STMicroelectronics - ST72F561AR9T3 Datasheet

What is "Embedded - Microcontrollers"?

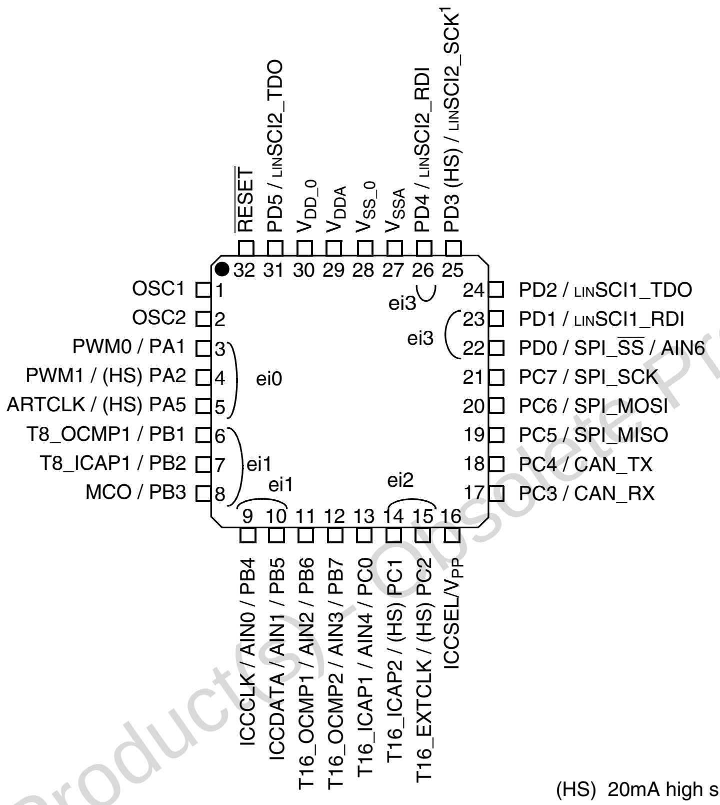
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	CANbus, LINbusSCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	48
Program Memory Size	60KB (60K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f561ar9t3

PIN DESCRIPTION (Cont'd)

Figure 4. LQFP 32-Pin Package Pinout



(HS) 20mA high sink capability

eix associated external interrupt vector

(*) : by option bit:
T16_ICAP1 can be moved to PD4
T16_ICAP2 can be moved to PD1
T16_OCMP1 can be moved to PD3
T16_OCMP2 can be moved to PD5

For external pin connection guidelines, refer to "ELECTRICAL CHARACTERISTICS" on page 219.

Pin n°			Pin Name	Type	Level		Port						Main function (after reset)	Alternate function	
							Input				Output				
LQFP64	LQFP44	LQFP32			Input	Output	float	wpu	int	ana	OD	PP			
23	14	11	PB6 / AIN2 / T16_OCMP1	I/O	C_T		**X**	X		RB	X	X	Port B6	TIM16 Output Compare 1	ADC Analog Input 2
24	15	-	V_{SS_2}	S									Digital Ground Voltage		
25	16	-	V_{DD_2}	S									Digital Main Supply Voltage		
26	17	12	PB7 /AIN3 / T16_OCMP2	I/O	C_T		**X**	X		RB	X	X	Port B7	TIM16 Output Compare 2	ADC Analog Input 3
27	18	13	PC0 / AIN4 / T16_ICAP1	I/O	C_T		**X**	X		RB	X	X	Port C0	TIM16 Input Capture 1	ADC Analog Input 4
28	19	14	PC1 (HS) / T16_ICAP2	I/O	C_T	HS	**X**	ei2			X	X	Port C1	TIM16 Input Capture 2	
29	20	15	PC2 (HS) / T16_EXTCLK	I/O	C_T	HS	**X**		ei2		X	X	Port C2	TIM16 External Clock input	
30	21	-	PE4	I/O	T_T		**X**	X			X	X	Port E4		
31	-	-	NC	Not Connected											
32	22	16	V_{PP}	I									Flash programming voltage. Must be tied low in user mode.		
33	23	17	PC3 / CANRX	I/O	C_T		**X**	X			X	X	Port C3	CAN Receive Data Input	
34	24	18	PC4 / CANTX	I/O	C_T		**X**					X[2]	Port C4	CAN Transmit Data Output	
35	-	-	PE5	I/O	T_T		**X**	X			X	X	Port E5		
36	25	-	PE6 / AIN5	I/O	T_T		**X**	X		X	X	X	Port E6	ADC Analog Input 5	
37	26	19	PC5 /MISO	I/O	C_T		**X**	X			X	X	Port C5	SPI Master In/Slave Out	
38	27	20	PC6 / MOSI	I/O	C_T		**X**	X			X	X	Port C6	SPI Master Out/Slave In	
39	28	21	PC7 /SCK	I/O	C_T		**X**	X			X	X	Port C7	SPI Serial Clock	
40	-	-	V_{SS_1}	S									Digital Ground Voltage		
41	-	-	V_{DD_1}	S									Digital Main Supply Voltage		
42	29	22	PD0 / $\overline{SS}$/ AIN6	I/O	C_T		**X**	ei3		X	X	X	Port D0	SPI Slave Select	ADC Analog Input 6
43	-	-	PE7	I/O	T_T		**X**	X			X	X	Port E7		
44	-	-	PF0	I/O	T_T		**X**	X			X	X	Port F0		
45	30	-	PF1 / AIN7	I/O	T_T		**X**	X		X	X	X	Port F1	ADC Analog Input 7	
46	31	-	PF2 / AIN8	I/O	T_T		**X**	X		X	X	X	Port F2	ADC Analog Input 8	
47	32	23	PD1 / SCI1_RDI	I/O	C_T		**X**		ei3		X	X	Port D1	LINSCI1 Receive Data input	
48	33	24	PD2 / SCI1_TDO	I/O	C_T		**X**	X			X	X	Port D2	LINSCI1 Transmit Data output	
49	-	-	PF3 / AIN9	I/O	T_T		**X**	X		X	X	X	Port F3	ADC Analog Input 9	
50	-	-	PF4	I/O	T_T		**X**	X			X	X	Port F4		
51	-	-	TLI	I	C_T		**X**		X				Top level interrupt input pin		
52	34	-	PF5	I/O	T_T		**X**	X			X	X	Port F5		
53	35	25	PD3 (HS) / SCI2_SCK	I/O	C_T	HS	**X**	X			X	X	Port D3	LINSCI2 Serial Clock Output	

