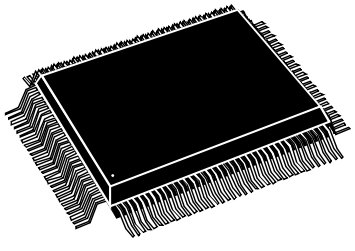
ST72589BW, ST72389BW

8-BIT MCU WITH NESTED INTERRUPTS, DOT MATRIX LCD, ADC, TIMERS, PWM-BRM, SPI, SCI, I²C, CAN INTERFACES

DATASHEET

- 16K ROM or 24 Kbytes EPROM/OTP/ FASTROM
- Master Reset and Power-on Reset
- Low consumption resonator main oscillator
- 4 Power saving modes
- Nested interrupt controller
- NMI dedicated non maskable interrupt pin
- 31 multifunctional bidirectional I/O lines with:
  - external interrupt capability (5 vectors)
  - 21 alternate function lines
- LCD driver with 60 segment outputs and 8 backplane outputs able to drive up to 60x8 (480) or 60x4 (240) LCD displays
- Real time base, Beep and Clock-out capabilities
- Software watchdog reset
- Two 16-bit timers with:
  - 2 input captures
  - 2 output compares
  - external clock input on one timer
  - PWM and Pulse generator modes
- 10-bit PWM (DAC) with 4 dedicated output pins
- SPI synchronous serial interface
- SCI asynchronous serial interface
- I2C multi master / slave interface
- CAN interface
- 8-bit ADC with 5 dedicated input pins



PQFP128
14 x 20

- 8-bit Data Manipulation
- 63 Basic Instructions
- 17 main Addressing Modes
- 8 x 8 Unsigned Multiply Instruction
- True Bit Manipulation

- Full hardware/software development package

Device Summary

Features	ST72589BW5	ST72389BW4
Program memory - bytes	24K OTP/FASTROM	16K ROM
RAM (stack) - bytes	1024 (256)	512 (256)
Std. Peripherals	LCD 60x8, Watchdog, 16-bit Timers, PWM-BRM, SPI, SCI, I2C, CAN, ADC	LCD 60x8, Watchdog, 16-bit Timers, SPI, SCI, ADC
Operating Supply	4.5V to 5.5V	
CPU Frequency	4 to 8 MHz (with 8 to 16 MHz oscillator)	
Temperature Range	-40°C to +85°C	
Packages	PQFP128	
Development device	ST72E589BW5	

Rev. 2.7

Table of Contents

1 GENERAL DESCRIPTION 4
1.1 INTRODUCTION 4
1.2 PIN DESCRIPTION 5
1.3 REGISTER & MEMORY MAP 8
1.4 MEMORIES AND PROGRAMMING MODES 12
2 CENTRAL PROCESSING UNIT 13
2.1 INTRODUCTION 13
2.2 MAIN FEATURES 13
2.3 CPU REGISTERS 13
3 SUPPLY, RESET AND CLOCK MANAGEMENT 16
3.1 RESET MANAGER 17
3.2 LOW CONSUMPTION OSCILLATOR 19
3.3 MAIN CLOCK CONTROLLER (MCC) 20
4 INTERRUPTS & POWER SAVING MODES 23
4.1 INTERRUPTS 23
4.2 POWER SAVING MODES 29
5 I/O PORTS 33
5.1 INTRODUCTION 33
5.2 FUNCTIONAL DESCRIPTION 33
5.3 I/O PORT IMPLEMENTATION 36
6 MISCELLANEOUS REGISTERS 39
6.1 I/O PORT INTERRUPT SENSITIVITY DESCRIPTION 39
6.2 I/O PORT ALTERNATE FUNCTIONS 39
6.3 MISCELLANEOUS REGISTERS DESCRIPTION 40
7 ON-CHIP PERIPHERALS 42
7.1 LCD DRIVER 42
7.2 WATCHDOG TIMER (WDG) 46
7.3 16-BIT TIMER 49
7.4 PWM/BRM GENERATOR (DAC) 67
7.5 SERIAL PERIPHERAL INTERFACE (SPI) 73
7.6 SERIAL COMMUNICATIONS INTERFACE (SCI) 86
7.7 I2C BUS INTERFACE (I2C) 99
7.8 CONTROLLER AREA NETWORK (CAN) 112
7.9 8-BIT A/D CONVERTER (ADC) 129
8 INSTRUCTION SET 133
8.1 CPU ADDRESSING MODES 133
8.2 INSTRUCTION GROUPS 136
9 ELECTRICAL CHARACTERISTICS 139
9.1 ABSOLUTE MAXIMUM RATINGS 139
9.2 RECOMMENDED OPERATING CONDITIONS 140
9.3 TIMING CHARACTERISTICS 140
9.4 ELECTRICAL CHARACTERISTICS 141

Table of Contents

9.5 I/O PORTS CHARACTERISTICS 142
9.6 SUPPLY, RESET AND CLOCK CHARACTERISTICS 143
9.7 MEMORY AND PERIPHERAL CHARACTERISTICS 144
10 PACKAGE CHARACTERISTICS 152
10.1 PACKAGE MECHANICAL DATA 152
11 DEVICE CONFIGURATION AND ORDERING INFORMATION 154
11.1 ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE 154
11.2 ST7 APPLICATION NOTES 156
12 SUMMARY OF CHANGES 158

1 GENERAL DESCRIPTION

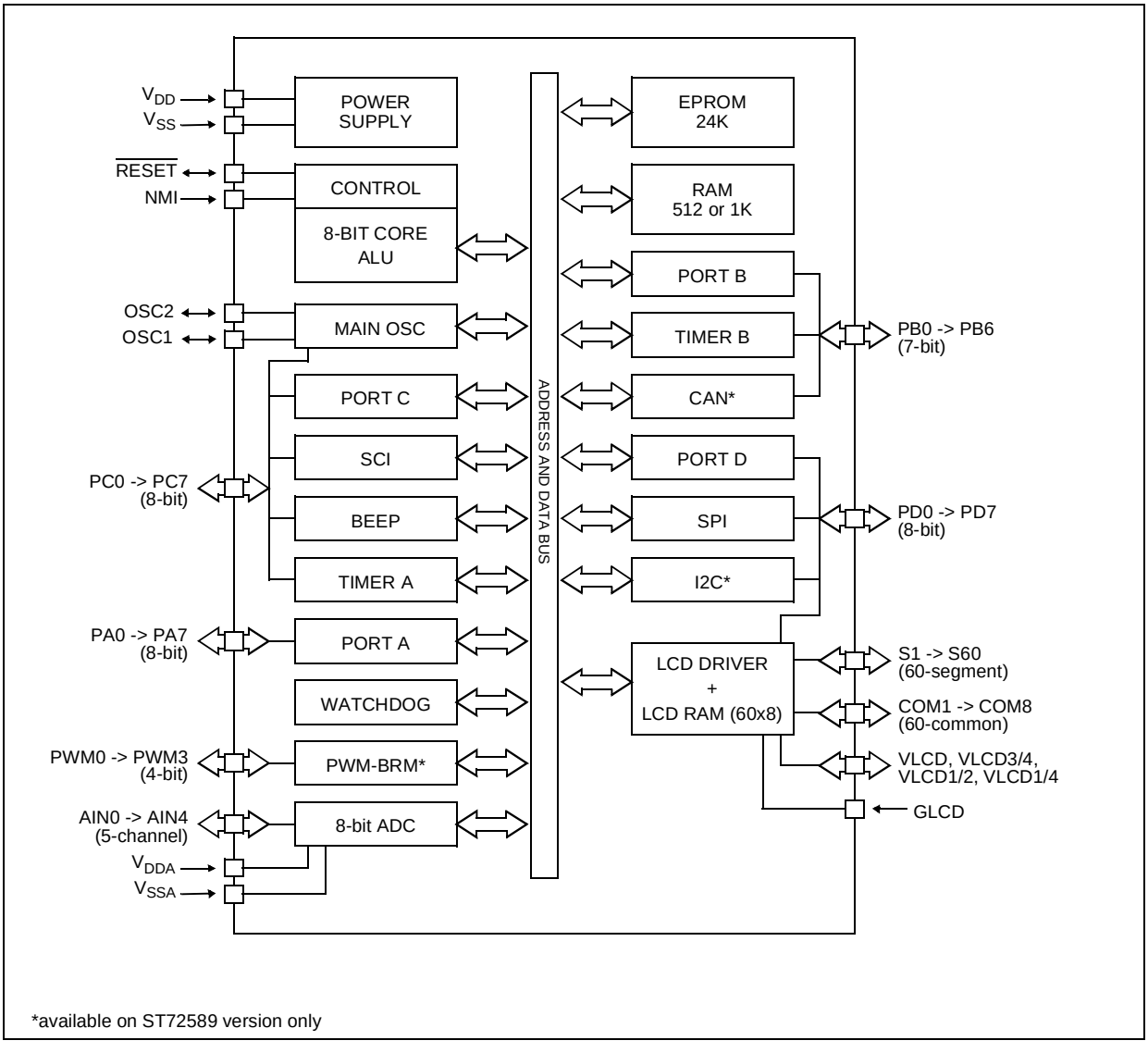
1.1 INTRODUCTION

The ST72589W and ST72389W Microcontroller Units are members of the ST7 family of Microcontrollers dedicated to high-end applications with LCD driver capability.

These devices are based on an industry-standard 8-bit core and feature an enhanced instruction set. Under software control, these microcontrollers may be placed in either WAIT, SLOW, ACTIVE-HALT or HALT modes, thus reducing power consumption.

The enhanced instruction set and addressing modes afford real programming potential. In addition to standard 8-bit data management, these microcontrollers feature true bit manipulation, 8x8 unsigned multiplication and indirect addressing modes.

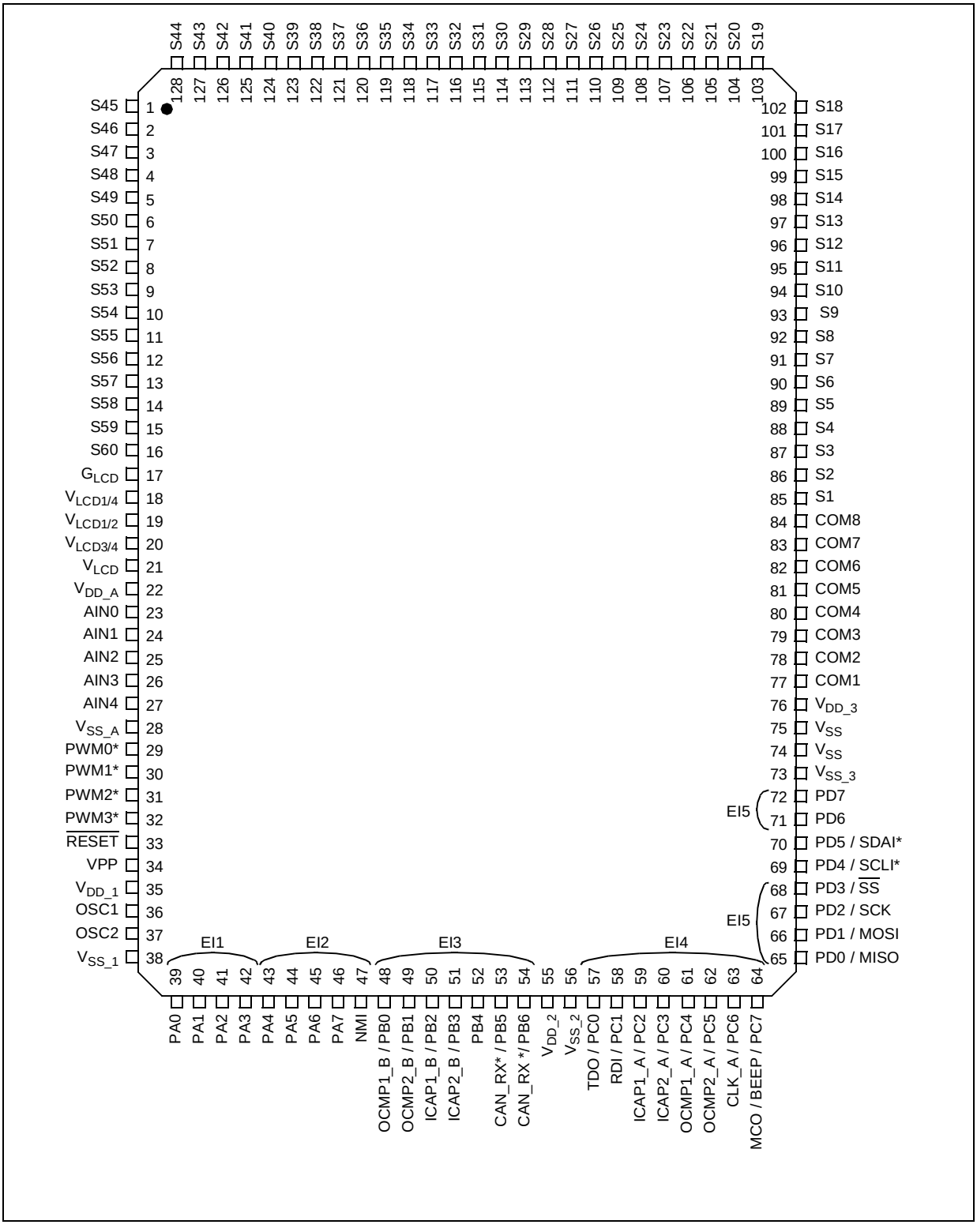
Figure 1. Device Block Diagram



*available on ST72589 version only

1.2 PIN DESCRIPTION

Figure 2. 128-Pin PQFP Package Pinout



PIN DESCRIPTION (Cont'd)

Legend / Abbreviations:

Type: I = input, O = output, S = supply, CK = Clock
Output level: LCD = V_{LCD}, $V_{LCD3/4}$, $V_{LCD1/2}$, $V_{LCD1/4}$, or G_{LCD} level.
Input level: C = CMOS $0.3V_{DD}/0.7V_{DD}$

Port configuration capabilities:

– Input: float = floating, wpu = weak pull-up, int = interrupt, ana = analog
– Output: OD = open drain, T = true open drain, PP = push-pull

Note: Reset configuration of each pin is bold.

