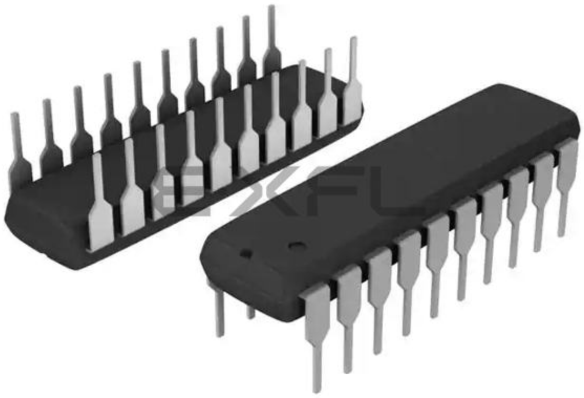
STMicroelectronics - ST7FLITE30F2B6 Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	16MHz
Connectivity	LINbusSCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	15
Program Memory Size	8KB (8K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	384 x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 5.5V
Data Converters	A/D 7x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	20-DIP (0.300", 7.62mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7flite30f2b6

Table of Contents

Address	Block	Register Label	Register Name	Reset Status	Remarks
0031h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
0032h		SPICR	SPI Control Register	0xh	R/W
0033h		SPICSR	SPI Control Status Register	00h	R/W
0034h	ADC	ADCCSR	A/D Control Status Register	00h	R/W
0035h		ADCDRH	A/D Data Register High	xxh	Read Only
0036h		ADCDRL	A/D control and Data Register Low	x0h	R/W
0037h	ITC	EICR	External Interrupt Control Register	00h	R/W
0038h	MCC	MCCSR	Main Clock Control/Status Register	00h	R/W
0039h	Clock and Reset	RCCR	RC oscillator Control Register	FFh	R/W
003Ah		SICSR	System Integrity Control/Status Register	0110 0xx0b	R/W
003Bh	Reserved area (1 byte)				
003Ch	ITC	EISR	External Interrupt Selection Register	00h	R/W
003Dh to 003Fh	Reserved area (3 bytes)				
0040h	LINSCI (LIN Master/Slave)	SCISR	SCI Status Register	C0h	Read Only
0041h		SCIDR	SCI Data Register	xxh	R/W
0042h		SCIBRR	SCI Baud Rate Register	00xx xxxxb	R/W
0043h		SCICR1	SCI Control Register 1	xxh	R/W
0044h		SCICR2	SCI Control Register 2	00h	R/W
0045h		SCICR3	SCI Control Register 3	00h	R/W
0046h		SCIERPR	SCI Extended Receive Prescaler Register	00h	R/W
0047h		SCIETPR	SCI Extended Transmit Prescaler Register	00h	R/W
0048h	Reserved area (1 byte)				
0049h	AWU	AWUPR	AWU Prescaler Register	FFh	R/W
004Ah		AWUCSR	AWU Control/Status Register	00h	R/W
004Bh	DM[3]	DMCR	DM Control Register	00h	R/W
004Ch		DMSR	DM Status Register	00h	R/W
004Dh		DMBK1H	DM Breakpoint Register 1 High	00h	R/W
004Eh		DMBK1L	DM Breakpoint Register 1 Low	00h	R/W
004Fh		DMBK2H	DM Breakpoint Register 2 High	00h	R/W
0050h		DMBK2L	DM Breakpoint Register 2 Low	00h	R/W
0051h to 007Fh	Reserved area (47 bytes)				

Legend: x=undefined, R/W=read/write

Notes:

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents.
2. The bits associated with unavailable pins must always keep their reset value.
3. For a description of the DM registers, see the ST7 ICC Reference Manual.

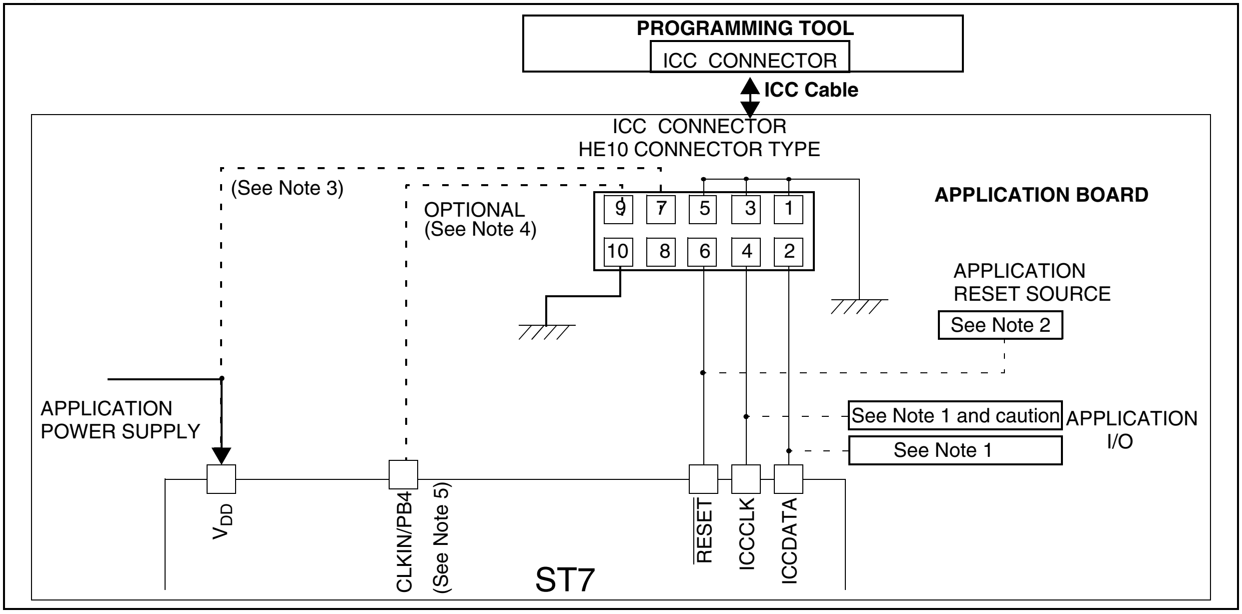
FLASH PROGRAM MEMORY (Cont'd)

4.4 ICC INTERFACE

ICP needs a minimum of 4 and up to 6 pins to be connected to the programming tool. These pins are:

- $\overline{\text{RESET}}$: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- CLKIN/PB4: main clock input for external source
- V_{DD}: application board power supply (optional, see Note 3)

Figure 5. Typical ICC Interface



Notes:

1. If the ICCCLK or ICCDATA pins are only used as outputs in the application, no signal isolation is necessary. As soon as the Programming Tool is plugged to the board, even if an ICC session is not in progress, the ICCCLK and ICCDATA pins are not available for the application. If they are used as inputs by the application, isolation such as a serial resistor has to be implemented if another device forces the signal. Refer to the Programming Tool documentation for recommended resistor values.

