STMicroelectronics - ST72F361J6T6 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	LINbusSCI, 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	1.5K x 8
Voltage - Supply (Vcc/Vdd)	3.8V ~ 5.5V
Data Converters	A/D 11x10b
Oscillator Type	External
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/st72f361j6t6

Pin n°			Pin Name	Type	Level		Port						Main function (after reset)	Alternate function	
LQFP64	LQFP44	LQFP32			Input	Output	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	I/O	C_T		**X**	X			X	X	Port C3		
34	24	18	PC4	I/O	C_T		**X**					X$^{2)}$	Port C4		
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	

FLASH PROGRAM MEMORY (Cont'd)

4.5 ICP (IN-CIRCUIT PROGRAMMING)

To perform ICP the microcontroller must be switched to ICC (In-Circuit Communication) mode by an external controller or programming tool.

Depending on the ICP code downloaded in RAM, Flash memory programming can be fully customized (number of bytes to program, program locations, or selection serial communication interface for downloading).

When using an STMicroelectronics or third-party programming tool that supports ICP and the specific microcontroller device, the user needs only to implement the ICP hardware interface on the application board (see Figure 7). For more details on the pin locations, refer to the device pinout description.

4.6 IAP (IN-APPLICATION PROGRAMMING)

This mode uses a BootLoader program previously stored in Sector 0 by the user (in ICP mode or by plugging the device in a programming tool).

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored, etc.). For example, it is possible to download code from the SPI, SCI or other type of serial interface and program it in the Flash. IAP mode can be used to program any of the Flash sectors except Sector 0, which is write/erase protected to allow recovery in case errors occur during the programming operation.

4.7 RELATED DOCUMENTATION

For details on Flash programming and ICC protocol, refer to the *ST7 Flash Programming Reference Manual* and to the *ST7 ICC Protocol Reference Manual*.

4.8 REGISTER DESCRIPTION

FLASH CONTROL/STATUS REGISTER (FCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	0

This register is reserved for use by Programming Tool software. It controls the Flash programming and erasing operations.

Table 5. Flash Control/Status Register Address and Reset Value

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0024h	**FCSR** Reset Value	0	0	0	0	0	0	0	0

CENTRAL PROCESSING UNIT (Cont'd)

Condition Code Register (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	I1	H	I0	N	Z	C

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.

Arithmetic Management Bits

Bit 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.

Bit 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's a copy of the result 7^{th} 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.

Bit 1 = **Z** *Zero*.

This bit is set and cleared by hardware. This bit indicates that the result of the last arithmetic, logical or data manipulation is zero.
0: The result of the last operation is different from zero.
1: The result of the last operation is zero.

This bit is accessed by the JREQ and JRNE test instructions.

Bit 0 = **C** *Carry/borrow.*

This bit is set and cleared by hardware and software. It indicates an overflow or an underflow has occurred during the last arithmetic operation.
0: No overflow or underflow has occurred.
1: An overflow or underflow has occurred.

This bit is driven by the SCF and RCF instructions and tested by the JRC and JRNC instructions. It is also affected by the "bit test and branch", shift and rotate instructions.

Interrupt Management Bits

Bit 5,3 = **I1, I0** *Interrupt*

The combination of the I1 and I0 bits gives the current interrupt software priority.

Interrupt Software Priority	I1	I0
Level 0 (main)	1	0
Level 1	0	1
Level 2	0	0
Level 3 (= interrupt disable)	1	1

These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (IxSPR). They can be also set/cleared by software with the RIM, SIM, IRET, HALT, WFI and PUSH/POP instructions.

See the interrupt management chapter for more details.

INTERRUPTS (Cont'd)

Table 10. Nested Interrupts Register Map and Reset Values

Address (Hex.)	**Register Label**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0025h		ei1		ei0		CLKM		TLI	
	ISPR0 Reset Value	I1_3 1	I0_3 1	I1_2 1	I0_2 1	I1_1 1	I0_1 1	 1	 1
0026h						ei3		ei2	
	ISPR1 Reset Value	I1_7 1	I0_7 1	I1_6 1	I0_6 1	I1_5 1	I0_5 1	I1_4 1	I0_4 1
0027h		LINSCI 2		TIMER 16		TIMER 8		SPI	
	ISPR2 Reset Value	I1_11 1	I0_11 1	I1_10 1	I0_10 1	I1_9 1	I0_9 1	I1_8 1	I0_8 1
0028h						ART		LINSCI 1	
	ISPR3 Reset Value	 1	 1	 1	 1	I1_13 1	I0_13 1	I1_12 1	I0_12 1
0029h	**EICR0** Reset Value	IS31 0	IS30 0	IS21 0	IS20 0	IS11 0	IS10 0	IS01 0	IS00 0
002Ah	**EICR1** Reset Value	 0	 0	 0	 0	 0	 0	TLIS 0	TLIE 0

ON-CHIP PERIPHERALS (Cont'd)

10.3 PWM AUTO-RELOAD TIMER (ART)

10.3.1 Introduction

The Pulse Width Modulated Auto-Reload Timer on-chip peripheral consists of an 8-bit auto reload counter with compare/capture capabilities and of a 7-bit prescaler clock source.

