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	LINbusSCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	48
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1.5K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f361ar6t6

5 CENTRAL PROCESSING UNIT

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.

Accumulator (A)

The Accumulator is an 8-bit general purpose register used to hold operands and the results of the arithmetic and logic calculations and to manipulate data.

Index Registers (X and Y)

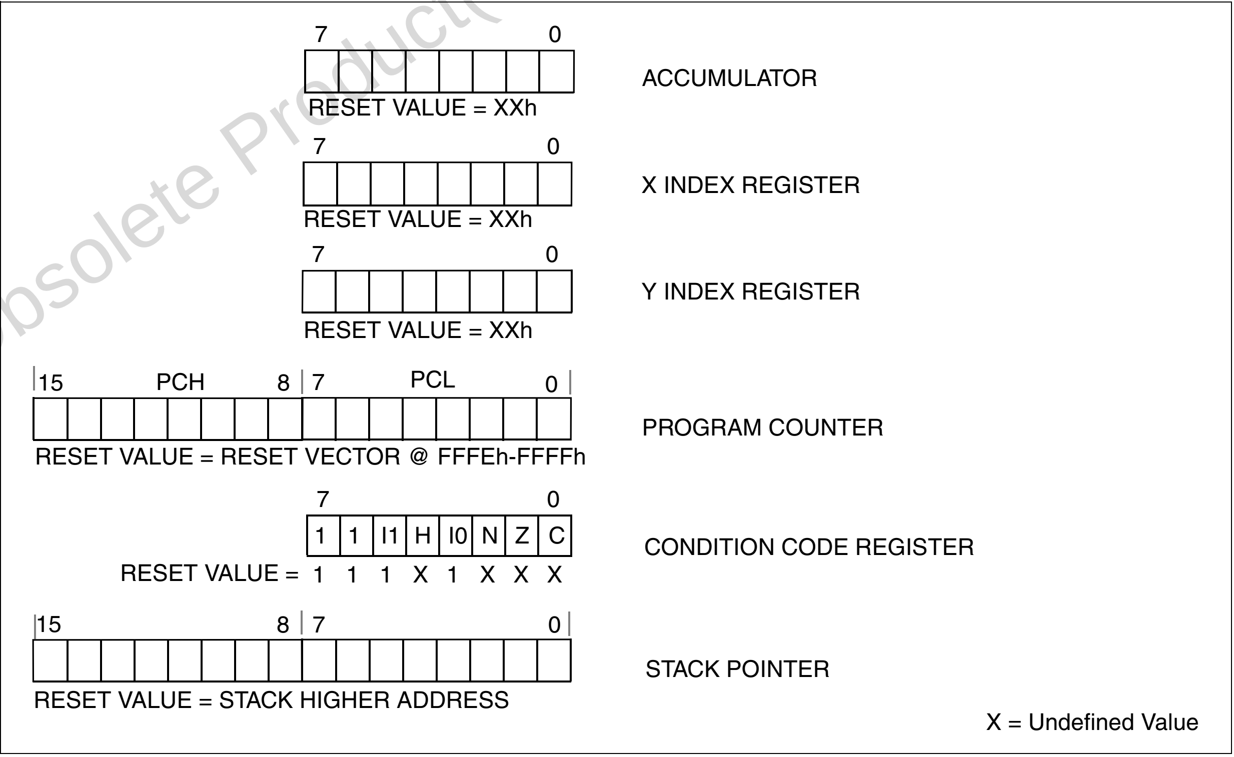
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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

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).

Figure 8. CPU Registers



6 SUPPLY, RESET AND CLOCK MANAGEMENT

The device includes a range of utility features for securing the application in critical situations (for example, in case of a power brown-out), and reducing the number of external components. An overview is shown in Figure 11.

For more details, refer to dedicated parametric section.

Main features

- Optional PLL for multiplying the frequency by 2
- Reset Sequence Manager (RSM)
- Multi-Oscillator Clock Management (MO)
  - 4 Crystal/Ceramic resonator oscillators
- System Integrity Management (SI)
  - Main supply Low voltage detection (LVD)
  - Auxiliary Voltage detector (AVD) with interrupt capability for monitoring the main supply

6.1 PHASE LOCKED LOOP

If the clock frequency input to the PLL is in the range 2 to 4 MHz, the PLL can be used to multiply the frequency by two to obtain an f_{OSC2} of 4 to 8 MHz. The PLL is enabled by option byte. If the PLL is disabled, then $f_{OSC2} = f_{OSC}/2$.

Caution: The PLL is not recommended for applications where timing accuracy is required. See "PLL Characteristics" on page 187.

Figure 10. PLL Block Diagram

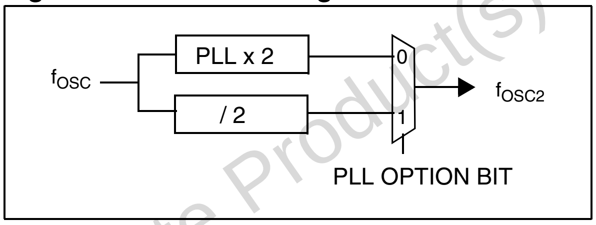
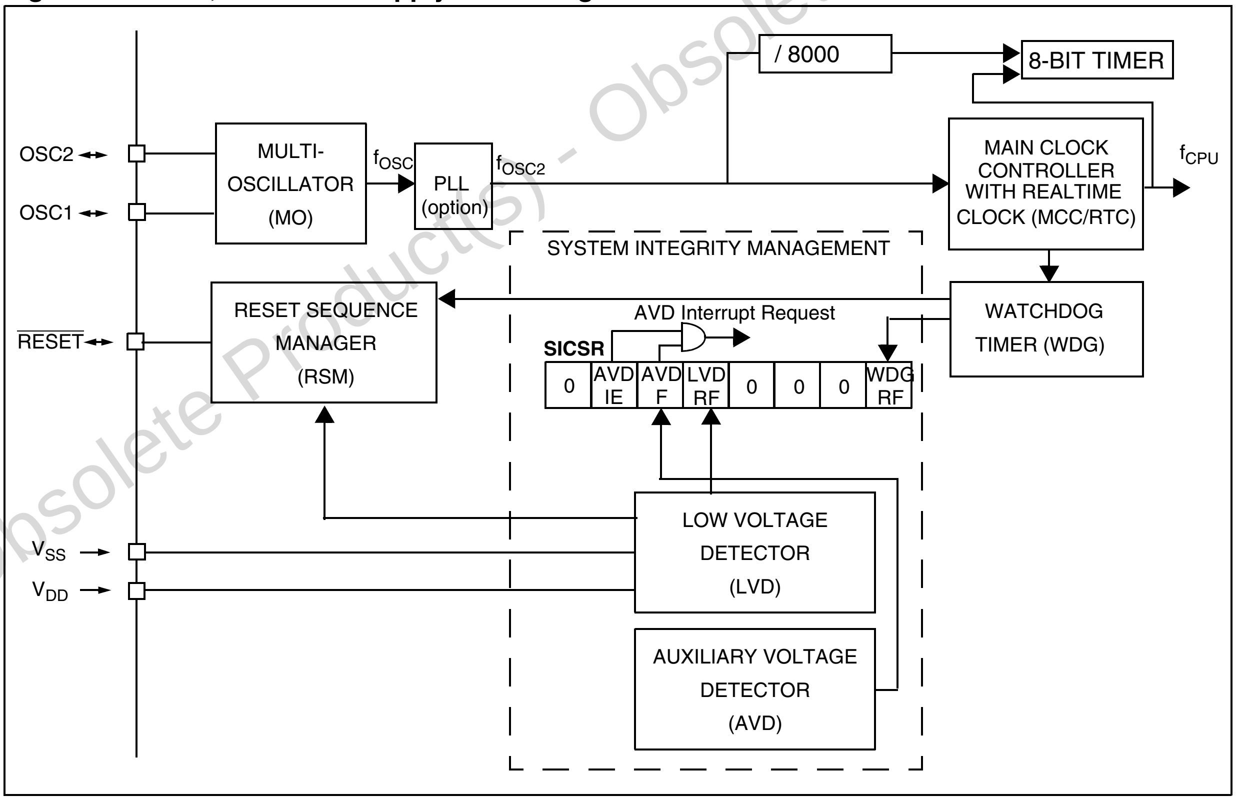


