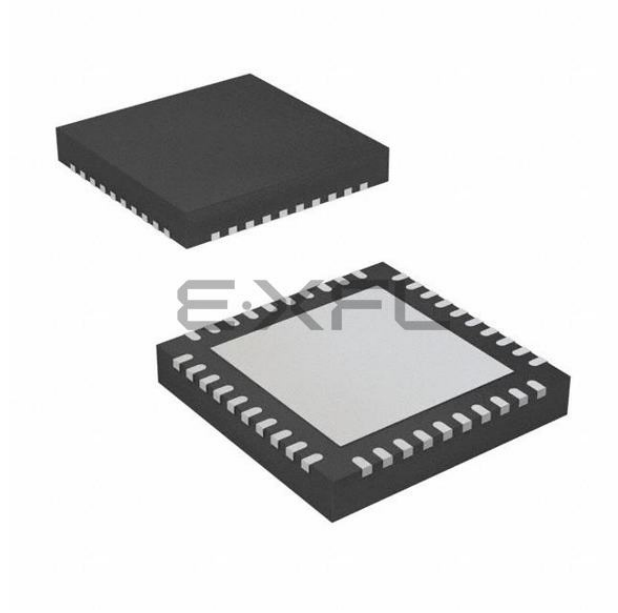


Welcome to E-XFL.COM

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, SCI, USB
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	19
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384 x 8
Voltage - Supply (Vcc/Vdd)	4V ~ 5.5V
Data Converters	A/D 12x8b
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	40-VFQFN Exposed Pad
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f63bk2u1tr

1 Introduction

The ST7263B microcontrollers form a sub-family of the ST7 MCUs dedicated to USB applications. The devices are based on an industry-standard 8-bit core and feature an enhanced instruction set. They operate at a 24 MHz or 12 MHz oscillator frequency. Under software control, the ST7263B MCUs may be placed in either Wait or Halt modes, thus reducing power consumption. The enhanced instruction set and addressing modes afford real programming potential. In addition to standard 8-bit data management, the ST7263B MCUs feature true bit manipulation, 8x8 unsigned multiplication and indirect addressing modes. The devices include an ST7 core, up to 32 Kbytes of program memory, up to 1024 bytes of RAM, 27 I/O lines and the following on-chip peripherals:

- USB low speed interface with 3 endpoints with programmable in/out configuration using the DMA architecture with embedded 3.3 V voltage regulator and transceivers (no external components are needed).
- 8-bit analog-to-digital converter (ADC) with 12 multiplexed analog inputs
- Industry standard asynchronous SCI serial interface
- Watchdog
- 16-bit Timer featuring an External clock input, 2 input Captures, 2 output Compares with Pulse Generator capabilities
- Fast I²C multimaster interface
- Low voltage reset (LVD) ensuring proper power-on or power-off of the device

The ST72F63B devices are Flash versions. They support programming in IAP mode (In-application programming) via the on-chip USB interface.

Table 2. Device overview

Features	ST7263BHx			ST7263BDx	ST7263BKx				ST7263BEx			
Program memory - Kbytes (Flash / ROM)	32	16	8	32	32	16	8	4	32	16	8	4
RAM (stack) - bytes	1024 (128)	512 (128)	384 (128)	1024 (128)	1024 (128	512 (128)	384 (128)	384 (128	1024 (128)	512 (128)	384 (128)	384 (128)
Standard Peripherals	Watchdog timer, 16-bit timer, USB											
Other Peripherals	SCI, I²C, ADC						SCI, AD	ADC	SCI, I²C			
I/Os (high current)	27 (10)				19 (10)				14 (6)			
Operating Supply	4.0 V to 5.5 V											
CPU frequency	8 MHz (with 24 MHz oscillator) or 4 MHz (with 12 MHz oscillator)											
Operating temp.	0 °C to +70 °C											
Packages	LQFP48 (7x7)			QFN40 (6x6)	SDIP32/ SO34		QFN40 (6x6)	SDIP32/ SO34	SO24			

Table 4. Device pin description (SO24) (continued)

Pin n°	Pin name	Type	Level		Port /control						Main function (after reset)	Alternate function
					Input				Output			
SO24			Input	Output	float	wpu	int	ana	OD	PP		
20	PA0/MCO	I/O	CT				X			X	Port A0	Main Clock output
21	V_{SSA}	S									Analog ground	
22	USBDP	I/O									USB bidirectional data (data +)	
23	USBDM	I/O									USB bidirectional data (data -)	
24	USBVCC	O									USB power supply	

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 organization 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
- Readout protection
- Register access security system (RASS) to prevent accidental programming or erasing

4.3 Structure

The Flash memory is organized 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 7*). 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 8*). 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 7. Sectors available in Flash devices

Flash size (Kbytes)	Available sectors
4	Sector 0
8	Sectors 0,1
> 8	Sectors 0,1, 2

4.6 IAP (in-application programming)

This mode uses a BootLoader program previously stored in Sector 0 by the user (in ICP mode or by plugging the device in a programming tool).

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored, etc.). For example, it is possible to download code from the SCI or other type of serial interface and program it in the Flash. IAP mode can be used to program any of the Flash sectors except Sector 0, which is write/erase protected to allow recovery in case errors occur during the programming operation.

