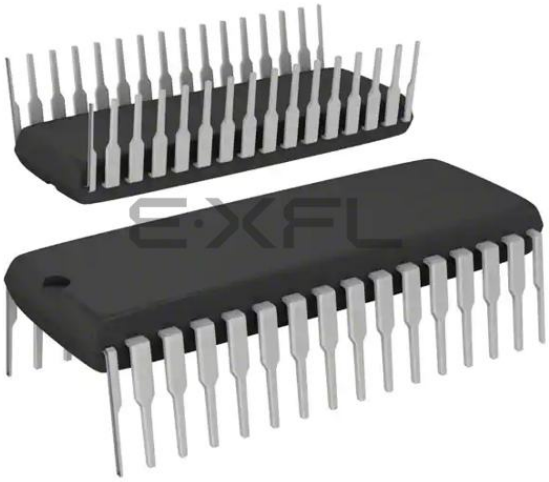


Welcome to E-XFL.COM

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	24
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	32-SDIP (0.400", 10.16mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7foxk1b6

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

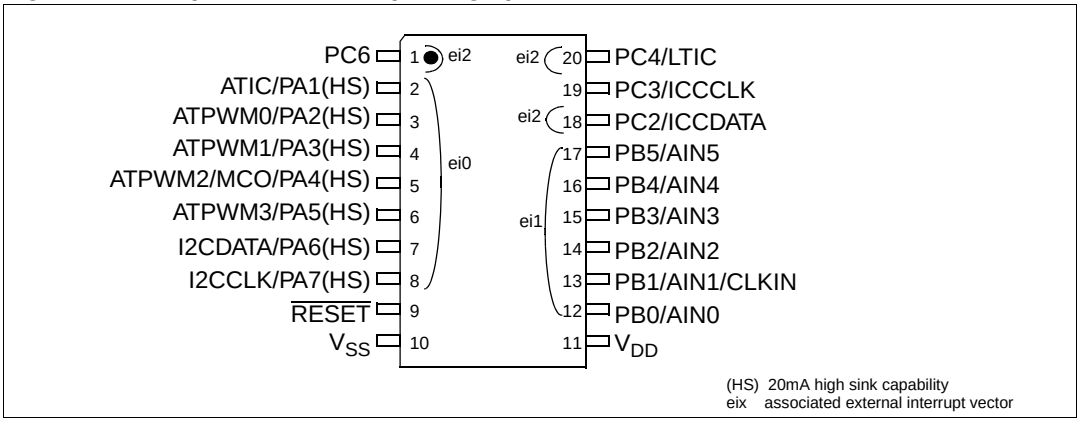
Pin number		Pin name	Type	Level		Port/control						Main function (after reset)	Alternate function
						Input				Output			
LQFP32	SDIP32			Input	Output	float	wpu	int	ana	OD(1)	PP		
14	18	PB0/AIN0	I/O	C_T		**x**	ei1		x	x	x	**Port B0**	AIN0
15	19	PB1/AIN1/CLKIN	I/O	C_T		**x**			x	x	x	**Port B1**	AIN1/External clock source
16	20	PB2/AIN2	I/O	C_T		**x**			x	x	x	**Port B2**	AIN2
17	21	PB3/AIN3/MOSI(2)	I/O	C_T		**x**			x	x	x	**Port B3**	AIN3/SPI Master out /Slave in data
18	22	PB4/AIN4/MISO(2)	I/O	C_T		**x**			x	x	x	**Port B4**	AIN4/SPI Master in/Slave out data
19	23	PB5/AIN5/ EXTCLK_A(2)	I/O	C_T		**x**			x	x	x	**Port B5**	AIN5/Timer A input clock
20	24	PB6/AIN6/SCK(2)	I/O	C_T		**x**			x	x	x	**Port B6**	AIN6/SPI serial clock
21	25	PB7/AIN7/$\overline{SS}$(2)/ OCMP2_A(2)	I/O	C_T		**x**			x	x	x	**Port B7**	AIN7/SPI slave select (active low)/ Timer A Output Compare 2
22	26	PC0/AIN8/ ICAP1_A(2)	I/O	C_T		**x**	ei2		x	x	x	**Port C0**	AIN8/Timer A Input Capture 1
23	27	PC1/AIN9/ ICAP2_A(2)	I/O	C_T		**x**			x	x	x	**Port C1**	AIN9/Timer A Input Capture 2
24	28	PC2/ICCDATA	I/O	C_T		**x**				x	x	**Port C2**	ICCDATA
25	29	PC3/ICCCLK	I/O	C_T		x	**x**			x	x	**Port C3**	ICCCLK
26	30	PC4/LTIC	I/O	C_T		**x**	ei2			x	x	**Port C4**	LTIC
27	31	PC5/BREAK2(4)	I/O	C_T		**x**				x	x	**Port C5**	BREAK2
28	32	PC6	I/O	C_T		**x**				x	x	**Port C6**	
29	1	PC7/BREAK1	I/O	C_T		**x**				x	x	**Port C7**	BREAK1

