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, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	32
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 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/st72f325j6tae

Contents

5 Central processing unit (CPU)

5.1 Introduction

This CPU has a full 8-bit architecture and contains six internal registers allowing efficient 8-bit data manipulation.

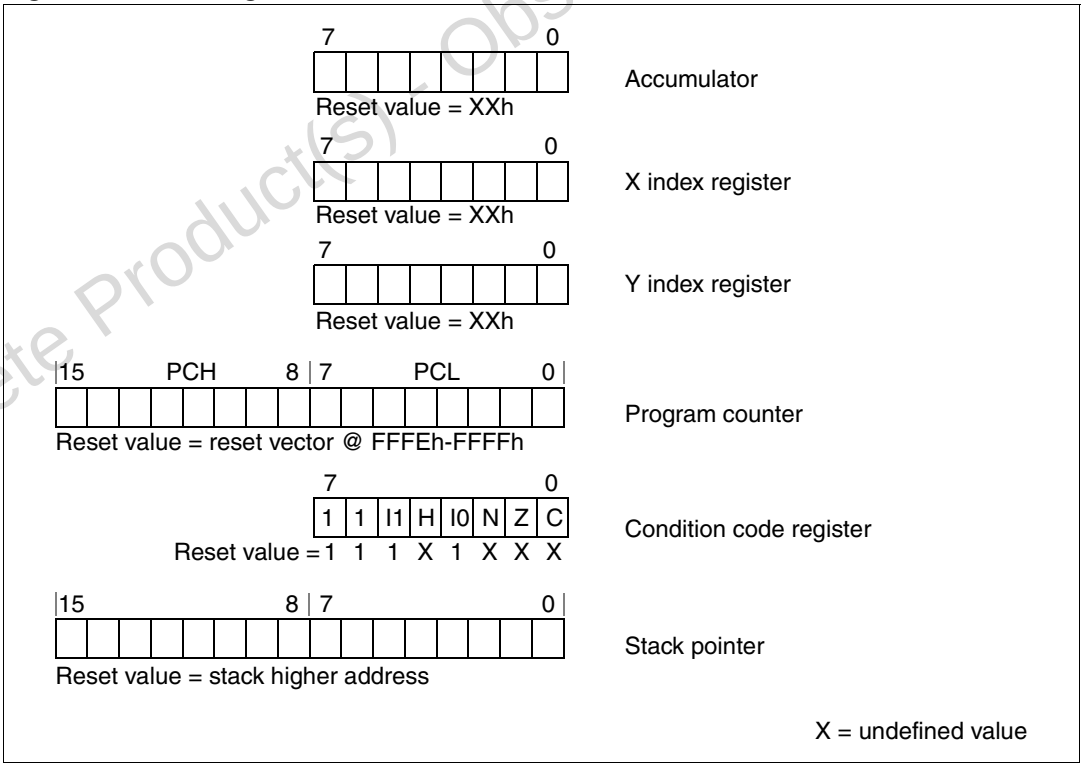
5.2 Main features

- Enable executing 63 basic instructions
- Fast 8-bit by 8-bit multiply
- 17 main addressing modes (with indirect addressing mode)
- Two 8-bit index registers
- 16-bit stack pointer
- Low power Halt and Wait modes
- Priority maskable hardware interrupts
- Non-maskable software/hardware interrupts

5.3 CPU registers

The six CPU registers shown in *Figure 8* are not present in the memory mapping and are accessed by specific instructions.

Figure 8. CPU registers



5.3.1 Accumulator (A)

The accumulator is an 8-bit general purpose register used to hold operands and the results of the arithmetic and logic calculations as well as data manipulations.

5.3.2 Index registers (X and Y)

These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation (the Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

The Y register is not affected by the interrupt automatic procedures.

5.3.3 Program counter (PC)

The program counter is a 16-bit register containing the address of the next instruction to be executed by the CPU. It is made of two 8-bit registers PCL (Program Counter Low which is the LSB) and PCH (Program Counter High which is the MSB).

5.3.4 Condition code (CC) register

The 8-bit condition code register contains the interrupt masks and four flags representative of the result of the instruction just executed. This register can also be handled by the PUSH and POP instructions.

These bits can be individually tested and/or controlled by specific instructions.