Figure 11. Clock, Reset and Supply Block Diagram



/ 8000
8-BIT TIMER
OSC2
OSC1
MULTI-OSCILLATOR (MO)
fOSC
PLL (option)
fOSC2
MAIN CLOCK CONTROLLER WITH REALTIME CLOCK (MCC/RTC)
fCPU
SYSTEM INTEGRITY MANAGEMENT
RESET
RESET SEQUENCE MANAGER (RSM)
AVD Interrupt Request
SICSR
0
AVD IE
AVD F
LVD RF
0
0
0
WDG RF
WATCHDOG TIMER (WDG)
LOW VOLTAGE DETECTOR (LVD)
AUXILIARY VOLTAGE DETECTOR (AVD)
VSS
VDD

6.3 RESET SEQUENCE MANAGER (RSM)

6.3.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 2:

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

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

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

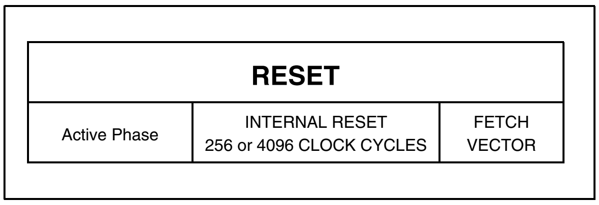
The basic RESET sequence consists of three phases as shown in Figure 1:

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

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.

The RESET vector fetch phase duration is two clock cycles.

Figure 12. RESET Sequence Phases



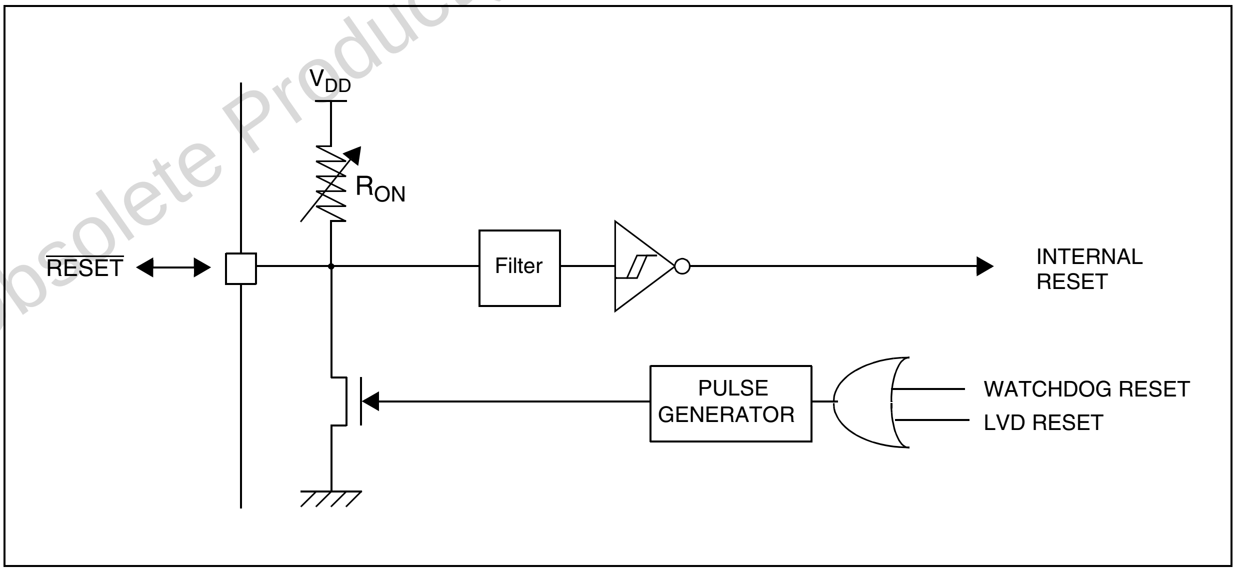
Caution: When the ST7 is unprogrammed or fully erased, the Flash is blank and the RESET vector is not programmed. For this reason, it is recommended to keep the RESET pin in low state until programming mode is entered, in order to avoid unwanted behavior.

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

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

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see Figure 3). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

Figure 13. Reset Block Diagram



INTERRUPTS (Cont'd)

Table 9. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT[1)]	Address Vector
	RESET	Reset	N/A	Highest Priority	yes	FFFEh-FFFFh
	TRAP	Software interrupt			no	FFFCh-FFFDh
0	TLI	External top level interrupt	EICR		yes	FFFAh-FFFBh
1	MCC/RTC	Main clock controller time base interrupt	MCCSR		yes	FFF8h-FFF9h
2	ei0/AWUFH	External interrupt ei0/ Auto wake-up from Halt	EICR/ AWUCSR		yes[2)]	FFF6h-FFF7h
3	ei1/AVD	External interrupt ei1/Auxiliary Voltage Detector	EICR/ SICSR			FFF4h-FFF5h
4	ei2	External interrupt ei2	EICR			FFF2h-FFF3h
5	ei3	External interrupt ei3	EICR			FFF0h-FFF1h
6	not used					FFEEh-FFEFh
7	not used					FFECh-FFEDh
8	SPI	SPI peripheral interrupts	SPICSR		yes	FFEAh-FFEBh
9	TIMER8	8-bit TIMER peripheral interrupts	T8_TCR1		no	FFE8h-FFE9h
10	TIMER16	16-bit TIMER peripheral interrupts	TCR1		no	FFE6h-FFE7h
11	LINSCI2	LINSCI2 Peripheral interrupts	SCI2CR1		no	FFE4h-FFE5h
12	LINSCI1	LINSCI1 Peripheral interrupts (LIN Master/ Slave)	SCI1CR1	Lowest Priority	no[3)]	FFE2h-FFE3h
13	PWM ART	8-bit PWM ART interrupts	PWMCR		yes	FFE0h-FFE1h

Notes:

1. Valid for HALT and ACTIVE HALT modes except for the MCC/RTC interrupt source which exits from ACTIVE HALT mode only.
2. Except AVD interrupt
3. It is possible to exit from Halt using the external interrupt which is mapped on the RDI pin.

INTERRUPTS (Cont'd)

7.6 EXTERNAL INTERRUPTS

7.6.1 I/O Port Interrupt Sensitivity

The external interrupt sensitivity is controlled by the ISxx bits in the EICR register (Figure 21). This control allows up to four fully independent external interrupt source sensitivities.