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.

Case 2 Switching from AWU RC to internal RC

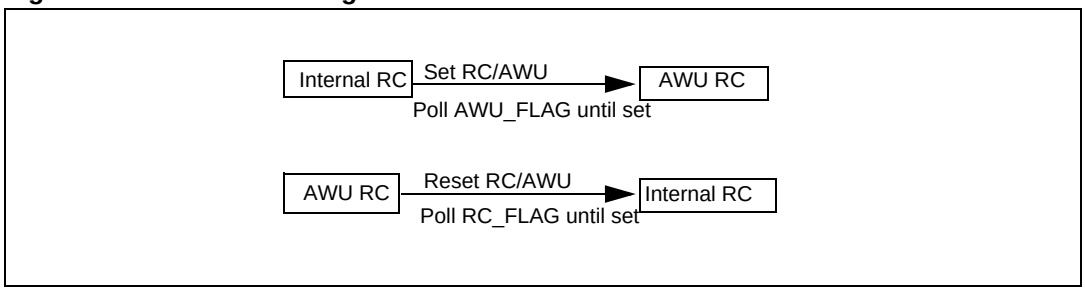
1. Reset the RC/AWU bit to enable the internal RC oscillator
2. Using a 4-bit counter, wait until 8 internal RC cycles have elapsed. The counter is running on internal RC clock.
3. Wait till the AWU_FLAG is cleared (1AWU RC cycle) and the RC_FLAG is set (2 RC cycles)
4. The switch to the internal RC 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 11. Clock switching



9.2.4 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.

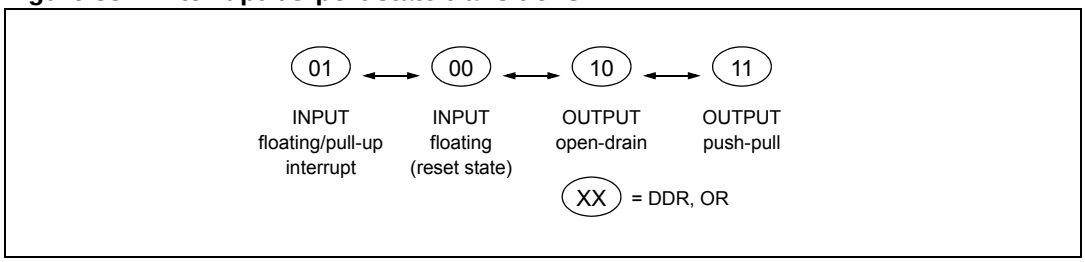
Caution: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

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 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 33*. Other transitions are potentially risky and should be avoided, since they may present unwanted side-effects such as spurious interrupt generation.