CC Reset value: 111x1xxx

7	6	5	4	3	2	1	0
1	1	I1	H	I0	N	Z	C
		RW	RW	RW	RW	RW	RW

Table 8. Arithmetic management bits

Bit	Name	Function
4	H	*Half carry* This bit is set by hardware when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instructions. It is reset by hardware during the same instructions. 0: No half carry has occurred. 1: A half carry has occurred. This bit is tested using the JRH or JRNH instruction. The H bit is useful in BCD arithmetic subroutines.
2	N	*Negative* This bit is set and cleared by hardware. It is representative of the result sign of the last arithmetic, logical or data manipulation. It is a copy of the result 7th bit. 0: The result of the last operation is positive or null. 1: The result of the last operation is negative (that is, the most significant bit is a logic 1). This bit is accessed by the JRMI and JRPL instructions.

The basic RESET sequence consists of three phases as shown in *Figure 13*:

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

Caution: When the ST7 is unprogrammed or fully erased, the Flash is blank and the RESET vector is not programmed. For this reason, it is recommended to keep the $\overline{\text{RESET}}$ pin in low state until programming mode is entered, in order to avoid unwanted behavior.

The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilize and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay should be selected by option byte to correspond to the stabilization time of the external oscillator used in the application (see *Section 21.1.1: Flash configuration*).

The RESET vector fetch phase duration is 2 clock cycles.

Figure 12. Reset block diagram

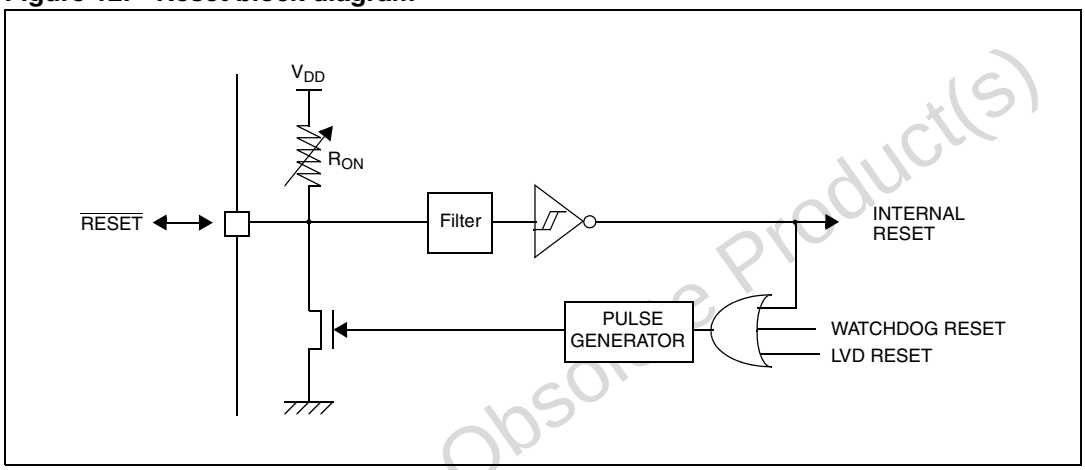
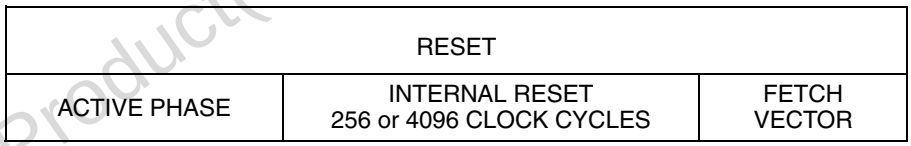


