STMicroelectronics - ST72F361AR9T3 Datasheet

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	LINbusSCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	48
Program Memory Size	60KB (60K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f361ar9t3

PIN DESCRIPTION (Cont'd)

For external pin connection guidelines, refer to "ELECTRICAL CHARACTERISTICS" on page 178.

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 Schmitt trigger
	T_T= TTL 0.8V / 2V with Schmitt trigger
Output level:	HS = 20mA high sink (on N-buffer only)

Port and control configuration:

- Input: float = floating, wpu = weak pull-up, int = interrupt[1], ana = analog, RB = robust
- Output: OD = open drain, PP = push-pull

Refer to "I/O PORTS" on page 45 for more details on the software configuration of the I/O ports.

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

Pin n°			Pin Name	Type	Level		Port						Main function (after reset)	Alternate function	
							Input				Output				
LQFP64	LQFP44	LQFP32			Input	Output	float	wpu	int	ana	OD	PP			
1	1	1	OSC1[3]	I									External clock input or Resonator oscillator inverter input		
2	2	2	OSC2[3]	I/O									Resonator oscillator inverter output		
3	-	-	PA0 / ARTIC1	I/O	C_T		**X**	ei0			X	X	Port A0	ART Input Capture 1	
4	3	3	PA1 / PWM0	I/O	C_T		**X**		ei0		X	X	Port A1	ART PWM Output 0	
5	4	4	PA2 (HS) / PWM1	I/O	C_T	HS	**X**	ei0			X	X	Port A2	ART PWM Output 1	
6	5	-	PA3 / PWM2	I/O	C_T		**X**		ei0		X	X	Port A3	ART PWM Output 2	
7	6	-	PA4 / PWM3	I/O	C_T		**X**	ei0			X	X	Port A4	ART PWM Output 3	
8	-	-	V_{SS_3}	S									Digital Ground Voltage		
9	-	-	V_{DD_3}	S									Digital Main Supply Voltage		
10	7	5	PA5 (HS) / ARTCLK	I/O	C_T	HS	**X**		ei0		X	X	Port A5	ART External Clock	
11	8	-	PA6 (HS) / ARTIC2	I/O	C_T	HS	**X**	ei0			X	X	Port A6	ART Input Capture 2	
12	-	-	PA7 / T8_OCMP2	I/O	C_T		**X**		ei0		X	X	Port A7	TIM8 Output Compare 2	
13	-	-	PB0 /T8_ICAP2	I/O	C_T		**X**	ei1			X	X	Port B0	TIM8 Input Capture 2	
14	9	6	PB1 /T8_OCMP1	I/O	C_T		**X**		ei1		X	X	Port B1	TIM8 Output Compare 1	
15	10	7	PB2 / T8_ICAP1	I/O	C_T		**X**	ei1			X	X	Port B2	TIM8 Input Capture 1	
16	11	8	PB3 / MCO	I/O	C_T		**X**		ei1		X	X	Port B3	Main clock out (f_{OSC2})	
17	-	-	PE0 / AIN12	I/O	T_T		**X**	X		RB	X	X	Port E0	ADC Analog Input 12	
18	-	-	PE1 / AIN13	I/O	T_T		**X**	X		RB	X	X	Port E1	ADC Analog Input 13	
19	12	9	PB4 / AIN0 / ICCCLK	I/O	C_T		**X**	ei1		RB	X	X	Port B4	ICC Clock input	ADC Analog Input 0
20	-	-	PE2 / AIN14	I/O	T_T		**X**	X		RB	X	X	Port E2	ADC Analog Input 14	
21	-	-	PE3 / AIN15	I/O	T_T		**X**	X		RB	X	X	Port E3	ADC Analog Input 15	
22	13	10	PB5 / AIN1 / ICCDATA	I/O	C_T		**X**		ei1	RB	X	X	Port B5	ICC Data input	ADC Analog Input 1

4 FLASH PROGRAM MEMORY

4.1 INTRODUCTION

The ST7 dual voltage High Density Flash (HDFlash) is a non-volatile memory that can be electrically erased as a single block or by individual sectors and programmed on a Byte-by-Byte basis using an external V_{PP} supply.

The HDFlash devices can be programmed and erased off-board (plugged in a programming tool) or on-board using ICP (In-Circuit Programming) or IAP (In-Application Programming).

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

4.2 MAIN FEATURES

- 3 Flash programming modes:
  - Insertion in a programming tool. In this mode, all sectors including option bytes can be programmed or erased.
  - ICP (In-Circuit Programming). In this mode, all sectors including option bytes can be programmed or erased without removing the device from the application board.
  - IAP (In-Application Programming) In this mode, all sectors except Sector 0, can be programmed or erased without removing the device from the application board and while the application is running.
- ICT (In-Circuit Testing) for downloading and executing user application test patterns in RAM
- Read-out protection
- Register Access Security System (RASS) to prevent accidental programming or erasing

4.3 STRUCTURE

The Flash memory is organised in sectors and can be used for both code and data storage.

Depending on the overall Flash memory size in the microcontroller device, there are up to three user sectors (see Table 3). Each of these sectors can be erased independently to avoid unnecessary erasing of the whole Flash memory when only a partial erasing is required.

