ST7FOXA0

8-bit MCU with single voltage Flash memory, ADC, timers

Features

- Memories
  - 2 Kbytes single voltage extended Flash (XFlash) Program memory with Read-Out Protection
    In-Circuit Programming and In-Application programming (ICP and IAP)
    Endurance: 1K write/erase cycles guaranteed
    Data retention: 20 years at 55 °C
  - 128 bytes RAM
- Clock, Reset and Supply Management
  - Low voltage supervisor (LVD) for safe power-on/off
  - Clock sources: Internal trimmable 8 MHz RC oscillator, auto wakeup internal low power - low frequency oscillator or external clock
  - External reset source and watchdog reset
  - Five power saving modes: Halt, Active-Halt, Auto Wakeup from Halt, Wait and Slow
- I/O Ports
  - 5 multifunctional bidirectional I/Os
  - 1 additional output line
  - 5 high sink outputs

SO8 DIP8

- 2 timers
  - One 8-bit Lite timer with prescaler including: watchdog,
    1 real time base and 1 input capture
  - Single 12-bit Auto-reload timer with 1 PWM output, input capture, output compare, dead-time generation and enhanced one pulse mode functions
- A/D converter: 5 input channels
- Interrupt management
  - 11 interrupt vectors plus TRAP and RESET
- Instruction set
  - 8-bit data manipulation
  - 63 basic instructions with illegal opcode detection
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instructions
- Development tools
  - Full HW/SW development package
  - DM (Debug Module)

Table 1. Device summary

Features	ST7FOXA0
Program memory - bytes	2K
RAM (stack) - bytes	128 (64)
Timers	1 x 8-bit timer, 1 x 12-bit AT (1 PWM)
ADC	1 x 10-bit
Packages	SO8 150”, DIP8 300”

Contents

List of tables

List of figures

1 Description

The ST7FOXA0 is a member of the ST7 microcontroller family. All ST7 devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

The device is positioned at the entry level of the 8-bit microcontroller range providing an attractive cost while at the same time embedding the most advanced features.

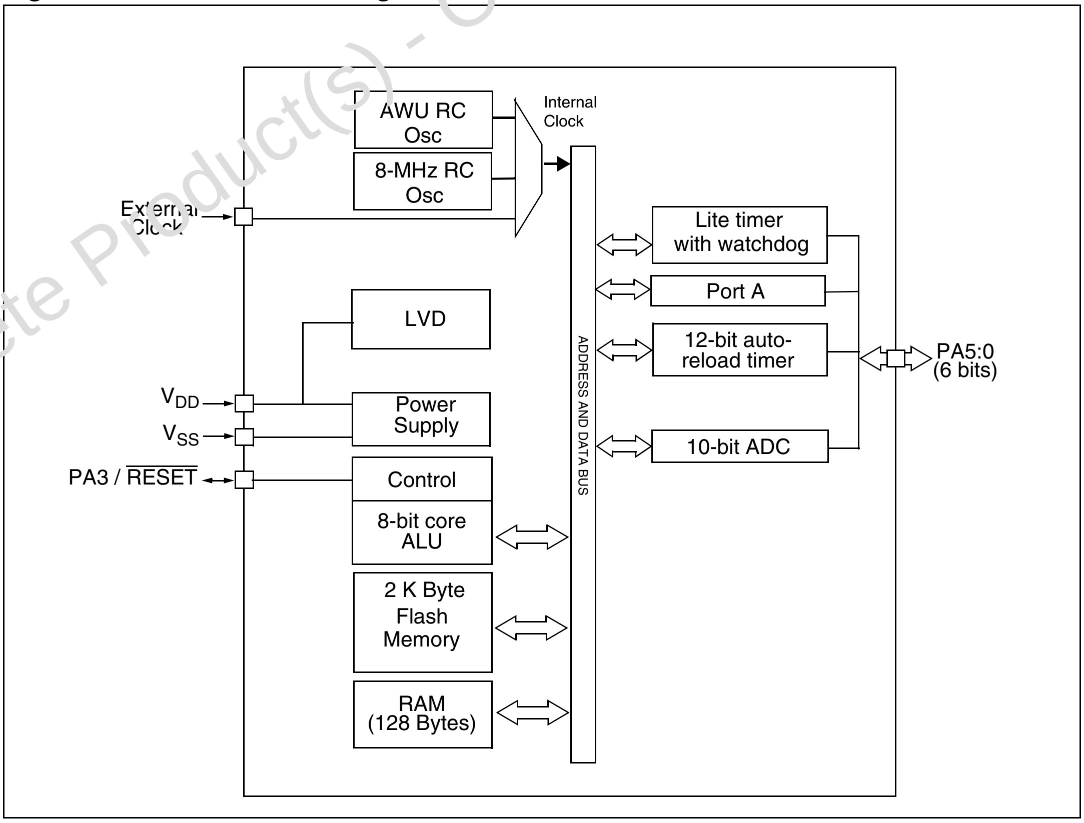
The ST7FOXA0 features Flash memory with byte-by-byte In-Circuit Programming (ICP) and In-Application Programming (IAP) capability.

Under software control, the ST7FOXA0 device can be placed in Wait, Slow, or Halt mode, reducing power consumption when the application is in idle or standby state.

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

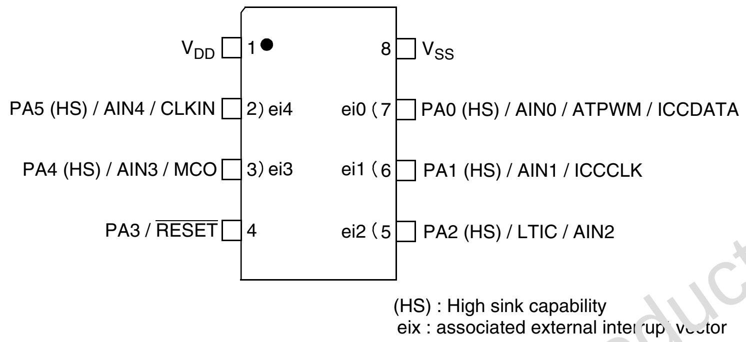
The ST7FOXA0 features an on-chip Debug Module (DM) to support In-Circuit Debugging (ICD). For a description of the DM registers, refer to the ST7 ICC Protocol Reference Manual.

Figure 1. General block diagram



2 Pin description

Figure 2. 8-pin package pinout



Legend / Abbreviations for *Table 2*:

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

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

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

Port and control configuration:

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

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

Table 2. Device pin description (8-pin package)

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_{DD} (1)	S									Main power supply	
2	PA5/AIN4/ CLKIN	I/O	C_T	HS	**X**	ei4		X	X	X	**Port A5**	Analog input 4 or External Clock Input
3	PA4/AIN3/MCO	I/O	C_T	HS	**X**	ei3		X	X	X	**Port A4**	Analog input 3 or Main clock output
4	PA3/$\overline{RESET}$ (2)	O				X			X	X	**Port A3**	$\overline{RESET}$ (2)
5	PA2/AIN2/LTIC	I/O	C_T	HS	**X**	ei2		X	X	X	**Port A2**	Analog input 2 or Lite Timer Input Capture
6	PA1/AIN1/ ICCCLK	I/O	C_T	HS	X	**ei1**		X	X	X	**Port A1**	Analog input 1 or 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 pull-up
7	PA0/AIN0/ ATPWM/ ICCDATA	I/O	C_T	HS	**X**	ei0		X	X	X	**Port A0**	Analog input 0 or Auto-Reload Timer PWM or In Circuit Communication Data
8	V_{SS} (1)	S									Ground	

1. It is mandatory to connect all available V_{DD} and V_{DDA} pins to the supply voltage and all V_{SS} and V_{SSA} pins to ground.
2. After a reset, the multiplexed PA3/RESET pin will act as $\overline{RESET}$. To configure this pin as output (Port A3), write 55h to MUXCR0 and AAh to MUXCR1.

3 Register and memory mapping

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 128 bytes of register locations, 128 bytes of RAM and 2 Kbytes of Flash program memory. The RAM space includes up to 64 bytes for the stack from C0h to FFh.

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

The Flash memory contains two sectors (see *Figure 3*) mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (FFE0h-FFFFh).

The size of Flash Sector 0 and other device options are configurable by option bytes (refer to *Section 13.1 on page 110*).

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

Figure 3. ST7FOXA0 memory map

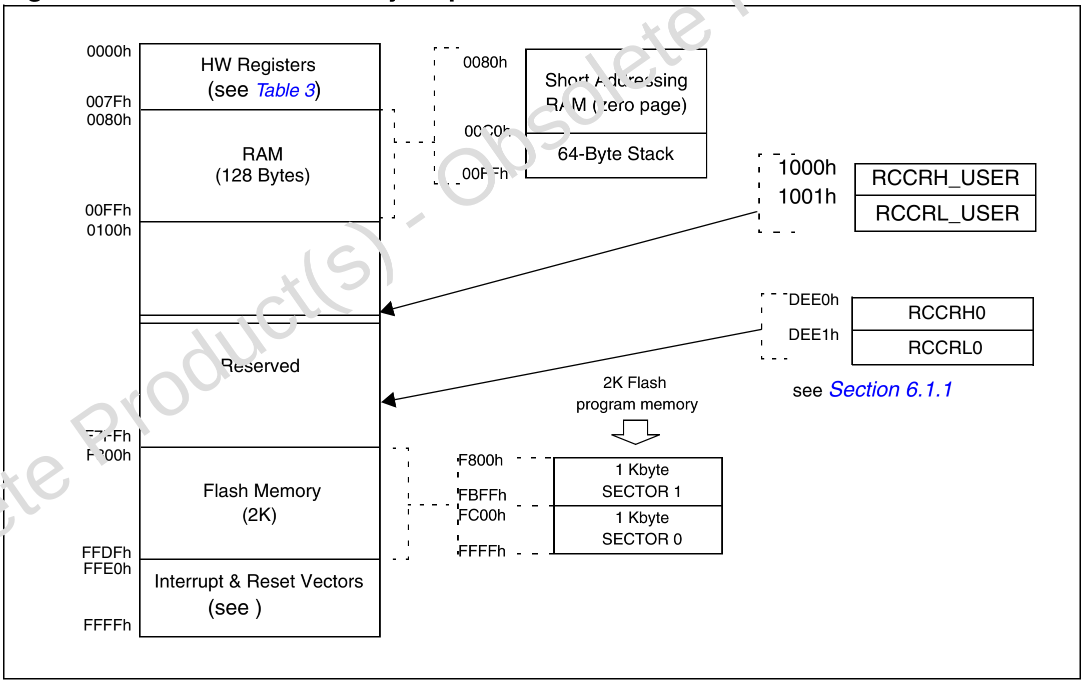


Table 3. ST7FOXA0 Hardware register map[1]

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 [2] 08h 02h [3]	R/W R/W R/W
0003h to 000Ah	Reserved area (8 bytes)				

Table 3. ST7FOXA0 Hardware register map(1) (continued)

Address	Block	Register label	Register name	Reset status	Remarks
000Bh 000Ch	LITE TIMER	LTCSR LTICR	Lite Timer Control/Status Register Lite Timer Input Capture Register	0xh 00h	R/W Read Only
000Dh 000Eh 000Fh 0010h 0011h 0012h 0013h	AUTO-RELOAD TIMER	ATCSR CNTRH CNTRL ATRH ATRL PWMCR PWM0CSR	Timer Control/Status Register Counter Register High Counter Register Low Auto-Reload Register High Auto-Reload 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	AUTO-RELOAD 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
0030h	RC Calibration	RCC_CSR	RC calibration Control/Status register	00h	R/W
0031h to 0033h			Reserved area (3 bytes)		
0034h 0035h 0036h	ADC	ADCCSR ADCDRH ADCDRL	A/D Control Status Register A/D Data Register High A/D Data Register Low	00h xxh 00h	R/W Read Only R/W
0037h	ITC	EICR1	External Interrupt Control Register 1	00h	R/W
0038h	MCC	MCCSR	Main Clock Control/Status Register	00h	R/W
0039h 003Ah	Clock and Reset	RCCRH RCCRL	RC oscillator Control Register High RC oscillator Control Register Low	FFh 0000 0x00b	R/W R/W
003Bh to 003Ch			Reserved area (2 bytes)		
003Dh	ITC	EICR2	External Interrupt Control Register 2	00h	R/W
003Eh 003Fh	Clock controller	PSCR CKCNTCSR	Prescaler Register Clock Controller Control/Status Register	03h 09h	R/W R/W
0040h to 0046h			Reserved area (7 bytes)		
0047h 0048h	MuxIO-reset	MUXCR0 MUXCR1	Mux IO-Reset Control Register 0 Mux IO-Reset Control Register 1	00h 00h	R/W R/W
0049h 004Ah	AWU	AWUPR AWUCSR	AWU Prescaler Register AWU Control/Status Register	FFh 00h	R/W R/W

Table 3. ST7FOXA0 Hardware register map[1] (continued)

Address	Block	Register label	Register name	Reset status	Remarks
004Bh 004Ch 004Dh 004Eh 004Fh 0050h	DM [4]	DMCR DMSR DMBK1H DMBK1L DMBK2H DMBK2L	DM Control Register DM Status Register DM Breakpoint Register 1 High DM Breakpoint Register 1 Low DM Breakpoint Register 2 High DM Breakpoint Register 2 Low	00h 00h 00h 00h 00h 00h	R/W R/W R/W R/W R/W R/W
0051h to 007Fh	Reserved area (47 bytes)				

1. Legend: x=undefined, R/W=read/write.
2. 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.
3. The bits associated with unavailable pins must always keep their reset value.
4. For a description of the Debug Module registers, see ICC protocol reference manual.

4 Flash programmable 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 can be programmed or erased.
- In-Circuit Programming. In this mode, Flash sectors 0 and 1, option byte row can be programmed or erased without removing the device from the application board.
- In-Application Programming. In this mode, sector 1 can be programmed or erased without removing the device from the application board and while the application is running.

4.3.1 In-Circuit Programming (ICP)

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

Switch the ST7 to ICC mode (In-Circuit Communications). This is done by driving a specific signal sequence on the ICCCLK/DATA pins while the $\overline{\text{RESET}}$ pin is pulled low. When the ST7 enters ICC mode, it fetches a specific Reset vector which points to the ST7 System Memory containing the ICC protocol routine. This routine enables the ST7 to receive bytes from the ICC interface.

- Download ICP Driver code in RAM from the ICCDATA pin
- Execute ICP Driver code in RAM to program the Flash memory

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

4.3.2 In Application Programming (IAP)

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

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored etc.)
IAP mode can be used to program any memory areas except Sector 0, which is Write/Erase protected to allow recovery in case errors occur during the programming operation.

4.4 ICC interface

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

- $\overline{RESET}$: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- OSC1: main clock input for external source
- V_{DD}: application board power supply (optional, see Note 3)

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

2 *During the ICP session, the programming tool must control the $\overline{RESET}$ 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<1 kΩ). A schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with R>1 kΩ or a reset management IC with open drain output and pull-up resistor>1 kΩ, 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.*

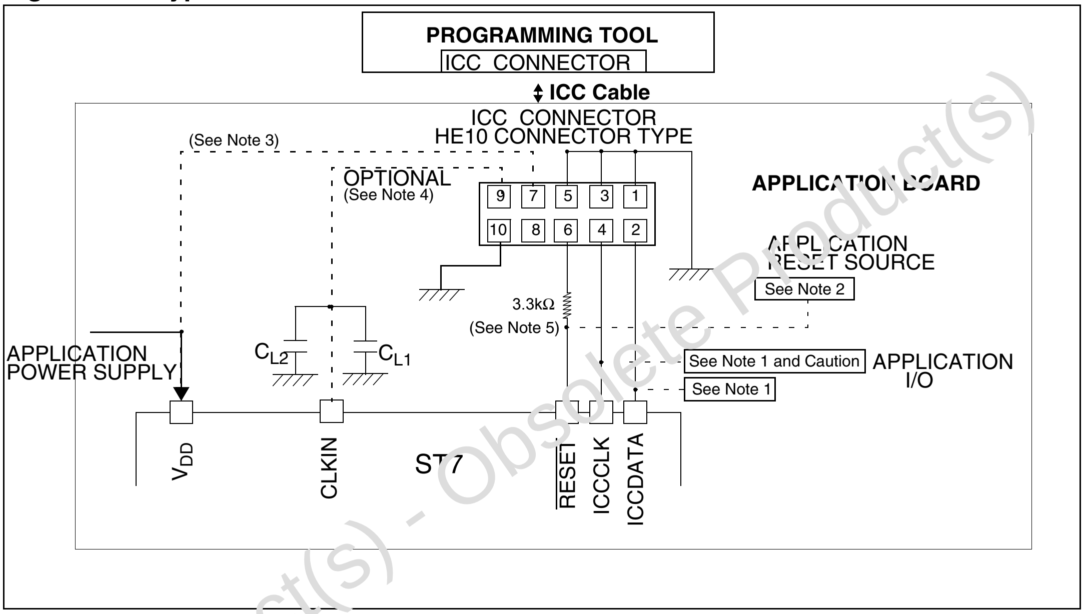
4 *In “enabled option byte” mode (38-pulse ICC mode), the internal RC oscillator is forced as a clock source, regardless of the selection in the option byte. In “disabled option byte” mode (35-pulse ICC mode), pin 9 has to be connected to the CLKIN pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte.*

5 *A serial resistor must be connected to ICC connector pin 6 in order to prevent contention on PA3/RESET pin. Contention may occur if a tool forces a state on RESET pin while PA3 pin forces the opposite state in output mode. The resistor value is defined to limit the current*

below 2mA at 5V. If PA3 is used as output push-pull, then the application must be switched off to allow the tool to take control of the RESET pin (PA3). To allow the programming tool to drive the RESET pin below V_{IL}, special care must also be taken when a pull-up is placed on PA3 for application reasons.

Caution: During normal operation the ICCCLK pin must be internally or externally pulled- up (external pull-up of 10 kΩ mandatory in noisy environment) 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



4.5 Memory protection

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

4.5.1 Read-out protection

Read-Out Protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller.

- In Flash devices, this protection is removed by reprogramming the option. In this case, the program memory is automatically erased and the device can be reprogrammed. The read-out protection is enabled and removed through the FMP_R bit in the option byte.

4.5.2 Flash write/erase protection

Write/Erase Protection, when set, makes it impossible to both overwrite and erase program memory. Its purpose is to provide advanced security to applications and prevent any change being made to the memory content. Write/Erase Protection is enabled through the FMP_W bit in the option byte.

Caution: **Once set, Write/Erase Protection can never be removed. A write-protected Flash device is no longer reprogrammable.**

4.6 Related documentation

For details on Flash programming and ICC protocol, refer to the ST7 Flash Programming Reference Manual and to the ST7 ICC Protocol Reference Manual.

4.7 Description of Flash Control/Status register (FCSR)

This register controls the XFlash erasing and programming using ICP, IAP or other programming methods.