Figure 13. RESET sequence phases

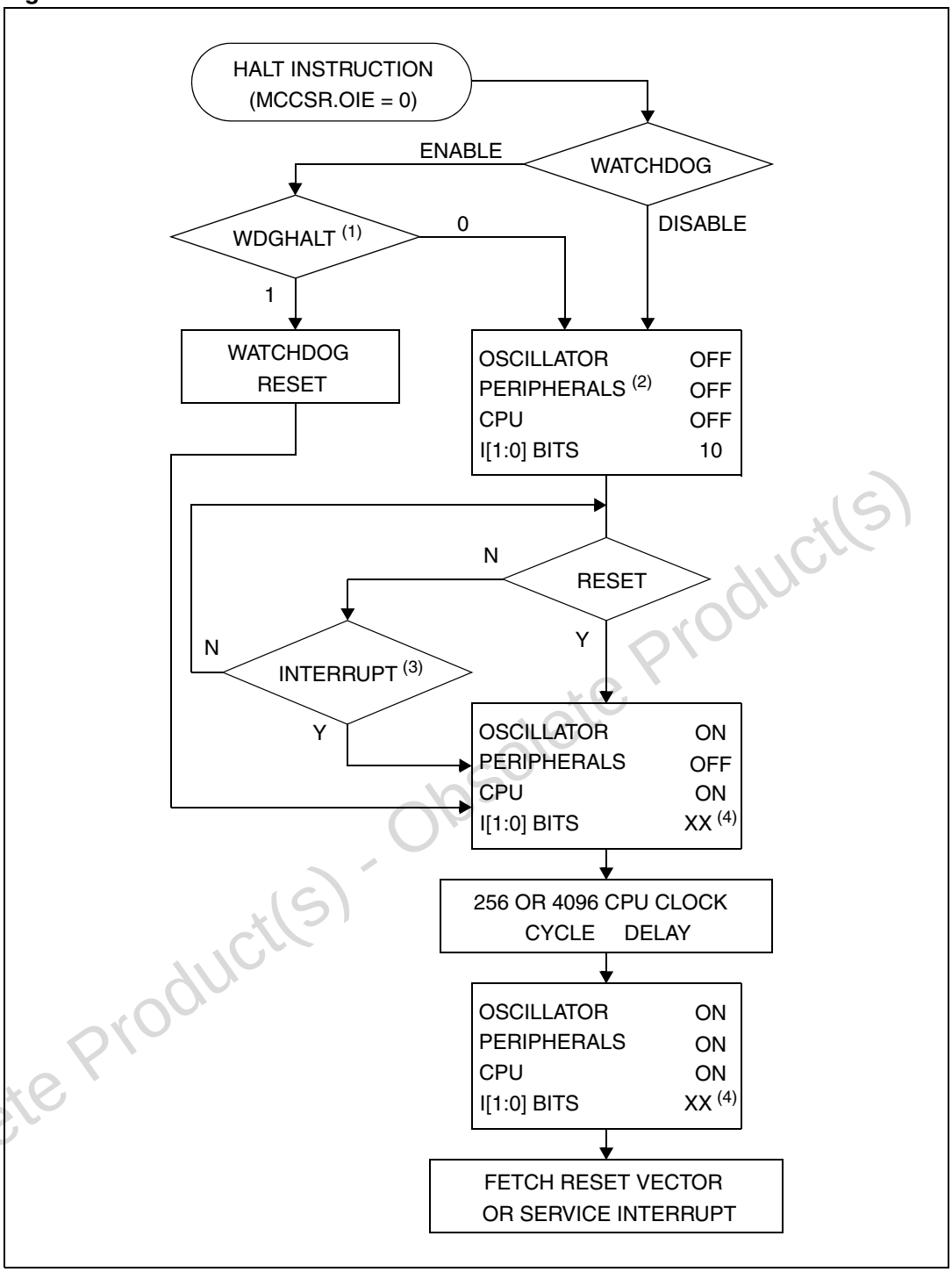


6.5.2 Asynchronous external $\overline{\text{RESET}}$ pin

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device. See *Section 19.9: Control pin characteristics* for more details.

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see *Figure 14*). This detection is asynchronous and therefore the MCU can enter reset state even in Halt mode.

Figure 30. Halt mode flowchart

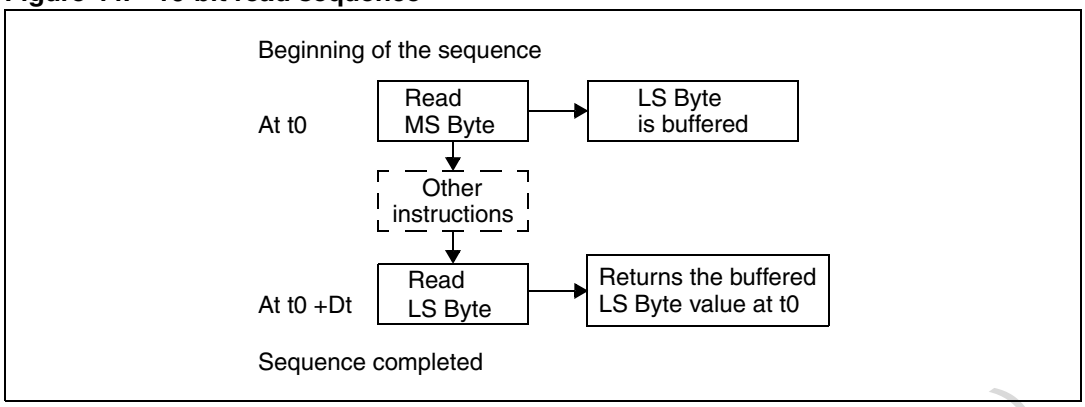


1. WDGHALT is an option bit. See *Section 21.1.1: Flash configuration* for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to *Table 21: Interrupt mapping* for more 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.