The first two sectors have a fixed size of 4 Kbytes (see Figure 6). They are mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

Table 4. Sectors available in Flash devices

Flash Size (bytes)	Available Sectors
4K	Sector 0
8K	Sectors 0,1
> 8K	Sectors 0,1, 2

4.3.1 Read-out Protection

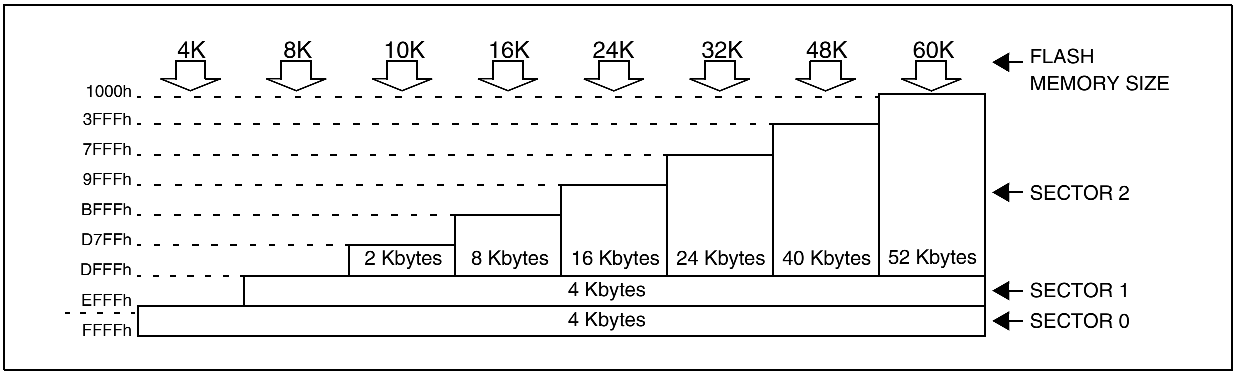
Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller.

In Flash devices, this protection is removed by reprogramming the option. In this case, the entire program memory is first automatically erased and the device can be reprogrammed.

Read-out protection selection depends on the device type:

- In Flash devices it is enabled and removed through the FMP_R bit in the option byte.
- In ROM devices it is enabled by mask option specified in the Option List.

Figure 6. Memory Map and Sector Address



POWER SAVING MODES (Cont'd)

8.6.1 Register Description

AWUFH CONTROL/STATUS REGISTER (AWUCSR)
Read/Write (except bit 2 read only)
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	AWU F	AWU M	AWU EN

Bits 7:3 = Reserved.

Bit 2 = **AWUF** *Auto Wake-Up Flag*
This bit is set by hardware when the AWU module generates an interrupt and cleared by software on reading AWUCSR.
0: No AWU interrupt occurred
1: AWU interrupt occurred

Bit 1 = **AWUM** *Auto Wake-Up Measurement*
This bit enables the AWU RC oscillator and connects its output to the ICAP1 input of the 16-bit timer. This allows the timer to be used to measure the AWU RC oscillator dispersion and then compensate this dispersion by providing the right value in the AWUPR register.
0: Measurement disabled
1: Measurement enabled

Bit 0 = **AWUEN** *Auto Wake-Up From Halt Enabled*
This bit enables the Auto Wake-Up From Halt feature: once HALT mode is entered, the AWUFH wakes up the microcontroller after a time delay defined by the AWU prescaler value. It is set and cleared by software.
0: AWUFH (Auto Wake-Up From Halt) mode disabled
1: AWUFH (Auto Wake-Up From Halt) mode enabled

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

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

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

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

In AWU mode, the period that the MCU stays in Halt Mode (t_{AWU} in Figure 30) is defined by

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

This prescaler register can be programmed to modify the time that the MCU stays in Halt mode before waking up automatically.

Note: If 00h is written to AWUPR, depending on the product, an interrupt is generated immediately after a HALT instruction or the AWUPR remains unchanged.

Table 11. AWU Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
002Bh	**AWUCSR** Reset Value	0	0	0	0	0	AWUF 0	AWUM 0	AWUEN 0
002Ch	**AWUPR** Reset Value	AWUPR7 1	AWUPR6 1	AWUPR5 1	AWUPR4 1	AWUPR3 1	AWUPR2 1	AWUPR1 1	AWUPR0 1

I/O PORTS (Cont'd)

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

Analog alternate function

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

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

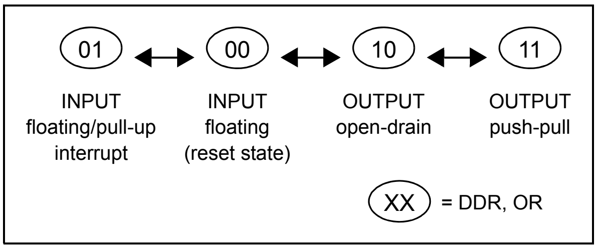
WARNING: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

9.3 I/O PORT IMPLEMENTATION

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific feature of the I/O port such as ADC Input or true 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 33 on page 49. Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 33. Interrupt I/O Port State Transitions



9.4 LOW POWER MODES

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

9.5 INTERRUPTS

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

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

I/O PORTS (Cont'd)