Address	Block	Register Label	Register Name	Reset Status	Remarks
0048h	LINSCI1 (LIN Master/ Slave)	SCI1ISR	SCI1 Status Register	C0h	Read Only
0049h		SCI1DR	SCI1 Data Register	xxh	R/W
004Ah		SCI1BRR	SCI1 Baud Rate Register	00h	R/W
004Bh		SCI1CR1	SCI1 Control Register 1	xxh	R/W
004Ch		SCI1CR2	SCI1 Control Register 2	00h	R/W
004Dh		SCI1CR3	SCI1Control Register 3	00h	R/W
004Eh		SCI1ERPR	SCI1 Extended Receive Prescaler Register	00h	R/W
004Fh		SCI1ETPR	SCI1 Extended Transmit Prescaler Register	00h	R/W
0050h	Reserved Area (1 byte)				
0051h	16-BIT TIMER	T16CR2	Timer Control Register 2	00h	R/W
0052h		T16CR1	Timer Control Register 1	00h	R/W
0053h		T16CSR	Timer Control/Status Register	00h	R/W
0054h		T16IC1HR	Timer Input Capture 1 High Register	xxh	Read Only
0055h		T16IC1LR	Timer Input Capture 1 Low Register	xxh	Read Only
0056h		T16OC1HR	Timer Output Compare 1 High Register	80h	R/W
0057h		T16OC1LR	Timer Output Compare 1 Low Register	00h	R/W
0058h		T16CHR	Timer Counter High Register	FFh	Read Only
0059h		T16CLR	Timer Counter Low Register	FCh	Read Only
005Ah		T16ACHR	Timer Alternate Counter High Register	FFh	Read Only
005Bh		T16ACLR	Timer Alternate Counter Low Register	FCh	Read Only
005Ch		T16IC2HR	Timer Input Capture 2 High Register	xxh	Read Only
005Dh		T16IC2LR	Timer Input Capture 2 Low Register	xxh	Read Only
005Eh		T16OC2HR	Timer Output Compare 2 High Register	80h	R/W
005Fh		T16OC2LR	Timer Output Compare 2 Low Register	00h	R/W
0060h	LINSCI2 (LIN Master)	SCI2SR	SCI2 Status Register	C0h	Read Only
0061h		SCI2DR	SCI2 Data Register	xxh	R/W
0062h		SCI2BRR	SCI2 Baud Rate Register	00h	R/W
0063h		SCI2CR1	SCI2 Control Register 1	xxh	R/W
0064h		SCI2CR2	SCI2 Control Register 2	00h	R/W
0065h		SCI2CR3	SCI2 Control Register 3	00h	R/W
0066h		SCI2ERPR	SCI2 Extended Receive Prescaler Register	00h	R/W
0067h		SCI2ETPR	SCI2 Extended Transmit Prescaler Register	00h	R/W

10 ON-CHIP PERIPHERALS

10.1 WINDOW WATCHDOG (WWDG)

10.1.1 Introduction

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

10.1.2 Main Features

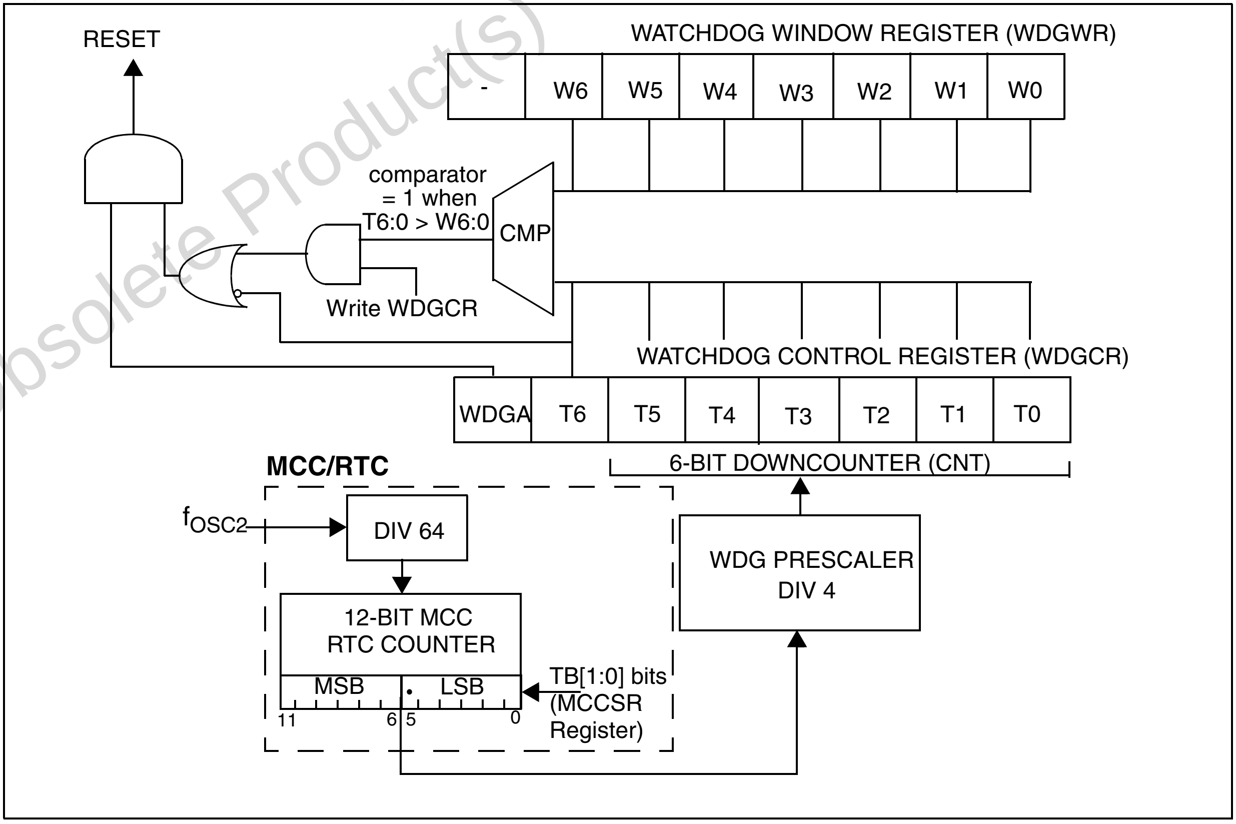
- Programmable free-running downcounter
- Conditional reset
  - Reset (if watchdog activated) when the downcounter value becomes less than 40h
  - Reset (if watchdog activated) if the downcounter is reloaded outside the window (see Figure 4)
- Hardware/Software Watchdog activation (selectable by option byte)
- Optional reset on HALT instruction (configurable by option byte)

