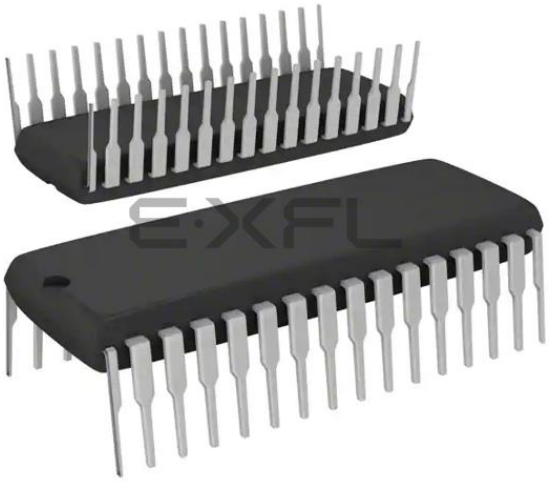


Welcome to E-XFL.COM

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	16MHz
Connectivity	SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	22
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 6x10b
Oscillator Type	Internal
Operating Temperature	-10°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/st72f262g2b5

To obtain the most recent version of this datasheet,
please check at www.st.com>products>technical literature>datasheet
Please note that the list of known limitations can be found at the end of this document on page 168.

1 INTRODUCTION

The ST72260Gx, ST72262Gx and ST72264Gx devices are members of the ST7 microcontroller family. They can be grouped as follows :

– ST72264Gx devices are designed for mid-range applications with ADC, I^2C and SCI interface capabilities.

– ST72262Gx devices target the same range of applications but without I^2C interface or SCI.

– ST72260Gx devices are for applications that do not need ADC, I^2C peripherals or SCI.

All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

The ST72F260G, ST72F262G, and ST72F264G versions feature single-voltage FLASH memory with byte-by-byte In-Circuit Programming (ICP) capabilities.

Under software control, all devices can be placed in WAIT, SLOW, Active-HALT or HALT mode, reducing power consumption when the application is in idle or stand-by state.

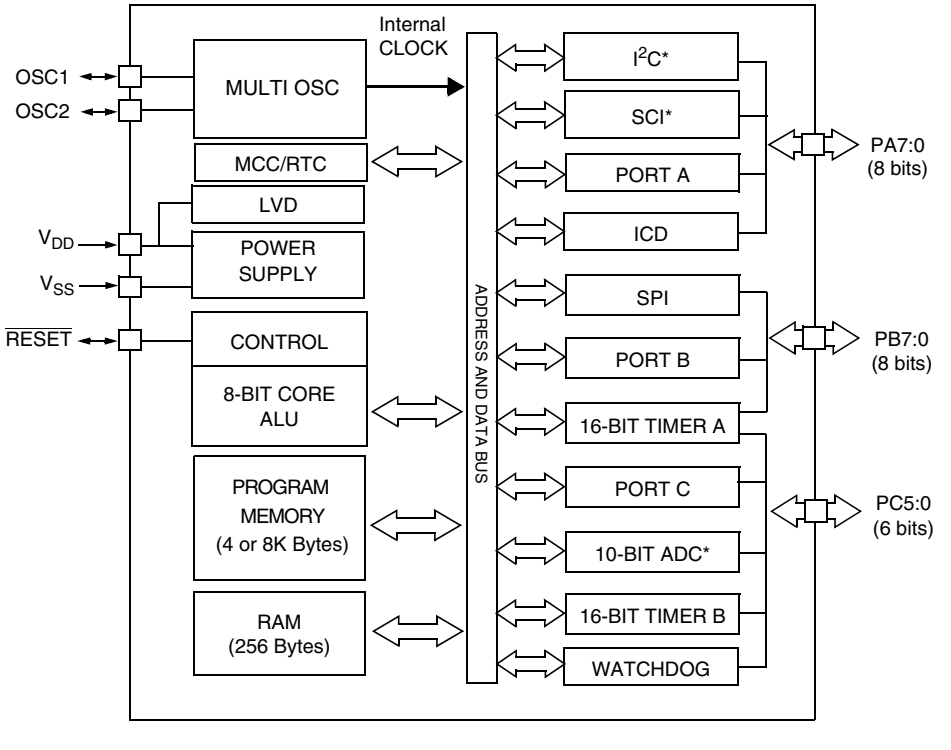
The enhanced instruction set and addressing modes of the ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned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

For easy reference, all parametric data is located in Section 13 on page 126.

Related Documentation

AN1365: Guidelines for migrating ST72C254 applications to ST72F264

Figure 1. General Block Diagram



*Not available on some devices, see device summary on page 1.

FLASH PROGRAM MEMORY (Cont'd)

4.4 ICC interface

ICP needs a minimum of 4 and up to 7 pins to be connected to the programming tool. These pins are:

- $\overline{\text{RESET}}$: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- ICCSEL: ICC selection (not required on devices without ICCSEL pin)
- OSC1: main clock input for external source (not required on devices without OSC1/OSC2 pins)
- V_{DD}: application board power supply (optional, see Note 3)

Notes:

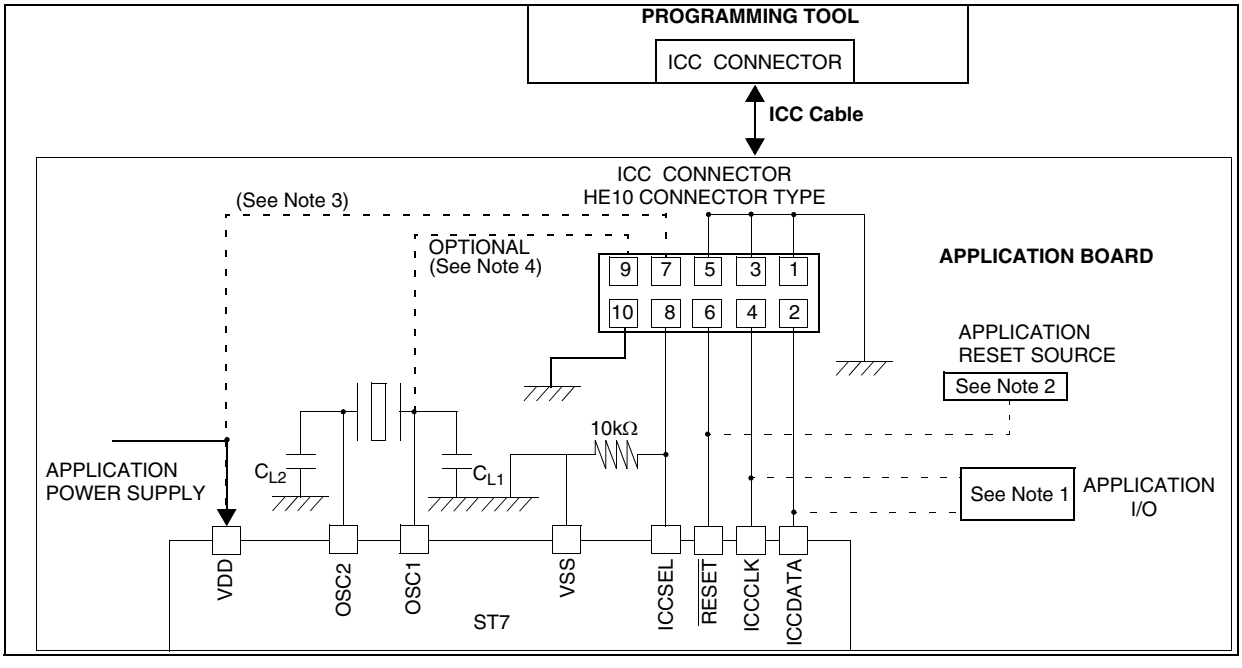
1. If the ICCCLK or ICCDATA pins are only used as outputs in the application, no signal isolation is necessary. As soon as the Programming Tool is plugged to the board, even if an ICC session is not in progress, the ICCCLK and ICCDATA pins are not available for the application. If they are used as inputs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