Table 1. Device Pin Description

Pin n°	Pin Name	Type	Level		Port						Main function (after reset)	Alternate function
					Input				Output			
PQFP128			Input	Output	float	wpu	int	ana	OD	PP		
1 ... 16	S45 ... S60	O		LCD							LCD Segment Analog Outputs	
17	G_{LCD}	S									LCD Ground Reference Voltage	
18	$V_{LCD1/4}$	S									LCD Supply Reference Voltage	
19	$V_{LCD1/2}$	S										
20	$V_{LCD3/4}$	S										
21	V_{LCD}	S										
22	V_{DDA}	S									Analog Power Supply Voltage	
23	AIN0	I						X			ADC Analog Input 0	
24	AIN1	I						X			ADC Analog Input 1	
25	AIN2	I						X			ADC Analog Input 2	
26	AIN3	I						X			ADC Analog Input 3	
27	AIN4	I						X			ADC Analog Input 4	
28	V_{SSA}	S									Analog Ground Voltage	
29	PWM0* or NC	O									Pulse Width Modulator output 0*	
30	PWM1*or NC	O									Pulse Width Modulator output 1*	
31	PWM2* or NC	O									Pulse Width Modulator output 2*	
32	PWM3* or NC	O									Pulse Width Modulator output 3*	
33	$\overline{RESET}$	I/O									Top priority non maskable interrupt.	
34	V_{PP}	I									Must be tied low in user mode. In the programming mode when available, this pin acts as the programming voltage input V_{PP}.	
35	V_{DD_1}	S									Digital Main Supply Voltage	
36	OSC1	CK									These pins connect a parallel-resonant crystal or an external source to the on-chip main oscillator.	
37	OSC2	CK										
38	V_{SS_1}	S									Digital Ground Voltage	
39	PA0	I/O	C		**X**		EI1		X	X	Port A0	
40	PA1	I/O	C		**X**				X	X	Port A1	
41	PA2	I/O	C		**X**				X	X	Port A2	
42	PA3	I/O	C		**X**				X	X	Port A3	

Pin n°	Pin Name	Type	Level		Port						Main function (after reset)	Alternate function
					Input				Output			
PQFP128			Input	Output	float	wpu	int	ana	OD	PP		
43	PA4	I/O	C		X		EI2		X	X	Port A4	
44	PA5	I/O	C		X				X	X	Port A5	
45	PA6	I/O	C		X				X	X	Port A6	
46	PA7	I/O	C		X				X	X	Port A7	
47	NMI	I									No maskable interrupt input pin (floating)	
48	PB0/OCMP1_B	I/O	C		X		EI3		X	X	Port B0	Timer B Output Compare 1
49	PB1/OCMP2_B	I/O	C		X				X	X	Port B1	Timer B Output Compare 2
50	PB2/ICAP1_B	I/O	C		X				X	X	Port B2	Timer B Input Capture 1
51	PB3/ICAP2_B	I/O	C		X				X	X	Port B3	Timer B Input Capture 2
52	PB4	I/O	C		X				X	X	Port B4	
53	PB5/CANTX*	I/O	C		X				X	X	Port B5	CAN Transmit Data Output*
54	PB6/CANRX*	I/O	C		X				X	X	Port B6	CAN Receive Data Input*
55	V_{DD_2}	S									Digital Main Supply Voltage	
56	V_{SS_2}	S									Digital Ground Voltage	
57	PC0/TDO	I/O	C		X		EI4		X	X	Port C0	SCI Transmit Data Out
58	PC1/RDI	I/O	C		X				X	X	Port C1	SCI Receive Data In
59	PC2/ICAP1_A	I/O	C		X				X	X	Port B2	Timer A Input Capture 1
60	PC3/ICAP2_A	I/O	C		X				X	X	Port B3	Timer A Input Capture 2
61	PC4/OCMP1_A	I/O	C		X				X	X	Port B0	Timer A Output Compare 1
62	PC5/OCMP2_A	I/O	C		X				X	X	Port B1	Timer A Output Compare 2
63	PC6/EXTCLK_A	I/O	C		X				X	X	Port C6	Timer A External Clock
64	PC7/MCO/BEEP	I/O	C		X				X	X	Port C7	Main clock-out / Beep signal
65	PD0/MISO	I/O	C		X		EI5		X	X	Port D0	SPI Master In / Slave Out Data
66	PD1/MOSI	I/O	C		X				X	X	Port D1	SPI Master Out / Slave In Data
67	PD2/SCK	I/O	C		X				X	X	Port D2	SPI Serial Clock
68	PD3/$\overline{SS}$	I/O	C		X				X	X	Port D3	SPI Slave Select (active low)
69	PD4/SCLI*	I/O	C		X				X	X	Port D4	I2C Clock**
70	PD5/SDAI*	I/O	C		X				X	X	Port D5	I2C Data**
71	PD6	I/O	C		X		EI5		X	X	Port D6	
72	PD7	I/O	C		X				X	X	Port D7	
73	V_{SS_3}	S									Digital Ground Voltage	
74	V_{SS}	S									Ground Voltage	
75	V_{SS}	S									Ground Voltage	
76	V_{DD_3}	S									Digital Main Supply Voltage	
77 to 84	COM1 to COM8	O	C	LCD							LCD Common (backplane) analog output	
85 to 128	S1 to S44	O		LCD							LCD Segment Analog Outputs	

* available on ST72589 version only.

** available on ST72589 version only. Port D4 and D5 in open-drain output only for ST72589.

1.3 REGISTER & MEMORY MAP

As shown in the Figure 3, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register location, up to 1Kbyte of RAM, 60 bytes of LCD RAM and up to 24Kbytes of user program memory. The RAM space includes up to 256 bytes for the stack from 0100h to 01FFh.

The highest address bytes contain the user reset and interrupt vectors.

Figure 3. Memory Map

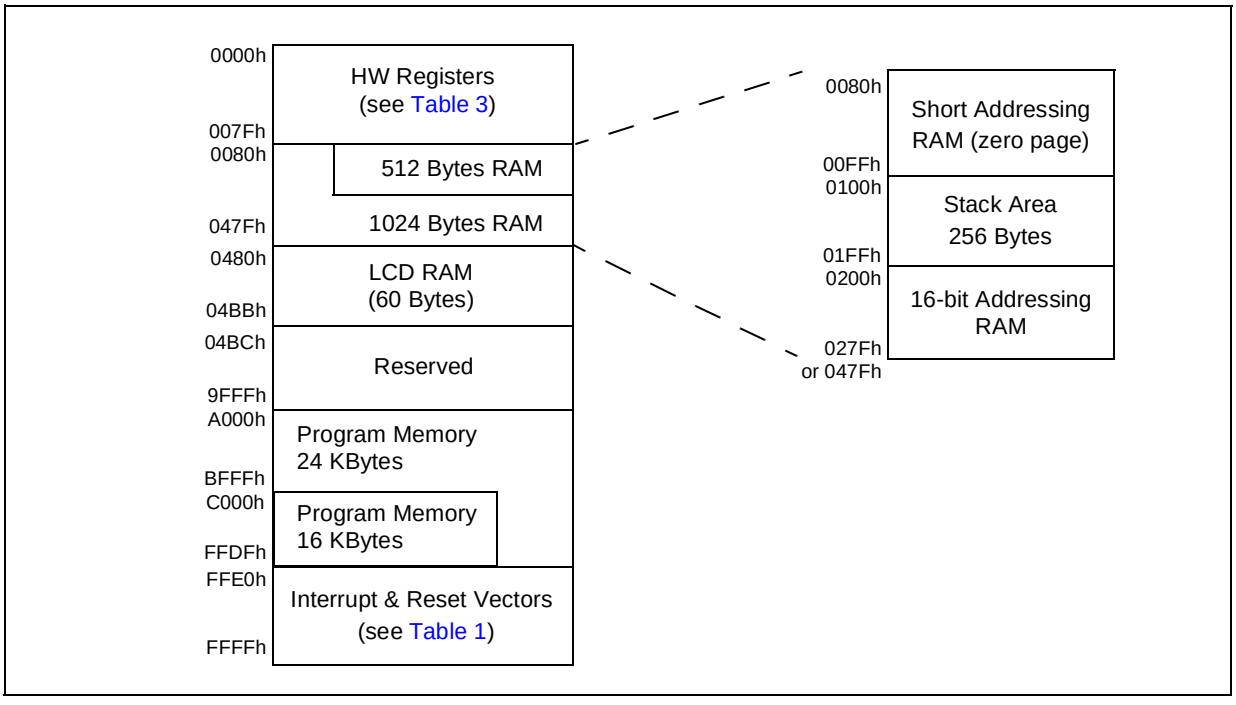


Table 2. Interrupt Vector Map

Vector Address	Description	Remarks
FFE0-FFE1h	I2C interrupt vector*	Internal Interrupt
FFE2-FFE3h	SCI interrupt vector	
FFE4-FFE5h	TIMER B interrupt vector	
FFE6-FFE7h	TIMER A interrupt vector	
FFE8-FFE9h	SPI interrupt vector	
FFEA-FFEBh	CAN interrupt vector*	
FFEC-FFEDh	Not used	
FFEE-FFEFh	MCC interrupt vector	
FFF0-FFF1h	External interrupt vector (EI5: port D)	External Interrupt
FFF2-FFF3h	External interrupt vector (EI4: port C)	
FFF4-FFF5h	External interrupt vector (EI3: port B)	
FFF6-FFF7h	External interrupt vector (EI2: port A7..4)	
FFF8-FFF9h	External interrupt vector (EI1: port A3..0)	
FFFA-FFFBh	Non maskable external interrupt vector (NMI)	
FFFC-FFFDh	TRAP (software) interrupt vector	CPU Interrupt
FFFE-FFFFh	RESET vector	

* available on ST72589 version only.

Table 3. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h 0001h 0002h	Port A	PADR PADDR PAOR	Port A Data Register Port A Data Direction Register Port A Option Register	00h 00h 00h	R/W R/W R/W
0003h	Reserved Area (1 Byte)				
0004h 0005h 0006h	Port B	PBDR PBDDR PBOR	Port B Data Register Port B Data Direction Register Port B Option Register	00h 00h 00h	R/W R/W R/W.
0007h	Reserved Area (1 Byte)				
0008h 0009h 000Ah	Port C	PCDR PCDDR PCOR	Port C Data Register Port C Data Direction Register Port C Option Register	00h 00h 00h	R/W R/W R/W
000Bh	Reserved Area (1 Byte)				
000Ch 000Dh 000Eh	Port D	PDDR PDDDR PDOR	Port D Data Register Port D Data Direction Register Port D Option Register	00h 00h 00h	R/W R/W R/W
000Fh to 001Bh	Reserved Area (13 Bytes)				
001Ch 001Dh 001Eh 001Fh	ITC	ISPR0 ISPR1 ISPR2 ISPR3	Interrupt Software Priority Register 0 Interrupt Software Priority Register 1 Interrupt Software Priority Register 2 Interrupt Software Priority Register 3	FFh FFh FFh FFh	R/W R/W R/W R/W
0020h		MISCR1	Miscellaneous Register 1	00h	R/W
0021h 0022h 0023h	SPI	SPIDR SPICR SPISR	SPI Data I/O Register SPI Control Register SPI Status Register	xxh 0xh 00h	R/W R/W Read Only
0024h	WATCHDOG	WDGCR	Watchdog Control Register	7Fh	R/W
0025h	Reserved Area (1 Byte)				
0026h	MCC	MCCSR	Main Clock Control / Status Register	00h	R/W
0027h	Reserved Area (1 Byte)				
0028h 0029h 002Ah 002Bh 002Ch 002Dh 002Eh	I^2C*	I2CCR I2CSR1 I2CSR2 I2CCCR I2COAR1 I2COAR2 I2CDR	I^2C Control Register I^2C Status Register 1 I^2C Status Register 2 I^2C Clock Control Register I^2C Own Address Register 1 I^2C Own Address Register 2 I^2C Data Register	00h 00h 00h 00h 00h 00h 00h	R/W Read Only Read Only R/W R/W R/W R/W

