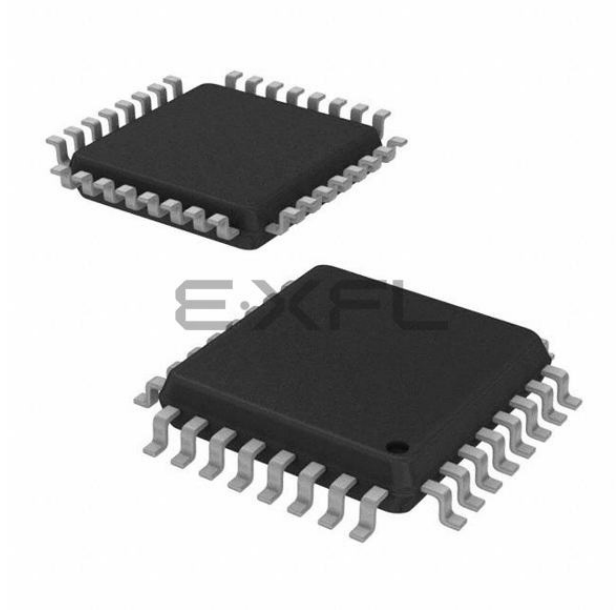


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

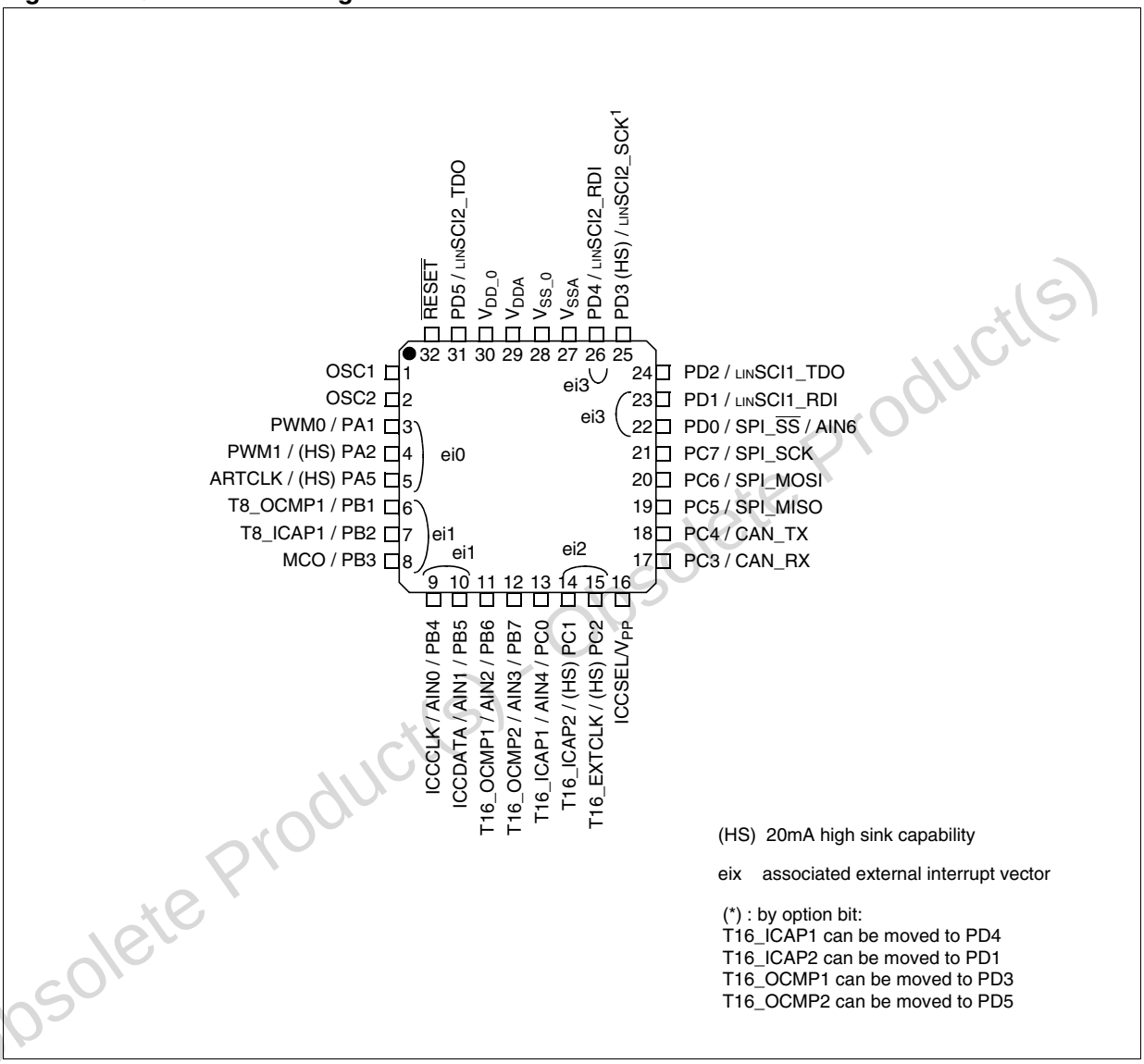
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded"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

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	CANbus, LINbusSCI, SPI
Peripherals	LVD, POR, PWM, WDT
Number of I/O	24
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 6x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	32-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st72f561k9t3

PIN DESCRIPTION (Cont'd)

Figure 4. LQFP 32-Pin Package Pinout



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

Address	Block	Register Label	Register Name	Reset Status	Remarks
0068h	Active CAN	CMCR	CAN Master Control Register		R/W
0069h		CMSR	CAN Master Status Register		R/W
006Ah		CTSR	CAN Transmit Status Register		R/W
006Bh		CTPR	CAN Transmit Priority Register		R/W
006Ch		CRFR	CAN Receive FIFO Register		R/W
006Dh		CIER	CAN Interrupt Enable Register		R/W
006Eh		CDGR	CAN Diagnosis Register		R/W
006Fh		CPSR	CAN Page Selection Register		R/W
0070h		PAGES	PAGE REGISTER 0		R/W
0071h			PAGE REGISTER 1		R/W
0072h			PAGE REGISTER 2		R/W
0073h			PAGE REGISTER 3		R/W
0074h			PAGE REGISTER 4		R/W
0075h			PAGE REGISTER 5		R/W
0076h			PAGE REGISTER 6		R/W
0077h			PAGE REGISTER 7		R/W
0078h			PAGE REGISTER 8		R/W
0079h			PAGE REGISTER 9		R/W
007Ah			PAGE REGISTER 10		R/W
007Bh			PAGE REGISTER 11		R/W
007Ch			PAGE REGISTER 12		R/W
007Dh			PAGE REGISTER 13		R/W
007Eh			PAGE REGISTER 14		R/W
007Fh			PAGE REGISTER 15		R/W

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.