2. During the ICP session, the programming tool must control the $\overline{\text{RESET}}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor<1K). A schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with R>1K or a reset management IC with open drain output and pull-up resistor>1K, no additional components are needed. In all cases the user must ensure that no external reset is generated by the application during the ICC session.

3. The use of Pin 7 of the ICC connector depends on the Programming Tool architecture. This pin must be connected when using most ST Programming Tools (it is used to monitor the application power supply). Please refer to the Programming Tool manual.

4. Pin 9 has to be connected to the OSC1 pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte. ST7 devices with multi-oscillator capability need to have OSC2 grounded in this case.

Figure 6. Typical ICC Interface



6.3 RESET SEQUENCE MANAGER (RSM)

6.3.1 Introduction

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

- External $\overline{\text{RESET}}$ source pulse
- Internal LVD RESET (Low Voltage Detection)
- Internal WATCHDOG RESET

These sources act on the $\overline{\text{RESET}}$ pin and it is always kept low during the delay phase.

The RESET service routine vector is fixed at addresses FFFEh-FFFFh in the ST7 memory map.

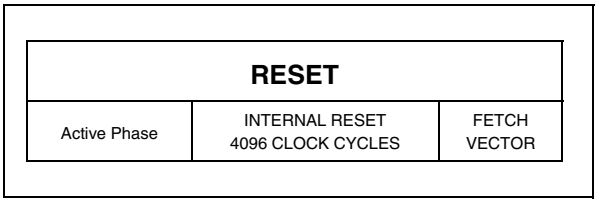
The basic RESET sequence consists of 3 phases as shown in Figure 11:

- Active Phase depending on the RESET source
- 4096 CPU clock cycle delay (selected by option byte)
- RESET vector fetch

The 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay should be selected by option byte to correspond to the stabilization time of the external oscillator used in the application.

The RESET vector fetch phase duration is 2 clock cycles.

