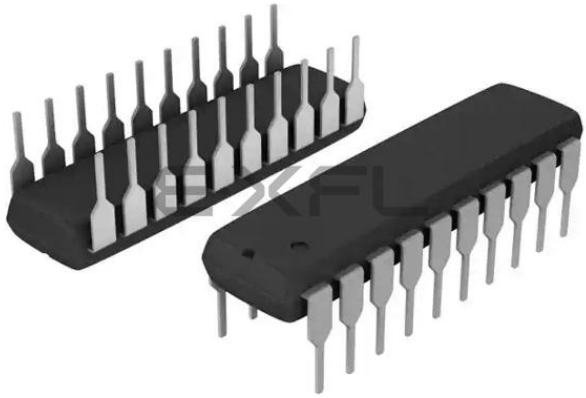


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	I²C
Peripherals	LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	4KB (4K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7foxf1b6

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

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 (32-pin packages)

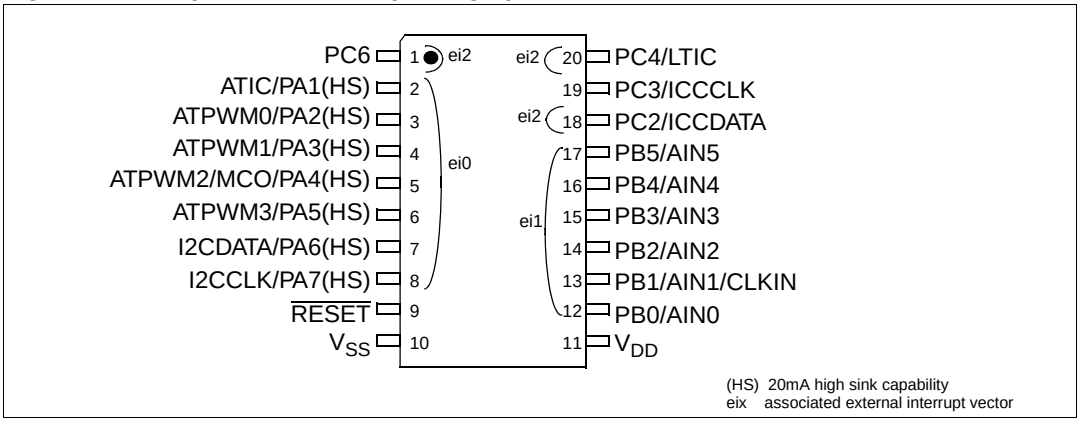
Pin number		Pin name	Type	Level		Port/control						Main function (after reset)	Alternate function
						Input				Output			
LQFP32	SDIP32			Input	Output	float	wpu	int	ana	OD[1]	PP		
1	5	PA3(HS)/ATPWM1	I/O	C_T	HS	**x**	ei0			x	x	**Port A3 (HS)**	ATPWM1
2	6	PA4(HS)/ ATPWM2/MCO	I/O	C_T	HS	**x**				x	x	**Port A4 (HS)**	ATPWM2/ MCO
3	7	PA5 (HS)ATPWM3	I/O	C_T	HS	**x**				x	x	**Port A5 (HS)**	ATPWM3
4	8	PA6(HS)/ I2CDATA/SCK(2)	I/O	C_T	HS	**x**		ei0		T		**Port A6 (HS)**	I2CDATA/SPI serial clock
5	9	PA7(HS)/I2CCLK/$\overline{SS}$ (2)	I/O	C_T	HS	**x**				T		**Port A7 (HS)**	I2CCLK/SPI slave select (active low)
6	10	RESET					**x**			x		Reset	
8	12	V_{DD}(3)	S									Digital Supply Voltage	
9	13	V_{SS}(3)	S									Digital Ground Voltage	
10	14	OSC1/CLKIN	I									Resonator oscillator inverter input or External clock input	
11	15	OSC2	O									Resonator oscillator output	
12	16	V_{SSA}(3)	S									Analog Ground Voltage	
13	17	V_{DDA}(3)	S									Analog Supply Voltage	

Table 2. Device pin description (32-pin packages) (continued)

Pin number		Pin name	Type	Level		Port/control						Main function (after reset)	Alternate function
						Input				Output			
LQFP32	SDIP32			Input	Output	float	wpu	int	ana	OD[1]	PP		
30	2	PA0 (HS)[5]/OCMP1_A[2]	I/O	C_T	HS [5]	**x**	ei0			x	x	**Port A0** (HS)[5]/ Timer A Output Compare 1	
31	3	PA1 (HS)/ATIC	I/O	C_T	HS	**x**				x	x	**Port A1 (HS)**	ATIC
32	4	PA2 (HS)/ATPWM0	I/O	C_T	HS	**x**				x	x	**Port A2 (HS)**	ATPWM0

1. In the open-drain output column, T defines a true open-drain I/O (P-Buffer and protection diode to V_{DD} are not implemented).
2. Available on ST7FOXK2 only.
3. 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.
4. BREAK2 available on ST7FOXK2 only
5. Available on ST7FOXK1 only.