INTERRUPTS (Cont'd)

Table 8. Dedicated Interrupt Instruction Set

Instruction	New Description	Function/Example	I1	H	I0	N	Z	C
HALT	Entering Halt mode		1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC	I1	H	I0	N	Z	C
JRM	Jump if I1:0 = 11 (level 3)	I1:0 = 11 ?						
JRNM	Jump if I1:0 <> 11	I1:0 <> 11 ?						
POP CC	Pop CC from the Stack	Mem => CC	I1	H	I0	N	Z	C
RIM	Enable interrupt (level 0 set)	Load 10 in I1:0 of CC	1		0			
SIM	Disable interrupt (level 3 set)	Load 11 in I1:0 of CC	1		1			
TRAP	Software trap	Software NMI	1		1			
WFI	Wait for interrupt		1		0			

Note: During the execution of an interrupt routine, the HALT, POPCC, RIM, SIM and WFI instructions change the current software priority up to the next IRET instruction or one of the previously mentioned instructions.

MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK (Cont'd)

10.2.4 Low Power Modes

Mode	Description
WAIT	No effect on MCC/RTC peripheral. MCC/RTC interrupt cause the device to exit from WAIT mode.
ACTIVE HALT	No effect on MCC/RTC counter (OIE bit is set), the registers are frozen. MCC/RTC interrupt cause the device to exit from ACTIVE HALT mode.
HALT and AWUF HALT	MCC/RTC counter and registers are frozen. MCC/RTC operation resumes when the MCU is woken up by an interrupt with "exit from HALT" capability.

10.2.5 Interrupts

The MCC/RTC interrupt event generates an interrupt if the OIE bit of the MCCSR register is set and the interrupt mask in the CC register is not active (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Time base overflow event	OIF	OIE	Yes	No[1)]

Note:

The MCC/RTC interrupt wakes up the MCU from ACTIVE HALT mode, not from HALT or AWUF HALT mode.

10.2.6 Register Description

MCC CONTROL/STATUS REGISTER (MCCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
MCO	CP1	CP0	SMS	TB1	TB0	OIE	OIF

Bit 7 = **MCO** *Main clock out selection*
This bit enables the MCO alternate function on the corresponding I/O port. It is set and cleared by software.
0: MCO alternate function disabled (I/O pin free for general-purpose I/O)
1: MCO alternate function enabled (f_{OSC2} on I/O port)

Bits 6:5 = **CP[1:0]** *CPU clock prescaler*
These bits select the CPU clock prescaler which is applied in the different slow modes. Their action is conditioned by the setting of the SMS bit. These two bits are set and cleared by software

f_{CPU} in SLOW mode	CP1	CP0
f_{OSC2} / 2	0	0
f_{OSC2} / 4	0	1
f_{OSC2}/ 8	1	0
f_{OSC2} / 16	1	1

Bit 4 = **SMS** *Slow mode select*
This bit is set and cleared by software.
0: Normal mode. $f_{CPU} = f_{OSC2}$
1: Slow mode. f_{CPU} is given by CP1, CP0
See Section 8.2 "SLOW MODE" and Section 10.2 "MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK MCC/RTC" for more details.

Bits 3:2 = **TB[1:0]** *Time base control*

These bits select the programmable divider time base. They are set and cleared by software.

Counter Prescaler	Time Base		TB1	TB0
	f_{OSC2} =4 MHz	f_{OSC2}=8 MHz		
16000	4ms	2ms	0	0
32000	8ms	4ms	0	1
80000	20ms	10ms	1	0
200000	50ms	25ms	1	1

A modification of the time base is taken into account at the end of the current period (previously set) to avoid an unwanted time shift. This allows to use this time base as a real time clock.

Bit 1 = **OIE** *Oscillator interrupt enable*
This bit set and cleared by software.
0: Oscillator interrupt disabled
1: Oscillator interrupt enabled
This interrupt can be used to exit from ACTIVE HALT mode.
When this bit is set, calling the ST7 software HALT instruction enters the ACTIVE HALT power saving mode.

10.4 16-BIT TIMER

10.4.1 Introduction

The timer consists of a 16-bit free-running counter driven by a programmable prescaler.

It may be used for a variety of purposes, including pulse length measurement of up to two input signals (*input capture*) or generation of up to two output waveforms (*output compare* and *PWM*).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the CPU clock prescaler.

Some ST7 devices have two on-chip 16-bit timers. They are completely independent, and do not share any resources. They are synchronized after a MCU reset as long as the timer clock frequencies are not modified.

This description covers one or two 16-bit timers. In ST7 devices with two timers, register names are prefixed with TA (Timer A) or TB (Timer B).

10.4.2 Main Features

- Programmable prescaler: f_{CPU} divided by 2, 4 or 8
- Overflow status flag and maskable interrupt
- External clock input (must be at least four times slower than the CPU clock speed) with the choice of active edge
- 1 or 2 Output Compare functions each with:
  - 2 dedicated 16-bit registers
  - 2 dedicated programmable signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- 1 or 2 Input Capture functions each with:
  - 2 dedicated 16-bit registers
  - 2 dedicated active edge selection signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Pulse width modulation mode (PWM)
- One Pulse mode
- Reduced Power Mode
- 5 alternate functions on I/O ports (ICAP1, ICAP2, OCMP1, OCMP2, EXTCLK)*

The Block Diagram is shown in Figure 48.

***Note:** Some timer pins may not be available (not bonded) in some ST7 devices. Refer to the device pin out description.