Figure 11. RESET Sequence Phases

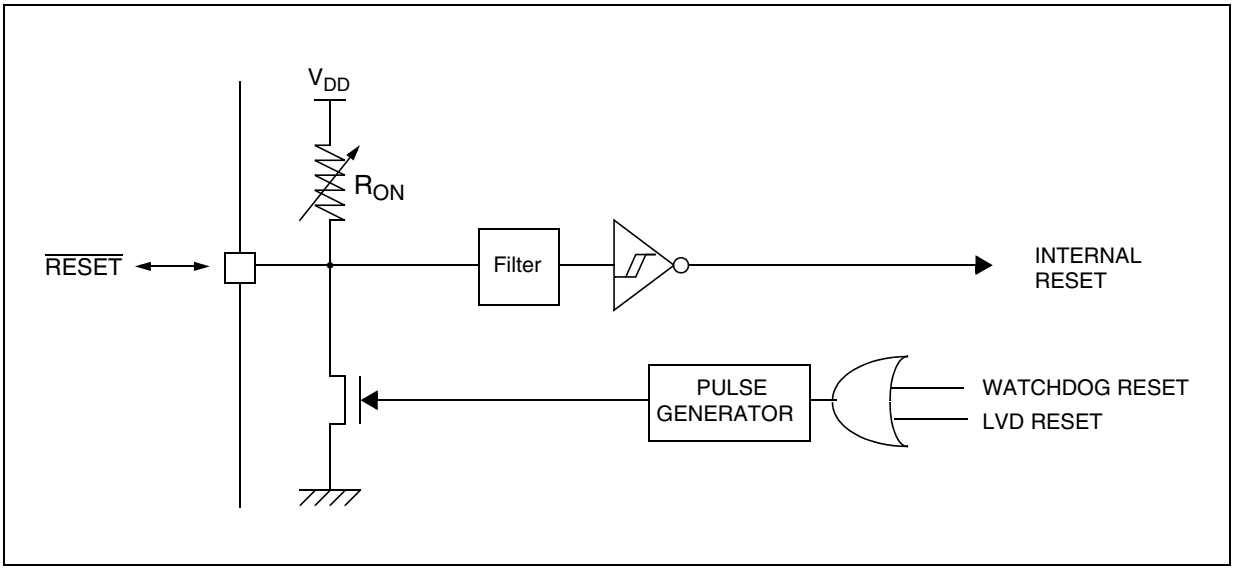


6.3.2 Asynchronous External $\overline{\text{RESET}}$ pin

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

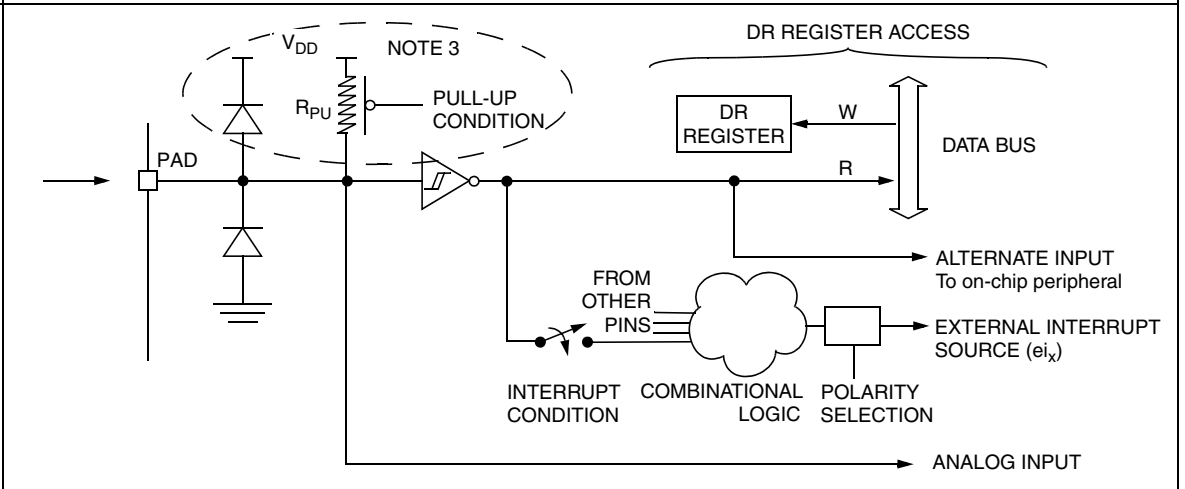
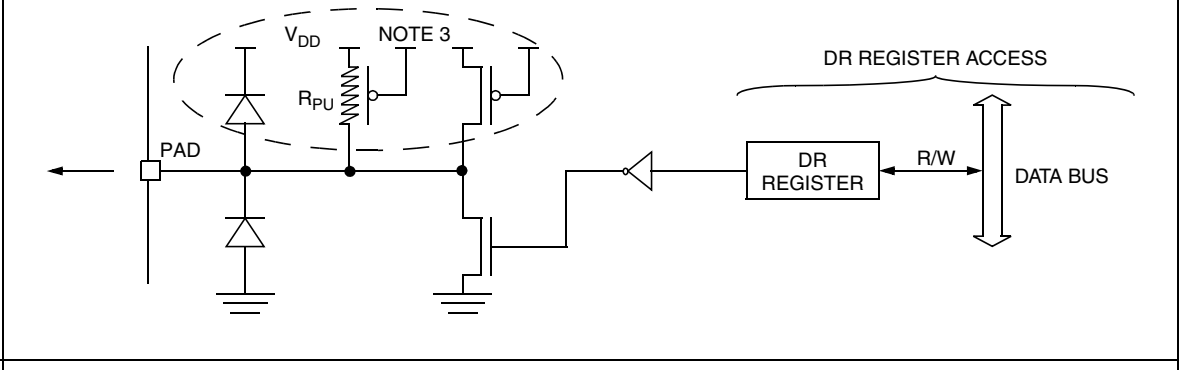
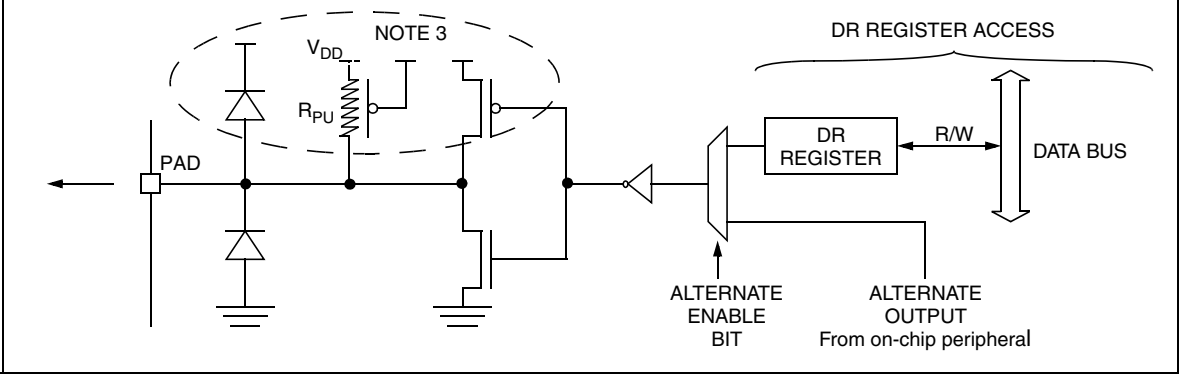
A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see Figure 13). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

Figure 12. Reset Block Diagram



I/O PORTS (Cont'd)

Table 8. I/O Configurations

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

Notes:
1. When the I/O port is in input configuration and the associated alternate function is enabled as an output, reading the DR register will read the alternate function output status.
2. When the I/O port is in output configuration and the associated alternate function is enabled as an input, the alternate function reads the pin status given by the DR register content.
3. For true open drain, these elements are not implemented.

I/O PORTS (Cont'd)

9.7 DEVICE-SPECIFIC I/O PORT CONFIGURATION

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

Interrupt Ports

PA7, PA5, PA3:0, PB7:0, PC5:0 (with pull-up)

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

True Open Drain Interrupt Ports

PA6, PA4 (without pull-up)

MODE	DDR	OR
floating input	0	0
floating interrupt input	0	1
open drain (high sink ports)	1	X

Table 9. Port Configuration

Port	Pin Name	Input (DDR = 0)		Output (DDR = 1)		
		OR = 0	OR = 1	OR = 0	OR = 1	High-Sink
Port A	PA7	floating	pull-up interrupt	open drain	push-pull	Yes
	PA6	floating	floating interrupt	true open-drain		
	PA5	floating	pull-up interrupt	open drain	push-pull	
	PA4	floating	floating interrupt	true open-drain		
	PA3:0	floating	pull-up interrupt	open drain	push-pull	
Port B	PB7:0	floating	pull-up interrupt	open drain	push-pull	No
Port C	PC5:0	floating	pull-up interrupt	open drain	push-pull	

10 MISCELLANEOUS REGISTERS

The miscellaneous registers allow control over several different features such as the external interrupts or the I/O alternate functions.

10.1 I/O PORT INTERRUPT SENSITIVITY

The external interrupt sensitivity is controlled by the ISxx bits of the Miscellaneous register and the OPTION BYTE. This control allows you to have two fully independent external interrupt source sensitivities with configurable sources (using the EXTIT option bit) as shown in Figure 29 and Figure 30.

Each external interrupt source can be generated on four different events on the pin:

- Falling edge
- Rising edge
- Falling and rising edge
- Falling edge and low level

To guarantee correct functionality, the sensitivity bits in the MISCR1 register must be modified only when the I[1:0] bits in the CC register are set to 1 (interrupt masked). See Section 9.8 "I/O PORT REGISTER DESCRIPTION" on page 43 and Section 10.3 "MISCELLANEOUS REGISTER DESCRIPTION" on page 46 for more details on the programming.