16-bit read sequence

The 16-bit read sequence (from either the Counter Register or the Alternate Counter Register) is illustrated in *Figure 44*.

Figure 44. 16-bit read sequence



The user must read the MS Byte first; the LS Byte value is then 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 timer mode is used (input capture, output compare, one pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFFFh to 0000h, after which

- the TOF bit of the SR register is set
- a timer interrupt is generated if
  - the TOIE bit of the CR1 register is set and
  - the 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).

13.3.4 Output compare

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

This function can be used to control an output waveform or indicate when a period of time has elapsed.

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 OC*i*E 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.

	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.

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 62: Timer clock selection*).

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 CR1 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:

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

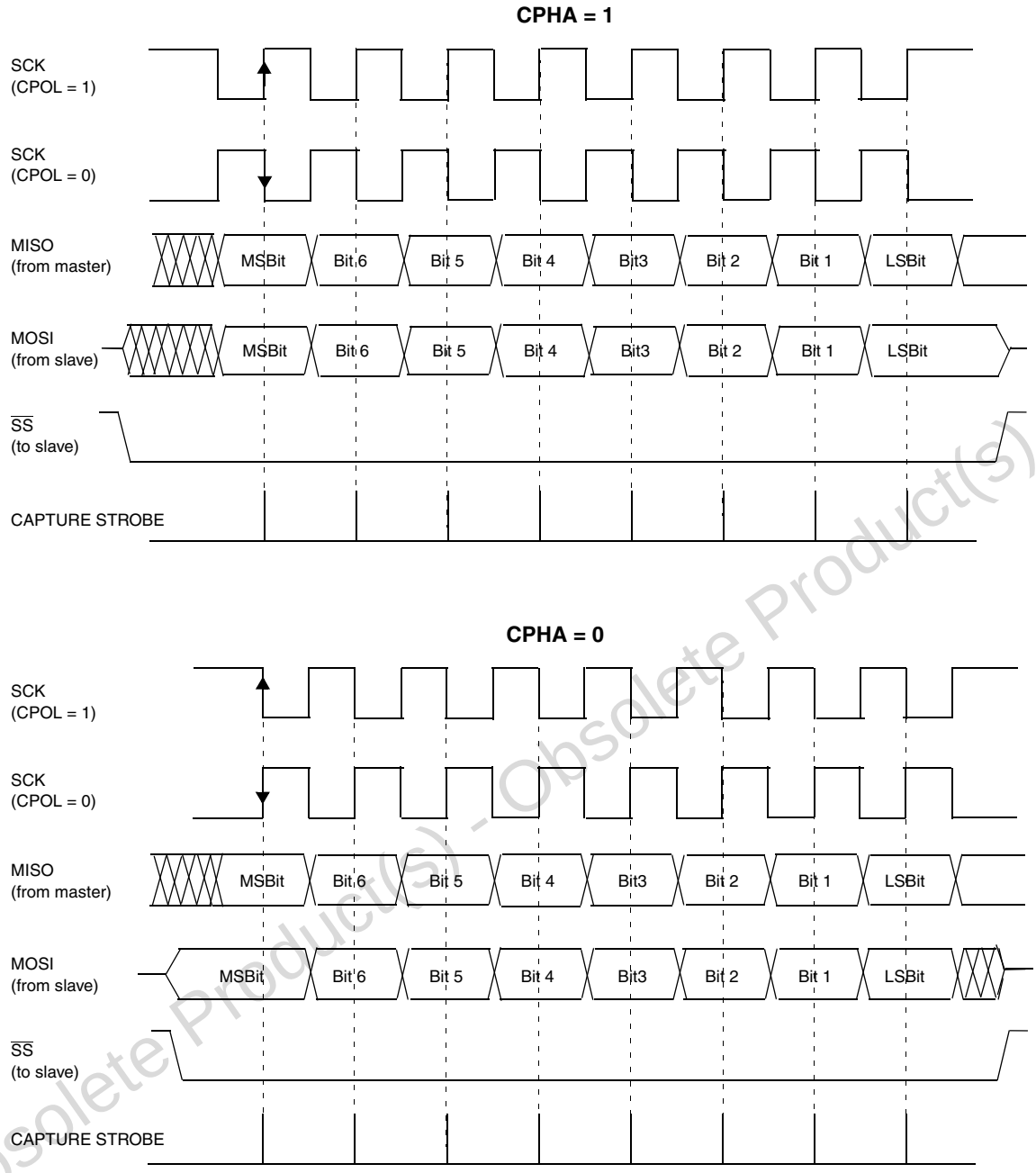
Where:

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 62: Timer clock selection*)

Figure 61. Data clock timing diagram



Note: This figure should not be used as a replacement for parametric information.
Refer to Section 19: Electrical characteristics.