10.1.3 Functional Description

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

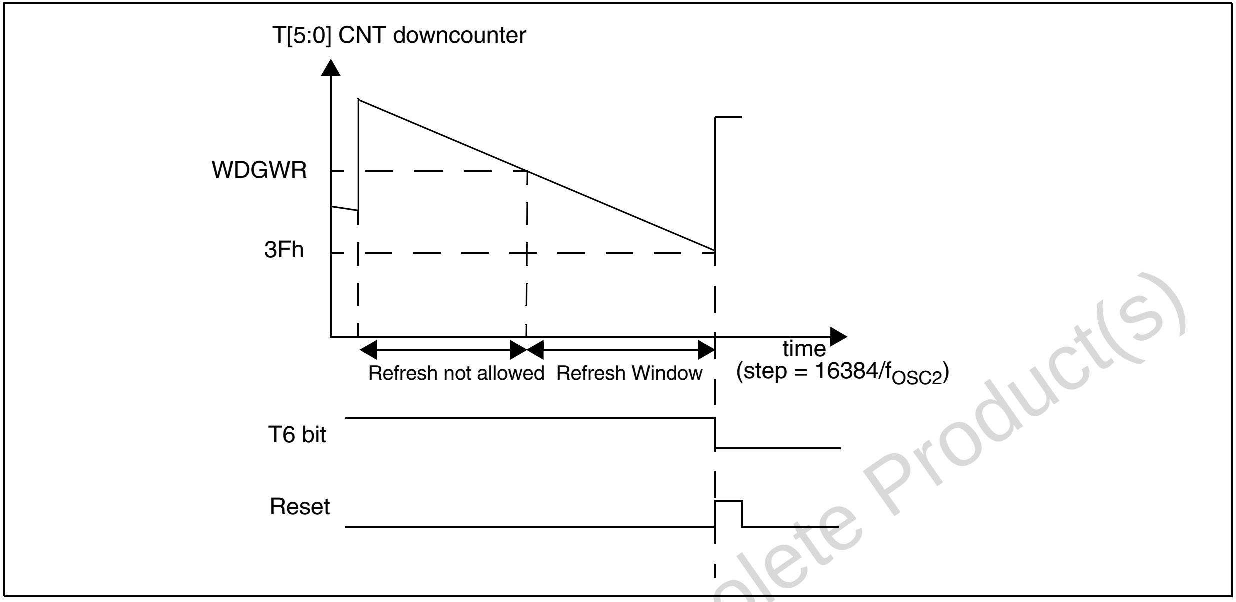
If the watchdog is activated (the WDGA bit is set) and when the 7-bit downcounter (T[6:0] bits) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 30µs. If the software reloads the counter while the counter is greater than the value stored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

Figure 34. Watchdog Block Diagram



WINDOW WATCHDOG (Cont'd)

Figure 37. Window Watchdog Timing Diagram



10.1.6 Low Power Modes

Mode	Description		
SLOW	No effect on Watchdog: The downcounter continues to decrement at normal speed.		
WAIT	No effect on Watchdog: The downcounter continues to decrement.		
HALT	**OIE bit in MCCSR register**	**WDGHALT bit in Option Byte**	
	0	0	No Watchdog reset is generated. The MCU enters Halt mode. The Watchdog counter is decremented once and then stops counting and is no longer able to generate a watchdog reset until the MCU receives an external interrupt or a reset. If an interrupt is received (refer to interrupt table mapping to see interrupts which can occur in halt mode), the Watchdog restarts counting after 256 or 4096 CPU clocks. If a reset is generated, the Watchdog is disabled (reset state) unless Hardware Watchdog is selected by option byte. For application recommendations see Section 0.1.8 below.
	0	1	A reset is generated instead of entering halt mode.
ACTIVE HALT	1	x	No reset is generated. The MCU enters Active Halt mode. The Watchdog counter is not decremented. It stop counting. When the MCU receives an oscillator interrupt or external interrupt, the Watchdog restarts counting immediately. When the MCU receives a reset the Watchdog restarts counting after 256 or 4096 CPU clocks.

10.1.7 Hardware Watchdog Option

If Hardware Watchdog is selected by option byte, the watchdog is always active and the WDGA bit in the WDGCR is not used. Refer to the Option Byte description.

10.1.8 Using Halt Mode with the WDG (WDGHALT option)

The following recommendation applies if Halt mode is used when the watchdog is enabled.

– Before executing the HALT instruction, refresh the WDG counter, to avoid an unexpected WDG reset immediately after waking up the microcontroller.

PWM AUTO-RELOAD TIMER (Cont'd)

Output compare and Time base interrupt

On overflow, the OVF flag of the ARTCSR register is set and an overflow interrupt request is generated if the overflow interrupt enable bit, OIE, in the ARTCSR register, is set. The OVF flag must be reset by the user software. This interrupt can be used as a time base in the application.

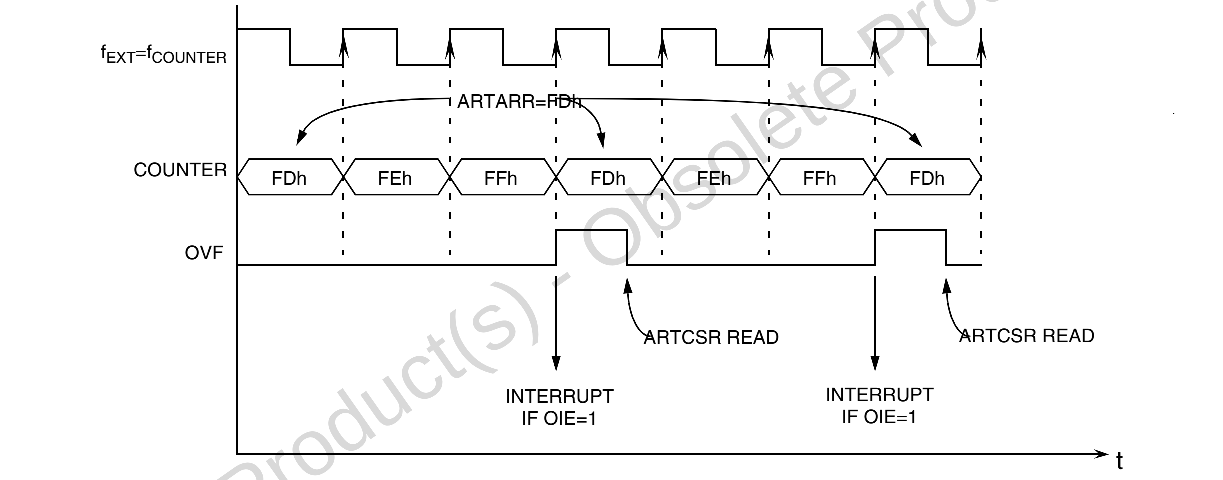
External clock and event detector mode

Using the f_{EXT} external prescaler input clock, the auto-reload timer can be used as an external clock event detector. In this mode, the ARTARR register is used to select the n_{EVENT} number of events to be counted before setting the OVF flag.

n_{EVENT} = 256 - ARTARR

Caution: The external clock function is not available in HALT mode. If HALT mode is used in the application, prior to executing the HALT instruction, the counter must be disabled by clearing the TCE bit in the ARTCSR register to avoid spurious counter increments.