Address	Block	Register Label	Register Name	Reset Status	Remarks
002Fh 0030h	Reserved Area (2 Bytes)				
0031h	TIMER A	TACR2	Timer A Control Register 2	00h	R/W
0032h		TACR1	Timer A Control Register 1	00h	R/W
0033h		TASR	Timer A Status Register	xxh	Read Only
0034h		TAIC1HR	Timer A Input Capture 1 High Register	xxh	Read Only
0035h		TAIC1LR	Timer A Input Capture 1 Low Register	xxh	Read Only
0036h		TAOC1HR	Timer A Output Compare 1 High Register	80h	R/W
0037h		TAOC1LR	Timer A Output Compare 1 Low Register	00h	R/W
0038h		TACHR	Timer A Counter High Register	FFh	Read Only
0039h		TACLR	Timer A Counter Low Register	FCh	Read Only
003Ah		TAACHR	Timer A Alternate Counter High Register	FFh	Read Only
003Bh		TAACLR	Timer A Alternate Counter Low Register	FCh	Read Only
003Ch		TAIC2HR	Timer A Input Capture 2 High Register	xxh	Read Only
003Dh		TAIC2LR	Timer A Input Capture 2 Low Register	xxh	Read Only
003Eh		TAOC2HR	Timer A Output Compare 2 High Register	80h	R/W
003Fh		TAOC2LR	Timer A Output Compare 2 Low Register	00h	R/W
0040h		MISCR2	Miscellaneous Register 2	00h	R/W
0041h	TIMER B	TBCR2	Timer B Control Register 2	00h	R/W
0042h		TBCR1	Timer B Control Register 1	00h	R/W
0043h		TBSR	Timer B Status Register	xxh	Read Only
0044h		TBIC1HR	Timer B Input Capture 1 High Register	xxh	Read Only
0045h		TBIC1LR	Timer B Input Capture 1 Low Register	xxh	Read Only
0046h		TBOC1HR	Timer B Output Compare 1 High Register	80h	R/W
0047h		TBOC1LR	Timer B Output Compare 1 Low Register	00h	R/W
0048h		TBCHR	Timer B Counter High Register	FFh	Read Only
0049h		TBCLR	Timer B Counter Low Register	FCh	Read Only
004Ah		TBACHR	Timer B Alternate Counter High Register	FFh	Read Only
004Bh		TBACLR	Timer B Alternate Counter Low Register	FCh	Read Only
004Ch		TBIC2HR	Timer B Input Capture 2 High Register	xxh	Read Only
004Dh		TBIC2LR	Timer B Input Capture 2 Low Register	xxh	Read Only
004Eh		TBOC2HR	Timer B Output Compare 2 High Register	80h	R/W
004Fh		TBOC2LR	Timer B Output Compare 2 Low Register	00h	R/W
0050h	SCI	SCISR	SCI Status Register	C0h	Read Only
0051h		SCIDR	SCI Data Register	xxh	R/W
0052h		SCIBRR	SCI Baud Rate Register	00xx xxxx	R/W
0053h		SCICR1	SCI Control Register 1	xxh	R/W
0054h		SCICR2	SCI Control Register 2	00h	R/W
0055h		SCIERPR	SCI Extended Receive Prescaler Register	00h	R/W
0056h			Reserved area	---	
0057h		SCIETPR	SCI Extended Transmit Prescaler Register	00h	R/W
0058h	LCD	LCDCR	LCD Control Register	00h	R/W
0059h	Reserved Area (1 Byte)				

Address	Block	Register Label	Register Name	Reset Status	Remarks
005Ah 005Bh 005Ch 005Dh 005Eh 005Fh 0060h to 006Fh	CAN*	CANISR CANICR CANCSR CANBRPR CANBTR CANPSR	CAN Interrupt Status Register CAN Interrupt Control Register CAN Control / Status Register CAN Baud Rate Prescaler Register CAN Bit Timing Register CAN Page Selection Register First address to Last address of CAN page X	00h 00h 00h 00h 23h 00h --	R/W R/W R/W R/W R/W R/W See CAN Description
0070h 0071h	ADC	ADCDR ADCCSR	Data Register Control/Status Register	xxh 00h	Read Only R/W
0072h 0073h	Reserved Area (2 Bytes)				
0074h 0075h 0076h 0077h 0078h 0079h	PWMBRM*	PWM0 BRM10 PWM1 PWM2 BRM32 PWM3	10-bit PWM / BRM Registers	00h 00h 00h 00h 00h 00h	R/W R/W R/W R/W R/W R/W

*** Note**: available on ST72589 version only.

1.4 MEMORIES AND PROGRAMMING MODES

1.4.1 EPROM Program Memory

The program memory of the OTP and EPROM devices can be programmed with EPROM programming tools available from STMicroelectronics

EPROM Erasure

EPROM devices are erased by exposure to high intensity UV light admitted through the transparent window. This exposure discharges the floating gate to its initial state through induced photo current.

It is recommended that the EPROM devices be kept out of direct sunlight, since the UV content of sunlight can be sufficient to cause functional failure. Extended exposure to room level fluorescent lighting may also cause erasure.

An opaque coating (paint, tape, label, etc...) should be placed over the package window if the product is to be operated under these lighting conditions. Covering the window also reduces I_{DD} in power-saving modes due to photo-diode leakage currents.

2 CENTRAL PROCESSING UNIT

2.1 INTRODUCTION

This CPU has a full 8-bit architecture and contains six internal registers allowing efficient 8-bit data manipulation.

2.2 MAIN FEATURES

- Enable executing 63 basic instructions
- Fast 8-bit by 8-bit multiply
- 17 main addressing modes (with indirect addressing mode)
- Two 8-bit index registers
- 16-bit stack pointer
- Low power HALT and WAIT modes
- Priority maskable hardware interrupts
- Non-maskable software/hardware interrupts

2.3 CPU REGISTERS

The 6 CPU registers shown in Figure 4 are not present in the memory mapping and are accessed by specific instructions.

Accumulator (A)

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

Index Registers (X and Y)

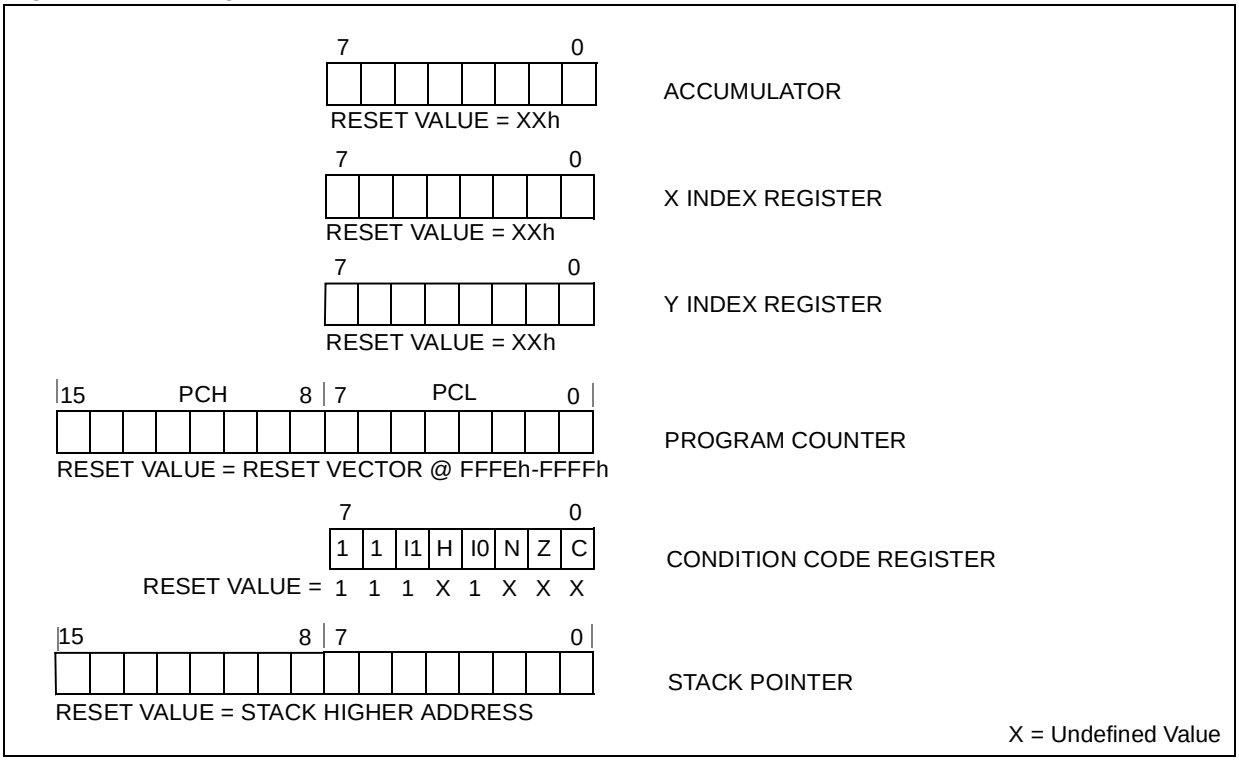
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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

Program Counter (PC)

The program counter is a 16-bit register containing the address of the next instruction to be executed by the CPU. It is made of two 8-bit registers PCL (Program Counter Low which is the LSB) and PCH (Program Counter High which is the MSB).

Figure 4. CPU Registers



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 (i.e. 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.

CENTRAL PROCESSING UNIT (Cont'd)

Stack Pointer (SP)

Read/Write

Reset Value: 01 FFh

15							8
0	0	0	0	0	0	0	1

7							0
SP7	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 5).

Since the stack is 256 bytes deep, the 8 most significant bits are forced by hardware. Following an MCU Reset, or after a Reset Stack Pointer instruction (RSP), the Stack Pointer contains its reset value (the SP7 to SP0 bits are set) which is the stack higher address.

The least significant byte of the Stack Pointer (called S) can be directly accessed by a LD instruction.

Note: When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

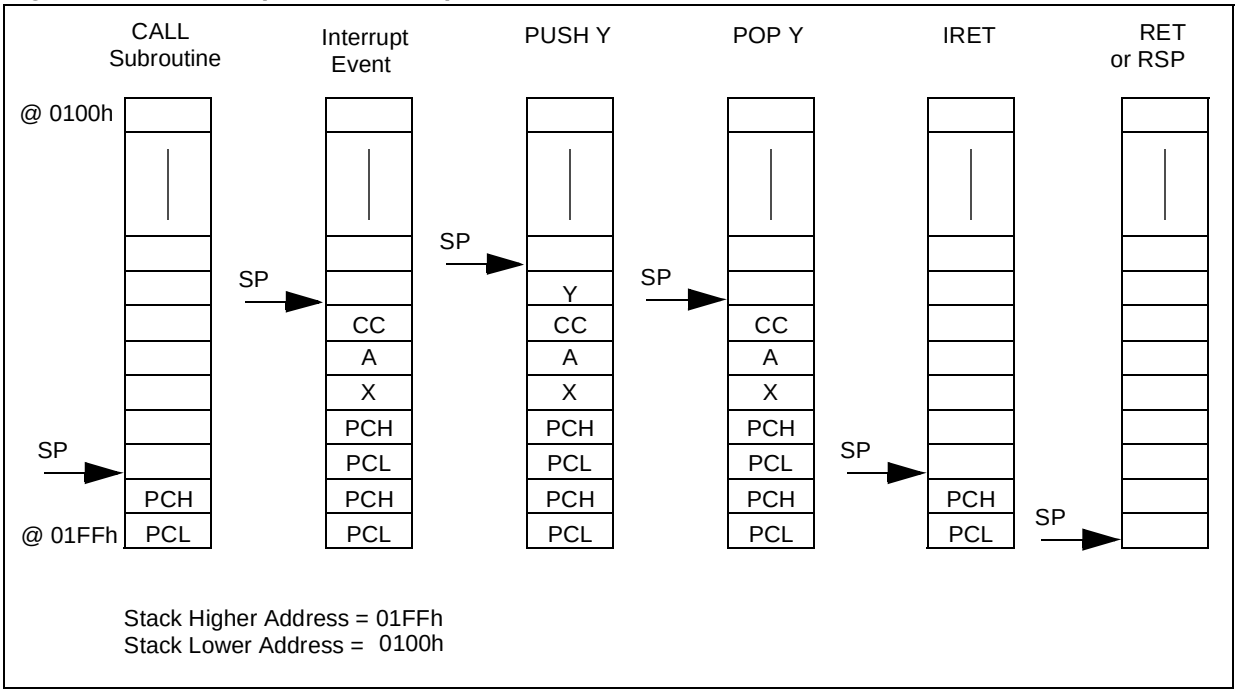
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 5.

– When an interrupt is received, the SP is decremented and the context is pushed on the stack.

– On return from interrupt, the SP is incremented and the context is popped from the stack.

A subroutine call occupies two locations and an interrupt five locations in the stack area.

Figure 5. Stack Manipulation Example

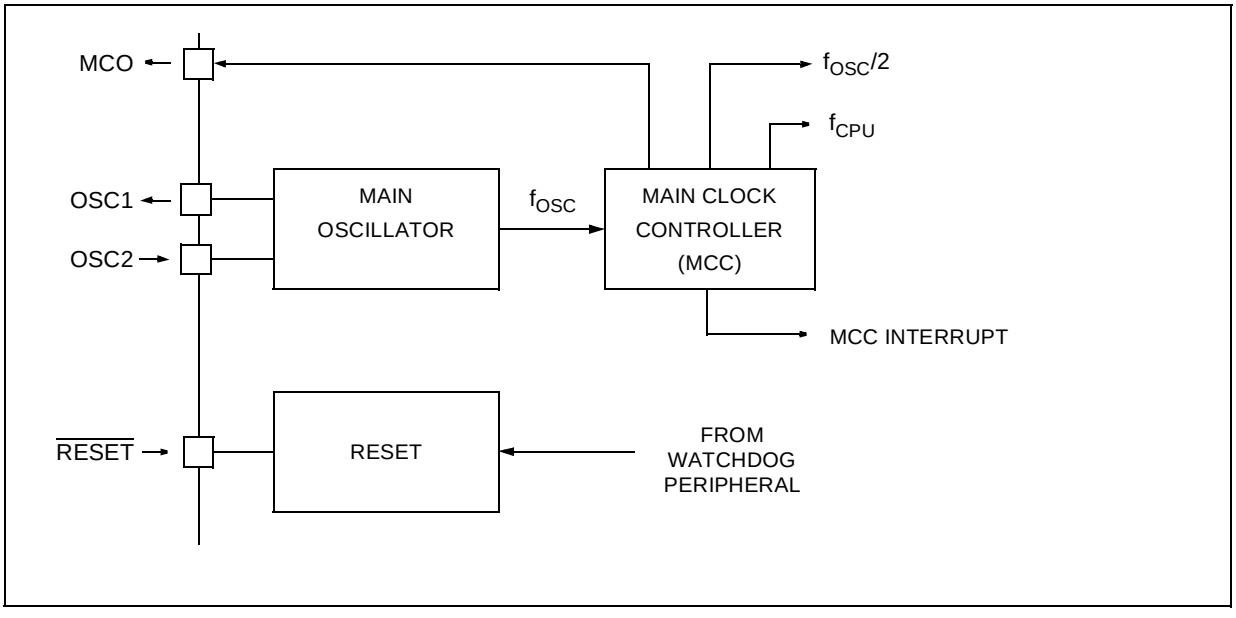


3 SUPPLY, RESET AND CLOCK MANAGEMENT

This chapter describes the following generic features to guaranty the ST7 correct operation. An overview is shown in Figure 6.

