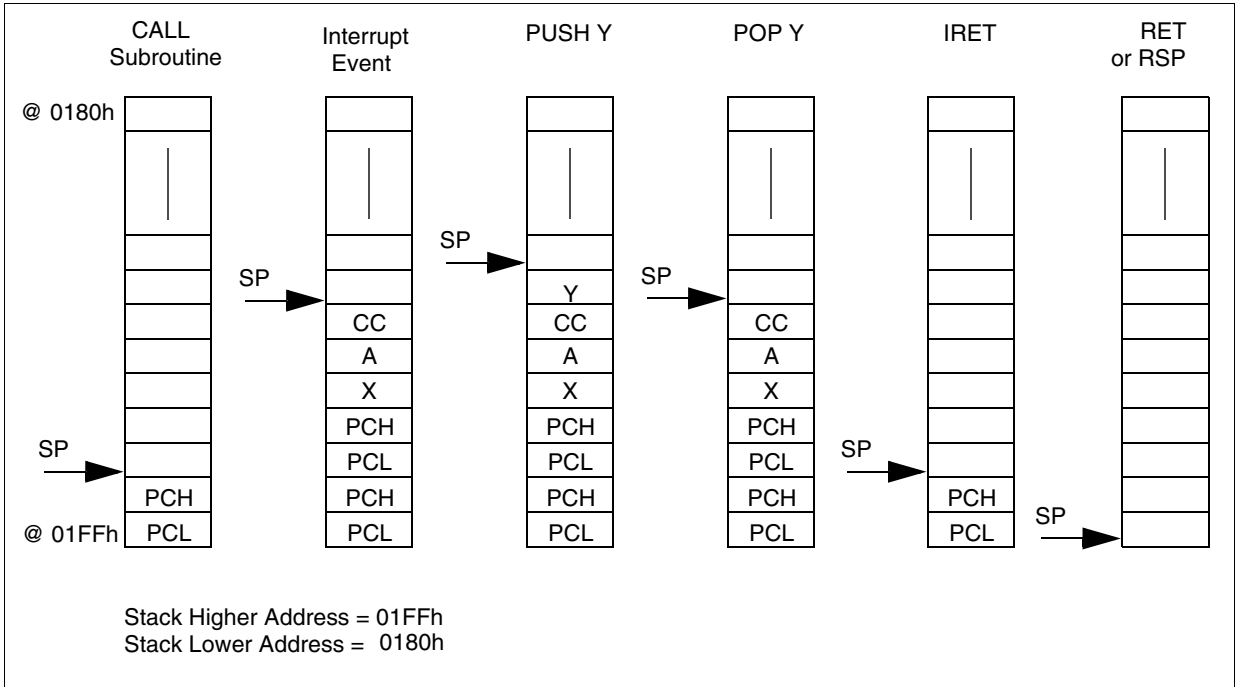
Note: When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an under-flow.

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, available on ST7LITE15 and ST7LITE19 devices only)
- 1 to 16 MHz or 32kHz External crystal/ceramic resonator (selected by option byte)
- External Clock Input (enabled by option byte)
- PLL for multiplying the frequency by 8 or 4 (enabled by option byte)
- For clock ART counter only: PLL32 for multiplying the 8 MHz frequency by 4 (enabled by option byte). The 8 MHz input frequency is mandatory and can be obtained in the following ways:
 - 1 MHz RC + PLLx8
 - 16 MHz external clock (internally divided by 2)
 - 2 MHz. external clock (internally divided by 2) + PLLx8
 - Crystal oscillator with 16 MHz output frequency (internally divided by 2)

■ Reset Sequence Manager (RSM)

■ System Integrity Management (SI)

- Main supply Low voltage detection (LVD) with reset generation (enabled by option byte)
- Auxiliary Voltage detector (AVD) with interrupt capability for monitoring the main supply (enabled by option byte)

7.1 INTERNAL RC OSCILLATOR ADJUSTMENT

The device contains an internal RC oscillator with an accuracy of 1% for a given device, temperature and voltage range (4.5V-5.5V). It must be calibrated to obtain the frequency required in the application. This is done by software writing a calibration value in the RCCR (RC Control Register).

Whenever the microcontroller is reset, the RCCR returns to its default value (FFh), i.e. each time the device is reset, the calibration value must be loaded in the RCCR. Predefined calibration values are stored in EEPROM for 3 and 5V V_{DD} supply voltages at 25°C, as shown in the following table.

RCCR	Conditions	ST7LITE19 Address	ST7LITE15 Address
RCCR0	$V_{DD}=5V$ $T_A=25^{\circ}C$ $f_{RC}=1MHz$	1000h and FFDEh	FFDEh
RCCR1	$V_{DD}=3V$ $T_A=25^{\circ}C$ $f_{RC}=700KHz$	1001h and FFDFh	FFDFh

Note:

- See “ELECTRICAL CHARACTERISTICS” on page 91. for more information on the frequency and accuracy of the RC oscillator.
- To improve clock stability, it is recommended to place a decoupling capacitor between the V_{DD} and V_{SS} pins.
- These two bytes are systematically programmed by ST, including on FASTROM devices. Consequently, customers intending to use FASTROM service must not use these two bytes.
- RCCR0 and RCCR1 calibration values will be 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 need to be recalibrated.

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

7.2 PHASE LOCKED LOOP

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

- The x4 PLL is intended for operation with V_{DD} in the 2.4V to 3.3V range
- The x8 PLL is intended for operation with V_{DD} in the 3.3V to 5.5V range

Refer to [Section 15.1](#) for the option byte description.