When reading an input signal on a non-bonded pin, the value will always be '1'.

10.4.3 Functional Description

10.4.3.1 Counter

The main block of the Programmable Timer is a 16-bit free running upcounter and its associated 16-bit registers. The 16-bit registers are made up of two 8-bit registers called high and low.

Counter Register (CR):

- Counter High Register (CHR) is the most significant byte (MS Byte).
- Counter Low Register (CLR) is the least significant byte (LS Byte).

Alternate Counter Register (ACR)

- Alternate Counter High Register (ACHR) is the most significant byte (MS Byte).
- Alternate Counter Low Register (ACLR) is the least significant byte (LS Byte).

These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR), (see note at the end of paragraph titled 16-bit read sequence).

Writing in the CLR register or ACLR register resets the free running counter to the FFFCh value.
Both counters have a reset value of FFFCh (this is the only value which is reloaded in the 16-bit timer). The reset value of both counters is also FFFCh in One Pulse mode and PWM mode.

The timer clock depends on the clock control bits of the CR2 register, as illustrated in Table 17 Clock Control Bits. The value in the counter register repeats every 131072, 262144 or 524288 CPU clock cycles depending on the CC[1:0] bits.
The timer frequency can be $f_{CPU}/2$, $f_{CPU}/4$, $f_{CPU}/8$ or an external frequency.

8-BIT TIMER (Cont'd)

Figure 63. Input Capture Block Diagram

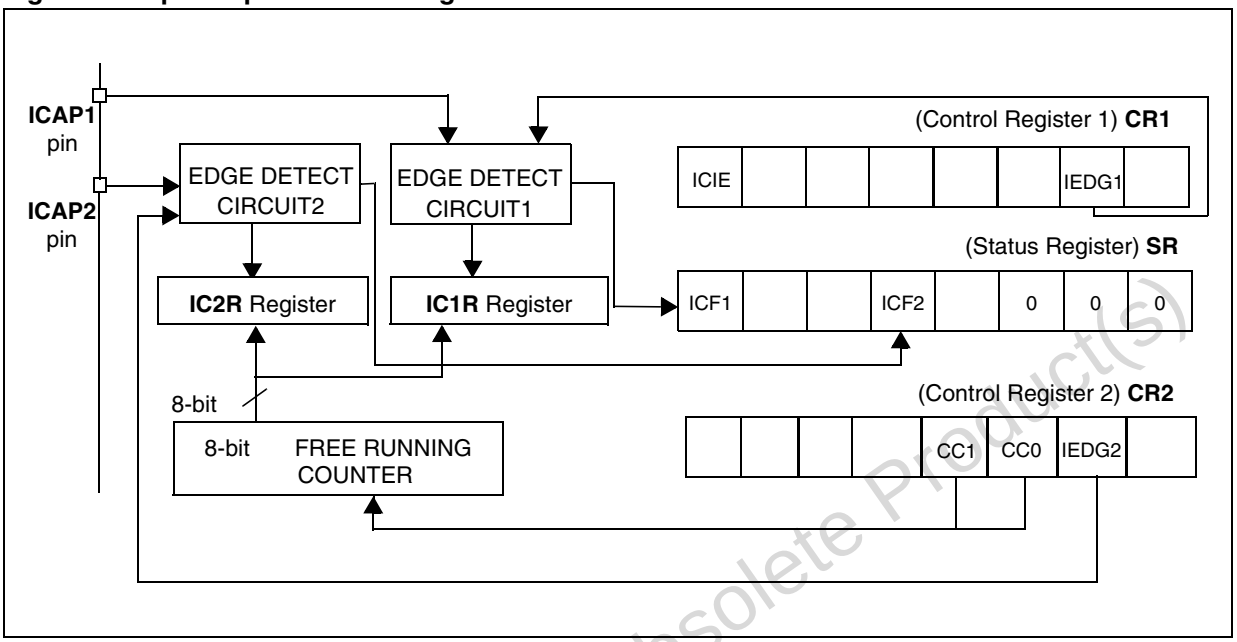
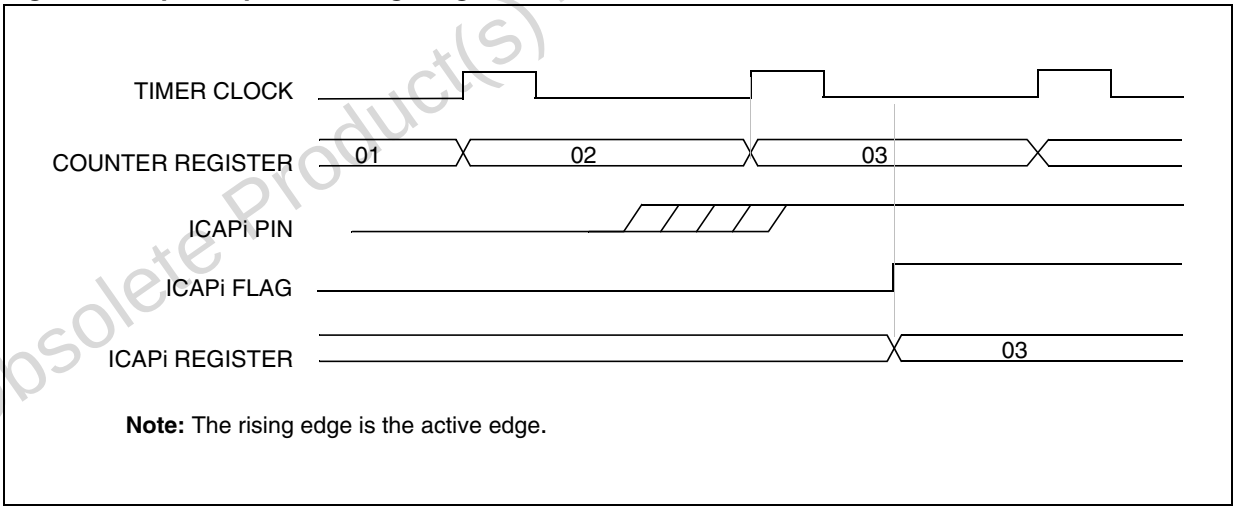


Figure 64. Input Capture Timing Diagram



Note: The rising edge is the active edge.