Figure 4. 20-pin SO and DIP package pinout



Legend / Abbreviations for *Table 3*: Type: I = input, O = output, S = supply

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

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

Port and control configuration:

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

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

Figure 16. Low voltage detector vs reset

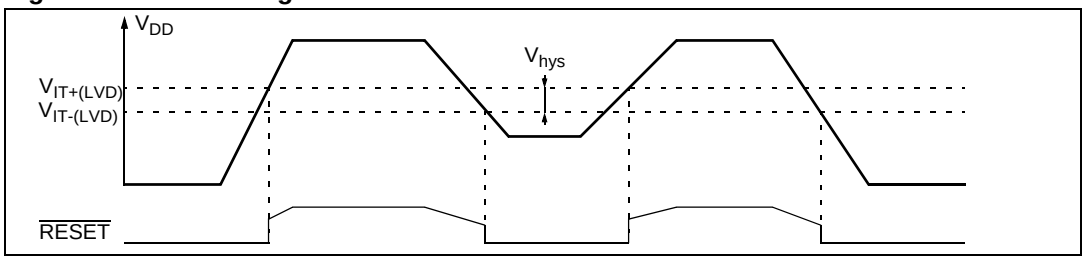
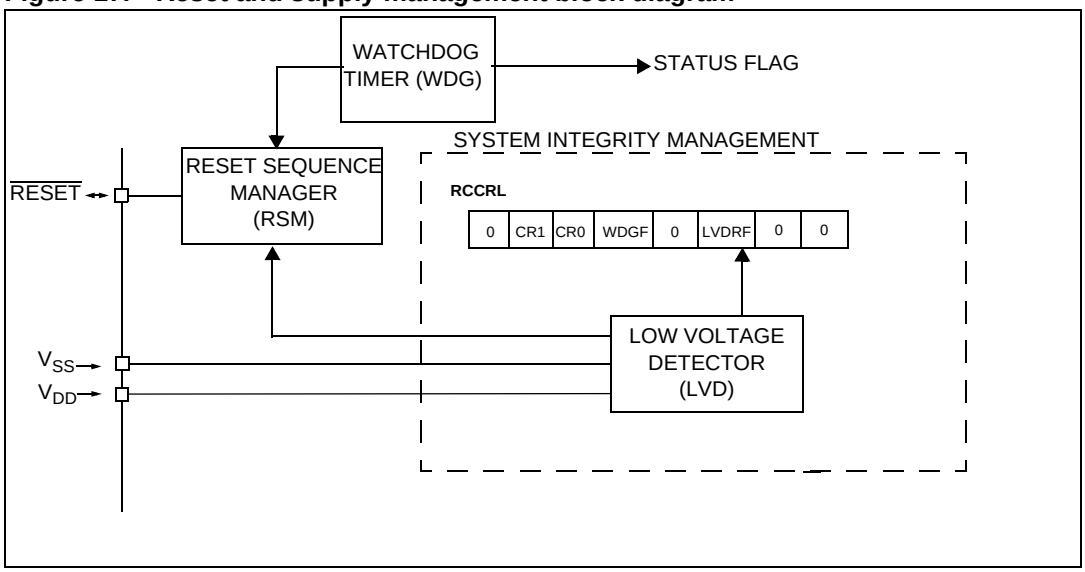


Figure 17. Reset and supply management block diagram



6.5.4 RC Control Register Low (RCCRL)

Reset value: 011x 0x00 (xxh)

7							0
0	CR1	CR0	WDGRF	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 34*.

Bit 4 = **WDGRF** *Watchdog Reset flag*

This bit indicates that the last reset was generated by the watchdog peripheral. It is set by hardware (watchdog reset) and cleared by software (writing zero) or an LVD Reset (to ensure a stable cleared state of the WDGRF flag when CPU starts). The WDGRF and the LVDRF flags areis used to select the reset source (see *Table 10: Reset source selection on page 48*).

