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, LINbus, SCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	34
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f344s2t6

In Flash devices, this protection is removed by reprogramming the option. In this case, both program and data E^2 memory are automatically erased, and the device can be reprogrammed.

Read-out protection selection depends on the device type:

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

4.5.2 Flash write/erase protection

Write/erase protection, when set, makes it impossible to both overwrite and erase program memory. It does not apply to E^2 data. Its purpose is to provide advanced security to applications and prevent any change being made to the memory content.

Warning: Once set, Write/erase protection can never be removed. A write-protected Flash device is no longer reprogrammable.

Write/erase protection is enabled through the FMP_W bit in the option byte.

4.6 Register description

4.6.1 Flash control/status register (FCSR)

Reset value: 0000 0000 (00h)
1st RASS Key: 0101 0110 (56h)
2nd RASS Key: 1010 1110 (AEh)

7							0
0	0	0	0	0	OPT	LAT	PGM
Read/Write							

Note: This register is reserved for programming using ICP, IAP or other programming methods. It controls the XFlash programming and erasing operations. For details on XFlash programming, refer to the ST7 Flash Programming Reference Manual.

When an EPB or another programming tool is used (in socket or ICP mode), the RASS keys are sent automatically.

Read operation (E2LAT = 0)

The EEPROM can be read as a normal ROM location when the E2LAT bit of the EECSR register is cleared.

On this device, Data EEPROM can also be used to execute machine code. Take care not to write to the Data EEPROM while executing from it. This would result in an unexpected code being executed.

Write operation (E2LAT = 1)

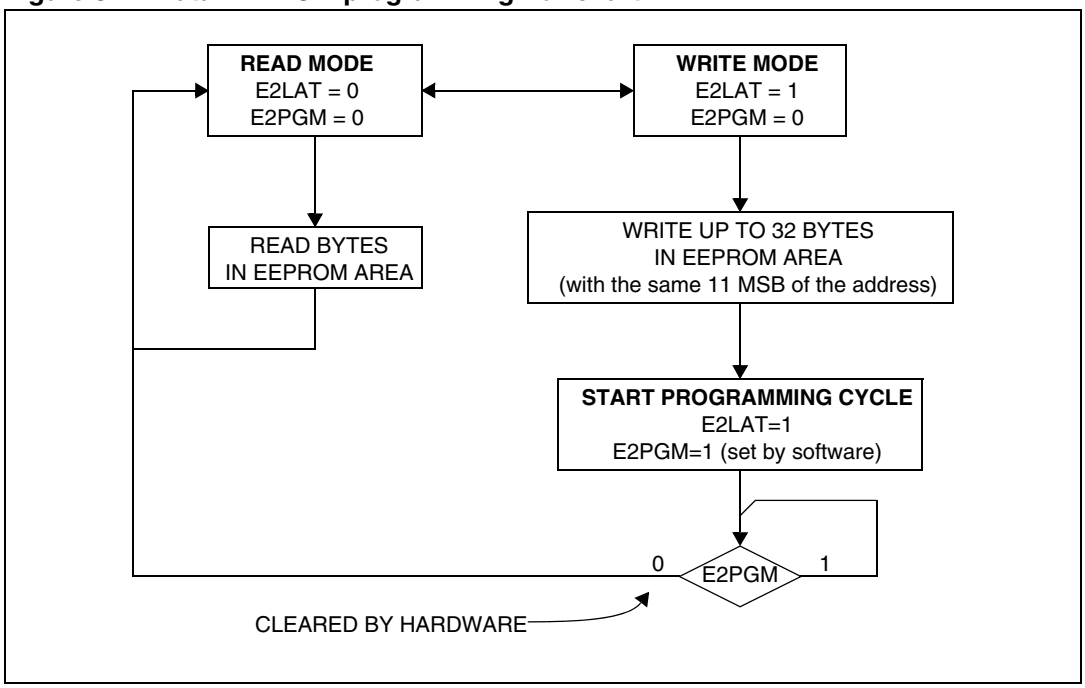
To access the write mode, the E2LAT bit has to be set by software (the E2PGM bit remains cleared). When a write access to the EEPROM area occurs, the value is latched inside the 32 data latches according to its address.