Table 15. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all IO port registers		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PAOR**								
0003h	**PBDR**	MSB							LSB
0004h	**PBDDR**								
0005h	**PBOR**								
0006h	**PCDR**	MSB							LSB
0007h	**PCDDR**								
0008h	**PCOR**								
0009h	**PDDR**	MSB							LSB
000Ah	**PDDDR**								
000Bh	**PDOR**								
000Ch	**PEDR**	MSB							LSB
000Dh	**PEDDR**								
000Eh	**PEOR**								
000Fh	**PFDR**	MSB							LSB
0010h	**PFDDR**								
0011h	**PFOR**								

ON-CHIP PERIPHERALS (Cont'd)

10.3 PWM AUTO-RELOAD TIMER (ART)

10.3.1 Introduction

The Pulse Width Modulated Auto-Reload Timer on-chip peripheral consists of an 8-bit auto reload counter with compare/capture capabilities and of a 7-bit prescaler clock source.

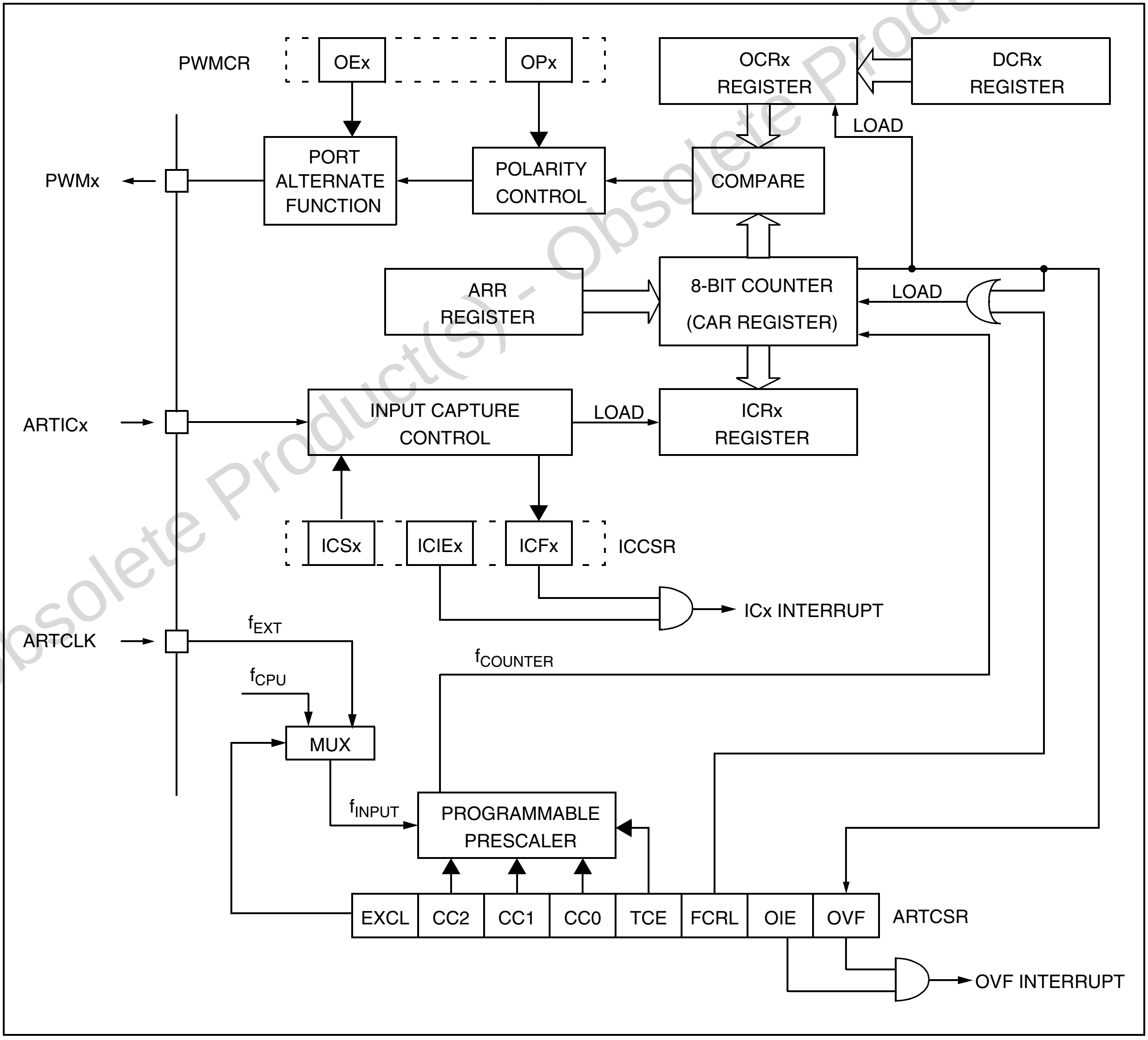
These resources allow five possible operating modes:

- Generation of up to four independent PWM signals
- Output compare and Time base interrupt
- Up to two input capture functions
- External event detector
- Up to two external interrupt sources

The three first modes can be used together with a single counter frequency.

The timer can be used to wake up the MCU from WAIT and HALT modes.