4.7 Related documentation

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

4.8 Register description

Flash Control/status register (FCSR)

This register is reserved for use by programming tool software. It controls the Flash programming and erasing operations.

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
Read/write							

6.2 Clock system

6.2.1 General description

The MCU accepts either a crystal or ceramic resonator, or an external clock signal to drive the internal oscillator. The internal clock (f_{CPU}) is derived from the external oscillator frequency (f_{OSC}), which is divided by 3 (and by 2 or 4 for USB, depending on the external clock used). The internal clock is further divided by 2 by setting the SMS bit in the miscellaneous register.

Using the OSC24/12 bit in the option byte, a 12 MHz or a 24 MHz external clock can be used to provide an internal frequency of either 2, 4 or 8 MHz while maintaining a 6 MHz for the USB (refer to *Figure 18*).

The internal clock signal (f_{CPU}) is also routed to the on-chip peripherals. The CPU clock signal consists of a square wave with a duty cycle of 50%.

The internal oscillator is designed to operate with an AT-cut parallel resonant quartz or ceramic resonator in the frequency range specified for f_{osc}. The circuit shown in *Figure 17* is recommended when using a crystal, and *Table 8* lists the recommended capacitance. The crystal and associated components should be mounted as close as possible to the input pins in order to minimize output distortion and start-up stabilization time.

Table 8. Recommended Values for 24 MHz crystal resonator

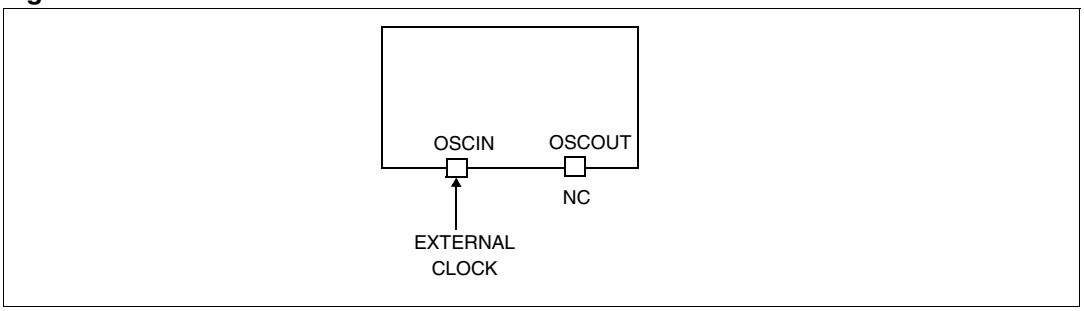
Recommended capacitance and resistance			
R_{SMAX}(1)	20 Ω	25 Ω	70 Ω
C_{OSCIN}	56pF	47pF	22pF
C_{OSCOUT}	56pF	47pF	22pF
R_P	1-10 MΩ	1-10 MΩ	1-10 MΩ

1. R_{SMAX} is the equivalent serial resistor of the crystal (see crystal specification).

6.2.2 External clock

An external clock may be applied to the OSCIN input with the OSCOUT pin not connected, as shown on *Figure 16*. The t_{OXOV} specifications do not apply when using an external clock input. The equivalent specification of the external clock source should be used instead of t_{OXOV} (see *Table 62: Control timing characteristics*).

Figure 16. External clock source connections



HALT instruction

If the Watchdog reset on HALT option is selected by option byte, a HALT instruction causes an immediate reset generation if the Watchdog is activated (WDGA bit is set).

Using Halt mode with the WDG (option)

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

In this case, the HALT instruction stops the oscillator. When the oscillator is stopped, the WDG stops counting and is no longer able to generate a reset until the microcontroller receives an external interrupt or a reset.

If an external interrupt is received, the WDG restarts counting after 4096 CPU clocks. If a reset is generated, the WDG is disabled (reset state).