10.2 I/O PORT ALTERNATE FUNCTIONS

The MISCR registers manage four I/O port miscellaneous alternate functions:

- Main clock signal (f_{CPU}) output on PC2
- SPI pin configuration:
  - $\overline{SS}$ pin internal control to use the PB7 I/O port function while the SPI is active.
  - Master output capability on the MOSI pin (PB4) deactivated while the SPI is active.
  - Slave output capability on the MISO pin (PB5) deactivated while the SPI is active.

These functions are described in detail in the Section 10.3 "MISCELLANEOUS REGISTER DESCRIPTION" on page 46.

Figure 29. Ext. Interrupt Sensitivity (EXTIT=0)

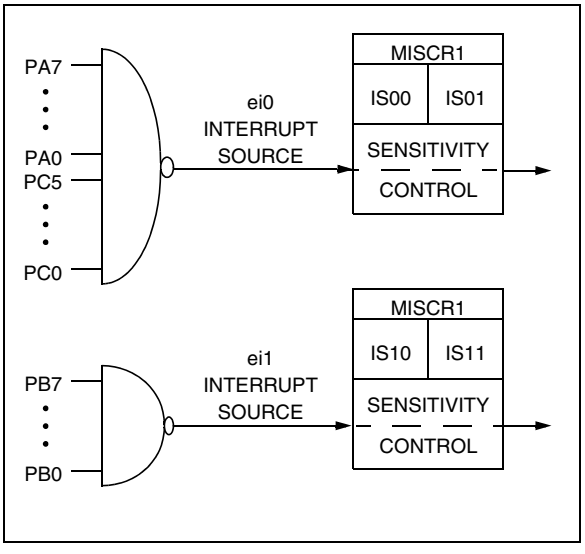
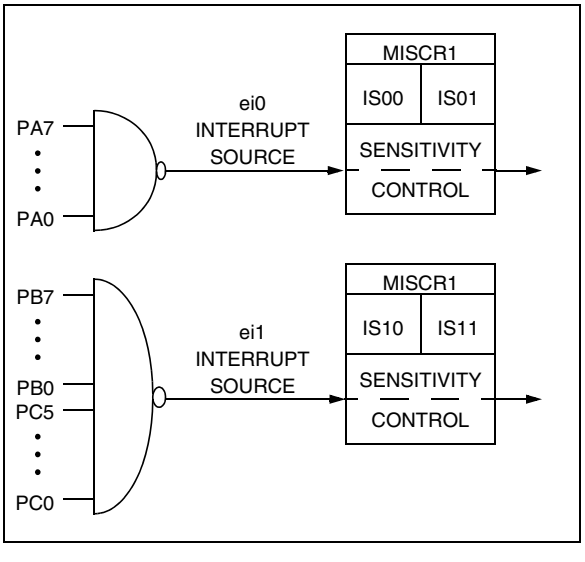


Figure 30. Ext. Interrupt Sensitivity (EXTIT=1)



WATCHDOG TIMER (Cont'd)

Figure 33. Exact Timeout Duration (t_{min} and t_{max})

WHERE:

$t_{min0} = (LSB + 128) \times 64 \times t_{OSC2}$

$t_{max0} = 16384 \times t_{OSC2}$

$t_{OSC2} = 125ns$ if $f_{OSC2}=8$ MHz

CNT = Value of T[5:0] bits in the WDGCR register (6 bits)
MSB and LSB are values from the table below depending on the timebase selected by the TB[1:0] bits in the MCCSR register

TB1 Bit (MCCSR Reg.)	TB0 Bit (MCCSR Reg.)	Selected MCCSR Timebase	MSB	LSB
0	0	2ms	4	59
0	1	4ms	8	53
1	0	10ms	20	35
1	1	25ms	49	54

To calculate the minimum Watchdog Timeout (t_{min}):

IF $CNT < \left[\frac{MSB}{4}\right]$ **THEN** $t_{min} = t_{min0} + 16384 \times CNT \times t_{osc2}$

ELSE $t_{min} = t_{min0} + \left[16384 \times \left(CNT - \left[\frac{4CNT}{MSB}\right]\right) + (192 + LSB) \times 64 \times \left[\frac{4CNT}{MSB}\right]\right] \times t_{osc2}$

To calculate the maximum Watchdog Timeout (t_{max}):

IF $CNT \leq \left[\frac{MSB}{4}\right]$ **THEN** $t_{max} = t_{max0} + 16384 \times CNT \times t_{osc2}$

ELSE $t_{max} = t_{max0} + \left[16384 \times \left(CNT - \left[\frac{4CNT}{MSB}\right]\right) + (192 + LSB) \times 64 \times \left[\frac{4CNT}{MSB}\right]\right] \times t_{osc2}$

Note: In the above formulae, division results must be rounded down to the next integer value.

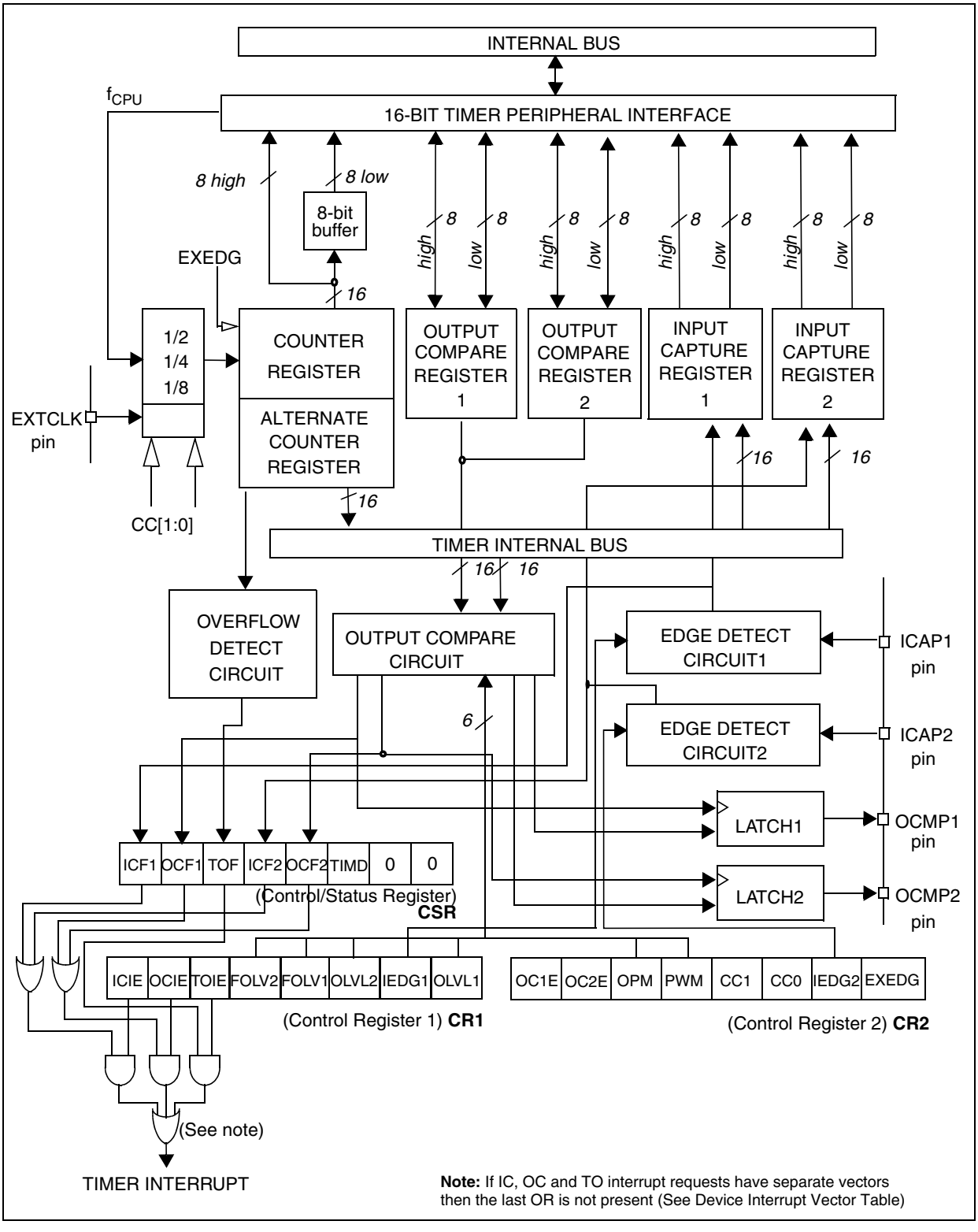
Example:

With 2ms timeout selected in MCCSR register

Value of T[5:0] Bits in WDGCR Register (Hex.)	Min. Watchdog Timeout (ms) t_{min}	Max. Watchdog Timeout (ms) t_{max}
00	1.496	2.048
3F	128	128.552

16-BIT TIMER (Cont'd)

Figure 35. Timer Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

11.4.3.2 Slave Select Management

As an alternative to using the $\overline{SS}$ pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 49)

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

In Master mode:

– $\overline{SS}$ internal must be held high continuously

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 48):

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

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

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

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

Figure 48. Generic $\overline{SS}$ Timing Diagram

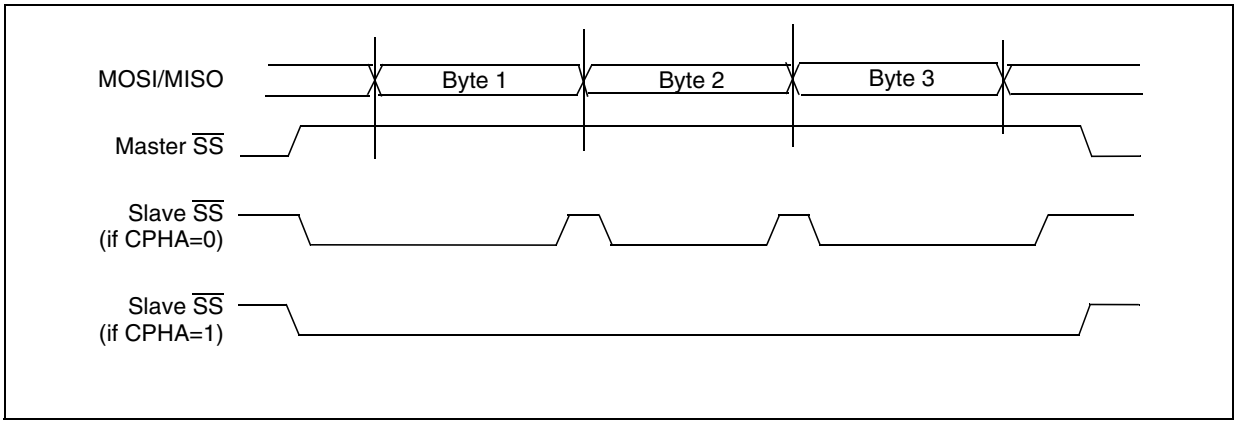
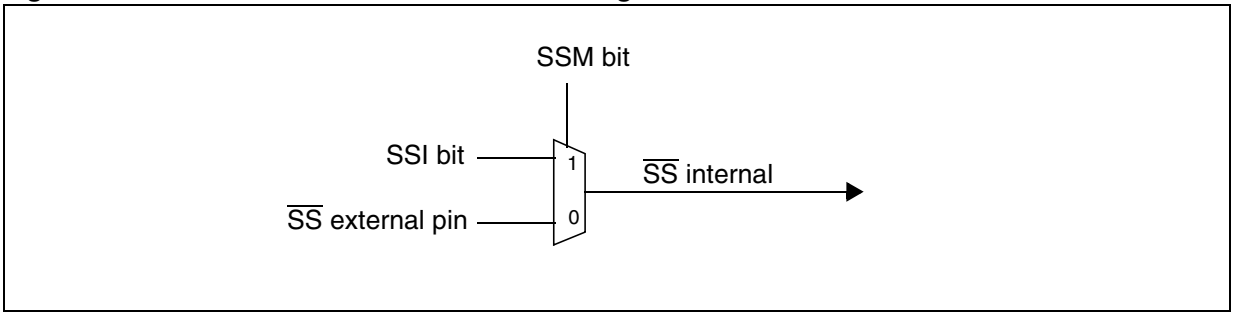


Figure 49. Hardware/Software Slave Select Management



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

11.5.4.3 Receiver

The SCI can receive data words of either 8 or 9 bits. When the M bit is set, word length is 9 bits and the MSB is stored in the R8 bit in the SCICR1 register.

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists or a buffer (RDR) between the internal bus and the received shift register (see Figure 53).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the SCIBRR and the SCIERPR registers.

– Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

– The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR.

– An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

– The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

Clearing the RDRF bit is performed by the following software sequence done by:

1. An access to the SCISR register

2. A read to the SCIDR register.

The RDRF bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Break Character

When a break character is received, the SPI handles it as a framing error.

Idle Character

When a idle frame is detected, there is the same procedure as a data received character plus an interrupt if the ILIE bit is set and the I bit is cleared in the CCR register.

Overrun Error

An overrun error occurs when a character is received when RDRF has not been reset. Data can not be transferred from the shift register to the RDR register as long as the RDRF bit is not cleared.

When a overrun error occurs:

– The OR bit is set.

– The RDR content will not be lost.

– The shift register will be overwritten.