Table 10. Reset source selection

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

Bit 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.

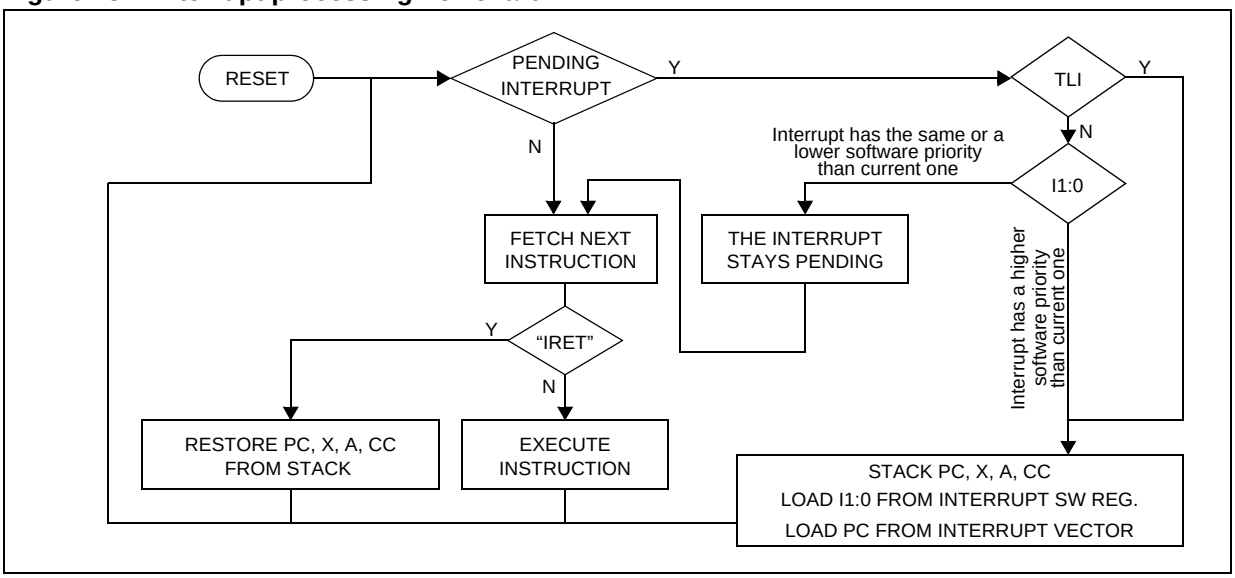
The LVDRF flag is not cleared when another RESET type occurs (external or watchdog), the LVDRF flag remains set to keep trace of the original failure.
In this case, a watchdog reset can be detected by software while an external reset can not.

Bits 1:0 = Reserved, must be kept cleared

Table 13. Interrupt software priority levels

Interrupt software priority	Level	I1	I0
Level 0 (main)	Low ↓ High	1	0
Level 1		0	1
Level 2			0
Level 3 (= interrupt disable)		1	1

Figure 18. Interrupt processing flowchart



Halt mode recommendations

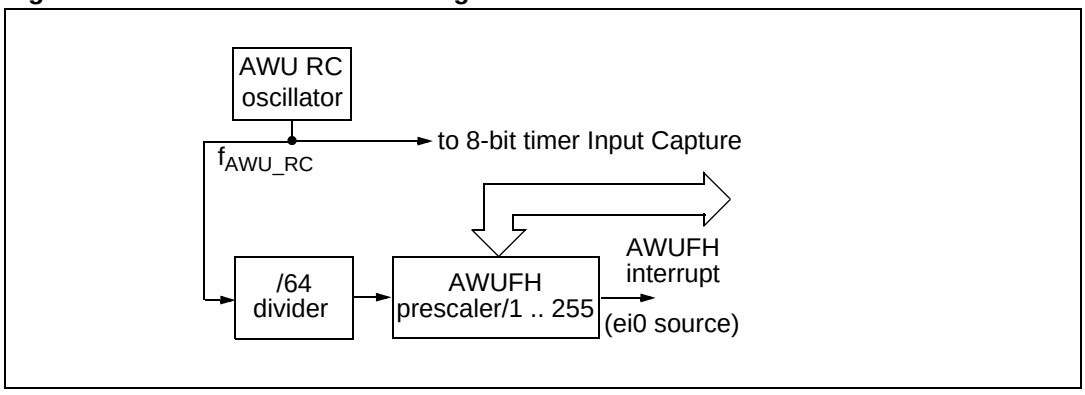
- Make sure that an external event is available to wake up the microcontroller from Halt mode.
- When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as “Input Pull-up with Interrupt” before executing the HALT instruction. The main reason for this is that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

8.5 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 29. AWUFH mode block diagram



9.2.3 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* and *Table 3* describe which peripheral signals can be input/output to which ports.