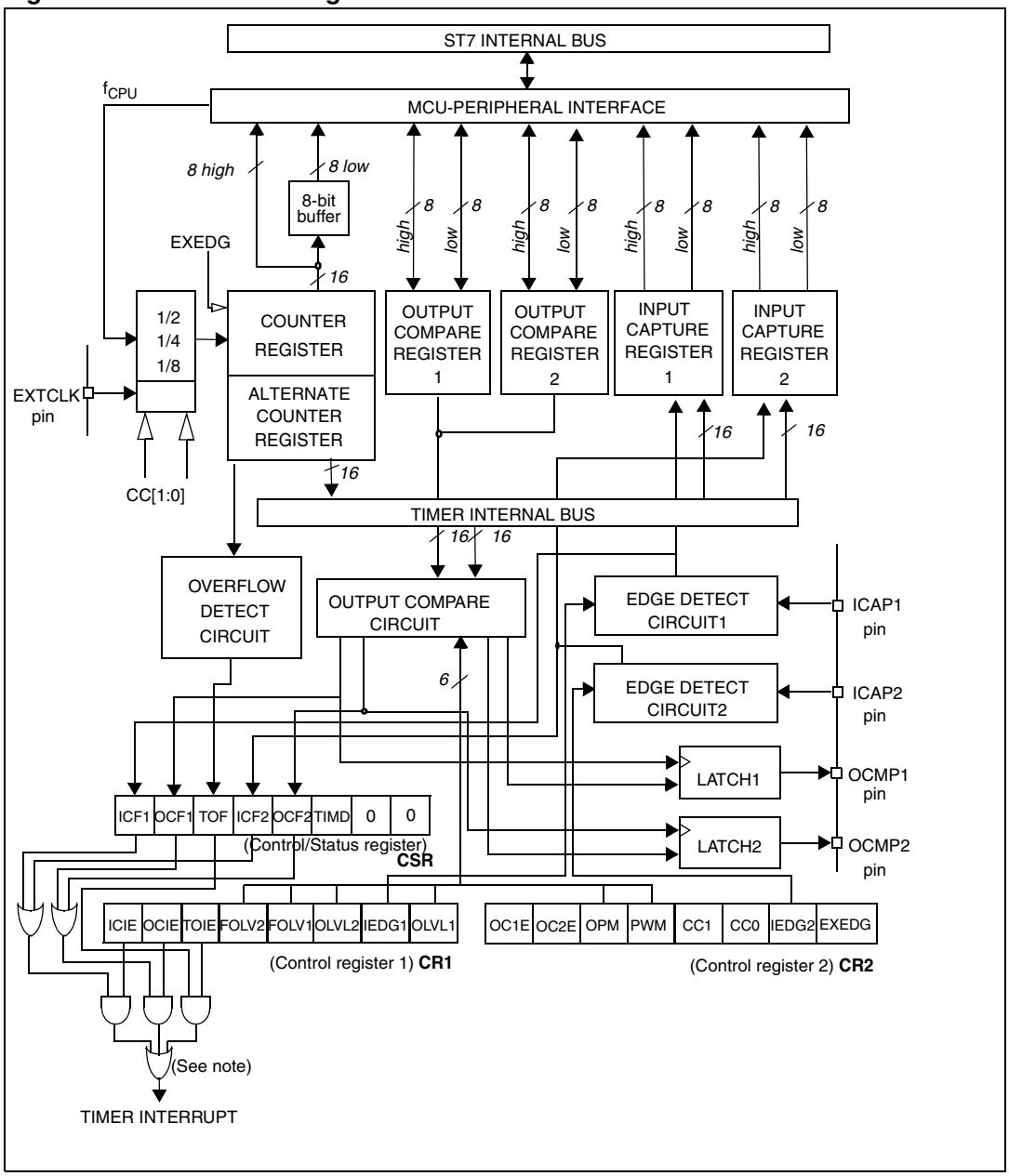
Recommendations:

- Make sure that an external event is available to wake up the microcontroller from Halt mode.
- Before executing the HALT instruction, refresh the WDG counter, to avoid an unexpected WDG reset immediately after waking up the microcontroller.
- When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as “Input Pull-up with interrupt” before executing the HALT instruction. The main reason for this is that the I/O may be wrongly configured due to external interference or by an unforeseen logical condition.
- For the same reason, reinitialize the level sensitiveness of each external interrupt as a precautionary measure.
- The opcode for the HALT instruction is 0x8E. To avoid an unexpected HALT instruction due to a program counter failure, it is advised to clear all occurrences of the data value 0x8E from memory. For example, avoid defining a constant in ROM with the value 0x8E.
- As the HALT instruction clears the I bit in the CC register to allow interrupts, the user may choose to clear all pending interrupt bits before executing the HALT instruction. This avoids entering other peripheral interrupt routines after executing the external interrupt routine corresponding to the wakeup event (reset or external interrupt).

11.1.7 Interrupts

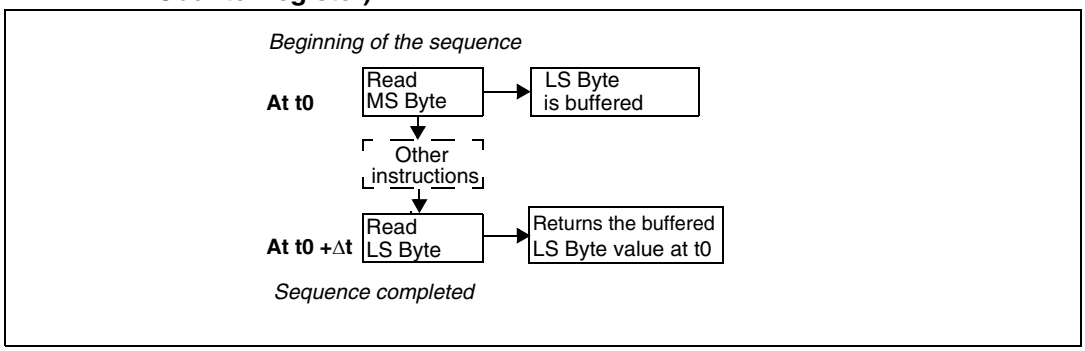
None.

Figure 27. Timer block diagram



1. If IC, OC and TO interrupt requests have separate vectors then the last OR is not present (See device Interrupt Vector Table).

Figure 28. 16-bit read sequence (from either the Counter register or the Alternate Counter register)



The user must read the MS Byte first, then the LS Byte value is buffered automatically.

