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	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K 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/st72f344s4t6tr

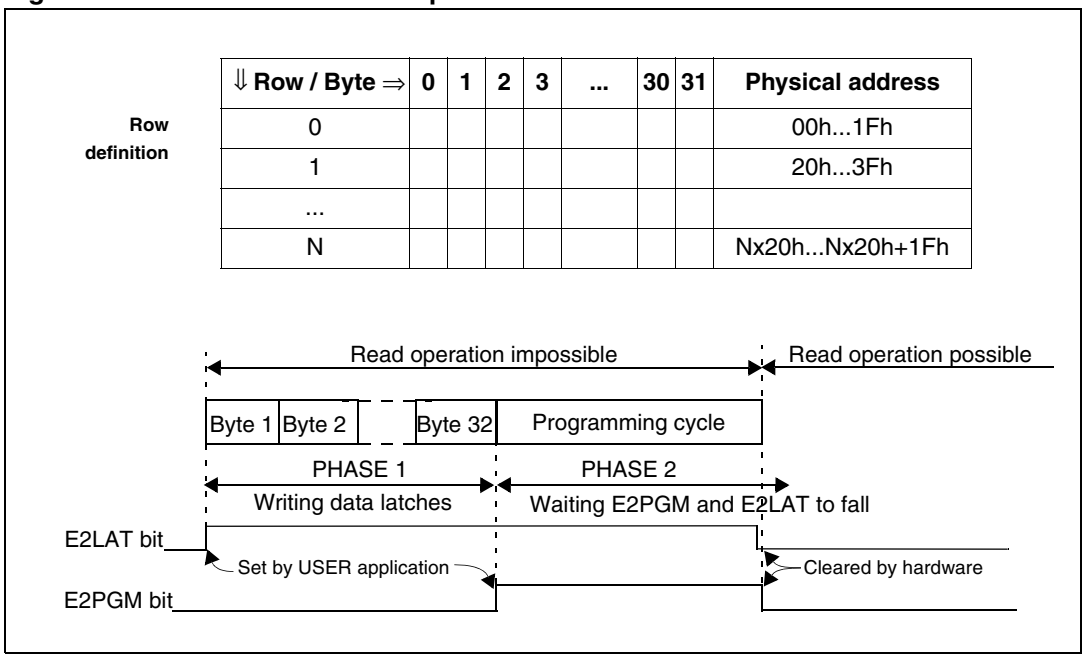
List of figures

Table 3. Device pin description (continued)

Pin n°			Pin name	Type	Level		Port						Main function (after reset)	Alternate function
							Input (1)				Output			
LQFP32	LQFP44	LQFP48			Input	Output	float	wpu	int	ana	OD	PP		
27	1	2	PE1/RDI	I/O	C_T		X		ei0		X	X	Port E1	SCI receive data in
28	2	3	PB0	I/O	C_T		X	ei2			X	X	Port B0	
-	3	4	PB1 (3)	I/O	C_T		X	ei2			X	X	Port B1	
-	4	5	PB2 (3)	I/O	C_T		X	ei2			X	X	Port B2	
29	5	6	PB3	I/O	C_T		X		ei2		X	X	Port B3	
30	6	7	PB4 (HS)	I/O	C_T	HS	X	ei3			X	X	Port B4	
31	7	8	PD0/AIN0	I/O	C_T		X	X		X	X	X	Port D0	ADC analog input 0
32	8	9	PD1/AIN1	I/O	C_T		X	X		X	X	X	Port D1	ADC analog input 1
-	9	10	PD2/AIN2	I/O	C_T		X	X		X	X	X	Port D2	ADC analog input 2
-	10	11	PD3/AIN3	I/O	C_T		X	X		X	X	X	Port D3	ADC analog input 3
-	11	12	PD4/AIN4	I/O	C_T		X	X		X	X	X	Port D4	ADC analog input 4
-	12	13	PD5/AIN5	I/O	C_T		X	X		X	X	X	Port D5	ADC analog input 5

1. In the interrupt input column, “eiX” defines the associated external interrupt vector. If the weak pull-up column (wpu) is merged with the interrupt column (int), then the I/O configuration is pull-up interrupt input, else the configuration is floating interrupt input.
2. It is mandatory to connect all available V_{DD} and V_{DDA} pins to the supply voltage and all V_{SS} and V_{SSA} pins to ground.
3. Pulled-up by hardware when not present on the package.
4. In the open drain output column, “T” defines a true open drain I/O (P-Buffer and protection diode to V_{DD} are not implemented).
5. Internal weak pull-down.

Figure 9. Data EEPROM write operation



Note: If a programming cycle is interrupted (by reset action), the integrity of the data in memory will not be guaranteed.