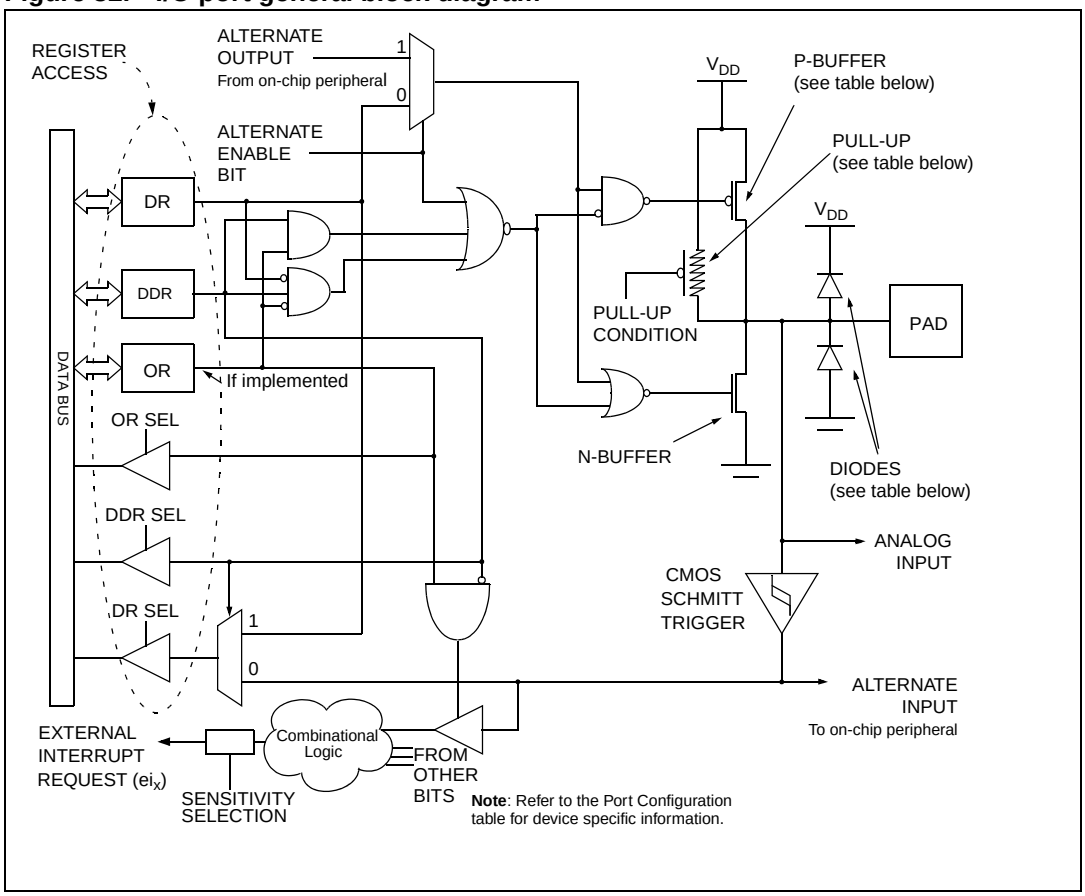
A signal coming from an on-chip peripheral can be output on an I/O. To do this, enable the on-chip peripheral as an output (enable bit in the peripheral's control register). The peripheral configures the I/O as an output and takes priority over standard I/O programming. The I/O's state is readable by 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 32. I/O port general block diagram



9.6 Interrupts

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

Table 27. 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

9.7 Device-specific I/O port configuration

The I/O port register configurations are summarized in *Section 9.7.1: Standard ports* and *Section 9.7.2: Other ports*.