These resources allow five possible operating modes:

– Generation of up to four independent PWM signals

– Output compare and Time base interrupt

– Up to two input capture functions

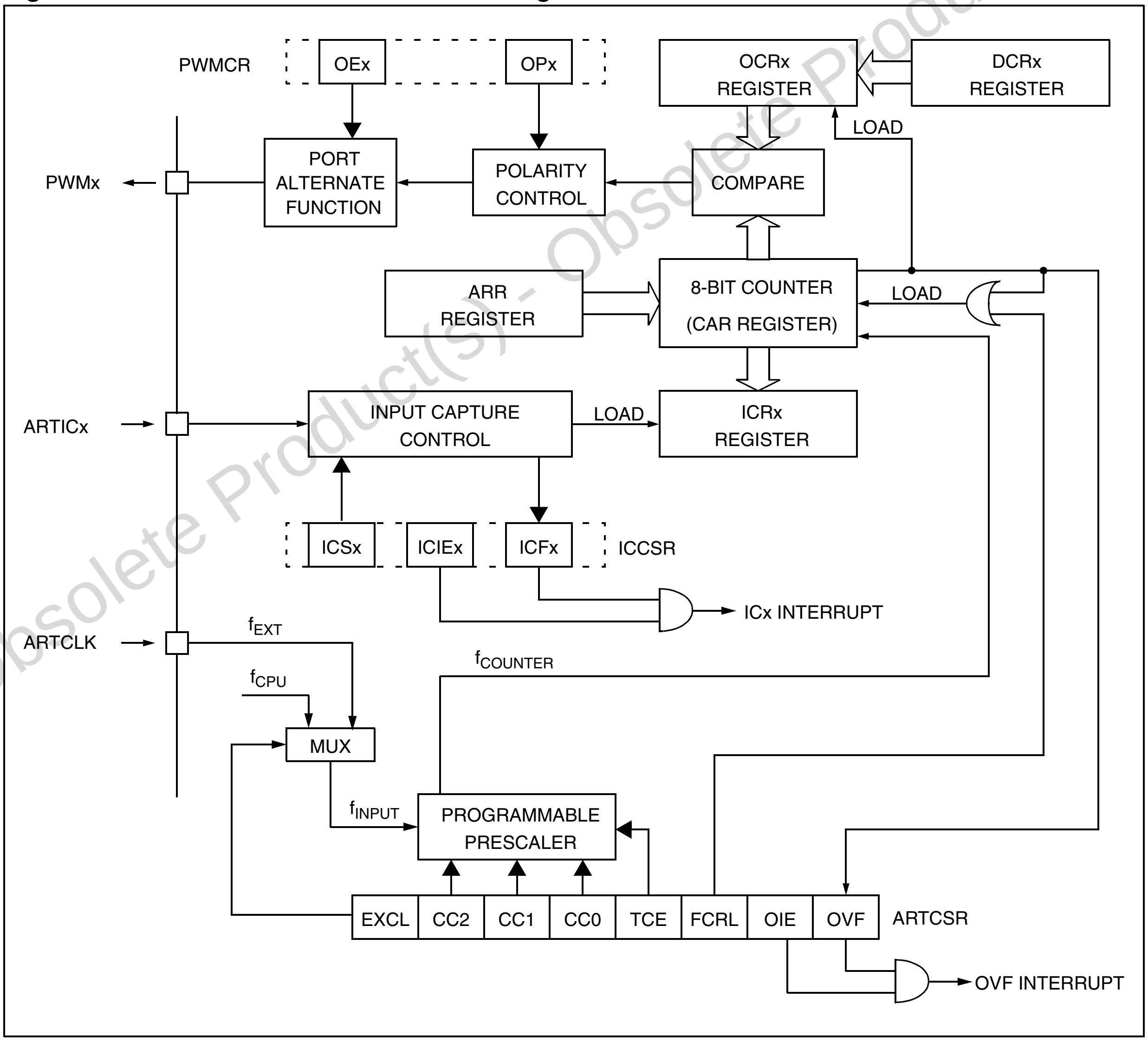
– External event detector

– Up to two external interrupt sources

The three first modes can be used together with a single counter frequency.

The timer can be used to wake up the MCU from WAIT and HALT modes.

Figure 40. PWM Auto-Reload Timer Block Diagram



PWM AUTO-RELOAD TIMER (Cont'd)

10.3.2 Functional Description

Counter

The free running 8-bit counter is fed by the output of the prescaler, and is incremented on every rising edge of the clock signal.

It is possible to read or write the contents of the counter on the fly by reading or writing the Counter Access register (ARTCAR).

When a counter overflow occurs, the counter is automatically reloaded with the contents of the ARTARR register (the prescaler is not affected).

Counter clock and prescaler

The counter clock frequency is given by:

$f_{COUNTER} = f_{INPUT} / 2^{CC[2:0]}$

The timer counter's input clock (f_{INPUT}) feeds the 7-bit programmable prescaler, which selects one of the eight available taps of the prescaler, as defined by CC[2:0] bits in the Control/Status Register (ARTCSR). Thus the division factor of the prescaler can be set to 2^n (where n = 0, 1,..7).

This f_{INPUT} frequency source is selected through the EXCL bit of the ARTCSR register and can be either the f_{CPU} or an external input frequency f_{EXT}.

The clock input to the counter is enabled by the TCE (Timer Counter Enable) bit in the ARTCSR register. When TCE is reset, the counter is stopped and the prescaler and counter contents are frozen. When TCE is set, the counter runs at the rate of the selected clock source.

Counter and Prescaler Initialization

After RESET, the counter and the prescaler are cleared and $f_{INPUT} = f_{CPU}$.

The counter can be initialized by:

– Writing to the ARTARR register and then setting the FCRL (Force Counter Re-Load) and the TCE (Timer Counter Enable) bits in the ARTCSR register.

– Writing to the ARTCAR counter access register,

In both cases the 7-bit prescaler is also cleared, whereupon counting will start from a known value.

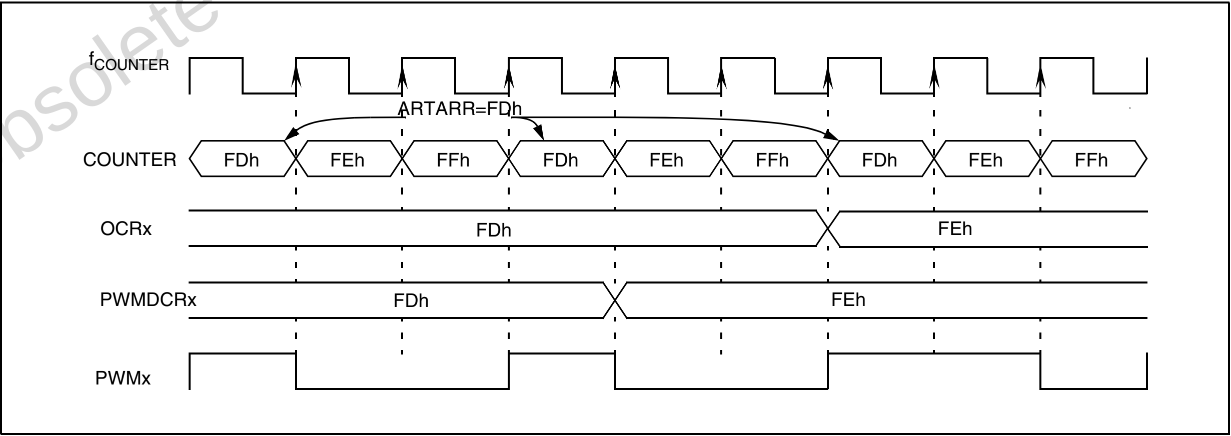
Direct access to the prescaler is not possible.