1st RASS Key: 0101 0110 (56h)

2nd RASS Key: 1010 1110 (AEh)

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

Reset value: 000 0000 (00h)

7							0
0	0	0	0	0	OPT	LAT	PGM
Read/write							

Table 4. Flash register mapping 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 Central processing unit

5.1 Introduction

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

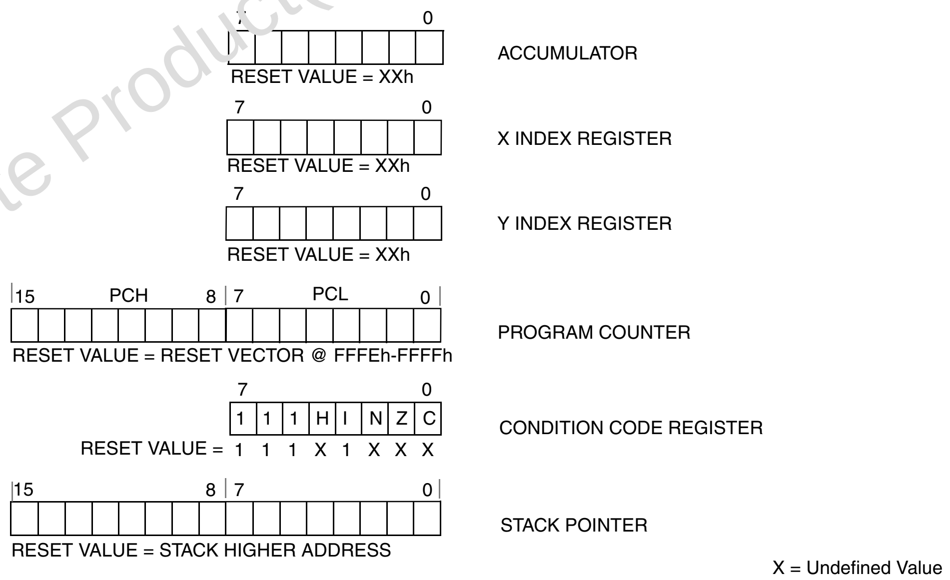
5.2 Main features

- 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

5.3 CPU registers

The six CPU registers shown in *Figure 5*. They are not present in the memory mapping and are accessed by specific instructions.

Figure 5. CPU registers



5.3.1 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.

5.3.2 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).

5.3.3 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).

5.3.4 Condition Code register (CC)

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.

Reset value: 111x 1xxx

7							0
1	1	1	H	I	N	Z	C
Read/write							

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

Arithmetic management bits

Bit 4 = **H** *Half carry bit*

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 bit*

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 bit*

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 bit*

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 bit*

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.

5.3.5 Stack Pointer (SP)

Reset value: 00FFh

15							8	7							0
0	0	0	0	0	0	0	0	1	1	SP5	SP4	SP3	SP2	SP1	SP0
Read/write															

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 6*).

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) can be directly accessed by a LD instruction.

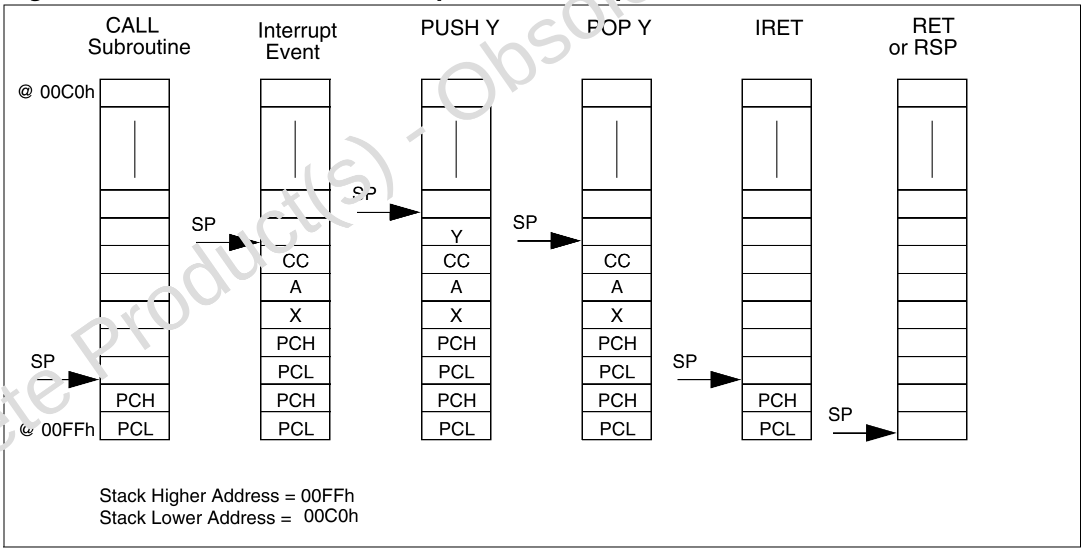
Note: When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in *Figure 6*.

- 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 6. ST7FOXA0 stack manipulation example



6 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. The main features are the following:

- Clock management
  - 8 MHz internal RC oscillator (enabled by option byte)
  - Auto wakeup RC oscillator (enabled by option byte)
  - External clock input (enabled by option byte)
- Reset Sequence Manager (RSM)
- System Integrity management (SI)
  - Main supply Low voltage detection (LVD) with reset generation (enabled by option byte)

6.1 RC oscillator adjustment

6.1.1 Internal RC oscillator

The device contains an internal RC oscillator with a specific accuracy for a given device, temperature and voltage range (4.5 V - 5.5 V). It must be calibrated to obtain the frequency required in the application. This is done by software writing a 10-bit calibration value in the RCCRH (RC Control register High) and in the bits 6:5 in the RCCRL (RC Control register Low).

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 for 5 V V_{DD} supply voltage at 25 °C (see *Table 5*).

Table 5. Predefined RC oscillator calibration values

RCCR	Conditions	ST7FOX Address
RCCRH	V_{DD}= 5V T_A= 25°C f_{RC} = 8 MHz	DEE0h(1) (CR[9:2])
RCCRL		DEE1h(1) (CR[1:0])

1. The DEE0h and DEE1h addresses are located in a reserved area in non-volatile memory. They are read-only bytes for the application code. This area cannot be erased or programmed by any ICC operations. For compatibility reasons with the RCCRL register, CR[1:0] bits are stored in the 5th and 6th position of DEE1 address.

In 38-pulse ICC mode, the internal RC oscillator is forced as a clock source, regardless of the selection in the option byte.

Section 12: Electrical characteristics on page 92 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 100 nF, between the V_{DD} and V_{SS} pins and also between the V_{DDA} and V_{SSA} pins as close as possible to the ST7 device.

These bytes are systematically programmed by ST.

6.1.2 Customized RC calibration

If the application requires a higher frequency accuracy or if the voltage or temperature conditions change in the application, the frequency may need to be recalibrated. Two non-volatile bytes (RCCRH_USER and RCCRL_USER) are reserved for storing these new values. These two-byte area is Electrically Erasable Programmable Read Only Memory.

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

How to program RCCRH_USER and RCCRL_USER

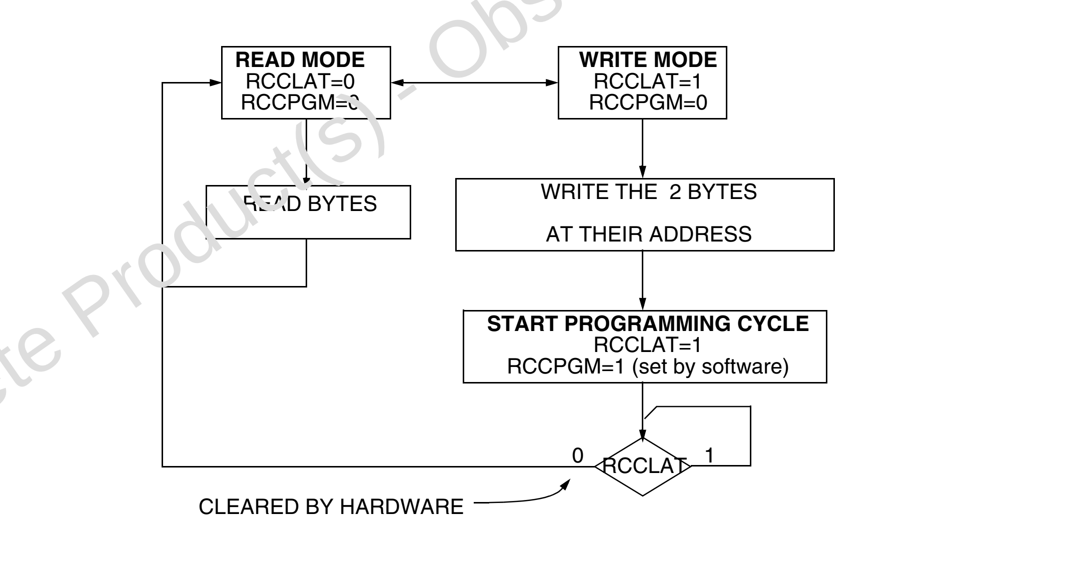
To access the write mode, the RCCLAT bit has to be set by software (the RCCPGM bit remains cleared). When a write access to this two-byte area occurs, the values are latched.

When RCCPGM bit is set by the software, the latched data are programmed in the EEPROM cells. To avoid wrong programming, the user must take care to only access these two-byte addresses.

At the end of the programming cycle, the RCCPGM and RCCLAT bits are cleared simultaneously.

Note: *During the programming cycle, it is forbidden to access the latched data (see Figure 7).*

Figure 7. RCCRH_USER and RCCRL_USER programming flowchart



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

Access error handling

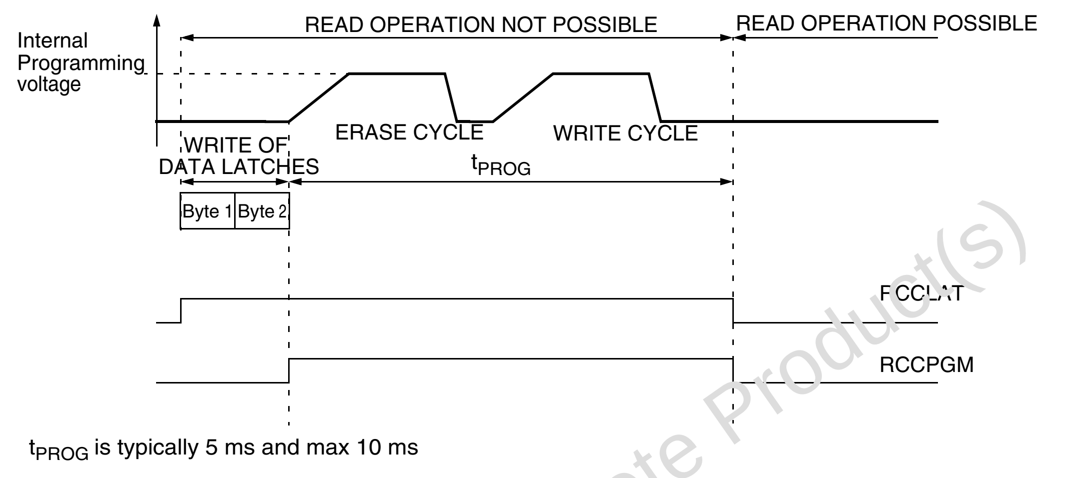
If a read access occurs while RCCLAT=1, then the data bus will not be driven.

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

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

Caution: When the Read-Out Protection is enabled through an option bit (see *Section 13.1: Option bytes*), these two bytes are protected against Read-out (including a re-write protection). In Flash devices, when this protection is removed by reprogramming the option byte, these two bytes are automatically erased.

Figure 8. RC user calibration programming cycle



6.1.3 Auto wakeup RC oscillator

The ST7FOX also contains an Auto wakeup RC oscillator. This RC oscillator should be enabled to enter Auto wakeup from halt mode.

The Auto wakeup (AWU) RC oscillator can also be configured as the startup clock through the CKSEL[1:0] option bits (see *Section 13.1: Option bytes on page 110*).

This is recommended for applications where very low power consumption is required.

Switching from one startup clock to another can be done in run mode as follows (see *Figure 9*):

Case 1 Switching from internal RC to AWU

1. Set the RC/AWU bit in the CKCNTCSR register to enable the AWU RC oscillator
2. The RC_FLAG is cleared and the clock output is at 1.
3. Wait 3 AWU RC cycles till the AWU_FLAG is set
4. The switch to the AWU clock is made at the positive edge of the AWU clock signal
5. Once the switch is made, the internal RC is stopped

Case 2 Switching from AWU RC to internal RC

1. Reset the RC/AWU bit to enable the internal RC oscillator
2. Using a 4-bit counter, wait until 8 internal RC cycles have elapsed. The counter is running on internal RC clock.
3. Wait till the AWU_FLAG is cleared (1AWU RC cycle) and the RC_FLAG is set (2 RC cycles)
4. The switch to the internal RC clock is made at the positive edge of the internal RC clock signal
5. Once the switch is made, the AWU RC is stopped

Note: 1 When the internal RC is not selected, it is stopped so as to save power consumption.

2 When the internal RC is selected, the AWU RC is turned on by hardware when entering Auto wakeup from Halt mode.

3 When the external clock is selected, the AWU RC oscillator is always on.

Figure 9. Clock switching

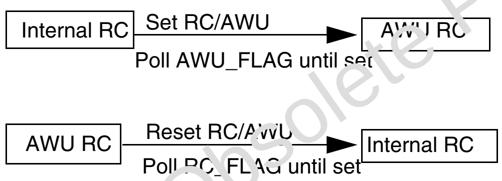
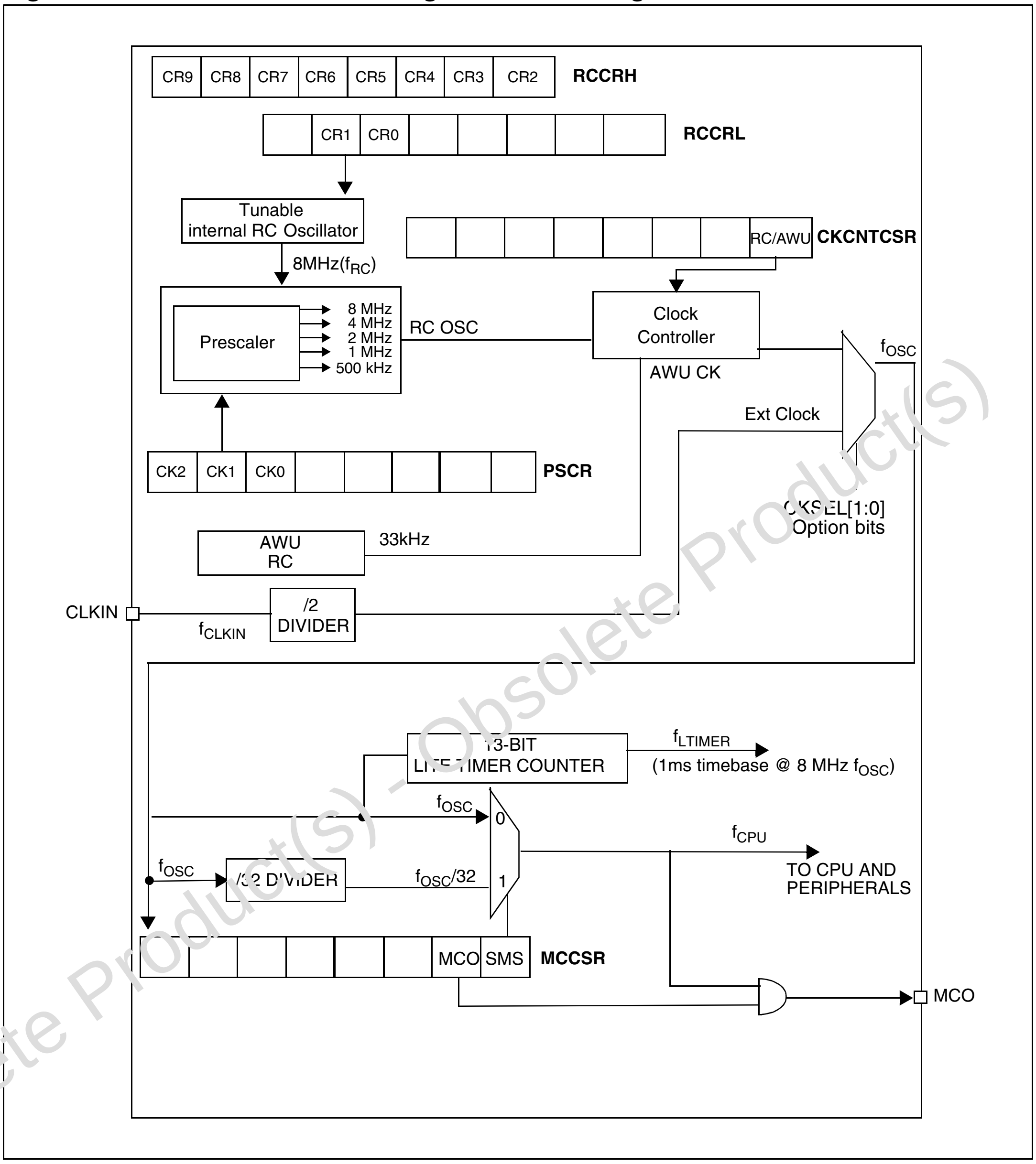


Figure 10. ST7FOXA0 clock management block diagram



6.2 Multi-oscillator (MO)

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

- An external source
- An internal high frequency RC oscillator

6.2.1 External clock source

In this external clock mode, a clock signal (square, sinus or triangle) with ~50% duty cycle has to drive CLKIN.

6.2.2 Internal RC oscillator

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

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

6.3 Reset sequence manager (RSM)

6.3.1 Introduction

The reset sequence manager includes three RESET sources as shown in *Figure 12*.

- External RESET source pulse
- Internal LVD RESET (Low Voltage Detection)
- Internal WATCHDOG RESET

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to Section 10.2.1 on page 84 for further details.

These sources act on the 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 mapping.

The basic RESET sequence consists of 3 phases as shown in *Figure 11*:

- Active Phase depending on the RESET source
- 256 or 512 CPU clock cycle delay (see *Table 6*)

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 RESET pin in low state until programming mode is entered, in order to avoid unwanted behavior.

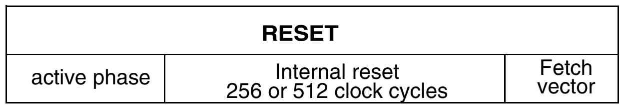
The 256 or 512 CPU clock cycle delay allows the oscillator to stabilize and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay is automatically selected depending on the clock source chosen by option byte.

The Reset vector fetch phase duration is 2 clock cycles.

Table 6. CPU clock delay during Reset sequence

Clock source	CPU clock cycle delay
Internal RC 8 MHz Oscillator	512
Internal RC 32 kHz Oscillator	256
External clock connected to CLKIN pin	512

Figure 11. ST7FOXA0 reset sequence phases



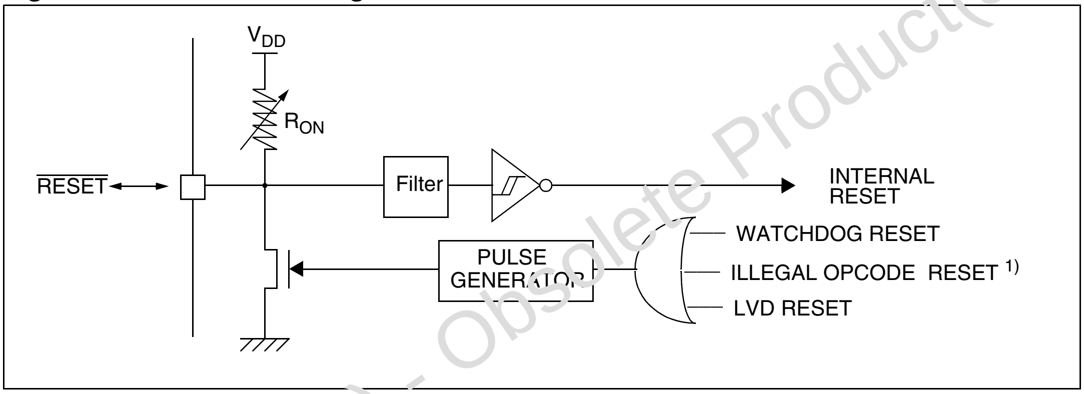
6.3.2 Asynchronous external RESET pin

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

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see *Figure 13: Reset sequences*). This detection is asynchronous and therefore the MCU can enter reset state even in Halt mode.