9.7.1 Standard ports

Table 28. PA5:0, PB7:0, PC7:4 and PC2:0 pins

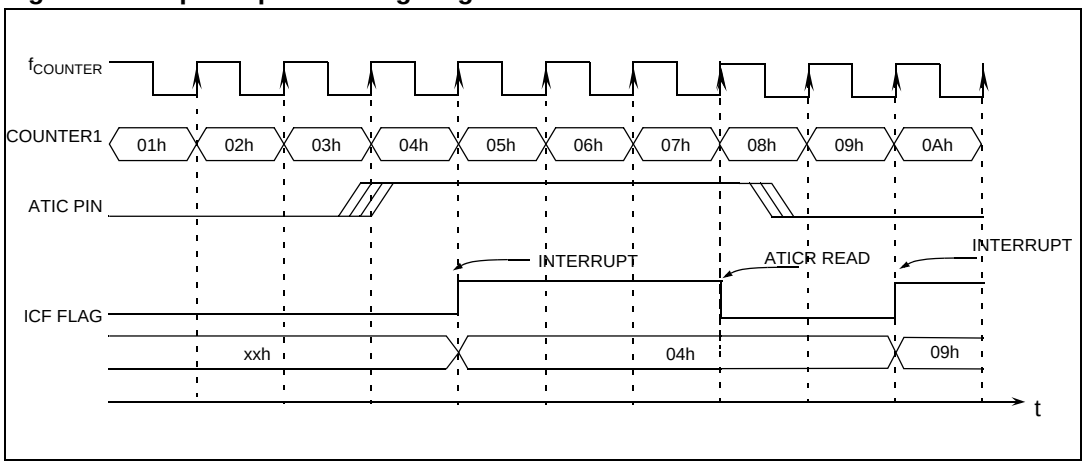
Mode	DDR	OR
floating input	0	0
pull-up interrupt input	0	1
open drain output	1	0
push-pull output	1	1

9.7.2 Other ports

Table 29. PA7:6 pins

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

Figure 45. Input capture timing diagram



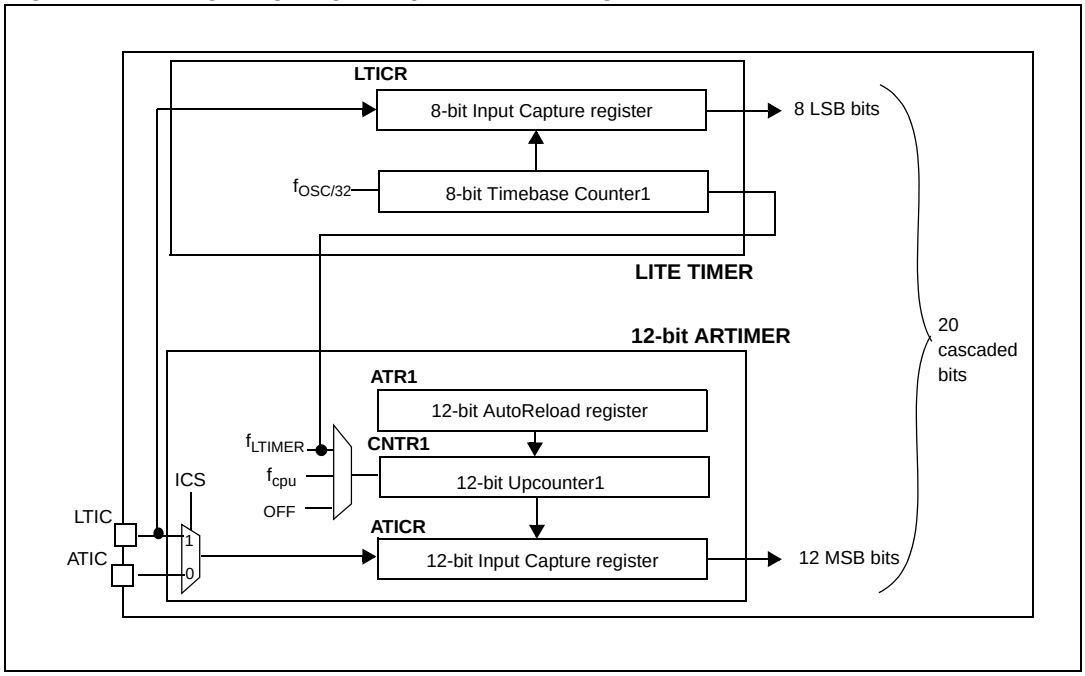
Long range input capture

Pulses that last more than 8 µs can be measured with an accuracy of 4 µs if f_{OSC} equals 8 MHz in the following conditions:

- The 12-bit AT4 timer is clocked by the Lite timer (RTC pulse: CK[1:0] = 01 in the ATCSR register)
- The ICS bit in the ATCSR2 register is set so that the LTIC pin is used to trigger the AT4 timer capture.
- The signal to be captured is connected to LTIC pin
- Input Capture registers LTICR, ATICRH and ATICRL are read

This configuration allows to cascade the Lite timer and the 12-bit AT4 timer to get a 20-bit Input Capture value. Refer to *Figure 46*.