This buffered value remains unchanged until the 16-bit read sequence is completed, even if the user reads the MS Byte several times.

After a complete reading sequence, if only the CLR register or ACLR register are read, they return the LS Byte of the count value at the time of the read.

Whatever the timer mode used (input capture, output compare, One Pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFFFh to 0000h 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 CLR register.

Note: The TOF bit is not cleared by accesses to ACLR register. The advantage of accessing the ACLR register rather than the CLR 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).

External clock

The external clock (where available) is selected if CC0 = 1 and CC1 = 1 in the CR2 register.

The status of the EXEDG bit in the CR2 register determines the type of level transition on the external clock pin EXTCLK that will trigger the free running counter.

The counter is synchronized with the falling edge of the internal CPU clock.

Input Capture

In this section, the index, *i*, may be 1 or 2 because there are two input capture functions in the 16-bit timer.

The two 16-bit input capture registers (IC1R and IC2R) are used to latch the value of the free running counter after a transition is detected on the ICAP*i* pin (see *Figure 32*).

Table 19. IC/R register

	MS Byte	LS Byte
ICiR	IC*i*HR	IC*i*LR

IC*i*R register is a read-only register.

The active transition is software programmable through the IEDG*i* bit of Control registers (CR*i*).

Timing resolution is one count of the free running counter: (f_{CPU}/CC[1:0]).

Procedure

To use the input capture function select the following in the CR2 register:

1. Select the timer clock (CC[1:0]) (see *Table 24*).
2. Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit (the ICAP2 pin must be configured as floating input or input with pull-up without interrupt if this configuration is available).
3. Select the following in the CR1 register:
   a) Set the ICIE bit to generate an interrupt after an input capture coming from either the ICAP1 pin or the ICAP2 pin
   b) Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1pin must be configured as floating input or input with pull-up without interrupt if this configuration is available).

When an input capture occurs:

- ICF*i* bit is set.
- The IC*i*R register contains the value of the free running counter on the active transition on the ICAP*i* pin (see *Figure 33*).
- A timer interrupt is generated if the ICIE bit is set and the I bit is cleared in the CC register. Otherwise, the interrupt remains pending until both conditions become true.

Clearing the input Capture interrupt request (that is, clearing the ICF*i* bit) is done in two steps:

1. Reading the SR register while the ICF*i* bit is set.
2. An access (read or write) to the IC*i*LR register.

Note: 1 After reading the ICiHR register, transfer of input capture data is inhibited and ICFi will never be set until the ICiLR register is also read.

2 The ICiR register contains the free running counter value which corresponds to the most recent input capture.

3 The two input capture functions can be used together even if the timer also uses the two output compare functions.

4 In One Pulse mode and PWM mode only input Capture 2 can be used.

5 The alternate inputs (ICAP1 and ICAP2) are always directly connected to the timer. So any transitions on these pins activates the input capture function.
Moreover if one of the ICAPi pins is configured as an input and the second one as an output, an interrupt can be generated if the user toggles 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 generation in order to measure events that go beyond the timer range (FFFFh).

Figure 32. Input Capture block diagram

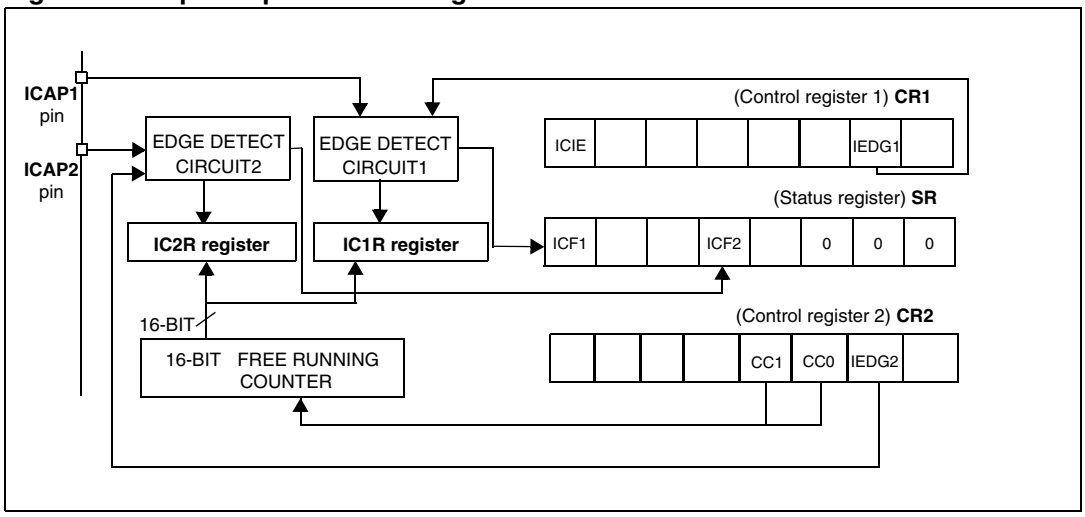
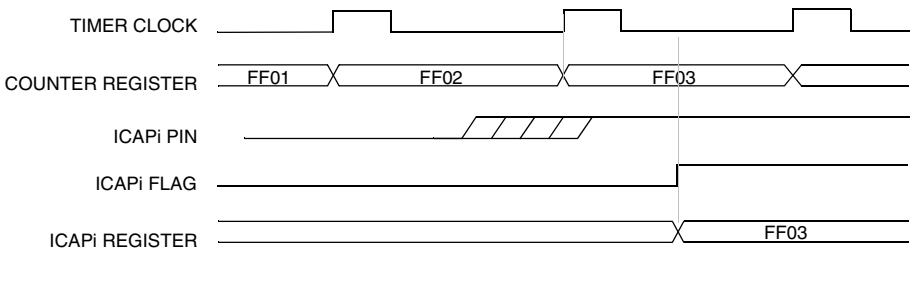