When E2PGM bit is set by the software, all the previous bytes written in the data latches (up to 32) are programmed in the EEPROM cells. The effective high address (row) is determined by the last EEPROM write sequence. To avoid wrong programming, the user must take care that all the bytes written between two programming sequences have the same high address: only the five Least Significant Bits of the address can change.

The programming cycle is fully completed when the E2PGM bit is cleared.

Note: Care should be taken during the programming cycle. Writing to the same memory location will over-program the memory (logical AND between the two write access data results) because the data latches are only cleared at the end of the programming cycle and by the falling edge of the E2LAT bit.
It is not possible to read the latched data.
This note is illustrated by the Figure 10.

Figure 8. Data EEPROM programming flowchart

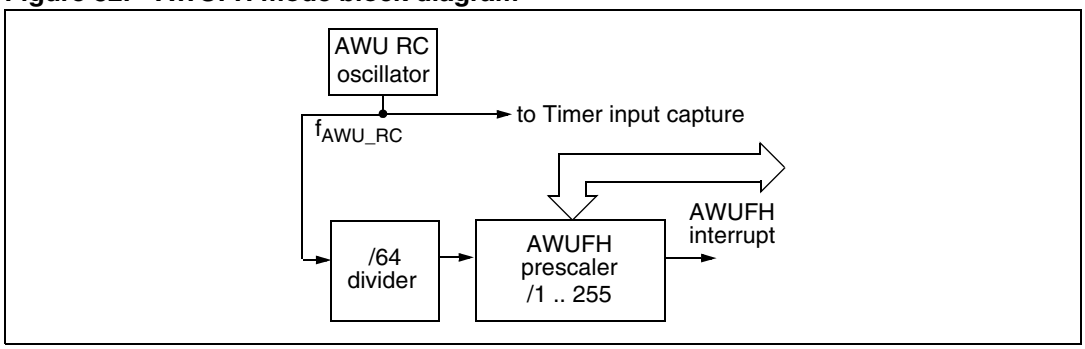


9.6 Auto-wakeup from Halt mode

Auto-wakeup from Halt (AWUFH) mode is similar to Halt mode with the addition of an internal RC oscillator for wake-up. Compared to Active-Halt mode, AWUFH has lower power consumption because the main clock is not kept running, but there is no accurate real-time clock available.

It is entered by executing the HALT instruction when the AWUEN bit in the AWUCSR register has been set and the OIE bit in the MCCSR register is cleared (see *Section 11.2: Main clock controller with real-time clock and beeper (MCC/RTC)* for more details).

Figure 32. AWUFH mode block diagram



As soon as Halt mode is entered, and if the AWUEN bit has been set in the AWUCSR register, the AWU RC oscillator provides a clock signal (f_{AWU_RC}). Its frequency is divided by a fixed divider and a programmable prescaler controlled by the AWUPR register. The output of this prescaler provides the delay time. When the delay has elapsed the AWUF flag is set by hardware and an interrupt wakes-up the MCU from Halt mode. At the same time the main oscillator is immediately turned on and a 256 or 4096 cycle delay is used to stabilize it. After this startup delay, the CPU resumes operation by servicing the AWUFH interrupt. The AWU flag and its associated interrupt are cleared by software reading the AWUCSR register.

To compensate for any frequency dispersion of the AWU RC oscillator, it can be calibrated by measuring the clock frequency f_{AWU_RC} and then calculating the right prescaler value. Measurement mode is enabled by setting the AWUM bit in the AWUCSR register in Run mode. This connects internally f_{AWU_RC} to the ICAP2 input of the 16-bit timer A, allowing the f_{AWU_RC} to be measured using the main oscillator clock as a reference timebase.

Similarities with Halt mode

The following AWUFH mode behavior is the same as normal Halt mode:

- The MCU can exit AWUFH mode by means of any interrupt with exit from Halt capability or a reset (see *Section 9.4: Halt mode*).
- When entering AWUFH mode, the I[1:0] bits in the CC register are forced to 10b to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.
- In AWUFH mode, the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. None of the peripherals are clocked except those which get their clock supply from another clock generator (such as an external or auxiliary oscillator like the AWU oscillator).
- The compatibility of the Watchdog operation with the AWUFH mode is configured by the WDGHALT option bit in the option byte. Depending on this setting, the HALT instruction, when executed while the Watchdog system is enabled, can generate a Watchdog reset.