Figure 40. PWM Auto-Reload Timer Block Diagram



PWM AUTO-RELOAD TIMER (Cont'd)

10.3.2 Functional Description

Counter

The free running 8-bit counter is fed by the output of the prescaler, and is incremented on every rising edge of the clock signal.

It is possible to read or write the contents of the counter on the fly by reading or writing the Counter Access register (ARTCAR).

When a counter overflow occurs, the counter is automatically reloaded with the contents of the ARTARR register (the prescaler is not affected).

Counter clock and prescaler

The counter clock frequency is given by:

$f_{COUNTER} = f_{INPUT} / 2^{CC[2:0]}$

The timer counter's input clock (f_{INPUT}) feeds the 7-bit programmable prescaler, which selects one of the eight available taps of the prescaler, as defined by CC[2:0] bits in the Control/Status Register (ARTCSR). Thus the division factor of the prescaler can be set to 2^n (where n = 0, 1,..7).

This f_{INPUT} frequency source is selected through the EXCL bit of the ARTCSR register and can be either the f_{CPU} or an external input frequency f_{EXT}.

The clock input to the counter is enabled by the TCE (Timer Counter Enable) bit in the ARTCSR register. When TCE is reset, the counter is stopped and the prescaler and counter contents are frozen. When TCE is set, the counter runs at the rate of the selected clock source.

Counter and Prescaler Initialization

After RESET, the counter and the prescaler are cleared and $f_{INPUT} = f_{CPU}$.

The counter can be initialized by:

– Writing to the ARTARR register and then setting the FCRL (Force Counter Re-Load) and the TCE (Timer Counter Enable) bits in the ARTCSR register.

– Writing to the ARTCAR counter access register,

In both cases the 7-bit prescaler is also cleared, whereupon counting will start from a known value.

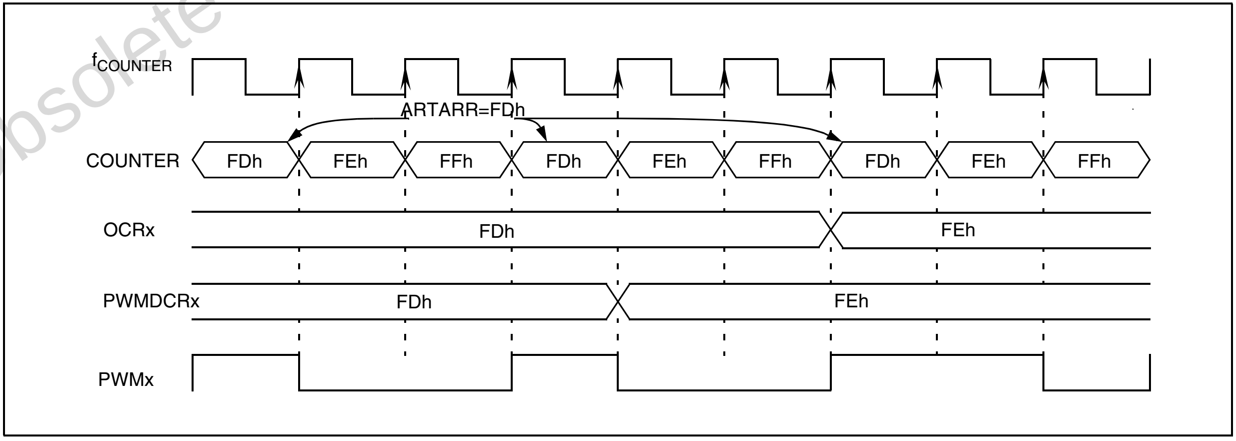
Direct access to the prescaler is not possible.

Output compare control

The timer compare function is based on four different comparisons with the counter (one for each PWMx output). Each comparison is made between the counter value and an output compare register (OCRx) value. This OCRx register can not be accessed directly, it is loaded from the duty cycle register (PWMDCRx) at each overflow of the counter.

This double buffering method avoids glitch generation when changing the duty cycle on the fly.

Figure 41. Output Compare Control



PWM AUTO-RELOAD TIMER (Cont'd)

Independent PWM signal generation

This mode allows up to four Pulse Width Modulated signals to be generated on the PWMx output pins with minimum core processing overhead. This function is stopped during HALT mode.

Each PWMx output signal can be selected independently using the corresponding OEx bit in the PWM Control register (PWMCR). When this bit is set, the corresponding I/O pin is configured as output push-pull alternate function.

The PWM signals all have the same frequency which is controlled by the counter period and the ARTARR register value.

$f_{PWM} = f_{COUNTER} / (256 - ARTARR)$

When a counter overflow occurs, the PWMx pin level is changed depending on the corresponding OPx (output polarity) bit in the PWMCR register.

When the counter reaches the value contained in one of the output compare register (OCRx) the corresponding PWMx pin level is restored.

It should be noted that the reload values will also affect the value and the resolution of the duty cycle of the PWM output signal. To obtain a signal on a PWMx pin, the contents of the OCRx register must be greater than the contents of the ARTARR register.

The maximum available resolution for the PWMx duty cycle is:

Resolution = 1 / (256 - ARTARR)

Note: To get the maximum resolution (1/256), the ARTARR register must be 0. With this maximum resolution, 0% and 100% can be obtained by changing the polarity.