Figure 33. Interrupt I/O port state transitions



9.4 Unused I/O pins

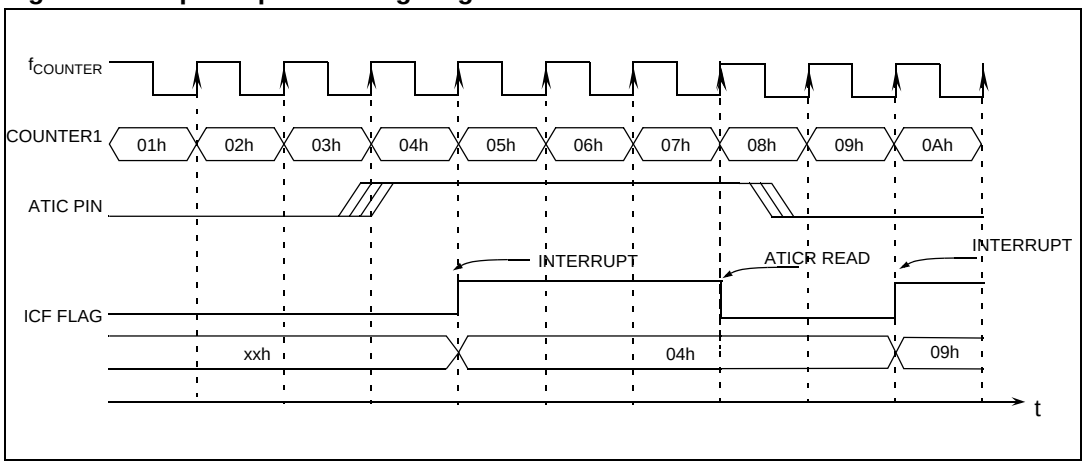
Unused I/O pins must be connected to fixed voltage levels. Refer to *Section 12.9: I/O port pin characteristics*.

9.5 Low power modes

Table 26. 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.

Figure 45. Input capture timing diagram



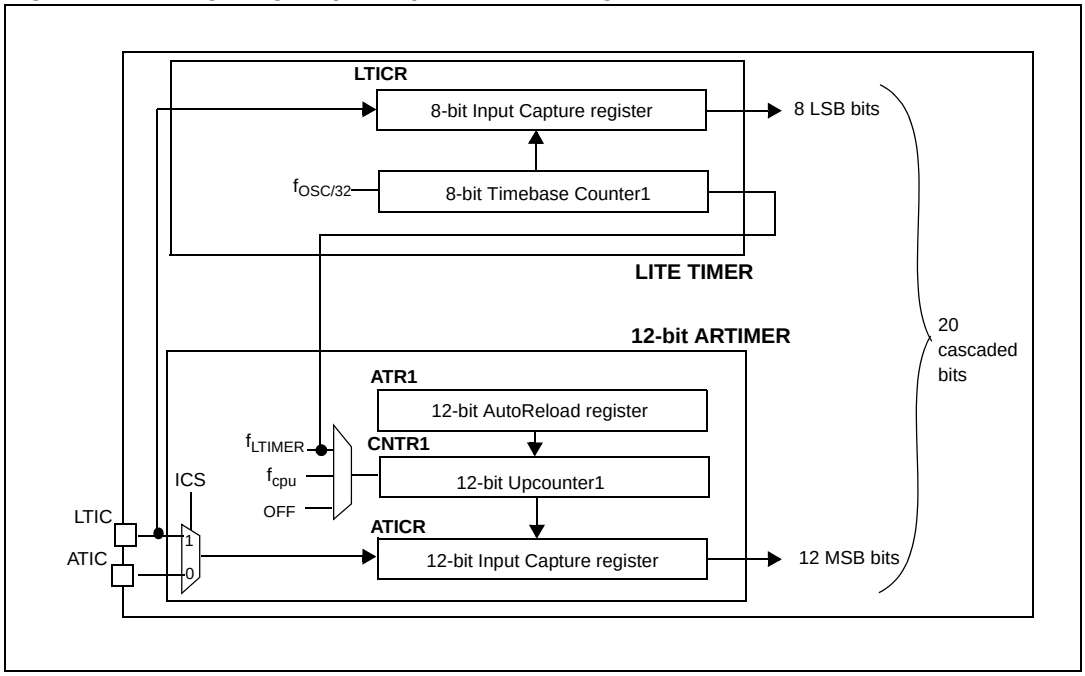
Long range input capture

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



PWMx Duty Cycle register High (DCRxH)

Reset value: 0000 0000 (00h)

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

Bits 15:12 = Reserved.