Figure 44. External Event Detector Example (3 counts)



PWM AUTO-RELOAD TIMER (Cont'd)

Input Capture Function

Input Capture mode allows the measurement of external signal pulse widths through ARTICRx registers.

Each input capture can generate an interrupt independently on a selected input signal transition. This event is flagged by a set of the corresponding CFx bits of the Input Capture Control/Status register (ARTICCSR).

These input capture interrupts are enabled through the CIEx bits of the ARTICCSR register.

The active transition (falling or rising edge) is software programmable through the CSx bits of the ARTICCSR register.

The read only input capture registers (ARTICRx) are used to latch the auto-reload counter value when a transition is detected on the ARTICx pin (CFx bit set in ARTICCSR register). After fetching the interrupt vector, the CFx flags can be read to identify the interrupt source.

Note: After a capture detection, data transfer in the ARTICRx register is inhibited until the next read (clearing the CFx bit).
The timer interrupt remains pending while the CFx flag is set when the interrupt is enabled (CIEx bit set). This means, the ARTICRx register has to be read at each capture event to clear the CFx flag.

The timing resolution is given by auto-reload counter cycle time ($1/f_{COUNTER}$).

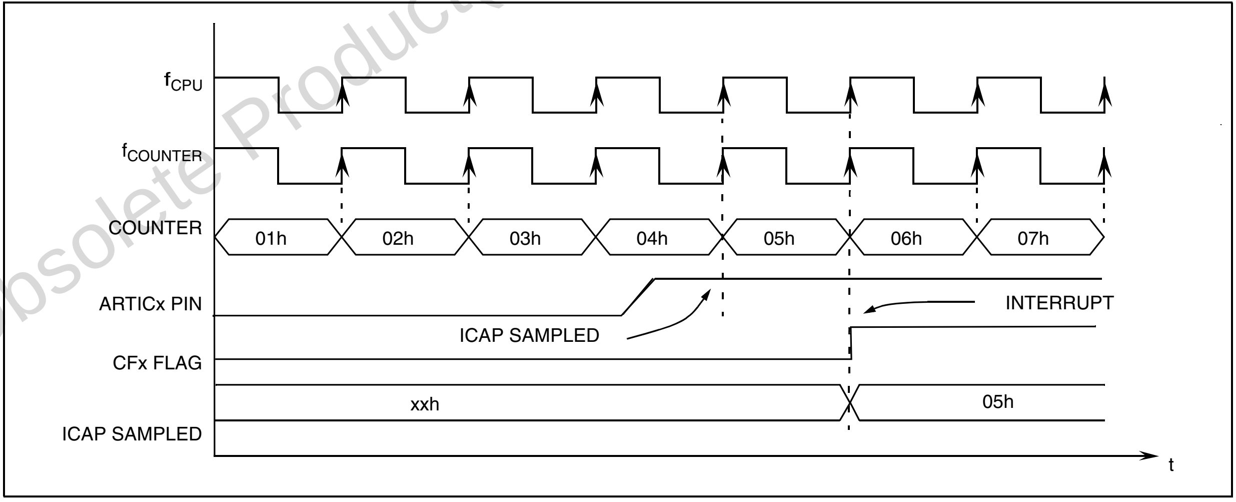
Note: During HALT mode, input capture is inhibited (the ARTICRx is never reloaded) and only the external interrupt capability can be used.

Note: The ARTICx signal is synchronized on CPU clock. It takes two rising edges until ARTICRx is latched with the counter value. Depending on the prescaler value and the time when the ICAP event occurs, the value loaded in the ARTICRx register may be different.

If the counter is clocked with the CPU clock, the value latched in ARTICRx is always the next counter value after the event on ARTICx occurred (Figure 45).

If the counter clock is prescaled, it depends on the position of the ARTICx event within the counter cycle (Figure 46).

Figure 45. Input Capture Timing Diagram, $f_{COUNTER} = f_{CPU}$



16-BIT TIMER (Cont'd)

Figure 52. Input Capture Block Diagram

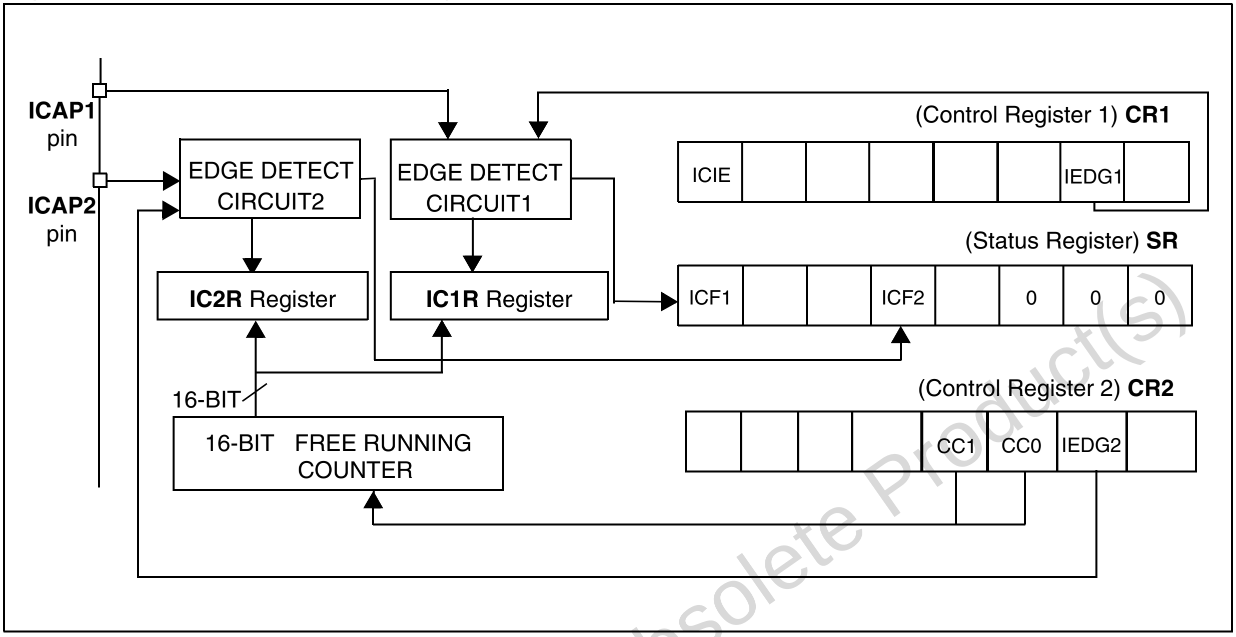
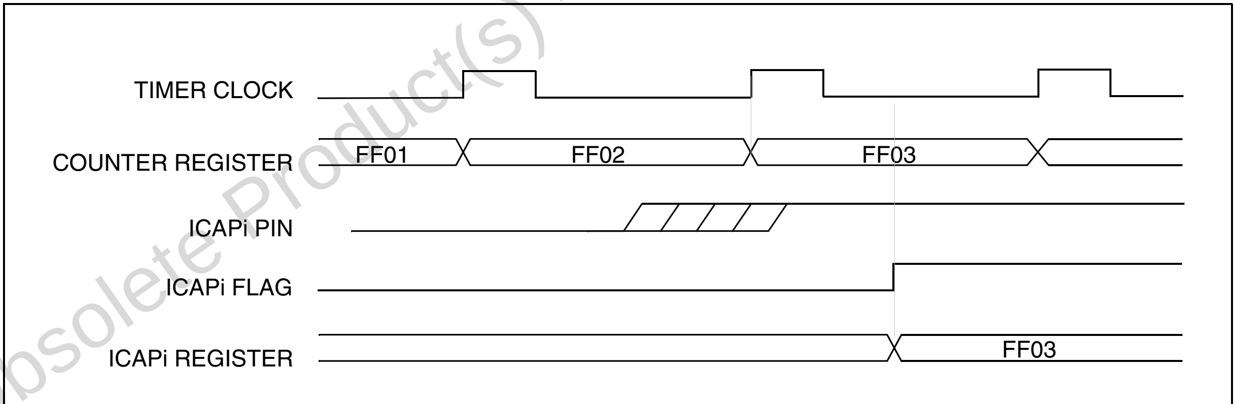