15.4 Functional description

The block diagram of the Serial Control Interface, is shown in *Figure 64*. 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 15.7* for the definitions of each bit.

15.4.1 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.

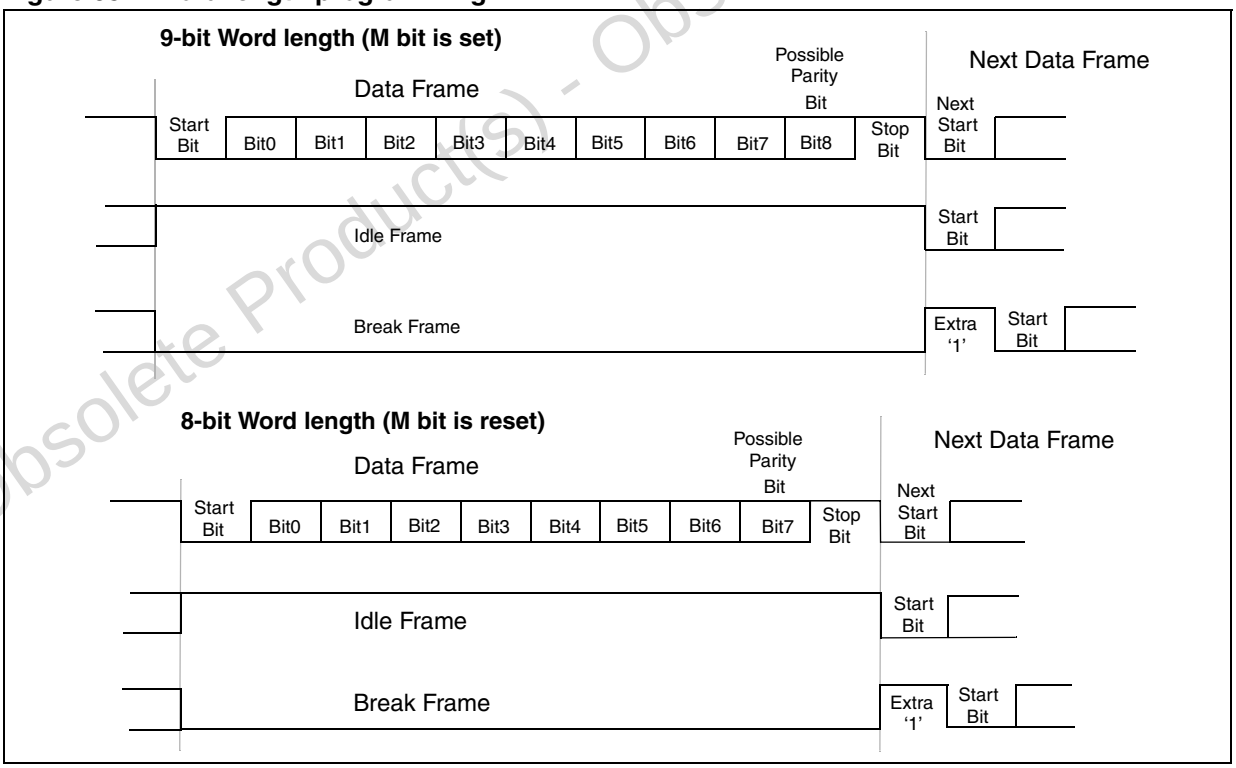
The TDO pin is in high state during the stop bit.

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.

Figure 65. Word length programming



A receiver wakes up by Idle Line detection when the Receive line has recognized an Idle Frame. Then the RWU bit is reset by hardware but the IDLE bit is not set.

Receiver wakes up by Address Mark detection when it received a ‘1’ as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

Caution: In Mute mode, do not write to the SCICR2 register. If the SCI is in Mute mode during the read operation (RWU = 1) and a address mark wake-up event occurs (RWU is reset) before the write operation, the RWU bit is set again by this write operation. Consequently the address byte is lost and the SCI is not woken up from Mute mode.

Parity control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the SCICR1 register. Depending on the frame length defined by the M bit, the possible SCI frame formats are as listed in *Table 71*.