Figure 33. Input Capture timing diagram



1. The rising edge is the active edge.

11.3.4 Functional description

The block diagram of the Serial Control Interface, is shown in *Figure 41* It contains 6 dedicated registers:

- Two control registers (SCICR1 & SCICR2)
- A status register (SCISR)
- A baud rate register (SCIBRR)

Refer to the register descriptions in *Section 11.3.7* for the definitions of each bit.

Serial data format

Word length may be selected as being either 8 or 9 bits by programming the M bit in the SCICR1 register (see *Figure 41*).

The TDO pin is in low state during the start bit.

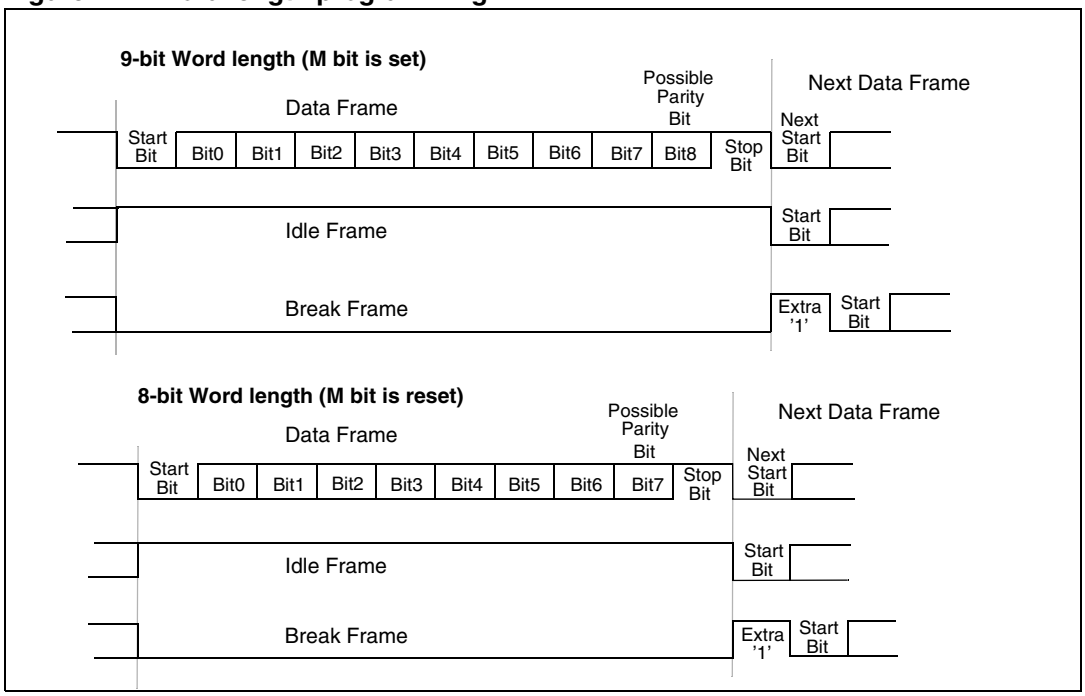
The TDO pin is in high state during the stop bit.

An Idle character is interpreted as an entire frame of “1”s followed by the start bit of the next frame which contains data.

A Break character is interpreted on receiving “0”s for some multiple of the frame period. At the end of the last break frame the transmitter inserts an extra “1” bit to acknowledge the start bit.

Transmission and reception are driven by their own baud rate generator.

Figure 42. Word length programming



Setting the START bit while the BUSY bit is cleared causes the interface to switch to Master mode (M/SL bit set) and generates a Start condition.

Once the Start condition is sent, the EVF and SB bits are set by hardware with an interrupt if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register with the Slave address byte, **holding the SCL line low** (see *Figure 48* Transfer sequencing EV5).

Slave address transmission

Then the slave address byte is sent to the SDA line via the internal shift register.

After completion of this transfer (and acknowledge from the slave if the ACK bit is set), the EVF bit is set by hardware with interrupt generation if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the CR register (for example set PE bit), **holding the SCL line low** (see *Figure 48* Transfer sequencing EV6).

Next the master must enter Receiver or Transmitter mode.

Master receiver

Following the address transmission and after the SR1 and CR registers have been accessed, the master 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 by hardware 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 48* Transfer sequencing EV7).

To close the communication: before reading the last byte from the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

Note: In order to generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just before reading the second last data byte.

Master transmitter

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

The master waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see *Figure 48* Transfer sequencing EV8).

When the acknowledge bit is received, the interface sets, EVF and BTF bits with an interrupt if the ITE bit is set.