2. During the ICP session, the programming tool must control the $\overline{\text{RESET}}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor<1K). A schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with R>1K or a reset management IC with open drain output and pull-up resistor>1K, no additional components are needed. In all cases the user must ensure that no external reset is generated by the application during the ICC session.

3. The use of Pin 7 of the ICC connector depends on the Programming Tool architecture. This pin must be connected when using most ST Programming Tools (it is used to monitor the application power supply). Please refer to the Programming Tool manual.

4. Pin 9 must be connected to the PB4 pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte. ST7 devices with multi-oscillator capability must have OSC2 grounded in this case.

5. With any programming tool, while the ICP option is disabled, the external clock must be provided on PB4.

6. In 38-pulse ICC mode, the internal RC oscillator is forced as a clock source, regardless of the selection in the option byte. For ST7LITE30 devices which do not support the internal RC oscillator, the "option byte disabled" mode must be used (35-pulse ICC mode entry, clock provided by the tool).

Caution: During normal operation ICCCLK pin must be pulled- up, internally or externally (external pull-up of 10k mandatory in noisy environment). This avoids entering ICC mode unexpectedly during a reset. In the application, even if the pin is configured as output, any reset puts it back in input pull-up.

CPU REGISTERS (cont'd)

CONDITION CODE REGISTER (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	1	H	I	N	Z	C

The 8-bit Condition Code register contains the interrupt mask 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.

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 instruction. 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 3 = **I** *Interrupt mask*

This bit is set by hardware when entering in interrupt or by software to disable all interrupts except the TRAP software interrupt. This bit is cleared by software.
0: Interrupts are enabled.
1: Interrupts are disabled.

This bit is controlled by the RIM, SIM and IRET instructions and is tested by the JRM and JRNM instructions.

Note: Interrupts requested while I is set are latched and can be processed when I is cleared. By default an interrupt routine is not interruptible because the I bit is set by hardware at the start of the routine and reset by the IRET instruction at the end of the routine. If the I bit is cleared by software in the interrupt routine, pending interrupts are serviced regardless of the priority level of the current interrupt routine.

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 is a copy of the 7th bit of the result.
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.

CPU REGISTERS (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 01FFh

15							8
0	0	0	0	0	0	0	1

7							0
1	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see Figure 11).

Since the stack is 128 bytes deep, the 9 most significant bits are forced by hardware. Following an

7 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.

Main features

- Clock Management
  - 1 MHz internal RC oscillator (enabled by option byte, available on ST7LITE35 and ST7LITE39 devices only)
  - 1 to 16 MHz or 32kHz External crystal/ceramic resonator (selected by option byte)
  - External Clock Input (enabled by option byte)
  - PLL for multiplying the frequency by 8 or 4 (enabled by option byte)
- Reset Sequence Manager (RSM)
- System Integrity Management (SI)
  - Main supply Low voltage detection (LVD) with reset generation (enabled by option byte)
  - Auxiliary Voltage detector (AVD) with interrupt capability for monitoring the main supply (enabled by option byte)

7.1 INTERNAL RC OSCILLATOR ADJUSTMENT

The device contains an internal RC oscillator with an accuracy of 1% for a given device, temperature and voltage range (4.5V-5.5V). It must be calibrated to obtain the frequency required in the application. This is done by software writing a 8-bit calibration value in the RCCR (RC Control Register) and in the bits [6:5] in the SICSR (SI Control Status Register).

Whenever the microcontroller is reset, the RCCR returns to its default value (FFh), i.e. each time the device is reset, the calibration value must be loaded in the RCCR. Predefined calibration values are stored in EEPROM for 3V and 5V V_{DD} supply voltages at 25°C, as shown in the following table.

RCCR	Conditions	ST7LITE3 Addresses
RCCRH0	V_{DD}=5V T_A=25°C f_{RC}=1MHz	DEE0h [1] (CR[9:2] bits)
RCCRL0		DEE1h [1] (CR[1:0] bits)
RCCRH1	V_{DD}=3.3V T_A=25°C f_{RC}=1MHz	DEE2h [1] (CR[9:2] bits)
RCCRL1		DEE3h [1] (CR[1:0] bits)

[1] DEE0h, DEE1h, DEE2h and DEE3h addresses are located in a reserved area of non-volatile memory. They are read-only bytes for the application code. This area cannot be erased or programmed by any ICC operation.

For compatibility reasons with the SICSR register, CR[1:0] bits are stored in the 5th and 6th position of DEE1 and DEE3 addresses.

Note:

– In 38-pulse ICC mode, the internal RC oscillator is forced as a clock source, regardless of the selection in the option byte. For ST7LITE30 devices which do not support the internal RC oscillator, the “option byte disabled” mode must be used (35-pulse ICC mode entry, clock provided by the tool).

– See “ELECTRICAL CHARACTERISTICS” on page 131. for more information on the frequency and accuracy of the RC oscillator.

– To improve clock stability and frequency accuracy, it is recommended to place a decoupling capacitor, typically 100nF, between the V_{DD} and V_{SS} pins as close as possible to the ST7 device

– These bytes are systematically programmed by ST, including on FASTROM devices. Consequently, customers intending to use FASTROM service must not use these bytes.

– RCCR0 and RCCR1 calibration values will not be erased if the read-out protection bit is reset after it has been set . See “Read out Protection” on page 14.

Caution: If the voltage or temperature conditions change in the application, the frequency may need to be recalibrated.

Refer to application note AN1324 for information on how to calibrate the RC frequency using an external reference signal.

7.2 PHASE LOCKED LOOP

The PLL can be used to multiply a 1MHz frequency from the RC oscillator or the external clock by 4 or 8 to obtain f_{OSC} of 4 or 8 MHz. The PLL is enabled and the multiplication factor of 4 or 8 is selected by 2 option bits.