SERIAL PERIPHERAL INTERFACE (cont'd)

10.6.3.2 Slave Select Management

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

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

In Master mode:

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

In Slave Mode:

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

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

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

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

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

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

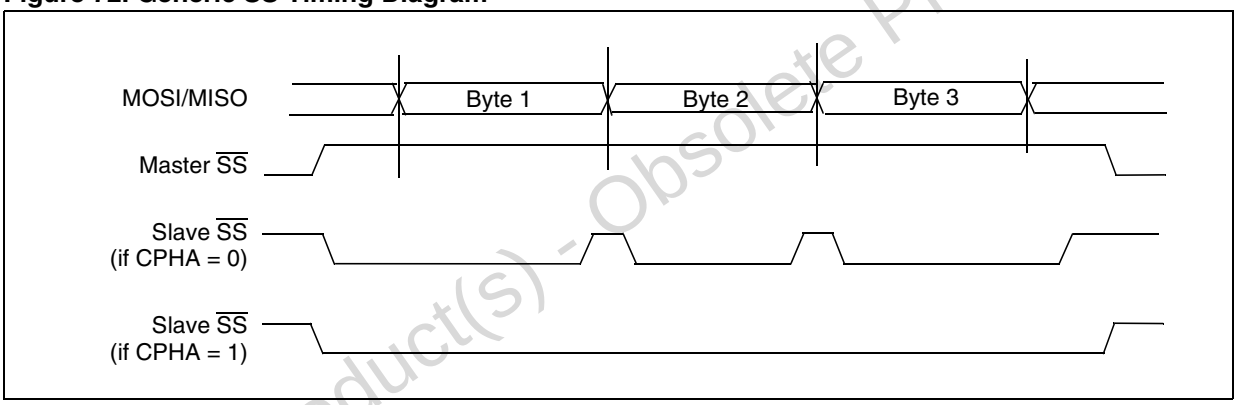
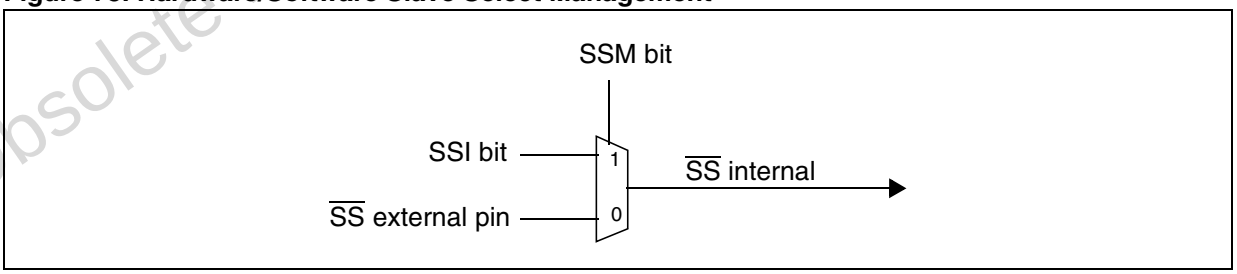
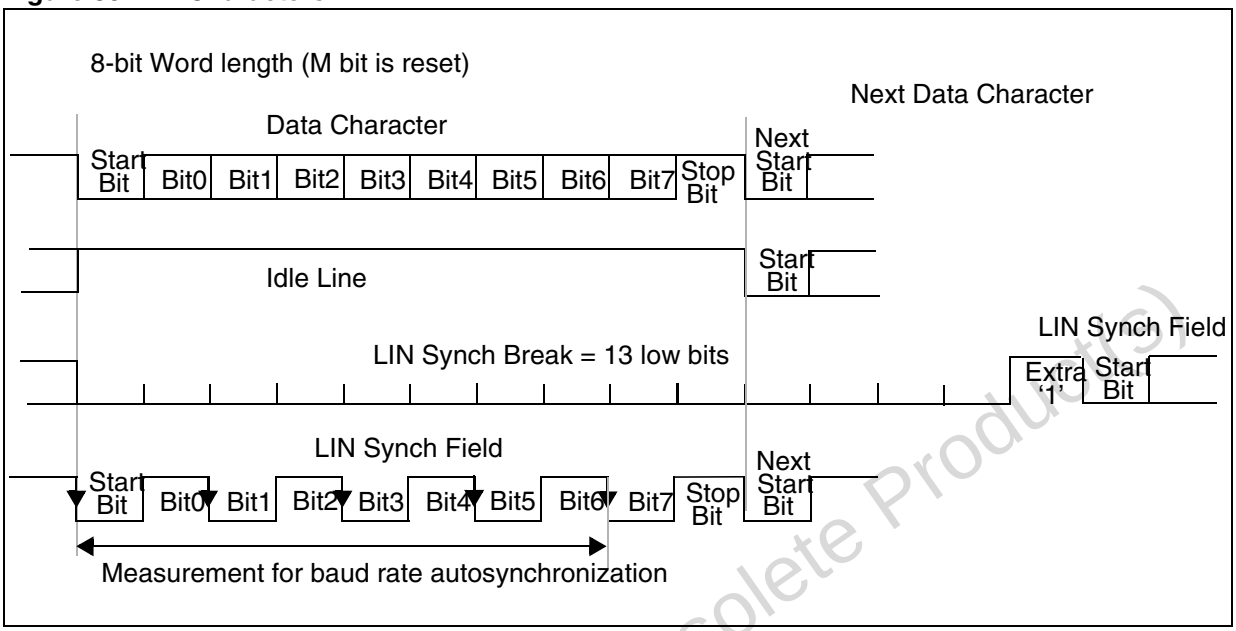


Figure 73. Hardware/Software Slave Select Management



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

Figure 80. LIN Characters



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

10.7.9.7 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%.