To close the communication: after writing the last byte to the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

Error cases

- **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and BERR bits are set by hardware with an interrupt if ITE is set.
  Note that BERR will not be set if an error is detected during the first or second pulse of

11.5.7 Register description

I²C Control register (CR)

Reset value: 0000 0000 (00h)

7							0
0	0	PE	ENGC	START	ACK	STOP	ITE
Read/write							

[7:6] Reserved. Forced to 0 by hardware.

5 **PE** *Peripheral enable.*
This bit is set and cleared by software.
0: Peripheral disabled
1: Master/Slave capability

Note: When PE=0, all the bits of the CR register and the SR register except the Stop bit are reset. All outputs are released while PE=0.
When PE=1, the corresponding I/O pins are selected by hardware as alternate functions.
*To enable the I²C interface, write the CR register **TWICE** with PE=1 as the first write only activates the interface (only PE is set).*

4 **ENGC** *Enable General Call.*
This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0). The 00h General Call address is acknowledged (01h ignored).
0: General Call disabled
1: General Call enabled

Note: In accordance with the I^2C standard, when GCAL addressing is enabled, an I^2C slave can only receive data. It will not transmit data to the master.

3 **START** *Generation of a Start condition*. This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0) or when the Start condition is sent (with interrupt generation if ITE=1).
In master mode:
0: No start generation
1: Repeated start generation
In slave mode:
0: No start generation
1: Start generation when the bus is free

I²C Data register (DR)

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. The following data bytes are then received one by one after reading the DR register.

Reset value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0
Read/write							

I²C Own Address register (OAR)

Reset value: 0000 0000 (00h)

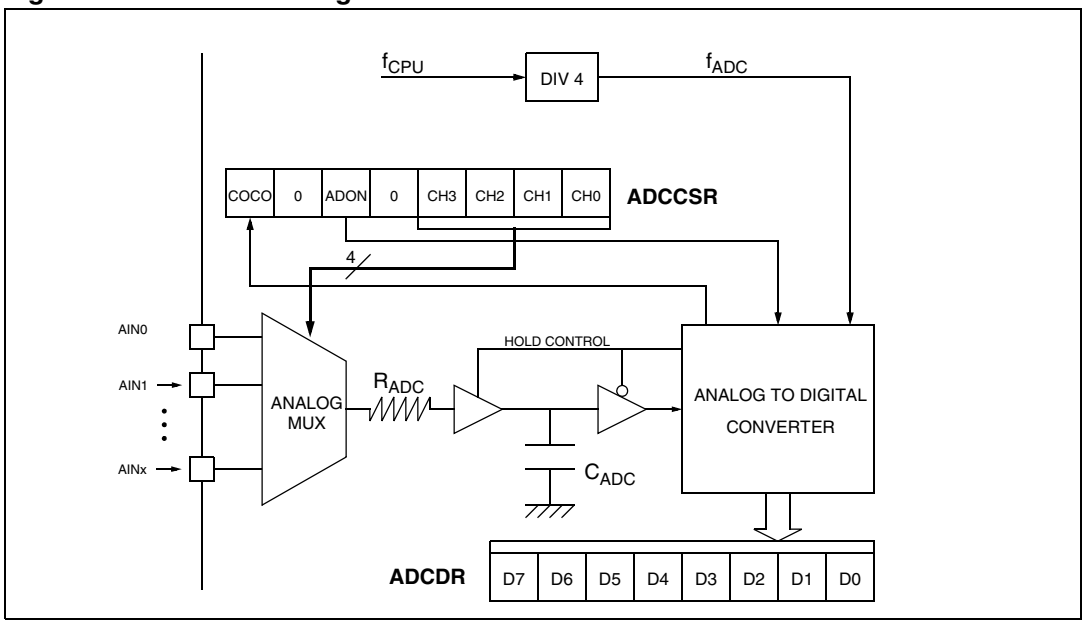
7							0
ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
Read/write							

[7:1] **ADD[7:1]** *Interface address.*
These bits define the I²C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

0 **ADD0** *Address direction bit.*
This bit is don't care, the interface acknowledges either 0 or 1. It is not cleared when the interface is disabled (PE=0).

Note: Address 01h is always ignored.

Figure 50. ADC block diagram



Digital A/D conversion result

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

If the input voltage (V_{AIN}) is greater than or equal to V_{DDA} (high-level voltage reference) then the conversion result in the DR register is FFh (full scale) without overflow indication.

If input voltage (V_{AIN}) is lower than or equal to V_{SSA} (low-level voltage reference) then the conversion result in the DR register is 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDR register. The accuracy of the conversion is described in the parametric 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 allotted time.

A/D conversion phases

The A/D conversion is based on two conversion phases as shown in *Figure 51*:

- Sample capacitor loading [duration: t_{LOAD}]
  During this phase, the V_{AIN} input voltage to be measured is loaded into the C_{ADC} sample capacitor.
- A/D conversion [duration: t_{CONV}]
  During this phase, the A/D conversion is computed (8 successive approximations cycles) and the C_{ADC} sample capacitor is disconnected from the analog input pin to get the optimum analog to digital conversion accuracy.

While the ADC is on, these two phases are continuously repeated.