5.4 Power saving modes

5.4.1 Wait mode

The data EEPROM can enter Wait mode on execution of the WFI instruction of the microcontroller or when the microcontroller enters Active Halt mode.The data EEPROM will immediately enter this mode if there is no programming in progress, otherwise the data EEPROM will finish the cycle and then enter Wait mode.

5.4.2 Active-halt mode

Refer to *Wait mode*.

5.4.3 Halt mode

The data EEPROM immediately enters Halt mode if the microcontroller executes the Halt instruction. Therefore the EEPROM will stop the function in progress, and data may be corrupted.

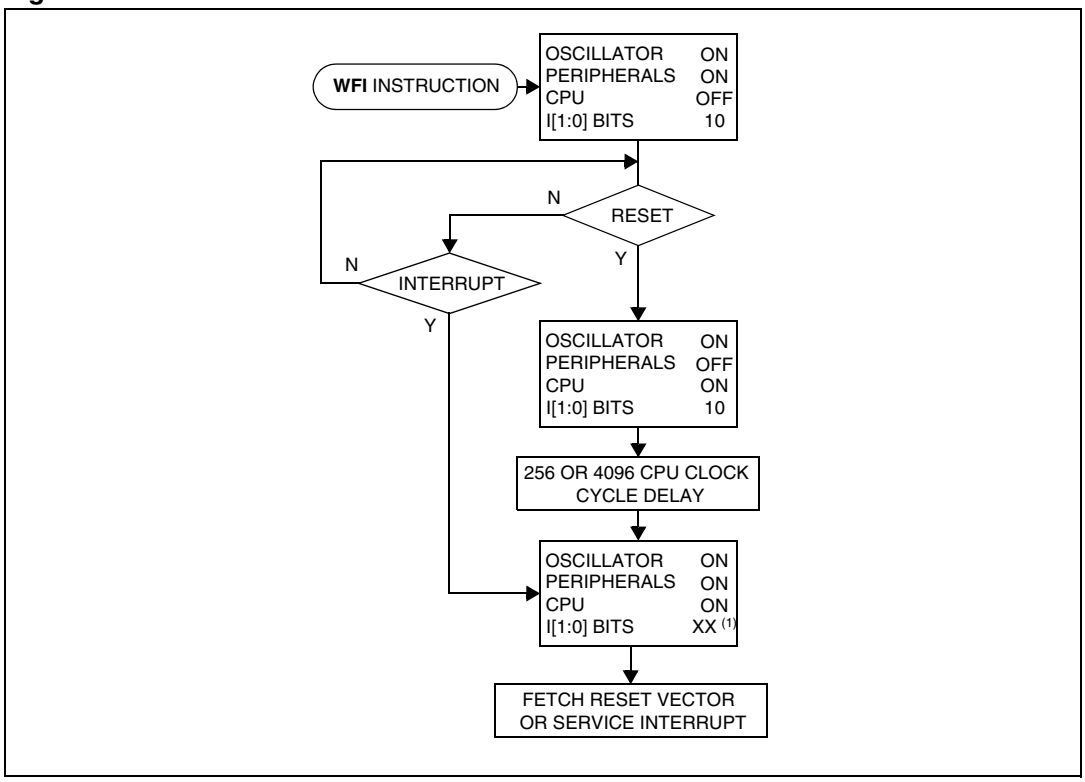
5.5 Access error handling

If a read access occurs while E2LAT = 1, then the data bus will not be driven.

If a write access occurs while E2LAT = 0, then the data on the bus will not be latched.

If a programming cycle is interrupted (by reset action), the integrity of the data in memory will not be guaranteed.

Figure 27. Wait mode flowchart



Note: Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped.

9.4 Halt mode

The Halt mode is the lowest power consumption mode of the MCU. It is entered by executing the 'HALT' instruction when the OIE bit of the Main Clock Controller Status register (MCCSR) is cleared (see *Section 11.2: Main clock controller with real-time clock and beeper (MCC/RTC)* for more details on the MCCSR register) and when the AWUEN bit in the AWUCSR register is cleared.

The MCU can exit Halt mode on reception of either a specific interrupt (see *Table 17: Interrupt mapping*) or a reset. When exiting Halt mode by means of a reset or an interrupt, the oscillator is immediately turned on and the 256 or 4096 CPU cycle delay is used to stabilize the oscillator. After the start up delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see *Figure 29*).

When entering Halt 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 Halt mode, the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. All peripherals are not clocked except the ones which get their clock supply from another clock generator (such as an external or auxiliary oscillator).

The compatibility of Watchdog operation with Halt mode is configured by the "WDGHALT" option bit of the option byte. The HALT instruction when executed while the Watchdog

system is enabled, can generate a Watchdog reset (see *Section 11.1: Window watchdog (WWDG)* for more details).