Table 71. Frame formats

M bit	PCE bit	SCI frame
0	0	ǀ SB ǀ 8 bit data ǀ STB ǀ
0	1	ǀ SB ǀ 7-bit data ǀ PB ǀ STB ǀ
1	0	ǀ SB ǀ 9-bit data ǀ STB ǀ
1	1	ǀ SB ǀ 8-bit data PB ǀ STB ǀ

Legend: SB = Start Bit, STB = Stop Bit, PB = Parity Bit

Note: In case of wake-up by an address mark, the MSB bit of the data is taken into account and not the parity bit

Even parity: the parity bit is calculated to obtain an even number of ‘1’s inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit is 0 if even parity is selected (PS bit = 0).

Odd parity: the parity bit is calculated to obtain an odd number of ‘1’s inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit is 1 if odd parity is selected (PS bit = 1).

Transmission mode: If the PCE bit is set then the MSB bit of the data written in the data register is not transmitted but is changed by the parity bit.

Reception mode: If the PCE bit is set then the interface checks if the received data byte has an even number of ‘1’s if even parity is selected (PS = 0) or an odd number of ‘1’s if odd parity is selected (PS = 1). If the parity check fails, the PE flag is set in the SCISR register and an interrupt is generated if PIE is set in the SCICR1 register.

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.

Closing slave communication

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

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

Then the interface waits for a read of the SR2 register (see *Figure 70: Transfer sequencing* EV4).

Error cases

- **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and the BERR bits are set with an interrupt if the ITE bit is set.
  If it is a Stop then the interface discards the data, released the lines and waits for another Start condition.
  If it is a Start then the interface discards the data and waits for the next slave address on the bus.
- **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set with an interrupt if the ITE bit is set.
  The AF bit is cleared by reading the I2CSR2 register. However, if read before the completion of the transmission, the AF flag will be set again, thus possibly generating a new interrupt. Software must ensure either that the SCL line is back at 0 before reading the SR2 register, or be able to correctly handle a second interrupt during the 9th pulse of a transmitted byte.

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

How to release the SDA / SCL lines

Set and subsequently clear the STOP bit while BTF is set. The SDA/SCL lines are released after the transfer of the current byte.

SMBus compatibility

The ST7 I^2C is compatible with the SMBus V1.1 protocol. It supports all SMBus addressing modes, SMBus bus protocols and CRC-8 packet error checking. Refer to *SMBus Slave Driver For ST7 I^2C Peripheral* (AN1713).

16.4.2 Master mode

To switch from default Slave mode to Master mode a Start condition generation is needed.

Start condition

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, **holding the SCL line low** (see *Figure 70: Transfer sequencing* EV5).

17 10-bit A/D converter (ADC)

17.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 16 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 16 different sources.

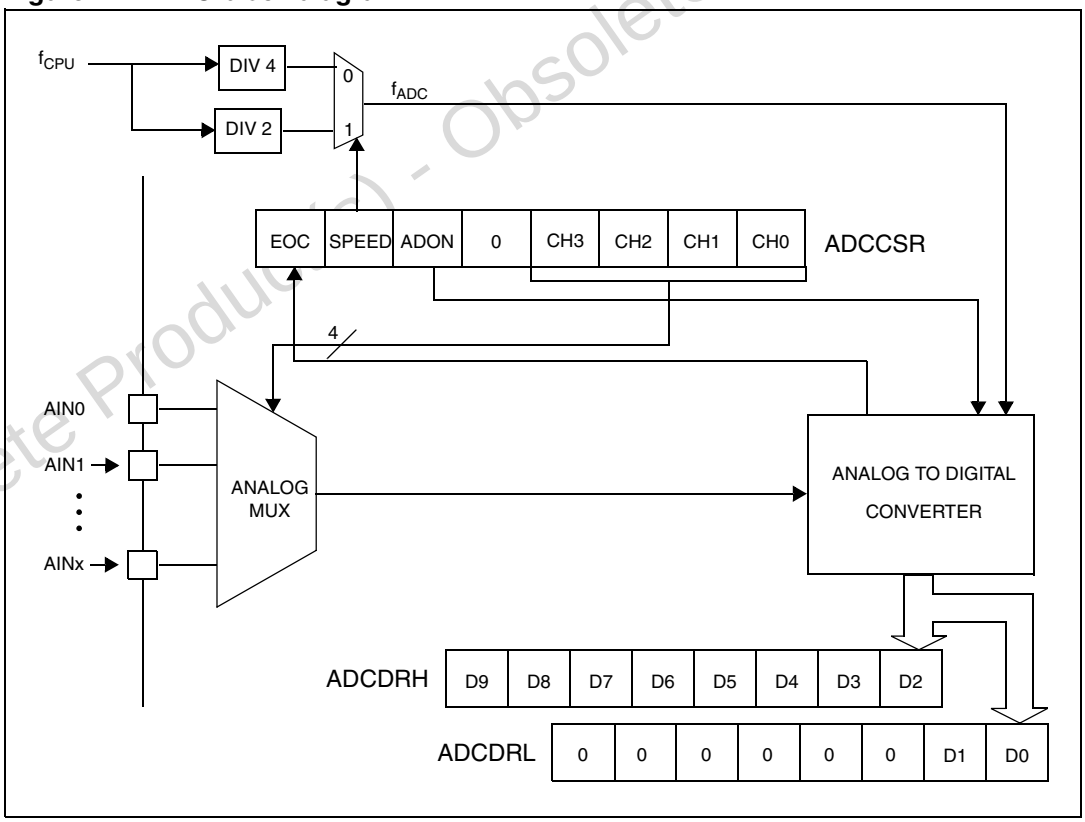
The result of the conversion is stored in a 10-bit data register. The A/D converter is controlled through a control/status register.