At the end of each conversion, the sample capacitor is kept loaded with the previous measurement load. The advantage of this behavior is that it minimizes the current consumption on the analog pin in case of single input channel measurement.

13 Electrical characteristics

13.1 Parameter conditions

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

13.1.1 Minimum and maximum values

Unless otherwise specified the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage and frequencies by tests in production on 100% of the devices with an ambient temperature at T_A=25 °C and T_A=$T_A$max (given by the selected temperature range).

Data based on characterization results, design simulation and/or technology characteristics are indicated in the table footnotes and are not tested in production. Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value plus or minus three times the standard deviation (mean±3Σ).

13.1.2 Typical values

Unless otherwise specified, typical data are based on T_A=25 °C, V_{DD}=5 V. They are given only as design guidelines and are not tested.

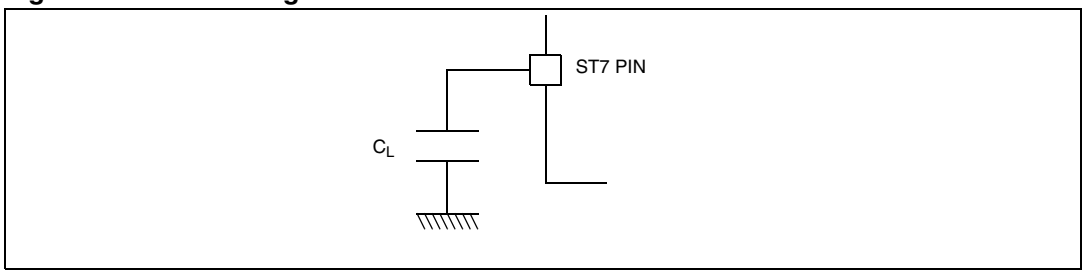
13.1.3 Typical curves

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

13.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in *Figure 52*.

Figure 52. Pin loading conditions



13.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in *Figure 53*.

13.4 Supply current characteristics

The following current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To get the total device consumption, the two current values must be added (except for Halt mode for which the clock is stopped).

Table 60. Supply current characteristics

Symbol	Parameter	Conditions		Typ	Max	Unit
$\Delta I_{DD(\Delta Ta)}$	Supply current variation vs. temperature	Constant V_{DD} and f_{CPU}		-	10(1)	%
I_{DD}	CPU Run mode	I/Os in input mode	f_{CPU} = 4 MHz	7.5	9(2)(1)	mA
			f_{CPU} = 8 MHz	10.5	13(2)	
	CPU Wait mode		f_{CPU} = 4 MHz	6	8(1)	mA
			f_{CPU} = 8 MHz	8.5	11(2)	
	CPU Halt mode(3)	LVD disabled		25	40(1)	µA
	USB Suspend mode(4)	LVD disabled		100	120	µA
		LVD enabled		230	-	

1. Not tested in production, guaranteed by characterization.
2. Oscillator and watchdog running. All others peripherals disabled.
3. USB Transceiver and ADC are powered down.
4. CPU in Halt mode. Current consumption of external pull-up (1.5Kohms to USBVCC) and pull-down (15Kohms to V_{SSA}) not included.

Figure 55. Typ. I_{DD} in Run at f_{CPU} = 4 and 8 MHz

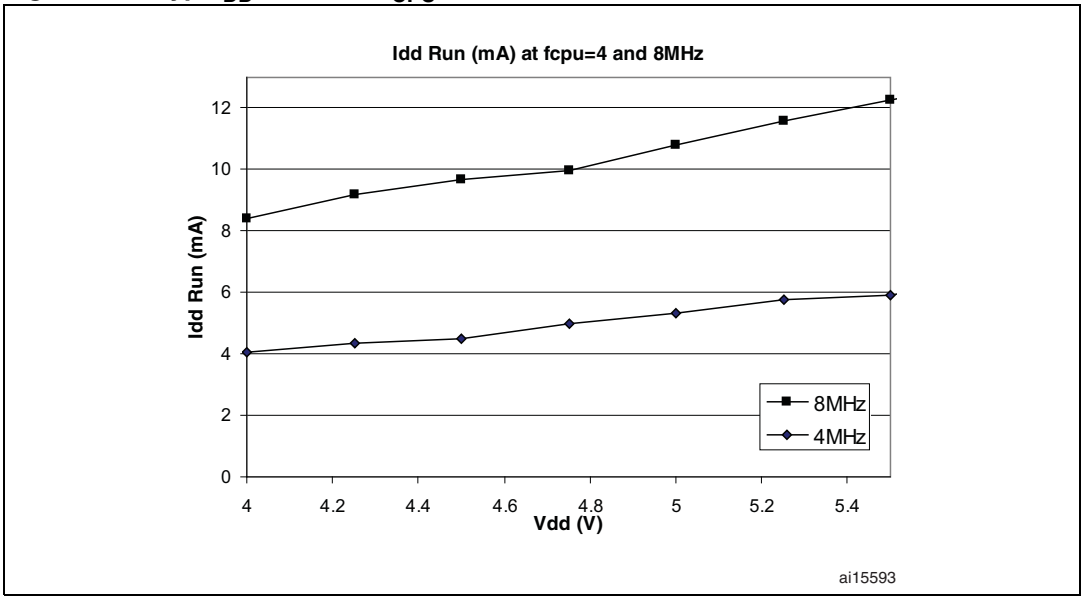
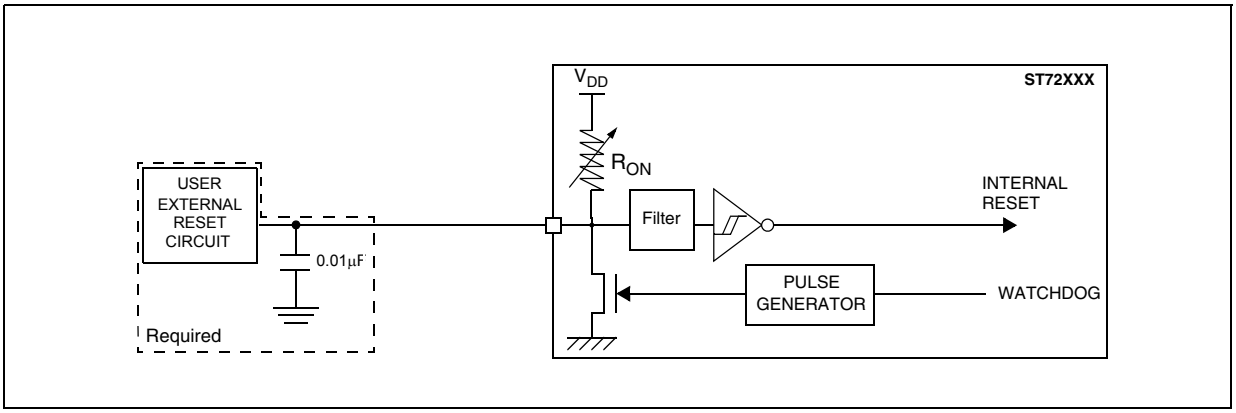