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

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

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

7.6.2 Auxiliary Voltage Detector (AVD)

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

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

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

abled through the option byte.

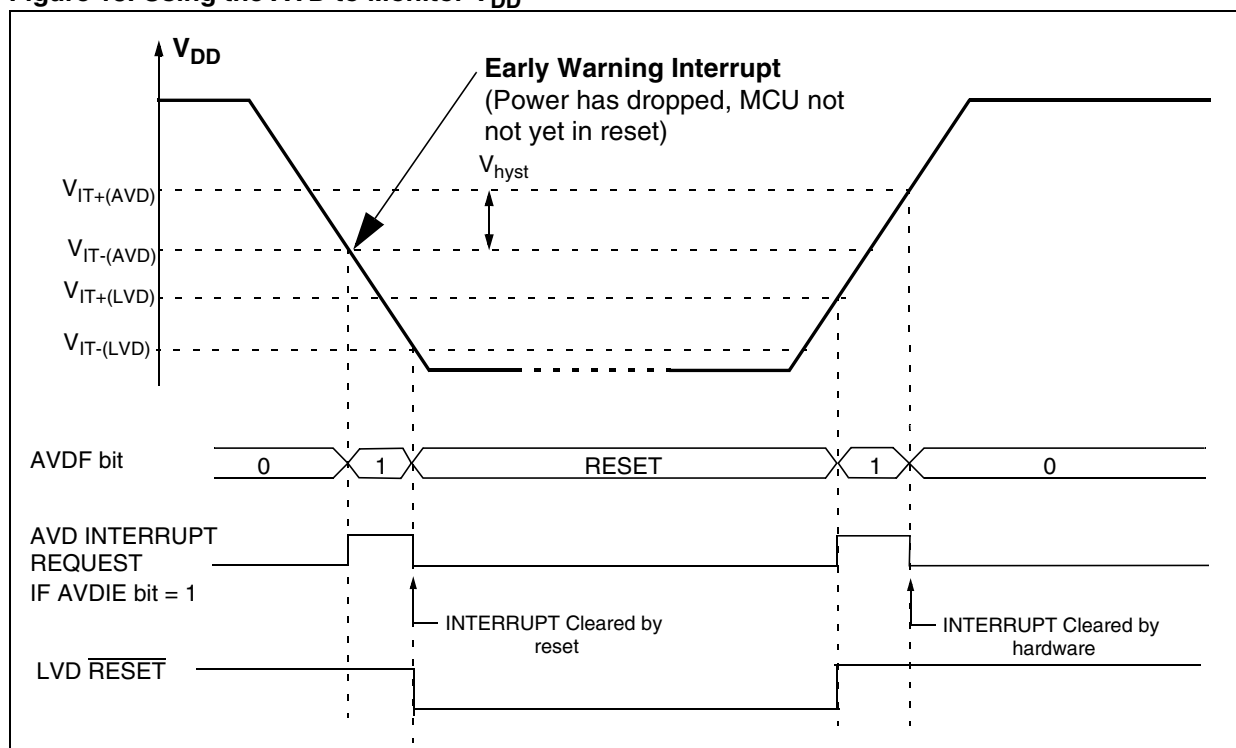
7.6.2.1 Monitoring the V_{DD} Main Supply

The AVD voltage threshold value is relative to the selected LVD threshold configured by option byte (see [section 15.1 on page 121](#)).

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

In the case of a drop in voltage, the AVD interrupt acts as an early warning, allowing software to shut down safely before the LVD resets the microcontroller. See [Figure 18](#).

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



SYSTEM INTEGRITY MANAGEMENT (Cont'd)**7.6.3 Low Power Modes**

Mode	Description
WAIT	No effect on SI. AVD interrupts cause the device to exit from Wait mode.
HALT	The SICSR register is frozen. The AVD remains active.

set and the interrupt mask in the CC register is reset (RIM instruction).

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

7.6.3.1 Interrupts

The AVD interrupt event generates an interrupt if the corresponding Enable Control Bit (AVDIE) is

POWER SAVING MODES (Cont'd)

Figure 25. ACTIVE-HALT Timing Overview

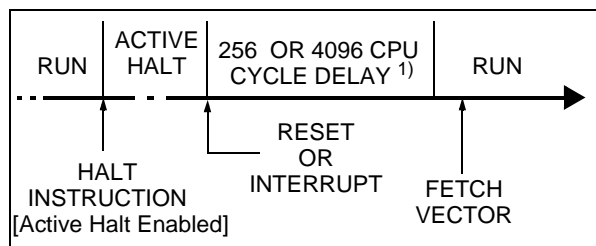
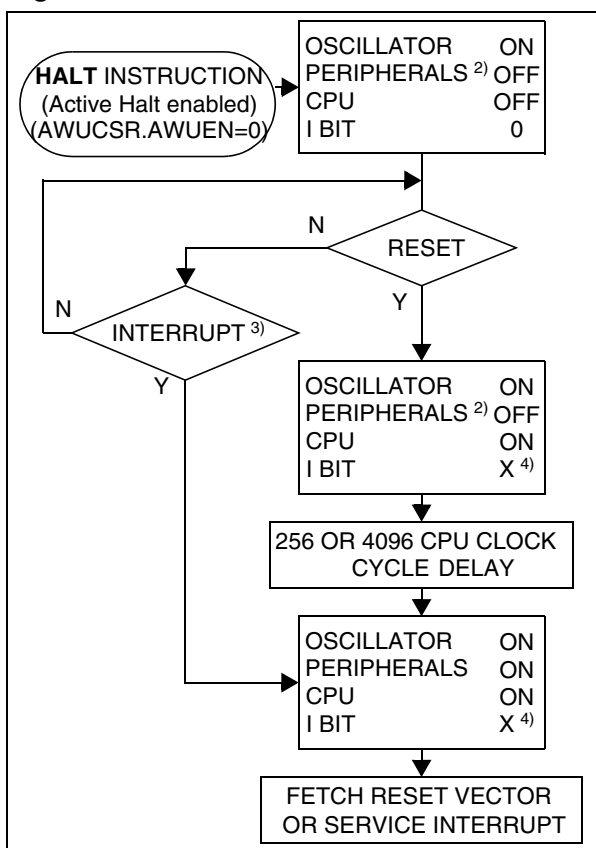