Figure 28. Halt timing overview

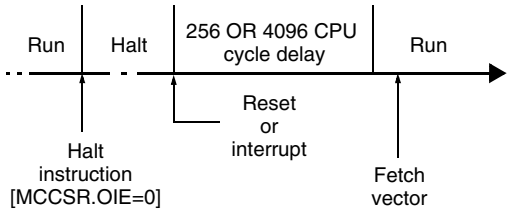
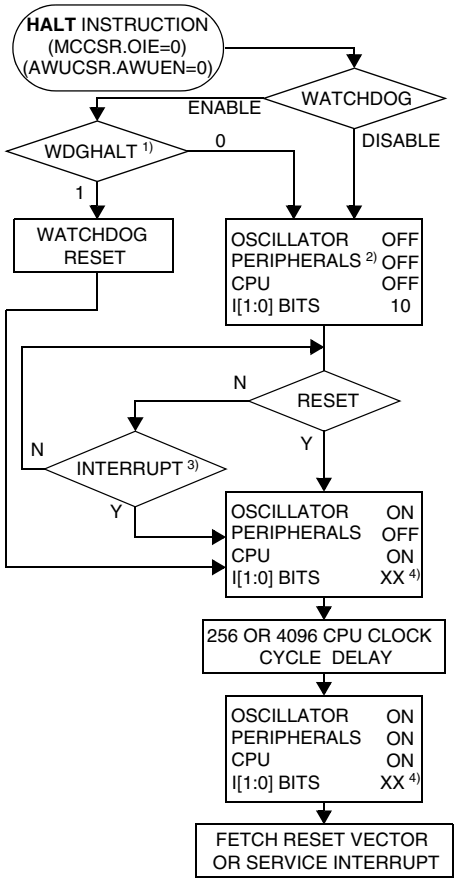


Figure 29. Halt mode flowchart



Note: 1 WDGHALT is an option bit. See Section 15.1: Option bytes for more details.

2 Peripheral clocked with an external clock source can still be active.

3 Only some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to Table 17: Interrupt mapping for more details.

4 Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped.

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

When a match is found between the Output Compare register and the free running counter, the output compare function:

- Assigns pins with a programmable value if the OCIE bit is set
- Sets a flag in the status register
- Generates an interrupt if enabled

Two 16-bit registers, Output Compare Register 1 (OC1R) and Output Compare Register 2 (OC2R), contain the value to be compared to the counter register each timer clock cycle.

Table 46. OC*i*R register

	MS Byte	LS Byte
OC*i*R	OC*i*HR	OC*i*LR

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC*i*R value to 8000h.

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

Procedure:

To use the output compare function, select the following in the CR2 register:

- Set the OC*i*E bit if an output is needed then the OCMP*i* pin is dedicated to the output compare *i* signal.
- Select the timer clock (CC[1:0]) (see *Table 50: Clock control bits*).

And select the following in the CR1 register:

- Select the OLVL*i* bit to applied to the OCMP*i* pins after the match occurs.
- Set the OCIE bit to generate an interrupt if it is needed.

When a match is found between OCRi register and CR register:

- OCF*i* bit is set.
- The OCMP*i* pin takes OLVL*i* bit value (OCMP*i* pin latch is forced low during reset).
- A timer interrupt is generated if the OCIE bit is set in the CR2 register and the I bit is cleared in the CC register (CC).

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

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

Where:

- Δt = Output compare period (in seconds)
- f_{CPU} = CPU clock frequency (in Hertz)
- PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see *Table 50*)

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

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

Where:

- Δt = Output compare period (in seconds)
- f_{EXT} = External timer clock frequency (in Hertz)

11.3.6 Summary of timer modes

Table 49. Timer modes

Modes	Available resources			
	Input Capture 1	Input Capture 2	Output Compare 1	Output Compare 2
Input Capture (1 and/or 2)	Yes	Yes	Yes	Yes
Output Compare (1 and/or 2)	Yes	Yes	Yes	Yes
One-pulse mode	No	Not recommended (1)	No	Partially (2)
PWM mode	No	Not recommended (3)	No	No

1. See note 4 in *One-pulse mode*.
2. See note 5 in *One-pulse mode*.
3. See note 4 in *Pulse-width modulation mode*.

11.3.7 Register description

Each timer is associated with three control and status registers, and with six pairs of data registers (16-bit values) related to the two input captures, the two output compares, the counter and the alternate counter.

Control register 1 (CR1)

Read/Write

Reset value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
Read/Write							

Bit 7 = **ICIE** *Input Capture Interrupt Enable.*

0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the ICF1 or ICF2 bit of the SR register is set.