- RESET Manager
- Low Consumption Crystal Oscillators
- Main Clock controller (MCC)

Figure 6. Clock and RESET Management Overview



3.1 RESET MANAGER

3.1.1 Introduction

There are three sources of Reset:

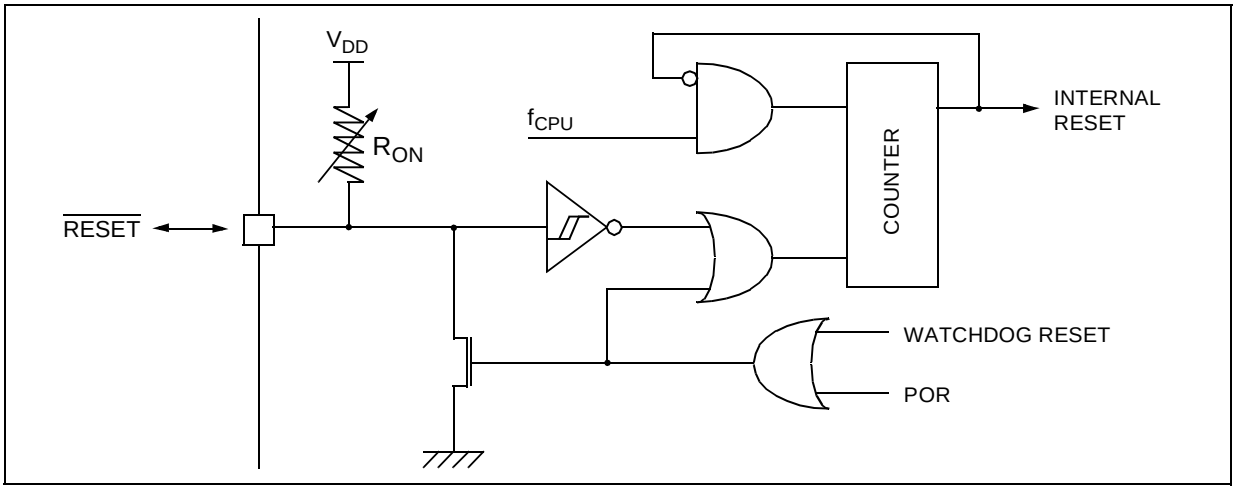
– $\overline{\text{RESET}}$ pin (external source)

– Power-On Reset (internal source)

– WATCHDOG (internal source)

The Reset Service Routine vector is located at address FFFEh-FFFFh.

Figure 7. Reset Block Diagram



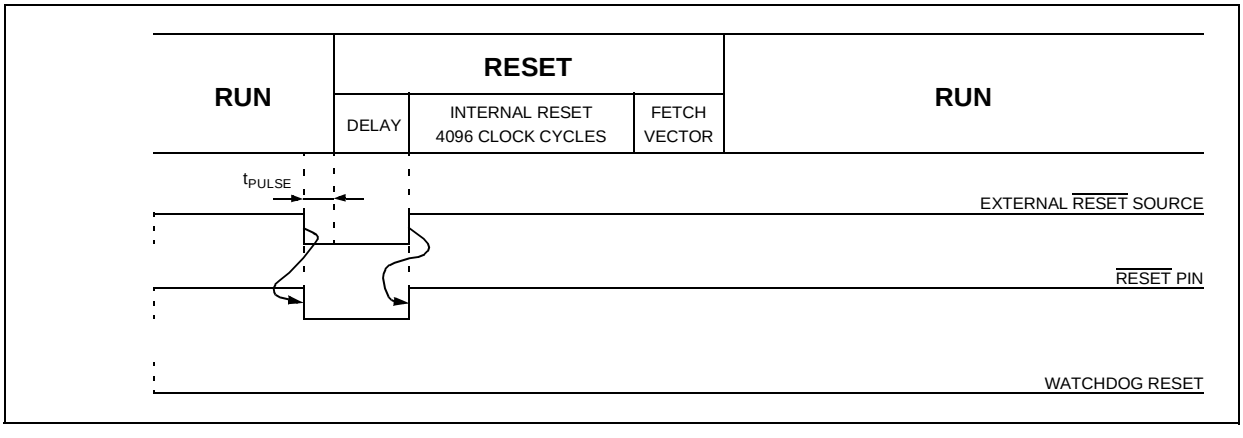
3.1.2 External Reset

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor (see Figure 7). This pull-up has not a fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device.

A RESET signal originating from an external source must have a duration of at least t_{PULSE} in order to be recognized. The RESET sequence associated to this RESET source is shown in Figure 8.

When the RESET is generated by a internal source, during the two first phases of the RESET sequence, the device $\overline{\text{RESET}}$ pin acts as an output that is pulled low.

Figure 8. External RESET Sequences

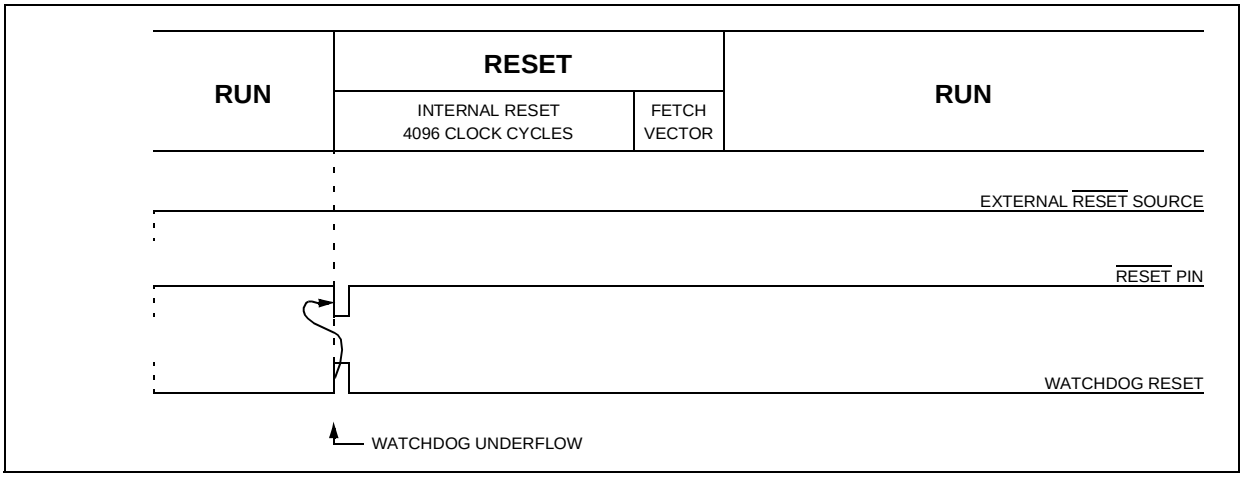


RESET MANAGER (Cont'd)

3.1.3 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter underflow is reduced to 2 phases (see Figure 9).

Figure 9. Watchdog RESET Sequence



3.1.4 Reset Operation

The duration of the Reset condition, which is also reflected on the output pin, is fixed at 4096 internal CPU Clock cycles. A Reset signal originating from an external source must have a duration of at least 1.5 internal CPU Clock cycles in order to be recognised. At the end of the Power-On Reset cycle, the MCU may be held in the Reset condition by an External Reset signal. The RESET pin may thus be used to ensure V_{DD} has risen to a point where the MCU can operate correctly before the User program is run. Following a Reset event, or after exiting Halt mode, a 4096 CPU Clock cycle delay period is initiated in order to allow the oscillator to stabilise and to ensure that recovery has taken place from the Reset state.

During the Reset cycle, the device Reset pin acts as an output that is pulsed low. In its high state, an internal pull-up resistor is connected to the Reset pin. This resistor can be pulled low by external circuitry to reset the device.

3.1.5 Power-on Reset

This circuit detects the ramping up of V_{DD}, and generates a pulse that is used to reset the application at V_{POR} supply voltage.

Power-On Reset is designed exclusively to cope with power-up conditions, and should not be used in order to attempt to detect a drop in the power supply voltage.

Caution: to re-initialize the Power-On Reset, the power supply voltage must fall below V_{TN}, prior to rise above V_{POR}. If this condition is not respected, on subsequent power-up the Reset pulse may not be generated. An external Reset pulse may be required to correctly reactivate the circuit.

3.2 LOW CONSUMPTION OSCILLATOR

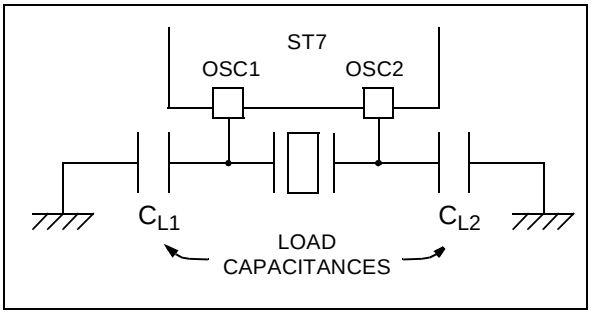
The oscillator of the ST72589 and ST72389 devices is a Crystal/Ceramic Resonator Oscillator. Its architecture is based on a constant current to minimize the consumption. It can be used either with an external resonator or an external source.

This oscillator allows a high accuracy to supply the clock for the ST7 CPU and its internal peripherals.

Using a Crystal/Ceramic Resonator

The resonator and the load capacitances have to be connected as shown in Figure 10 and have to be mounted as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time.

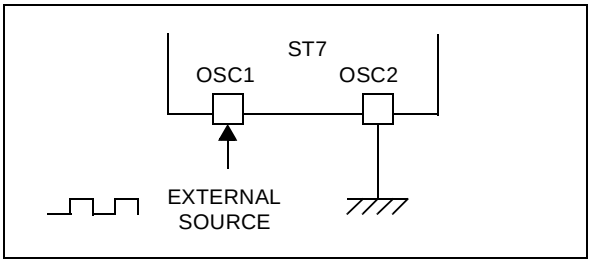
Figure 10. Main Crystal/Ceramic Resonator



Using an External Clock Source

In this mode, a square clock signal with ~50% duty cycle has to drive the OSC1 pin while the OSC2 pin is tied to ground (see Figure 11).

Figure 11. Main External Clock Source



3.3 MAIN CLOCK CONTROLLER (MCC)

The MCC block supplies the clock for the ST7 CPU and its internal peripherals. It allows to manage the power saving modes such as the SLOW and ACTIVE-HALT modes. The whole functionality is managed by the Main Clock Control/Status Register (MCCSR) and the Miscellaneous Register 2 (MISCR2).