17.2 Main features

- 10-bit conversion
- Up to 16 channels with multiplexed input
- Linear successive approximation
- Data register (DR) which contains the results
- Conversion complete status flag
- On/off bit (to reduce consumption)

The block diagram is shown in *Figure 72*.

Figure 72. ADC block diagram



17.6.3 Data register (ADCDRL)

ADCDRL Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved						D[1:0]	
-						RO	

Table 95. ADCDRL register description

Bit	Name	Function
7:2	-	Reserved. Forced by hardware to 0.
1:0	D[1:0]	*LSB of Converted Analog Value*

17.6.4 ADC register map and reset values

Table 96. ADC register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0070h	ADCCSR Reset value	EOC 0	SPEED 0	ADON 0	 0	CH3 0	CH2 0	CH1 0	CH0 0
0071h	ADCDRH Reset value	D9 0	D8 0	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0
0072h	ADCDRL Reset value	 0	 0	 0	 0	 0	 0	D1 0	D0 0

19.7 EMC (electromagnetic compatibility) characteristics

Susceptibility tests are performed on a sample basis during product characterization.

19.7.1 Functional EMS (electromagnetic susceptibility)

Based on a simple running application on the product (toggling two LEDs through I/O ports), the product is stressed by two electromagnetic events until a failure occurs (indicated by the LEDs).

- **ESD**: Electrostatic discharge (positive and negative) is applied on all pins of the device until a functional disturbance occurs. This test conforms with the IEC 1000-4-2 standard.
- **FTB**: A burst of fast transient voltage (positive and negative) is applied to V_{DD} and V_{SS} through a 100pF capacitor until a functional disturbance occurs. This test conforms with the IEC 1000-4-4 standard.

A device reset allows normal operations to be resumed. The test results given in *Table 124* below are based on the EMS levels and classes defined in application note AN1709.

Designing hardened software to avoid noise problems

EMC characterization and optimization are performed at component level with a typical application environment and simplified MCU software. It should be noted that good EMC performance is highly dependent on the user application and the software in particular.

Therefore it is recommended that the user applies EMC software optimization and prequalification tests in relation with the EMC level requested for his application.

Software recommendations

The software flowchart must include the management of runaway conditions such as:

- Corrupted program counter
- Unexpected reset
- Critical Data corruption (control registers...)

Prequalification trials

Most of the common failures (unexpected reset and program counter corruption) can be reproduced by manually forcing a low state on the $\overline{\text{RESET}}$ pin or the oscillator pins for 1 second.

To complete these trials, ESD stress can be applied directly on the device, over the range of specification values. When unexpected behavior is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Table 126. ESD absolute maximum ratings

Symbol	Ratings	Conditions	Max. value(1)	Unit
$V_{ESD(HBM)}$	Electrostatic discharge voltage (Human Body Model)	T_A = +25°C	2000	V
$V_{ESD(MM)}$	Electrostatic discharge voltage (Machine Model)		200	

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

Static and dynamic latch-up

- **LU**: Three complementary static tests are required on 10 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard. For more details, refer to the application note AN1181.
- **DLU**: Electrostatic discharges (one positive then one negative test) are applied to each pin of three samples when the micro is running to assess the latch-up performance in dynamic mode. Power supplies are set to the typical values, the oscillator is connected as near as possible to the pins of the micro and the component is put in reset mode. This test conforms to the IEC1000-4-2 and SAEJ1752/3 standards. For more details, refer to the application note AN1181.