Bit 6 = **OCIE** *Output Compare Interrupt Enable.*

0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the OCF1 or OCF2 bit of the SR register is set.

Bit 5 = **TOIE** *Timer Overflow Interrupt Enable.*

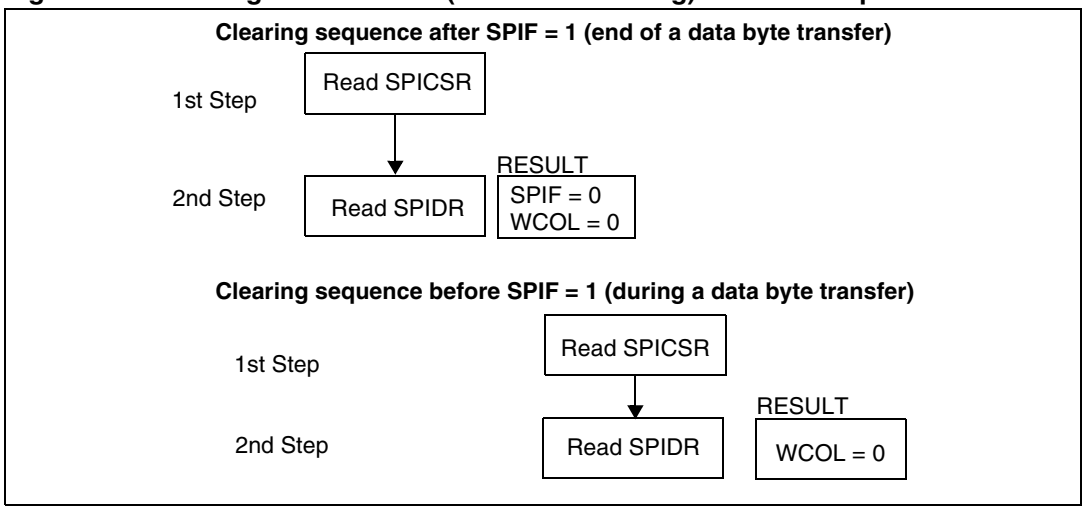
0: Interrupt is inhibited.
1: A timer interrupt is enabled whenever the TOF bit of the SR register is set.

Bit 4 = **FOLV2** *Forced Output Compare 2.*

This bit is set and cleared by software.

0: No effect on the OCMP2 pin.
1: Forces the OLVL2 bit to be copied to the OCMP2 pin, if the OC2E bit is set and even if there is no successful comparison.

Figure 61. Clearing the WCOL bit (write collision flag) software sequence



1. Writing to the SPIDR register instead of reading it does not reset the WCOL bit.

Single master and multimaster configurations

There are two types of SPI systems: Single master system and Multimaster system.

- Single master system

  A typical single master system may be configured using a device as the master and four devices as slaves (see *Figure 62*).

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

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

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

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

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

- Multimaster system

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

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

11.5.4 Functional description

The block diagram of the Serial Control Interface, is shown in *Figure 63*. It contains six dedicated registers:

- 2 control registers (SCICR1 and SCICR2)
- A status register (SCISR)
- A baud rate register (SCIBRR)
- An extended prescaler receiver register (SCIERPR)
- An extended prescaler transmitter register (SCIETPR)

Refer to the register descriptions in *Section 11.5.7* 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 64*).

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.

SCI clock tolerance

During reception, each bit is sampled 16 times. The majority of the 8th, 9th and 10th samples is considered as the bit value. For a valid bit detection, all the three samples should have the same value otherwise the noise flag (NF) is set. For example: if the 8th, 9th and 10th samples are 0, 1 and 1 respectively, then the bit value is “1”, but the Noise Flag bit is set because the three samples values are not the same.

Consequently, the bit length must be long enough so that the 8th, 9th and 10th samples have the desired bit value. This means the clock frequency should not vary more than 6/16 (37.5%) within one bit. The sampling clock is resynchronized at each start bit, so that when receiving 10 bits (one start bit, 1 data byte, 1 stop bit), the clock deviation must not exceed 3.75%.

Note: The internal sampling clock of the microcontroller samples the pin value on every falling edge. Therefore, the internal sampling clock and the time the application expects the sampling to take place may be out of sync. For example: If the baud rate is 15.625 kbaud (bit length is 64 µs), then the 8th, 9th and 10th samples will be at 28 µs, 32 µs and 36 µs respectively (the first sample starting ideally at 0 µs). But if the falling edge of the internal clock occurs just before the pin value changes, the samples would then be out of sync by ~4 µs. This means the entire bit length must be at least 40 µs (36 µs for the 10th sample + 4 µs for synchronization with the internal sampling clock).