The MCC block described in Figure 12 consists of:

– a programmable CPU clock prescaler

– a time base counter with interrupt capability

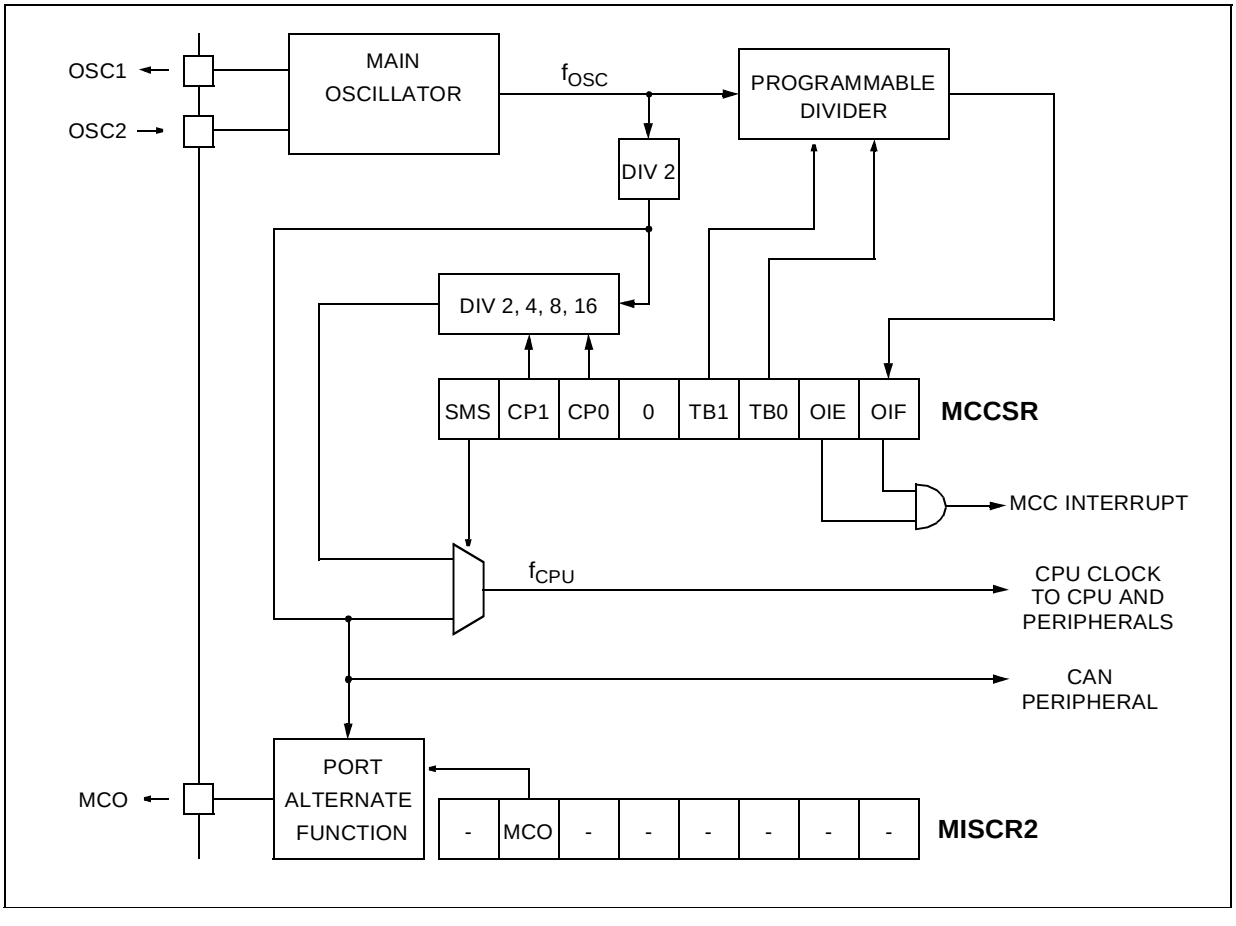
– a clock-out signal to supply external devices

The prescaler allows to select the main clock frequency and is controlled with three bits of the MCCSR: CP1, CP0 and SMS.

The counter allows to generate an interrupt based on a accurate real time clock. Four different time bases depending directly on f_{OSC} are available. The whole functionality is controlled by four bits of the MCCSR register: TB1, TB0, OIE and OIF.

The clock-out capability allows to configure a dedicated I/O port pin as an $f_{OSC}/2$ clock out to drive external devices. It is controlled by a bit in the MISCR2 register: MCO.

Figure 12. Main Clock Controller (MCC) Block Diagram



MAIN CLOCK CONTROLLER (Cont'd)

MISCELLANEOUS REGISTER 2 (MISCR2)

See description in MISCELLANEOUS Register Section.

MAIN CLOCK CONTROL/STATUS REGISTER (MCCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
SMS	CP1	CP0	0	TB1	TB0	OIE	OIF

Bit 0 = **SMS** *Slow mode select*
This bit is set and cleared by software.
0: Normal mode. $f_{CPU} = f_{OSC} / 2$
1: Slow mode. f_{CPU} is given by CP1, CP0
See low power consumption mode and MCC chapters for more details.

Bit 2:1 = **CP1-CP0** *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
$f_{OSC} / 4$	0	0
$f_{OSC} / 8$	0	1
$f_{OSC} / 16$	1	0
$f_{OSC} / 32$	1	1

Bit 4 = **Reserved**, always read as 0.

Bit 3:2 = **TB1-TB0** *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_{OSC}=8MHz	f_{OSC}=16MHz		
32000	4ms	2ms	0	0
64000	8ms	4ms	0	1
160000	20ms	10ms	1	0
400000	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 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 disable
1: Oscillator interrupt enable
This interrupt allows to exit from ACTIVE-HALT mode. When this bit is set, calling the ST7 software HALT instruction accesses the ACTIVE-HALT power saving mode.

Bit 0 = **OIF** *Oscillator interrupt flag*

This bit is set by hardware and cleared by software reading the CSR register. It indicates when set that the main oscillator has measured the selected elapsed time (TB1:0).
0: timeout not reached
1: timeout reached

Warning: BRES and BSET instructions must not be used on the MCCSR register to avoid unwanted clearing of OIF bit.

MAIN CLOCK CONTROLLER (Cont'd)

Table 4. Main Clock Controller Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0026h	**MCCSR**	SMS	CP1	CP0		TB1	TB0	OIE	OIF
	Reset Value	0	0	0	0	0	0	0	0

4 INTERRUPTS & POWER SAVING MODES

4.1 INTERRUPTS

4.1.1 Introduction

The ST7 enhanced interrupt management provides the following features:

- Hardware interrupts
- Software interrupt (TRAP)
- Nested or concurrent interrupt management with flexible interrupt priority and level management:
  - Up to 4 software programmable nesting levels
  - Up to 16 interrupt vectors fixed by hardware
  - 3 non maskable events: NMI, RESET, TRAP

This interrupt management is based on:

– Bit 5 and bit 3 of the CPU CC register (I1:0),

– Interrupt software priority registers (ISPRx),

– Fixed interrupt vector addresses located at the high addresses of the memory map (FFE0h to FFFFh) sorted by hardware priority order.

This enhanced interrupt controller guarantees full upward compatibility with the standard (not nested) ST7 interrupt controller.

4.1.2 Interrupt Masking and Processing Flow

The interrupt masking is managed by the I1 and I0 bits of the CC register and the ISPRx registers which give the interrupt software priority level of each interrupt vector (see Table 5). The processing flow is shown in Figure 13

When an interrupt request has to be serviced:

– Normal processing is suspended at the end of the current instruction execution.

– The PC, X, A and CC registers are saved onto the stack.

– I1 and I0 bits of CC register are set according to the corresponding values in the ISPRx registers of the serviced interrupt vector.

– The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to “Interrupt Mapping” table for vector addresses).

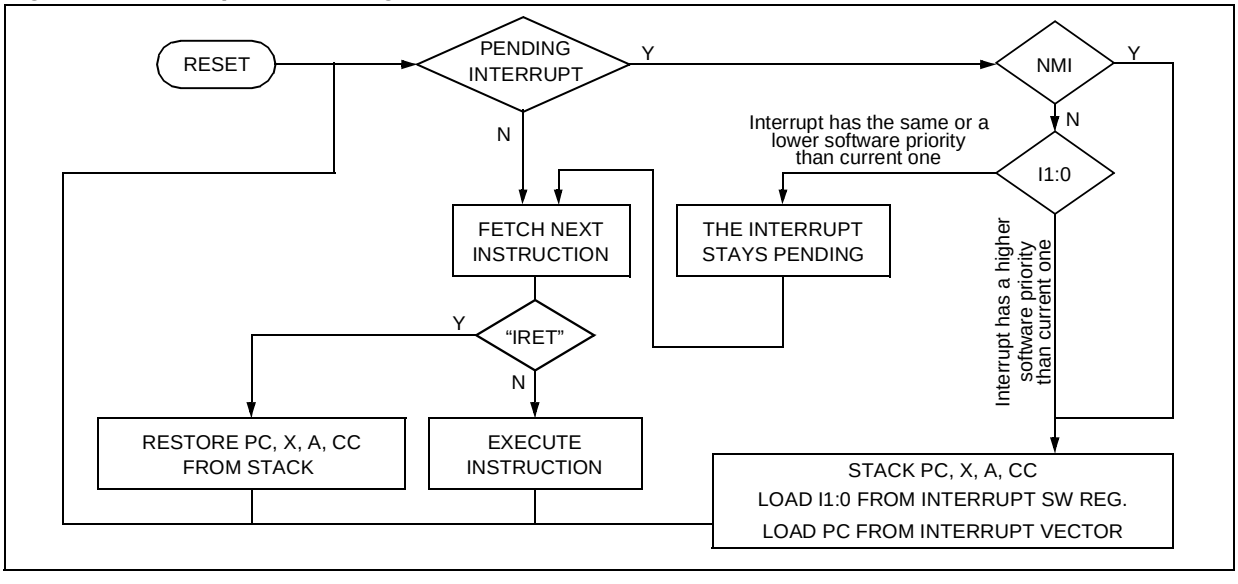
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

Note: As a consequence of the IRET instruction, the I1 and I0 bits will be restored from the stack and the program in the previous level will resume.

Table 5. Interrupt Software Priority Levels

Interrupt software priority	Level	I1	I0
Level 0 (main)	Low	1	0
Level 1	▼	0	1
Level 2		0	0
Level 3 (= interrupt disable)	High	1	1

Figure 13. Interrupt Processing Flowchart



INTERRUPTS (Cont'd)

Servicing Pending Interrupts

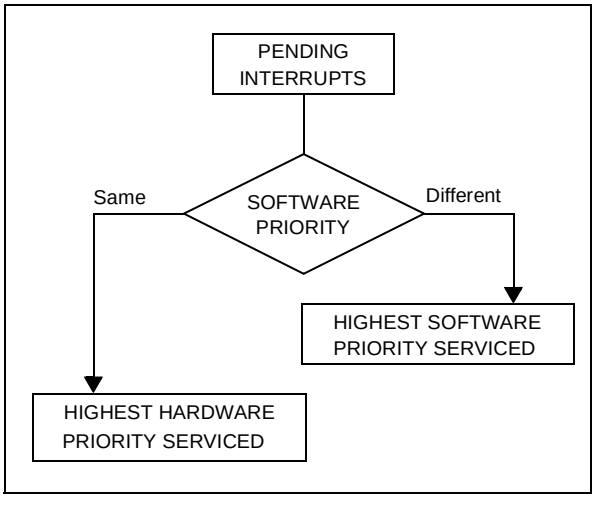
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– the highest software priority interrupt is serviced,

– if several interrupts have the same software priority then the interrupt with the highest hardware priority is serviced first.

Figure 14 describes this decision process.

Figure 14. Priority Decision Process



When an interrupt request is not serviced immediately, it is latched and then processed when its software priority combined with the hardware priority becomes the highest one.

Note 1: The hardware priority is exclusive while the software one is not. This allows the previous process to succeed with only one interrupt.
Note 2: RESET, TRAP and NMI are non maskable and they can be considered as having the highest software priority in the decision process.

Different Interrupt Vector Sources

Two interrupt source types are managed by the ST7 interrupt controller: the non-maskable type (RESET, NMI, TRAP) and the maskable type (external or from internal peripherals).

Non-Maskable Sources

These sources are processed regardless of the state of the I1 and I0 bits of the CC register (see Figure 13). After stacking the PC, X, A and CC registers (except for RESET), the corresponding vector is loaded in the PC register and the I1 and I0 bits of the CC are set to disable interrupts (level 3). These sources allow the processor to exit HALT mode.

- NMI (Non Maskable Hardware Interrupt)

This hardware interrupt occurs when a specific edge is detected on the dedicated NMI pin. Its detailed specification is given in the Miscellaneous register chapter.

- TRAP (Non Maskable Software Interrupt)

This software interrupt is serviced when the TRAP instruction is executed. It will be serviced according to the flowchart on Figure 13 as an NMI.

- RESET

The RESET source has the highest priority in the ST7. This means that the first current routine has the highest software priority (level 3) and the highest hardware priority.
See the RESET chapter for more details.

Maskable Sources

Maskable interrupt vector sources can be serviced if the corresponding interrupt is enabled and if its own interrupt software priority (in ISPRx registers) is higher than the one currently being serviced (I1 and I0 in CC register). If any of these two conditions is false, the interrupt is latched and thus remains pending.

- External Interrupts

External interrupts allow the processor to exit from HALT low power mode.
External interrupt sensitivity is software selectable through the Miscellaneous registers (MISCRx).
External interrupt triggered on edge will be latched and the interrupt request automatically cleared upon entering the interrupt service routine.
If several input pins of a group connected to the same interrupt line are selected simultaneously, these will be logically ORed.

- Peripheral Interrupts

Usually the peripheral interrupts cause the MCU to exit from HALT mode except those mentioned in the "Interrupt Mapping" table.
A peripheral interrupt occurs when a specific flag is set in the peripheral status registers and if the corresponding enable bit is set in the peripheral control register.
The general sequence for clearing an interrupt is based on an access to the status register followed by a read or write to an associated register.
Note: The clearing sequence resets the internal latch. A pending interrupt (i.e. waiting for being serviced) will therefore be lost if the clear sequence is executed.

INTERRUPTS (Cont'd)

4.1.3 Interrupts and Low Power Modes

All interrupts allow the processor to exit the WAIT low power mode. On the contrary, only external and other specified interrupts allow the processor to exit the HALT modes (see column "Exit from HALT" in "Interrupt Mapping" table). When several pending interrupts are present while exiting HALT mode, the first one serviced can only be an interrupt with exit from HALT mode capability and it is selected through the same decision process shown in Figure 14

Note: If an interrupt, that is not able to Exit from HALT mode, is pending with the highest priority when exiting HALT mode, this interrupt is serviced after the first one serviced.

4.1.4 Concurrent and Nested Interrupt Management

The following Figure 15 and Figure 16 show two different interrupt management modes. The first is called concurrent mode and does not allow an interrupt to be interrupted, unlike the nested mode in Figure 16 The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0, NMI. The software priority is given for each interrupt.

Warning: A stack overflow may occur without notifying the software of the failure.