Figure 46. Long range input capture block diagram



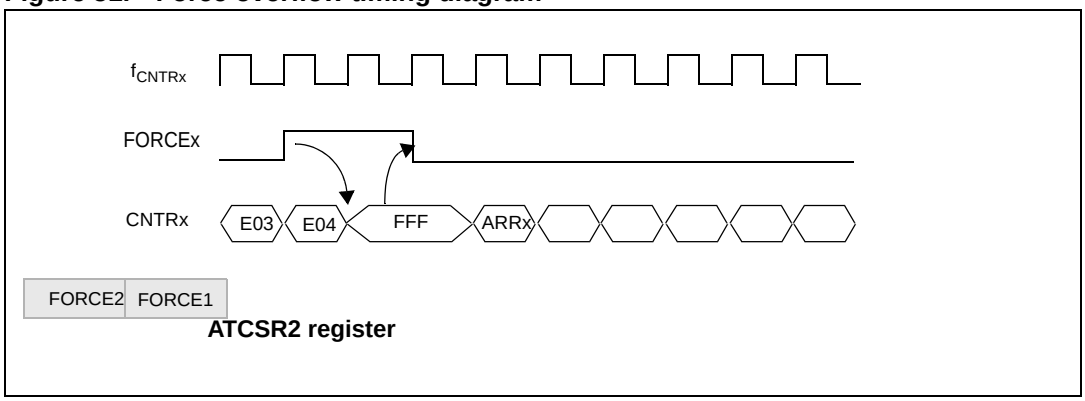
Force update

In order not to wait for the counter$_x$ overflow to load the value into active DCRx registers, a programmable counter$_x$ overflow is provided. For both counters, a separate bit is provided which when set, make the counters start with the overflow value, i.e. FFFh. After overflow, the counters start counting from their respective auto reload register values.

These bits are FORCE1 and FORCE2 in the ATCSR2 register. FORCE1 is used to force an overflow on Counter 1 and, FORCE2 is used for Counter 2. These bits are set by software and reset by hardware after the respective counter overflow event has occurred.

This feature can be used at any time. All related features such as PWM generation, Output Compare, Input Capture, One-pulse (refer to *Figure 50: Dynamic DCR2/3 update in one pulse mode*) etc. can be used this way.

Figure 51. Force overflow timing diagram



10.2.4 Low power modes

Table 35. Effect of low power modes on autoreload timer

Mode	Description
Wait	No effect on AT timer
Halt	AT timer halted.

10.2.5 Interrupts

Table 36. Description of interrupt events

Interrupt Event	Event Flag	Enable Control bit	Exit from Wait	Exit from Halt	Exit from Active-Halt
Overflow Event	OVF1	OVIE1	Yes	No	Yes
AT4 IC Event	ICF	ICIE	Yes	No	No
Overflow Event2	OVF2	OVIE2	Yes	No	No

Note: The AT4 IC is connected to an interrupt vector. The OVF event is mapped on a separate vector (see Interrupts chapter).
They generate an interrupt if the enable bit is set in the ATCSR register and the interrupt mask in the CC register is reset (RIM instruction).

10.3 Lite timer 2 (LT2)

10.3.1 Introduction

The Lite timer can be used for general-purpose timing functions. It is based on two free-running 8-bit upcounters, a watchdog function and an 8-bit Input Capture register.

10.3.2 Main features

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

Table 45. 16-bit timer register map and reset values (continued)

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
5E	TAACHR Reset value	MSB 1	1	1	1	1	1	1	LSB 1
5F	TAACLR Reset value	MSB 1	1	1	1	1	1	0	LSB 0
60	TAICHR2 Reset value	MSB -	-	-	-	-	-	-	LSB -
61	TAICLR2 Reset value	MSB -	-	-	-	-	-	-	LSB -
62	TAOCHR2 Reset value	MSB -	-	-	-	-	-	-	LSB -
63	TAOCLR2 Reset value	MSB -	-	-	-	-	-	-	LSB -

I²C Status register 1 (I2CSR1)

Reset value: 0000 0000 (00h)

7							0
EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB
Read Only							

Bit 7 = **EVF** *Event flag*

This bit is set by hardware as soon as an event occurs. It is cleared by software reading SR2 register in case of error event or as described in *Figure 71*. It is also cleared by hardware when the interface is disabled (PE=0).