Clock deviation causes

The causes which contribute to the total deviation are:

- D_{TRA}: Deviation due to transmitter error (Local oscillator error of the transmitter or the transmitter is transmitting at a different baud rate).
- D_{QUANT}: Error due to the baud rate quantization of the receiver.
- D_{REC}: Deviation of the local oscillator of the receiver: This deviation can occur during the reception of one complete SCI message assuming that the deviation has been compensated at the beginning of the message.
- D_{TCL}: Deviation due to the transmission line (generally due to the transceivers)

All the deviations of the system should be added and compared to the SCI clock tolerance:

$D_{TRA} + D_{QUANT} + D_{REC} + D_{TCL} < 3.75\%$

Noise error causes

See also description of noise error in *Receiver*.

- Start bit: the noise flag (NF) is set during start bit reception if one of the following conditions occurs:
  - A valid falling edge is not detected. A falling edge is considered to be valid if the three consecutive samples before the falling edge occurs are detected as '1' and,

Bit 1 = **RWU** *Receiver wake-up.*

This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.

0: Receiver in active mode
1: Receiver in mute mode

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

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

Bit 0 = **SBK** *Send break.*

This bit set is used to send break characters. It is set and cleared by software.

0: No break character is transmitted
1: Break characters are transmitted

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

Data register (SCIDR)

Reset value: Undefined

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

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
Read/Write							

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).
The TDR register provides the parallel interface between the internal bus and the output shift register (see *Figure 63*).
The RDR register provides the parallel interface between the input shift register and the internal bus (see *Figure 63*).

Baud rate register (SCIBRR)

Reset value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0
Read/Write							

Bits 7:6 = **SCP[1:0]** *First SCI Prescaler*

These 2 prescaling bits allow several standard clock division ranges as shown in *Figure 59*.

Note: In 10-bit addressing mode, to switch the master to Receiver mode, the software must generate a repeated Start condition and resend the header sequence with the least significant bit set (11110xx1).

Master receiver

Following the address transmission and after 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 69* 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 69* 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 pulse of each 9-bit transaction:
  Single Master Mode
  If a Start or Stop is issued during the first pulse of a 9-bit transaction, the BERR flag will not be set and transfer will continue however the BUSY flag will be reset. To work around this, slave devices should issue a NACK when they receive a misplaced Start or Stop. The reception of a NACK or BUSY by the master in the middle of communication gives the possibility to re-initiate transmission.
  Multimaster Mode
  Normally the BERR bit would be set whenever unauthorized transmission takes place while transfer is already in progress. However, an issue will arise if an external master generates an unauthorized Start or Stop while the I^2C master is on the first pulse of a 9-bit transaction. It is possible to work around this by polling the BUSY bit during I^2C

1. The I^2C 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 I-bit in the CC register is reset (RIM instruction).

11.6.7 Register description

I^2C control register (CR)

Reset value: 0000 0000 (00h)

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

Bits 7:6 = Reserved. Forced to 0 by hardware.

Bit 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 I^2C interface, write the CR register **TWICE** with PE=1 as the first write only activates the interface (only PE is set).*

Bit 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 I2C standard, when GCAL addressing is enabled, an I2C slave can only receive data. It will not transmit data to the master.

Bit 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

11.8.3 Functional description

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_{AREF} (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 *Section 13: Electrical characteristics*.

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 converter configuration

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 CS[3:0] bits to assign the analog channel to convert.

Starting the conversion

In the ADCCSR register:

- 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 resets the EOC bit.

To read the 10 bits, perform the following steps:

1. Poll the EOC bit
2. Read the ADCDRL register
3. Read the ADCDRH register. This clears EOC automatically.

Note: The data is not latched, so both the low and the high data register must be read before the next conversion is complete, so it is recommended to disable interrupts while reading the conversion result.

To read only 8 bits, perform the following steps:

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

13.4 Internal RC oscillator characteristics

Table 92. Internal RC oscillator characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal RC oscillator frequency (1)	RCCR = FF (reset value), T_A = 25 °C, V_{DD} = 5 V		625		kHz
		RCCR = RCCR0 (2), T_A = 25 °C, V_{DD} = 5 V		1000		
		RCCR = FF (reset value), T_A = 25 °C, V_{DD} = 3 V		612		
		RCCR = RCCR1(2), T_A = 25 °C, V_{DD} = 3 V		1000		
ACC_{RC}	Accuracy of Internal RC oscillator with RCCR=RCCR0 (2)	T_A = 25 °C, V_{DD} = 5 V	-1		+1	%
		T_A = 25 °C, V_{DD} = 4.5 to 5.5 V (3)	-1		+1	%
		T_A = 25 to +85 °C, V_{DD} = 5 V (3)	-3		+3	%
		T_A = 25 to +85 °C, V_{DD} = 4.5 to 5.5 V (3)	-3.5		+3.5	%
		T_A = -40 to +25 °C, V_{DD} = 4.5 to 5.5 V (3)	-3		+7	%
$I_{DD(RC)}$	RC oscillator current consumption	T_A = 25 °C, V_{DD} = 5 V		600 (3)		µA
$t_{su(RC)}$	RC oscillator setup time	T_A = 25 °C, V_{DD} = 5 V			10 (2)	µs