– The x4 PLL is intended for operation with V_{DD} in the 2.7V to 3.3V range

– The x8 PLL is intended for operation with V_{DD} in the 3.3V to 5.5V range

Refer to Section 15.1 for the option byte description.

If the PLL is disabled and the RC oscillator is enabled, then $f_{OSC =}$ 1MHz.

7.5 RESET SEQUENCE MANAGER (RSM)

7.5.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 15:

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

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to section 12.2.1 on page 128 for further details.

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 3 phases as shown in Figure 14:

- Active Phase depending on the RESET source
- 256 or 4096 CPU clock cycle delay (see table below)
- RESET vector fetch

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

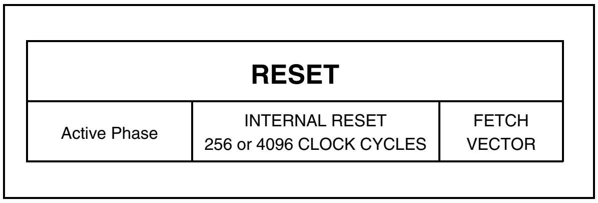
The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay is automatically selected depending on the clock source chosen by option byte:

The RESET vector fetch phase duration is 2 clock cycles.

Clock Source	CPU clock cycle delay
Internal RC Oscillator	256
External clock (connected to CLKIN pin)	256
External Crystal/Ceramic Oscillator (connected to OSC1/OSC2 pins)	4096

If the PLL is enabled by option byte, it outputs the clock after an additional delay of $t_{STARTUP}$ (see Figure 12).

Figure 14. RESET Sequence Phases



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

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device. See 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 16). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

7.6 SYSTEM INTEGRITY MANAGEMENT (SI)

The System Integrity Management block contains the Low voltage Detector (LVD) and Auxiliary Voltage Detector (AVD) functions. It is managed by the SICSR register.

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to section 12.2.1 on page 128 for further details.

7.6.1 Low Voltage Detector (LVD)

The Low Voltage Detector function (LVD) generates a static reset when the V_{DD} supply voltage is below a $V_{IT-(LVD)}$ reference value. This means that it secures the power-up as well as the power-down keeping the ST7 in reset.

The $V_{IT-(LVD)}$ reference value for a voltage drop is lower than the $V_{IT+(LVD)}$ reference value for power-on in order to avoid a parasitic reset when the MCU starts running and sinks current on the supply (hysteresis).

The LVD Reset circuitry generates a reset when V_{DD} is below:

– $V_{IT+(LVD)}$when V_{DD} is rising

– $V_{IT-(LVD)}$ when V_{DD} is falling

The LVD function is illustrated in Figure 17.

The voltage threshold can be configured by option byte to be low, medium or high.

Provided the minimum V_{DD} value (guaranteed for the oscillator frequency) is above $V_{IT-(LVD)}$, the MCU can only be in two modes:

– under full software control

– in static safe reset

In these conditions, secure operation is always ensured for the application without the need for external reset hardware.

During a Low Voltage Detector Reset, the $\overline{\text{RESET}}$ pin is held low, thus permitting the MCU to reset other devices.

Notes:

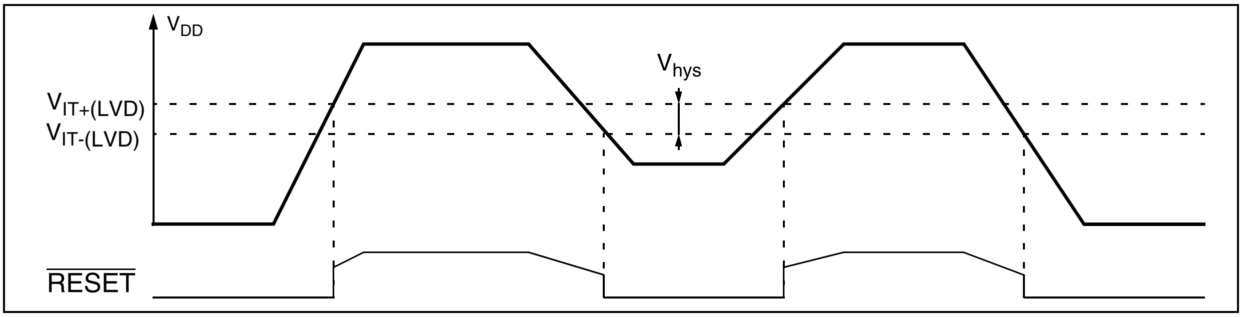
The LVD allows the device to be used without any external RESET circuitry.

Use of LVD with capacitive power supply: with this type of power supply, if power cuts occur in the application, it is recommended to pull V_{DD} down to 0V to ensure optimum restart conditions. Refer to circuit example in Figure 99 on page 154 and note 4.

The LVD is an optional function which can be selected by option byte.

It is recommended to make sure that the V_{DD} supply voltage rises monotonously when the device is exiting from Reset, to ensure the application functions properly.

Figure 17. Low Voltage Detector vs Reset



SYSTEM INTEGRITY MANAGEMENT (Cont'd)

7.6.3 Low Power Modes

Mode	Description
WAIT	No effect on SI. AVD interrupts cause the device to exit from Wait mode.
HALT	The SICSR register is frozen. The AVD becomes inactive and the AVD interrupt cannot be used to exit from Halt mode.

7.6.3.1 Interrupts

The AVD interrupt event generates an interrupt if the corresponding Enable Control Bit (AVDIE) is set and the interrupt mask in the CC register is reset (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
AVD event	AVDF	AVDIE	Yes	No

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

7.6.4 Register Description

SYSTEM INTEGRITY (SI) CONTROL/STATUS REGISTER (SICSR)

Read/Write

Reset Value: 0110 0xx0 (6xh)

7							0
0	CR1	CR0	WDG RF	LOCKED	LVDRF	AVDF	AVDIE

Bit 7 = Reserved, must be kept cleared.

Bits 6:5 = **CR[1:0]** *RC Oscillator Frequency Adjustment bits*

These bits, as well as CR[9:2] bits in the RCCR register must be written immediately after reset to adjust the RC oscillator frequency and to obtain an accuracy of 1%. Refer to section 7.3 on page 24

Bit 4 = **WDGRF** *Watchdog reset flag*
This bit indicates that the last Reset was generated by the Watchdog peripheral. It is set by hardware (watchdog reset) and cleared by software (by reading SICSR register) or an LVD Reset (to ensure a stable cleared state of the WDGRF flag when CPU starts).
Combined with the LVDRF flag information, the flag description is given by the following table.