10.7.9.8 Clock Deviation Causes

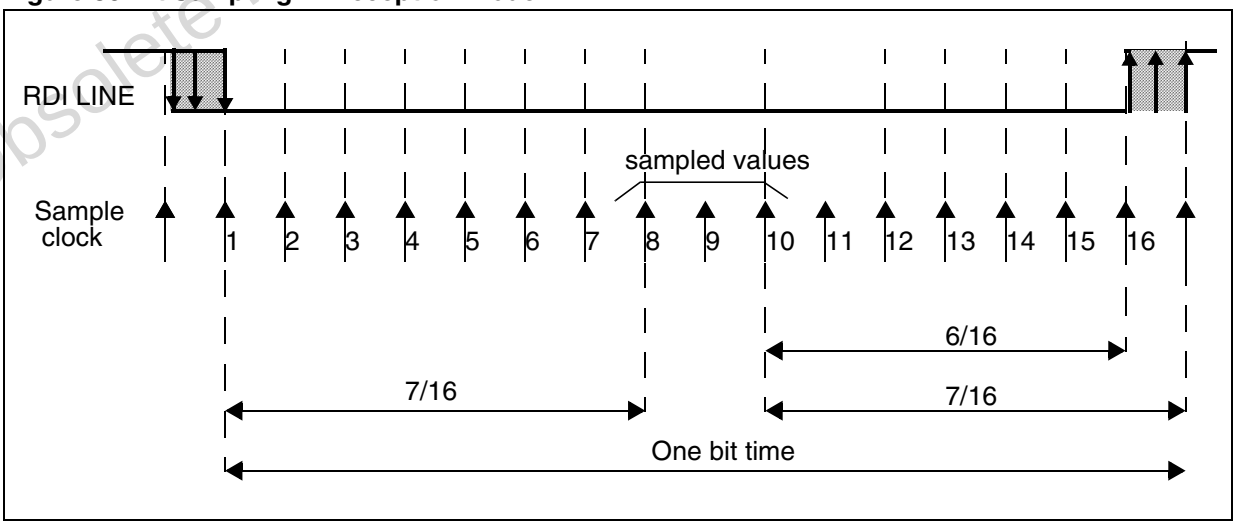
The causes which contribute to the total deviation are:

- 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 86.Bit Sampling in Reception Mode



10.8 LINSCI SERIAL COMMUNICATION INTERFACE (LIN Master Only)

10.8.1 Introduction

The Serial Communications Interface (SCI) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The SCI offers a very wide range of baud rates using two baud rate generator systems.

10.8.2 Main Features

- Full duplex, asynchronous communications
- NRZ standard format (Mark/Space)
- Dual baud rate generator systems
- Independently programmable transmit and receive baud rates up to 500K baud.
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- 2 receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- Separate enable bits for Transmitter and Receiver
- 4 error detection flags:
  - Overrun error
  - Noise error
  - Frame error
  - Parity error
- 5 interrupt sources with flags:
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error detected
- Transmitter clock output
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode
- LIN Synch Break send capability

10.8.3 General Description

The interface is externally connected to another device by three pins (see Figure 88 on page 153). Any SCI bidirectional communication requires a minimum of two pins: Receive Data In (RDI) and Transmit Data Out (TDO):

– SCLK: Transmitter clock output. This pin outputs the transmitter data clock for synchronous transmission (no clock pulses on start bit and stop bit, and a software option to send a clock pulse on the last data bit). This can be used to control peripherals that have shift registers (e.g. LCD drivers). The clock phase and polarity are software programmable.

– 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 frames 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 frame is complete.

This interface uses two 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.

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

10.8.4.3 Receiver

The SCI can receive data words of either 8 or 9 bits. When the M bit is set, word length is 9 bits and the MSB is stored in the R8 bit in the SCICR1 register.

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists or a buffer (RDR) between the internal bus and the received shift register (see Figure 88 on page 153).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the SCIBRR and the SCIERPR registers.

– Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

– The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR.

– An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

– The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

Clearing the RDRF bit is performed by the following software sequence done by:

1. An access to the SCISR register

2. A read to the SCIDR register.

The RDRF bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Break Character

When a break character is received, the SCI handles it as a framing error.

Idle Character

When an idle frame is detected, there is the same procedure as a data received character plus an interrupt if the ILIE bit is set and the I bit is cleared in the CCR register.

Overrun Error

An overrun error occurs when a character is received when RDRF has not been reset. Data cannot be transferred from the shift register to the RDR register until the RDRF bit is cleared.

When a overrun error occurs:

– The OR bit is set.

– The RDR content is not lost.

– The shift register is overwritten.

– An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

The OR bit is reset by an access to the SCISR register followed by a SCIDR register read operation.

Noise Error

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

When noise is detected in a frame:

– The NF is set at the rising edge of the RDRF bit.

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The NF bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

Framing Error

A framing error is detected when:

– The stop bit is not recognized on reception at the expected time, following either a de-synchronization or excessive noise.

– A break is received.

When the framing error is detected:

– the FE bit is set by hardware

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The FE bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

beCAN CONTROLLER (Cont'd)

In the worst case configuration, if the CAN cell speed is set to the maximum baud rate, one bit time is 8 CPU cycle. In this case the minimum time between the end of the acknowledge and the critical period is 52 CPU cycles (48 for the 6 bit times + 4 for the (PROP SEG + $T_{Seg\ 1}$). According to the previous code timing, we need less than 15 cycles from the time we see the dominant state to the time we perform the FIFO release (one full loop + the actual release) therefore the application will never release the FIFO at the critical time when this workaround is implemented.

Timing analysis

- Time spent in the workaround

Inside a CAN frame, the longest period that the Rx pin stays in recessive state is 5 bits. At the end of the frame, the time between the acknowledge dominant bit and the end of reception (signaled by REC bit status) is $8t_{CANbit}$, therefore the maximum time spent in the workaround is: $8t_{CANbit}+t_{loop}+t_{test}+t_{release}$ in this case or $8t_{CANbit}+25t_{CPU}$.

At low speed, this time could represent a long delay for the application, therefore it makes sense to evaluate how frequently this delay occurs.

In order to reach the critical FMP = 2, the CAN node needs to receive two messages without servicing them. Then in order to reach the critical window, the cell has to receive a third one and the application has to release the mailbox at the same time, at the end of the reception.

In the application, messages are not processed only if either the interrupt are disabled or higher level interrupts are being serviced.