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

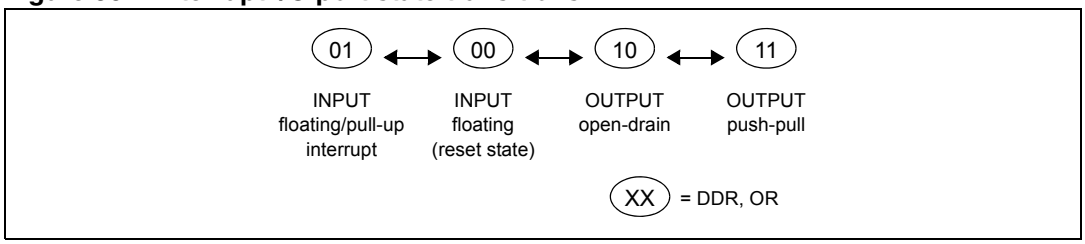
Warning: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

10.3 I/O port implementation

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific feature of the I/O port such as ADC Input or true open drain.

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in *Figure 36*. Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 36. Interrupt I/O port state transitions



10.4 Low-power modes

Table 29. Description

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.

10.5 Interrupts

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

Table 30. Description of interrupt events

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

10.5.1 I/O port implementation

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

Standard ports: PA[5:4], PC[7:0], PD[5:0], PE0, PF[7:6], PF4

Table 31. I/O port register configurations (standard ports)

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

Interrupt ports: PB4, PB[2:0], PF[1:0] (with pull-up)

Table 32. I/O port register configurations (interrupt ports 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

PA3, PE1, PB3, PF2 (without pull-up)

Table 33. I/O port register configurations (interrupt ports without pull-up)

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

True open-drain ports: PA[7:6], PD[7:6]

Table 34. I/O port register configurations (true open drain ports)

Mode	DDR
floating input	0
open drain (high sink ports)	1

11 On-chip peripherals

11.1 Window watchdog (WWDG)

11.1.1 Introduction

The window watchdog is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The Watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit becomes cleared. An MCU reset is also generated if the 7-bit downcounter value (in the control register) is refreshed before the downcounter has reached the window register value. This implies that the counter must be refreshed in a limited window.

11.1.2 Main features

- Programmable free-running downcounter
- Conditional reset
  - Reset (if watchdog activated) when the downcounter value becomes less than 40h
  - Reset (if watchdog activated) if the downcounter is reloaded outside the window (see *Figure 40*)
- Hardware/Software Watchdog activation (selectable by option byte)
- Optional reset on HALT instruction (configurable by option byte)

11.1.3 Functional description

The counter value stored in the WDGCR register (bits T[6:0]), is decremented every 16384 f_{OSC2} cycles (approx.), and the length of the timeout period can be programmed by the user in 64 increments.

If the watchdog is activated (the WDGA bit is set) and when the 7-bit downcounter (T[6:0] bits) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 30 µs. If the software reloads the counter while the counter is greater than the value stored in the window register, then a reset is generated.

Table 43. Beep control

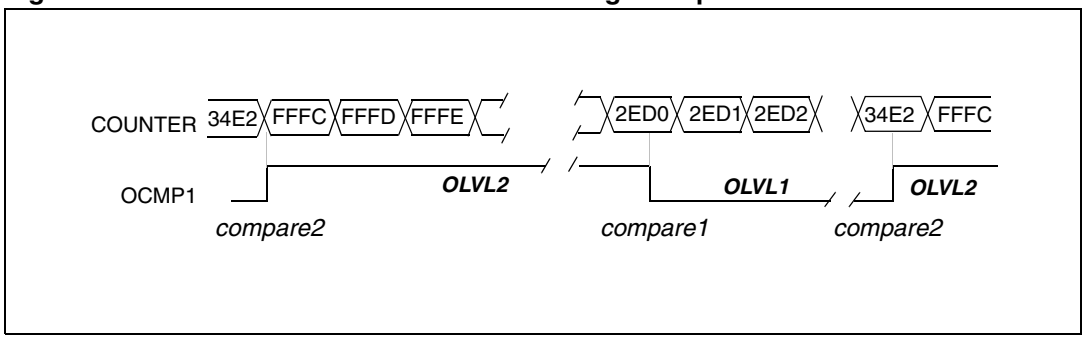
BC1	BC0	Beep mode with f_{OSC2} = 8 MHz	
0	0	Off	
0	1	~2-kHz	Output beep signal ~50% duty cycle
1	0	~1-kHz	
1	1	~500-Hz	

The beep output signal is available in Active-halt mode but has to be disabled to reduce the consumption.