Figure 42. PWM Auto-reload Timer Function

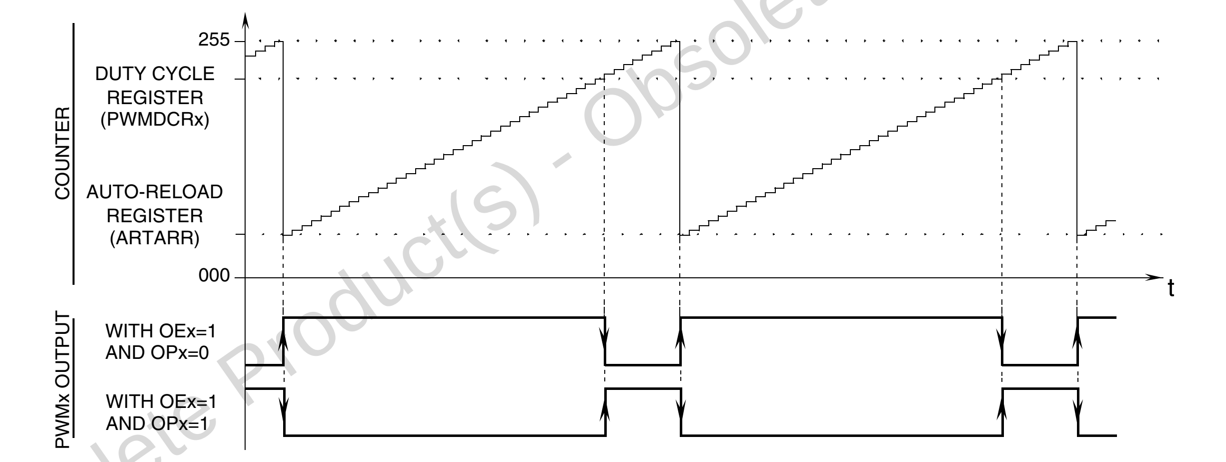
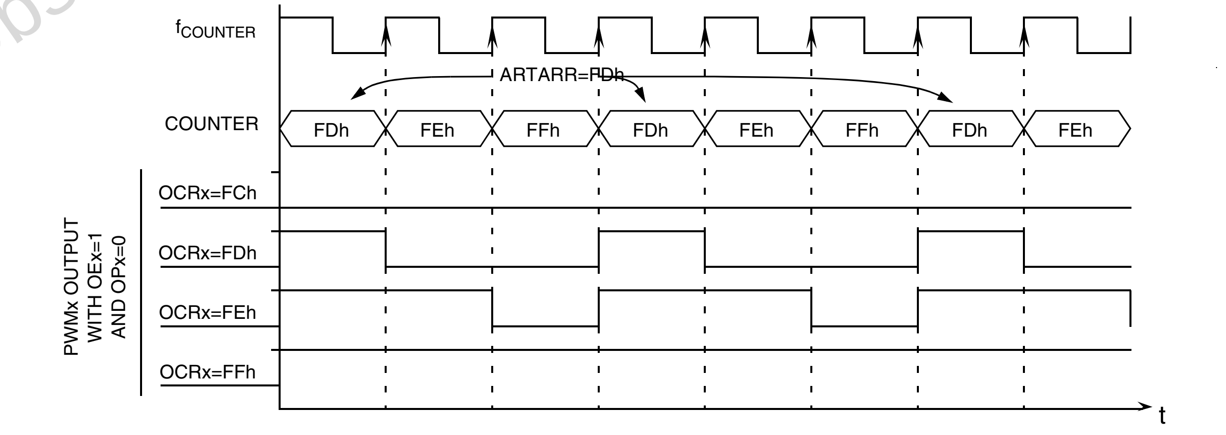


Figure 43. PWM Signal from 0% to 100% Duty Cycle



16-BIT TIMER (Cont'd)

Table 19. 16-Bit Timer Register Map

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
51	**CR2**	OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG
52	**CR1**	ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
53	**CSR**	ICF1	OCF1	TOF	ICF2	OCF2	TIMD		
54	**IC1HR**	MSB							LSB
55	**IC1LR**	MSB							LSB
56	**OC1HR**	MSB							LSB
57	**OC1LR**	MSB							LSB
58	**CHR**	MSB							LSB
59	**CLR**	MSB							LSB
5A	**ACHR**	MSB							LSB
5B	**ACLR**	MSB							LSB
5C	**IC2HR**	MSB							LSB
5D	**IC2LR**	MSB							LSB
5E	**OC2HR**	MSB							LSB
5F	**OC2LR**	MSB							LSB

10.5 8-BIT TIMER (TIM8)

10.5.1 Introduction

The timer consists of a 8-bit free-running counter driven by a programmable prescaler.

It may be used for a variety of purposes, including pulse length measurement of up to two input signals (*input capture*) or generation of up to two output waveforms (*output compare* and *PWM*).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the clock prescaler.