Output compare control

The timer compare function is based on four different comparisons with the counter (one for each PWMx output). Each comparison is made between the counter value and an output compare register (OCRx) value. This OCRx register can not be accessed directly, it is loaded from the duty cycle register (PWMDCRx) at each overflow of the counter.

This double buffering method avoids glitch generation when changing the duty cycle on the fly.

Figure 41. Output Compare Control



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 ($1/f_{COUNTER}$).

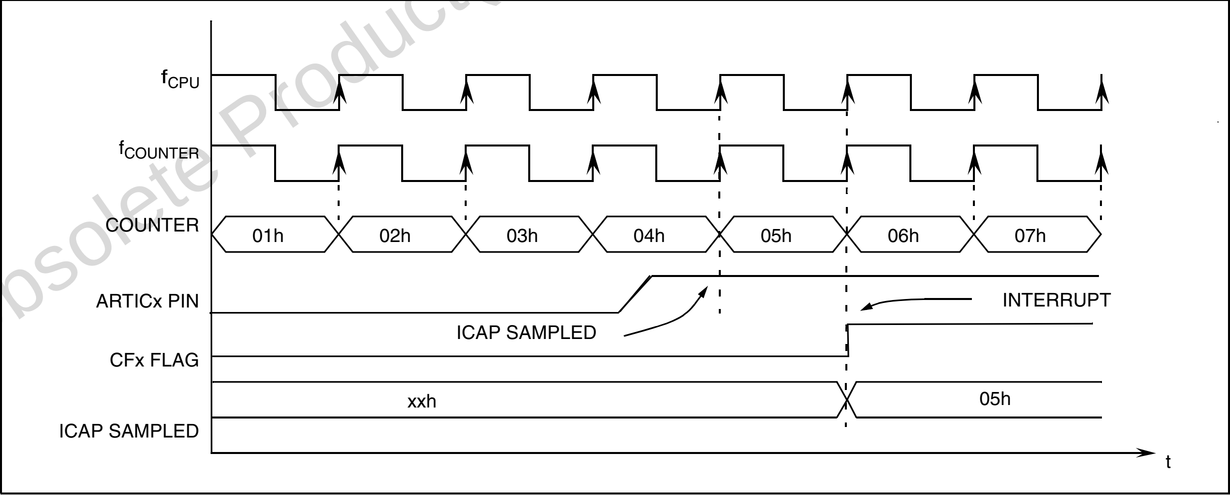
Note: During HALT mode, input capture is inhibited (the ARTICRx is never reloaded) and only the external interrupt capability can be used.

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).

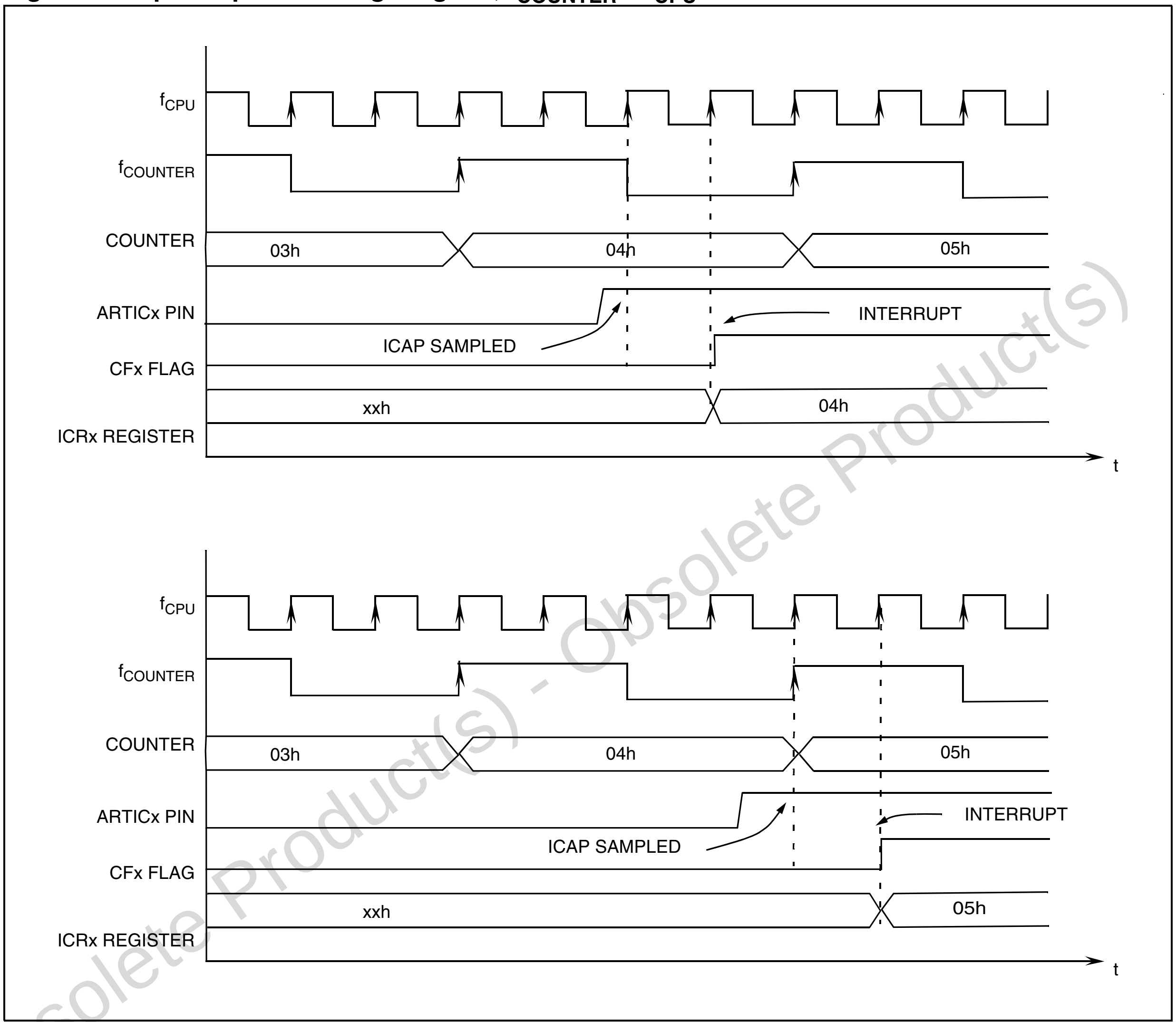
If the counter clock is prescaled, it depends on the position of the ARTICx event within the counter cycle (Figure 46).