Table 44. Main clock controller register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
002Bh	**SICSR** Reset value	0	AVDIE 0	AVDF 0	LVDRF x	LOCKED 0	0	0	WDGRF x
002Ch	**MCCSR** Reset value	MCO 0	CP1 0	CP0 0	SMS 0	TB1 0	TB0 0	OIE 0	OIF 0
002Dh	**MCCBCR** Reset value	0	0	0	0	0	0	BC1 0	BC0 0

Figure 54. Pulse width modulation mode timing example



1. OC1R = 2ED0h, OC2R = 34E2, OLVL1 = 0, OLVL2 = 1

Pulse-width modulation mode

Pulse-width modulation (PWM) mode enables the generation of a signal with a frequency and pulse length determined by the value of the OC1R and OC2R registers.

Pulse-width modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

In PWM mode, double buffering is implemented on the output compare registers. Any new values written in the OC1R and OC2R registers are loaded in their respective shadow registers (double buffer) only at the end of the PWM period (OC2) to avoid spikes on the PWM output pin (OCMP1). The shadow registers contain the reference values for comparison in PWM “double buffering” mode.

Note: There is a locking mechanism for transferring the OCiR value to the buffer. After a write to the OCiHR register, the transfer of the new compare value to the buffer is inhibited until OCiLR is also written.

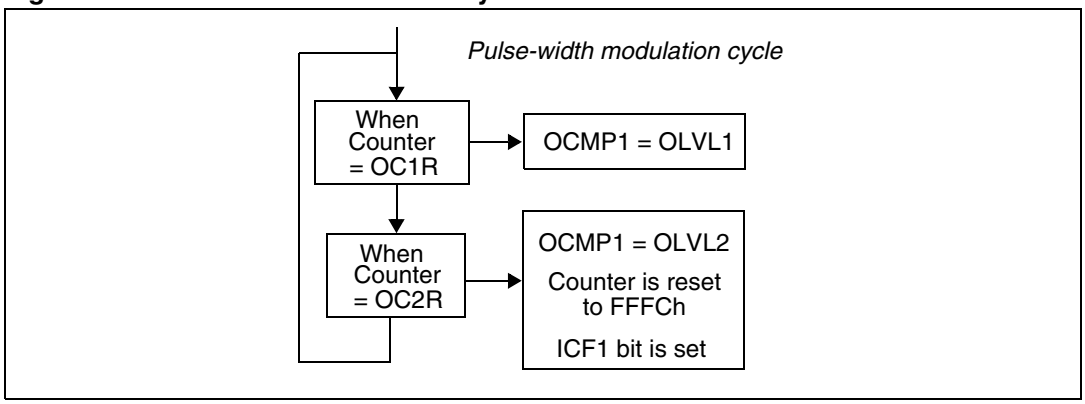
Unlike in Output Compare mode, the compare function is always enabled in PWM mode.

Procedure:

To use pulse-width modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if (OLVL1=0 and OLVL2=1) using the formula in the opposite column.
3. Select the following in the CR1 register:
   - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC1R register.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC2R register.
4. Select the following in the CR2 register:
   - Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.
   - Set the PWM bit.
   - Select the timer clock (CC[1:0]) (see *Table 50: Clock control bits*).

Figure 55. Pulse width modulation cycle



If OLVL1=1 and OLVL2=0 the length of the positive pulse is the difference between the OC2R and OC1R registers.

If OLVL1=OLVL2 a continuous signal will be seen on the OCMP1 pin.