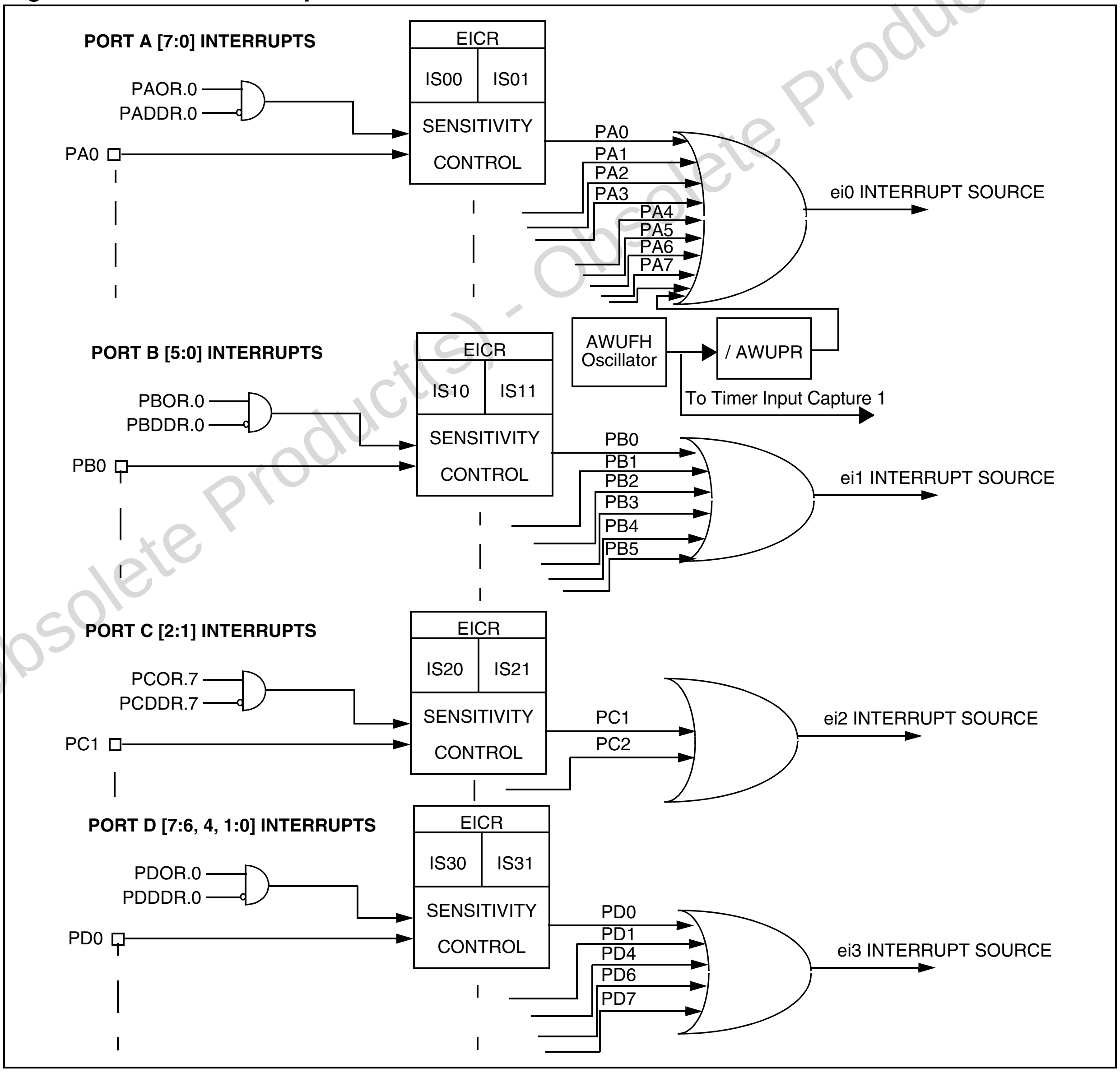
Each external interrupt source can be generated on four (or five) different events on the pin:

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

To guarantee correct functionality, the sensitivity bits in the EICR register can be modified only when the I1 and I0 bits of the CC register are both set to 1 (level 3). This means that interrupts must be disabled before changing sensitivity.

The pending interrupts are cleared by writing a different value in the ISx[1:0] of the EICR.

Figure 21. External Interrupt Control Bits



ON-CHIP PERIPHERALS (Cont'd)

10.3.3 Register Description

CONTROL / STATUS REGISTER (ARTCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
EXCL	CC2	CC1	CC0	TCE	FCRL	OIE	OVF

Bit 7 = **EXCL** *External Clock*
This bit is set and cleared by software. It selects the input clock for the 7-bit prescaler.
0: CPU clock.
1: External clock.

Bit 6:4 = **CC[2:0]** *Counter Clock Control*
These bits are set and cleared by software. They determine the prescaler division ratio from f_{INPUT}.

$f_{COUNTER}$	With f_{INPUT}=8 MHz	CC2	CC1	CC0
f_{INPUT}	8 MHz	0	0	0
f_{INPUT} / 2	4 MHz	0	0	1
f_{INPUT} / 4	2 MHz	0	1	0
f_{INPUT} / 8	1 MHz	0	1	1
f_{INPUT} / 16	500 kHz	1	0	0
f_{INPUT} / 32	250 kHz	1	0	1
f_{INPUT} / 64	125 kHz	1	1	0
f_{INPUT} / 128	62.5 kHz	1	1	1

Bit 3 = **TCE** *Timer Counter Enable*
This bit is set and cleared by software. It puts the timer in the lowest power consumption mode.
0: Counter stopped (prescaler and counter frozen).
1: Counter running.

Bit 2 = **FCRL** *Force Counter Re-Load*
This bit is write-only and any attempt to read it will yield a logical zero. When set, it causes the contents of ARTARR register to be loaded into the counter, and the content of the prescaler register to be cleared in order to initialize the timer before starting to count.

Bit 1 = **OIE** *Overflow Interrupt Enable*
This bit is set and cleared by software. It allows to enable/disable the interrupt which is generated when the OVF bit is set.
0: Overflow Interrupt disable.
1: Overflow Interrupt enable.

Bit 0 = **OVF** *Overflow Flag*
This bit is set by hardware and cleared by software reading the ARTCSR register. It indicates the transition of the counter from FFh to the ARTARR value.
0: New transition not yet reached
1: Transition reached

COUNTER ACCESS REGISTER (ARTCAR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0

Bit 7:0 = **CA[7:0]** *Counter Access Data*

These bits can be set and cleared either by hardware or by software. The ARTCAR register is used to read or write the auto-reload counter "on the fly" (while it is counting).

AUTO-RELOAD REGISTER (ARTARR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0

Bit 7:0 = **AR[7:0]** *Counter Auto-Reload Data*

These bits are set and cleared by software. They are used to hold the auto-reload value which is automatically loaded in the counter when an overflow occurs. At the same time, the PWM output levels are changed according to the corresponding OPx bit in the PWMCR register.

This register has two PWM management functions:

– Adjusting the PWM frequency
– Setting the PWM duty cycle resolution

PWM Frequency vs Resolution:

ARTARR value	Resolution	f_{PWM}	
		Min	Max
0	8-bit	~0.244 kHz	31.25 kHz
[0..127]	> 7-bit	~0.244 kHz	62.5 kHz
[128..191]	> 6-bit	~0.488 kHz	125 kHz
[192..223]	> 5-bit	~0.977 kHz	250 kHz
[224..239]	> 4-bit	~1.953 kHz	500 kHz

16-BIT TIMER (Cont'd)