RESET Sources	LVDRF	WDGRF
External $\overline{\text{RESET}}$ pin	0	0
Watchdog	0	1
LVD	1	X

Bit 3 = **LOCKED** *PLL Locked Flag*
This bit is set by hardware. It is cleared only by a power-on reset. It is set automatically when the PLL reaches its operating frequency.
0: PLL not locked
1: PLL locked

Bit 2 = **LVDRF** *LVD reset flag*
This bit indicates that the last Reset was generated by the LVD block. It is set by hardware (LVD reset) and cleared by software (by reading). When the LVD is disabled by OPTION BYTE, the LVDRF bit value is undefined.

Bit 1 = **AVDF** *Voltage Detector flag*
This read-only bit is set and cleared by hardware. If the AVDIE bit is set, an interrupt request is generated when the AVDF bit is set. Refer to Figure 19 and to Section 7.6.2.1 for additional details.
0: V_{DD} over AVD threshold
1: V_{DD} under AVD threshold

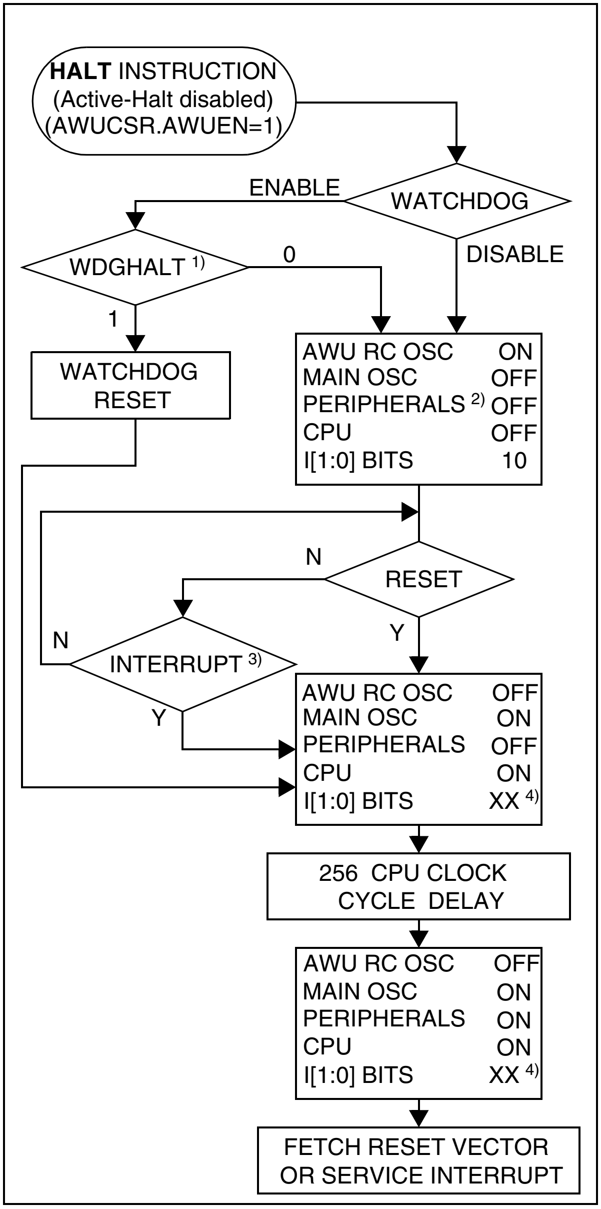
Bit 0 = **AVDIE** *Voltage Detector interrupt enable*
This bit is set and cleared by software. It enables an interrupt to be generated when the AVDF flag is set. The pending interrupt information is automatically cleared when software enters the AVD interrupt routine.
0: AVD interrupt disabled
1: AVD interrupt enabled

Application notes

The LVDRF flag is not cleared when another RESET type occurs (external or watchdog), the LVDRF flag remains set to keep trace of the original failure.
In this case, a watchdog reset can be detected by software while an external reset can not.

POWER SAVING MODES (Cont'd)

Figure 30. AWUFH Mode Flow-chart



Notes:

1. WDGHALT is an option bit. See option byte section for more details.

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

3. Only an AWUFH interrupt and some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to Table 6, "Interrupt Mapping," on page 36 for more details.

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

11 ON-CHIP PERIPHERALS

11.1 WATCHDOG TIMER (WDG)

11.1.1 Introduction

The Watchdog timer 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 counter's contents before the T6 bit becomes cleared.

11.1.2 Main Features

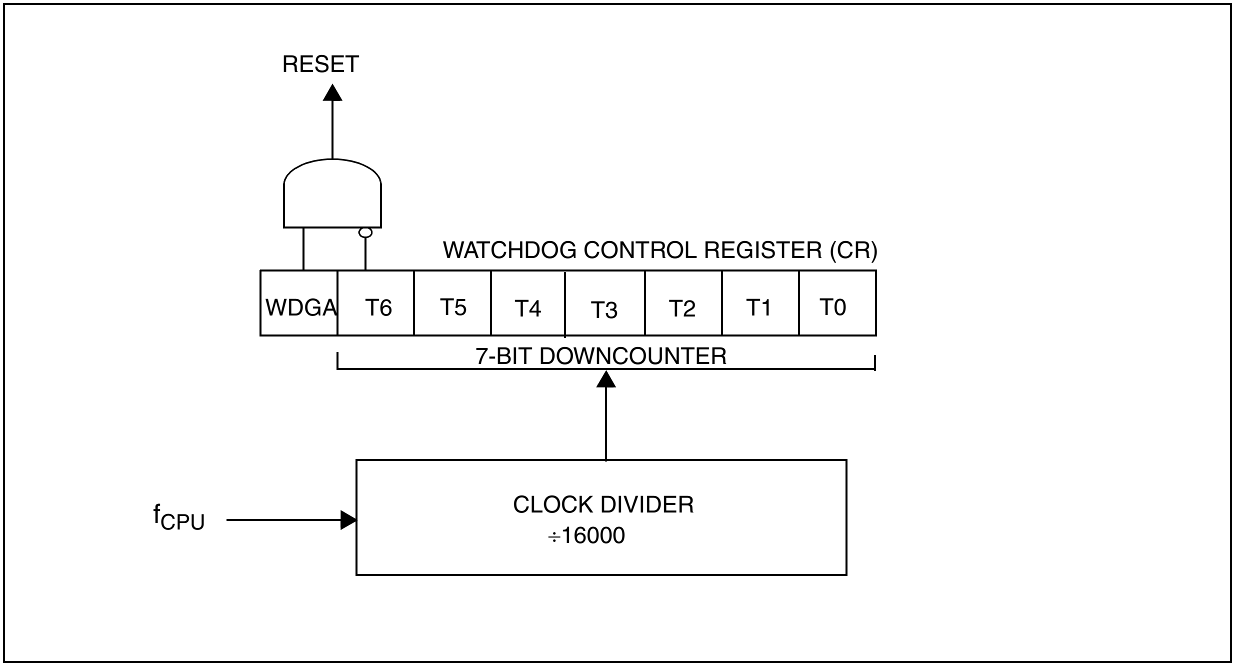
- Programmable free-running downcounter (64 increments of 16000 CPU cycles)
- Programmable reset
- Reset (if watchdog activated) when the T6 bit reaches zero
- Optional reset on HALT instruction (configurable by option byte)
- Hardware Watchdog selectable by option byte