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

$$OCiR\ Value = \frac{t * f_{CPU}}{PRESC} - 5$$

Where:

- t = Signal or pulse 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 *Table 50: Clock control bits*)

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

$$OCiR = t * f_{EXT} - 5$$

Where:

- t = Signal or pulse period (in seconds)
- f_{EXT} = External timer clock frequency (in Hertz)

The Output Compare 2 event causes the counter to be initialized to FFFCh (See *Figure 54*)

Note: 1 The OCF1 and OCF2 bits cannot be set by hardware in PWM mode; therefore, the Output Compare interrupt is inhibited.

2 The ICF1 bit is set by hardware when the counter reaches the OC2R value; it can produce a timer interrupt if the ICIE bit is set and the I bit is cleared.

3 In PWM mode, the ICAP1 pin cannot be used to perform an input capture because it is disconnected to the timer. The ICAP2 pin can be used to perform an input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each period, and ICF1 can also generate an interrupt if ICIE is set.

4 When the pulse-width modulation (PWM) and One-pulse mode (OPM) bits are both set, the PWM mode is the only active one.

11.4.7 Interrupts

Table 53. Interrupt events

Interrupt event	Event flag	Enable control bit	Exit from Wait	Exit from Halt
SPI end of transfer event	SPIF	SPIE	Yes	Yes
Master mode fault event	MODF			No
Overrun error	OVR			

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

11.4.8 Register description

SPI control register (SPICR)

Reset value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0
Read/Write							

Bit 7 = **SPIE** *Serial Peripheral Interrupt Enable*

This bit is set and cleared by software.

0: Interrupt is inhibited
1: An SPI interrupt is generated whenever an End of Transfer event, Master Mode Fault or Overrun error occurs (SPIF = 1, MODF = 1 or OVR = 1 in the SPICSR register)

Bit 6 = **SPE** *Serial Peripheral Output Enable*

This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see *Master mode fault (MODF)*). The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.

0: I/O pins free for general purpose I/O
1: SPI I/O pin alternate functions enabled

Bit 5 = **SPR2** *Divider Enable*

This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to *Table 54: SPI master mode SCK frequency*.

0: Divider by 2 enabled
1: Divider by 2 disabled

Note: This bit has no effect in slave mode.

11.5 SCI serial communication interface

11.5.1 Introduction

The serial communications interface (SCI) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The SCI offers a very wide range of baud rates using two baud rate generator systems.

11.5.2 Main features

- Full-duplex, asynchronous communications
- NRZ standard format (Mark/Space)
- Dual baud rate generator systems
- Independently programmable transmit and receive baud rates up to 500,000 baud
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- 2 receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- Separate enable bits for Transmitter and Receiver
- 4 error detection flags:
  - Overrun error
  - Noise error
  - Frame error
  - Parity error
- 5 interrupt sources with flags:
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error detected
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode

11.5.3 General description

The interface is externally connected to another device by three pins (see *Figure 63*). Any SCI bidirectional communication requires a minimum of two pins: Receive Data In (RDI) and Transmit Data Out (TDO):

- SCLK: Transmitter clock output. This pin outputs the transmitter data clock for synchronous transmission (no clock pulses on start bit and stop bit, and a software option to send a clock pulse on the last data bit). This can be used to control

cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: Data is not received
1: Received data is ready to be read

Bit 4 = **IDLE** *Idle line detect.*

This bit is set by hardware when an Idle Line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Idle Line is detected
1: Idle Line is detected

Note: The IDLE bit is not set again until the RDRF bit has been set itself (that is, a new idle line occurs).

Bit 3 = **OR** *Overrun error.*

This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RDRF = 1. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Overrun error
1: Overrun error is detected

Note: When this bit is set, the RDR register content is not lost but the shift register is overwritten.

Bit 2 = **NF** *Noise flag.*

This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No noise is detected
1: Noise is detected

Note: This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.

Bit 1 = **FE** *Framing error.*

This bit is set by hardware when a desynchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).

0: No Framing error is detected
1: Framing error or break character is detected

Note: This bit does not generate an interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt. If the word currently being transferred causes both frame error and overrun error, it is transferred and only the OR bit is set.

Bit 0 = **PE** *Parity error.*

This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.

0: No parity error
1: Parity error

Control register 1 (SCICR1)

Reset value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE
Read/Write							

Bit 7 = **R8** *Receive data bit 8.*

This bit is used to store the 9th bit of the received word when M = 1.