Figure 49. Counter Timing Diagram, Internal Clock Divided by 2

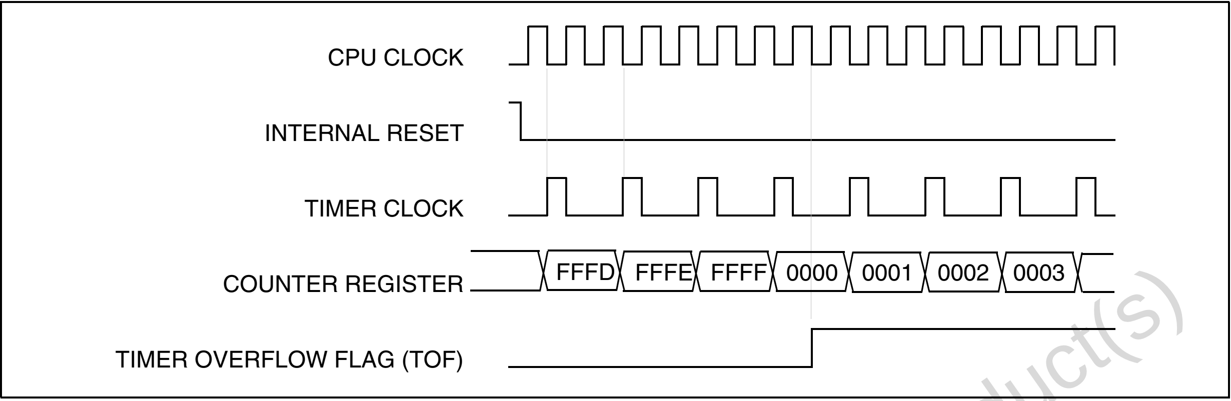


Figure 50. Counter Timing Diagram, Internal Clock Divided by 4

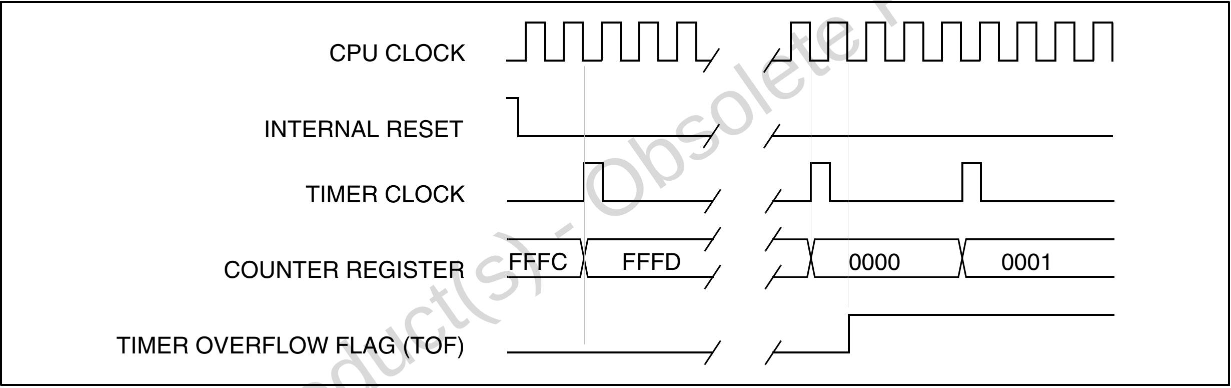
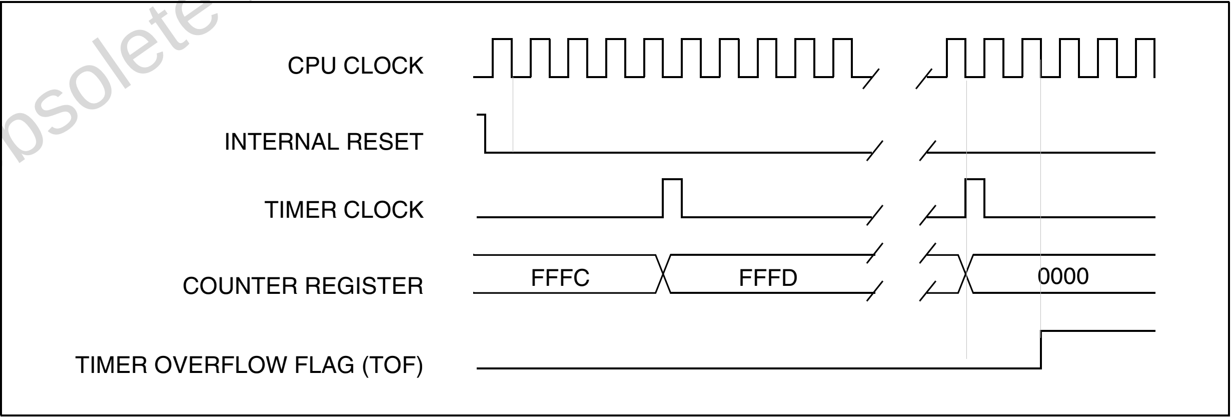


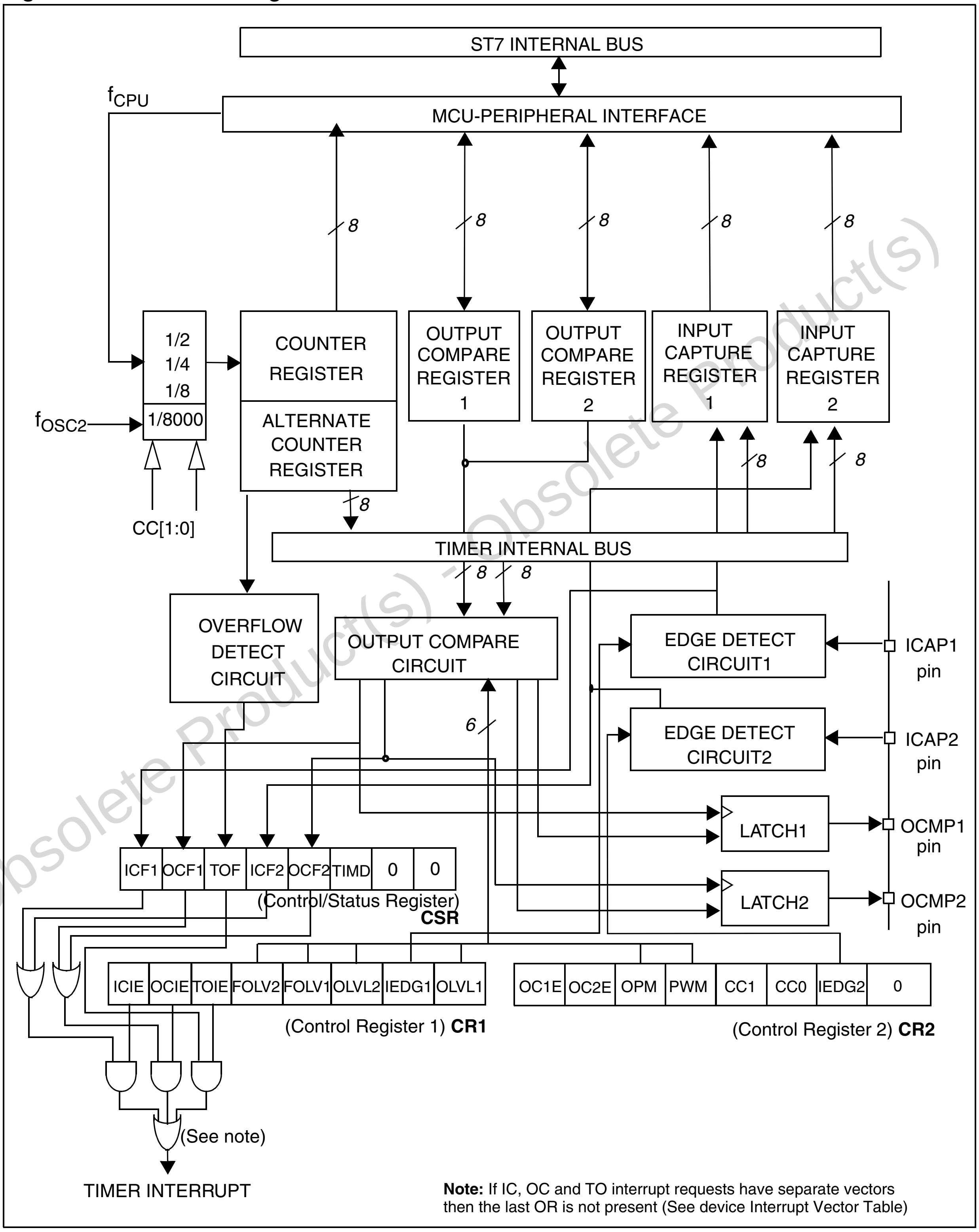
Figure 51. Counter Timing Diagram, Internal Clock Divided By 8



Note: The MCU is in reset state when the internal reset signal is high, when it is low the MCU is running.