Figure 53. Input Capture Timing Diagram



Note: The rising edge is the active edge.

16-BIT TIMER (Cont'd)

10.4.3.4 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 17 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 OC*i*R 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 17 Clock Control Bits)

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)

Clearing the output compare interrupt request (that is, clearing the OCF*i* bit) is done by:

1. Reading the SR register while the OCF*i* bit is set.

2. An access (read or write) to the OC*i*LR register.

The following procedure is recommended to prevent the OCF*i* bit from being set between the time it is read and the write to the OC*i*R register:

– Write to the OC*i*HR register (further compares are inhibited).

– Read the SR register (first step of the clearance of the OCF*i* bit, which may be already set).

– Write to the OC*i*LR register (enables the output compare function and clears the OCF*i* bit).

8-BIT TIMER (Cont'd)

Whatever the timer mode used (input capture, output compare, one pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFh to 00h then:

– The TOF bit of the SR register is set.

– A timer interrupt is generated if:

  – TOIE bit of the CR1 register is set and

  – I bit of the CC register is cleared.

If one of these conditions is false, the interrupt remains pending to be issued as soon as they are both true.

Clearing the overflow interrupt request is done in two steps:

1. Reading the SR register while the TOF bit is set.
2. An access (read or write) to the CTR register.

Notes: The TOF bit is not cleared by accesses to ACTR register. The advantage of accessing the ACTR register rather than the CTR 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).

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

10.7.5.4 Conventional Baud Rate Generation

The baud rates for the receiver and transmitter (Rx and Tx) are set independently and calculated as follows:

$$Tx = \frac{f_{CPU}}{(16*PR)*TR} \qquad Rx = \frac{f_{CPU}}{(16*PR)*RR}$$

with:

PR = 1, 3, 4 or 13 (see SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64,128

(see SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64,128

(see SCR[2:0] bits)

All these bits are in the SCIBRR register.

Example: If f_{CPU} is 8 MHz (normal mode) and if PR = 13 and TR = RR = 1, the transmit and receive baud rates are 38400 baud.

Note: The baud rate registers MUST NOT be changed while the transmitter or the receiver is enabled.

10.7.5.5 Extended Baud Rate Generation

The extended prescaler option gives a very fine tuning on the baud rate, using a 255 value prescaler, whereas the conventional Baud Rate Generator retains industry standard software compatibility.

The extended baud rate generator block diagram is described in Figure 3.

The output clock rate sent to the transmitter or to the receiver will be the output from the 16 divider divided by a factor ranging from 1 to 255 set in the SCIERPR or the SCIETPR register.

Note: The extended prescaler is activated by setting the SCIETPR or SCIERPR register to a value other than zero. The baud rates are calculated as follows:

$$Tx = \frac{f_{CPU}}{16*ETPR*(PR*TR)} \qquad Rx = \frac{f_{CPU}}{16*ERPR*(PR*RR)}$$