11.1.3 Functional Description

The counter value stored in the CR register (bits T[6:0]), is decremented every 16000 machine cycles, 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 timer (bits T[6:0]) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 30µs.

Figure 33. Watchdog Block Diagram



DUAL 12-BIT AUTORELOAD TIMER 3 (Cont'd)

11.2.3 Functional Description

11.2.3.1 PWM Mode

This mode allows up to four Pulse Width Modulated signals to be generated on the PWMx output pins.

PWM Frequency

The four PWM signals can have the same frequency (f_{PWM}) or can have two different frequencies. This is selected by the ENCNTR2 bit which enables single timer or dual timer mode (see Figure 34 and Figure 35).

The frequency is controlled by the counter period and the ATR register value. In dual timer mode, PWM2 and PWM3 can be generated with a different frequency controlled by CNTR2 and ATR2.

$$f_{PWM} = f_{COUNTER} / (4096 - ATR)$$

Following the above formula,

– If $f_{COUNTER}$ is 4 Mhz, the maximum value of f_{PWM} is 2 MHz (ATR register value = 4094),the minimum value is 1 KHz (ATR register value = 0).

Duty Cycle

The duty cycle is selected by programming the DCRx registers. These are preload registers. The DCRx values are transferred in Active duty cycle registers after an overflow event if the corresponding transfer bit (TRANx bit) is set.

The TRAN1 bit controls the PWMx outputs driven by counter 1 and the TRAN2 bit controls the PWMx outputs driven by counter 2.

PWM generation and output compare are done by comparing these active DCRx values with the counter.

The maximum available resolution for the PWMx duty cycle is:

$$Resolution = 1 / (4096 - ATR)$$

where ATR is equal to 0. With this maximum resolution, 0% and 100% duty cycle can be obtained by changing the polarity.

At reset, the counter starts counting from 0.

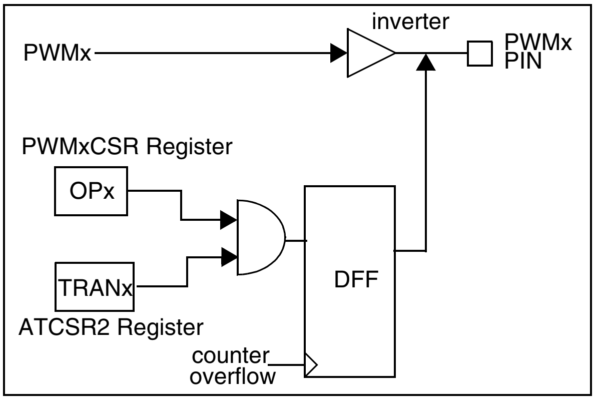
When a upcounter overflow occurs (OVF event), the preloaded Duty cycle values are transferred to the active Duty Cycle registers and the PWMx signals are set to a high level. When the upcounter matches the active DCRx value the PWMx signals are set to a low level. To obtain a signal on a PWMx pin, the contents of the corresponding active DCRx register must be greater than the contents of the ATR register.

The maximum value of ATR is 4094 because it must be lower than the DCR value which must be 4095 in this case.

Polarity Inversion

The polarity bits can be used to invert any of the four output signals. The inversion is synchronized with the counter overflow if the corresponding transfer bit in the ATCSR2 register is set (reset value). See Figure 36.

Figure 36. PWM Polarity Inversion



The Data Flip Flop (DFF) applies the polarity inversion when triggered by the counter overflow input.

Output Control

The PWMx output signals can be enabled or disabled using the OEx bits in the PWMCR register.

DUAL 12-BIT AUTORELOAD TIMER 3 (Cont'd)

Bit 4 = **OVFIE2** *Overflow interrupt 2 enable*
This bit is read/write by software and controls the overflow interrupt of counter2.
0: Overflow interrupt disabled.
1: Overflow interrupt enabled.

Bit 3 = **OVF2** *Overflow Flag.*
This bit is set by hardware and cleared by software by reading the ATCSR2 register. It indicates the transition of the counter2 from FFFh to ATR2 value.
0: No counter overflow occurred
1: Counter overflow occurred

Bit 2 = **ENCNTR2** *Enable counter2*

This bit is read/write be software and switches the second counter CNTR2. If this bit is set, PWM2 and PWM3 will be generated using CNTR2.
0: CNTR2 stopped.
1: CNTR2 starts running.

Bit 1= **TRAN2** *Transfer enable2*
This bit is read/write by software, cleared by hardware after each completed transfer and set by hardware after reset. It controls the transfers on CNTR2.

It allows the value of the Preload DCRx registers to be transferred to the Active DCRx registers after the next overflow event.

The OPx bits are transferred to the shadow OPx bits in the same way.

(*Only DCR2/DCR3 can be controlled with this bit*)

Bit 0 = **TRAN1** *Transfer enable* 1
This bit is read/write by software, cleared by hardware after each completed transfer and set by hardware after reset. It controls the transfers on CNTR1. It allows the value of the Preload DCRx registers to be transferred to the Active DCRx registers after the next overflow event.

The OPx bits are transferred to the shadow OPx bits in the same way.