8-BIT TIMER (Cont'd)

Figure 59. Timer Block Diagram



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

8-BIT TIMER (Cont'd)

Figure 63. Input Capture Block Diagram

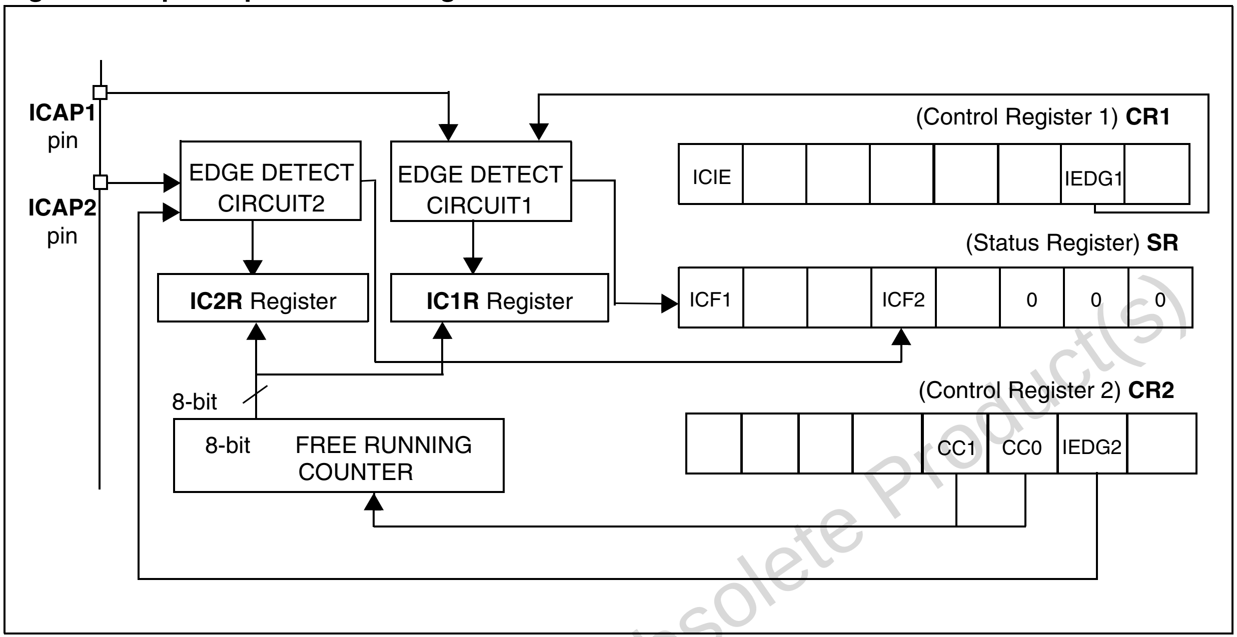
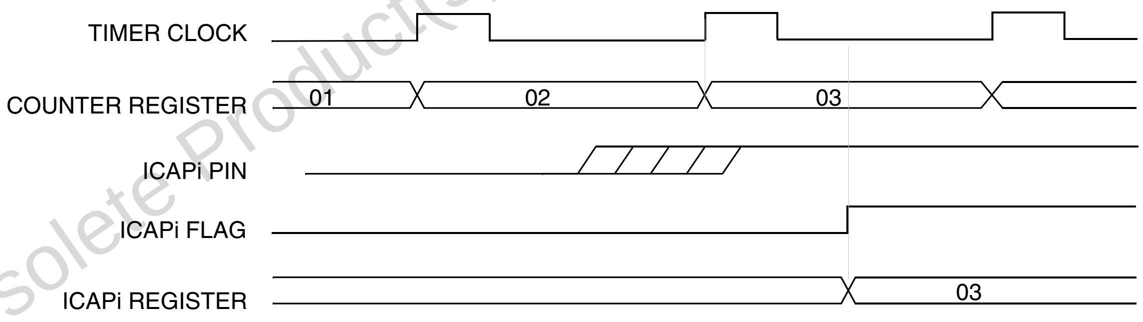


Figure 64. Input Capture Timing Diagram



Note: The rising edge is the active edge.

SERIAL PERIPHERAL INTERFACE (cont'd)

10.6.3.2 Slave Select Management

As an alternative to using the $\overline{SS}$ pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 73).

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

In Master mode:

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

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 72):

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

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

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

– $\overline{SS}$ internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If $\overline{SS}$ is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 10.6.5.3).

Figure 72. Generic $\overline{SS}$ Timing Diagram

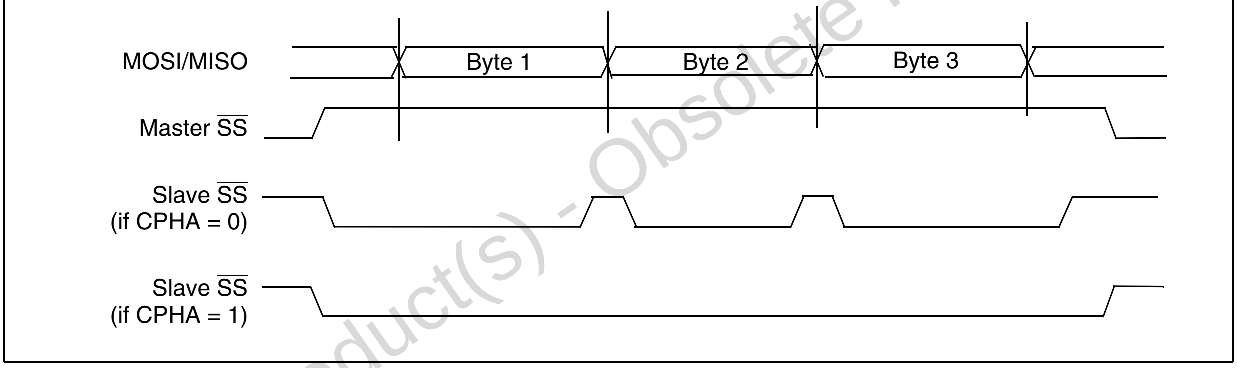
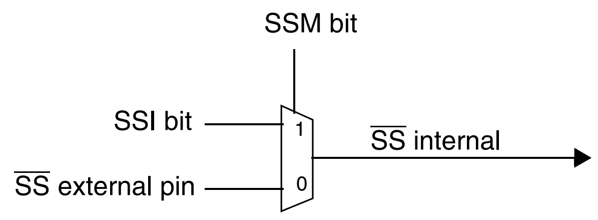

MOSI/MISO
Byte 1
Byte 2
Byte 3
Master $\overline{SS}$
Slave $\overline{SS}$ (if CPHA = 0)
Slave $\overline{SS}$ (if CPHA = 1)

Figure 73. Hardware/Software Slave Select Management



SERIAL PERIPHERAL INTERFACE (cont'd)

10.6.6 Low Power Modes

Mode	Description
WAIT	No effect on SPI. SPI interrupt events cause the device to exit from WAIT mode.
HALT	SPI registers are frozen. In HALT mode, the SPI is inactive. SPI operation resumes when the device is woken up by an interrupt with "exit from HALT mode" capability. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetching). If several data are received before the wake-up event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the Device.

10.6.6.1 Using the SPI to wake up the device from Halt mode

In slave configuration, the SPI is able to wake up the device from HALT mode through a SPIF interrupt. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetch). If multiple data transfers have been performed before software clears the SPIF bit, then the OVR bit is set by hardware.

Note: When waking up from HALT mode, if the SPI remains in Slave mode, it is recommended to perform an extra communications cycle to bring the SPI from HALT mode state to normal state. If the SPI exits from Slave mode, it returns to normal state immediately.

Caution: The SPI can wake up the device from HALT mode only if the Slave Select signal (external $\overline{SS}$ pin or the SSI bit in the SPICSR register) is low when the device enters HALT mode. So, if Slave selection is configured as external (see Section 10.6.3.2), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters HALT mode.