1. If the RC oscillator clock is selected, to improve clock stability and frequency accuracy, it is recommended to place a decoupling capacitor, typically 100 nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device. It is also recommended to perform the calibration on board.
2. See *Internal RC oscillator*.
3. Expected results. Data based on characterization, not tested in production.

Figure 88. Typical RC frequency vs. RCCR

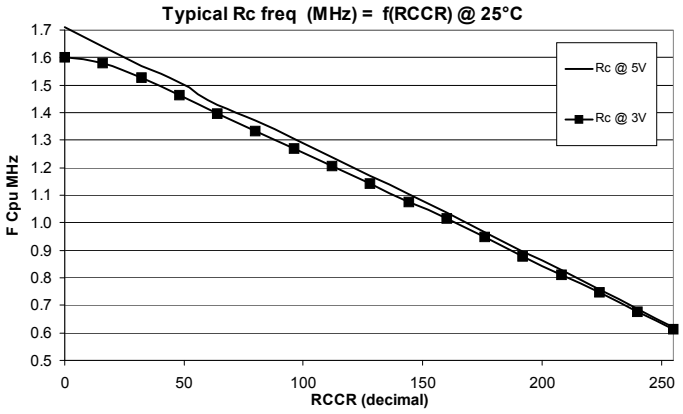
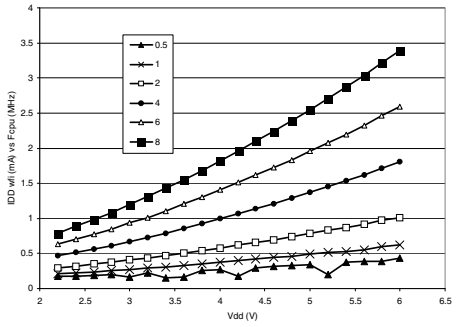
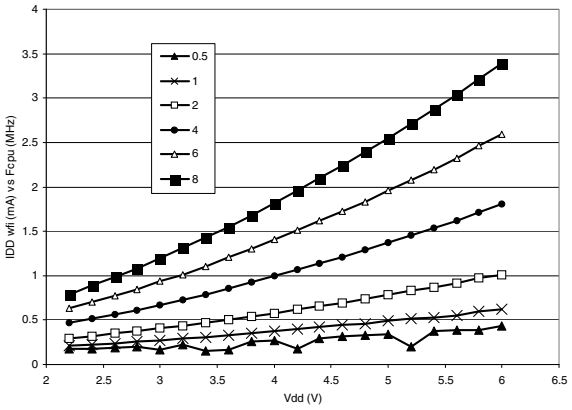


Figure 92. Typical I_{DD} in Wait vs. f_{CPU}



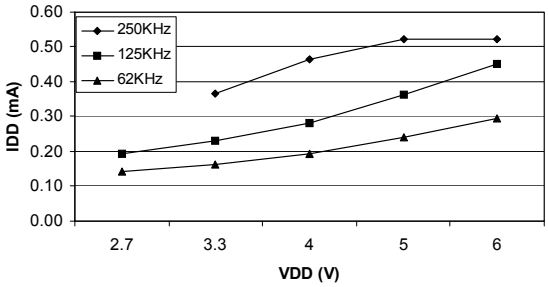
1. Graph displays data beyond the normal operating range of 3 V - 5.5 V.

Figure 93. Typical I_{DD} in Wait at f_{CPU} = 8 MHz



1. Graph displays data beyond the normal operating range of 3 V - 5.5 V.

Figure 94. Typical I_{DD} in Slow-wait vs. f_{CPU}



1. Graph displays data beyond the normal operating range of 3 V - 5.5 V.

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

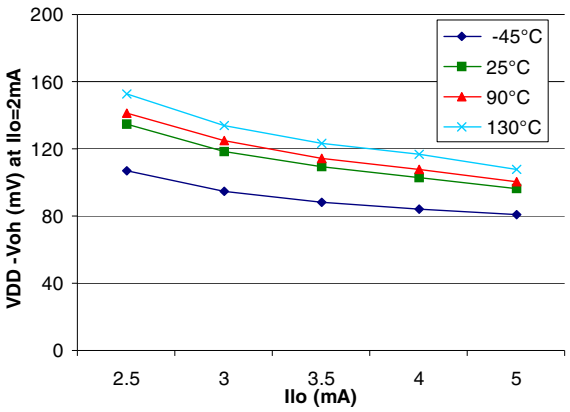


Figure 109. Typical V_{OL} vs. V_{DD} (HS I/Os, I_{IO} = 12 mA)