AUTORELOAD REGISTER2 (ATR2H)
Read / Write
Reset Value: 0000 0000 (00h)

15							8
0	0	0	0	ATR11	ATR10	ATR9	ATR8

AUTORELOAD REGISTER (ATR2L)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
ATR7	ATR6	ATR5	ATR4	ATR3	ATR2	ATR1	ATR0

Bits 11:0 = **ATR2[11:0]** *Autoreload Register 2.*
This is a 12-bit register which is written by software. The ATR2 register value is automatically loaded into the upcounter CNTR2 when an overflow of CNTR2 occurs. The register value is used to set the PWM2/PWM3 frequency when ENCNTR2 is set.

DEAD TIME GENERATOR REGISTER (DTGR)

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

7							0
DTE	DT6	DT5	DT4	DT3	DT2	DT1	DT0

Bits 7 = **DTE** *Dead Time Enable*
This bit is read/write by software. It enables a dead time generation on PWM0/PWM1.
0: No Dead time insertion.
1: Dead time insertion enabled.

Bit 6:0 = **DT[6:0]** *Dead Time Value*

These bits are read/write by software. They define the dead time inserted between PWM0/PWM1. Dead time is calculated as follows:

Dead Time = DT[6:0] x Tcounter1

SERIAL PERIPHERAL INTERFACE (cont'd)

11.4.5 Error Flags

11.4.5.1 Master Mode Fault (MODF)

Master mode fault occurs when the master device's $\overline{\text{SS}}$ pin is pulled low.

When a Master mode fault occurs:

- The MODF bit is set and an SPI interrupt request is generated if the SPIE bit is set.
- The SPE bit is reset. This blocks all output from the device and disables the SPI peripheral.
- The MSTR bit is reset, thus forcing the device into slave mode.

Clearing the MODF bit is done through a software sequence:

1. A read access to the SPICSR register while the MODF bit is set.
2. A write to the SPICR register.

Notes: To avoid any conflicts in an application with multiple slaves, the $\overline{\text{SS}}$ pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits may be restored to their original state during or after this clearing sequence.

Hardware does not allow the user to set the SPE and MSTR bits while the MODF bit is set except in the MODF bit clearing sequence.

In a slave device, the MODF bit can not be set, but in a multimaster configuration the device can be in slave mode with the MODF bit set.

The MODF bit indicates that there might have been a multimaster conflict and allows software to handle this using an interrupt routine and either perform a reset or return to an application default state.

11.4.5.2 Overrun Condition (OVR)

An overrun condition occurs when the master device has sent a data byte and the slave device has not cleared the SPIF bit issued from the previously transmitted byte.

When an Overrun occurs:

- The OVR bit is set and an interrupt request is generated if the SPIE bit is set.

In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read to the SPIDR register returns this byte. All other bytes are lost.

The OVR bit is cleared by reading the SPICSR register.

11.4.5.3 Write Collision Error (WCOL)

A write collision occurs when the software tries to write to the SPIDR register while a data transfer is taking place with an external device. When this happens, the transfer continues uninterrupted and the software write will be unsuccessful.

Write collisions can occur both in master and slave mode. See also Section 11.4.3.2 Slave Select Management.

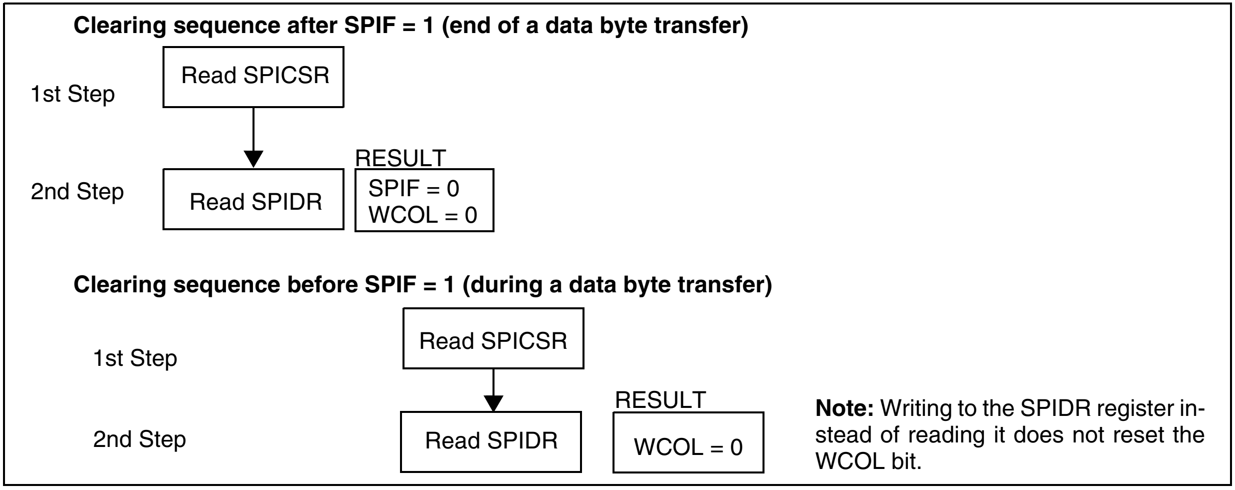
Note: A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

The WCOL bit in the SPICSR register is set if a write collision occurs.

No SPI interrupt is generated when the WCOL bit is set (the WCOL bit is a status flag only).

Clearing the WCOL bit is done through a software sequence (see Figure 53).

Figure 53. Clearing the WCOL Bit (Write Collision Flag) Software Sequence


Clearing sequence after SPIF = 1 (end of a data byte transfer)
1st Step
Read SPICSR
2nd Step
Read SPIDR
RESULT
SPIF = 0
WCOL = 0
Clearing sequence before SPIF = 1 (during a data byte transfer)
1st Step
Read SPICSR
2nd Step
Read SPIDR
RESULT
WCOL = 0

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

LINSCI™ SERIAL COMMUNICATION INTERFACE (cont'd)

11.5.4 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:

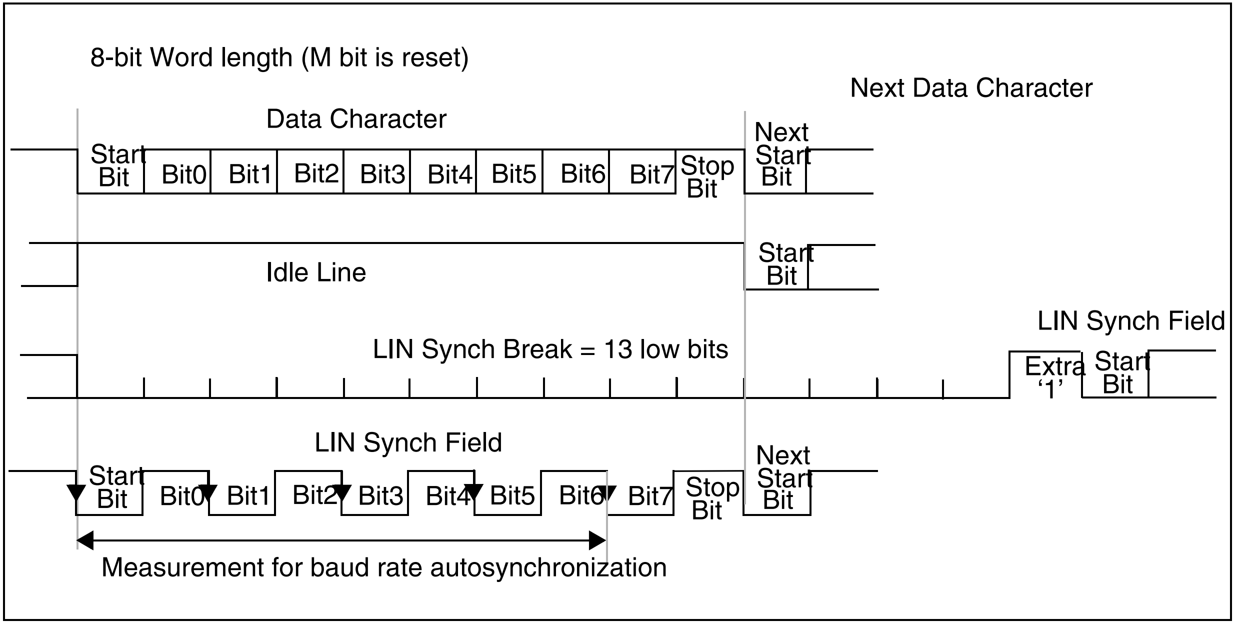
– A conventional type for commonly-used baud rates

– 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 (LIN Mode) (cont'd)

Figure 58. LIN Characters



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

11.5.9.7 LINSCI Clock Tolerance

LINSCI Clock Tolerance when unsynchronized

When LIN slaves are unsynchronized (meaning no characters have been transmitted for a relatively long time), the maximum tolerated deviation of the LINSCI clock is +/-15%.

If the deviation is within this range then the LIN Synch Break is detected properly when a new reception occurs.

This is made possible by the fact that masters send 13 low bits for the LIN Synch Break, which can be interpreted as 11 low bits (13 bits -15% = 11.05) by a "fast" slave and then considered as a LIN Synch Break. According to the LIN specification, a LIN Synch Break is valid when its duration is greater than t_{SBRKTS} = 10. This means that the LIN Synch Break must last at least 11 low bits.

Note: If the period desynchronization of the slave is +15% (slave too slow), the character "00h" which represents a sequence of 9 low bits must not be interpreted as a break character (9 bits + 15% = 10.35). Consequently, a valid LIN Synch break must last at least 11 low bits.

LINSCI Clock Tolerance when Synchronized

When synchronization has been performed, following reception of a LIN Synch Break, the LINSCI, in LIN mode, has the same clock deviation tolerance as in SCI mode, which is explained below:

During reception, each bit is oversampled 16 times. The mean of the 8th, 9th and 10th samples is considered as the bit value.

Consequently, the clock frequency should not vary more than 6/16 (37.5%) within one bit.

The sampling clock is resynchronized at each start bit, so that when receiving 10 bits (one start bit, 1 data byte, 1 stop bit), the clock deviation should not exceed 3.75%.

11.5.9.8 Clock Deviation Causes

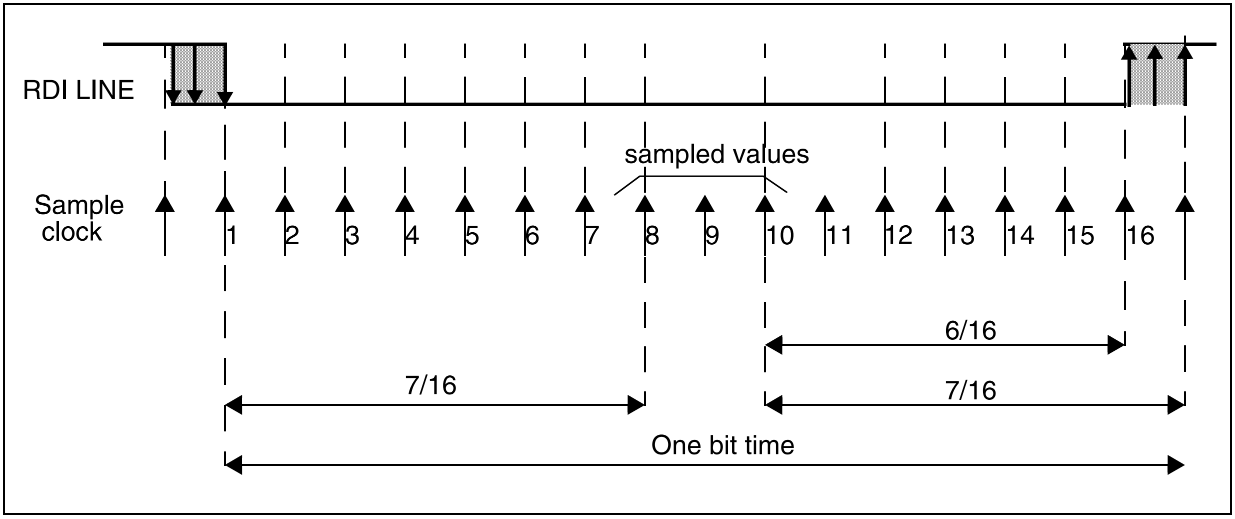
The causes which contribute to the total deviation are:

- D_{TRA}: Deviation due to transmitter error. **Note:** The transmitter can be either a master or a slave (in case of a slave listening to the response of another slave).
- D_{MEAS}: Error due to the LIN Synch measurement performed by the receiver.
- D_{QUANT}: Error due to the baud rate quantization of the receiver.
- D_{REC}: Deviation of the local oscillator of the receiver: This deviation can occur during the reception of one complete LIN message assuming that the deviation has been compensated at the beginning of the message.
- D_{TCL}: Deviation due to the transmission line (generally due to the transceivers)

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

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

Figure 64.Bit Sampling in Reception Mode



13.7 EMC (ELECTROMAGNETIC COMPATIBILITY) CHARACTERISTICS

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

13.7.1 Functional EMS (Electro Magnetic Susceptibility)

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

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

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

13.7.1.1 Designing hardened software to avoid noise problems

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

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

Software recommendations:

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

– Corrupted program counter

– Unexpected reset

– Critical Data corruption (control registers...)