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



PWM AUTO-RELOAD TIMER (Cont'd)

Figure 46. input Capture Timing Diagram, $f_{COUNTER} = f_{CPU} / 4$



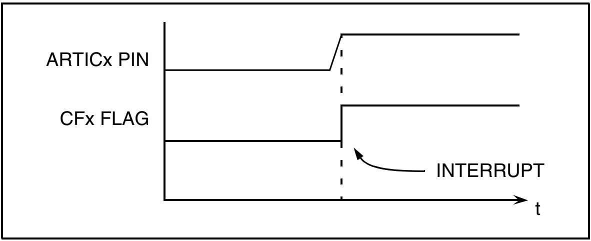
External Interrupt Capability

This mode allows the Input capture capabilities to be used as external interrupt sources. The interrupts are generated on the edge of the ARTICx signal.

The edge sensitivity of the external interrupts is programmable (CSx bit of ARTICCSR register) and they are independently enabled through CIEx bits of the ARTICCSR register. After fetching the interrupt vector, the CFx flags can be read to identify the interrupt source.

During HALT mode, the external interrupts can be used to wake up the micro (if the CIEx bit is set). In this case, the interrupt synchronization is done directly on the ARTICx pin edge (Figure 47).

Figure 47. ART External Interrupt in Halt Mode



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)

INPUT CAPTURE 1 HIGH REGISTER (IC1HR)

Read Only
Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the input capture 1 event).

7							0
MSB							LSB

INPUT CAPTURE 1 LOW REGISTER (IC1LR)

Read Only
Reset Value: Undefined

This is an 8-bit read only register that contains the low part of the counter value (transferred by the input capture 1 event).

7							0
MSB							LSB

OUTPUT COMPARE 1 HIGH REGISTER (OC1HR)

Read/Write
Reset Value: 1000 0000 (80h)

This is an 8-bit register that contains the high part of the value to be compared to the CHR register.

7							0
MSB							LSB

OUTPUT COMPARE 1 LOW REGISTER (OC1LR)

Read/Write
Reset Value: 0000 0000 (00h)

This is an 8-bit register that contains the low part of the value to be compared to the CLR register.

7							0
MSB							LSB

8-BIT TIMER (Cont'd)