0: No event

1: One of the following events has occurred:

– BTF=1 (byte received or transmitted)
– ADSL=1 (Address matched in Slave mode while ACK=1)
– SB=1 (Start condition generated in Master mode)
– AF=1 (No acknowledge received after byte transmission)
– STOPF=1 (Stop condition detected in Slave mode)
– ARLO=1 (Arbitration lost in Master mode)
– BERR=1 (Bus error, misplaced Start or Stop condition detected)
– ADD10=1 (Master has sent header byte)
– Address byte successfully transmitted in Master mode.

Bit 6 = **ADD10** *10-bit addressing in Master mode*

This bit is set by hardware when the master has sent the first byte in 10-bit address mode. It is cleared by software reading SR2 register followed by a write in the DR register of the second address byte. It is also cleared by hardware when the peripheral is disabled (PE=0).

0: No ADD10 event occurred.

1: Master has sent first address byte (header)

Bit 5 = **TRA** *Transmitter/Receiver bit*

When BTF is set, TRA=1 if a data byte has been transmitted. It is cleared automatically when BTF is cleared. It is also cleared by hardware after detection of Stop condition (STOPF=1), loss of bus arbitration (ARLO=1) or when the interface is disabled (PE=0).

0: Data byte received (if BTF=1)

1: Data byte transmitted

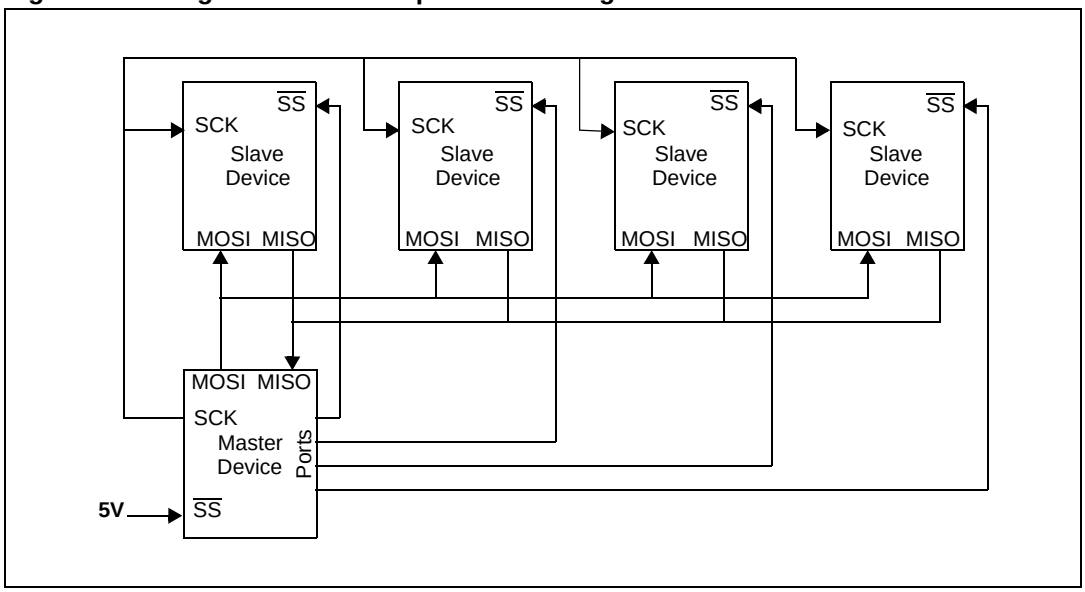
Bit 4 = **BUSY** *Bus busy* bit

This bit is set by hardware on detection of a Start condition and cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. The BUSY flag of the I2CSR1 register is cleared if a Bus Error occurs.

0: No communication on the bus

1: Communication ongoing on the bus

Figure 79. Single master / multiple slave configuration



10.6.7 Low power modes

Table 50. Low power mode descriptions

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

10.6.8 Interrupts

Table 51. Interrupt events

Interrupt event	Event flag	Enable control bit	Exit from Wait	Exit from Halt
SPI End of Transfer Event	SPIF	SPIE	Yes	No
Master Mode Fault Event	MODF			No
Overrun Error	OVR			

Note: The SPI interrupt events are connected to the same interrupt vector (see Interrupts chapter).

They generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

Table 62. 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

11.1.7 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 63. 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.