Therefore if:

$t_{IT\ higher\ level} + t_{IT\ disable} + t_{IT\ CAN} < 2 \times t_{CAN\ frame}$

the application will never wait in the workaround

$t_{IT\ higher\ level}$: This the sum of the duration of all the interrupts with a level strictly higher than the CAN interrupt level

$t_{IT\ disable}$: This is the longest time the application disables the CAN interrupt (or all interrupts)

$t_{IT\ CAN}$: This is the maximum duration between the beginning of the CAN interrupt and the actual location of the workaround

$t_{CAN\ frame}$: This is minimum CAN frame duration

Figure 111. Critical Window Timing Diagram

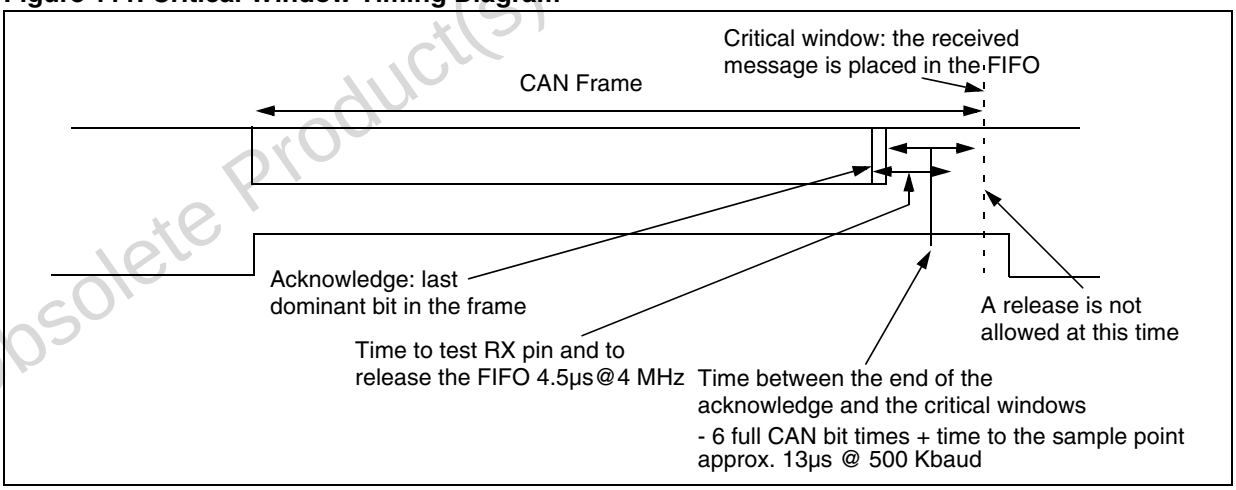
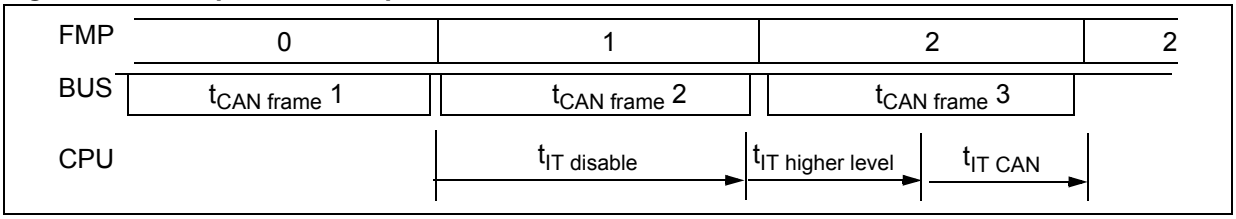


Figure 112. Reception of a Sequence of Frames



beCAN CONTROLLER (Cont'd)

MAILBOX DATA LENGTH CONTROL REGISTER (MDLC)

All bits of this register is write protected when the mailbox is not in empty state.

Read / Write
Reset Value: xxxx xxxx (xxh)

7							0
0	0	0	0	DLC3	DLC2	DLC1	DLC0

Bit 7 = Reserved, must be kept cleared.

Bits 6:4 = Reserved, forced to 0 by hardware.

Bits 3:0 = **DLC[3:0]** *Data Length Code*
This field defines the number of data bytes a data frame contains or a remote frame request.

MAILBOX DATA REGISTERS (MDAR[7:0])

All bits of this register are write protected when the mailbox is not in empty state.

Read / Write
Reset Value: Undefined

7							0
DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

Bits 7:0 = **DATA[7:0]** *Data*
A data byte of the message. A message can contain from 0 to 8 data bytes.

beCAN CONTROLLER (Cont'd)

CAN FILTER CONFIGURATION REG.1 (CFCR2)

All bits of this register are set and cleared by software.
Read / Write
Reset Value: 0000 0000 (00h)

7							0
0	FSC51	FSC50	FACT5	0	FSC41	FSC40	FACT4

Bit 7 = Reserved. Forced to 0 by hardware.

Bits 6:5 = **FSC5[1:0]** *Filter Scale Configuration*
These bits define the scale configuration of Filter 5.

Bit 4 = **FACT5** *Filter Active*
The software sets this bit to activate filter 5. To modify the Filter 5 registers (CF5R[0:7]) the FACT5 bit must be cleared.
0: Filter 5 is not active
1: Filter 5 is active

Bit 3 = Reserved. Forced to 0 by hardware.

Bits 2:1 = **FSC4[1:0]** *Filter Scale Configuration*
These bits define the scale configuration of Filter 4.