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

Figure 12. Reset block diagram



1. See *Section 10.2.1: Illegal opcode reset on page 84* for more details on illegal opcode reset conditions.

6.3.3 External power-on reset

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

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

6.3.4 Internal Low Voltage Detector (LVD) reset

Two different Reset sequences caused by the internal LVD circuitry can be distinguished:

- Power-On reset
- Voltage Drop reset

The device RESET pin acts as an output that is pulled low when V_{DD} is lower than V_{IT+} (rising edge) or V_{DD} lower than V_{IT-} (falling edge) as shown in *Figure 13.*

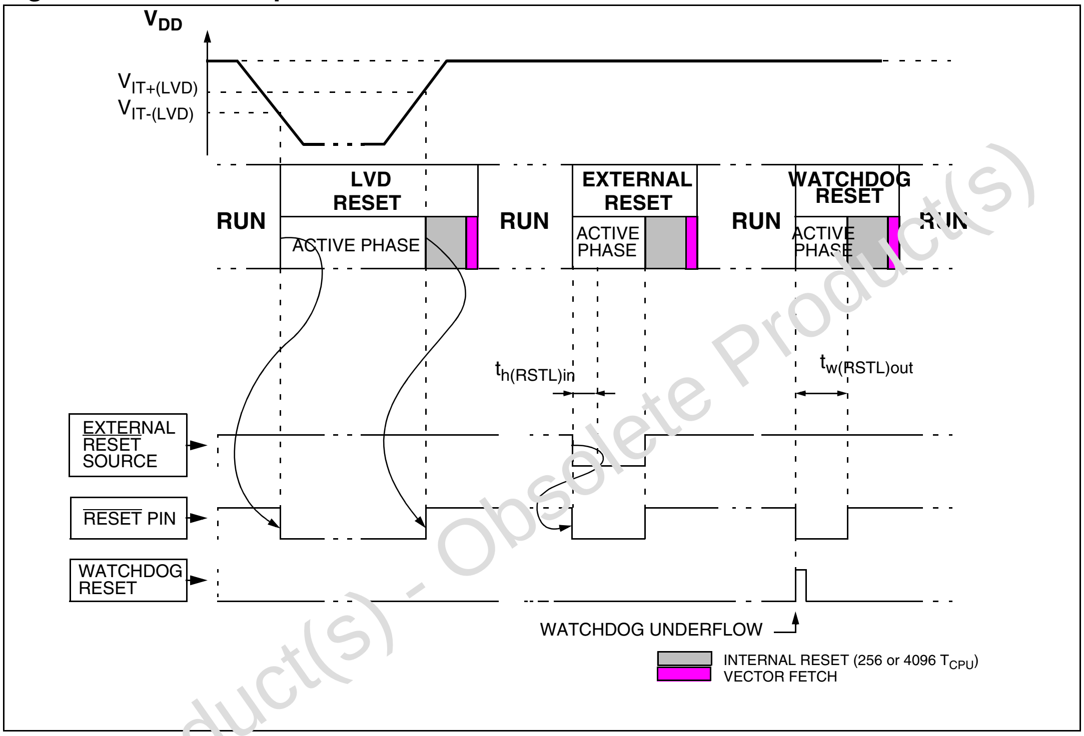
The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

6.3.5 Internal watchdog reset

The Reset sequence generated by an internal watchdog counter overflow is shown in *Figure 13: Reset sequences*

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

Figure 13. Reset sequences



6.3.6 Multiplexed IO reset control register 1 (MUXCR1)

Reset value: 0000 0000 (00h)

7							0
MIR15	MIR14	MIR13	MIR12	MIR11	MIR10	MIR9	MIR8
Read/write once only							

6.3.7 Multiplexed IO reset control register 0 (MUXCR0)

Reset value: 0000 0000 (00h)

7							0
MIR7	MIR6	MIR5	MIR4	MIR3	MIR2	MIR1	MIR0
Read/write once only							

Bits 15:0 = **MIR[15:0]**

This 16-bit register is read/write by software but can be written only once between two reset events. It is cleared by hardware after a reset; When both MUXCR0 and MUXCR1 registers are at 00h, the multiplexed PA3/RESET pin will act as RESET. To configure this pin as output (Port A3), write 55h to MUXCR0 and AAh to MUXCR1.

These registers are one-time writable only.

To configure PA3 as general purpose output:
After power-on / reset, the application program has to configure the I/O port by writing to these registers as described above. Once the pin is configured as an I/O output, it cannot be changed back to a reset pin by the application code.

To configure PA3 as RESET:
An internally generated reset (such as POR, WDG, illegal opcode) will clear the two registers and the pin will act again as a reset function. Otherwise, a power-down is required to put the pin back in reset configuration.

Table 7. Multiplexed IO register map and reset values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0047h	MUXCR0 Reset Value	MIR7 0	MIR6 0	MIR5 0	MIR4 0	MIR3 0	MIR2 0	MIR1 0	MIR0 0
0048h	MUXCR1 Reset Value	MIR15 0	MIR14 0	MIR13 0	MIR12 0	MIR11 0	MIR10 0	MIR9 0	MIR8 0

6.4 System Integrity management (SI)

The System Integrity Management block contains the Low voltage Detector (LVD).

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to Section 10.2.1 on page 84 for further details.

6.4.1 Low Voltage Detector (LVD)

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

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

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

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

The LVD function is illustrated in *Figure 14*.

The voltage threshold can be enabled/disabled by option byte. See *Section 13.1 on page 110*.

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

- Under full software control
- In static safe reset

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

During a Low Voltage Detector Reset, the $\overline{RESET}$ pin is held low, thus permitting the MCU to reset other devices.

Note: Use of LVD with capacitive power supply: with this type of power supply, if power cuts occur in the application, it is recommended to pull V_{DD} down to 0 V to ensure optimum restart conditions. Refer to circuit example in Figure 39 on page 106 and note 4.

The LVD is an optional function which can be selected by option byte. See Section 13.1 on page 110.

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

If the LVD is disabled, an external circuitry must be used to ensure a proper power-on reset.

It is recommended to make sure that the V_{DD} supply voltage rises monotonously when the device is exiting from Reset, to ensure the application functions properly.

Caution: If an LVD reset occurs after a watchdog reset has occurred, the LVD will take priority and will clear the watchdog flag.

Figure 14. Low voltage detector vs reset

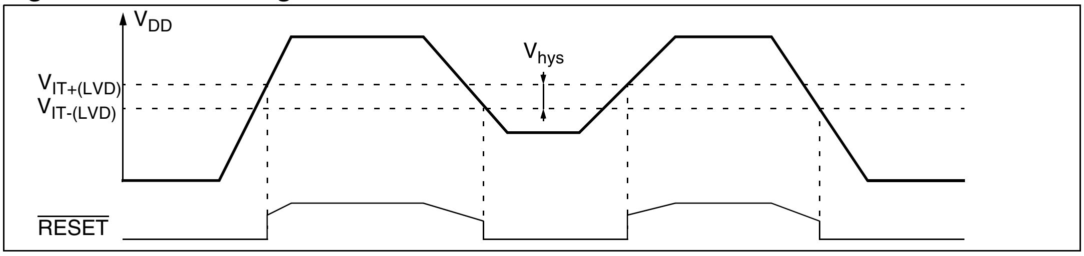
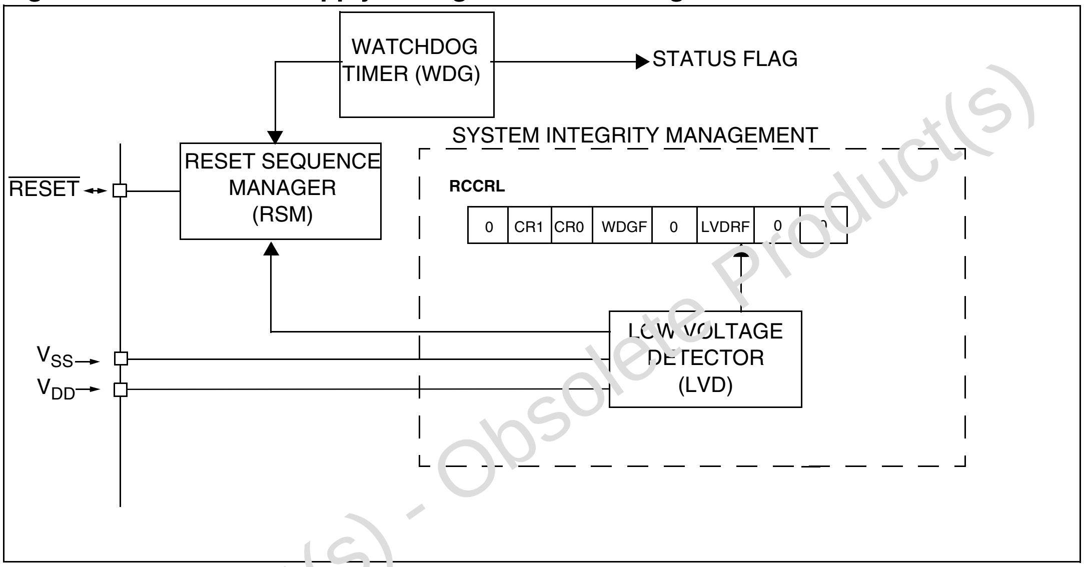


Figure 15. Reset and supply management block diagram



6.5 Register description

6.5.1 RC calibration control/status register (RCC_CSR)

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	RCCLAT	RCCPGM
Read/write							

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

0: Read mode

1: Write mode

Bit 1 = **RCCLAT** *Latch Access Transfer bit:* 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 RCCPGM bit is cleared

Bit 0 = **RCCPGM** *Programming Control and Status bit*

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 RCCPGM bit is cleared during the programming cycle, the memory data is not guaranteed.*

6.5.2 Main Clock Control/Status Register (MCCSR)

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	MCO	SMS
Read/write							

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

Bit 1 = **MCO** *Main Clock Out enable bit*

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 selection bit*

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 ($f_{CPU} = f_{OSC}/32$)

6.5.3 RC Control Register High (RCCRH)

Reset value: 1111 1111 (FFh)

7							0
CR9	CR8	CR7	CR6	CR5	CR4	CR3	CR2
Read/write							

Bits 7:0 = **CR[9:2]** *RC Oscillator Frequency Adjustment bits*

These bits must be written immediately after reset to adjust the RC oscillator frequency. The application can store the correct value for each voltage range in Flash memory and write it to this register at start-up.

00h = maximum available frequency

FFh = lowest available frequency

These bits are used with the CR[1:0] bits in the RCCRL register. Refer to *Chapter 6.5.4*.

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.

6.5.4 RC Control Register Low (RCCRL)

Reset value: 0000 0000 (00h)

7							0
0	CR1	CR0	0	0	LVDRF	0	0
Read/write							

Bit 7 = Reserved, must be kept cleared

Bits 6:5 = **CR[1:0]** *RC Oscillator Frequency Adjustment bits*

These bits, as well as CR[9:2] bits in the RCCRH register must be written immediately after reset to adjust the RC oscillator frequency. Refer to *Section 6.1.1: Internal RC oscillator on page 25*.

Bits 4:3 = Reserved, must be kept cleared

Bit 2 = **LVDRF** *LVD reset flag*

This bit indicates that the last Reset was generated by the LVD block. It is set by hardware (LVD reset) and cleared by software (by reading). When the LVD is disabled by option byte, the LVDRF bit value is undefined.

The LVDRF flag is not cleared when another RESET type occurs (external or watchdog), the LVDRF flag remains set to keep trace of the original failure.
In this case, a watchdog reset can be detected by software while an external reset can not.

Bits 1:0 = Reserved, must be kept cleared

6.5.5 Prescaler register (PSCR)

Reset value: 0000 0011 (03h)

7							0
CK2	CK1	CK0	0	0	0	1	1
Read/write							

Bits 7:5 = **CK[2:0]** internal RC Prescaler Selection

These bits are set by software and cleared by hardware after a reset. These bits select the prescaler of the internal RC oscillator. See *Figure 10: ST7FOXA0 clock management block diagram on page 29* and *Table 8*.

If the internal RC is used with a supply operating range below 3.3 V, a division ratio of at least 2 must be enabled in the RC prescaler.

Table 8. Internal RC prescaler selection bits

CK2	CK1	CK0	f_{OSC}
0	0	1	$f_{RC/2}$
0	1	0	$f_{RC/4}$
0	1	1	$f_{RC/8}$
1	0	0	$f_{RC/16}$
others			f_{RC}

Bits 4:0 = Reserved, must be kept at their reset value.

6.5.6 Clock controller control/status register (CKCNTCSR)

Reset value: 0000 1001 (09h)

7							0
0	0	0	0	AWU_FLAG	RC_FLAG	0	RC/AWU
Read/write							

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

Bit 3 = **AWU_FLAG** *AWU Selection bit*

This bit is set and cleared by hardware.

0: No switch from AWU to RC requested

1: AWU clock activated and temporization completed

Bit 2 = **RC_FLAG** *RC Selection bit*

This bit is set and cleared by hardware.

0: No switch from RC to AWU requested

1: RC clock activated and temporization completed

Bit 1 = Reserved, must be kept cleared.

Bit 0 = **RC/AWU** *RC/AWU Selection bit*
0: RC enabled
1: AWU enabled (default value)

Table 9. Clock register mapping and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0030h	RCC_CSR	- 0	- 0	- 0	- 0	- 0	- 0	RCCLAT 0	RCCPGM 0
003Ah	MCCSR Reset Value	- 0	- 0	- 0	- 0	- 0	- 0	MCO 0	SMS 0
003Bh	RCCRH Reset Value	CR9 1	CR8 1	CR7 1	CR6 1	CR5 1	CR4 1	CR3 1	CR2 1
003Ch	RCCRL Reset Value	- 0	CR1 1	CR0 1	- 0	- 0	LVDRF x	- 0	- 0
003Dh	PSCR Reset Value	CK2 0	CK1 0	CK0 0	- 0	- 0	- 0	- 1	- 1
0051h	CKCNTCSR Reset Value	- 0	- 0	- 0	- 0	AWU_ FLAG 1	RC_FLA G 0	- 0	RC/AWU 1

7 Power saving modes

7.1 Introduction

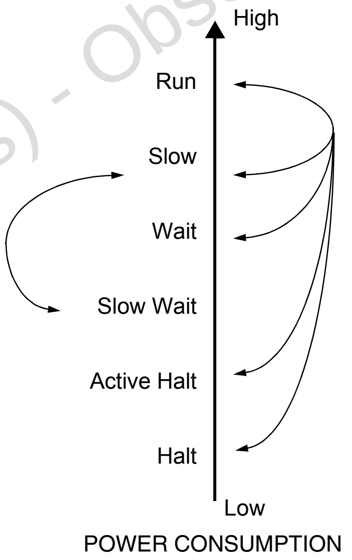
To give a large measure of flexibility to the application in terms of power consumption, four main power saving modes are implemented in the ST7 (see *Figure 16*):

- Slow
- Wait (and Slow-Wait)
- Active Halt
- Auto wakeup From Halt (AWUFH)
- Halt

After a reset the normal operating mode is selected by default (Run mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency (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



7.2 Slow mode

This mode has two targets:

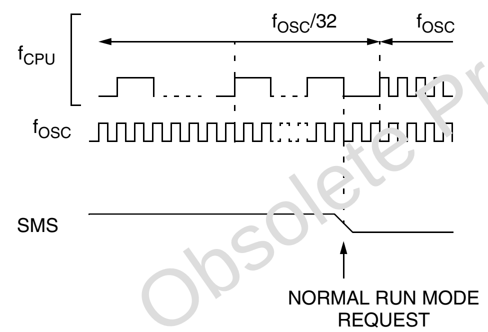
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

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

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

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

Figure 17. Slow mode clock transition



7.3 Wait mode

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

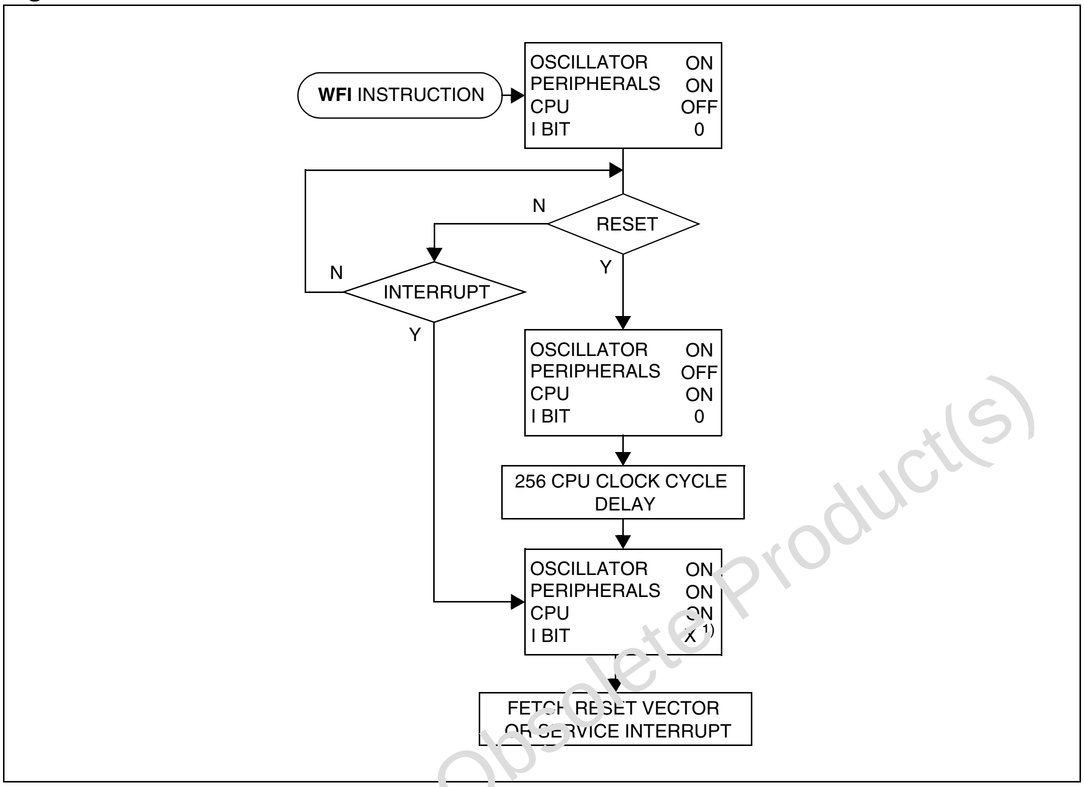
This power saving mode is selected by calling the 'WFI' instruction.