Figure 26. ACTIVE-HALT Mode Flow-chart



Notes:

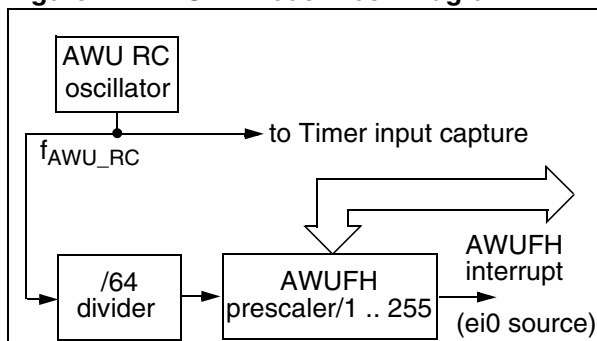
1. This delay occurs only if the MCU exits ACTIVE-HALT mode by means of a RESET.
2. Peripherals clocked with an external clock source can still be active.
3. Only the RTC1 interrupt and some specific interrupts can exit the MCU from ACTIVE-HALT mode. Refer to Table 5, "Interrupt Mapping," on page 35 for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I bit of the CC register is set during the interrupt routine and cleared when the CC register is popped.

9.6 AUTO WAKE UP FROM HALT MODE

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

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

Figure 27. AWUFH Mode Block Diagram



As soon as HALT mode is entered, and if the AWUEN bit has been set in the AWUCSR register, the AWU RC oscillator provides a clock signal (f_{AWU_RC}). Its frequency is divided by a fixed divider and a programmable prescaler controlled by the AWUPR register. The output of this prescaler provides the delay time. When the delay has elapsed the AWUF flag is set by hardware and an interrupt wakes-up the MCU from Halt mode. At the same time the main oscillator is immediately turned on and a 256 or 4096 cycle delay is used to stabilize it. After this start-up delay, the CPU resumes operation by servicing the AWUFH interrupt. The AWU flag and its associated interrupt are cleared by software reading the AWUCSR register.

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

I/O PORTS (Cont'd)

10.7 DEVICE-SPECIFIC I/O PORT CONFIGURATION

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

Standard Ports

PA7:0, PB6:0

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

Interrupt Ports

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

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

Table 10. Port Configuration (Standard ports)

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7:0	floating	pull-up	open drain	push-pull
Port B	PB6:0	floating	pull-up	open drain	push-pull

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

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7:0	floating	pull-up interrupt	open drain	push-pull
Port B	PB6:0	floating	pull-up interrupt	open drain	push-pull

Table 11. I/O Port Register Map and Reset Values

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

WATCHDOG TIMER (Cont'd)**11.1.5 Interrupts**

None.

11.1.6 Register Description**CONTROL REGISTER (CR)**

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

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

This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset.