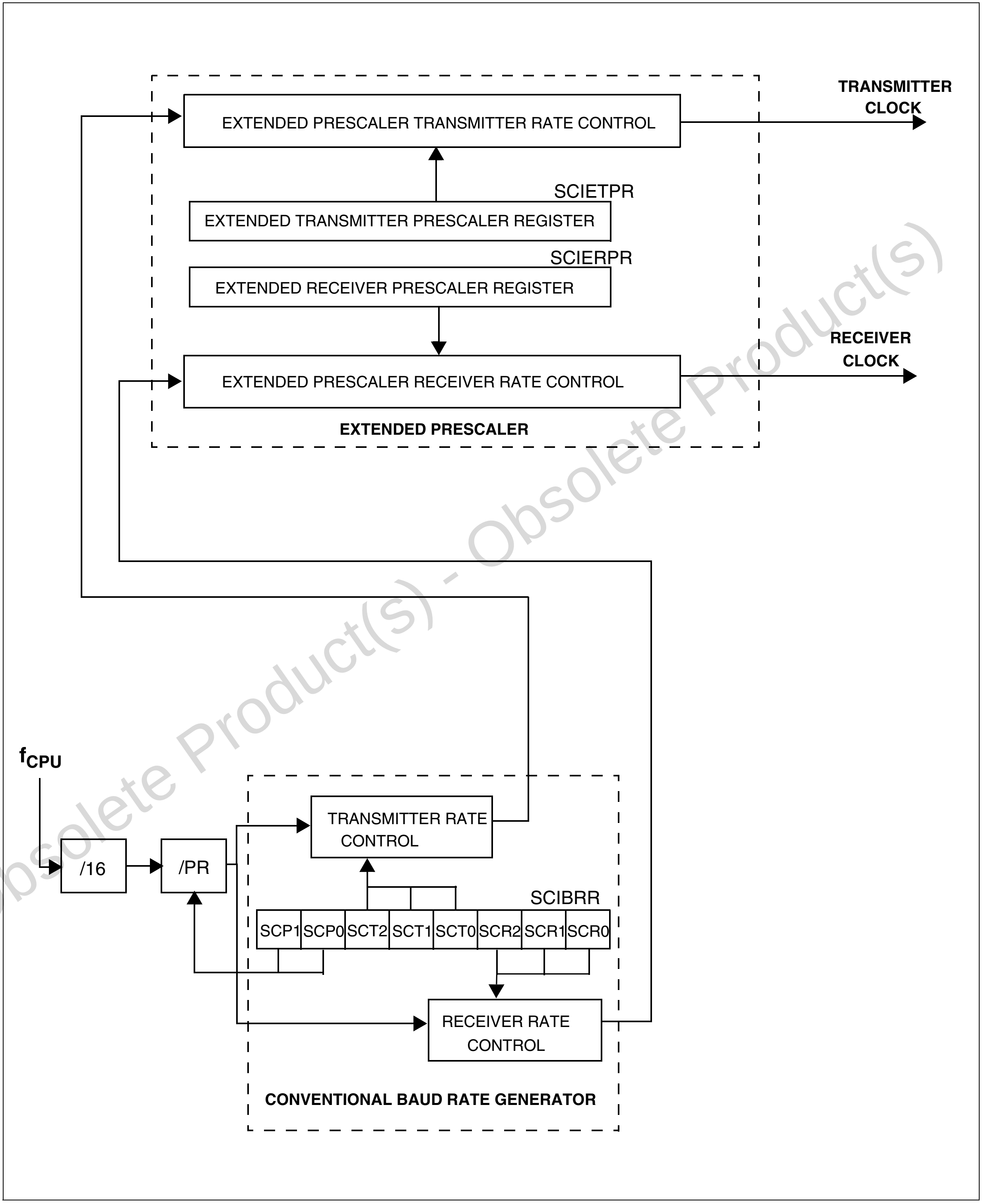
with:

ETPR = 1, ..., 255 (see SCIETPR register)

ERPR = 1, ..., 255 (see SCIERPR register)

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

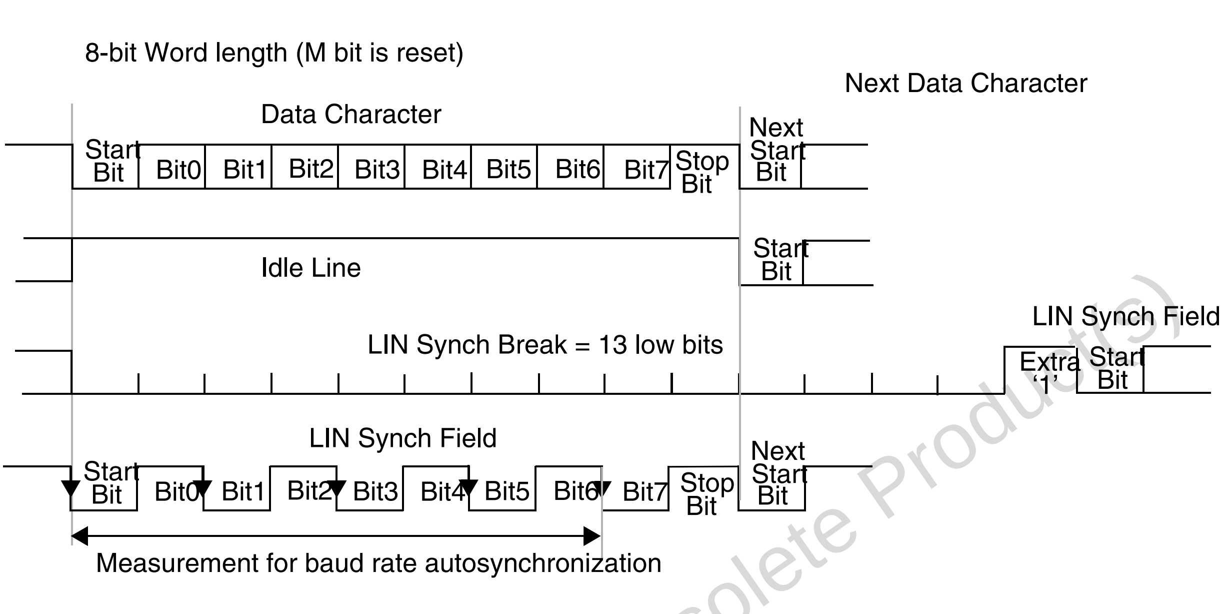
Figure 79. SCI Baud Rate and Extended Prescaler Block Diagram



EXTENDED PRESCALER TRANSMITTER RATE CONTROL
TRANSMITTER CLOCK
SCIETPR
EXTENDED TRANSMITTER PRESCALER REGISTER
SCIERPR
EXTENDED RECEIVER PRESCALER REGISTER
RECEIVER CLOCK
EXTENDED PRESCALER RECEIVER RATE CONTROL
EXTENDED PRESCALER
f_{CPU}
/16
/PR
TRANSMITTER RATE CONTROL
SCIBRR
SCP1
SCP0
SCT2
SCT1
SCT0
SCR2
SCR1
SCR0
RECEIVER RATE CONTROL
CONVENTIONAL BAUD RATE GENERATOR

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

Figure 80. LIN Characters



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

LIN PRESCALER FRACTION REGISTER (LPFR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	LPFR 3	LPFR 2	LPFR 1	LPFR 0

Bits 7:4 = Reserved.

Bits 3:0 = **LPFR[3:0]** *Fraction of LDIV*

These 4 bits define the fraction of the LIN Divider (LDIV):

LPFR[3:0]	Fraction (LDIV)
0h	0
1h	1/16
...	...
Eh	14/16
Fh	15/16

1. When initializing LDIV, the LPFR register must be written first. Then, the write to the LPR register will effectively update LDIV and so the clock generation.

2. In LIN Slave mode, if the LPR[7:0] register is equal to 00h, the transceiver and receiver input clocks are switched off.

Examples of LDIV coding:

Example 1: LPR = 27d and LPFR = 12d

This leads to:

Mantissa (LDIV) = 27d

Fraction (LDIV) = 12/16 = 0.75d

Therefore LDIV = 27.75d

Example 2: LDIV = 25.62d

This leads to:

LPFR = rounded(16*0.62d)

= rounded(9.92d) = 10d = Ah

LPR = mantissa (25.620d) = 25d = 1Bh

Example 3: LDIV = 25.99d

This leads to:

LPFR = rounded(16*0.99d)

= rounded(15.84d) = 16d

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

10.8.7 SCI Synchronous Transmission

The SCI transmitter allows the user to control a one way synchronous serial transmission. The SCLK pin is the output of the SCI transmitter clock. No clock pulses are sent to the SCLK pin during start bit and stop bit. Depending on the state of the LBCL bit in the SCICR3 register, clock pulses are or are not be generated during the last valid data bit (address mark). The CPOL bit in the SCICR3 register allows the user to select the clock polarity, and the CPHA bit in the SCICR3 register allows the user to select the phase of the external clock (see Figure 91, Figure 92 and Figure 93).