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* for a description of the Wait mode flowchart.

Figure 18. Wait mode flowchart



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

7.4 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:

Table 10. Enabling/disabling active-halt and halt modes

LTCSR TBIE bit	ATCSR OVFIE bit	ATCSRCK1 bit	ATCSRCK0 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	

7.4.1 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.

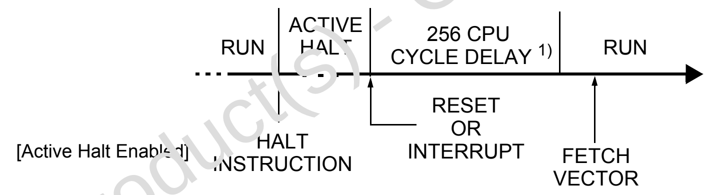
- When exiting Active-Halt mode by means of a Reset, a 256 CPU cycle delay occurs. After the start up delay, the CPU resumes operation by fetching the Reset vector which woke it up (see *Figure 20*).
- When exiting Active-Halt mode by means of an interrupt, the CPU immediately resumes operation by servicing 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 wakeup 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).

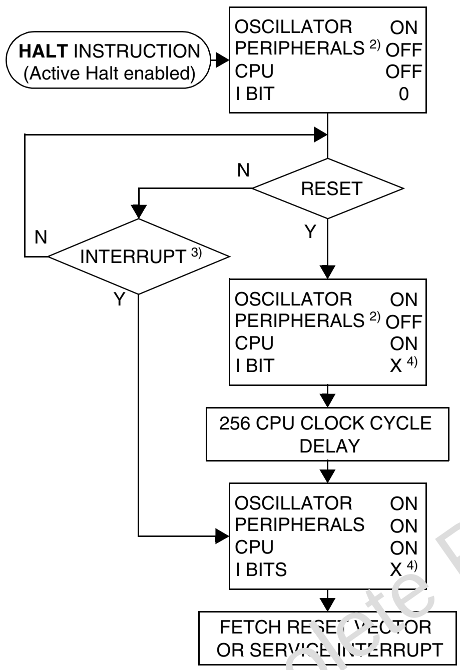
Caution: As soon as Active-Halt is enabled, executing a HALT instruction while the Watchdog is active does not generate a Reset if the WDGHALT bit is reset.
This means that the device cannot spend more than a defined delay in this power saving mode.

Figure 19. Active-halt timing overview



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

Figure 20. Active-halt mode flowchart



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.

7.4.2 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.

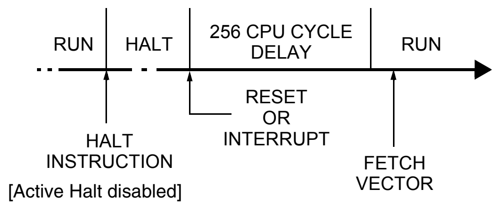
The MCU can exit Halt mode on reception of either a specific interrupt (see *Table :*) or a Reset. When exiting Halt mode by means of a Reset or an interrupt, the main oscillator is immediately turned on and the 256 CPU cycle delay is used to stabilize it. After the start up delay, the CPU resumes operation 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 be stopped, including the operation of the on-chip peripherals. All peripherals are not clocked except the ones which get their clock supply from another clock generator (such as an external or auxiliary oscillator).

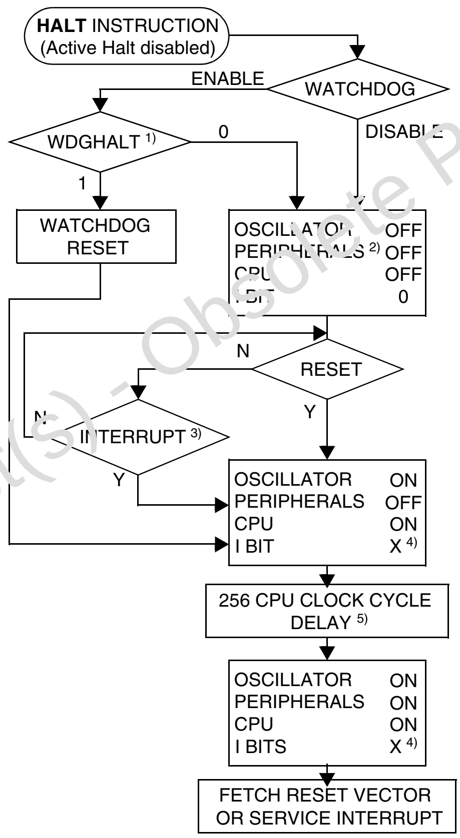
The compatibility of Watchdog operation with Halt mode is configured by the “WDGHALT” option bit of the option byte. The HALT instruction when executed while the Watchdog system is enabled, can generate a Watchdog Reset (see *Section 13.1: Option bytes* for more details).

Figure 21. Halt timing overview



1. A reset pulse of at least 42 µs must be applied when exiting from Halt mode.

Figure 22. Halt mode flowchart



1. WDGHALT is an option bit. See option byte section for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to *Table :* 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. The CPU clock must be switched to 1 MHz (RC/8) or AWU RC before entering Halt mode.

Halt mode recommendations

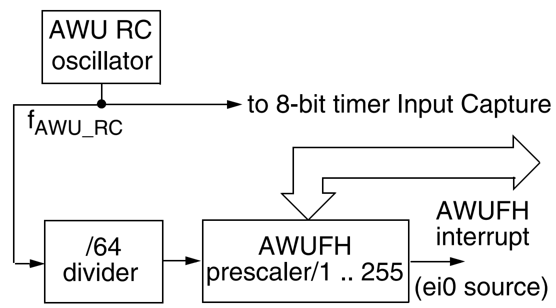
- Make sure that an external event is available to wake up the microcontroller from Halt mode.
- When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as “Input Pull-up with Interrupt” before executing the HALT instruction. The main reason for this is that 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 wakeup event (reset or external interrupt).

7.5 Auto wakeup from halt mode

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

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

Figure 23. 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 following actions are performed:

- the AWUF flag is set by hardware,
- an interrupt wakes-up the MCU from Halt mode,
- the main oscillator is immediately turned on and the 256 CPU 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 8-bit Lite timer, allowing the f_{AWU_RC} to be measured using the main oscillator clock as a reference timebase.

Similarities with halt mode

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

- The MCU can exit AWUFH mode by means of any interrupt with exit from Halt capability or a reset (see *Section 7.4: Active-halt and halt modes*).
- When entering AWUFH mode, the I bit in the CC register is forced to 0 to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.
- In AWUFH mode, the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. None of the peripherals are clocked except those which get their clock supply from another clock generator (such as an external or auxiliary oscillator like the AWU oscillator).
- The compatibility of watchdog operation with AWUFH mode is configured by the WDGHALT option bit in the option byte. Depending on this setting, the HALT instruction when executed while the watchdog system is enabled, can generate a watchdog Reset.

Figure 24. AWUF halt timing diagram

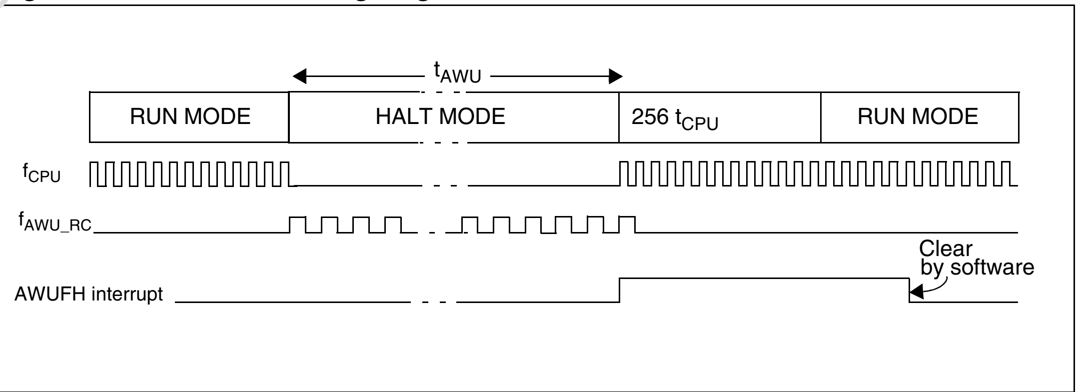
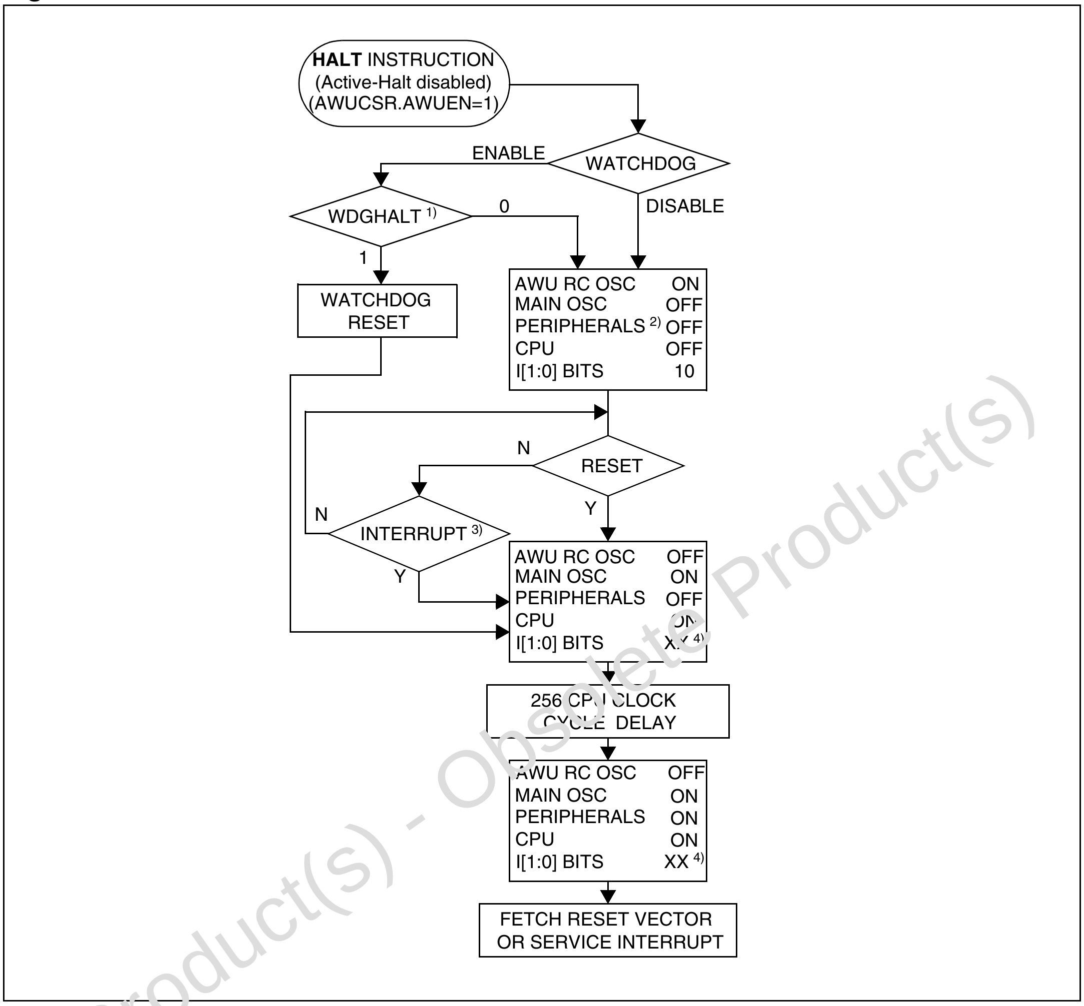


Figure 25. AWUFH mode flowchart



1. WDGHALT is an option bit. See option byte section for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only an AWUFH interrupt and some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to *Table :* for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped.

7.5.1 Register description

7.5.2 AWUFH Control/Status Register (AWUCSR)

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	0	AWU F	AWUM	AWUEN
Read/Write							

Bits 7:3 = Reserved

Bit 2 = **AWUF** *Auto wakeup flag*

This bit is set by hardware when the AWU module generates an interrupt and cleared by software on reading AWUCSR. Writing to this bit does not change its value.

0: No AWU interrupt occurred

1: AWU interrupt occurred

Bit 1 = **AWUM** *Auto wakeup Measurement bit*

This bit enables the AWU RC oscillator and connects its output to the Input Capture of the 8-bit Lite timer. This allows the timer to be used to measure the AWU RC oscillator dispersion and then compensate this dispersion by providing the right value in the AWUPRE register.

0: Measurement disabled

1: Measurement enabled

Bit 0 = **AWUEN** *Auto wakeup From Halt Enabled bit*

This bit enables the Auto wakeup from halt feature: once Halt mode is entered, the AWUFH wakes up the microcontroller after a time delay dependent on the AWU prescaler value. It is set and cleared by software.

0: AWUFH (Auto wakeup from Halt) mode disabled

1: AWUFH (Auto wakeup from Halt) mode enabled

Note: Whatever the clock source, this bit should be set to enable the AWUFH mode once the HALT instruction has been executed.

7.5.3 AWUFH Prescaler Register (AWUPR)

Reset value: 1111 1111 (FFh)

7							0
AWUPR7	AWUPR6	AWUPR5	AWUPR4	AWUPR3	AWUPR2	AWUPR1	AWUPR0
Read/Write							

Bits 7:0= **AWUPR[7:0]** *Auto wakeup Prescaler*

These 8 bits define the AWUPR Dividing factor (see *Table 11*).

Table 11. Configuring the dividing factor

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

In AWU mode, the time during which the MCU stays in Halt mode, t_{AWU}, is given by the equation below. See also *Figure 24 on page 49*.

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

The AWUPR prescaler register can be programmed to modify the time during which the MCU stays in Halt mode before waking up automatically.

Note: If 00h is written to AWUPR, the AWUPR remains unchanged.

Table 12. AWU register mapping and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0048h	**AWUCSR** Reset Value	0	0	0	0	0	AWUF	AWUM	AWUEN
0049h	**AWUPR** Reset Value	AWUPR7 1	AWUPR6 1	AWUPR5 1	AWUPR4 1	AWUPR3 1	AWUPR2 1	AWUPR1 1	AWUPR0 1

8 I/O ports

8.1 Introduction

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

8.2 Functional description

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

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

Figure 26 shows the generic I/O block diagram.

8.2.1 Input modes

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

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

Note: 1 *Writing to the DR modifies the latch value but does not change the state of the input pin.*

2 *Do not use read/modify/write instructions (BSET/BRES) to modify the DR register.*

External interrupt function

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

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

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

External interrupts are hardware interrupts. Fetching the corresponding interrupt vector automatically clears the request latch. Changing the sensitivity 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 reenable them after the complete previous sequence in order to avoid an external interrupt occurring on the unwanted edge.

This corresponds to the following steps:

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

b) Select rising edge

c) Enable the external interrupt through the OR register

d) Select the desired sensitivity if different from rising edge

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

2. To disable an external interrupt:

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

b) Select falling edge

c) Disable the external interrupt through the OR register

d) Select rising edge

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

8.2.2 Output modes

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

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

Table 13. DR Value and output pin status

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

8.2.3 Alternate functions

Many ST7s I/Os have one or more alternate functions. These may include output signals from, or input signals to, on-chip peripherals. *Table 2* describes which peripheral signals can be input/output to which ports.

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

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

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

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

Figure 26. I/O port general block diagram

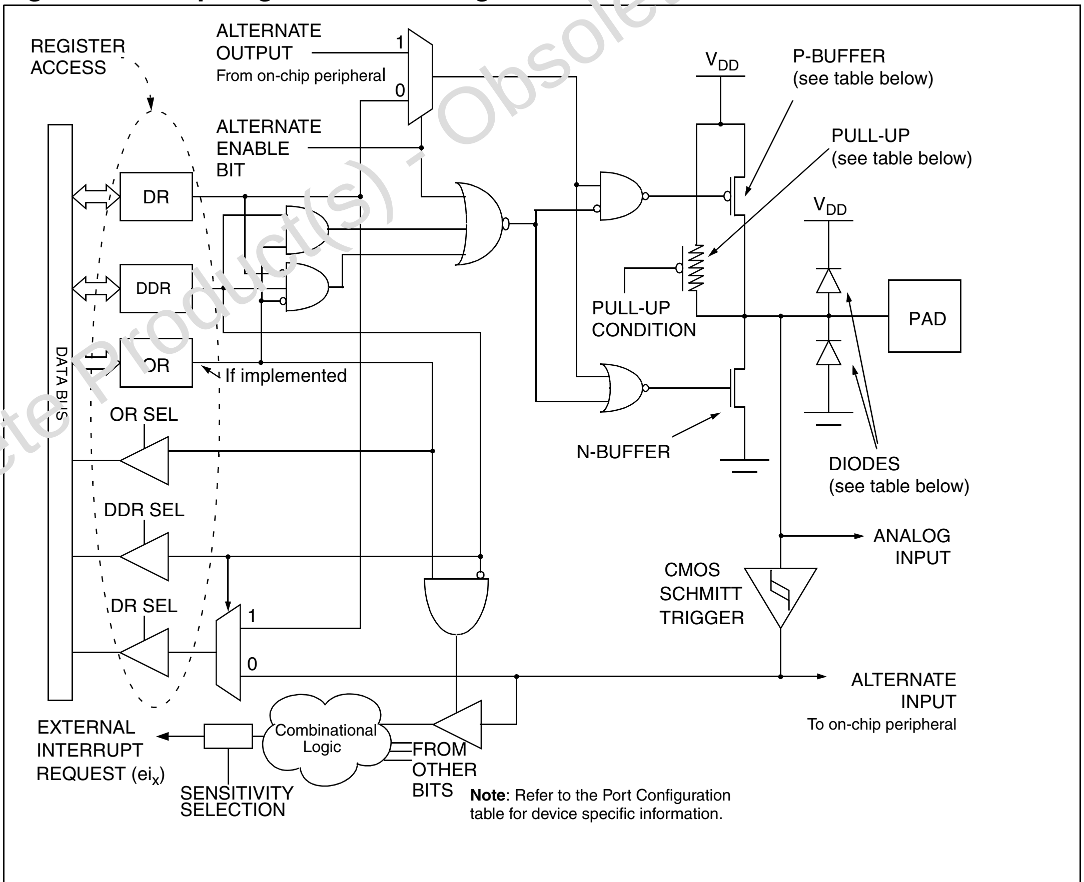
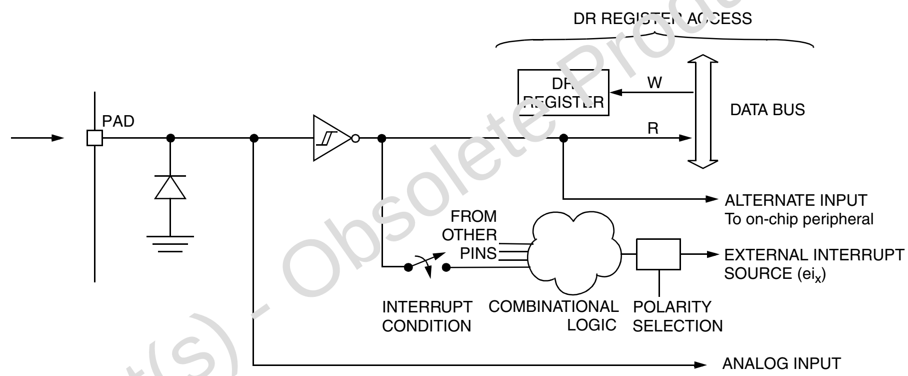
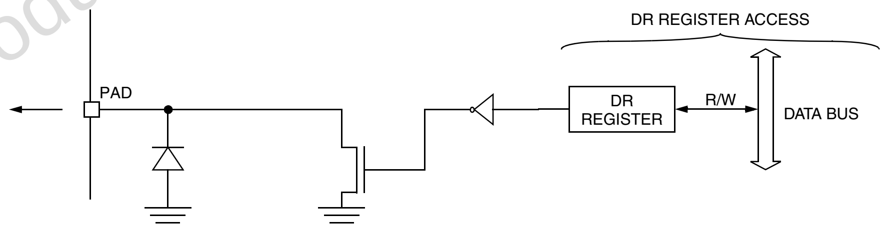
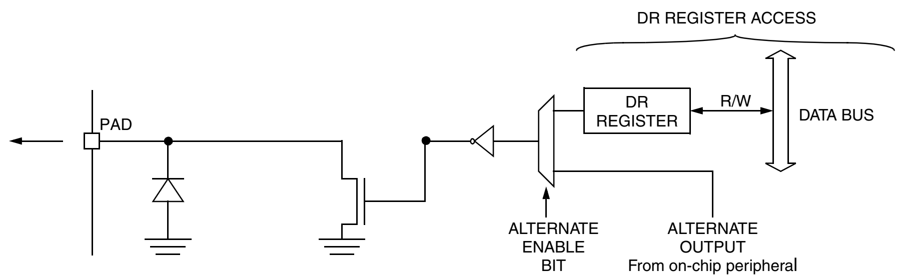


Table 14. I/O port mode options [1]

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 Interrupt	On			
Output	Push-pull	Off	On		
	Open Drain (logic level)		Off		

1. Off means implemented not activated, On means implemented and activated.

Table 15. ST7FOXA0 I/O port configuration

	Hardware configuration
INPUT [1]	
OPEN-DRAIN OUTPUT[2]	
PUSH-PULL OUTPUT[2]	

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.