0: Watchdog disabled

1: Watchdog enabled

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

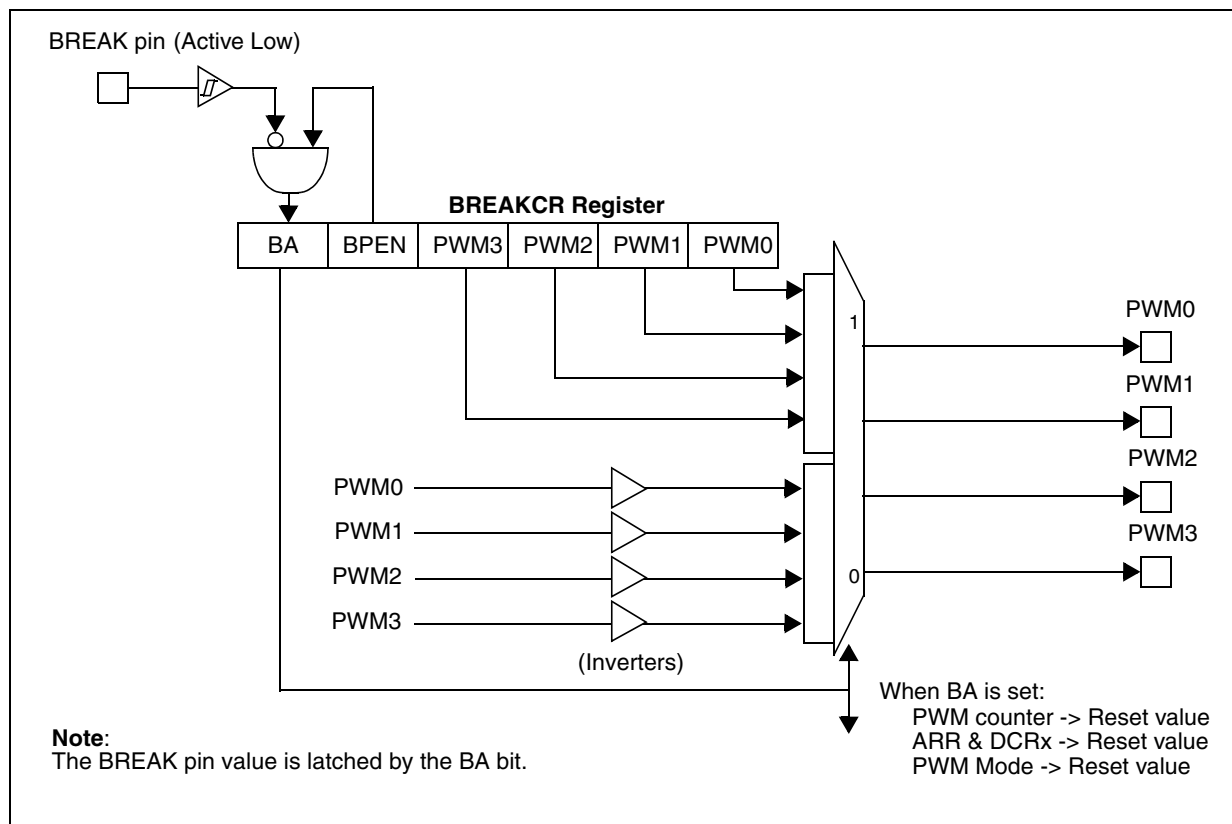
Bit 6:0 = **T[6:0]** *7-bit timer (MSB to LSB).*

These bits contain the decremented value. A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

WATCHDOG TIMER (Cont'd)**Table 13. Watchdog Timer Register Map and Reset Values**

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

Figure 37. Block Diagram of Break Function

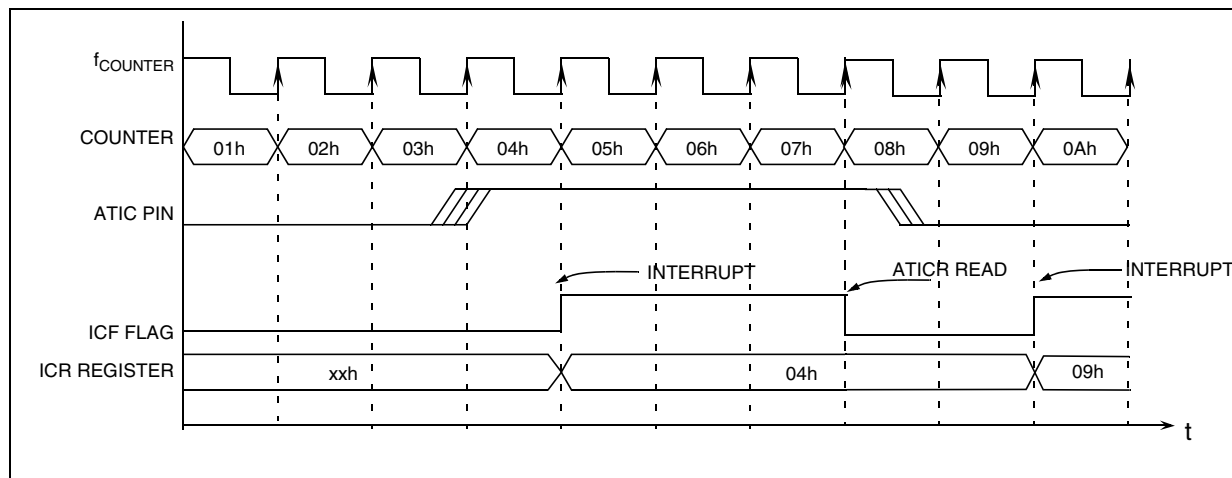


11.2.3.1 Input Capture

The 12-bit ATICR register is used to latch the value of the 12-bit free running upcounter after a rising or falling edge is detected on the ATIC pin. When an input capture occurs, the ICF bit is set and the ATICR register contains the value of the

free running upcounter. An IC interrupt is generated if the ICIE bit is set. The ICF bit is reset by reading the ATICR register when the ICF bit is set. The ATICR is a read only register and always contains the free running upcounter value which corresponds to the most recent input capture. Any further input capture is inhibited while the ICF bit is set.

Figure 38. Input Capture Timing Diagram



12-BIT AUTORELOAD TIMER (Cont'd)

Table 14. Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0D	ATCSR Reset Value	0	ICF 0	ICIE 0	CK1 0	CK0 0	OVF 0	OVFIE 0	CMPIE 0
0E	CNTRH Reset Value	0	0	0	0	CNTR11 0	CNTR10 0	CNTR9 0	CNTR8 0
0F	CNTRL Reset Value	CNTR7 0	CNTR8 0	CNTR7 0	CNTR6 0	CNTR3 0	CNTR2 0	CNTR1 0	CNTR0 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	OE3 0	0	OE2 0	0	OE1 0	0	OE0 0
13	PWM0CSR Reset Value	0	0	0	0	0	0	OP0 0	CMPF0 0
14	PWM1CSR Reset Value	0	0	0	0	0	0	OP1 0	CMPF1 0
15	PWM2CSR Reset Value	0	0	0	0	0	0	OP2 0	CMPF2 0
16	PWM3CSR Reset Value	0	0	0	0	0	0	OP3 0	CMPF3 0
17	DCR0H Reset Value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
18	DCR0L Reset Value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
19	DCR1H Reset Value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
1A	DCR1L Reset Value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
1B	DCR2H Reset Value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
1C	DCR2L Reset Value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
1D	DCR3H Reset Value	0	0	0	0	DCR11 0	DCR10 0	DCR9 0	DCR8 0
1E	DCR3L Reset Value	DCR7 0	DCR6 0	DCR5 0	DCR4 0	DCR3 0	DCR2 0	DCR1 0	DCR0 0
1F	ATICRH Reset Value	0	0	0	0	ICR11 0	ICR10 0	ICR9 0	ICR8 0
20	ATICRL Reset Value	ICR7 0	ICR6 0	ICR5 0	ICR4 0	ICR3 0	ICR2 0	ICR1 0	ICR0 0

SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.3.2 Slave Select Management

As an alternative to using the \overline{SS} pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 44)

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

In Master mode:

- \overline{SS} internal must be held high continuously

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 43):

If CPHA=1 (data latched on 2nd clock edge):

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

If CPHA=0 (data latched on 1st clock edge):

- \overline{SS} internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If \overline{SS} is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 11.4.5.3).

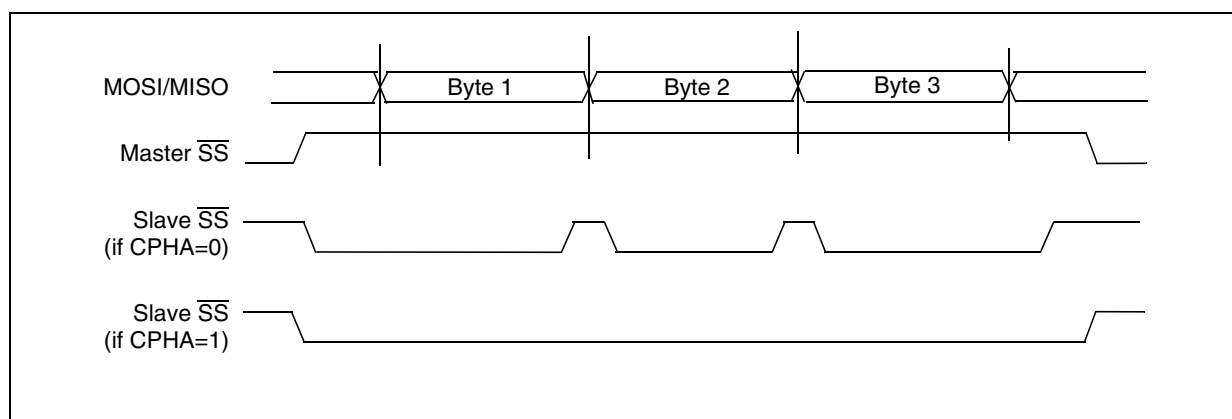
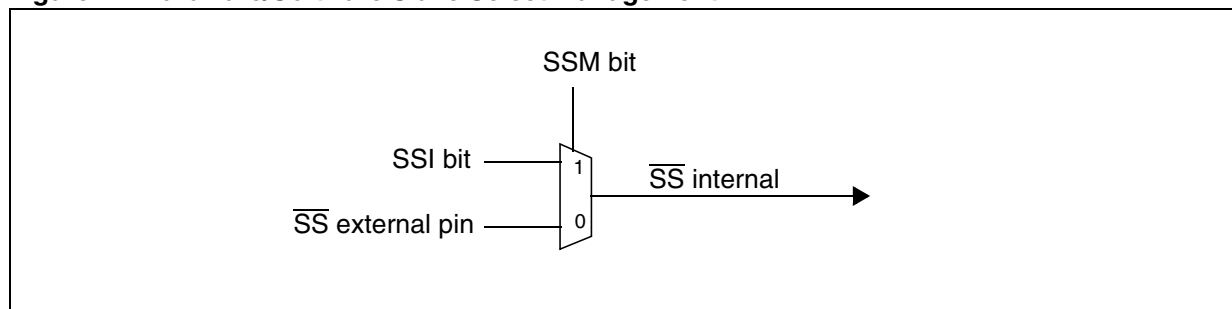
Figure 43. Generic \overline{SS} Timing Diagram

Figure 44. Hardware/Software Slave Select Management



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 46](#)).

0: No write collision occurred

1: A write collision has been detected

Bit 5 = **OVR** *SPI Overrun error (Read only).*

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

0: No overrun error

1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only).*

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

0: No master mode fault detected

1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **SOD** *SPI Output Disable.*

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

0: SPI output enabled (if SPE=1)

1: SPI output disabled

Bit 1 = **SSM** *SS Management.*

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

0: Hardware management (SS managed by external pin)

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

Bit 0 = **SSI** *SS Internal Mode.*

This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the 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 41](#)).

11.5 10-BIT A/D CONVERTER (ADC)

11.5.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 7 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 7 different sources.

The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

11.5.2 Main Features

- 10-bit conversion
- Up to 7 channels with multiplexed input
- Linear successive approximation

- Data register (DR) which contains the results
- Conversion complete status flag
- On/off bit (to reduce consumption)

The block diagram is shown in [Figure 48](#).