– An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

The OR bit is reset by an access to the SCISR register followed by a SCIDR register read operation.

Noise Error

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise. Normal data bits are considered valid if three consecutive samples (8th, 9th, 10th) have the same bit value, otherwise the NF flag is set. In the case of start bit detection, the NF flag is set on the basis of an algorithm combining both valid edge detection and three samples (8th, 9th, 10th). Therefore, to prevent the NF flag getting set during start bit reception, there should be a valid edge detection as well as three valid samples.

When noise is detected in a frame:

– The NF flag is set at the rising edge of the RDRF bit.

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The NF flag is reset by a SCISR register read operation followed by a SCIDR register read operation.

During reception, if a false start bit is detected (e.g. 8th, 9th, 10th samples are 011,101,110), the frame is discarded and the receiving sequence is not started for this frame. There is no RDRF bit set for this frame and the NF flag is set internally (not accessible to the user). This NF flag is accessible along with the RDRF bit when a next valid frame is received.

Note: If the application Start Bit is not long enough to match the above requirements, then the NF Flag may get set due to the short Start Bit. In this case, the NF flag may be ignored by the application software when the first valid byte is received.

See also Section 11.5.4.10.

I²C BUS INTERFACE (Cont'd)

11.6.4 Functional Description

Refer to the CR, SR1 and SR2 registers in Section 11.6.7. for the bit definitions.

By default the I²C interface operates in Slave mode (M/SL bit is cleared) except when it initiates a transmit or receive sequence.

First the interface frequency must be configured using the FRi bits in the OAR2 register.

11.6.4.1 Slave Mode

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register; then it is compared with the address of the interface or the General Call address (if selected by software).

Note: In 10-bit addressing mode, the comparison includes the header sequence (11110xx0) and the two most significant bits of the address.

Header matched (10-bit mode only): the interface generates an acknowledge pulse if the ACK bit is set.

Address not matched: the interface ignores it and waits for another Start condition.

Address matched: the interface generates in sequence:

– Acknowledge pulse if the ACK bit is set.

– EVF and ADSL bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV1).
Next, in 7-bit mode read the DR register to determine from the least significant bit (Data Direction Bit) if the slave must enter Receiver or Transmitter mode.

In 10-bit mode, after receiving the address sequence the slave is always in receive mode. It will enter transmit mode on receiving a repeated Start condition followed by the header sequence with matching address bits and the least significant bit set (11110xx1).

Slave Receiver

Following the address reception and after SR1 register has been read, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

– Acknowledge pulse if the ACK bit is set

– EVF and BTF bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV2).

Slave Transmitter

Following the address reception and after SR1 register has been read, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV3).

When the acknowledge pulse is received:

– The EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Closing slave communication

After the last data byte is transferred a Stop Condition is generated by the master. The interface detects this condition and sets:

– EVF and STOPF bits with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR2 register (see Figure 59 Transfer sequencing EV4).
Error Cases

– **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and the BERR bits are set with an interrupt if the ITE bit is set.
If it is a Stop then the interface discards the data, released the lines and waits for another Start condition.
If it is a Start then the interface discards the data and waits for the next slave address on the bus.

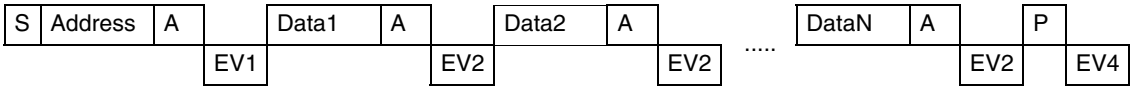
– **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set with an interrupt if the ITE bit is set.
The AF bit is cleared by reading the I2CSR2 register. However, if read before the completion of the transmission, the AF flag will be set again, thus possibly generating a new interrupt. Software must ensure either that the SCL line is back at 0 before reading the SR2 register, or be able to correctly handle a second interrupt during the 9th pulse of a transmitted byte.

Note: In case of errors, SCL line is not held low; however, the SDA line can remain low if the last bits transmitted are all 0. While AF=1, the SCL line may be held low due to SB or BTF flags that are set at the same time. It is then necessary to release both lines by software.

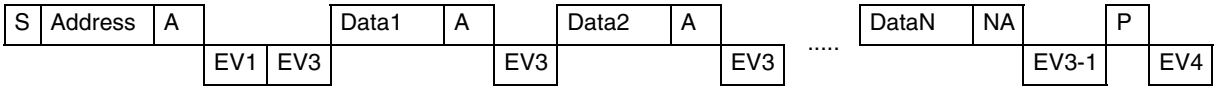
I²C BUS INTERFACE (Cont'd)

Figure 59. Transfer Sequencing

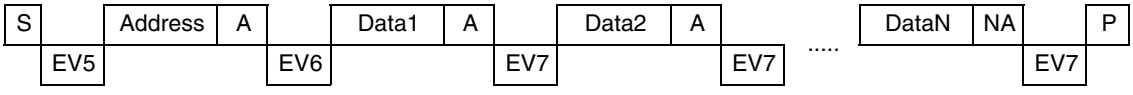
7-bit Slave receiver:



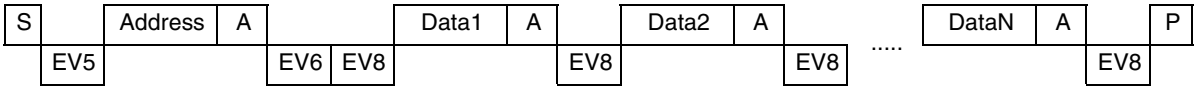
7-bit Slave transmitter:



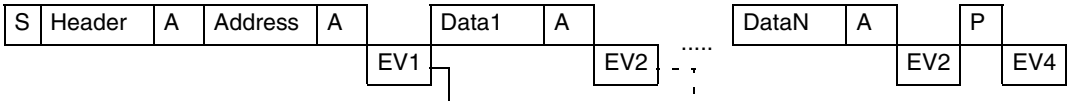
7-bit Master receiver:



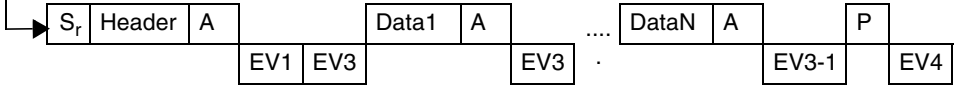
7-bit Master transmitter:



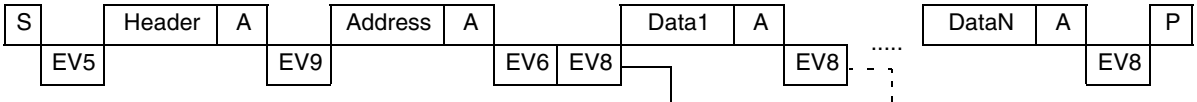
10-bit Slave receiver:



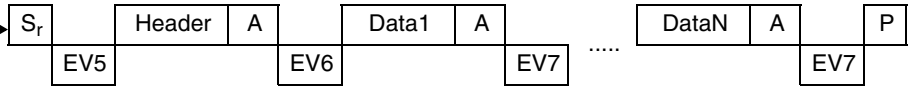
10-bit Slave transmitter:



10-bit Master transmitter



10-bit Master receiver:



Legend: S=Start, S_r = Repeated Start, P=Stop, A=Acknowledge, NA=Non-acknowledge, EVx=Event (with interrupt if ITE=1)

EV1: EVF=1, ADSL=1, cleared by reading SR1 register.