8.2.4 Analog alternate function

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

Analog Recommendations

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

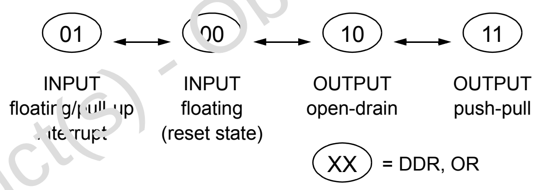
Caution: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

8.3 I/O port implementation

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

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in *Figure 27*. Other transitions are potentially risky and should be avoided, since they may present unwanted side-effects such as spurious interrupt generation.

Figure 27. Interrupt I/O port state transitions



8.4 Unused I/O pins

Unused I/O pins must be connected to fixed voltage levels. Refer to *Section 12.8: I/O port pin characteristics*.

8.5 Low power modes

Table 16. Effect of low power modes on I/O ports

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

8.6 Interrupts

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

Table 17. Description of interrupt events

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

See application notes AN1045 software implementation of I^2C bus master, and AN1048 - software LCD driver

8.7 Device-specific I/O port configuration

The I/O port register configurations are summarized in *Table 18*.

Table 18. Port configuration

Port	Pin name	Input (DDR=0)		Output (DDR=1)	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA0:2, PA4:5 (1)	floating	pull-up interrupt (1)	open drain	push-pull
	PA3 (2)	-	-	open drain	push-pull

1. IS4[1:0] = 01 is the only safe configuration to avoid spurious interrupt in HALT and AWUFH modes. Refer to *11.3.2: External Interrupt Control Register 2 (EICR2) on page 91*.
2. After reset, to configure PA3 as a general purpose output, the application has to program the MUXCR0 and MUXCR1 registers. See *Section 6.3.6: Multiplexed IO reset control register 1 (MUXCR1) on page 34* and *Section 6.3.7: Multiplexed IO reset control register 0 (MUXCR0) on page 34*

Table 19. 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	1	0	0	LSB 0
0002h	PAOR Reset Value	MSB 0	0	0	0	0	0	1	LSB 0

9 On-chip peripherals

9.1 Lite Timer (LT)

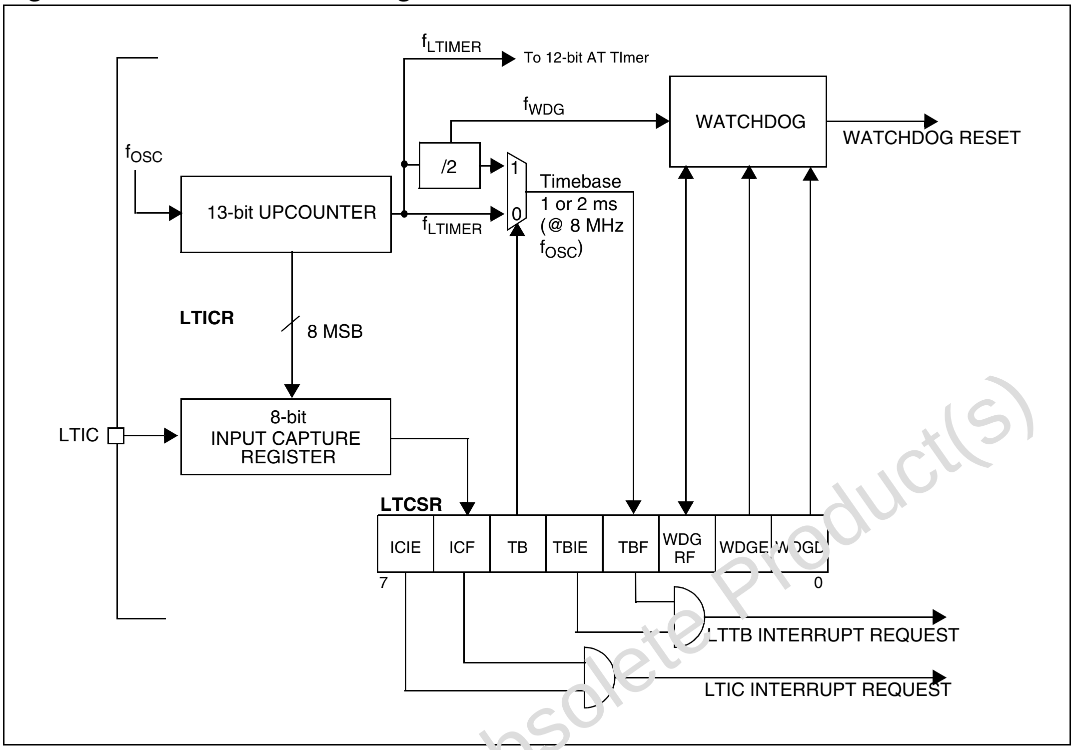
9.1.1 Introduction

The Lite Timer can be used for general-purpose timing functions. It is based on a free-running 13-bit upcounter with two software-selectable timebase periods, an 8-bit input capture register and watchdog function.

9.1.2 Main features

- Real-time Clock
  - 13-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 28. Lite timer block diagram



9.1.3 Functional description

The value of the 13-bit counter cannot be read or written by software. After an MCU reset, it starts incrementing from 0 at a frequency of f_{OSC}. A counter overflow event occurs when the counter rolls over from 1F3Fh 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.

9.1.4 Watchdog

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

To prevent this watchdog reset occurring, 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 29*.

If the watchdog is not enabled immediately after reset, the first watchdog timeout will be shorter than 2 ms, because this period is counted starting from reset. Moreover, if a 2 ms 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.

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: Once the WDGRF bit is set, if the watchdog is enabled, the microcontoller is immediatly reset, even if the WDGD bit is set by software.

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)

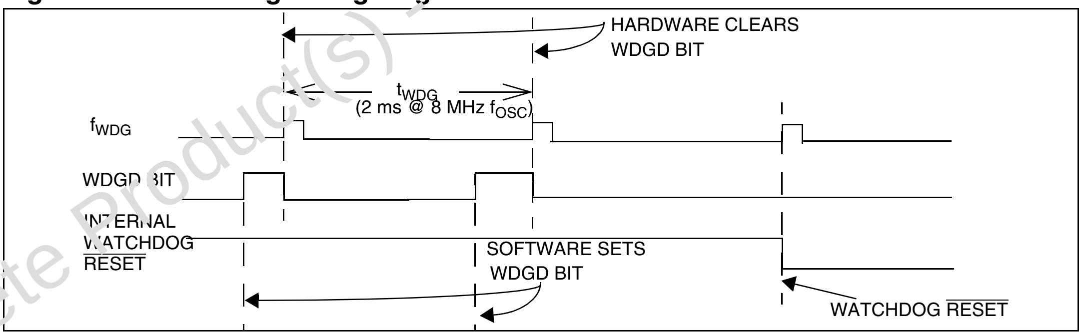
If the Watchdog reset on HALT option is not selected by option byte, the Halt mode can be used when the watchdog is enabled.

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 or 512 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.

Figure 29. Watchdog timing diagram



9.1.5 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 MSB 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.

An overflow can be detected through the timebase event. This overflow occurs when the counter rolls over from 1F3Fh to 00h, that is, from F9h to 00h if only the 8 MSB of the LTIC counter are taken into account. In this case, the TB bit in the LTCSR register must be reset to detect all overflows.

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.

9.1.6 Low power modes

Table 20. Effect on Lite timer

Mode	Description
Wait	No effect on Lite timer
Active-halt	No effect on Lite timer
Halt	Lite timer stops counting

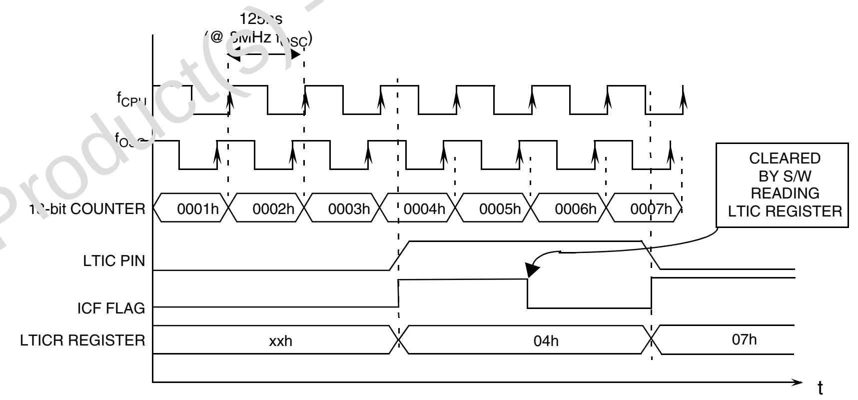
9.1.7 Interrupts

Table 21. Interrupt events

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	Yes	No	No

Note: The TBF and ICF interrupt events are connected to separate interrupt vectors (see Interrupts chapter). They generate an interrupt if the enable bit is set in the LTCSR register and the interrupt mask in the CC register is reset (RIM instruction).

Figure 30. Input capture timing diagram



9.1.8 Register description

Lite Timer Control/Status Register (LTCSR)

Reset Value: 0000 0x00 (0xh)

7							0
ICIE	ICF	TB	TBIE	TBF	WDGRF	WDGE	WDGD
Read/Write							

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 initialize 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 (1 ms @ 8 MHz)

1: Timebase period = t_{OSC} * 16000 (2 ms @ 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. 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)

Reset Value: 0000 0000 (00h)

7							0
ICR7	ICR6	ICR5	ICR4	ICR3	ICR2	ICR1	ICR0
Read only							

Bits 7:0 = **ICR[7:0]** *Input Capture Value*

These bits are read by software and cleared by hardware after a reset. If the ICF bit in the LTCSR is cleared, the value of the 8-bit up-counter will be captured when a rising or falling edge occurs on the LTIC pin.

Table 22. 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 0	TB 0	TBIE 0	TBF 0	WDGRF x	WDGE 0	WDGD 0
0C	**LTICR** Reset Value	ICR7 0	ICR6 0	ICR5 0	ICR4 0	ICR3 0	ICR2 0	ICR1 0	ICR0 0

9.2 12-bit Autoreload Timer (AT)

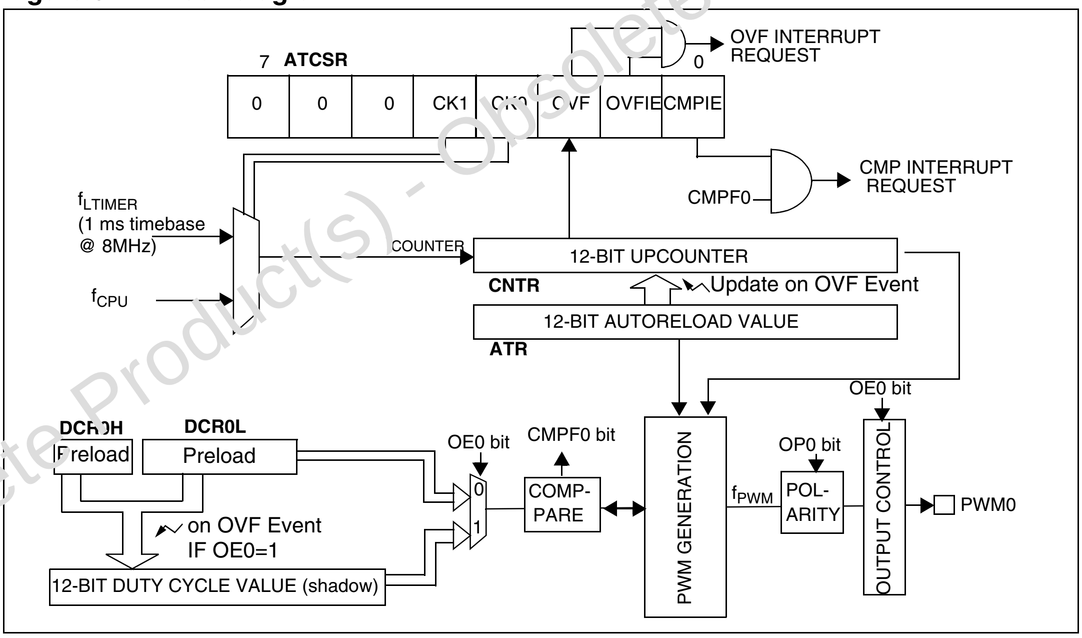
9.2.1 Introduction

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.

9.2.2 Main Features

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

Figure 31. Block diagram



9.2.3 Functional description

PWM Mode

This mode allows a Pulse Width Modulated signals to be 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 71).

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, in order not to miss a compare event and to have the right signal applied on the PWM output.

Figure 32. PWM function

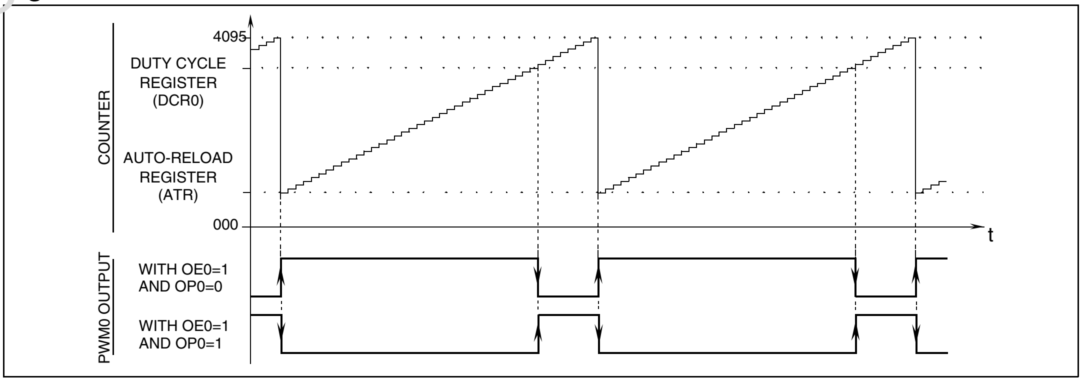
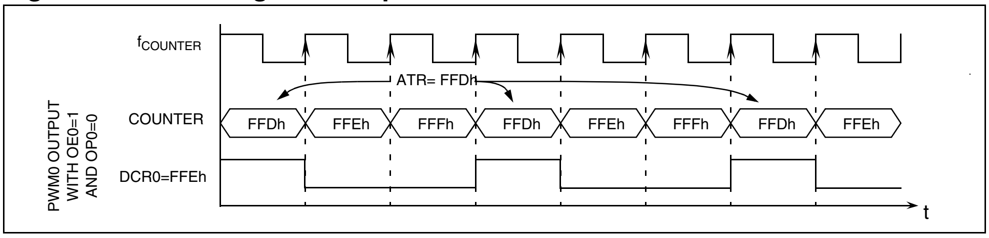


Figure 33. PWM Signal example



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 will be 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.

9.2.4 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

9.2.5 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

Note: 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

9.2.6 Register description

TImer Control Status Register (ATCSR)

Reset Value: 0000 0000 (00h)

7							0
0	0	0	CK1	CK0	OVF	OVFIE	CMPIE
Read/Write							

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

Bits 4:3 = **CK[1:0]** *Counter Clock Selection.*

These bits are set and cleared by software and cleared by hardware after a reset. They select the clock frequency of the counter.

Table 23. Counter clock selection

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) after it has been cleared by software.

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)

Reset Value: 0000 0000 (00h)

15							8
0	0	0	0	CN11	CN10	CN9	CN8
Read only							

Counter register low (CNTRL)

Reset value: 0000 0000 (00h)

7							0
CN7	CN6	CN5	CN4	CN3	CN2	CN1	CN0
Read only							

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. As there is no latch, it is recommended to read LSB first. In this case, CNTRH can be incremented between the two read operations and to have an accurate result when f_{timer}=f_{CPU}, special care must be taken when CNTRL values close to FFh are read.
When a counter overflow occurs, the counter restarts from the value specified in the ATR register.

Auto reload register high (ATRH)

Reset value: 0000 0000 (00h)

15							8
0	0	0	0	ATR11	ATR10	ATR9	ATR8
Read/Write							

Auto reload register low (ATRL)

Reset value: 0000 0000 (00h)

7							0
ATR7	ATR6	ATR5	ATR4	ATR3	ATR2	ATR1	ATR0
Read/Write							

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)

Reset value: 0000 0000 (00h)

15							8
0	0	0	0	DCR11	DCR10	DCR9	DCR8
Read/Write							

PWM0 duty cycle register low (DCR0L)

Reset value: 0000 0000 (00h)

7							0
DCR7	DCR6	DCR5	DCR4	DCR3	DCR2	DCR1	DCR0
Read/Write							

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 32*). 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)

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	OP0	CMPF0
Read/Write							

Bits 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.

PWM output control register (PWMCR)

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	OE0
Read/Write							

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 24. 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

9.3 10-bit A/D converter (ADC)

9.3.1 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 5 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 5 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.

9.3.2 Main features

- 10-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)

The block diagram is shown in *Figure 34*.