Bit 0 = **FACT4** *Filter Active*
The software sets this bit to activate Filter 4. To modify the Filter 4 registers (CF4R[0:7]), the FACT4 bit must be cleared.
0: Filter 4 is not active
1: Filter 4 is active

CAN FILTER MODE REGISTER (CFMR0)

All bits of this register are set and cleared by software.
Read / Write
Reset Value: 0000 0000 (00h)

7							0
FMH3	FML3	FMH2	FML2	FMH1	FML1	FMH0	FML0

Bit 7 = **FMH3** *Filter Mode High*
Mode of the high registers of Filter 3.
0: High registers are in mask mode
1: High registers are in identifier list mode

Bit 6 = **FML3** *Filter Mode Low*
Mode of the low registers of Filter 3.
0: Low registers are in mask mode
1: Low registers are in identifier list mode

Bit 5 = **FMH2** *Filter Mode High*
Mode of the high registers of Filter 2.
0: High registers are in mask mode
1: High registers are in identifier list mode

Bit 4 = **FML2** *Filter Mode Low*
Mode of the low registers of Filter 2.
0: Low registers are in mask mode
1: Low registers are in identifier list mode

Bit 3 = **FMH1** *Filter Mode High*
Mode of the high registers of Filter 1.
0: High registers are in mask mode
1: High registers are in identifier list mode

Bit 2 = **FML1** *Filter Mode Low*
Mode of the low registers of filter 1.
0: Low registers are in mask mode
1: Low registers are in identifier list mode

Bit 1 = **FMH0** *Filter Mode High*
Mode of the high registers of filter 0.
0: High registers are in mask mode
1: High registers are in identifier list mode

Bit 0 = **FML0** *Filter Mode Low*
Mode of the low registers of filter 0.
0: Low registers are in mask mode
1: Low registers are in identifier list mode

beCAN CONTROLLER (Cont'd)

CAN FILTER MODE REGISTER (CFMR1)

All bits of this register are set and cleared by software.
Read / Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	FMH5	FML5	FMH4	FML4

Bits 7:4 = Reserved. Forced to 0 by hardware.

Bit 3 = **FMH5** *Filter Mode High*
Mode of the high registers of Filter 5.
0: High registers are in mask mode
1: High registers are in identifier list mode

Bit 2 = **FML5** *Filter Mode Low*
Mode of the low registers of filter 5.
0: Low registers are in mask mode
1: Low registers are in identifier list mode

Bit 1 = **FMH4** *Filter Mode High*
Mode of the high registers of filter 4.
0: High registers are in mask mode
1: High registers are in identifier list mode

Bit 0 = **FML4** *Filter Mode Low*
Mode of the low registers of filter 4.
0: Low registers are in mask mode
1: Low registers are in identifier list mode

FILTER x REGISTER[7:0] (CFxR[7:0])

Read / Write
Reset Value: Undefined

7							0
FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0

In all configurations:

Bits 7:0 = **FB[7:0]** *Filter Bits*

Identifier
Each bit of the register specifies the level of the corresponding bit of the expected identifier.
0: Dominant bit is expected
1: Recessive bit is expected

Mask
Each bit of the register specifies whether the bit of the associated identifier register must match with the corresponding bit of the expected identifier or not.
0: Don't care, the bit is not used for the comparison
1: Must match, the bit of the incoming identifier must have the same level has specified in the corresponding identifier register of the filter.

Note: Each filter x is composed of 8 registers, CFxR[7:0]. Depending on the scale and mode configuration of the filter the function of each register can differ. For the filter mapping, functions description and mask registers association, refer to Section 0.1.4.3 Identifier Filtering.

A Mask/Identifier register in **mask mode** has the same bit mapping as in **identifier list** mode.

Note: To modify these registers, the corresponding FACT bit in the CFCR register must be cleared.

INSTRUCTION SET OVERVIEW (Cont'd)

11.1.1 Inherent

All Inherent instructions consist of a single byte. The opcode fully specifies all the required information for the CPU to process the operation.

Inherent Instruction	Function
NOP	No operation
TRAP	S/W Interrupt
WFI	Wait For Interrupt (Low Power Mode)
HALT	Halt Oscillator (Lowest Power Mode)
RET	Sub-routine Return
IRET	Interrupt Sub-routine Return
SIM	Set Interrupt Mask (level 3)
RIM	Reset Interrupt Mask (level 0)
SCF	Set Carry Flag
RCF	Reset Carry Flag
RSP	Reset Stack Pointer
LD	Load
CLR	Clear
PUSH/POP	Push/Pop to/from the stack
INC/DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
MUL	Byte Multiplication
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles

11.1.2 Immediate

Immediate instructions have 2 bytes, the first byte contains the opcode, the second byte contains the operand value.

Immediate Instruction	Function
LD	Load
CP	Compare
BCP	Bit Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Operations

11.1.3 Direct

In Direct instructions, the operands are referenced by their memory address.

The direct addressing mode consists of two submodes:

Direct (short)

The address is a byte, thus requires only one byte after the opcode, but only allows 00 - FF addressing space.

Direct (long)

The address is a word, thus allowing 64 Kbyte addressing space, but requires 2 bytes after the opcode.

11.1.4 Indexed (No Offset, Short, Long)

In this mode, the operand is referenced by its memory address, which is defined by the unsigned addition of an index register (X or Y) with an offset.

The indirect addressing mode consists of three submodes:

Indexed (No Offset)

There is no offset, (no extra byte after the opcode), and allows 00 - FF addressing space.

Indexed (Short)

The offset is a byte, thus requires only one byte after the opcode and allows 00 - 1FE addressing space.

Indexed (long)

The offset is a word, thus allowing 64 Kbyte addressing space and requires 2 bytes after the opcode.

11.1.5 Indirect (Short, Long)

The required data byte to do the operation is found by its memory address, located in memory (pointer).

The pointer address follows the opcode. The indirect addressing mode consists of two submodes:

Indirect (short)

The pointer address is a byte, the pointer size is a byte, thus allowing 00 - FF addressing space, and requires 1 byte after the opcode.

Indirect (long)

The pointer address is a byte, the pointer size is a word, thus allowing 64 Kbyte addressing space, and requires 1 byte after the opcode.

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

12.4.1 Supply and Clock Managers

The previous current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To obtain the total device consumption, the two current values must be added (except for HALT mode).

Symbol	Parameter	Conditions	Typ	Max[1)]	Unit
$I_{DD(RES)}$	Supply current of resonator oscillator[2)3)]		See Section 12.5.3 on page 227		µA
$I_{DD(PLL)}$	PLL supply current	V_{DD} = 5V	360		
$I_{DD(LVD)}$	LVD supply current	HALT mode, V_{DD} = 5V	150	300	

Notes:

1. Data based on characterization results, not tested in production.
2. Data based on characterization results done with the external components specified in Section 12.5.3, not tested in production.
3. As the oscillator is based on a current source, the consumption does not depend on the voltage.