Bit 6 = **T8** *Transmit data bit 8.*

This bit is used to store the 9th bit of the transmitted word when M = 1.

Bit 5 = **SCID** *Disabled for low power consumption*

When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption.This bit is set and cleared by software.

0: SCI enabled
1: SCI prescaler and outputs disabled

Bit 4 = **M** *Word length.*

This bit determines the word length. It is set or cleared by software.
0: 1 Start bit, 8 Data bits, 1 Stop bit
1: 1 Start bit, 9 Data bits, 1 Stop bit

Note: The M bit must not be modified during a data transfer (both transmission and reception).

Bit 3 = **WAKE** *Wakeup method.*

This bit determines the SCI Wake-Up method, it is set or cleared by software.
0: Idle Line
1: Address Mark

Bit 2 = **PCE** *Parity control enable.*

This bit selects the hardware parity control (generation and detection). When the parity control is enabled, the computed parity is inserted at the MSB position (9th bit if M = 1; 8th bit if M = 0) and parity is checked on the received data. This bit is set and cleared by software. Once it is set, PCE is active after the current byte (in reception and in transmission).

0: Parity control disabled
1: Parity control enabled

Bit 1 = **PS** *Parity selection.*

This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity is selected after the current byte.

0: Even parity
1: Odd parity

Table 59. SCP[1:0] configuration

PR prescaling factor	SCP1	SCP0
1	0	0
3		1
4	1	0
13		1

Bits 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*

These 3 bits, in conjunction with the SCP1 and SCP0 bits, define the total division applied to the bus clock to yield the transmit rate clock in conventional baud rate Generator mode.

Table 60. SCT[2:0] configuration

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2			1
4		1	0
8			1
16	1	0	0
32			1
64		1	0
128			1

*Note: This TR factor is used only when the ETPR fine tuning factor is equal to 00h; otherwise, TR is replaced by the (TR*ETPR) dividing factor.*

Bits 2:0 = **SCR[2:0]** *SCI Receiver rate divisor.*

These 3 bits, in conjunction with the SCP1 and SCP0 bits, define the total division applied to the bus clock to yield the receive rate clock in conventional Baud Rate Generator mode.

Table 61. SCR[2:0] configuration

RR dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2			1
4		1	0
8			1
16	1	0	0
32			1
64		1	0
128			1

11.6 I²C bus interface (I2C)

11.6.1 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).

11.6.2 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

11.6.3 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.

Bit 4 = **AF** *Acknowledge failure*.

This bit is set by hardware when no acknowledge is returned. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while AF=1 but by other flags (SB or BTF) that are set at the same time.

0: No acknowledge failure
1: Acknowledge failure

Bit 3 = **STOPF** *Stop detection (Slave mode).*

This bit is set by hardware when a Stop condition is detected on the bus after an acknowledge (if ACK=1). An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while STOPF=1.

0: No Stop condition detected
1: Stop condition detected

Bit 2 = **ARLO** *Arbitration lost*.

This bit is set by hardware when the interface loses the arbitration of the bus to another master. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

After an ARLO event the interface switches back automatically to Slave mode (M/SL=0).

The SCL line is not held low while ARLO=1.

0: No arbitration lost detected
1: Arbitration lost detected

Note: In a Multimaster environment, when the interface is configured in Master Receive mode it does not perform arbitration during the reception of the Acknowledge Bit. A mishandling of the ARLO bit from the I2CSR2 register may occur when a second master simultaneously requests the same data from the same slave, and the I^2C master does not acknowledge the data. The ARLO bit is then left at 0 instead of being set.

Bit 1 = **BERR** *Bus error.*

This bit is set by hardware when the interface detects a misplaced Start or Stop condition. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while BERR=1.

0: No misplaced Start or Stop condition
1: Misplaced Start or Stop condition

Note: If a Bus Error occurs, a Stop or a repeated Start condition should be generated by the Master to re-synchronize communication, get the transmission acknowledged and the bus released for further communication

Bit 0 = **GCAL** *General Call (Slave mode).*

This bit is set by hardware when a general call address is detected on the bus while ENGC=1. It is cleared by hardware detecting a Stop condition (STOPF=1) or when the interface is disabled (PE=0).