EV2: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

EV3: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

EV3-1: EVF=1, AF=1, BTF=1; AF is cleared by reading SR1 register. BTF is cleared by releasing the lines (STOP=1, STOP=0) or by writing DR register (DR=FFh). **Note:** If lines are released by STOP=1, STOP=0, the subsequent EV4 is not seen.

EV4: EVF=1, STOPF=1, cleared by reading SR2 register.

EV5: EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.

EV6: EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).

EV7: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

EV8: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

EV9: EVF=1, ADD10=1, cleared by reading SR1 register followed by writing DR register.

I²C BUS INTERFACE (Cont'd)

I²C CLOCK CONTROL REGISTER (CCR)

Read / Write
Reset Value: 0000 0000 (00h)

7							0
FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0

Bit 7 = **FM/SM** *Fast/Standard I²C mode.*
This bit is set and cleared by software. It is not cleared when the interface is disabled (PE=0).
0: Standard I²C mode
1: Fast I²C mode

Bit 6:0 = **CC[6:0]** *7-bit clock divider.*
These bits select the speed of the bus (F_{SCL}) depending on the I²C mode. They are not cleared when the interface is disabled (PE=0).

Refer to the Electrical Characteristics section for the table of values.

Note: The programmed F_{SCL} assumes no load on SCL and SDA lines.

I²C DATA REGISTER (DR)

Read / Write
Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **D[7:0]** *8-bit Data Register.*
These bits contain the byte to be received or transmitted on the bus.

– Transmitter mode: Byte transmission start automatically when the software writes in the DR register.
– Receiver mode: the first data byte is received automatically in the DR register using the least significant bit of the address.
Then, the following data bytes are received one by one after reading the DR register.

I²C BUS INTERFACE (Cont'd)

Table 21. I²C Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0028h	**I2CCR**			PE	ENGC	START	ACK	STOP	ITE
	Reset Value	0	0	0	0	0	0	0	0
0029h	**I2CSR1**	EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB
	Reset Value	0	0	0	0	0	0	0	0
002Ah	**I2CSR2**				AF	STOPF	ARLO	BERR	GCAL
	Reset Value	0	0	0	0	0	0	0	0
02Bh	**I2CCCR**	FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0
	Reset Value	0	0	0	0	0	0	0	0
02Ch	**I2COAR1**	ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
	Reset Value	0	0	0	0	0	0	0	0
002Dh	**I2COAR2**	FR1	FR0				ADD9	ADD8	
	Reset Value	0	1	0	0	0	0	0	0
002Eh	**I2CDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0

10-BIT A/D CONVERTER (ADC) (Cont'd)

11.7.3.2 Digital A/D Conversion Result

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

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

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

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

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

11.7.3.3 A/D Conversion

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the «I/O ports» chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

In the ADCCSR register:

– Select the CH[2:0] bits to assign the analog channel to convert.

ADC Conversion mode

In the ADCCSR register:

- Set the SPEED or the SLOW bits

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

When a conversion is complete:

– The EOC bit is set by hardware.

– The result is in the ADCDR registers.

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

To read the 10 bits, perform the following steps:

1. Poll EOC bit
2. Read ADCDRL. This locks the ADCDRH until it is read.
3. Read ADCDRH. This clears EOC automatically.

To read only 8 bits, perform the following steps:

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

11.7.4 Low Power Modes

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

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

11.7.5 Interrupts

None.

CLOCK CHARACTERISTICS (Cont'd)

13.5.4 RC Oscillators

The ST7 internal clock can be supplied with an internal RC oscillator.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$f_{OSC\ (RCINT)}$	Internal RC oscillator frequency See Figure 73	T_A=25°C, V_{DD}=5V	2	3.5	6	MHz

Figure 72. Typical Application with RC oscillator

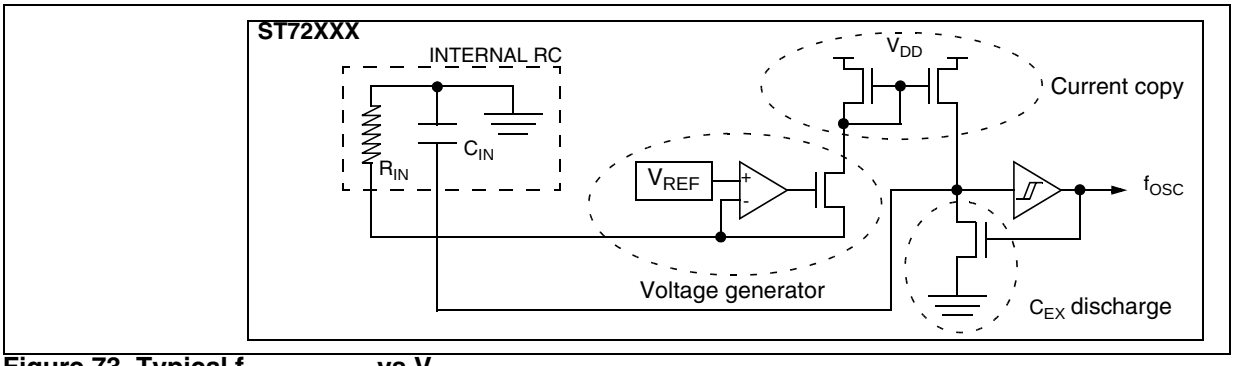
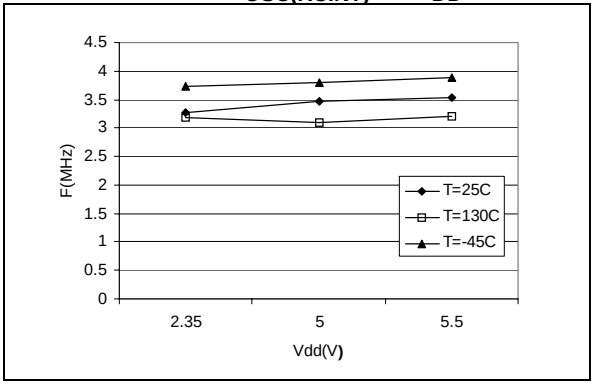