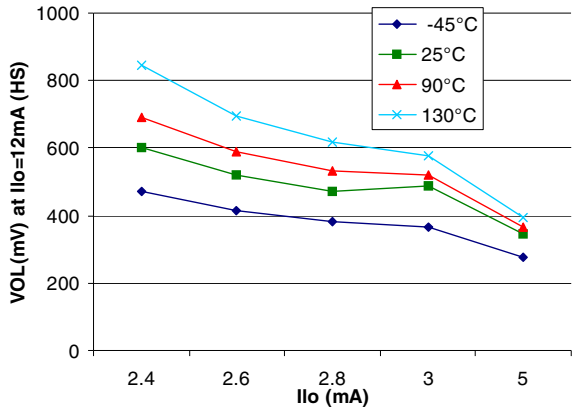
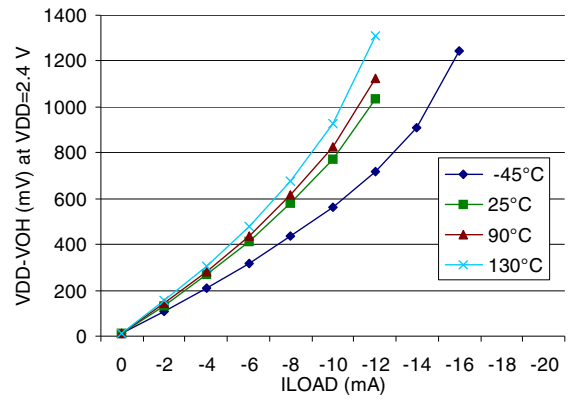


Figure 110. Typical $V_{DD} - v_{OH}$ at V_{DD} = 2.4 V (std I/Os)



13.12 10-bit ADC characteristics

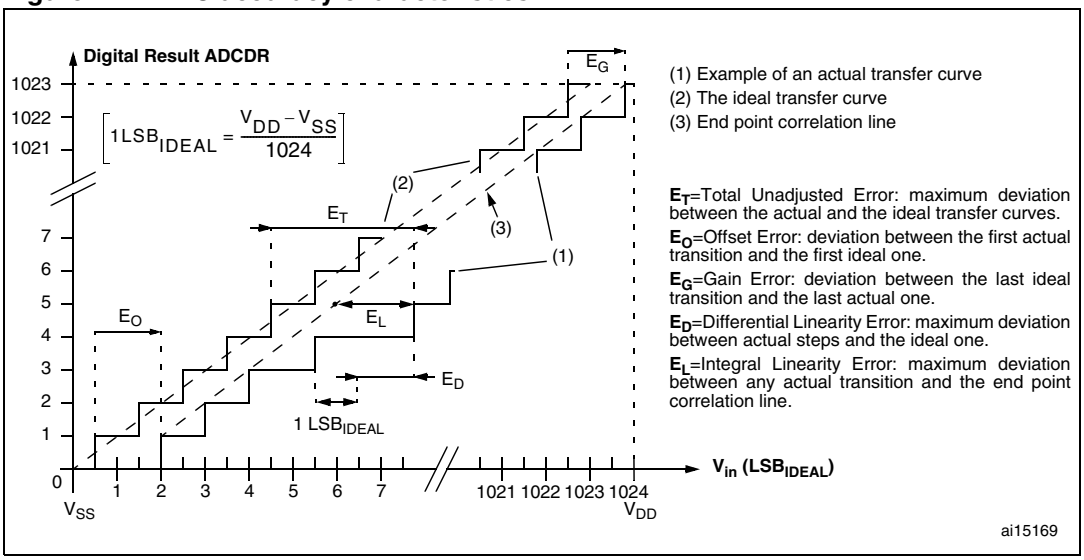
T_A = -40 °C to 85 °C, unless otherwise specified

Table 112. ADC accuracy

Symbol	Parameter	Conditions (1)(2)	Typ	Max (3)	Unit
$\|E_T\|$	Total unadjusted error	f_{CPU} = 8 MHz, f_{ADC} = 4 MHz, R_{AIN} < 10kΩ, V_{DD} = 2.7 V to 5.5 V	4	8	LSB
$\|E_O\|$	Offset error		-1	-2	
$\|E_G\|$	Gain Error		-2	-4	
$\|E_D\|$	Differential linearity error		3	6	