10.6.7 Interrupts

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 Interrupts chapter). They generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

LINSCI™ SERIAL COMMUNICATION INTERFACE (cont'd)

10.7.4 General Description

The interface is externally connected to another device by two pins:

– TDO: Transmit Data Output. When the transmitter is disabled, the output pin returns to its I/O port configuration. When the transmitter is enabled and nothing is to be transmitted, the TDO pin is at high level.

– RDI: Receive Data Input is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

Through these pins, serial data is transmitted and received as characters comprising:

– An Idle Line prior to transmission or reception

– A start bit

– A data word (8 or 9 bits) least significant bit first

– A Stop bit indicating that the character is complete

This interface uses three types of baud rate generator:

– A conventional type for commonly-used baud rates

– An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies

– A LIN baud rate generator with automatic resynchronization

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

CONTROL REGISTER 2 (SCICR2)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU[1)]	SBK[1)]

[1)]This bit has a different function in LIN mode, please refer to the LIN mode register description.

Bit 7 = **TIE** *Transmitter interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: In SCI interrupt is generated whenever TDRE = 1 in the SCISR register

Bit 6 = **TCIE** *Transmission complete interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TC = 1 in the SCISR register

Bit 5 = **RIE** *Receiver interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever OR = 1 or RDRF = 1 in the SCISR register

Bit 4 = **ILIE** *Idle line interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever IDLE = 1 in the SCISR register.

Bit 3 = **TE** *Transmitter enable*
This bit enables the transmitter. It is set and cleared by software.
0: Transmitter is disabled
1: Transmitter is enabled

Notes:

– During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word.

– When TE is set there is a 1 bit-time delay before the transmission starts.

Bit 2 = **RE** *Receiver enable*
This bit enables the receiver. It is set and cleared by software.
0: Receiver is disabled in the SCISR register
1: Receiver is enabled and begins searching for a start bit

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

Notes:

– 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 wakeup 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 will send a BREAK word at the end of the current word.

DATA REGISTER (SCIDR)
Read/Write
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

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 1).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 1).

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

If LHE bit is set due to this error during Fields other than LIN Synch Field or if LASE bit is reset then the current received Header is discarded and the SCI searches for a new Break Field.

Note on LIN Header Time-out Limit

According to the LIN specification, the maximum length of a LIN Header which does not cause a timeout is equal to 1.4 * (34 + 1) = 49 T_{BIT_MASTER}.

T_{BIT_MASTER} refers to the master baud rate.

When checking this timeout, the slave node is desynchronized for the reception of the LIN Break and Synch fields. Consequently, a margin must be allowed, taking into account the worst case: This occurs when the LIN identifier lasts exactly 10 T_{BIT_MASTER} periods. In this case, the LIN Break and Synch fields last 49 - 10 = 39T_{BIT_MASTER} periods.

Assuming the slave measures these first 39 bits with a desynchronized clock of 15.5%. This leads to a maximum allowed Header Length of:

39 x (1/0.845) T_{BIT_MASTER} + 10T_{BIT_MASTER}

= 56.15 T_{BIT_SLAVE}

A margin is provided so that the time-out occurs when the header length is greater than 57 T_{BIT_SLAVE} periods. If it is less than or equal to 57 T_{BIT_SLAVE} periods, then no timeout occurs.

LIN Header Length

Even if no timeout occurs on the LIN Header, it is possible to have access to the effective LIN header Length (T_{HEADER}) through the LHL register. This allows monitoring at software level the T_{FRAME_MAX} condition given by the LIN protocol.

This feature is only available when LHDM bit = 1 or when LASE bit = 1.

Mute Mode and Errors

In mute mode when LHDM bit = 1, if an LHE error occurs during the analysis of the LIN Synch Field or if a LIN Header Time-out occurs then the LHE bit is set but it does not wake up from mute mode. In this case, the current header analysis is discarded. If needed, the software has to reset LSF bit. Then the SCI searches for a new LIN header.

In mute mode, if a framing error occurs on a data (which is not a break), it is discarded and the FE bit is not set.

When LHDM bit = 1, any LIN header which respects the following conditions causes a wake-up from mute mode:

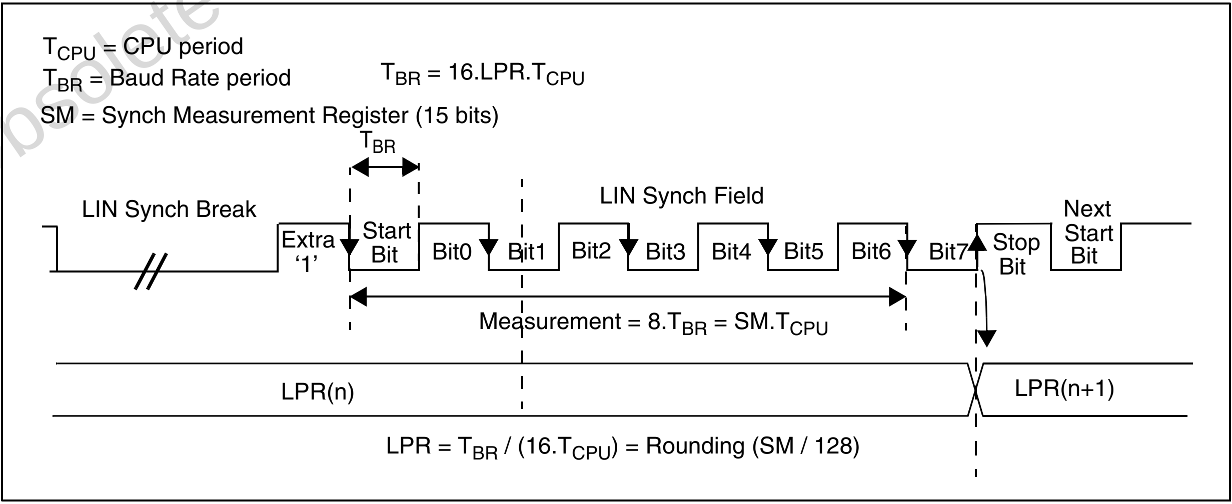
- A valid LIN Break Field (at least 11 dominant bits followed by a recessive bit)

- A valid LIN Synch Field (without deviation error)

- A LIN Identifier Field without framing error. Note that a LIN parity error on the LIN Identifier Field does not prevent wake-up from mute mode.

- No LIN Header Time-out should occur during Header reception.

Figure 83. LIN Synch Field Measurement



T_{CPU} = CPU period
T_{BR} = Baud Rate period
T_{BR} = 16.LPR.T_{CPU}
SM = Synch Measurement Register (15 bits)
T_{BR}
LIN Synch Break
LIN Synch Field
Extra '1'
Start Bit
Bit0
Bit1
Bit2
Bit3
Bit4
Bit5
Bit6
Bit7
Stop Bit
Next Start Bit
Measurement = 8.T_{BR} = SM.T_{CPU}
LPR(n)
LPR(n+1)
LPR = T_{BR} / (16.T_{CPU}) = Rounding (SM / 128)

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Master/Slave) (Cont'd)

Table 24. LINSCI1 Register Map and Reset Values

Addr. (Hex.)	Register Name	7	6	5	4	3	2	1	0
48	**SCI1SR** Reset Value	TDRE 1	TC 1	RDRF 0	IDLE 0	OR/LHE 0	NF 0	FE 0	PE 0
49	**SCI1DR** Reset Value	DR7 -	DR6 -	DR5 -	DR4 -	DR3 -	DR2 -	DR1 -	DR0 -
4A	**SCI1BRR** **LPR** (LIN Slave Mode) Reset Value	SCP1 LPR7 0	SCP0 LPR6 0	SCT2 LPR5 0	SCT1 LPR4 0	SCT0 LPR3 0	SCR2 LPR2 0	SCR1 LPR1 0	SCR0 LPR0 0
4B	**SCI1CR1** Reset Value	R8 x	T8 0	SCID 0	M 0	WAKE 0	PCE 0	PS 0	PIE 0
4C	**SCI1CR2** Reset Value	TIE 0	TCIE 0	RIE 0	ILIE 0	TE 0	RE 0	RWU 0	SBK 0
4D	**SCI1CR3** **Reset Value**	LDUM 0	LINE 0	LSLV 0	LASE 0	LHDM 0	LHIE 0	LHDF 0	LSF 0
4E	**SCI1ERPR** **LHLR** (LIN Slave Mode) Reset Value	ERPR7 LHL7 0	ERPR6 LHL6 0	ERPR5 LHL5 0	ERPR4 LHL4 0	ERPR3 LHL3 0	ERPR2 LHL2 0	ERPR1 LHL1 0	ERPR0 LHL0 0
4F	**SCI1ETPR** **LPFR** (LIN Slave Mode) Reset Value	ETPR7 0 0	ETPR6 0 0	ETPR5 0 0	ETPR4 0 0	ETPR3 LPFR3 0	ETPR2 LPFR2 0	ETPR1 LPFR1 0	ETPR0 LPFR0 0

12.4.2 On-Chip Peripherals

$T_A = 25°C$, $f_{CPU} = 8$ MHz.