10.5.2 Main Features

- Programmable prescaler: f_{CPU} divided by 2, 4 , 8 or f_{OSC2} divided by 8000.
- Overflow status flag and maskable interrupt
- Output compare functions with
  - 2 dedicated 8-bit registers
  - 2 dedicated programmable signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Input capture functions with
  - 2 dedicated 8-bit registers
  - 2 dedicated active edge selection signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Pulse width modulation mode (PWM)
- One pulse mode
- Reduced Power Mode
- 4 alternate functions on I/O ports (ICAP1, ICAP2, OCMP1, OCMP2)*

The Block Diagram is shown in Figure 59.

***Note:** Some timer pins may not be available (not bonded) in some ST7 devices. Refer to the device pin out description.

When reading an input signal on a non-bonded pin, the value will always be ‘1’.

10.5.3 Functional Description

10.5.3.1 Counter

The main block of the Programmable Timer is a 8-bit free running upcounter and its associated 8-bit registers.

These two read-only 8-bit registers contain the same value but with the difference that reading the ACTR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR).

Writing in the CTR register or ACTR register resets the free running counter to the FCh value.
Both counters have a reset value of FCh (this is the only value which is reloaded in the 8-bit timer).
The reset value of both counters is also FCh in One Pulse mode and PWM mode.

The timer clock depends on the clock control bits of the CR2 register, as shown in Table 19 Clock Control Bits. The value in the counter register repeats every 512, 1024, 2048 or 20480000 f_{CPU} clock cycles depending on the CC[1:0] bits.
The timer frequency can be $f_{CPU}/2$, $f_{CPU}/4$, $f_{CPU}/8$ or f_{OSC2} /8000.

For example, if $f_{OSC2}/8000$ is selected, and f_{OSC2} = 8 MHz, the timer frequency will be 1 ms. Refer to Table 19 on page 105.

8-BIT TIMER (Cont'd)

Whatever the timer mode used (input capture, output compare, one pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFh to 00h then:

– The TOF bit of the SR register is set.

– A timer interrupt is generated if:

  – TOIE bit of the CR1 register is set and

  – I bit of the CC register is cleared.

If one of these conditions is false, the interrupt remains pending to be issued as soon as they are both true.

Clearing the overflow interrupt request is done in two steps:

1. Reading the SR register while the TOF bit is set.
2. An access (read or write) to the CTR register.

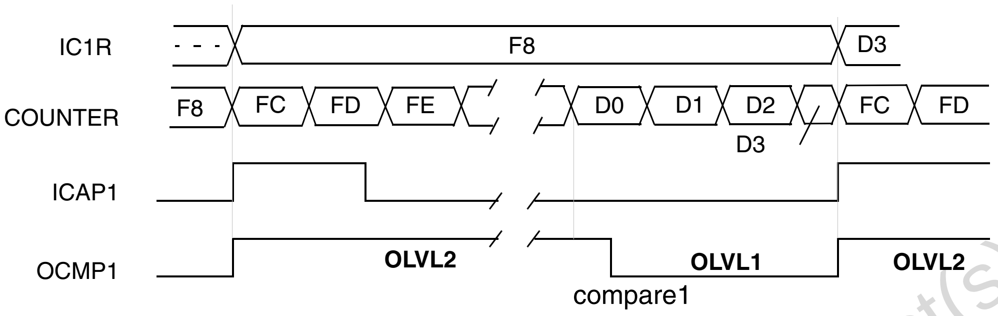
Notes: The TOF bit is not cleared by accesses to ACTR register. The advantage of accessing the ACTR register rather than the CTR register is that it allows simultaneous use of the overflow function and reading the free running counter at random times (for example, to measure elapsed time) without the risk of clearing the TOF bit erroneously.

The timer is not affected by WAIT mode.

In HALT mode, the counter stops counting until the mode is exited. Counting then resumes from the previous count (MCU awakened by an interrupt) or from the reset count (MCU awakened by a Reset).

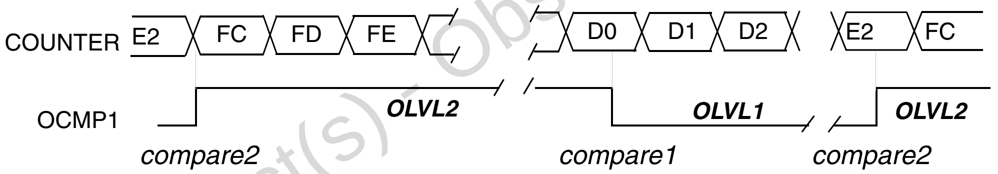
8-BIT TIMER (Cont'd)

Figure 68. One Pulse Mode Timing Example



Note: IEDG1 = 1, OC1R = D0h, OLVL1 = 0, OLVL2 = 1

Figure 69. Pulse Width Modulation Mode Timing Example



Note: OC1R = D0h, OC2R = E2, OLVL1 = 0, OLVL2 = 1

8-BIT TIMER (Cont'd)

10.5.4 Low Power Modes

Mode	Description
WAIT	No effect on 8-bit Timer. Timer interrupts cause the device to exit from WAIT mode.
HALT	8-bit Timer registers are frozen. In HALT mode, the counter stops counting until Halt mode is exited. Counting resumes from the previous count when the MCU is woken up by an interrupt with "exit from HALT mode" capability or from the counter reset value when the MCU is woken up by a RESET. If an input capture event occurs on the ICAP*i* pin, the input capture detection circuitry is armed. Consequently, when the MCU is woken up by an interrupt with "exit from HALT mode" capability, the ICF*i* bit is set, and the counter value present when exiting from HALT mode is captured into the IC*i*R register.