Figure 73. Typical $f_{OSC(RCINT)}$ vs V_{DD}



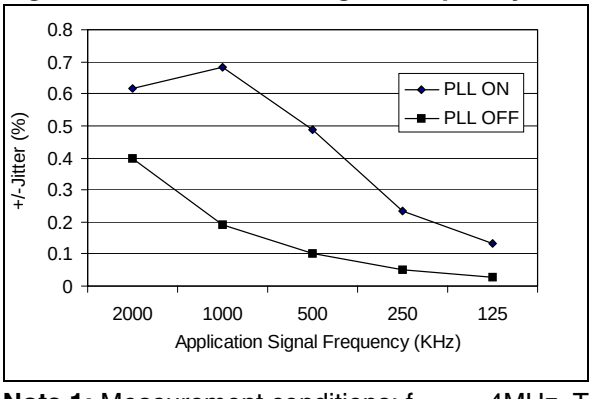
CLOCK CHARACTERISTICS (Cont'd)

13.5.5 PLL Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{DD(PLL)}$	PLL Operating Range	T_A 0 to 70°C	3.5		5.5	V
		T_A -40 to +85°C	4.5		5.5	
f_{OSC}	PLL input frequency range		2		4	MHz
$\Delta f_{CPU}/f_{CPU}$	Instantaneous PLL jitter 1)	f_{OSC} = 4 MHz.		1.0	2.5	%
		f_{OSC} = 2 MHz.		2.5	4.0	%

Note:
1. Data characterized but not tested.

Figure 74. PLL Jitter vs. Signal frequency[1]



The user must take the PLL jitter into account in the application (for example in serial communication or sampling of high frequency signals). The PLL jitter is a periodic effect, which is integrated over several CPU cycles. Therefore the longer the period of the application signal, the less it will be impacted by the PLL jitter.

Figure 74 shows the PLL jitter integrated on application signals in the range 125kHz to 2MHz. At frequencies of less than 125KHz, the jitter is negligible.

Note 1: Measurement conditions: f_{CPU} = 4MHz, T_A= 25°C

13.8 I/O PORT PIN CHARACTERISTICS

13.8.1 General Characteristics

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

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage[1]			V_{ss} - 0.3		0.3xV_{DD}	V
V_{IH}	Input high level voltage[1]			0.7xV_{DD}		V_{DD} + 0.3	
V_{hys}	Schmitt trigger voltage hysteresis [1]				400		mV
$I_{INJ(PIN)}$[2]	Injected current on Flash device pins PB0 and PB1					+4	mA
	Injected Current on other I/O pins	V_{DD}=5V				±4	
$\Sigma I_{INJ(PIN)}$[2]	Total injected current (sum of all I/O and control pins)					±25	
I_L	Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$				±1	µA
I_S	Static current consumption	Floating input mode[3]			400		
R_{PU}	Weak pull-up equivalent resistor [4]	V_{IN}=V_{SS}	V_{DD}=5V	50	85	250	kΩ
			V_{DD}=3V	170[1]	190	230[1]	
C_{IO}	I/O pin capacitance				5		pF
$t_{f(IO)out}$	Output high to low level fall time [1]	C_L=50pF Between 10% and 90%			25		ns
$t_{r(IO)out}$	Output low to high level rise time [1]				25		
$t_{w(IT)in}$	External interrupt pulse time[5]			1			t_{CPU}

Notes:

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

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

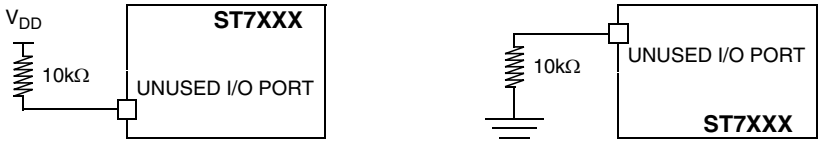
Caution: Negative current injection not allowed on Flash device pins PB0 and PB1.

3. Configuration not recommended, all unused pins must be kept at a fixed voltage: using the output mode of the I/O for example and leaving the I/O unconnected on the board or an external pull-up or pull-down resistor (see Figure 76). Data based on design simulation and/or technology characteristics, not tested in production.

4. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in Figure 77).

5. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

Figure 76. Two typical Applications with unused I/O Pin configured as input



Note: I/O can be left unconnected if it is configured as output (0 or 1) by the software. This has the advantage of greater EMC robustness and lower cost.

17 REVISION HISTORY

Table 33. Revision History

Date	Rev.	Main changes
February-2005	2.0	Added “SMBus V1.1 Compliant” for I²C on page 1 Added one note in Section 6.4.1 on page 24 Added SMBus compatibility information in Section 11.6 on page 103 and at the end of Section 11.6.4.1 on page 105 Changed note 1 in Section 13.2 on page 127 Added note 3 in Section 13.3.2 on page 129 Changed I_S value and note 3 in Section 13.8.1 on page 144 Added note in Figure 76 on page 144 Changed Figure 91 on page 151 and notes and added note 4 to Figure 92 on page 151 Added “LEAD-FREE PACKAGE INFORMATION” on page 161 Added ST72F264G2H6E in Table 28, “Supported Part Numbers,” on page 164 Changed Section 15.3 on page 166 Changed “ST72264 ROM/FASTROM MICROCONTROLLER OPTION LIST (Last update: 15 January 2004)” on page 165
01-Jun-05	3	Added -40°C to +85°C operating range in “Device Summary” on first page for LFBGA package (lead-free LFBGA package) Added illegal opcode reset on page 1, and in Section 12.2.1 on page 123 Changed notes under Figure 91 on page 151 Changed Vt_{POR} max. for ROM and note 3. Removed V_{HYS} min and maxin Section 13.3.2 on page 129 Changed Reset V_{IL}/V_{IH} in Section 13.9 on page 150 Added ROM current consumption in Section 13.4.1 on page 131 Added Active HALT min.in Section 13.4.2 on page 133 Removed note under table in Section 13.7.2 on page 142 Changed note on PB0/PB1 to apply to Flash only in Section 13.2 on page 127 and Section 13.8 on page 144 Added V_{DD} range for ADC operation, f_{ADC} min , conversion time and accuracy for ROM devices in Section 13.12 on page 157. Added 16-bit timer PWM Section 16.2.3 on page 169 Added SCI wrong break duration Section 16.2.9 on page 169 Moved errata sheet to Section 16 on page 168 and updated section for ROM and Flash devices