0: No general call address detected on the bus
1: General call address detected on the bus

13.6 Clock and timing characteristics

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A.

Table 95. General timings

Symbol	Parameter (1)	Conditions	Min	Typ (2)	Max	Unit
$t_{c(INST)}$	Instruction cycle time	f_{CPU}=8 MHz	2	3	12	t_{CPU}
			250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time (3) $t_{v(IT)} = \Delta t_{c(INST)} + 10$	f_{CPU}=8 MHz	10		22	t_{CPU}
			1.25		2.75	µs

1. Guaranteed by Design. Not tested in production.
2. Data based on typical application software.
3. Time measured between interrupt event and interrupt vector fetch. Dtc(INST) is the number of t_{CPU} cycles needed to finish the current instruction execution.

Table 96. External clock source

Symbol	Parameter	Conditions	Min	Max	Unit
V_{OSC1H}	OSC1 input pin high level voltage	see *Figure 96*	0.7xV_{DD}	V_{DD}	V
V_{OSC1L}	OSC1 input pin low level voltage		V_{SS}	0.3xV_{DD}	
$t_{w(OSC1H)}$ $t_{w(OSC1L)}$	OSC1 high or low time (1)		15		ns
$t_{r(OSC1)}$ $t_{f(OSC1)}$	OSC1 rise or fall time (1)			15	
I_L	OSCx Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$		±1	µA

1. Data based on design simulation and/or technology characteristics, not tested in production.

Figure 96. Typical application with an external clock source

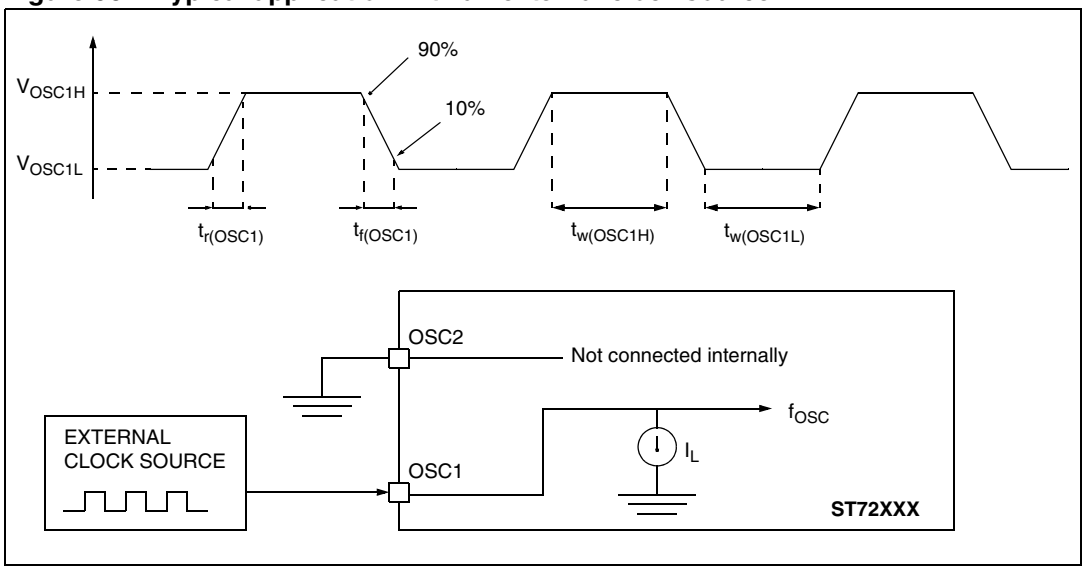


Figure 108. Typical V_{OL} vs. V_{DD} (HS I/Os, I_{IO} = 2 mA)

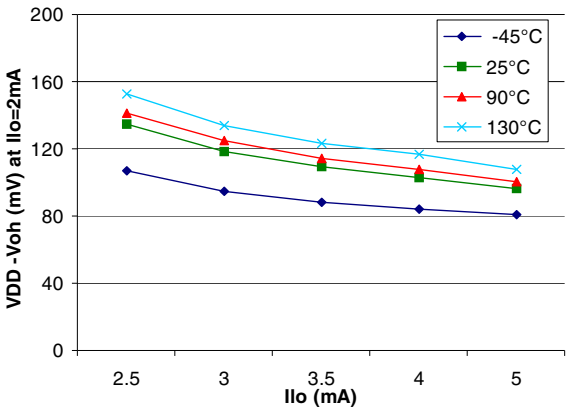


Figure 109. Typical V_{OL} vs. V_{DD} (HS I/Os, I_{IO} = 12 mA)