Prequalification trials:

Most of the common failures (unexpected reset and program counter corruption) can be reproduced by manually forcing a low state on the RESET pin or the Oscillator pins for 1 second.

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

Table 25: EMS test results

Symbol	Parameter	Conditions	Level/ Class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz conforms to IEC 1000-4-2	3B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100pF on V_{DD} and V_{DD} pins to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz conforms to IEC 1000-4-4	3B

Figure 94. Typical V_{DD}-V_{OH} at V_{DD}=5V

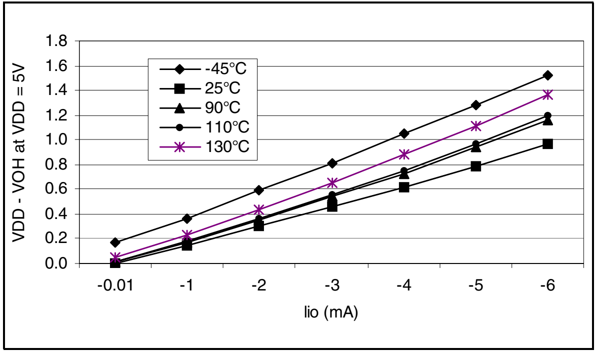


Figure 95. Typical V_{OL} vs. V_{DD} (standard I/Os)

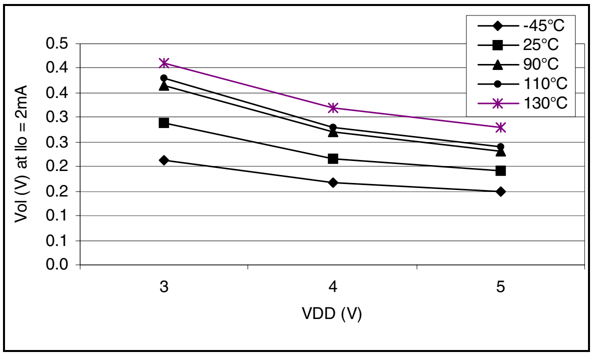


Figure 96. Typical V_{OL} vs. V_{DD} (high-sink I/Os)

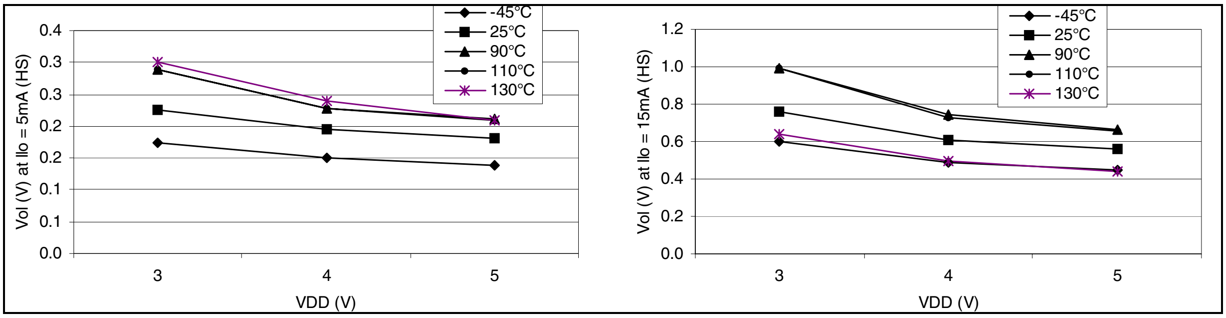
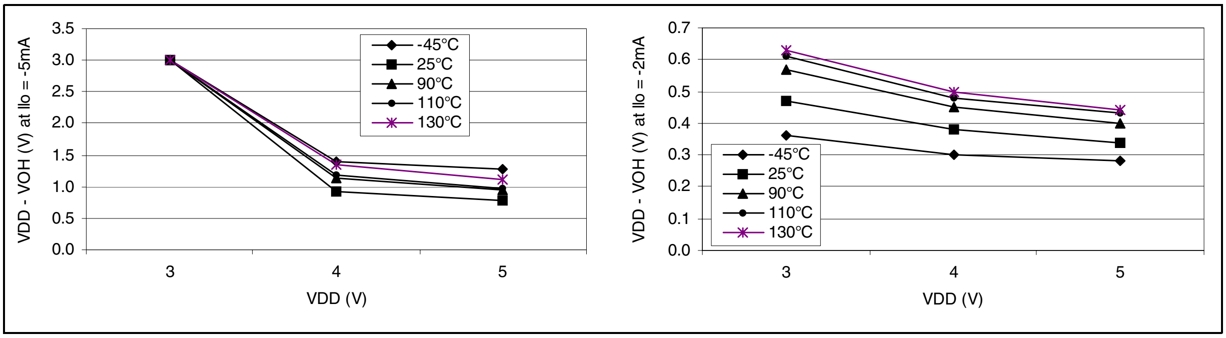


Figure 97. Typical V_{DD}-V_{OH} vs. V_{DD}



IMPORTANT NOTES (Cont'd)

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 110. Workaround is shown in bold characters.

Figure 110. LINSCI Interrupt routine

```
@interrupt void LINSCI_IT ( void ) /* LINSCI interrupt routine */
{
     /* clear flags */
     SCISR_buffer = SCISR;
     SCIDR_buffer = SCIDR;

     if ( SCISR_buffer & LHE )/* header error ? */
     {
          if (!LHLR)/* header time-out? */
          {
               if ( !(SCICR2 & RWU) )/* active mode ? */
               {
                    _asm("sim");/* disable interrupts */
                    SCISR;
                    SCIDR;/* Clear RDRF flag */
                    SCICR2 |= RWU;/* set mute mode */
                    SCISR;
                    SCIDR;/* Clear RDRF flag */
                    SCICR2 |= RWU;/* set mute mode */
                    _asm("rim");/* enable interrupts */
               }
          }
     }
}
```

Example using Cosmic compiler syntax