Figure 60. Counter Timing Diagram, Internal Clock Divided by 2

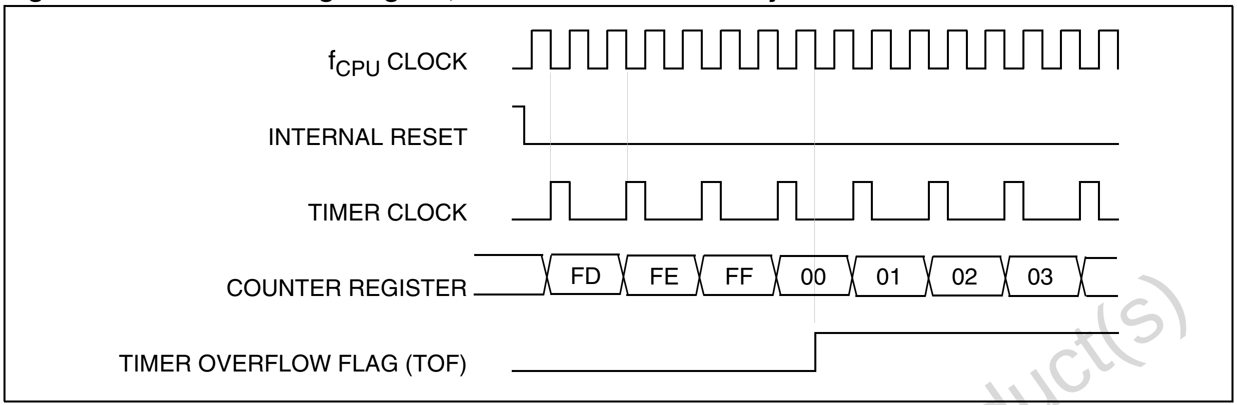


Figure 61. Counter Timing Diagram, Internal Clock Divided by 4

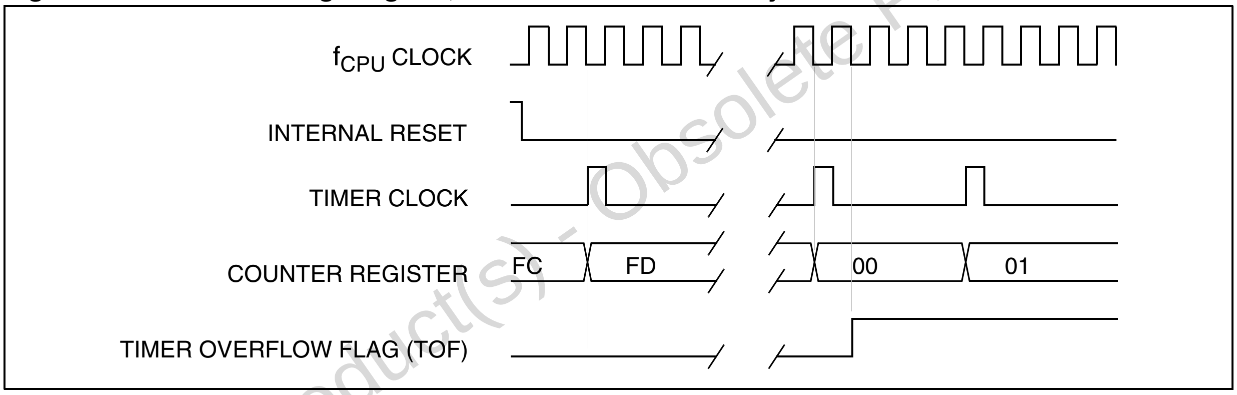
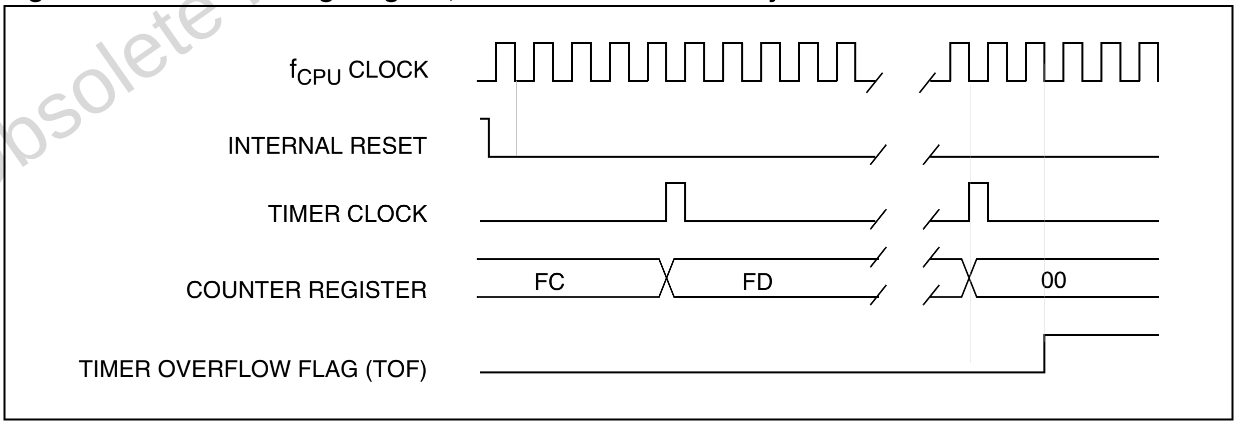


Figure 62. Counter Timing Diagram, Internal Clock Divided by 8



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)

10.5.3.4 One Pulse Mode

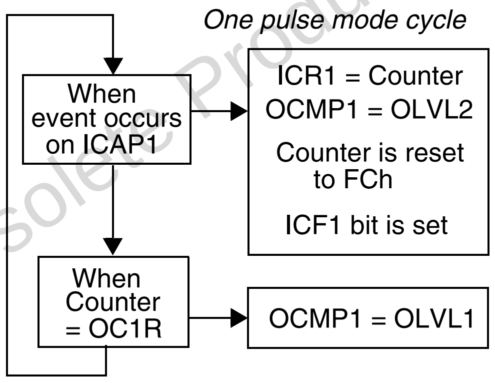
One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

The one pulse mode uses the Input Capture1 function and the Output Compare1 function.

Procedure:

To use one pulse mode:

1. Load the OC1R register with the value corresponding to the length of the pulse (see the formula in the opposite column).
2. Select the following in the CR1 register:
   – Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after the pulse.
   – Using the OLVL2 bit, select the level to be applied to the OCMP1 pin during the pulse.
   – Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1 pin must be configured as floating input).
3. Select the following in the CR2 register:
   – Set the OC1E bit, the OCMP1 pin is then dedicated to the Output Compare 1 function.
   – Set the OPM bit.
   – Select the timer clock CC[1:0] (see Table 19 Clock Control Bits).



Then, on a valid event on the ICAP1 pin, the counter is initialized to FCh and OLVL2 bit is loaded on the OCMP1 pin, the ICF1 bit is set and the value FFFDh is loaded in the IC1R register.

Because the ICF1 bit is set when an active edge occurs, an interrupt can be generated if the ICIE bit is set.

Clearing the Input Capture interrupt request (that is, clearing the ICF*i* bit) is done in two steps:

1. Reading the SR register while the ICF*i* bit is set.
2. An access (read or write) to the IC*i*LR register.

The OC1R register value required for a specific timing application can be calculated using the following formula:

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

Where:

t	= Pulse period (in seconds)
f_{CPU}	= PLL output x2 clock frequency in hertz (or $f_{OSC}/2$ if PLL is not enabled)
PRESC	= Timer prescaler factor (2, 4, 8 or 8000 depending on the CC[1:0] bits, see Table 19 Clock Control Bits)

When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin, (See Figure 68).

Notes:

1. The OCF1 bit cannot be set by hardware in one pulse mode but the OCF2 bit can generate an Output Compare interrupt.
2. When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.
3. If OLVL1=OLVL2 a continuous signal will be seen on the OCMP1 pin.
4. The ICAP1 pin can not be used to perform input capture. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each time a valid edge occurs on the ICAP1 pin and ICF1 can also generates interrupt if ICIE is set.
5. When one pulse mode is used OC1R is dedicated to this mode. Nevertheless OC2R and OCF2 can be used to indicate a period of time has been elapsed but cannot generate an output waveform because the level OLVL2 is dedicated to the one pulse mode.

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).