Figure 15. Concurrent interrupt management

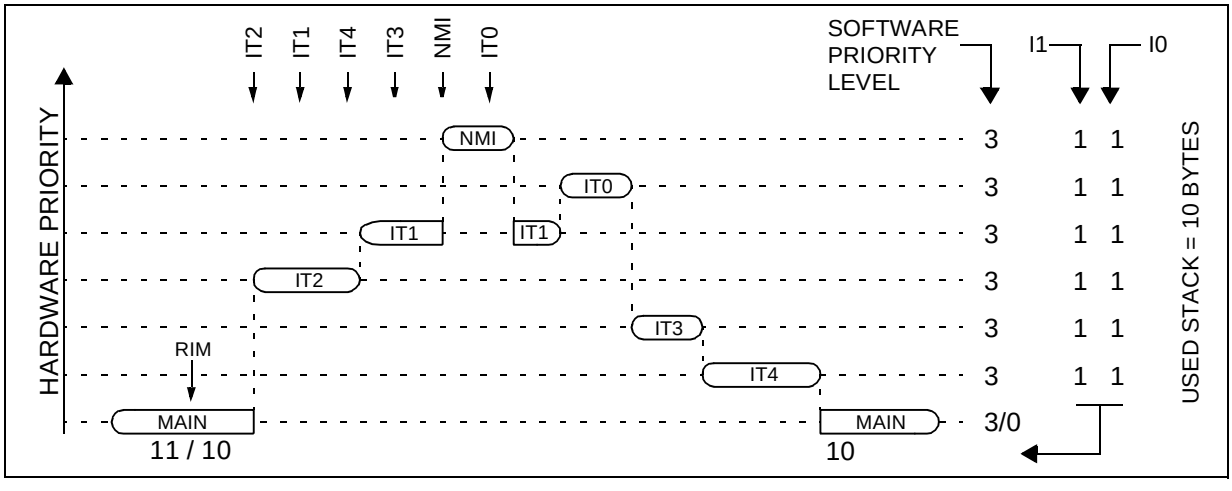
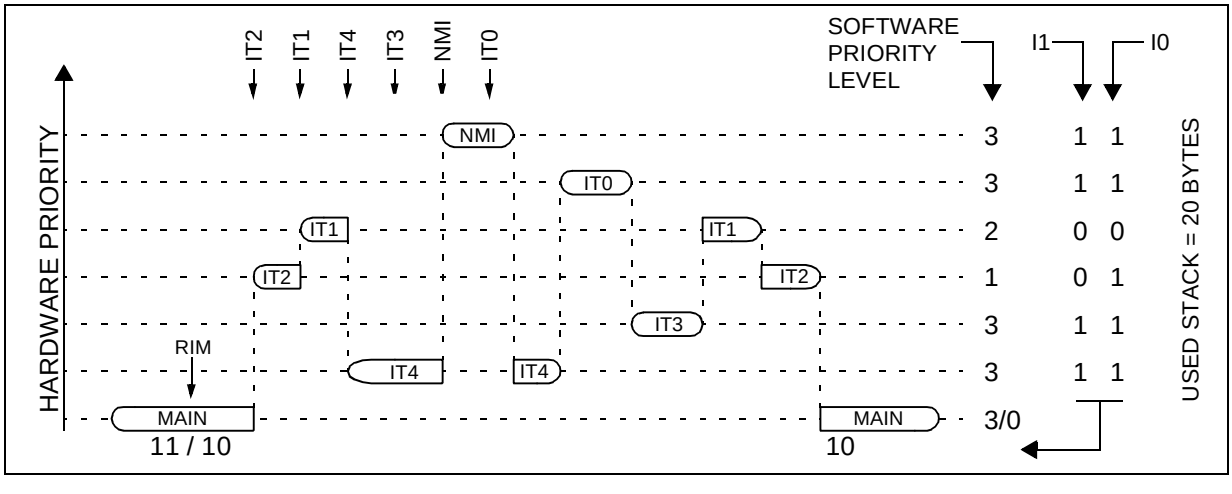


Figure 16. Nested interrupt management



INTERRUPTS (Cont'd)

4.1.5 Interrupt Register Description

CPU CC REGISTER INTERRUPT BITS

Read/Write

Reset Value: 111x 1010 (xAh)

7							0
1	1	**I1**	H	**I0**	N	Z	C

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

These two bits indicate the current interrupt software priority.

Interrupt Software Priority	Level	I1	I0
Level 0 (main)	Low	1	0
Level 1	↓	0	1
Level 2	↓	0	0
Level 3 (= interrupt disable*)	High	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 (ISPRx).

They can be also set/cleared by software with the RIM, SIM, HALT, WFI, IRET and PUSH/POP instructions (see "Interrupt Dedicated Instruction Set" table).

***Note**: NMI, TRAP and RESET events are non maskable sources and can interrupt a level 3 program.

INTERRUPT SOFTWARE PRIORITY REGISTERS (ISPRX)

Read/Write (bit 7:4 of **ISPR3** are read only)

Reset Values: 1111 1111 (FFh)

	7							0
ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1	I1_0	I0_0
ISPR1	I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4	I0_4
ISPR2	I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8	I0_8
ISPR3	1	1	1	1	I1_13	I0_13	I1_12	I0_12

These four registers contain the interrupt software priority of each interrupt vector.

– Each interrupt vector (except RESET and TRAP) has corresponding bits in these registers where its own software priority is stored. This correspondence is shown in the following table.

Vector address	ISPRx bits
FFFBh-FFFAh	I1_0 and I0_0 bits*
FFF9h-FFF8h	I1_1 and I0_1 bits
...	...
FFE1h-FFE0h	I1_13 and I0_13 bits

– Each I1_x and I0_x bit value in the ISPRx registers has the same meaning as the I1 and I0 bits in the CC register.

– Level 0 can not be written (I1_x=1, I0_x=0). In this case, the previously stored value is kept. (example: previous=CFh, write=64h, result=44h)

The RESET, TRAP and NMI vectors have no software priorities. When one is serviced, the I1 and I0 bits of the CC register are both set.

***Note**: Bits in the ISPRx registers which correspond to the NMI can be read and written but they are not significant in the interrupt process management.

Caution: If the I1_x and I0_x bits are modified while the interrupt x is executed the following behaviour has to be considered: If the interrupt x is still pending (new interrupt or flag not cleared) and the new software priority is higher than the previous one, the interrupt x is re-entered. Otherwise, the software priority stays unchanged up to the next interrupt request (after the IRET of the interrupt x).

INTERRUPTS (Cont'd)

Table 6. 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	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.

In order not to lose the current software priority level, the RIM, SIM, HALT, WFI and POP CC instructions should never be used in an interrupt routine.

Table 7. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Address Vector
	RESET	Reset	N/A	Highest Priority	yes	FFFEh-FFFFh
	TRAP	Software Interrupt			no	FFFCh-FFFDh
0	NMI	External Non Maskable Interrupt	MISCR1		yes	FFFAh-FFFBh
1	EI1	External Interrupt Port A3..0	N/A			FFF8h-FFF9h
2	EI2	External Interrupt Port A7..4				FFF6h-FFF7h
3	EI3	External Interrupt Port B6..0				FFF4h-FFF5h
4	EI4	External Interrupt Port C7..0				FFF2h-FFF3h
5	EI5	External Interrupt Port D7..0				FFF0h-FFF1h
6	MCC	Main Oscillator Time Base Interrupt	MCCSR			FFEEh-FFEFh
7	Not used					FFECh-FFEDh
8	CAN*	CAN Peripheral Interrupts	CANISR			FFEAh-FFEBh
9	SPI	SPI Peripheral Interrupts	SPISR		no	FFE8h-FFE9h
10	TIMER A	TIMER A Peripheral Interrupts	TASR			FFE6h-FFE7h
11	TIMER B	TIMER B Peripheral Interrupts	TBSR			FFE4h-FFE5h
12	SCIP	SCI Peripheral Interrupts	SCISR			FFE2h-FFE3h
13	I2C*	I2C Peripheral Interrupts	I2CSRx	Lowest Priority		FFE0h-FFE1h

INTERRUPTS (Cont'd)

Table 8. Nested Interrupts Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
001Ch		EI3		EI2		EI1		NMI	
	ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1		
	Reset Value	1	1	1	1	1	1	1	1
001Dh		ACC		MCC		EI5		EI4	
	ISPR1	I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4	I0_4
	Reset Value	1	1	1	1	1	1	1	1
001Eh		TIMER B		TIMER A		SPI		CAN	
	ISPR2	I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8	I0_8
	Reset Value	1	1	1	1	1	1	1	1
001Fh						I2C		SCI	
	ISPR3					I1_13	I0_13	I1_12	I0_12
	Reset Value	1	1	1	1	1	1	1	1

4.2 POWER SAVING MODES

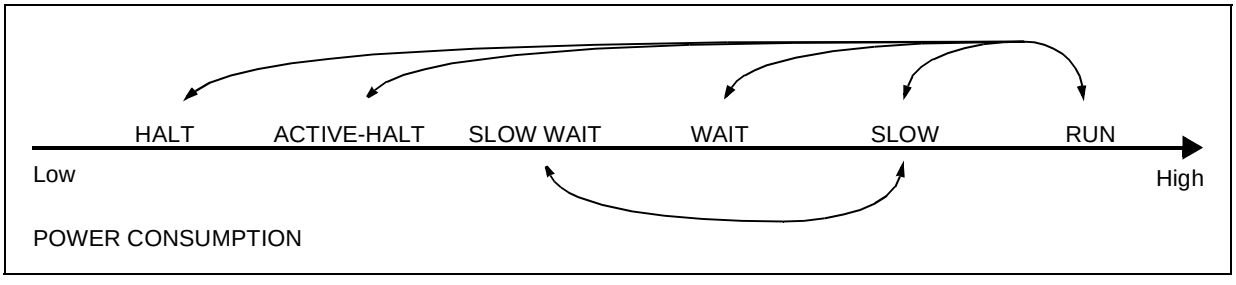
4.2.1 Introduction

To give a large measure of flexibility to the application in terms of power consumption, four main power saving modes are implemented in the ST7.

After a RESET the normal operating mode is selected by default (RUN mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency divided by 2 (f_{CPU}).

From Run mode, the different power saving modes may be selected by setting the relevant register bits or by calling the specific ST7 software instruction whose action depends on the oscillator status.

Figure 17. Power saving mode consumption / transitions



4.2.2 HALT Modes

The HALT modes are the lowest power consumption modes of the MCU. They are entered by executing the ST7 HALT instruction (see Figure 19).

Two different HALT modes can be distinguished:

– HALT: main oscillator is turned off,

– ACTIVE-HALT: only main oscillator is running.

The decision to enter either in HALT or ACTIVE-HALT mode is given by the main oscillator enable interrupt flag (OIE bit in CROSS-MCCSR register: see Table 9).

When entering HALT modes, the I1 and I0 bits in the CC Register are forced to level 0 (“10”) to enable interrupts.

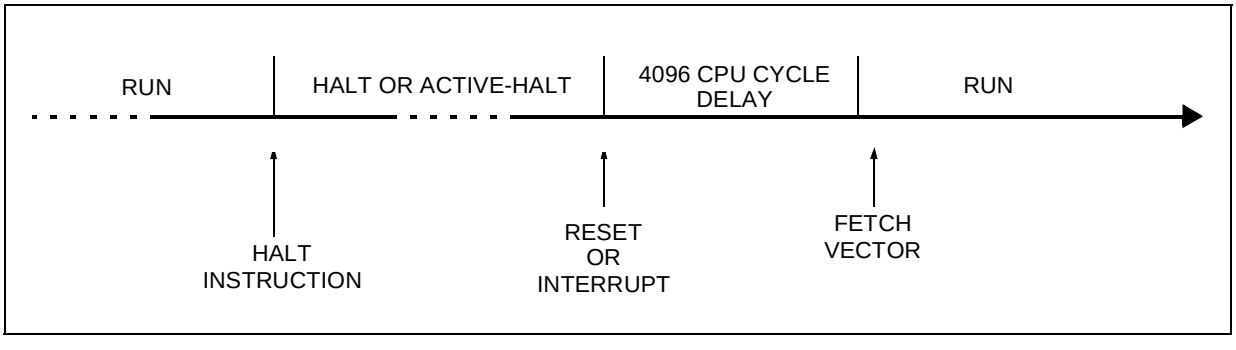
The MCU can exit HALT or ACTIVE-HALT modes on reception of either an external interrupt, an interrupt with Exit from Halt Mode capability or a reset (see Table 2). A 4096 CPU clock cycles delay is performed before the CPU operation resumes (see Figure 18).

After the start up delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up.

Table 9. HALT Modes selection

MCCSR OIE flag	Power Saving Mode entered when HALT instruction is executed
0	HALT (reset if watchdog enabled)
1	ACTIVE-HALT (no reset if watchdog enabled)

Figure 18. HALT /ACTIVE-HALT Modes timing overview



POWER SAVING MODES (Cont'd)

Standard HALT mode

In this mode the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. All peripherals are not clocked except the ones which get their clock supply from another clock generator (such as an external oscillator).
The HALT instruction when executed while the Watchdog system is enabled, generates a Watchdog RESET.
When exiting HALT mode by means of a RESET or an interrupt, the oscillator is immediately turned on and the 4096 CPU cycle delay is used to stabilize the oscillator.