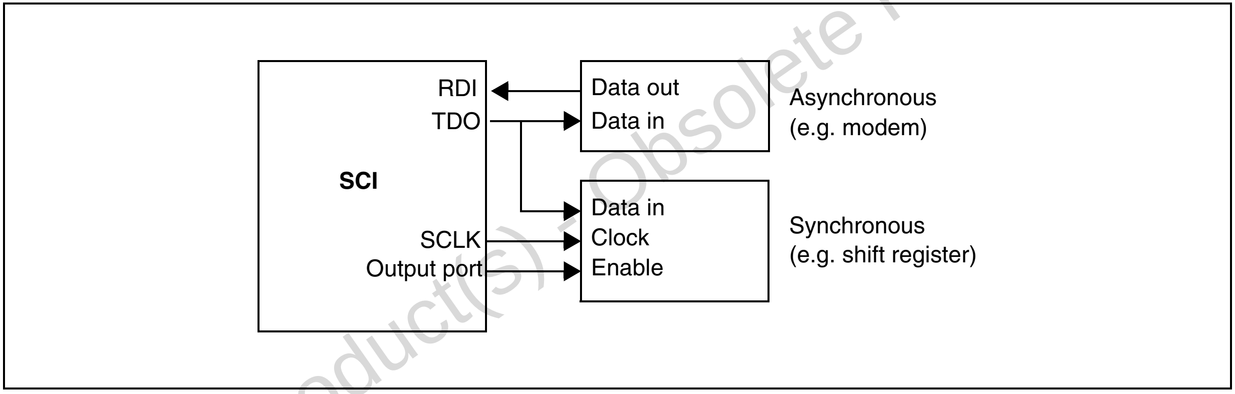
During idle, preamble and send break, the external SCLK clock is not activated.

These options allow the user to serially control peripherals which consist of shift registers, without losing any functions of the SCI transmitter which can still talk to other SCI receivers. These options do not affect the SCI receiver which is independent from the transmitter.

Note: The SCLK pin works in conjunction with the TDO pin. When the SCI transmitter is disabled (TE and RE = 0), the SCLK and TDO pins go into high impedance state.

Note: The LBCL, CPOL and CPHA bits have to be selected before enabling the transmitter to ensure that the clock pulses function correctly. These bits should not be changed while the transmitter is enabled.

Figure 91. SCI Example of Synchronous and Asynchronous Transmission

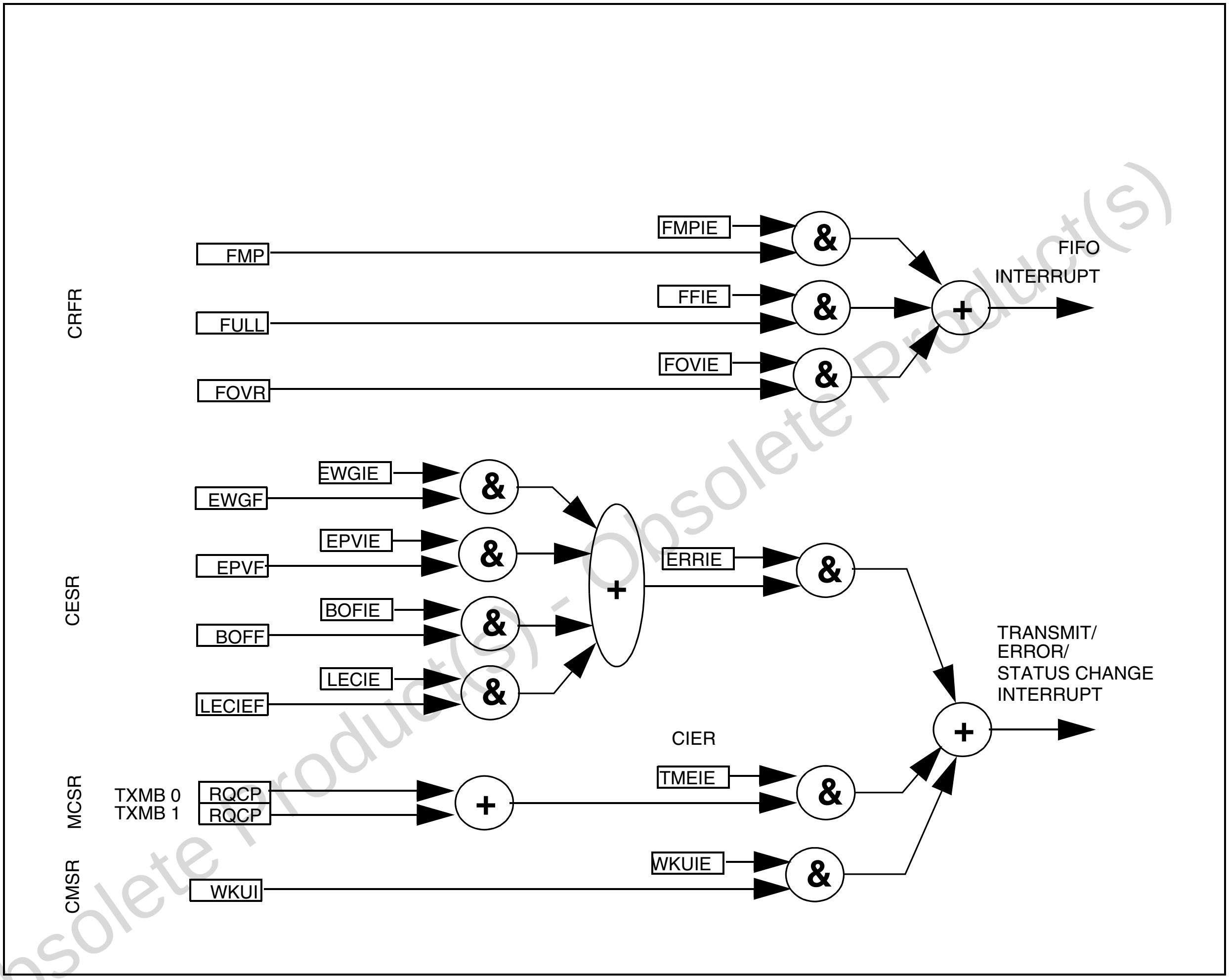


beCAN CONTROLLER (Cont'd)

10.9.5 Interrupts

Two interrupt vectors are dedicated to beCAN. Each interrupt source can be independently enabled or disabled by means of the CAN Interrupt Enable Register (CIER) and CAN Error Interrupt Enable register (CEIER).