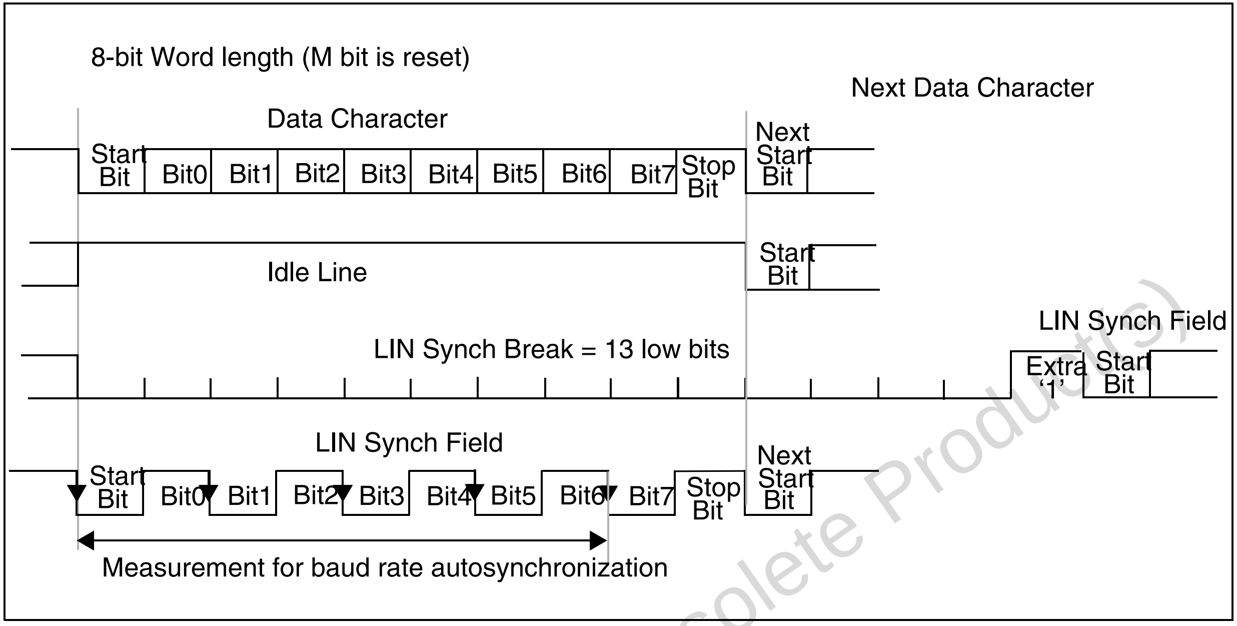
SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 22. SPI Register Map and Reset Values

Address (Hex.)	**Register Label**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
21	**SPIDR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
22	**SPICR** Reset Value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
23	**SPICSR** Reset Value	SPIF 0	WCOL 0	OVR 0	MODF 0	0	SOD 0	SSM 0	SSI 0

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

Figure 80. LIN Characters



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

SCICR2 register is set, the LHDM bit selects the Wake-Up method (replacing the WAKE bit).
0: LIN Synch Break Detection Method
1: LIN Identifier Field Detection Method

Bit 2 = **LHIE** *LIN Header Interrupt Enable*
This bit is set and cleared by software. It is only usable in LIN Slave mode.
0: LIN Header Interrupt is inhibited.
1: An SCI interrupt is generated whenever LHDF = 1.

Bit 1 = **LHDF** *LIN Header Detection Flag*
This bit is set by hardware when a LIN Header is detected and cleared by a software sequence (an access to the SCISR register followed by a read of the SCICR3 register). It is only usable in LIN Slave mode.
0: No LIN Header detected.
1: LIN Header detected.

Notes: The header detection method depends on the LHDM bit:

– If LHDM = 0, a header is detected as a LIN Synch Break.

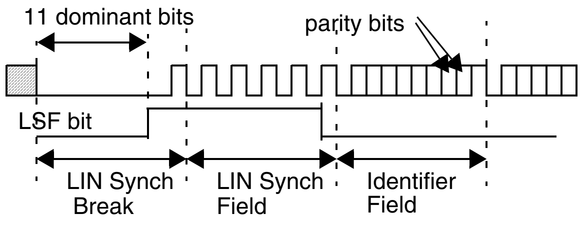
– If LHDM = 1, a header is detected as a LIN Identifier, meaning that a LIN Synch Break Field + a LIN Synch Field + a LIN Identifier Field have been consecutively received.

Bit 0 = **LSF** *LIN Synch Field State*

This bit indicates that the LIN Synch Field is being analyzed. It is only used in LIN Slave mode. In Auto Synchronization Mode (LASE bit = 1), when the SCI is in the LIN Synch Field State it waits or counts the falling edges on the RDI line.

It is set by hardware as soon as a LIN Synch Break is detected and cleared by hardware when the LIN Synch Field analysis is finished (see Figure 11). This bit can also be cleared by software to exit LIN Synch State and return to idle mode.
0: The current character is not the LIN Synch Field
1: LIN Synch Field State (LIN Synch Field undergoing analysis)

Figure 87. LSF Bit Set and Clear



LIN DIVIDER REGISTERS

LDIV is coded using the two registers LPR and LPFR. In LIN Slave mode, the LPR register is accessible at the address of the SCIBRR register and the LPFR register is accessible at the address of the SCIETPR register.