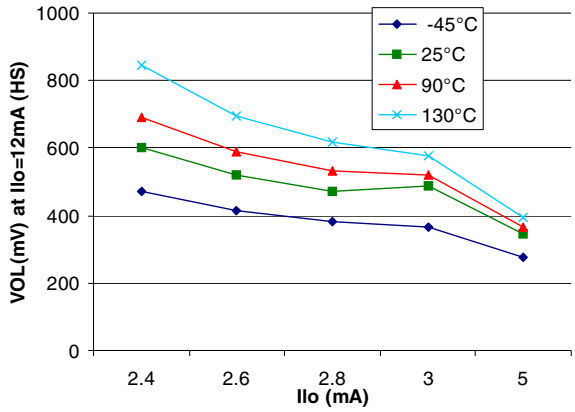
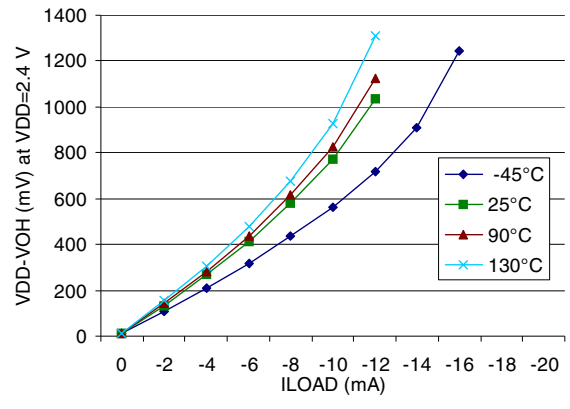


Figure 110. Typical $V_{DD} - v_{OH}$ at V_{DD} = 2.4 V (std I/Os)



13.11 Communication interface characteristics

13.11.1 I²C and I²C3SNS interfaces

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

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (SDAI and SCLI). The ST7 I²C and I2C3SNS interfaces meet the electrical and timing requirements of the Standard I²C communication protocol.

T_A = -40°C to 85°C, unless otherwise specified.

Table 110. I²C and I²C3SNS interfaces characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCL}	I²C SCL frequency	f_{CPU}=4 MHz to 8 MHz (1), V_{DD}= 2.7 V to 5.5 V		400	kHz
$f_{SCL3SNS}$	I²C3SNS SCL frequency (2)			400	kHz

1. The I²C and I2C3SNS interfaces will not function below the minimum clock speed of 4 MHz.
2. Not tested in production within the whole operating range. Guaranteed by design/validation test results.

The following table gives the values to be written in the I2CCCR register to obtain the required I²C SCL line frequency.

Table 111. SCL frequency table (multimaster I²C interface) (1)

f_{SCL}	I2CCCR value							
	f_{CPU} = 4 MHz.				f_{CPU} = 8 MHz.			
	V_{DD} = 3.3 V		V_{DD} = 5 V		V_{DD} = 3.3 V		V_{DD} = 5 V	
	R_P=3.3kΩ	R_P=4.7kΩ	R_P=3.3kΩ	R_P=4.7kΩ	R_P=3.3kΩ	R_P=4.7kΩ	R_P=3.3kΩ	R_P=4.7kΩ
400	NA	NA	NA	NA	84h	84h	84h	84h
300	NA	NA	NA	NA	86h	86h	85h	87h
200	84h	84h	84h	84h	8Ah	8Ah	8Bh	8Ch
100	11h	10h	11h	11h	25h	24h	28h	28h
50	25h	24h	25h	26h	4Bh	4Ch	53h	54h
20	60h	5Fh	60h	62h	FFh	FFh	FFh	FFh

1. R_P = External pull-up resistance, f_{SCL} = I²C speed, NA = Not achievable.

Note: For speeds around 200 kHz, achieved speed can have ±5% tolerance; for other speed ranges, achieved speed can have ±2% tolerance.
The above variations depend on the accuracy of the external components used.