PWMx Duty Cycle register Low (DCRxL)

Reset value: 0000 0000 (00h)

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

Bits 11:0 = **DCRx[11:0]** *PWMx Duty Cycle Value:* this 12-bit value is written by software. It defines the duty cycle of the corresponding PWM output signal (see *Figure 38*).

In PWM mode (OEx=1 in the PWMCR register) the DCR[11:0] bits define the duty cycle of the PWMx output signal (see *Figure 38*). In Output Compare mode, they define the value to be compared with the 12-bit upcounter value.

Input Capture register High (ATICRH)

Reset value: 0000 0000 (00h)

15							8
0	0	0	0	ICR11	ICR10	ICR9	ICR8
Read only							

Bits 15:12 = Reserved.

the user toggle the output pin and if the ICIE bit is set. This can be avoided if the input capture function i is disabled by reading the ICiHR (see note 1).

6 The TOF bit can be used with interrupt in order to measure event that go beyond the timer range (FFFFh).

Figure 60. Input capture block diagram

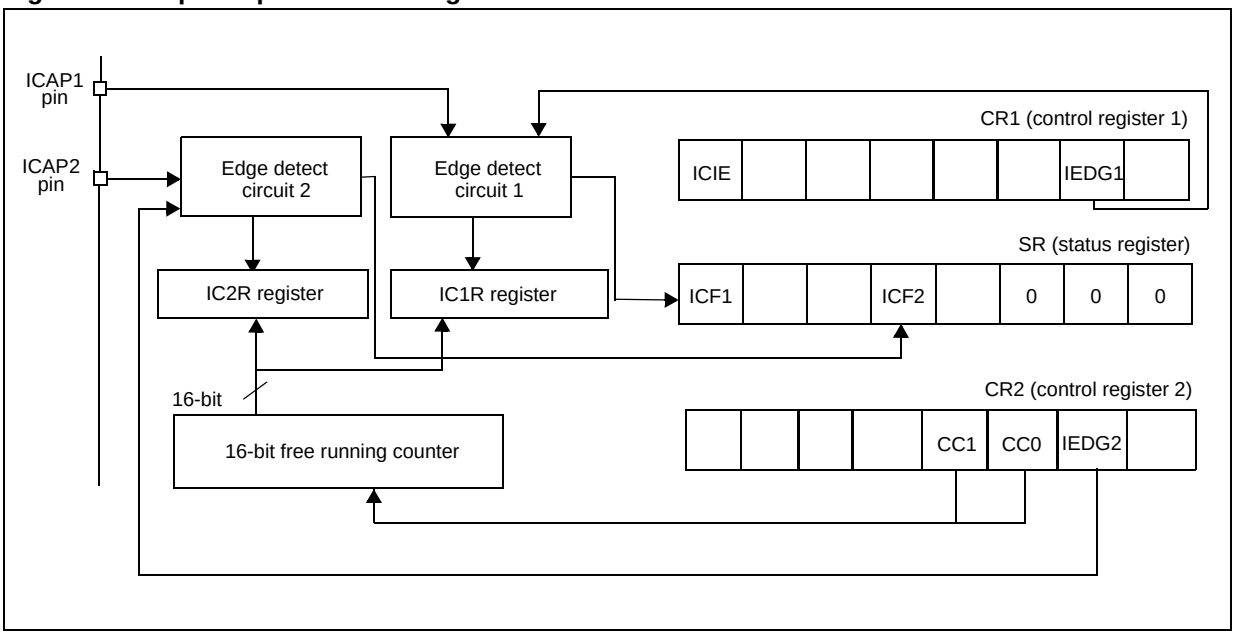
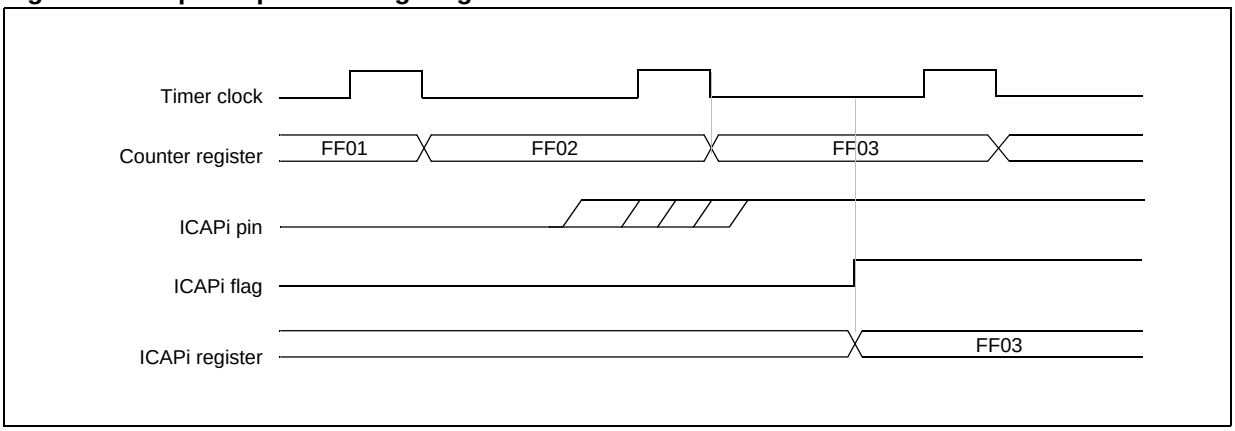