LIN PRESCALER REGISTER (LPR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
LPR7	LPR6	LPR5	LPR4	LPR3	LPR2	LPR1	LPR0

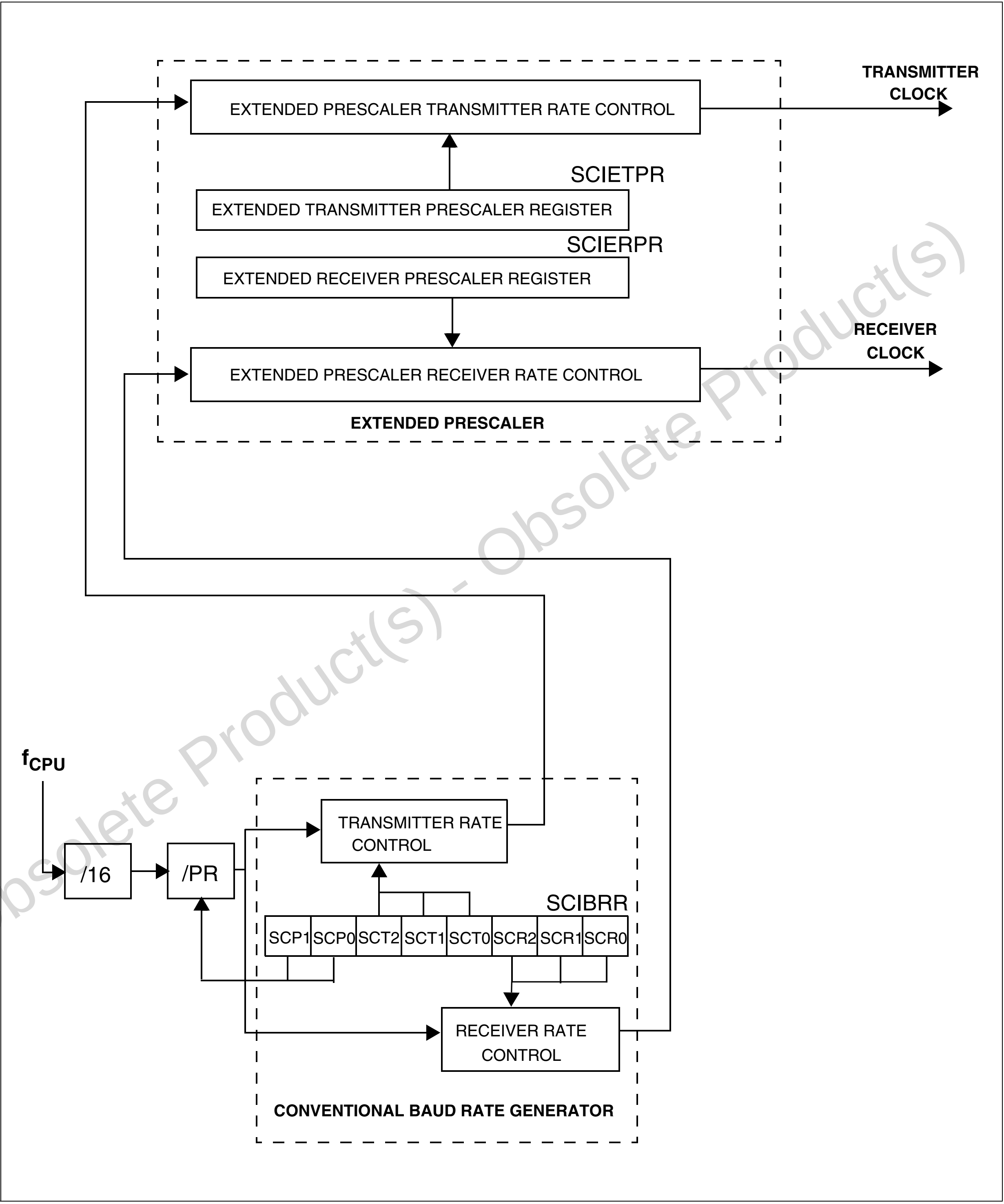
LPR[7:0] *LIN Prescaler (mantissa of LDIV)*
These 8 bits define the value of the mantissa of the LIN Divider (LDIV):

LPR[7:0]	Rounded Mantissa (LDIV)
00h	SCI clock disabled
01h	1
...	...
FEh	254
FFh	255

Caution: LPR and LPFR registers have different meanings when reading or writing to them. Consequently bit manipulation instructions (BRES or BSET) should never be used to modify the LPR[7:0] bits, or the LPFR[3:0] bits.

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

Figure 90. 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
fCPU
/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 Master Only) (Cont'd)

10.8.8 Register Description

STATUS REGISTER (SCISR)

Read Only

Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	PE

Bit 7 = **TDRE** *Transmit data register empty.*
This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE bit = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).
0: Data is not transferred to the shift register
1: Data is transferred to the shift register

Note: Data is not be transferred to the shift register until the TDRE bit is cleared.

Bit 6 = **TC** *Transmission complete.*
This bit is set by hardware when transmission of a frame containing Data is complete. An interrupt is generated if TCIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).
0: Transmission is not complete
1: Transmission is complete

Note: TC is not set after the transmission of a Preamble or a Break.

Bit 5 = **RDRF** *Received data ready flag.*
This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: Data is not received
1: Received data is ready to be read

Bit 4 = **IDLE** *Idle line detect.*
This bit is set by hardware when an Idle Line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Idle Line is detected
1: Idle Line is detected

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

Bit 3 = **OR** *Overrun error.*
This bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register while RDRF = 1. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Overrun error
1: Overrun error is detected

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

Bit 2 = **NF** *Noise flag.*
This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No noise is detected
1: Noise is detected

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

Bit 1 = **FE** *Framing error.*
This bit is set by hardware when a de-synchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Framing error is detected
1: Framing error or break character is detected

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

Bit 0 = **PE** *Parity error.*
This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.
0: No parity error
1: Parity error

12.3.2 Operating Conditions with Low Voltage Detector (LVD)

Subject to general operating conditions for T_A.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(LVD)}$	Reset release threshold (V_{DD} rise)		4.0[1)]	4.2	4.5	V
$V_{IT-(LVD)}$	Reset generation threshold (V_{DD} fall)		3.8	4.0	4.25[1)]	
$V_{hys(LVD)}$	LVD voltage threshold hysteresis[1)]	$V_{IT+(LVD)}$-$V_{IT-(LVD)}$	150	200	250	mV
Vt_{POR}	V_{DD} rise time rate[1)]		6			µs/V
					100	ms/V
$t_{g(VDD)}$	V_{DD} glitches filtered (not detected) by LVD[1)]	Measured at $V_{IT-(LVD)}$			40	ns