Figure 108. Event flags and Interrupt Generation



beCAN CONTROLLER (Cont'd)

– The **FIFO interrupt** can be generated by the following events:

  – Reception of a new message, FMP bits in the CRFR0 register incremented.

  – FIFO0 full condition, FULL bit in the CRFR0 register set.

  – FIFO0 overrun condition, FOVR bit in the CRFR0 register set.

– The **transmit, error and status change interrupt** can be generated by the following events:

  – Transmit mailbox 0 becomes empty, RQCP0 bit in the CTSR register set.

  – Transmit mailbox 1 becomes empty, RQCP1 bit in the CTSR register set.

  – Error condition, for more details on error conditions please refer to the CAN Error Status register (CESR).

  – Wake-up condition, SOF monitored on the CAN Rx signal.

10.9.6 Register Access Protection

Erroneous access to certain configuration registers can cause the hardware to temporarily disturb the whole CAN network. Therefore the following registers can be modified by software only while the hardware is in initialization mode:

CBTR0, CBTR1, CFCR0, CFCR1, CFMR and CDGR registers.

Although the transmission of incorrect data will not cause problems at the CAN network level, it can severely disturb the application. A transmit mailbox can be only modified by software while it is in empty state (refer to Figure 7. Transmit Mailbox States).

The filters must be deactivated before their value can be modified by software. The modification of the filter configuration (scale or mode) can be done by software only in initialization mode.

beCAN CONTROLLER (Cont'd)

Bit 4 = **TXOK0** *Transmission OK for mailbox 0*
- Read
This bit is set by hardware when the transmission request on mailbox 0 has been completed successfully. Please refer to Figure 7.

This bit is cleared by hardware when mailbox 0 is requested for transmission or when the software clears the RQCP0 bit.

Bits 3:2 = Reserved. Forced to 0 by hardware.

Bit 1 = **RQCP1** *Request Completed for Mailbox 1*
- Read/Clear
This bit is set by hardware to signal that the last request for mailbox 1 has been completed. The request could be a transmit or an abort request.

This bit is cleared by software.

Bit 0 = **RQCP0** *Request Completed for Mailbox 0*
- Read/Clear
This bit is set by hardware to signal that the last request for mailbox 0 has been completed. The request could be a transmit or an abort request.

This bit is cleared by software.

CAN TRANSMIT PRIORITY REGISTER (CTPR)

All bits of this register are read only.

Reset Value: 0000 1100 (0Ch)

7							0
0	LOW1	LOW0	0	TME1	TME0	0	CODE

Bit 7 = Reserved. Forced to 0 by hardware.

Bit 6 = **LOW1** *Lowest Priority Flag for Mailbox 1*
- Read
This bit is set by hardware when more than one mailbox are pending for transmission and mailbox 1 has the lowest priority.

Bit 5 = **LOW0** *Lowest Priority Flag for Mailbox 0*
- Read
This bit is set by hardware when more than one mailbox are pending for transmission and mailbox 0 has the lowest priority.

Note: These bits are set to zero when only one mailbox is pending.

Bit 4 = Reserved. Forced to 0 by hardware.

Bit 3 = **TME1** *Transmit Mailbox 1 Empty*
- Read
This bit is set by hardware when no transmit request is pending for mailbox 1.

Bit 2 = **TME0** *Transmit Mailbox 0 Empty*
- Read
This bit is set by hardware when no transmit request is pending for mailbox 0.

Bit 1:0 = **CODE** *Mailbox Code*
- Read
In case at least one transmit mailbox is free, the code value is equal to the number of the next transmit mailbox free.

In case all transmit mailboxes are pending, the code value is equal to the number of the transmit mailbox with the lowest priority.

11 INSTRUCTION SET

11.1 CPU ADDRESSING MODES

The CPU features 17 different addressing modes which can be classified in seven main groups:

Addressing Mode	Example
Inherent	nop
Immediate	ld A,#$55
Direct	ld A,$55
Indexed	ld A,($55,X)
Indirect	ld A,([$55],X)
Relative	jrne loop
Bit operation	bset byte,#5

The CPU Instruction set is designed to minimize the number of bytes required per instruction: To do so, most of the addressing modes may be subdivided in two submodes called long and short:

– Long addressing mode is more powerful because it can use the full 64 Kbyte address space, however it uses more bytes and more CPU cycles.

– Short addressing mode is less powerful because it can generally only access page zero (0000h - 00FFh range), but the instruction size is more compact, and faster. All memory to memory instructions use short addressing modes only (CLR, CPL, NEG, BSET, BRES, BTJT, BTJF, INC, DEC, RLC, RRC, SLL, SRL, SRA, SWAP)

The ST7 Assembler optimizes the use of long and short addressing modes.

Table 37. CPU Addressing Mode Overview

Mode			Syntax	Destination	Pointer Address (Hex.)	Pointer Size (Hex.)	Length (Bytes)
Inherent			nop				+ 0
Immediate			ld A,#$55				+ 1
Short	Direct		ld A,$10	00..FF			+ 1
Long	Direct		ld A,$1000	0000..FFFF			+ 2
No Offset	Direct	Indexed	ld A,(X)	00..FF			+ 0
Short	Direct	Indexed	ld A,($10,X)	00..1FE			+ 1
Long	Direct	Indexed	ld A,($1000,X)	0000..FFFF			+ 2
Short	Indirect		ld A,[$10]	00..FF	00..FF	byte	+ 2
Long	Indirect		ld A,[$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed	ld A,([$10],X)	00..1FE	00..FF	byte	+ 2
Long	Indirect	Indexed	ld A,([$10.w],X)	0000..FFFF	00..FF	word	+ 2
Relative	Direct		jrne loop	PC+/-127			+ 1
Relative	Indirect		jrne [$10]	PC+/-127	00..FF	byte	+ 2
Bit	Direct		bset $10,#7	00..FF			+ 1
Bit	Indirect		bset [$10],#7	00..FF	00..FF	byte	+ 2
Bit	Direct	Relative	btjt $10,#7,skip	00..FF			+ 2
Bit	Indirect	Relative	btjt [$10],#7,skip	00..FF	00..FF	byte	+ 3

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).

ST72561 devices are ROM versions. ST72P561 devices are Factory Advanced Service Technique ROM (FASTROM) versions: They are factory-programmed HDFlash devices.

ST72F561 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 7							0	STATIC OPTION BYTE 1 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