Table 127. Electrical sensitivities

Symbol	Parameter	Conditions	Class(1)
LU	Static latch-up class	T_A = +25°C T_A = +85°C T_A = +125°C	A A A
DLU	Dynamic latch-up class	V_{DD} = 5.5V, f_{OSC} = 4 MHz, T_A = +25°C	A

1. Class description: A Class is an STMicroelectronics internal specification. All its limits are higher than the JEDEC specifications, that means when a device belongs to Class A it exceeds the JEDEC standard. B Class strictly covers all the JEDEC criteria (international standard).

```
.ext1_rt
; entry to interrupt routine
LD A,#00

LD sema,A

IRET
```

Case 2: Writing to PxOR or PxDDR with global interrupts disabled:

```
SIM
; set the interrupt mask
LD A,PFDR

AND A,#$02

LD  X,A
; store the level before writing to PxOR/PxDDR
LD A,#$90

LD PFDDR,A
; Write into PFDDR
LD A,#$ff

LD PFOR,A
          ; Write to PFOR
LD A,PFDR

AND A,#$02

LD Y,A
; store the level after writing to PxOR/PxDDR
LD A,X
; check for falling edge
cp A,#$02

jrne OUT

TNZ Y

jrne OUT

LD A,#$01

LD sema,A
; set the semaphore to '1' if edge is detected
RIM
 ; reset the interrupt mask
LD A,sema
; check the semaphore status
CP A,#$01
```

```
jrne OUT

call call_routine
; call the interrupt routine
RIM

OUT:
RIM
JP while_loop

.call_routine
; entry to call_routine
PUSH A

PUSH X

PUSH CC

.ext1_rt
; entry to interrupt routine
LD A,#$00

LD sema,A

IRET
```

22.1.3 Clearing active interrupts outside interrupt routine

When an active interrupt request occurs at the same time as the related flag is being cleared, an unwanted reset may occur.

Note: Clearing the related interrupt mask will not generate an unwanted reset.

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, that is, when:

- The interrupt flag is cleared within its own interrupt routine
- The interrupt flag is cleared within any interrupt routine
- The interrupt flag is cleared in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request.

Example:

SIM
Reset interrupt flag
RIM

Nested interrupt context

The symptom does not occur when the interrupts are handled normally, that is, when:

- The interrupt flag is cleared within its own interrupt routine
- The interrupt flag is cleared within any interrupt routine with higher or identical priority level
- The interrupt flag is cleared in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

```
PUSH CC
SIM
Reset interrupt flag
POP CC
```

22.1.4 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 lead to generating one break more than expected.

Occurrence

The occurrence of the problem is random and proportional to the baud rate. With a transmit frequency of 19200 baud (f_{CPU} = 8 MHz 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:

- Disable interrupts
- Reset and Set TE (IDLE request)
- Set and Reset SBK (Break Request)
- Re-enable interrupts

22.1.5 16-bit timer PWM mode

In PWM mode, the first PWM pulse is missed after writing the value FFFCh in the OC1R register (OC1HR, OC1LR). It leads to either full or no PWM during a period, depending on the OLVL1 and OLVL2 settings.

22.1.6 TIMD set simultaneously with OC interrupt

If the 16-bit timer is disabled at the same time the output compare event occurs, the output compare flag then gets locked and cannot be cleared before the timer is enabled again.

Impact on the application

If the output compare interrupt is enabled, then the output compare flag cannot be cleared in the timer interrupt routine. Consequently, the interrupt service routine is called repeatedly.

Workaround

Disable the timer interrupt before disabling the timer. Again while enabling, first enable the timer, then the timer interrupts.

- Perform the following to disable the timer:
  - TACR1 or TBCR1 = 0x00h; // Disable the compare interrupt
  - TACSR | or TBCSR | = 0x40; // Disable the timer
- Perform the following to enable the timer again:
  - TACSR & or TBCSR & = ~0x40; // Enable the timer
  - TACR1 or TBCR1 = 0x40; // Enable the compare interrupt

22.1.7 I2C multimaster

In multimaster configurations, if the ST7 I2C receives a START condition from another I2C master after the START bit is set in the I2CCR register and before the START condition is generated by the ST7 I2C, it may ignore the START condition from the other I2C master. In this case, the ST7 master will receive a NACK from the other device. On reception of the NACK, ST7 can send a restart and Slave address to re-initiate communication.

22.1.8 Pull-up always active on PE2

The I/O port internal pull-up is always active on I/O port E2. As a result, if PE2 is in output mode low level, current consumption in Halt/Active Halt mode is increased.