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

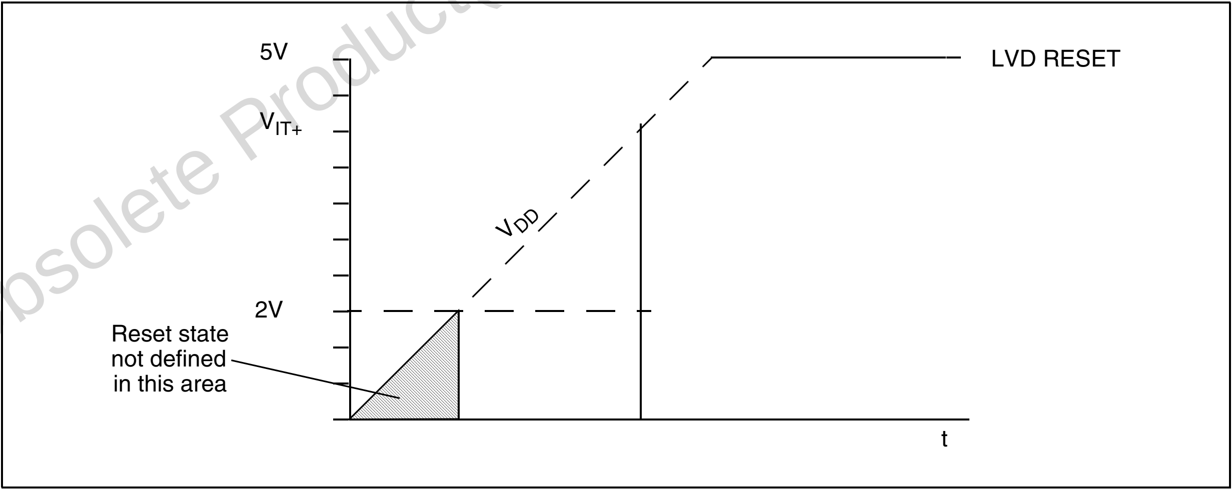
12.3.3 Auxiliary Voltage Detector (AVD) Thresholds

Subject to general operating conditions for T_A.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(AVD)}$	1⇒0 AVDF flag toggle threshold (V_{DD} rise)		4.4[1)]	4.6	4.9	V
$V_{IT-(AVD)}$	0⇒1 AVDF flag toggle threshold (V_{DD} fall)		4.2	4.4	4.65[1)]	
$V_{hys(AVD)}$	AVD voltage threshold hysteresis	$V_{IT+(AVD)}$-$V_{IT-(AVD)}$		250		mV
ΔV_{IT-}	Voltage drop between AVD flag set and LVD reset activated	$V_{IT-(AVD)}$-$V_{IT-(LVD)}$		450		

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

Figure 98. LVD Startup Behavior



Note: When the LVD is enabled, the MCU reaches its authorized operating voltage from a reset state. However, in some devices, the reset signal may be undefined until V_{DD} is approximately 2V. As a consequence, the I/Os may toggle when V_{DD} is below this voltage.

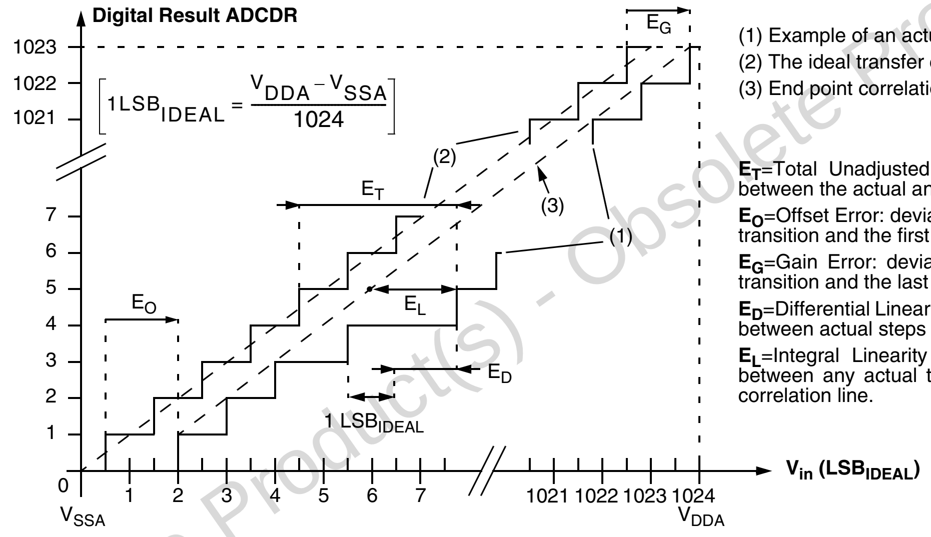
Because Flash write access is impossible below this voltage, the Flash memory contents will not be corrupted.

ADC CHARACTERISTICS (Cont'd)

ADC Accuracy with f_{CPU} = 8 MHz, f_{ADC} = 4 MHz R_{AIN} < 10kΩ, V_{DD} = 5V

Symbol	Parameter	Conditions	Typ	Max	Unit
$\lvert E_T \rvert$	Total unadjusted error[1)]		3.2	5	LSB
$\lvert E_O \rvert$	Offset error[1)]		1	4	
$\lvert E_G \rvert$	Gain Error[1)]		0.7		
$\lvert E_D \rvert$	Differential linearity error[1)]		1.5	2.3	
$\lvert E_L \rvert$	Integral linearity error[1)]		1.2	3.6	

Figure 123. ADC Accuracy Characteristics



(1) Example of an actual transfer curve
(2) The ideal transfer curve
(3) End point correlation line

E_T=Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.

E_O=Offset Error: deviation between the first actual transition and the first ideal one.

E_G=Gain Error: deviation between the last ideal transition and the last actual one.

E_D=Differential Linearity Error: maximum deviation between actual steps and the ideal one.

E_L=Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

Notes:

1. Data based on characterization results, not tested in production. ADC Accuracy vs. Negative Injection Current: Injecting negative current on any of the standard (non-robust) analog input pins should be avoided as this significantly reduces the accuracy of the conversion being performed on another analog input. It is recommended to add a Schottky diode (pin to ground) to standard analog pins which may potentially inject negative current. The effect of negative injection current on robust pins is specified in Section 12.9.

Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in Section 12.9 does not affect the ADC accuracy.