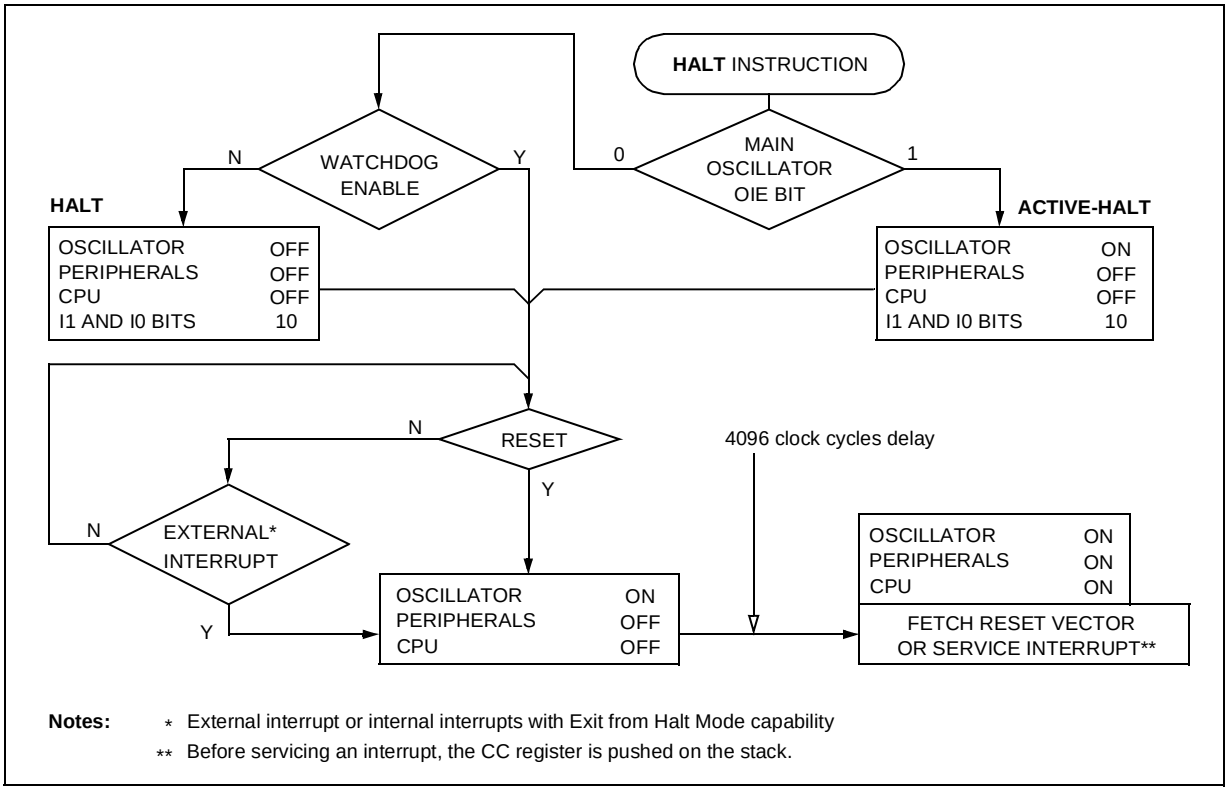
Specific ACTIVE-HALT mode

As soon as the interrupt capability of the main oscillator is selected (OIE bit set), the HALT instruction will make the device enter a specific ACTIVE-HALT power saving mode instead of the standard HALT one.
This mode consists of having only the main oscillator and its associated counter running to keep a wake-up time base. All other peripherals are not clocked except the ones which get their clock supply from another clock generator (such as external oscillator).

The safeguard against staying locked in this ACTIVE-HALT mode is insured by the oscillator interrupt.

Note: As soon as the interrupt capability of the oscillators is selected (OIE bit set), entering in ACTIVE-HALT mode while the Watchdog is active does not generate a RESET.
This means that the device cannot to spend more than a defined delay in this power saving mode.

Figure 19. HALT modes flow-chart



Notes: * External interrupt or internal interrupts with Exit from Halt Mode capability
** Before servicing an interrupt, the CC register is pushed on the stack.

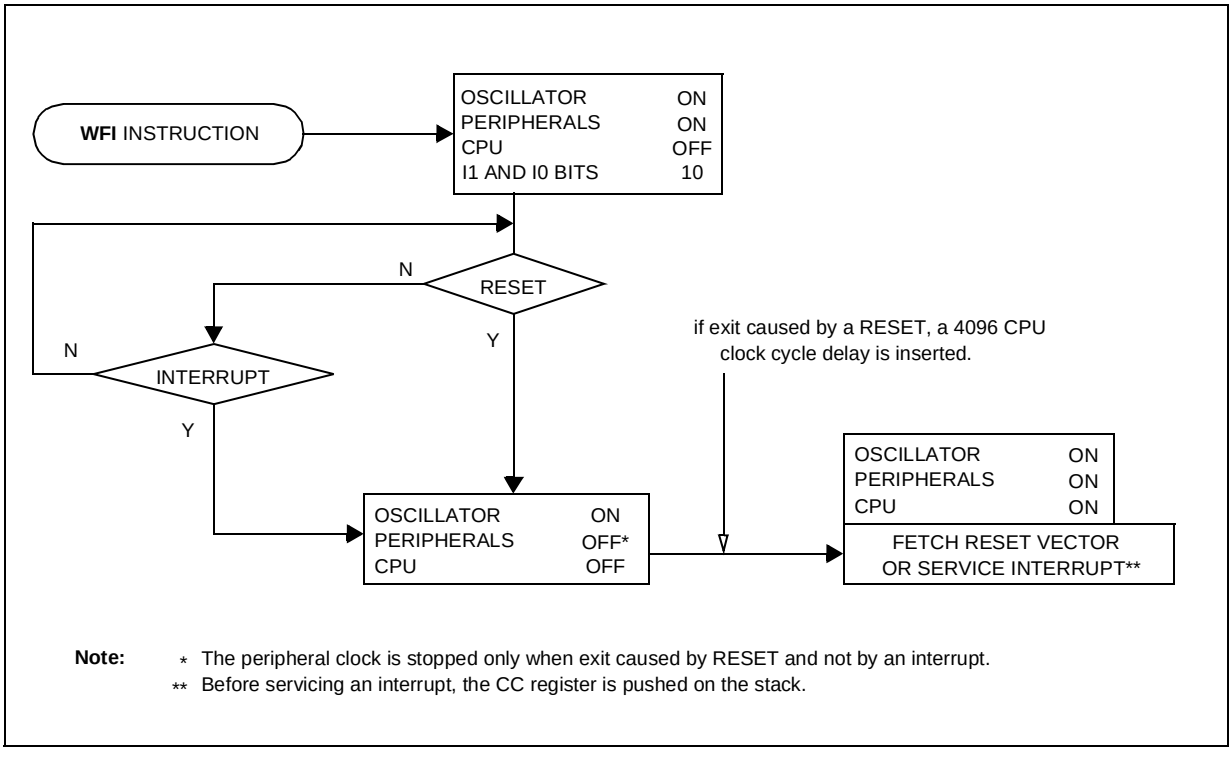
POWER SAVING MODES (Cont'd)

4.2.3 WAIT Mode

WAIT mode places the MCU in a low power consumption mode by stopping the CPU.
This power saving mode is selected by calling the "WFI" ST7 software instruction.
All peripherals remain active. During WAIT mode, the I1 and I0 bits of the CC register are forced to level 0 ("10"), to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in WAIT mode until an interrupt or Reset occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine.
The MCU will remain in WAIT mode until a Reset or an Interrupt occurs, causing it to wake up.

Refer to Figure 20.

Figure 20. WAIT mode flow-chart



Note: * The peripheral clock is stopped only when exit caused by RESET and not by an interrupt.
** Before servicing an interrupt, the CC register is pushed on the stack.

POWER SAVING MODES (Cont'd)

4.2.4 SLOW Mode

This mode has two targets:

– To reduce power consumption by decreasing the internal clock in the device,

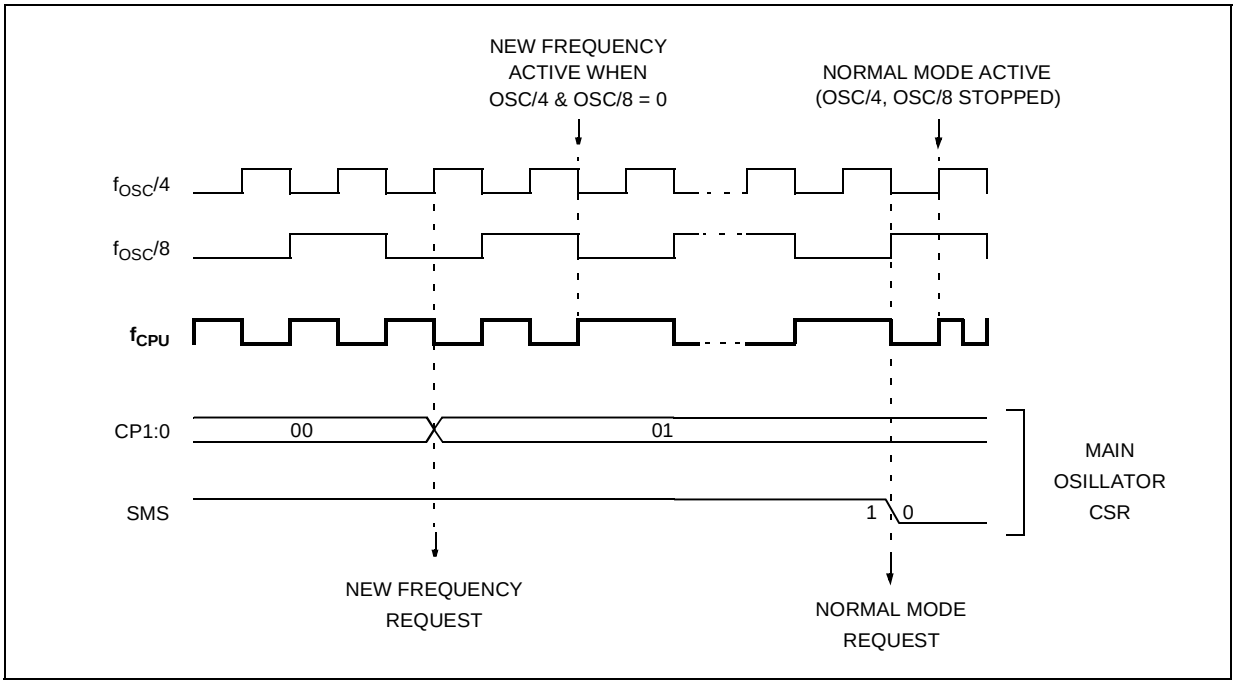
– To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

SLOW mode is controlled by three bits in the main oscillator CSR register: the SMS bit which enables or disables Slow mode and two CPx bits which select the internal slow frequency (f_{CPU}).

In this mode, the oscillator frequency can be divided by 4, 8, 16 or 32 instead of 2 in normal operating mode. The CPU and peripherals (except CAN, see Note) are clocked at this lower frequency.

Note: Before entering SLOW mode and in order to guarantee low power operation, the CAN peripheral must be placed by software in STANDBY mode.

Figure 21. SLOW Mode: timing diagram for internal CPU clock transitions



5 I/O PORTS

5.1 INTRODUCTION

The I/O ports offer different functional modes:
– transfer of data through digital inputs and outputs

and for specific pins:
– external interrupt generation
– alternate signal input/output for the on-chip peripherals.

An I/O port contains up to 8 pins. Each pin can be programmed independently as digital input (with or without interrupt generation) or digital output.

5.2 FUNCTIONAL DESCRIPTION

Each port has 2 main registers:

– Data Register (DR)

– Data Direction Register (DDR)

and one optional register:

– Option Register (OR)

Each I/O pin may be programmed using the corresponding register bits in the DDR and OR registers: bit X corresponding to pin X of the port. The same correspondence is used for the DR register.

The following description takes into account the OR register, (for specific ports which do not provide this register refer to the I/O Port Implementation section). The generic I/O block diagram is shown in Figure 1

5.2.1 Input Modes

The input configuration is selected by clearing the corresponding DDR register bit.

In this case, reading the DR register returns the digital value applied to the external I/O pin.

Different input modes can be selected by software through the OR register.

Notes:
1. Writing the DR register modifies the latch value but does not affect the pin status.
2. When switching from input to output mode, the DR register has to be written first to drive the correct level on the pin as soon as the port is configured as an output.
3. Do not use read/modify/write instructions (BSET or BRES) to modify the DR register

External interrupt function

When an I/O is configured as Input with Interrupt, an event on this I/O can generate an external interrupt request to the CPU.

Each pin can independently generate an interrupt request. The interrupt sensitivity is independently programmable using the sensitivity bits in the Miscellaneous register.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see pinout description and interrupt section). If several input pins are selected simultaneously as interrupt source, these are logically NANDed. For this reason if one of the interrupt pins is tied low, it masks the other ones.

In case of a floating input with interrupt configuration, special care must be taken when changing the configuration (see Figure 2).

The external interrupts are hardware interrupts, which means that the request latch (not accessible directly by the application) is automatically cleared when the corresponding interrupt vector is fetched. To clear an unwanted pending interrupt by software, the sensitivity bits in the Miscellaneous register must be modified.

5.2.2 Output Modes

The output configuration is selected by setting the corresponding DDR register bit. In this case, writing the DR register applies this digital value to the I/O pin through the latch. Then reading the DR register returns the previously stored value.

Two different output modes can be selected by software through the OR register: Output push-pull and open-drain.

DR register value and output pin status:

DR	Push-pull	Open-drain
0	V_{SS}	Vss
1	V_{DD}	Floating

5.2.3 Alternate Functions

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over the standard I/O programming.

When the signal is coming from an on-chip peripheral, the I/O pin is automatically configured in output mode (push-pull or open drain according to the peripheral).

When the signal is going to an on-chip peripheral, the I/O pin must be configured in input mode. In this case, the pin state is also digitally readable by addressing the DR register.