12.7 MEMORY CHARACTERISTICS

12.7.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode[1)]	HALT mode (or RESET)	1.6			V

12.7.2 FLASH Memory

DUAL VOLTAGE HDFLASH MEMORY						
Symbol	**Parameter**	**Conditions**	**Min[2)]**	**Typ**	**Max[2)]**	**Unit**
f_{CPU}	Operating frequency	Read mode	0		8	MHz
		Write / Erase mode	1		8	
V_{PP}	Programming voltage[3)]	$4.5V \leq V_{DD} \leq 5.5V$	11.4		12.6	V
I_{PP}	V_{PP} current[4)5)]	Read (V_{PP} = 12V)			200	µA
		Write / Erase			30	mA
t_{VPP}	Internal V_{PP} stabilization time			10[4)]		µs
t_{RET}	Data retention	T_A = 85°C	40			years
		T_A = 105°C	15			
		T_A = 125°C	7			
N_{RW}	Write erase cycles	T_A = 25°C	100			cycles
T_{PROG} T_{ERASE}	Programming or erasing temperature range		-40	25	85	°C

Notes:

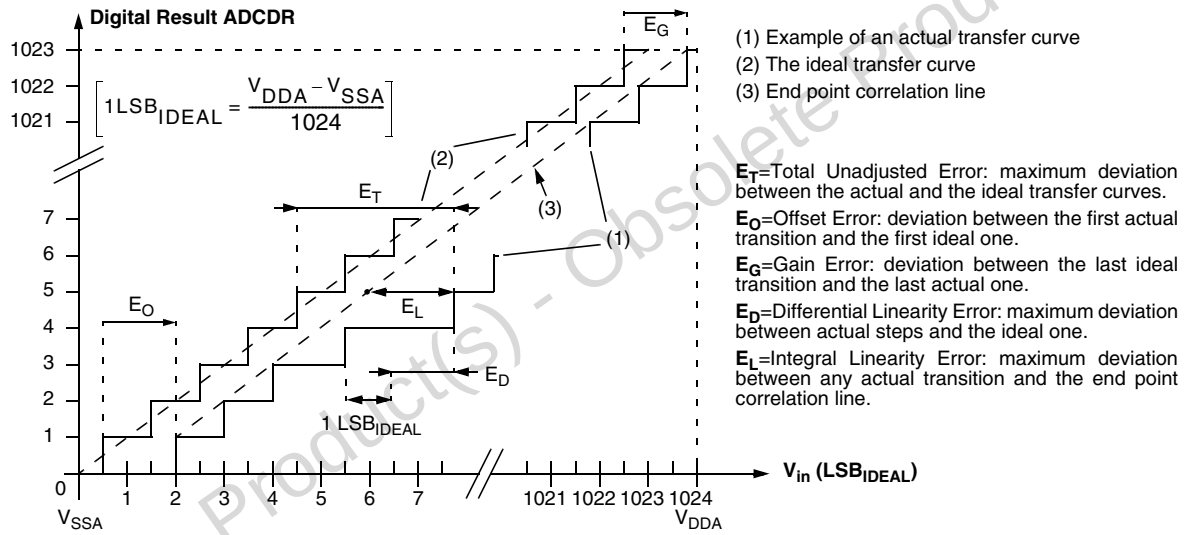
1. Minimum V_{DD} supply voltage without losing data stored in RAM (in HALT mode or under RESET) or in hardware registers (only in HALT mode). Not tested in production.
2. Data based on characterization results, not tested in production.
3. V_{PP} must be applied only during the programming or erasing operation and not permanently for reliability reasons.
4. Data based on simulation results, not tested in production.
5. In Write / erase mode the I_{DD} supply current consumption is the same as in Run mode (see Section 12.2.2)

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
$\|E_T\|$	Total unadjusted error[1)]		3.2	5	LSB
$\|E_O\|$	Offset error[1)]		1	4	
$\|E_G\|$	Gain Error[1)]		0.7		
$\|E_D\|$	Differential linearity error[1)]		1.5	2.3	
$\|E_L\|$	Integral linearity error[1)]		1.2	3.6	

Figure 145. ADC Accuracy Characteristics



Notes:

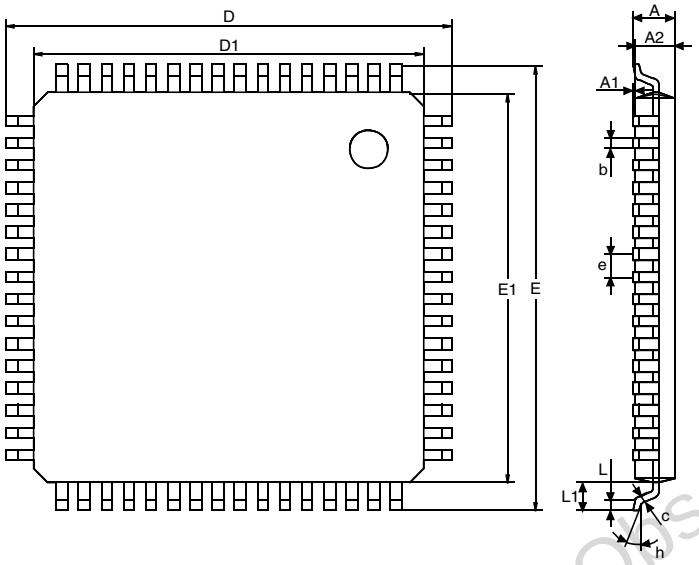
1. Data based on characterization results, not tested in production. ADC Accuracy vs. Negative Injection Current: Injecting negative current on any of the standard (non-robust) analog input pins should be avoided as this significantly reduces the accuracy of the conversion being performed on another 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.

13 PACKAGE CHARACTERISTICS

13.1 PACKAGE MECHANICAL DATA

Figure 146. 64-Pin Low Profile Quad Flat Package (14x14)



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.30	0.37	0.45	0.0118	0.0146	0.0177
c	0.09		0.20	0.0035		0.0079
D		16.00			0.6299	
D1		14.00			0.5512	
E		16.00			0.6299	
E1		14.00			0.5512	
e		0.80			0.0315	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	64					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.