Symbol	Parameter	Conditions		Typ	Unit
$I_{DD(TIM)}$	16-bit Timer supply current[1]		$V_{DD} = 5.0V$	50	µA
$I_{DD(TIM8)}$	8-bit Timer supply current [1]				
$I_{DD(ART)}$	ART PWM supply current[2]			75	
$I_{DD(SPI)}$	SPI supply current[3]			400	
$I_{DD(SCI)}$	SCI supply current[4]				
$I_{DD(ADC)}$	ADC supply current when converting[5]				

Notes:

1. Data based on a differential I_{DD} measurement between reset configuration (timer counter running at $f_{CPU}/4$) and timer counter stopped (only TIMD bit set). Data valid for one timer.
2. Data based on a differential I_{DD} measurement between reset configuration (timer stopped) and timer counter enabled (only TCE bit set).
3. Data based on a differential I_{DD} measurement between reset configuration (SPI disabled) and a permanent SPI master communication at maximum speed (data sent equal to 55h).This measurement includes the pad toggling consumption.
4. Data based on a differential I_{DD} measurement between SCI low power state (SCID = 1) and a permanent SCI data transmit sequence. Data valid for one SCI.
5. Data based on a differential I_{DD} measurement between reset configuration and continuous A/D conversions.

CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 112. $\overline{\text{RESET}}$ Pin Protection When LVD Is Enabled[1)2)]

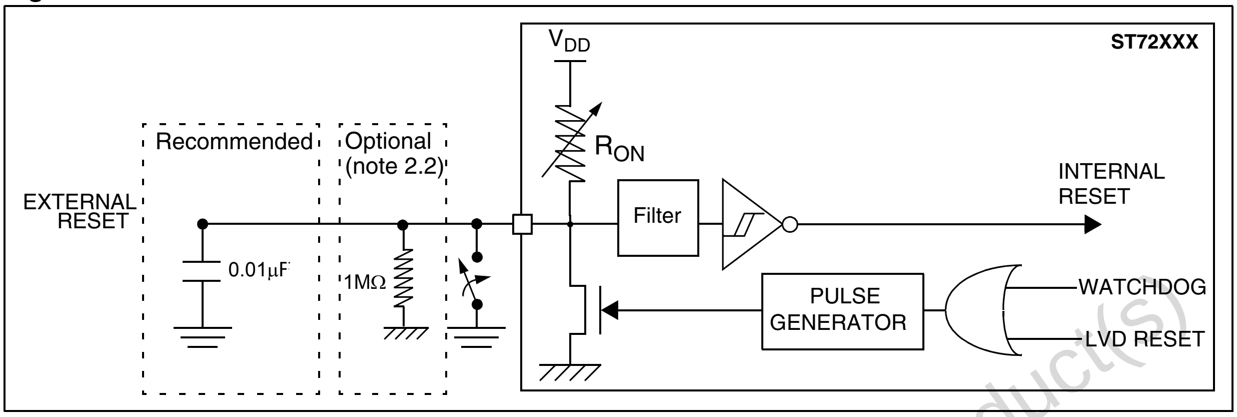
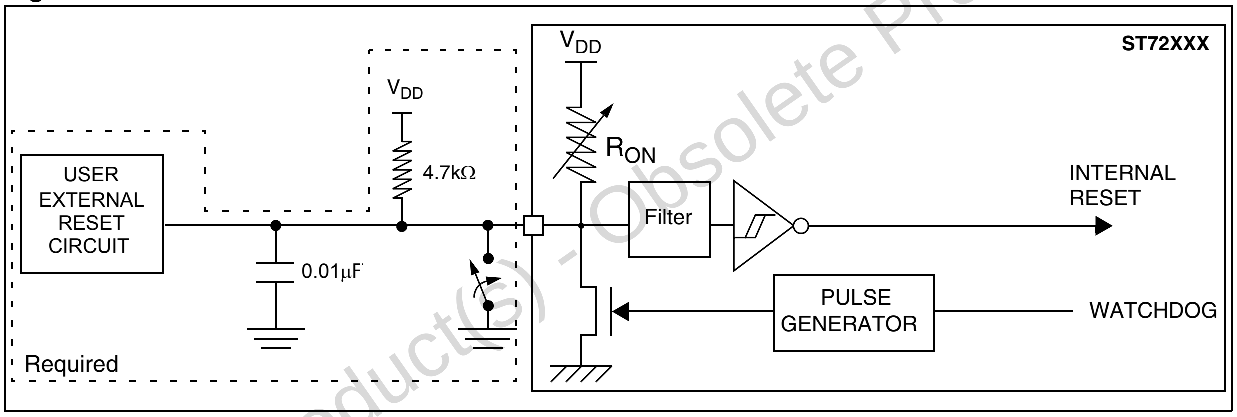


Figure 113. $\overline{\text{RESET}}$ Pin Protection When LVD Is Disabled[1)]



Note 1:

1.1 The reset network protects the device against parasitic resets.

1.2 The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).

1.3 Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{\text{RESET}}$ pin can go below the V_{IL} max. level specified in Section 12.10.1. Otherwise the reset will not be taken into account internally.

1.4 Because the reset circuit is designed to allow the internal RESET to be output in the $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin (by an external pull-up for example) is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in Section 12.2.2 on page 179.

Note 2:

2.1 When the LVD is enabled, it is mandatory not to connect a pull-up resistor. A 10nF pull-down capacitor is recommended to filter noise on the reset line.

2.2. In case a capacitive power supply is used, it is recommended to connect a1MW pull-down resistor to the $\overline{\text{RESET}}$ pin to discharge any residual voltage induced by this capacitive power supply (this will add 5µA to the power consumption of the MCU).

2.3. Tips when using the LVD:

– 1. Check that all recommendations related to reset circuit have been applied (see notes above)

– 2. Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709. If this cannot be done, it is recommended to put a 100nF + 1MW pull-down on the $\overline{\text{RESET}}$ pin.

– 3. The capacitors connected on the RESET pin and also the power supply are key to avoiding any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: Replace 10nF pull-down on the $\overline{\text{RESET}}$ pin with a 5µF to 20µF capacitor.

14 DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in user programmable versions (FLASH) as well as in factory coded versions (ROM/FASTROM).

ST72361 devices are ROM versions. ST72P361 devices are Factory Advanced Service Technique ROM (FASTROM) versions: They are factory-programmed HDFlash devices.

ST72F361 FLASH devices are shipped to customers with a default content (FFh), while ROM factory coded parts contain the code supplied by the customer. This implies that FLASH devices have to be configured by the customer using the Option Bytes while the ROM devices are factory-configured.