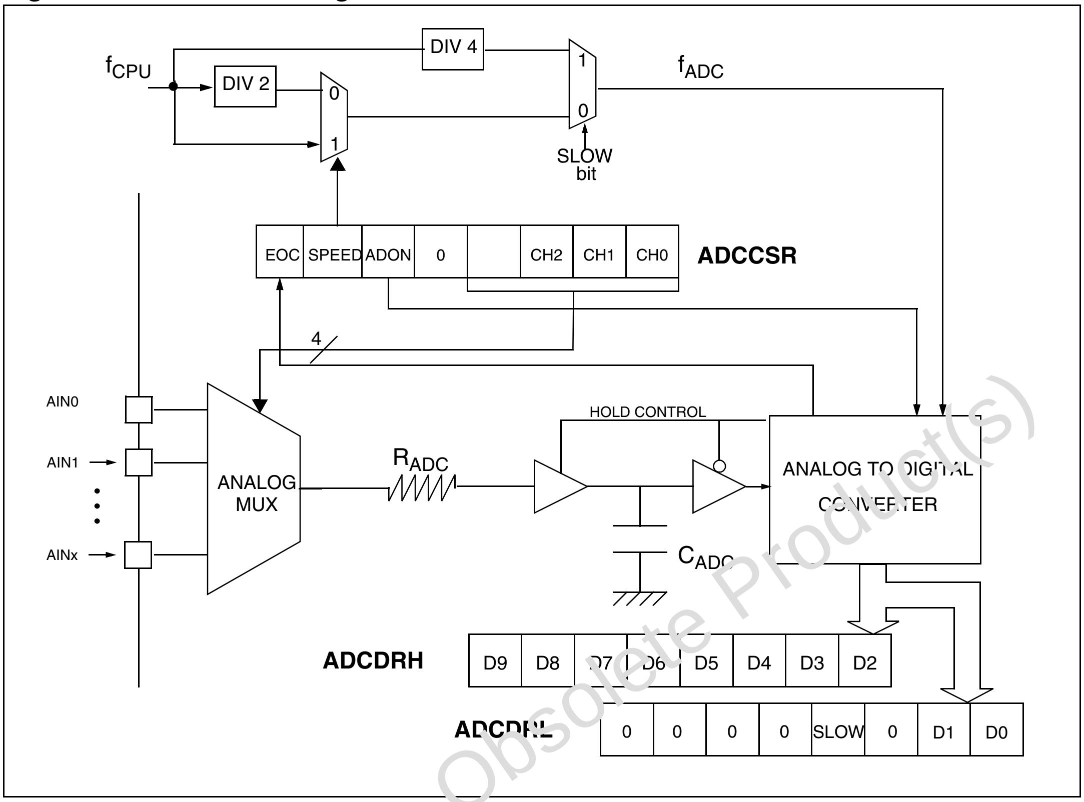
9.3.3 Functional description

Analog power supply

V_{DDA} and V_{SSA} are the high and low level reference voltage pins. In some devices (refer to 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 34. ADC block diagram



Digital A/D conversion result

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

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

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

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

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

Configuring the A/D conversion

The analog input ports must be configured as input, no pull-up, no interrupt (see *Section 8: I/O ports*). Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

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

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

When a conversion is complete:

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

A read to the ADCDRH or a write to any bit of the ADCCSR register resets the EOC bit.

To read the 10 bits, perform the following steps:

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

To read only 8 bits, perform the following steps:

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

Changing the conversion channel

The application can change channels during conversion. When software modifies the CH[2:0] bits in the ADCCSR register, the current conversion is stopped, the EOC bit is cleared, and the A/D converter starts converting the newly selected channel.

9.3.4 Low power modes

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

Table 25. Effect of low power modes on the A/D converter

Mode	Description
Wait	No effect on A/D Converter
Halt	A/D Converter disabled. After wakeup from Halt mode, the A/D Converter requires a stabilization time t_{STAB} (see Electrical Characteristics) before accurate conversions can be performed.

9.3.5 Interrupts

None.

9.3.6 Register description

Control/status register (ADCCSR)

Reset value: 0000 0000 (00h)

7							0
EOC	SPEED	ADON	0	0	CH2	CH1	CH0
Read only	Read/write						

Bit 7 = **EOC** *End of Conversion bit*

This bit is set by hardware. It is cleared by hardware when software reads the ADCDRH register or writes to any bit of the ADCCSR register.

0: Conversion is not complete

1: Conversion complete

Bit 6 = **SPEED** *ADC clock selection bit*

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 (ADCDRL register).

Bit 5 = **ADON** *A/D Converter on bit*

This bit is set and cleared by software.

0: A/D converter is switched off

1: A/D converter is switched on

Bits 4:3 = Reserved, must be kept cleared.

Bits 2:0 = **CH[2:0]** *Channel Selection*

These bits select the analog input to convert. They are set and cleared by software.

Table 26. Channel selection using CH[2:0]

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

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

Data register High (ADCDRH)

Reset value: xxxx xxxx (xxh)

7							0
D9	D8	D7	D6	D5	D4	D3	D2
Read only							

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

ADC Control/data register Low (ADCDRL)

Reset value: 0000 00xx (0xh)

7							0
0	0	0	0	SLOW	0	D1	D0
Read/write							

Bits 7:4 = Reserved. Forced by hardware to 0.

Bit 3 = **SLOW** *Slow mode bit*

This bit is set and cleared by software. It is used together with the SPEED bit in the ADCCSR register to configure the ADC clock speed as shown on the table below.

Table 27. Configuring the ADC clock speed

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

1. The maximum allowed value of f_{ADC} is 4 MHz (see *Section 12.10 on page 108*)

Bits 1:0 = **D[1:0]** *LSB of Analog Converted value*

Table 28. ADC register mapping and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0036h	**ADCCSR** Reset Value	EOC 0	SPEED 0	ADON 0	0 0	0 0	CH2 0	CH1 0	CH0 0
0037h	**ADCDRH** Reset Value	D9 x	D8 x	D7 x	D6 x	D5 x	D4 x	D3 x	D2 x
0038h	**ADCDRL** Reset Value	0 0	0 0	0 0	0	SLOW 0	0	D1 x	D0 x

10 Instruction set

10.1 ST7 addressing modes

The ST7 core features 17 different addressing modes which can be classified in seven main groups:

Table 29. Description of addressing modes

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 30. ST7 addressing mode overview

Mode			Syntax	Destination/ source	Pointer address	Pointer size	Length (bytes)
Inherent			nop				+ 0
Immediate			ld A,#$55				+ 1
Short	Direct		ld A,$10	00..FF			+ 1
Long	Direct		ld A,$1000	0000..FFFF			+ 2
No Offset	Direct	Indexed	ld A,(X)	00..FF			+ 0 (with X register) + 1 (with Y register)
Short	Direct	Indexed	ld A,($10,X)	00..1FE			+ 1
Long	Direct	Indexed	ld A,($1000,X)	0000..FFFF			+ 2
Short	Indirect		ld A,[$10]	00..FF	00..FF	byte	+ 2
Long	Indirect		ld A,[$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed	ld A,([$10],X)	00..1FE	00..FF	byte	+ 2

Table 30. ST7 addressing mode overview (continued)

Mode			Syntax	Destination/ source	Pointer address	Pointer size	Length (bytes)
Long	Indirect	Indexed	ld A,([$10.w],X)	0000..FFFF	00..FF	word	+ 2
Relative	Direct		jrne loop	PC-128/PC+127(1)			+ 1
Relative	Indirect		jrne [$10]	PC-128/PC+127(1)	00..FF	byte	+ 2
Bit	Direct		bset $10,#7	00..FF			+ 1
Bit	Indirect		bset [$10],#7	00..FF	00..FF	byte	+ 2
Bit	Direct	Relative	btjt $10,#7,skip	00..FF			+ 2
Bit	Indirect	Relative	btjt [$10],#7,skip	00..FF	00..FF	byte	+ 3

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

10.1.1 Inherent mode

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

Table 31. Instructions supporting inherent addressing mode

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

Table 31. Instructions supporting inherent addressing mode (continued)

Instruction	Function
MUL	Byte multiplication
SLL, SRL, SRA, RLC, RRC	Shift and rotate operations
SWAP	Swap nibbles

10.1.2 Immediate mode

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

Table 32. Instructions supporting inherent immediate addressing mode

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

10.1.3 Direct modes

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

The direct addressing mode consists of two submodes:

Direct (Short) addressing mode

The address is a byte, thus requires only 1 byte after the opcode, but only allows 00 - FF addressing space.

Direct (Long) addressing mode

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

10.1.4 Indexed modes (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 mode (No Offset)

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

Indexed mode (Short)

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

Indexed mode (Long)

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

10.1.5 Indirect modes (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 mode (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 mode (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.

10.1.6 Indirect indexed modes (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 mode (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 mode (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 33. Instructions supporting direct, indexed, indirect and indirect indexed addressing modes

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

Table 33. Instructions supporting direct, indexed, indirect and indirect indexed addressing modes (continued)

Instructions	Function
Short instructions only	
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

10.1.7 Relative modes (direct, indirect)

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

Table 34. Instructions supporting relative modes

Available Relative Direct/Indirect instructions	Function
JRxx	Conditional jump
CALLR	Call relative

The relative addressing mode consists of two submodes:

Relative mode (Direct)

The offset follows the opcode.

Relative mode (Indirect)

The offset is defined in memory, of which the address follows the opcode.

10.2 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:

Table 35. ST7 instruction set

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 rotate	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 by:

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.

10.2.1 Illegal opcode reset

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

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

Table 36. Illegal opcode detection

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 ?								

Table 36. Illegal opcode detection (continued)

Mnemo	Description	Function/Example	Dst	Src		H	I	N	Z	C
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 >								
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			

Table 36. Illegal opcode detection (continued)

Mnemo	Description	Function/Example	Dst	Src		H	I	N	Z	C
WFI	Wait for Interrupt						0			
XOR	Exclusive OR	A = A XOR M	A	M				N	Z	

11 Interrupts

The ST7 core may be interrupted by one of two different methods: Maskable hardware interrupts as listed in the “interrupt mapping” table and a non-maskable software interrupt (TRAP). The Interrupt processing flowchart is shown in *Figure 35*.

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).

11.1 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 35*.

11.2 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.

11.3 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.

Figure 35. Interrupt processing flowchart

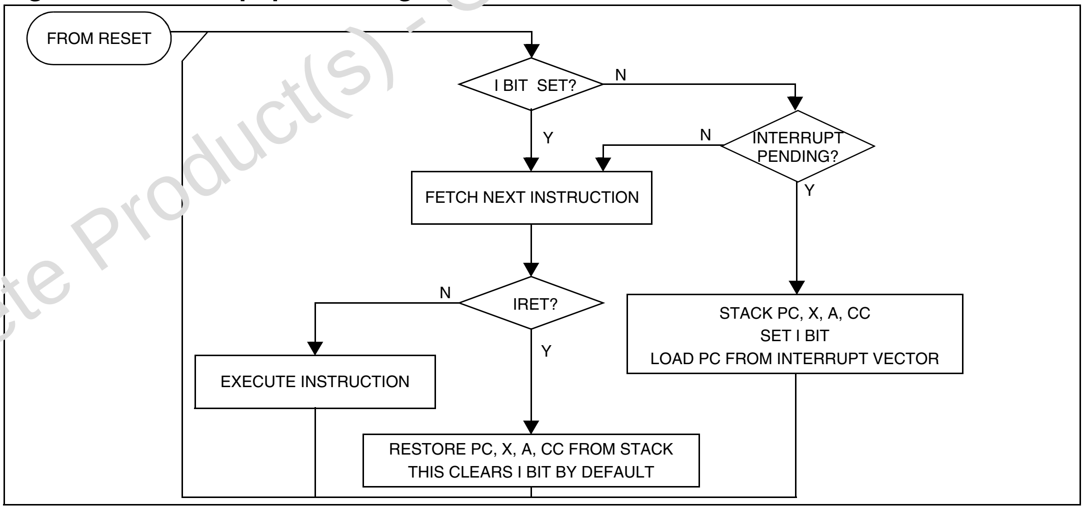


Table 37. ST7FOXA0 interrupt mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Address Vector
	RESET	Reset	N/A		yes	FFFEh-FFFFh
	TRAP	Software Interrupt	N/A		no	FFFCh-FFFDh
0	AWU	Auto wakeup Interrupt	AWUCSR		yes (1)	FFFAh-FFFBh
1	ei0	External Interrupt 0	N/A	Highest Priority	yes	FFF8h-FFF9h
2	ei1	External Interrupt 1	N/A		yes	FFF6h-FFF7h
3 (2)	ei2 (2)	External Interrupt 2 (2)	N/A		yes	FFF4h-FFF5h
4		Not used	N/A		no	FFF2h-FFF3h
5	ei3	External Interrupt 3	N/A		yes	FFF0h-FFF1h
6 (3)	ei4 (3)	External Interrupt 4 (3)	N/A		no (3)	FFEEh-FFEFh
7		Not used	N/A		no	FFECh-FFEDh
8	AT TIMER	AT TIMER Output Compare Interrupt	PWMxCSR or ATCSR		no	FFEAh-FFEBh
9	AT TIMER	AT TIMER Overflow Interrupt	ATCSR		yes (4)	FFE8h-FFE9h
10	LITE TIMER	LITE TIMER Input Capture Interrupt	LTCSR	Lowest Priority	no	FFE6h-FFE7h
11	LITE TIMER	LITE TIMER RTC1 Interrupt	LTCSR		yes (4)	FFE4h-FFE5h
12		Not used			no	FFE2h-FFE3h
13		Not used			no	FFE0h-FFE1h

1. This interrupt exits the MCU from “Auto wakeup from HALT” mode only.
2. Whatever the sensitivity configuration, this interrupt cannot exit the MCU from HALT, ACTIVE-HALT and AWUFH modes when a falling edge occurs.
3. This interrupt exits the MCU from “WAIT” and “ACTIVE-HALT” modes only. Moreover IS4[1:0] =01 is the only safe configuration to avoid spurious interrupt in Halt and AWUFH modes.
4. These interrupts exit the MCU from “ACTIVE-HALT” mode only.

11.3.1 External Interrupt Control Register 1 (EICR1)

Reset value: 0000 0000 (00h)

7							0
0	0	IS21	IS20	IS11	IS10	IS01	IS00
Read/write							

Bits 7:6 = Reserved, must be kept cleared.

Bits 5:4 = **IS2[1:0]** *ei2 sensitivity bits*

These bits define the interrupt sensitivity for ei2 according to *Table ?*.

Bits 3:2 = **IS1[1:0]** *ei1 sensitivity bits*

These bits define the interrupt sensitivity for ei1 according to *Table ?*.

Bits 1:0 = **IS0[1:0]** *ei0 sensitivity bits*

These bits define the interrupt sensitivity for ei0 according to *Table ?*.

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

2 *Changing the sensitivity of a particular external interrupt clears this pending interrupt. This can be used to clear unwanted pending interrupts. Refer to Section : External interrupt function.*

3 *Whatever the sensitivity configuration, ei2 cannot exit the MCU from HALT, ACTIVE-HALt and AWUFH modes when a falling edge occurs.*

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

11.3.2 External Interrupt Control Register 2 (EICR2)

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	IS41	IS40	IS31	IS30
Read/write							

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

Bits 3:2 = **IS4[1:0]** *ei4 sensitivity bits*

These bits define the interrupt sensitivity for ei1 according to *Table ?*.

Bits 1:0 = **IS0[1:0]** *ei3 sensitivity bits*

These bits define the interrupt sensitivity for ei0 according to *Table ?*.

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

2 Changing the sensitivity of a particular external interrupt clears this pending interrupt. This can be used to clear unwanted pending interrupts. Refer to Section : External interrupt function.

3 IS4[1:0] = 01 is the only safe configuration to avoid spurious interrupt in Halt and AWUFH modes.

Table 38. Interrupt register mapping and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0037h	EICR1 Reset Value	- 0	- 0	IS21 0	IS20 0	IS11 0	IS10 0	IS01 0	IS00 0
003Dh	EICR2 Reset Value	- 0	- 0	- 0	- 0	IS41 0	IS40 0	IS31 0	IS30 0

12 Electrical characteristics

12.1 Parameter conditions

Unless otherwise specified, all voltages are referred to V_{SS}.

12.1.1 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 ($mean\pm3\Sigma$).

12.1.2 Typical values

Unless otherwise specified, typical data are based on T_A = 25 °C, V_{DD} = 5 V (for the 4.5 V≤ V_{DD}≤ 5.5 V voltage range). They are given only as design guidelines and are not tested.

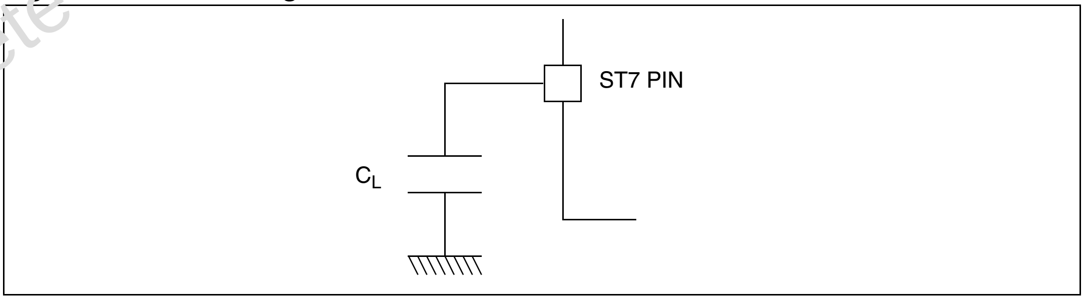
12.1.3 Typical curves

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

12.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in *Figure 36*.

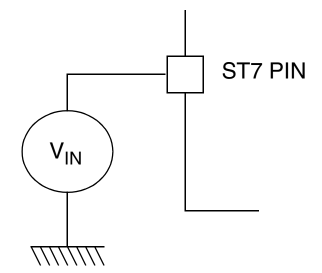
Figure 36. Pin loading conditions



12.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in *Figure 37*.

Figure 37. Pin input voltage



12.2 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.

Table 39. 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
$V_{ESD(HBM)}$	Electrostatic discharge voltage (Human Body model)	see *Section 12.7.3 on page 102*	
$V_{ESD(CDM)}$	Electrostatic discharge voltage (Charge Device model)	see *Section 12.7.3 on page 102*	

1. Directly connecting the RESET and I/O pins to V_{DD} or V_{SS} could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted Program Counter). To guarantee safe operation, this connection has to be done through a pull-up or pull-down resistor (typical: 4.7 kΩ for RESET, 10 kΩ for I/Os). Unused I/O pins must be tied in the same way to V_{DD} or V_{SS} according to their reset configuration.