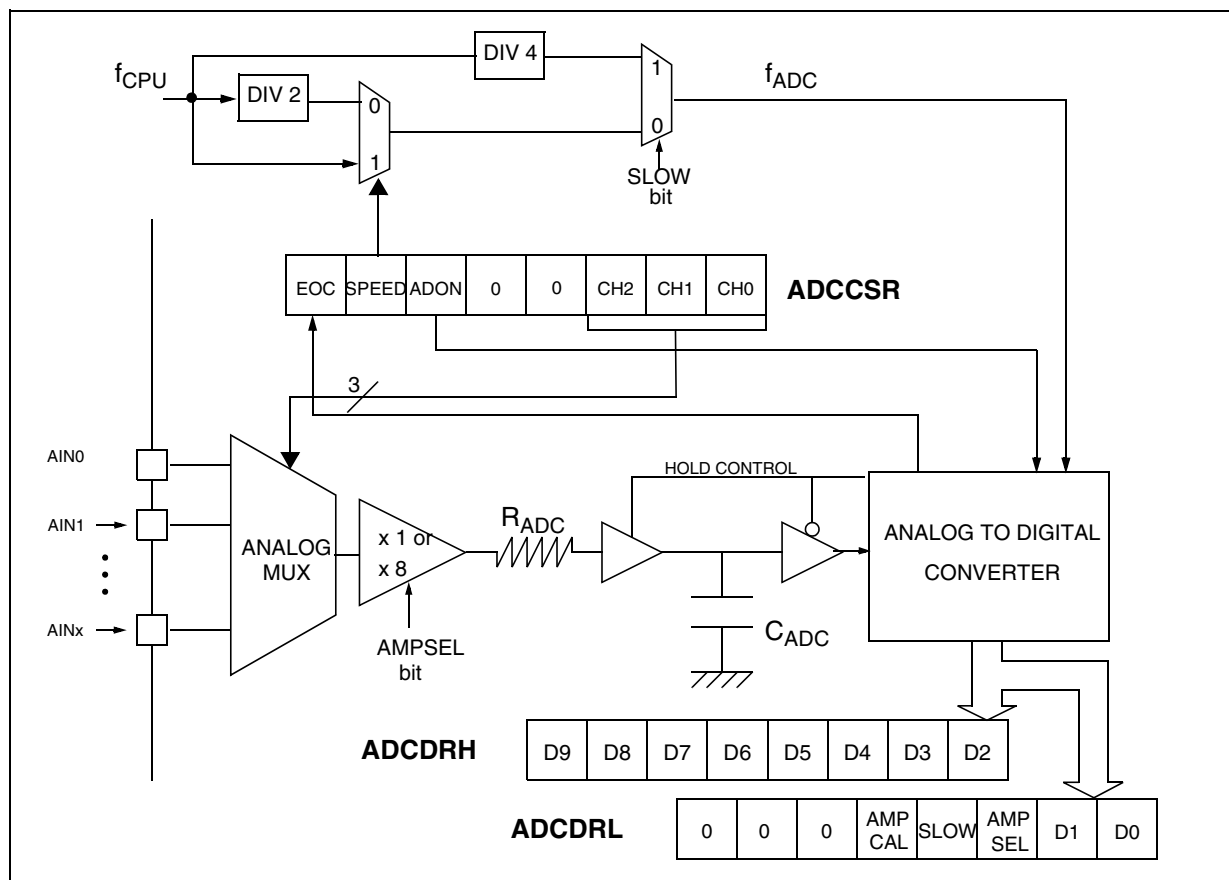
11.5.3 Functional Description

11.5.3.1 Analog Power Supply

V_{DDA} and V_{SSA} are the high and low level reference voltage pins. In some devices (refer to device pin out description) they are internally connected to the V_{DD} and V_{SS} pins.

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

Figure 48. ADC Block Diagram



ST7 ADDRESSING MODES (Cont'd)**12.1.1 Inherent**

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

Inherent Instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait For Interrupt (Low Power Mode)
HALT	Halt Oscillator (Lowest Power Mode)
RET	Sub-routine Return
IRET	Interrupt Sub-routine Return
SIM	Set Interrupt Mask
RIM	Reset Interrupt Mask
SCF	Set Carry Flag
RCF	Reset Carry Flag
RSP	Reset Stack Pointer
LD	Load
CLR	Clear
PUSH/POP	Push/Pop to/from the stack
INC/DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
MUL	Byte Multiplication
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles

12.1.2 Immediate

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

Immediate Instruction	Function
LD	Load
CP	Compare
BCP	Bit Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Operations

12.1.3 Direct

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

The direct addressing mode consists of two sub-modes:

Direct (short)

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

Direct (long)

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

12.1.4 Indexed (No Offset, Short, Long)

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

The indirect addressing mode consists of three sub-modes:

Indexed (No Offset)

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

Indexed (Short)

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

Indexed (long)

The offset is a word, thus allowing 64 Kbyte addressing space and requires 2 bytes after the opcode.

12.1.5 Indirect (Short, Long)

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

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

Indirect (short)

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

Indirect (long)

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

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 $T_A=25^{\circ}\text{C}$ and $T_A=T_{A\text{max}}$ (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 ($\text{mean} \pm 3\Sigma$).

13.1.2 Typical values

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

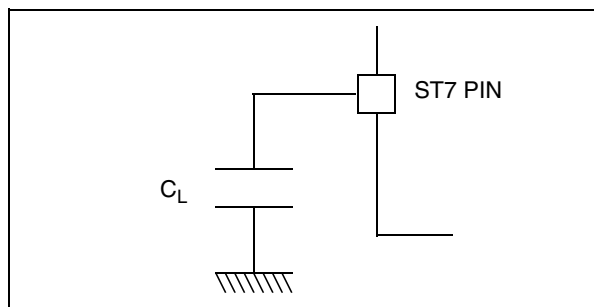
13.1.3 Typical curves

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

13.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in [Figure 49](#).