Figure 74. $\overline{\text{RESET}}$ pin protection when LVD is disabled



13.10 Communication interface characteristics

13.10.1 USB interface

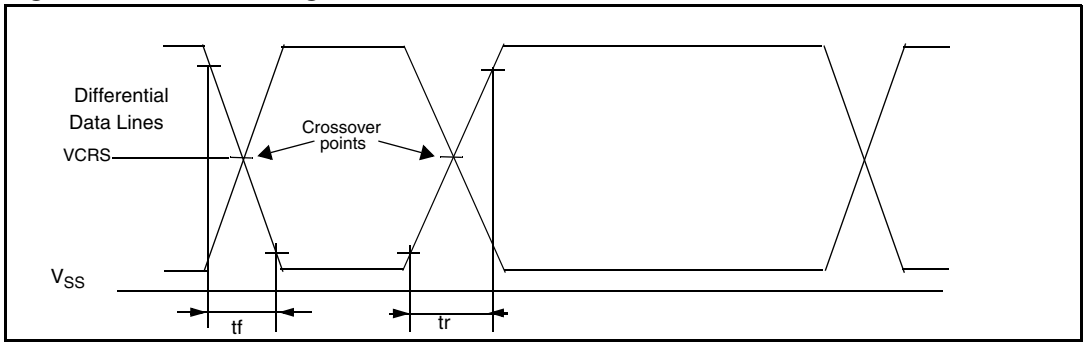
Operating conditions T_A = 0 to +70 °C, V_{DD} = 4.0 to 5.25 V unless otherwise specified.

Table 73. USB DC characteristics

Symbol	Parameter	Conditions	Min.	Max.	Unit
V_{DI}	Differential input sensitivity	I(D+, D-)	0.2	-	V(1)
V_{CM}	Differential common mode range	Includes V_{DI} range	0.8	2.5	
V_{SE}	Single ended receiver threshold		0.8	2.0	
V_{OL}	Static output low	R_L(2)of 1.5 KΩ to 3.6 V	-	0.3	
V_{OH}	Static output high	R_L(2) of 15 KΩ to V_{SS}	2.8	3.6	
USBV	USBVCC: voltage level(3)	V_{DD}=5 V	3.00	3.60	

1. All the voltages are measured from the local ground potential.
2. R_L is the load connected on the USB drivers.
3. To improve EMC performance (noise immunity), it is recommended to connect a 100nF capacitor to the USBVCC pin.

Figure 75. USB data signal rise and fall time



16.5 Halt mode power consumption with ADC on

Description

If the A/D converter is being used when Halt mode is entered, the power consumption in Halt mode may exceed the maximum specified in the datasheet.

Workaround

Switch off the ADC by software (ADON=0) before executing a HALT instruction.

16.6 SCI wrong BREAK duration

Description

A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 20 bits instead of 10 bits if M=0
- 22 bits instead of 11 bits if M=1.

In the same way, as long as the SBK bit is set, break characters are sent to the TDO pin. This may generate one break more than expected.

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (fCPU=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Workaround

If this wrong duration is not compliant with the communication protocol in the application, software can request that an Idle line be generated before the break character. In this case, the break duration is always correct assuming the application is not doing anything between the idle and the break. This can be ensured by temporarily disabling interrupts.

The exact sequence is:

1. Disable interrupts
2. Reset and Set TE (IDLE request)
3. Set and Reset SBK (Break Request)
4. Re-enable interrupts