Figure 61. Input capture timing diagram



1. The active edge is the rising edge.
2. The time between an event on the ICAPi pin and the appearance of the corresponding flag is from 2 to 3 CPU clock cycles. This depends on the moment when the ICAP event happens relative to the timer clock.

The OC*i*R register value required for a specific timing application can be calculated using the following formula:

Equation 1

$$\Delta \text{ OC}i\text{R} = \frac{\Delta t * f_{CPU}}{PRESC}$$

Where:

Δt = output compare period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

PRESC = timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see *: Timer A Control register 2 (TACR2) on page 134*)

If the timer clock is an external clock, the formula is:

Equation 2

$$\Delta \text{ OC}i\text{R} = \Delta t * f_{EXT}$$

Where:

Δt = output compare period (in seconds)

f_{EXT} = external timer clock frequency (in hertz)

Clearing the output compare interrupt request (that is, clearing the OCF*i* bit) is done by:

1. Reading the SR register while the OCF*i* bit is set.
2. Accessing (reading or writing) the OC*i*LR register.

The following procedure is recommended to prevent the OCF*i* bit from being set between the time it is read and the time it is written to the OC*i*R register:

- Write to the OC*i*HR register (further compares are inhibited).
- Read the SR register (first step in the clearance of the OCF*i* bit, which may be already set).
- Write to the OC*i*LR register (enables the output compare function and clears the OCF*i* bit).

Note: 1 After a processor write cycle to the OCiHR register, the output compare function is inhibited until the OCiLR register is also written.

2 If the OCiE bit is not set, the OCMPi pin is a general I/O port and the OLVLi bit does not appear when a match is found but an interrupt could be generated if the OCIE bit is set.

3 In both internal and external clock modes, OCFi and OCMPi are set while the counter value equals the OCiR register value (see Figure 63 for an example with $f_{CPU}/2$ and Figure 64 for an example with $f_{CPU}/4$). This behavior is the same in OPM or PWM mode.

4 The output compare functions can be used both for generating external events on the OCMPi pins even if the input capture mode is also used.

5 The value in the 16-bit OCiR register and the OLVi bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout.

10.5 I²C bus interface (I²C)

10.5.1 Introduction

The I²C Bus Interface serves as an interface between the microcontroller and the serial I²C bus. It provides both multimaster and slave functions, and controls all I²C bus-specific sequencing, protocol, arbitration and timing. It supports fast I²C mode (400 kHz).

10.5.2 Main features

- Parallel-bus/I²C protocol converter
- Multi-master capability
- 7-bit/10-bit Addressing
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection

I²C master features:

- Clock generation
- I²C bus busy flag
- Arbitration Lost Flag
- End of byte transmission flag
- Transmitter/Receiver Flag
- Start bit detection flag
- Start and Stop generation

I²C slave features:

- Stop bit detection
- I²C bus busy flag
- Detection of misplaced start or stop condition
- Programmable I²C Address detection
- Transfer problem detection
- End-of-byte transmission flag
- Transmitter/Receiver flag

10.5.3 General description

In addition to receiving and transmitting data, this interface converts it from serial to parallel format and vice versa, using either an interrupt or polled handshake. The interrupts are enabled or disabled by software. The interface is connected to the I²C bus by a data pin (SDAI) and by a clock pin (SCLI). It can be connected both with a standard I²C bus and a Fast I²C bus. This selection is made by software.

subsequent EV4 is not seen.