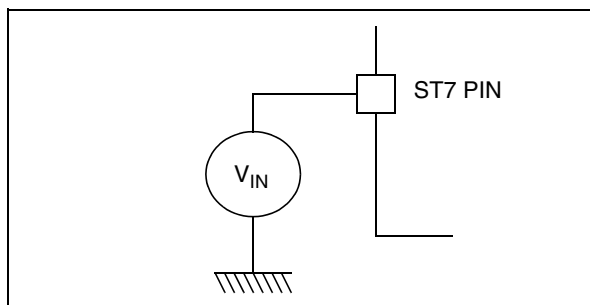
Figure 49. Pin loading conditions



13.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in [Figure 50](#).

Figure 50. Pin input voltage



OPERATING CONDITIONS (Cont'd)

13.3.4.2 Devices with "'6" order code suffix (tested for $T_A = -40$ to $+85^\circ\text{C}$) @ $V_{DD} = 2.7$ to 3.3V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal RC oscillator frequency	RCCR = FF (reset value), $T_A=25^\circ\text{C}$, $V_{DD}= 3.0\text{V}$		560		kHz
		RCCR=RCCR1 ²⁾ , $T_A=25^\circ\text{C}$, $V_{DD}= 3\text{V}$		700		
ACC _{RC}	Accuracy of Internal RC oscillator when calibrated with RCCR=RCCR1 ¹⁾²⁾	$T_A=25^\circ\text{C}$, $V_{DD}=3\text{V}$	-2		+2	%
		$T_A=25^\circ\text{C}$, $V_{DD}=2.7$ to 3.3V	-25		+25	%
		$T_A=-40$ to $+85^\circ\text{C}$, $V_{DD}=3\text{V}$	-15		15	%
I _{DD(RC)}	RC oscillator current consumption	$T_A=25^\circ\text{C}$, $V_{DD}=3\text{V}$		700 ¹⁾		μA
t _{su(RC)}	RC oscillator setup time	$T_A=25^\circ\text{C}$, $V_{DD}=3\text{V}$			10 ²⁾	μs
f _{PLL}	x4 PLL input clock			0.7 ¹⁾		MHz
t _{LOCK}	PLL Lock time ⁵⁾			2		ms
t _{STAB}	PLL Stabilization time ⁵⁾			4		ms
ACC _{PLL}	x4 PLL Accuracy	$f_{RC} = 1\text{MHz}$ @ $T_A=25^\circ\text{C}$, $V_{DD}=2.7$ to 3.3V		0.1 ⁴⁾		%
		$f_{RC} = 1\text{MHz}$ @ $T_A=40$ to $+85^\circ\text{C}$, $V_{DD}= 3\text{V}$		0.1 ⁴⁾		%
t _{w(JIT)}	PLL jitter period	$f_{RC} = 1\text{MHz}$		8 ³⁾		kHz
JIT _{PLL}	PLL jitter ($\Delta f_{CPU}/f_{CPU}$)			1 ³⁾		%
I _{DD(PLL)}	PLL current consumption	$T_A=25^\circ\text{C}$		190 ¹⁾		μA

Notes:

1. Data based on characterization results, not tested in production
2. RCCR1 is a factory-calibrated setting for 700MHz with ± 0.2 accuracy @ $T_A = 25^\circ\text{C}$, $V_{DD}=3\text{V}$. See "INTERNAL RC OSCILLATOR ADJUSTMENT" on page 23.
3. Guaranteed by design.
4. Averaged over a 4ms period. After the LOCKED bit is set, a period of t_{STAB} is required to reach ACC_{PLL} accuracy
5. After the LOCKED bit is set ACC_{PLL} is max. 10% until t_{STAB} has elapsed. See [Figure 11 on page 24](#).