10.5.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Input Capture 1 event/Counter reset in PWM mode	ICF1	ICIE	Yes	No
Input Capture 2 event	ICF2			
Output Compare 1 event (not available in PWM mode)	OCF1	OCIE		
Output Compare 2 event (not available in PWM mode)	OCF2			
Timer Overflow event	TOF	TOIE		

Note: The 8-bit Timer interrupt events are connected to the same interrupt vector (see Interrupts chapter). These events generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

10.5.6 Summary of Timer modes

MODES	AVAILABLE RESOURCES			
	Input Capture 1	Input Capture 2	Output Compare 1	Output Compare 2
Input Capture (1 and/or 2)	Yes	Yes	Yes	Yes
Output Compare (1 and/or 2)				
One Pulse Mode	No	Not Recommended[1)]	No	Partially [2)]
PWM Mode		Not Recommended[3)]		No

1) See note 4 in "One Pulse Mode" on page 100

2) See note 5 in "One Pulse Mode" on page 100

3) See note 4 in "Pulse Width Modulation Mode" on page 102

8-BIT TIMER (Cont'd)

10.5.8 8-bit Timer Register Map

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
3C	**CR2**	OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	0
3D	**CR1**	ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
3E	**CSR**	ICF1	OCF1	TOF	ICF2	OCF2	TIMD		
3F	**IC1R**	MSB							LSB
40	**OC1R**	MSB							LSB
41	**CTR**	MSB							LSB
42	**ACTR**	MSB							LSB
43	**IC2R**	MSB							LSB
44	**OC2R**	MSB							LSB

ON-CHIP PERIPHERALS (cont'd)

10.6 SERIAL PERIPHERAL INTERFACE (SPI)

10.6.1 Introduction

The Serial Peripheral Interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

10.6.2 Main Features

- Full duplex synchronous transfers (on three lines)
- Simplex synchronous transfers (on two lines)
- Master or slave operation
- 6 master mode frequencies (f_{CPU}/4 max.)
- f_{CPU}/2 max. slave mode frequency (see note)
- $\overline{SS}$ Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

10.6.3 General Description

Figure 70 on page 110 shows the serial peripheral interface (SPI) block diagram. There are three registers:

– SPI Control Register (SPICR)

– SPI Control/Status Register (SPICSR)

– SPI Data Register (SPIDR)

The SPI is connected to external devices through four pins:

– MISO: Master In / Slave Out data

– MOSI: Master Out / Slave In data

– SCK: Serial Clock out by SPI masters and input by SPI slaves

– $\overline{SS}$: Slave select:
This input signal acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave $\overline{SS}$ inputs can be driven by standard I/O ports on the master Device.

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

CONTROL REGISTER 2 (SCICR2)

Read/Write

Reset Value: 0000 0000 (00h)

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

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

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

Bit 6 = **TCIE** *Transmission complete interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TC = 1 in the SCISR register

Bit 5 = **RIE** *Receiver interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever OR = 1 or RDRF = 1 in the SCISR register

Bit 4 = **ILIE** *Idle line interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever IDLE = 1 in the SCISR register.

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

Notes:

– During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word.

– When TE is set there is a 1 bit-time delay before the transmission starts.

Bit 2 = **RE** *Receiver enable*
This bit enables the receiver. It is set and cleared by software.
0: Receiver is disabled in the SCISR register
1: Receiver is enabled and begins searching for a start bit

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

Notes:

– Before selecting Mute mode (by setting the RWU bit) the SCI must first receive a data byte, otherwise it cannot function in Mute mode with wake-up by Idle line detection.

– In Address Mark Detection Wake-Up configuration (WAKE bit = 1) the RWU bit cannot be modified by software while the RDRF bit is set.

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

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

DATA REGISTER (SCIDR)

Read/Write

Reset Value: Undefined

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

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).
The TDR register provides the parallel interface between the internal bus and the output shift register (see Figure 1).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 1).

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

EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERPR)

Read/Write

Reset Value: 0000 0000 (00h)

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

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

The extended Baud Rate Generator is activated when a value other than 00h is stored in this register. The clock frequency from the 16 divider (see Figure 3) is divided by the binary factor set in the SCIERPR register (in the range 1 to 255).

The extended baud rate generator is not active after a reset.

EXTENDED TRANSMIT PRESCALER DIVISION REGISTER (SCIETPR)

Read/Write

Reset Value:0000 0000 (00h)

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

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

The extended Baud Rate Generator is activated when a value other than 00h is stored in this register. The clock frequency from the 16 divider (see Figure 3) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255).

The extended baud rate generator is not active after a reset.

Note: In LIN slave mode, the Conventional and Extended Baud Rate Generators are disabled.

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

10.7.9.9 Error due to LIN Synch measurement

The LIN Synch Field is measured over eight bit times.