2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by V_{IN}>V_{DD} while a negative injection is induced by V_{IN}<V_{SS}. For true open-drain pads, there is no positive injection current, and the corresponding V_{IN} maximum must always be respected

Table 40. Current characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source)(1)	75	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink)(1)	150	
I_{IO}	Output current sunk by any standard I/O and control pin	20	
	Output current sunk by any high sink I/O pin	40	
	Output current source by any I/Os and control pin	- 25	
$I_{INJ(PIN)}$(2)(3)	Injected current on $\overline{RESET}$ pin	± 5	
	Injected current on OSC1/CLKIN and OSC2 pins	± 5	
	Injected current on any other pin(4)	± 5	
$\Sigma I_{INJ(PIN)}$(2)	Total injected current (sum of all I/O and control pins)(4)	± 20	

1. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$. For true open-drain pads, there is no positive injection current, and the corresponding V_{IN} maximum must always be respected
3. Negative injection disturbs the analog performance of the device. In particular, it induces leakage currents throughout the device including the analog inputs. To avoid undesirable effects on the analog functions, care must be taken:
   - Analog input pins must have a negative injection less than 0.8 mA (assuming that the impedance of the analog voltage is lower than the specified limits)
   - Pure digital pins must have a negative injection less than 1.6 mA. In addition, it is recommended to inject the current as far as possible from the analog input pins.
4. When several inputs are submitted to a current injection, the maximum $\Sigma I_{INJ(PIN)}$ is the absolute sum of the positive and negative injected currents (instantaneous values). These results are based on characterization with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

Table 41. Thermal characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature (see *Table 66: Thermal characteristics on page 121*)		

12.3 Operating conditions

12.3.1 General operating conditions

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

Table 42. General operating conditions

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	Supply voltage	f_{CPU} = 8 MHz max.	4.5	5.5	V
f_{CPU}	CPU clock frequency	4.5 V≤ V_{DD}≤5.5 V	up to 8		MHz

12.3.2 Operating conditions with Low Voltage Detector (LVD)

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

Table 43. Operating characteristics with LVD

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(LVD)}$	Reset release threshold (V_{DD} rise)		3.9	4.2	4.5	V
$V_{IT-(LVD)}$	Reset generation threshold (V_{DD} fall)		3.7	4.0	4.3	
V_{hys}	LVD voltage threshold hysteresis	$V_{IT+(LVD)}$-$V_{IT-(LVD)}$		150		mV
V_{tPOR}	V_{DD} rise time rate[1][2]		2			µs/V
$I_{DD(LVD)}$	LVD current consumption	V_{DD} = 5 V		80	140	µA

1. Not tested in production. The V_{DD} rise time rate condition is needed to ensure a correct device power-on and LVD reset release. When the V_{DD} slope is outside these values, the LVD may not release properly the reset of the MCU.
2. Use of LVD with capacitive power supply: with this type of power supply, if power cuts occur in the application, it is recommended to pull V_{DD} down to 0 V to ensure optimum restart conditions. Refer to circuit example in *Figure 39 on page 106*.

12.3.3 Internal RC oscillator

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

Internal RC oscillator calibrated at 5.0 V

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

Table 44. Internal RC oscillator characteristics (5.0 V calibration)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal RC oscillator frequency	RCCR = FF (reset value), T_A = 25 °C, V_{DD} = 5 V		4.4		MHz
		RCCR=RCCR0[1], T_A = 25 °C, V_{DD} = 5 V		8		
$f_{G(RC)}$	RC trimming granularity	T_A = 25 °C, V_{DD} = 5 V		6		kHz
ACC_{RC}	Accuracy of Internal RC oscillator with RCCR=RCCR0[1]	T_A = 25 °C, V_{DD} = 5 V [2] without user calibration		±7		%
		T_A = 25 °C, V_{DD} = 4.5 to 5.5 V [2] with user calibration	-2		2	%
		T_A = 0 to +85 °C, V_{DD} = 4.5 to 5.5 V[2] with user calibration	-2.5		4	%
		T_A = -40 to 0 °C, V_{DD} = 4.5 to 5.5 V[2] with user calibration	-4		2.5	%
$t_{su(RC)}$	RC oscillator setup time	T_A = 25 °C, V_{DD} = 5 V		4 [3]		µs

1. See *Section 6.1.1: Internal RC oscillator*
2. Guaranteed by characterization
3. Not tested in production

12.4 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).

12.4.1 Supply current

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

Table 45. Supply current characteristics

Symbol	Parameter	Conditions		Typ	Max	Unit
I_{DD}	Supply current in Run mode[1]	V_{DD}=5V	f_{CPU} = 4 MHz	2.5	4.5[2]	mA
			f_{CPU} = 8 MHz	5.0	9	
	Supply current in Wait mode[3]		f_{CPU} = 4 MHz	1.1	2[2]	
			f_{CPU} = 8 MHz	2	3.5	
	Supply current in Slow mode[4]		$f_{CPU}/32$ = 250 kHz	550	950	µA
	Supply current in Slow-Wait mode[5]		$f_{CPU}/32$ = 250 kHz	450	750	
	Supply current in AWUFH mode[6][7]			50	100[2]	
	Supply current in Active Halt mode			120	250	
	Supply current in Halt mode[8]		T_A = 85 °C	0.5	5	

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

12.4.2 On-chip peripherals

Table 46. On-chip peripheral characteristics

Symbol	Parameter	Conditions		Typ	Unit
$I_{DD(AT)}$	12-bit Auto-Reload timer supply current(1)	f_{CPU}=8 MHz	V_{DD}=5.0 V	30	µA
$I_{DD(ADC)}$	ADC supply current when converting(2)	f_{ADC}=4 MHz	V_{DD}=5.0 V	750	µA

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

12.5 Clock and timing characteristics

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

Table 47. 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

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. $\Delta t_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.

12.5.1 Auto wakeup from Halt oscillator (AWU)

Table 48. AWU from Halt characteristics

Symbol	Parameter(1)	Conditions	Min	Typ	Max	Unit
f_{AWU}	AWU Oscillator Frequency		16	32	64	kHz
t_{RCSRT}	AWU Oscillator startup time				50	µs

1. Guaranteed by Design. Not tested in production.

12.6 Memory characteristics

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

Table 49. RAM and hardware registers characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode[1]	Halt mode (or Reset)	1.6			V

1. Minimum V_{DD} supply voltage without losing data stored in RAM (in Halt mode or under Reset) or in hardware registers (only in Halt mode). Guaranteed by construction, not tested in production.

Table 50. Flash program memory characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DD}	Operating voltage for Flash Write/Erase	Refer to operating range of V_{DD} with T_A, *Section 12.3.1 on page 95*	4.5		5.5	V
t_{prog}	Programming time for 1~32 bytes[1]	T_A=–40 to +85 °C		5	10	ms
	Programming time for 4 kbytes	T_A=+25 °C		0.64	1.28	s
t_{RET}	Data retention[2]	T_A=+55 °C[3]	20			years
N_{RW}	Write erase cycles	T_A=+25 °C			1k	cycles
I_{DD}	Supply current[4]	Read / Write / Erase modes f_{CPU} = 8 MHz, V_{DD} = 5.5 V			2.6	mA
		No Read/No Write mode			100	µA
		Power down mode / Halt		0	0.1	µA

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

12.7 EMC (electromagnetic compatibility) characteristics

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

12.7.1 Functional EMS (electromagnetic susceptibility)

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

- **ESD**: Electrostatic 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 100 pF capacitor, until a functional disturbance occurs. This test conforms with the IEC 1000-4-4 standard.

A device reset allows normal operations to be resumed. The test results are given in the table below based on the EMS levels and classes defined in application note AN1709.

Designing hardened software to avoid noise problems

EMC characterization and optimization are performed at component level with a typical application environment and simplified MCU software. It should be noted that good EMC performance is highly dependent on the user application and the software in particular.

Therefore it is recommended that the user applies EMC software optimization and prequalification tests in relation with the EMC level requested for his application.

- Software recommendations
  The software flowchart must include the management of runaway conditions such as:
  - Corrupted Program Counter
  - Unexpected reset
  - Critical Data corruption (control registers...)
- Prequalification trials
  Most of the common failures (unexpected reset and Program Counter corruption) can be reproduced by manually forcing a low state on the $\overline{RESET}$ pin or the Oscillator pins for 1 second.
  To complete these trials, ESD stress can be applied directly on the device, over the range of specification values. When unexpected behavior is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Table 51. EMS test results

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

12.7.2 EMI (Electromagnetic interference)

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.

Table 52. ST7FOXA0 EMI characteristics[1]

Symbol	Parameter	Conditions	Monitored Frequency Band	Max vs. [f_{OSC}/f_{CPU}]	Unit
				-/8MHz	
S_{EMI}	Peak level	V_{DD}=5 V, T_A=+25 °C, SO8 package, conforming to SAE J 1752/3	0.1 MHz to 30 MHz	20	dBµV
			30 MHz to 130 MHz	20	
			130 MHz to 1 GHz	13	
			SAE EMI Level	2.5	-

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

12.7.3 Absolute maximum ratings (electrical sensitivity)

Based on two different tests (ESD and LU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity.

Electrostatic discharge (ESD)

Electrostatic discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends on the number of supply pins in the device (3 parts*(n+1) supply pin). Two models can be simulated: Human Body model and Machine model. This test conforms to the JESD22-A114A/A115A standard. For more details, refer to the application note AN1181.

Table 53. ESD absolute maximum ratings

Symbol	Ratings	Conditions	Maximum value(1)	Unit
$V_{ESD(HBM)}$	Electrostatic discharge voltage (Human Body model)	T_A=+25 °C	4000	V
$V_{ESD(CDM)}$	Electrostatic discharge voltage (Charge Device model)	T_A=+25 °C	500	

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

Static latch-up (LU)

Two complementary static tests are required on six parts to assess the latch-up performance.

- A supply overvoltage is applied to each power supply pin
- A current injection is applied to each input, output and configurable I/O pin.

These tests are compliant with the EIA/JESD 78 IC latch-up standard.

Table 54. Electrical sensitivities

Symbol	Parameter	Conditions	Class
LU	Static latch-up class	T_A = +85 °C	A

12.8 I/O port pin characteristics

12.8.1 General characteristics

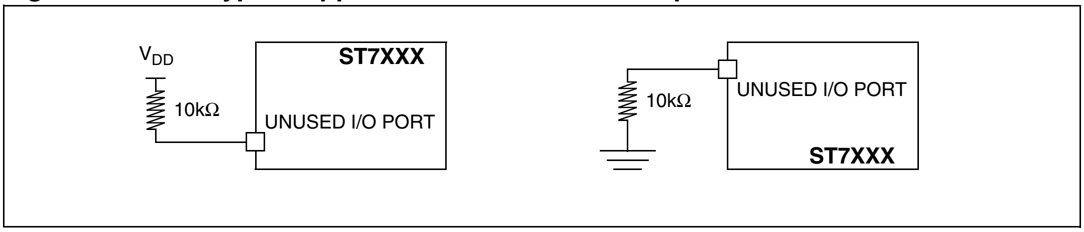
Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Table 55. General characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IL}	Input low level voltage		V_{SS} - 0.3		$0.3V_{DD}$	V
V_{IH}	Input high level voltage		$0.7V_{DD}$		V_{DD}+0.3	V
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		µA
R_{PU}	Weak pull-up equivalent resistor(3)	$V_{IN}=V_{SS}$, V_{DD}=5 V	100	120	140	kΩ
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time(1)	C_L = 50 pF Between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time(1)	C_L = 50 pF Between 10% and 90%		25		ns
$t_{w(IT)in}$	External interrupt pulse time(4)		1			t_{CPU}

1. Data based on validation/design results.
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 38*). 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.
4. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

Figure 38. Two typical applications with unused I/O pin



1. 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.
2. 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.

12.8.2 Output driving current

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

Table 56. Output driving current characteristics

Symbol	Parameter	Conditions		Min	Max	Unit
V_{OL} (1)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time	V_{DD} = 5 V	I_{IO}=+5 mA, $T_A \leq$ 85°C		1.2	V
			I_{IO}=+2mA, $T_A \leq$ 85°C		0.4	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time		I_{IO}=+20mA, $T_A \leq$ 85°C		1.3	
			I_{IO}=+8mA $T_A \leq$ 85°C		0.75	
V_{OH} (2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time		I_{IO}=-5mA, $T_A \leq$ 85°C	V_{DD}-1.5		
			I_{IO}=-2mA $T_A \leq$ 85°C	V_{DD}-0.8		

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section Table 40.* 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 Table 40.* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}.

12.9 Control pin characteristics

12.9.1 Asynchronous $\overline{RESET}$ pin

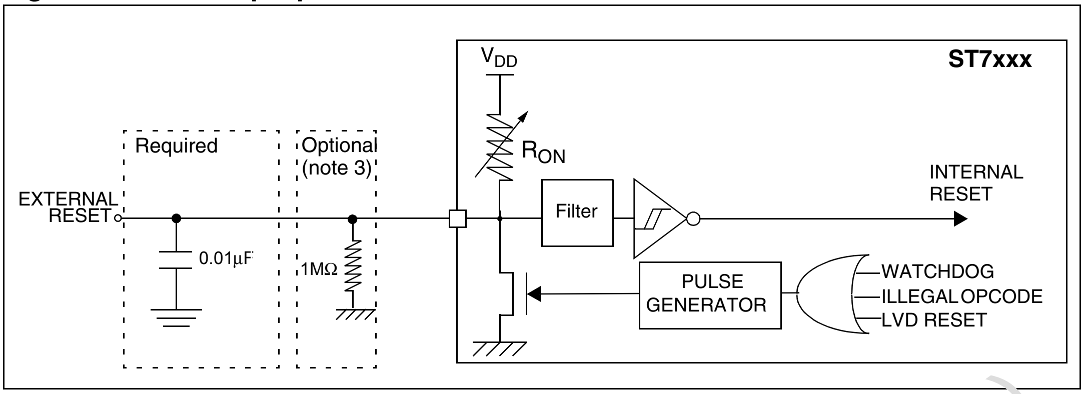
T_A = -40 to 85 °C, unless otherwise specified.

Table 57. Asynchronous $\overline{RESET}$ pin characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage			V_{SS} - 0.3		$0.3V_{DD}$	V
V_{IH}	Input high level voltage			$0.7V_{DD}$		V_{DD}+0.3	
V_{hys}	Schmitt trigger voltage hysteresis(1)				2		V
V_{OL}	Output low level voltage (2)	V_{DD}= 5 V	I_{IO} = +2 mA		200		mV
R_{ON}	Pull-up equivalent resistor(3)	V_{IN}=V_{SS}	V_{DD} = 5 V	30	50	70	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Internal reset sources			90(1)		µs
$t_{h(RSTL)in}$	External reset pulse hold time(4)			20			µs
$t_{g(RSTL)in}$	Filtered glitch duration				200		ns

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 Table 40. on page 94* 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 has to 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.

Figure 39. RESET pin protection when LVD is enabled

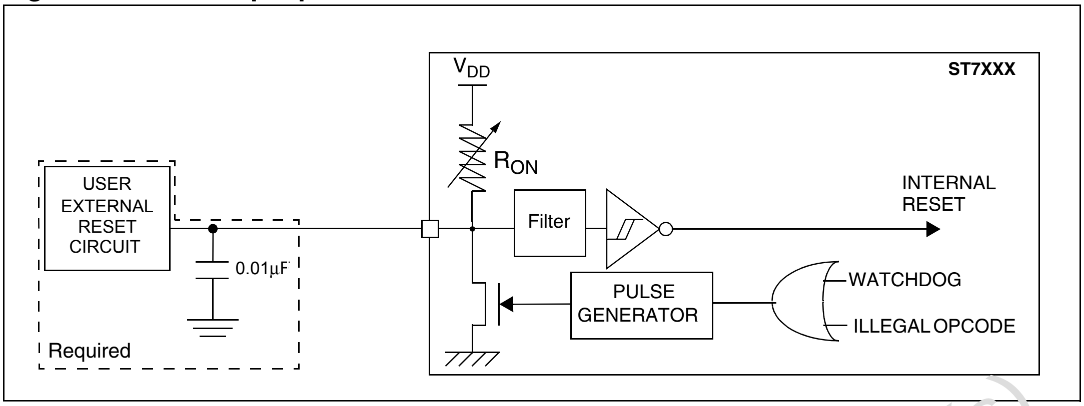


1. The reset network protects the device against parasitic resets. The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog). Whatever the reset source is (internal or external), the user must ensure that the level on the RESET pin can go below the V_{IL} max. level specified in *Section 12.9.1 on page 105*. 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 RESET pin, the user must ensure that the current sunk on the RESET pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in *Section Table 40. on page 94*.
2. When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.
3. In case a capacitive power supply is used, it is recommended to connect a 1MΩ pull-down resistor to the RESET pin to discharge any residual voltage induced by the capacitive effect of the power supply (this will add 5µA to the power consumption of the MCU).

Tips when using the LVD

- Check that all recommendations related to ICCCLK and reset circuit have been applied (see caution in *Table 2 on page 11* and notes above).
- Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100nF + 1MΩ pull-down on the RESET pin.
- The capacitors connected on the RESET pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the RESET pin with a 5µF to 20µF capacitor."

Figure 40. RESET pin protection when LVD is disabled



1. The reset network protects the device against parasitic resets.
   The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).
   Whatever the reset source is (internal or external), the user must ensure that the level on the RESET pin can go below the V_{IL} max. level specified in *Section 12.9.1 on page 105*. 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 RESET pin, the user must ensure that the current sunk on the RESET pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in *Section Table 40. on page 94*.
2. Please refer to *Section 10.2.1 on page 84* for more details on illegal opcode reset conditions.

12.10 10-bit ADC characteristics

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

Table 58. ADC characteristics

Symbol	Parameter	Conditions	Min	Typ[1]	Max	Unit
f_{ADC}	ADC clock frequency				4	MHz
V_{AIN}	Conversion voltage range		V_{SS}		V_{DD}	V
R_{AIN}	External input resistor	V_{DD} = 5 V, f_{ADC} = 4 MHz			8k[2]	Ω
		4.5 V ≤ V_{DD} ≤ 5.5 V, f_{ADC} = 2 MHz			10k[2]	
C_{ADC}	Internal sample and hold capacitor			3		pF
t_{STAB}	Stabilization time after ADC enable	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz		0[3]		µs
t_{ADC}	Conversion time (Sample+Hold)			3.5		
	- Sample capacitor loading time - Hold conversion time			4 10		$1/f_{ADC}$

1. Unless otherwise specified, typical data are based on T_A = 25 °C and V_{DD}-V_{SS} = 5 V. They are given only as design guidelines and are not tested.
2. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than the maximum value). Data guaranteed by Design, not tested in production.
3. The stabilization time of the A/D converter is masked by the first t_{LOAD}. The first conversion after the enable is then always valid.