14.1 FLASH OPTION BYTES

The option bytes allows the hardware configuration of the microcontroller to be selected. They have no address in the memory map and can be accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the FLASH is fixed to FFh. To program directly the FLASH devices using ICP, FLASH devices are shipped to customers with a reserved internal clock source enabled. In masked ROM devices, the option bytes are fixed in hardware by the ROM code (see option list).

OPTION BYTE 0

OPT7 = **WDGHALT** *Watchdog reset on HALT*
This option bit determines if a RESET is generated when entering HALT mode while the Watchdog is active.
0: No Reset generation when entering Halt mode
1: Reset generation when entering Halt mode

OPT6 = **WDGSW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware (watchdog always enabled)
1: Software (watchdog to be enabled by software)

OPT5 = Reserved, must be kept at default value.

OPT4 = **LVD** *Voltage detection*
This option bit enables the voltage detection block (LVD).

Selected Low Voltage Detector	VD
LVD Off	1
LVD On	0

OPT3 = **PLL OFF** *PLL activation*
This option bit activates the PLL which allows multiplication by two of the main input clock frequency. The PLL is guaranteed only with an input frequency between 2 and 4 MHz.
0: PLL x2 enabled
1: PLL x2 disabled
Caution: The PLL can be enabled only if the "OSC RANGE" (OPT11:10) bits are configured to "MP - 2~4 MHz". Otherwise, the device functionality is not guaranteed.

	STATIC OPTION BYTE 0								STATIC OPTION BYTE 1							
	7							0	7							0
	WDG		Reserved	LVD	PLLOFF	PKG		FMP_R	AFI_MAP		OSCTYPE		OSCRANGE		Reserved	RSTC
	HALT	SW				1	0		1	0	1	0	1	0		
Default(*)	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1

(*): Option bit values programmed by ST

16 IMPORTANT NOTES

16.1 ALL DEVICES

16.1.1 RESET Pin Protection with LVD Enabled

As mentioned in note 2 below Figure 112 on page 199, when the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

16.1.2 Clearing Active Interrupts Outside Interrupt Routine

When an active interrupt request occurs at the same time as the related flag or interrupt mask is being cleared, the CC register may be corrupted.

Concurrent interrupt context

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

– The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) 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 flag or interrupt mask
RIM
```

Nested interrupt context

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

– The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine

– The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine with higher or identical priority level

– The interrupt request is cleared (flag reset or interrupt mask) 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 flag or interrupt mask
POP CC
```

16.1.3 External Interrupt Missed

To avoid any risk of generating a parasitic interrupt, the edge detector is automatically disabled for one clock cycle during an access to either DDR and OR. Any input signal edge during this period will not be detected and will not generate an interrupt.

This case can typically occur if the application refreshes the port configuration registers at intervals during runtime.

Workaround

The workaround is based on software checking the level on the interrupt pin before and after writing to the PxOR or PxDDR registers. If there is a level change (depending on the sensitivity programmed for this pin) the interrupt routine is invoked using the call instruction with three extra PUSH instructions before executing the interrupt routine (this is to make the call compatible with the IRET instruction at the end of the interrupt service routine).

But detection of the level change does ensure that edge occurs during the critical 1 cycle duration and the interrupt has been missed. This may lead to occurrence of same interrupt twice (one hardware and another with software call).

To avoid this, a semaphore is set to '1' before checking the level change. The semaphore is changed to level '0' inside the interrupt routine. When a level change is detected, the semaphore status is checked and if it is '1' this means that the last interrupt has been missed. In this case, the interrupt routine is invoked with the call instruction.

There is another possible case, that is, if writing to PxOR or PxDDR is done with global interrupts disabled (interrupt mask bit set). In this case, the semaphore is changed to '1' when the level change is detected. Detecting a missed interrupt is done after the global interrupts are enabled (interrupt mask bit reset) and by checking the status of the semaphore. If it is '1' this means that the last interrupt was missed and the interrupt routine is invoked with the call instruction.

To implement the workaround, the following software sequence is to be followed for writing into the PxOR/PxDDR registers. The example is for Port PF1 with falling edge interrupt sensitivity. The

IMPORTANT NOTES (Cont'd)

16.1.5 Header Time-out Does Not Prevent Wake-up from Mute Mode

Normally, when LINSCI is configured in LIN slave mode, if a header time-out occurs during a LIN header reception (that is, header length > 57 bits), the LIN Header Error bit (LHE) is set, an interrupt occurs to inform the application but the LINSCI should stay in mute mode, waiting for the next header reception.

Problem Description

The LINSCI sampling period is Tbit / 16. If a LIN Header time-out occurs between the 9th and the 15th sample of the Identifier Field Stop Bit (refer to Figure 130), the LINSCI wakes up from mute mode. Nevertheless, LHE is set and LIN Header Detection Flag (LHDF) is kept cleared.

In addition, if LHE is reset by software before this 15th sample (by accessing the SCISR register and reading the SCIDR register in the LINSCI interrupt routine), the LINSCI will generate another LINSCI interrupt (due to the RDRF flag setting).

Impact on application

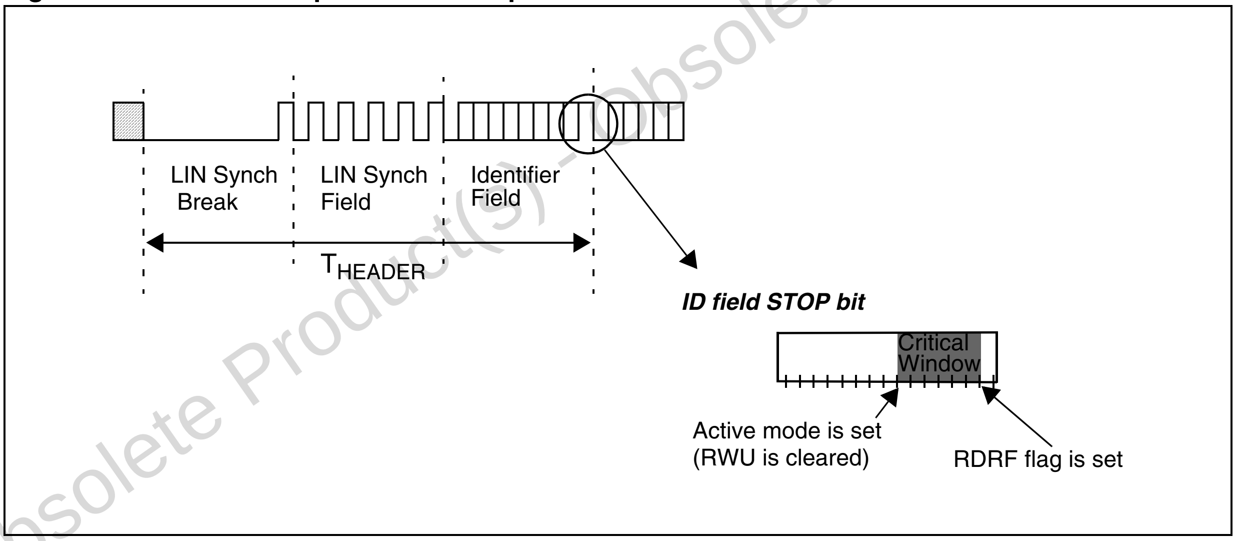
Software may execute the interrupt routine twice after header reception.

Moreover, in reception mode, as the receiver is no longer in mute mode, an interrupt will be generated on each data byte reception.

Workaround

The problem can be detected in the LINSCI interrupt routine. In case of timeout error (LHE is set and LHLR is loaded with 00h), the software can check the RWU bit in the SCICR2 register. If RWU is cleared, it can be set by software. Refer to Figure 131 on page 221. Workaround is shown in bold characters.

Figure 130. Header Reception Event Sequence



LIN Synch Break
LIN Synch Field
Identifier Field
T_{HEADER}
ID field STOP bit
Critical Window
Active mode is set (RWU is cleared)
RDRF flag is set