6. **EV4:** EVF=1, STOPF=1, cleared by reading SR2 register.
7. **EV5:** EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.
8. **EV6:** EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).
9. **EV7:** EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.
10. **EV8:** EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.
11. **EV9:** EVF=1, ADD10=1, cleared by reading SR1 register followed by writing DR register.

10.5.5 Low power modes

Table 46. Effect of low power modes on the I²C interface

Mode	Description
Wait	No effect on I²C interface. I²C interrupts cause the device to exit from Wait mode.
Halt	I²C registers are frozen. In Halt mode, the I²C interface is inactive and does not acknowledge data on the bus. The I²C interface resumes operation when the MCU is woken up by an interrupt with “exit from Halt mode” capability.

10.5.6 Interrupts

Figure 72. Event flags and interrupt generation

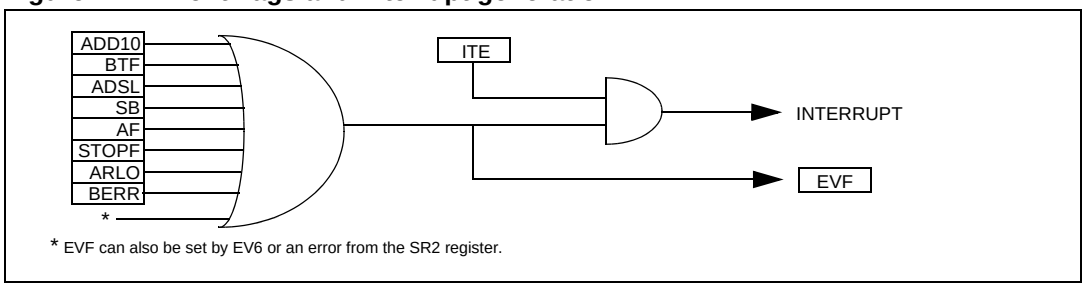


Table 47. Description of interrupt events

Interrupt Event(1)	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
10-bit Address Sent Event (Master mode)	ADD10	ITE	Yes	No
End of byte Transfer Event	BTF		Yes	No
Address Matched Event (Slave mode)	ADSL		Yes	No
Start Bit Generation Event (Master mode)	SB		Yes	No
Acknowledge Failure Event	AF		Yes	No
Stop Detection Event (Slave mode)	STOPF		Yes	No
Arbitration Lost Event (Multimaster configuration)	ARLO		Yes	No
Bus Error Event	BERR		Yes	No

1. The I²C 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 I-bit in the CC register is reset (RIM instruction).

Master mode transmit sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if the SPIE bit is set and the interrupt mask in the CCR register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPICSR register while the SPIF bit is set
2. A read to the SPIDR register

Note: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Slave mode operation

In slave mode, the serial clock is received on the SCK pin from the master device.

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see *Figure 77*).

Note: The slave must have the same CPOL and CPHA settings as the master.

   - Manage the $\overline{SS}$ pin as described in *Section : Slave select management* and *Figure 75*. If CPHA = 1 $\overline{SS}$ must be held low continuously. If CPHA = 0 $\overline{SS}$ must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