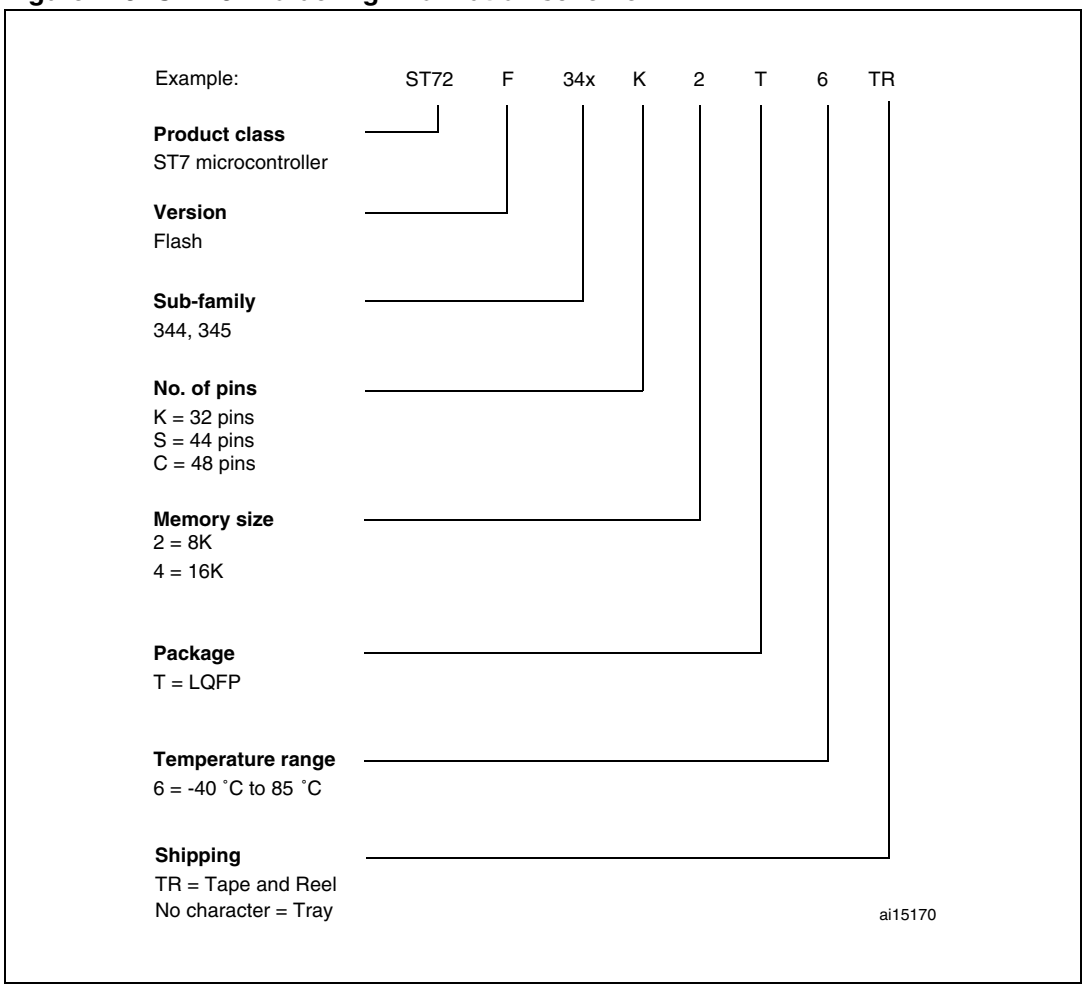
1. Data based on characterization results over the whole temperature range.
2. ADC accuracy vs. negative injection current: Injecting negative current on any of the analog input pins may reduce the accuracy of the conversion being performed on another analog input.
The effect of negative injection current on robust pins is specified in *Section 13.10*
Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in *Section 13.9* does not affect the ADC accuracy.
3. *Data based on characterization results, monitored in production to guarantee 99.73% within ± max value from -40 °C to +125 °C (± 3σ distribution limits).*

Figure 117. ADC accuracy characteristics



15.2 Device ordering information

Figure 123. ST7234x ordering information scheme



For a list of available options (e.g. memory size, package) and orderable part numbers or for further information on any aspect of this device, please contact the nearest ST sales office.

15.3 Development tools

Development tools for the ST7 microcontrollers include a complete range of hardware systems and software tools from STMicroelectronics and third-party tool suppliers. The range of tools includes solutions to help you evaluate microcontroller peripherals, develop and debug your application, and program your microcontrollers.

15.3.1 Starter kits

ST offers complete, affordable **starter kits**. Starter kits are complete, affordable hardware/software tool packages that include features and samples to help you quickly start developing your application.