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

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

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

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

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

10.7.9.10 Error due to Baud Rate Quantization

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

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

10.7.9.11 Impact of Clock Deviation on Maximum Baud Rate

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

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

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

Example:

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

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

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

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

LIN Slave systems

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

Hot Plugging Feature for LIN Slave Nodes

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

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

10.8.4.4 Conventional Baud Rate Generation

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

:

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

with:

PR = 1, 3, 4 or 13 (see SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64, 128

(see SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64, 128

(see SCR[2:0] bits)

All these bits are in the SCIBRR register.

Example: If f_{CPU} is 8 MHz (normal mode) and if PR = 13 and TR = RR = 1, the transmit and receive baud rates are 38400 baud.

Note: The baud rate registers MUST NOT be changed while the transmitter or the receiver is enabled.

10.8.4.5 Extended Baud Rate Generation

The extended prescaler option gives a very fine tuning on the baud rate, using a 255 value prescaler, whereas the conventional Baud Rate Generator retains industry standard software compatibility.

The extended baud rate generator block diagram is described in the Figure 90.

The output clock rate sent to the transmitter or to the receiver is the output from the 16 divider divided by a factor ranging from 1 to 255 set in the SCIERPR or the SCIETPR register.

Note: The extended prescaler is activated by setting the SCIETPR or SCIERPR register to a value other than zero. The baud rates are calculated as follows:

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

with:

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

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

10.8.4.6 Receiver Muting and Wake-up Feature

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant SCI service overhead for all non addressed receivers.

The non addressed devices may be placed in sleep mode by means of the muting function.

Setting the RWU bit by software puts the SCI in sleep mode:

All the reception status bits cannot be set.

All the receive interrupts are inhibited.

A muted receiver may be awakened by one of the following two ways:

– by Idle Line detection if the WAKE bit is reset,

– by Address Mark detection if the WAKE bit is set.

Receiver wakes-up by Idle Line detection when the Receive line has recognized an Idle Frame. Then the RWU bit is reset by hardware but the IDLE bit is not set.

Receiver wakes-up by Address Mark detection when it received a “1” as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

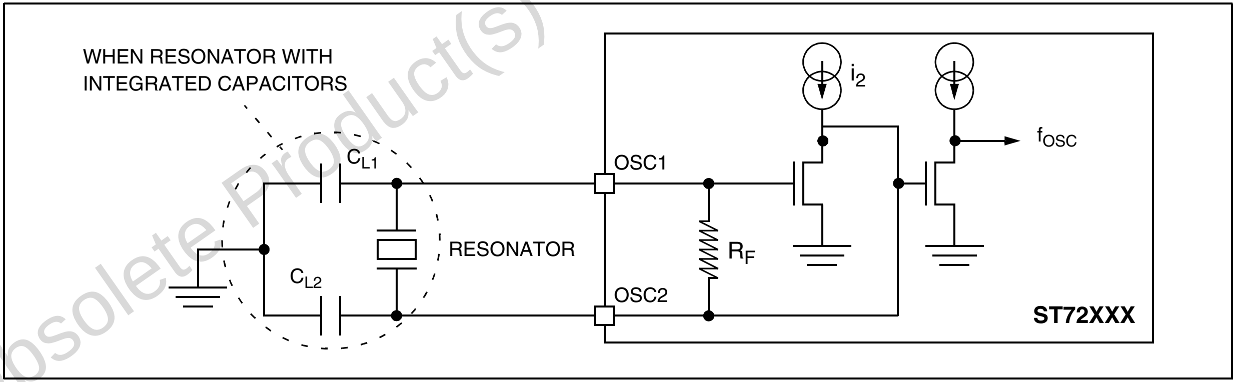
12.5.3 Crystal and Ceramic Resonator Oscillators

The ST7 internal clock can be supplied with four different Crystal/Ceramic resonator oscillators. All the information given in this paragraph is 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...). [1][2]

Symbol	Parameter	Conditions	Min	Max	Unit
f_{OSC}	Oscillator Frequency[3]	LP: Low power oscillator MP: Medium power oscillator MS: Medium speed oscillator HS: High speed oscillator	1 >2 >4 >8	2 4 8 16	MHz
R_F	Feedback resistor		20	40	kΩ
C_{L1} C_{L2}	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator (R_S)	R_S = 200Ω LP oscillator R_S = 200Ω MP oscillator R_S = 200Ω MS oscillator R_S = 100Ω HS oscillator	22 22 18 15	56 46 33 33	pF

Symbol	Parameter	Conditions	Typ	Max	Unit
i_2	OSC2 driving current	V_{DD} = 5V LP oscillator V_{IN} = V_{SS} MP oscillator MS oscillator HS oscillator	80 160 310 610	150 250 460 910	µA

Figure 100. Typical Application with a Crystal or Ceramic Resonator



Notes:

1. Resonator characteristics given by the crystal/ceramic resonator manufacturer.
2. $t_{SU(OSC)}$ is the typical oscillator start-up time measured between V_{DD} = 2.8V and the fetch of the first instruction (with a quick V_{DD} ramp-up from 0 to 5V (< 50µs).
3. The oscillator selection can be optimized in terms of supply current using an high quality resonator with small R_S value. Refer to crystal/ceramic resonator manufacturer for more details.