Figure 41. Typical application with ADC

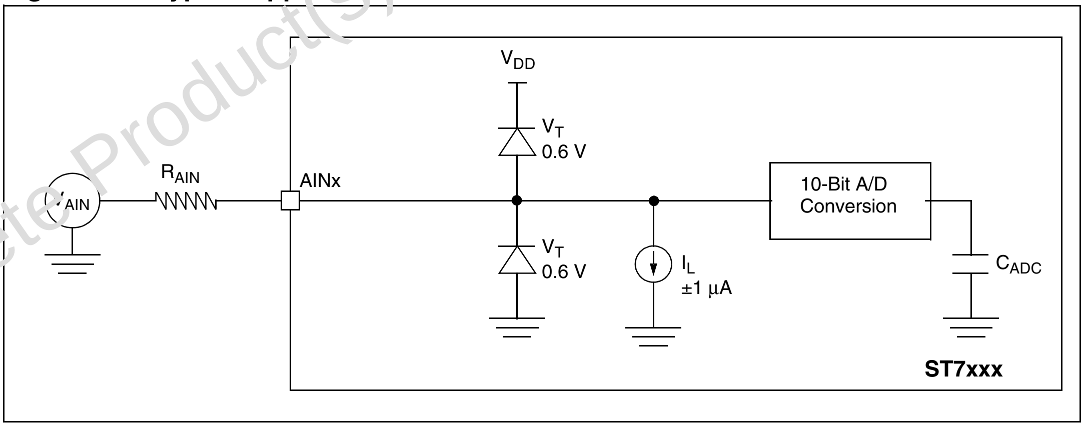
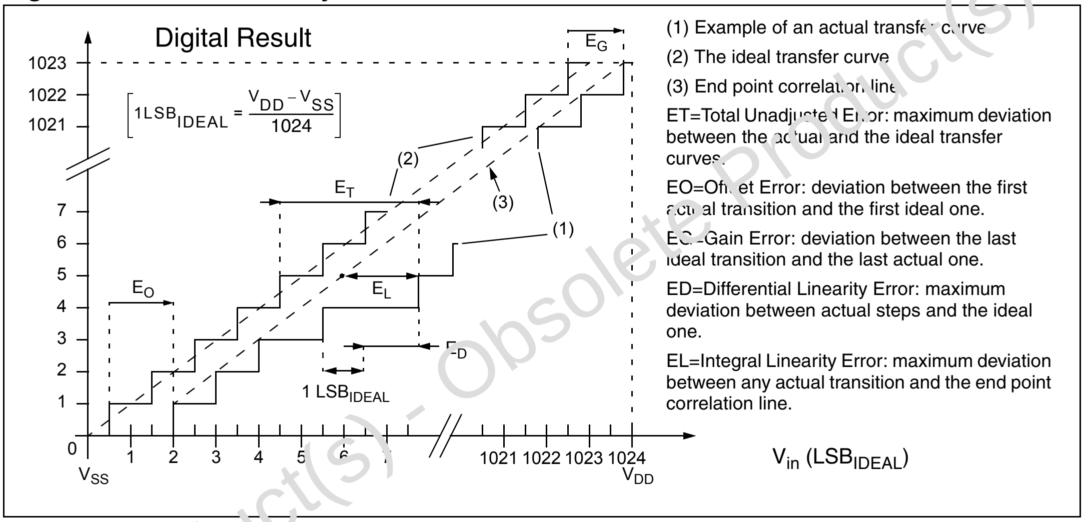


Table 59. ADC accuracy with V_{DD} = 4.5 to 5.5 V

Symbol (1)	Parameter	Conditions	Typ	Max	Unit
$\lvert E_T \rvert$	Total unadjusted error	f_{CPU}=8 MHz, f_{ADC}=4 MHz(1)	2.0	5.0	LSB
$\lvert E_O \rvert$	Offset error		0.9	2.5	
$\lvert E_G \rvert$	Gain Error		1.0	1.5	
$\lvert E_D \rvert$	Differential linearity error		1.2	3.5	
$\lvert E_L \rvert$	Integral linearity error		1.1	4.5	

1. Data based on characterization results over the whole temperature range.

Figure 42. ADC accuracy characteristics



13 Device configuration and ordering information

This device is available for production in user programmable version (Flash).

ST7FOXA0 XFlash devices are shipped to customers with a default program memory content (FFh).

13.1 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).

13.1.1 ST7FOXA0 Option byte 1

Bits 7:6 = **CKSEL[1:0]** *Start-up clock selection.*

These bits are used to select the startup frequency. By default, the internal RC is selected.

Table 60. Startup clock selection

Configuration	CKSEL1	CKSEL0
Internal RC as Startup Clock	0	0
AWU RC as a Startup Clock	0	1
Reserved	1	0
External Clock on pin PA5	1	1

Bit 5 = Reserved, must always be 1.

Bit 4 = Reserved, must always be 0.

Bit 3 = Reserved, must always be 1

Bit 2 = **LVD** *Low Voltage Detection selection.*

This option bit enables the low voltage detection block (LVD).

0: LVD on
1: LVD off (default value)

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

13.1.2 ST7FOXA0 Option byte 0

OPT 7:4 = Reserved, must always be set

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

These option bits indicate the size of sector 0 according to *Table 61*.

Table 61. Configuration of sector size

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

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 on page 19* 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.

0: Write protection off

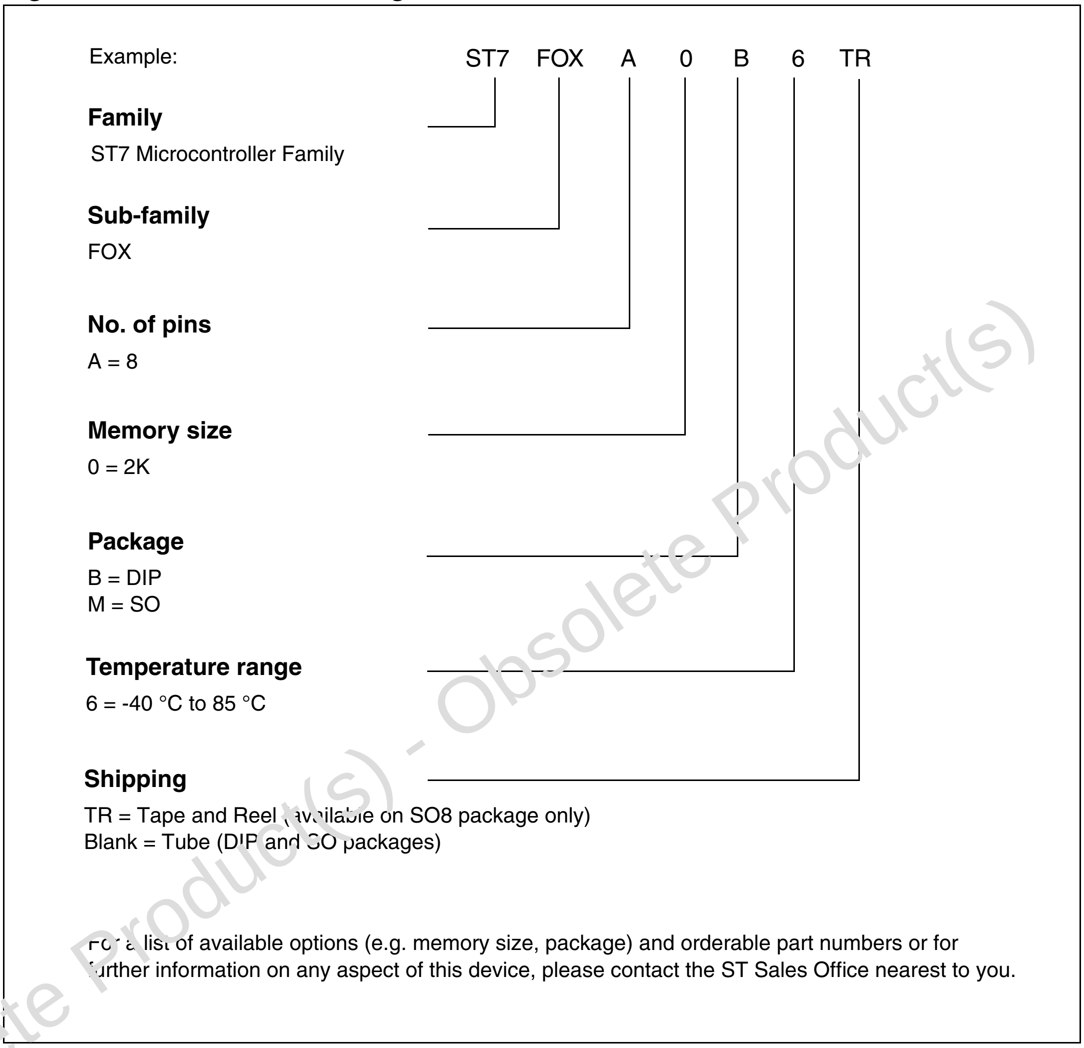
1: Write protection on

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

	Option byte 0 (7 … 0)								Option byte 1 (7 … 0)							
	Res	Res	Res	Res	SEC 1	SEC 0	FMP R	FMP W	CK SEL1	CK SEL0	Res	Res	Res	LVD	WDG SW	WDG HALT
Default value	1	1	1	1	0	0	0	0	0	0	1	0	1	1	1	1

13.2 Device ordering information

Figure 43. ST7FOXA0 ordering information scheme



ST7FOX failure analysis service

For ST7FOX family devices, STMicroelectronics agrees to accept return of defective parts subject to the FAR (Failure Analysis Report) procedure only if the customer reject rate exceeds 0.35 % for each delivered batch.

A batch is identified with a single trace code located on the top side marking.

13.3 Development tools

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.

13.3.1 Starter kits

ST offers complete, affordable **starter kits**. Starter kits are complete hardware/software tool packages that include features and samples to help you quickly start developing your application.

13.3.2 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 16Kbytes of code.

The range of hardware tools includes a full-featured **STice Emulator,** the low-cost **RLink** and the **ST7-STICK** in-circuit debugger/programmer. These tools are supported by the **ST7 Toolset** from STMicroelectronics, which includes the STVD7 integrated development environment (IDE) with high-level language debugger, editor, project manager and integrated programming interface.

13.3.3 Programming tools

During the development cycle, the **STice** emulator, the **ST7-STICK** 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 and **ST7 Socket Boards,** which provide all the sockets required for programming any of the devices in a specific ST7 sub-family 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.

13.3.4 Order codes for development and programming tools

Table 62 below lists the ordering codes for the ST7FOX development and programming tools. For additional ordering codes for spare parts and accessories, refer to the online product selector at www.st.com/mcu.

Table 62. Development tool order codes

MCU	Debugging and programming tool	ST socket boards
ST7FOXA0M6 ST7FOXA0B6	STX-RLINK(1)(2), ST7-STICK(3)(4), EMU3 or STice emulator(5)	ST7SB10-SU0 socket board(3)

1. USB connection to PC.
2. Available from ST or from Raisonance, www.raisonance.com.
3. Add suffix /EU, /UK or /US for the power supply for your region.
4. Parallel port connection to PC.
5. Contact local ST sales office for sales types.

13.4 ST7 application notes

Table 63. 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
AN1812	A high precision, low cost, single supply ADC for positive and negative input voltages
Example drivers	
AN 969	SCI communication between ST7 and PC
AN 970	SPI communication between ST7 and EEPROM
AN 971	I²C 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 wakeup on keystroke
AN1017	Using the ST7 Universal Serial Bus microcontroller
AN1041	Using ST7 PWM signal to generate analog output (sinusoïd)
AN1042	ST7 routine for I²C Slave mode Management
AN1044	Multiple interrupt sources management for ST7 MCUs
AN1045	ST7 S/W implementation of I²C bus master

Table 63. ST7 application notes (continued)

Identification	Description
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
AN1947	ST7MC PMAC sine wave motor control software library
General purpose	
AN1476	Low cost power supply for home appliances
AN1526	ST7FLITE0 quick reference note
AN1709	EMC design for ST microcontrollers
AN1752	ST72324 quick reference note
Product evaluation	
AN 910	Performance benchmarking
AN 990	ST7 benefits vs industry standard
AN1077	Overview of enhanced CAN controllers for ST7 and ST9 MCUs
AN1086	U435 can-do solutions for car multiplexing

Table 63. ST7 application notes (continued)

Identification	Description
AN1103	Improved B-EMF detection for low speed, low voltage with ST72141
AN1150	Benchmark ST72 vs PC16
AN1151	Performance comparison between ST72254 & PC16F876
AN1278	LIN (Local Interconnect Network) solutions
Product migration	
AN1131	Migrating applications from ST72511/311/214/124 to ST72521/321/324
AN1322	Migrating an application from ST7263 Rev.B to ST7263B
AN1365	Guidelines for migrating ST72C254 applications to ST72F264
AN1604	How to use ST7MDT1-TRAIN with ST72F264
AN2200	Guidelines for migrating ST7LITE1x applications to ST7FLITE1xB
Product optimization	
AN 982	Using ST7 with ceramic resonator
AN1014	How to minimize the ST7 power consumption
AN1015	Software techniques for improving microcontroller EMC performance
AN1040	Monitoring the Vbus signal for USB self-powered devices
AN1070	ST7 checksum self-checking capability
AN1181	Electrostatic discharge sensitive measurement
AN1324	Calibrating the RC oscillator of the ST7FLITE0 MCU using the mains
AN1502	Emulated data EEPROM with ST7 HD Flash memory
AN1529	Extending the current & voltage capability on the ST7265 V_{DDF} supply
AN1530	Accurate timebase for low-cost ST7 applications with internal RC oscillator
AN1605	Using an active RC to wake up the ST7LITE0 from power saving mode
AN1636	Understanding and minimizing ADC conversion errors
AN1828	PIR (passive infrared) detector using the ST7FLITE05/09/SUPERLITE
AN1946	Sensorless BLDC motor control and BEMF sampling methods with ST7MC
AN1953	PFC for ST7MC starter kit
AN1971	ST7LITE0 microcontrolled ballast
Programming and tools	
AN 978	ST7 Visual DeVELOP software key debugging features
AN 983	Key features of the Cosmic ST7 C-compiler package
AN 985	Executing code In ST7 RAM
AN 986	Using the indirect addressing mode with ST7
AN 987	ST7 serial test controller programming
AN 988	Starting with ST7 assembly tool chain

Table 63. ST7 application notes (continued)

Identification	Description
AN1039	ST7 math utility routines
AN1071	Half duplex USB-to-serial bridge using the ST72611 USB microcontroller
AN1106	Translating assembly code from HC05 to ST7
AN1179	Programming ST7 Flash microcontrollers in remote ISP mode (In-situ programming)
AN1446	Using the ST72521 emulator to debug an ST72324 target application
AN1477	Emulated data EEPROM with XFlash memory
AN1527	Developing a USB smartcard reader with ST7SCR
AN1575	On-board programming methods for XFlash and HD Flash ST7 MCUs
AN1576	In-application programming (IAP) drivers for ST7 HD Flash or XFlash MCUs
AN1577	Device firmware upgrade (DFU) Implementation for ST7 USB applications
AN1601	Software implementation for ST7DALI-EVAL
AN1603	Using the ST7 USB device firmware upgrade development kit (DFU-DK)
AN1635	ST7 customer ROM code release information
AN1754	Data logging program for testing ST7 applications via ICC
AN1796	Field updates for Flash memory based ST7 applications using a PC comm port
AN1900	Hardware implementation for ST7DALI-EVAL
AN1904	ST7MC three-phase AC induction motor control software library
AN1905	ST7MC three-phase BLDC motor control software library
System optimization	
AN1711	Software techniques for compensating ST7 ADC errors
AN1827	Implementation of SIGMA-DELTA ADC with ST7FLITE05/09
AN2009	PWM management for 3-phase BLDC motor drives using the ST7FMC
AN2030	Back EMF detection during PWM on time by ST7MC

14 Package mechanical data

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

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

Figure 44. 8-pin plastic small outline package - 150-mil width, package outline

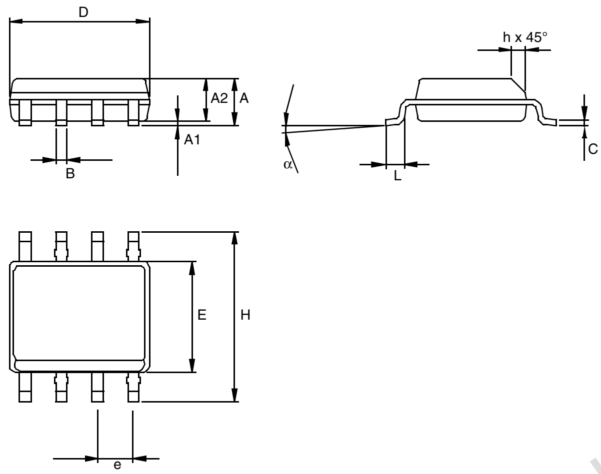


Table 64. 8-pin plastic small outline package, 150-mil width, mechanical data

Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A	1.35		1.75	0.0531		0.0689
A1	0.10		0.25	0.0039		0.0098
A2	1.10		1.65	0.0433		0.0650
B	0.33		0.51	0.0130		0.0201
C	0.19		0.25	0.0075		0.0098
D	4.80		5.00	0.1890		0.1969
E	3.80		4.00	0.1496		0.1575
e		1.27			0.0500	
H	5.80		6.20	0.2283		0.2441
h	0.25		0.50	0.0098		0.0197
α	0°		8°	0°		8°
L	0.40		1.27	0.0157		0.0500
	Number of Pins					
N	8					

1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 45. 8-pin plastic dual in-line outline package - 300-mil width, package outline

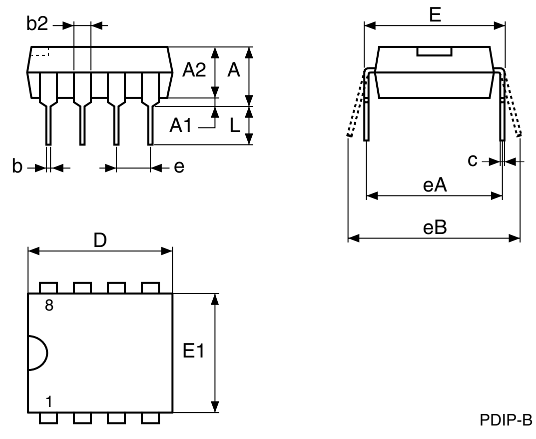


Table 65. 8-pin plastic dual in-line outline package - 300-mil width, mechanical data

Symbol	millimeters			inches(1)		
	Typ	Min	Max	Typ	Min	Max
A			5.33			0.2098
A1		0.38			0.0150	
A2	3.3	2.92	4.95	0.1299	0.1150	0.1949
b	0.46	0.36	0.56	0.0181	0.0142	0.0220
b2	1.52	1.14	1.78	0.0598	0.0449	0.0701
c	0.25	0.2	0.36	0.0098	0.0079	0.0142
D	9.27	9.02	10.16	0.3650	0.3551	0.4000
E	7.87	7.62	8.26	0.3098	0.3000	0.3252
E1	6.35	6.1	7.11	0.2500	0.2402	0.2799
e	2.54	-	-	0.1000		
eA	7.62	-	-	0.3000		
eB			10.92			0.4299
L	3.3	2.92	3.81	0.1299	0.1150	0.1500

1. Values in inches are converted from mm and rounded to 4 decimal digits.

14.1 Thermal characteristics

Table 66. Thermal characteristics

Symbol	Ratings		Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient)	SO8	130	°C/W
		DIP8	82	
T_{Jmax}	Maximum junction temperature(1)		150	°C
P_{Dmax}	Power dissipation(2)	SO8	180	mW
		DIP8	300	

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

15 Revision history

Table 67. Document revision history

Date	Revision	Changes
18-Oct-2007	1	Initial release
22-Nov-2007	2	Added LVD function Modified *Figure 3: ST7FOXA0 memory map on page 13* Modified note 4 in *Section 4.4: ICC interface on page 17* Added RCC_CSR in *Table 3: ST7FOXA0 Hardware register map on page 13* Added *Section 6.1.2: Customized RC calibration on page 26* *Section 12.7: EMC (electromagnetic compatibility) characteristics on page 100* modified Modified *Section 13.2: Device ordering information on page 112*
04-Feb-2008	3	ST7FOXU0 replaced by ST7FOXA0 Added LVD in *Figure 1: General block diagram on page 10* Modified *Figure 43: ST7FOXA0 ordering information scheme on page 112* Modified *Table 62: Development tool order codes on page 114*

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com