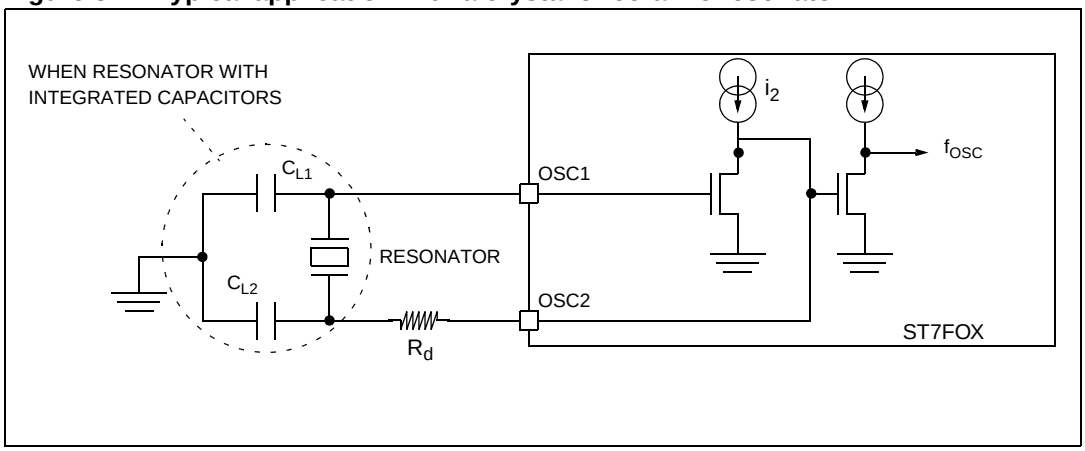
12.6.2 Crystal and ceramic resonator oscillators

The ST7 internal clock can be supplied with ten different Crystal/Ceramic resonator oscillators. All the information given in this paragraph are based on characterization results with specified typical external components. In the application, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. Refer to the crystal/ceramic resonator manufacturer for more details (frequency, package, accuracy...).

Table 80. Crystal/ceramic resonator oscillator characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{CrOSC}	Crystal oscillator frequency		2		16	MHz
C_{L1} C_{L2}	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator (R_S)		TBD			pF

Figure 87. Typical application with a crystal or ceramic resonator



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 95 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 96. 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^2C 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

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 94. 20-pin plastic small outline package, 300-mil width, package outline

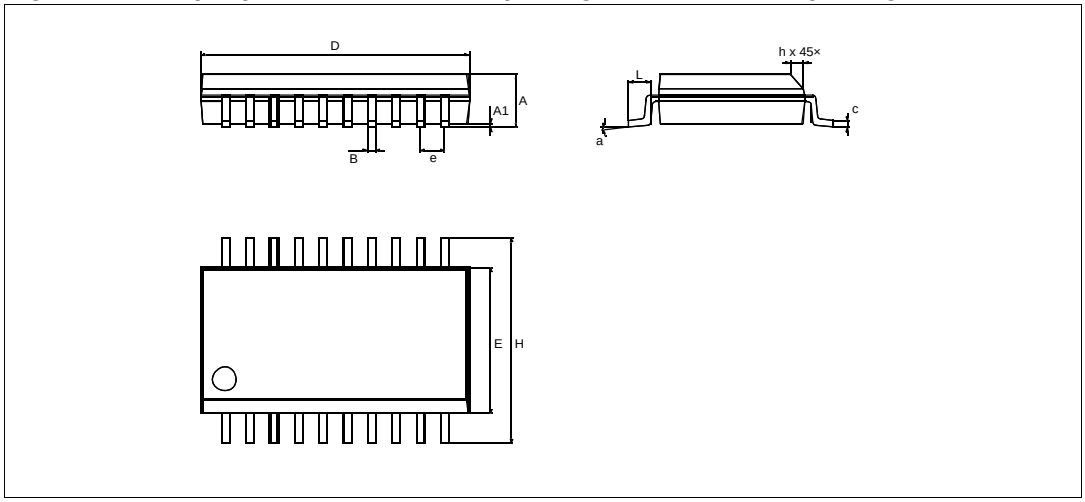


Table 97. 20-pin plastic small outline package, 300-mil width, mechanical data

Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A	2.35		2.65	0.0925		0.1043
A1	0.10		0.30	0.0039		0.0118
B	0.33		0.51	0.0130		0.0201
C	0.23		0.32	0.0091		0.0126
D	12.60		13.00	0.4961		0.5118
E	7.40		7.60	0.2913		0.2992
e		1.27			0.0500	
H	10.00		10.65	0.3937		0.4193
h	0.25		0.75	0.0098		0.0295
α	0°		8°	0°		8°
L	0.40		1.27	0.0157		0.0500
	Number of Pins					
N	20					

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