15.4 ST7 APPLICATION NOTES

Table 26. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
APPLICATION EXAMPLES	
AN1658	SERIAL NUMBERING IMPLEMENTATION
AN1720	MANAGING THE READ-OUT PROTECTION IN FLASH MICROCONTROLLERS
AN1755	A HIGH RESOLUTION/PRECISION THERMOMETER USING ST7 AND NE555
AN1756	CHOOSING A DALI IMPLEMENTATION STRATEGY WITH ST7DALI
EXAMPLE DRIVERS	
AN 969	SCI COMMUNICATION BETWEEN ST7 AND PC
AN 970	SPI COMMUNICATION BETWEEN ST7 AND EEPROM
AN 971	I ² C COMMUNICATION BETWEEN ST7 AND M24CXX EEPROM
AN 972	ST7 SOFTWARE SPI MASTER COMMUNICATION
AN 973	SCI SOFTWARE COMMUNICATION WITH A PC USING ST72251 16-BIT TIMER
AN 974	REAL TIME CLOCK WITH ST7 TIMER OUTPUT COMPARE
AN 976	DRIVING A BUZZER THROUGH ST7 TIMER PWM FUNCTION
AN 979	DRIVING AN ANALOG KEYBOARD WITH THE ST7 ADC
AN 980	ST7 KEYPAD DECODING TECHNIQUES, IMPLEMENTING WAKE-UP ON KEYSTROKE
AN1017	USING THE ST7 UNIVERSAL SERIAL BUS MICROCONTROLLER
AN1041	USING ST7 PWM SIGNAL TO GENERATE ANALOG OUTPUT (SINUSOID)
AN1042	ST7 ROUTINE FOR I ² C SLAVE MODE MANAGEMENT
AN1044	MULTIPLE INTERRUPT SOURCES MANAGEMENT FOR ST7 MCUS
AN1045	ST7 S/W IMPLEMENTATION OF I ² C BUS MASTER
AN1046	UART EMULATION SOFTWARE
AN1047	MANAGING RECEPTION ERRORS WITH THE ST7 SCI PERIPHERALS
AN1048	ST7 SOFTWARE LCD DRIVER
AN1078	PWM DUTY CYCLE SWITCH IMPLEMENTING TRUE 0% & 100% DUTY CYCLE
AN1082	DESCRIPTION OF THE ST72141 MOTOR CONTROL PERIPHERALS REGISTERS
AN1083	ST72141 BLDC MOTOR CONTROL SOFTWARE AND FLOWCHART EXAMPLE
AN1105	ST7 PCAN PERIPHERAL DRIVER
AN1129	PWM MANAGEMENT FOR BLDC MOTOR DRIVES USING THE ST72141
AN1130	AN INTRODUCTION TO SENSORLESS BRUSHLESS DC MOTOR DRIVE APPLICATIONS WITH THE ST72141
AN1148	USING THE ST7263 FOR DESIGNING A USB MOUSE
AN1149	HANDLING SUSPEND MODE ON A USB MOUSE
AN1180	USING THE ST7263 KIT TO IMPLEMENT A USB GAME PAD
AN1276	BLDC MOTOR START ROUTINE FOR THE ST72141 MICROCONTROLLER
AN1321	USING THE ST72141 MOTOR CONTROL MCU IN SENSOR MODE
AN1325	USING THE ST7 USB LOW-SPEED FIRMWARE V4.X
AN1445	EMULATED 16 BIT SLAVE SPI
AN1475	DEVELOPING AN ST7265X MASS STORAGE APPLICATION
AN1504	STARTING A PWM SIGNAL DIRECTLY AT HIGH LEVEL USING THE ST7 16-BIT TIMER
AN1602	16-BIT TIMING OPERATIONS USING ST7262 OR ST7263B ST7 USB MCUS
AN1633	DEVICE FIRMWARE UPGRADE (DFU) IMPLEMENTATION IN ST7 NON-USB APPLICATIONS
AN1712	GENERATING A HIGH RESOLUTION SINEWAVE USING ST7 PWMART
AN1713	SMBUS SLAVE DRIVER FOR ST7 I ² C PERIPHERALS
AN1753	SOFTWARE UART USING 12-BIT ART
GENERAL PURPOSE	

16 IMPORTANT NOTES

16.1 EXECUTION OF BTJX INSTRUCTION

When testing the address \$FF with the "BTJT" or "BTJF" instructions, the CPU may perform an incorrect operation when the relative jump is negative and performs an address page change.

To avoid this issue, including when using a C compiler, it is recommended to never use address \$00FF as a variable (using the linker parameter for example).

16.2 ADC CONVERSION SPURIOUS RESULTS

Spurious conversions occur with a rate lower than 50 per million. Such conversions happen when the measured voltage is just between 2 consecutive digital values.

Workaround

A software filter should be implemented to remove erratic conversion results whenever they may cause unwanted consequences.

16.3 A/D CONVERTER ACCURACY FOR FIRST CONVERSION

When the ADC is enabled after being powered down (for example when waking up from HALT, ACTIVE-HALT or setting the ADON bit in the ADCCSR register), the first conversion (8-bit or 10-bit) accuracy does not meet the accuracy specified in the datasheet.

Workaround

In order to have the accuracy specified in the datasheet, the first conversion after a ADC switch-on has to be ignored.

16.4 NEGATIVE INJECTION IMPACT ON ADC ACCURACY

Injecting a negative current on an analog input pins significantly reduces the accuracy of the AD Converter. Whenever necessary, the negative in-

jection should be prevented by the addition of a Schottky diode between the concerned I/Os and ground.

Injecting a negative current on digital input pins degrades ADC accuracy especially if performed on a pin close to ADC channel in use.

16.5 CLEARING ACTIVE INTERRUPTS OUTSIDE INTERRUPT ROUTINE

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

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, i.e. when:

- The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine
- The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine
- The interrupt request is cleared (flag reset or interrupt mask) in any part of the code while this interrupt is disabled

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

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

Ex:

SIM

reset flag or interrupt mask

RIM

16.6 USING PB4 AS EXTERNAL INTERRUPT

PB4 cannot be used as an external interrupt in HALT mode because the port pin PB4 is not active in this mode.

Dec-2004	2.0	<p>Changed FMP_R option bit description in section 15.1 on page 121</p> <p>Changed “DEVELOPMENT TOOLS” on page 124</p> <p>Added notes indicating that PB4 cannot be used as an external interrupt in HALT mode, section 16.6 on page 128 and Section 8.3 PERIPHERAL INTERRUPTS</p> <p>Added “NEGATIVE INJECTION IMPACT ON ADC ACCURACY” on page 128</p> <p>Added “CLEARING ACTIVE INTERRUPTS OUTSIDE INTERRUPT ROUTINE” on page 128</p> <p>Changed Table 23, “List of valid option combinations,” on page 122: PLLx4x8 selection when PLL off</p>
----------	-----	---