Slave mode transmit sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

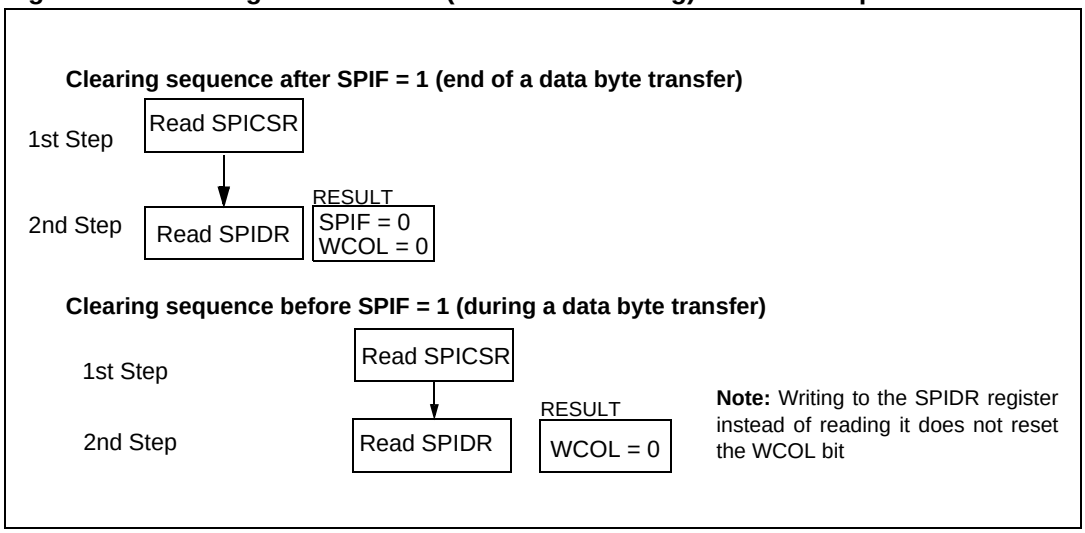
- The SPIF bit is set by hardware.
- An interrupt request is generated if SPIE bit is set and interrupt mask in the CCR register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPICSR register while the SPIF bit is set
2. A write or a read to the SPIDR register

Note: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Figure 78. Clearing the WCOL bit (write collision flag) software sequence



Single master and multimaster configurations

There are two types of SPI systems:

- Single Master System
- Multimaster System

Single Master System

A typical single master system may be configured using a device as the master and four devices as slaves (see *Figure 79*).

The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices.

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

Note: To prevent a bus conflict on the MISO line, the master allows only one active slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written to its SPIDR register.

Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Multimaster system

A multimaster system may also be configured by the user. Transfer of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system.

The multimaster system is principally handled by the MSTR bit in the SPICR register and the MODF bit in the SPICSR register.

Table 53. SPI register map and reset values

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

11.2.1 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 65. 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 ?								

12.4.2 On-chip peripherals

Table 73. On-chip peripheral characteristics

Symbol	Parameter	Conditions		Typ	Unit
$I_{DD(SPI)}$	SPI supply current(1)	f_{CPU}=8 MHz	V_{DD}=5.0 V	200	µA
$I_{DD(AT)}$	12-bit Auto-Reload timer supply current(2)	f_{CPU}=8 MHz	V_{DD}=5.0 V	50	µA
$I_{DD(I2C)}$	I^2C supply current(3)	f_{CPU}=8 MHz	V_{DD}=5.0 V	1000	µA
$I_{DD(ADC)}$	ADC supply current when converting(4)	f_{ADC}=4 MHz	V_{DD}=5.0 V	600	µA

1. Data based on a differential I_{DD} measurement between reset configuration and a permanent SPI master communication (data sent equal to 55h).
2. 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.
3. Data based on a differential I_{DD} measurement between reset configuration (I^2C disabled) and a permanent I^2C master communication at 100 kHz (data sent equal to 55h). This measurement include the pad toggling consumption (4.7 kOhm external pull-up on clock and data lines).
4. Data based on a differential I_{DD} measurement between reset configuration and continuous A/D conversions.

12.8.3 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 85. 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 86. Electrical sensitivities

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

12.9.2 Output driving current

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

Table 88. 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		1.0	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 67.* 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 67.* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}.

Table 96. 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

14.1 Thermal characteristics

Table 101. Thermal characteristics

Symbol	Ratings		Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient)	LQFP32 SDIP32 SO20 DIP20	55 58 76 63	°C/W
T_{Jmax}	Maximum junction temperature(1)		150	°C
P_{Dmax}	Power dissipation(2)		160	mW

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 ($I_{DD}xV_{DD}$) and P_{PORT} is the port power dissipation depending on the ports used in the application.