Note: Input pull-up configuration can cause unexpected value at the input of the alternate peripheral input. When an on-chip peripheral use a pin as input and output, this pin has to be configured in input floating mode.

I/O PORTS (Cont'd)

Figure 22. I/O Port General Block Diagram

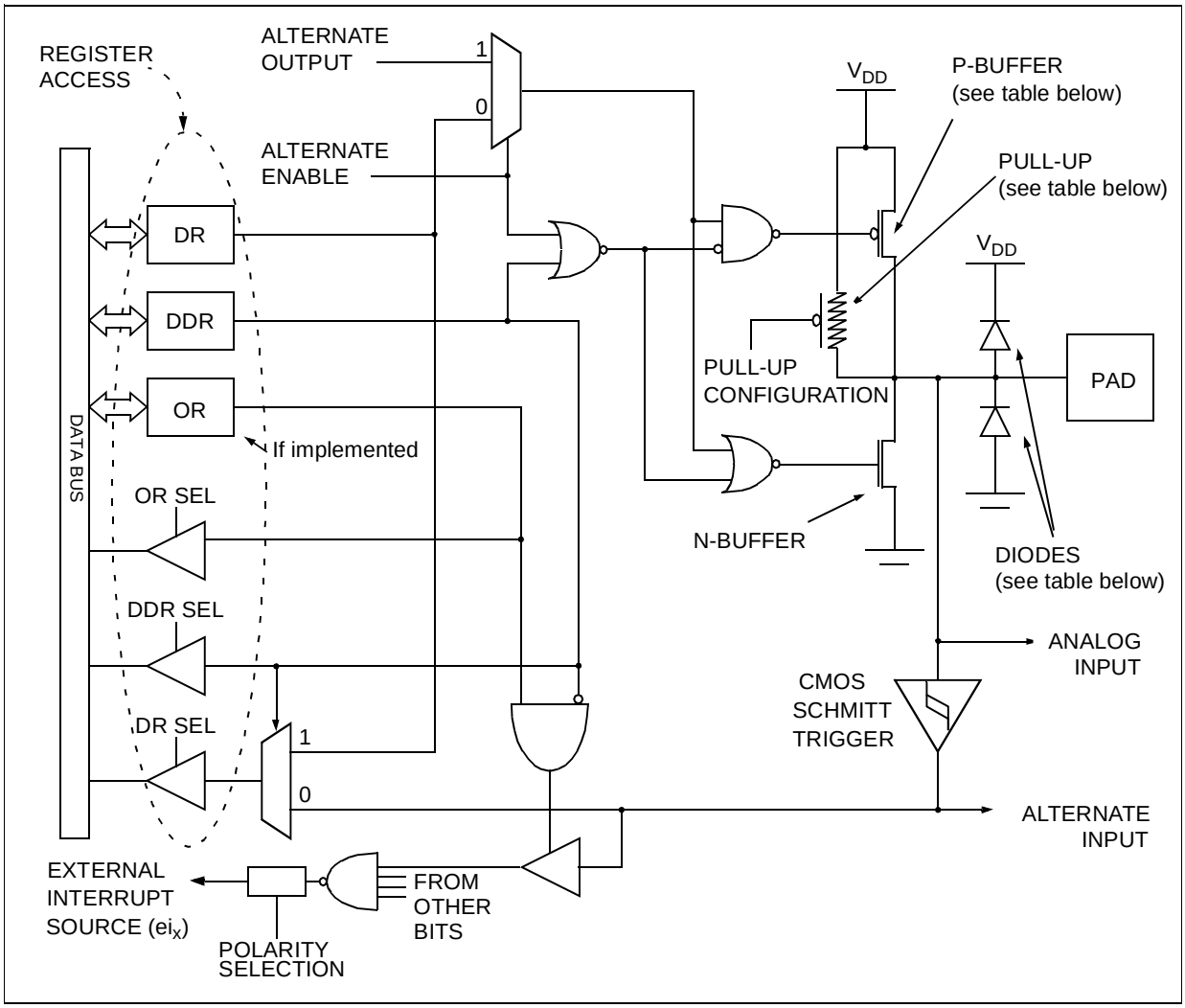


Table 10. I/O Port Mode Options

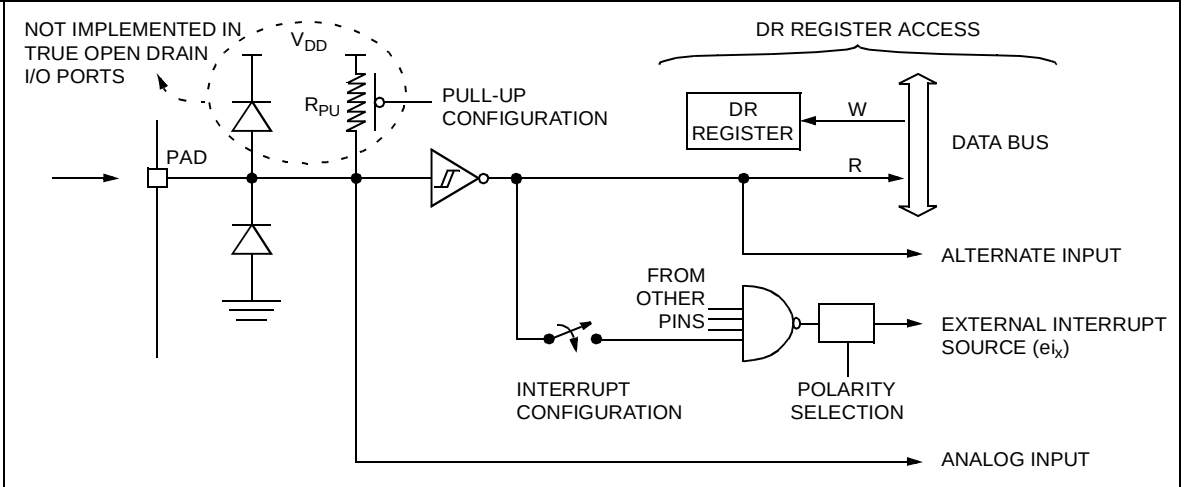
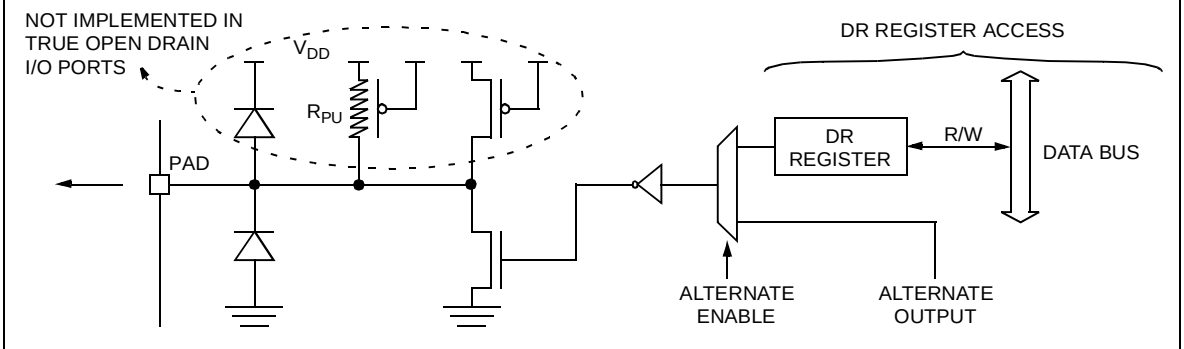
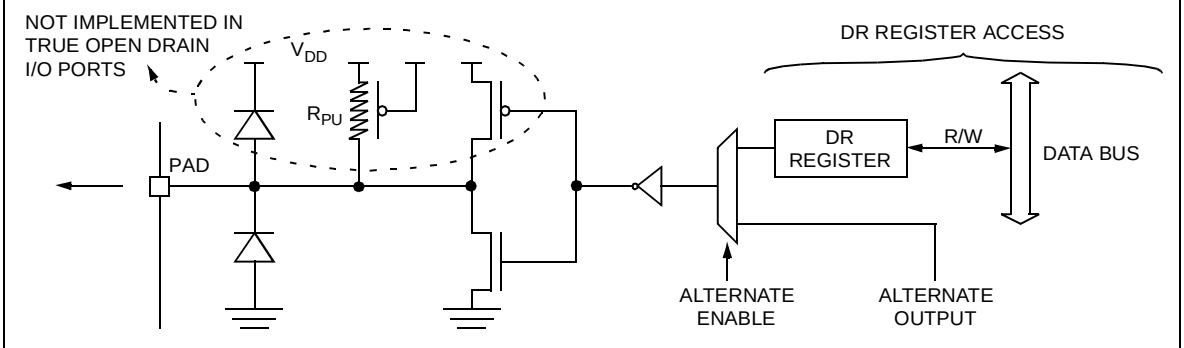
Configuration Mode		Pull-Up	P-Buffer	Diodes	
				to V_{DD}	to V_{SS}
Input	Floating with/without Interrupt	Off	Off	On	On
	Pull-up with/without Interrupt	On			
Output	Push-pull	Off	On		
	Open Drain (logic level)		Off		
	True Open Drain	NI	NI	NI (see note)	

Legend: NI - not implemented
Off - implemented not activated
On - implemented and activated

Note: The diode to V_{DD} is not implemented in the true open drain pads. A local protection between the pad and V_{SS} is implemented to protect the device against positive stress.

I/O PORTS (Cont'd)

Table 11. I/O Port Configurations

	Hardware Configuration
INPUT [1]	
OPEN-DRAIN OUTPUT [2]	
PUSH-PULL OUTPUT [2]	

Notes:

1. When the I/O port is in input configuration and the associated alternate function is enabled as an output, reading the DR register will read the alternate function output status.
2. When the I/O port is in output configuration and the associated alternate function is enabled as an input, the alternate function reads the pin status given by the DR register content.

I/O PORTS (Cont'd)

CAUTION: The alternate function must not be activated as long as the pin is configured as input with interrupt, in order to avoid generating spurious interrupts.

Analog alternate function

When the pin is used as an ADC input, the I/O must be configured as floating input. The analog multiplexer (controlled by the ADC registers) switches the analog voltage present on the selected pin to the common analog rail which is connected to the ADC input.

It is recommended not to change the voltage level or loading on any port pin while conversion is in progress. Furthermore it is recommended not to have clocking pins located close to a selected analog pin.

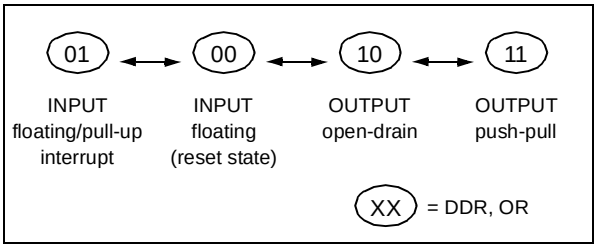
WARNING: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

5.3 I/O PORT IMPLEMENTATION

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific feature of the I/O port such as ADC Input or true open drain.

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in Figure 2 Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 23. Interrupt I/O Port State Transitions



The I/O port register configurations are summarized as follows.

Standard Interrupt Ports

PA7:0, PB6:0, PC7:0, PD7:6, PD3:0

MODE	DDR	OR
floating input	0	0
floating interrupt	0	1
open drain output	1	0
push-pull output	1	1

Open Drain Ports

PD5:4

MODE	DDR
floating input	0
open drain output	1

Dedicated Configurations

Table 12. Port Configuration

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7:PA0	floating	floating interrupt	open drain	push-pull
Port B	PB6:PB0				
Port C	PC7:PC0				
Port D	PD3:PD0				
	PD5:PD4	floating		open-drain	
	PD7:PD6	floating	floating interrupt	open drain	push-pull

I/O PORTS (Cont'd)

5.3.1 Register Description

DATA REGISTER (DR)

Port x Data Register
PxDR with x = A, B, C or D.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **D[7:0]** *Data Register 8 bits.*

The DR register has a specific behaviour according to the selected input/output configuration. Writing the DR register is always taken into account even if the pin is configured as an input; this allows to always have the expected level on the pin when toggling to output mode. Reading the DR register returns either the DR register latch content (pin configured as output) or the digital value applied to the I/O pin (pin configured as input).

DATA DIRECTION REGISTER (DDR)

Port x Data Direction Register
PxDDR with x = A, B, C or D.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0

Bit 7:0 = **DD[7:0]** *Data Direction Register 8 bits.*

The DDR register gives the input/output direction configuration of the pins. Each bits is set and cleared by software.

0: Input mode
1: Output mode

OPTION REGISTER (OR)

Port x Option Register
PxOR with x = A, B, C or D.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
O7	O6	O5	O4	O3	O2	O1	O0

Bit 7:0 = **O[7:0]** *Option Register 8 bits.*

For specific I/O pins, this register is not implemented. In this case the DDR register is enough to select the I/O pin configuration.

The OR register allows to distinguish: in input mode if the floating interrupt capability or the basic floating configuration is selected, in output mode if the push-pull or open drain configuration is selected.

Each bit is set and cleared by software.

Input mode:
0: floating input
1: floating input with interrupt

Output mode:
0: output open drain (with P-Buffer unactivated)
1: output push-pull

I/O PORTS (Cont'd)

Table 13. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all IO port registers		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PAOR**								
0004h	**PBDR**	MSB							LSB
0005h	**PBDDR**								
0006h	**PBOR**								
0008h	**PCDR**	MSB							LSB
0009h	**PCDDR**								
000Ah	**PCOR**								
000Ch	**PDDR**	MSB							LSB
000Dh	**PDDDR**								
000Eh	**PDOR**								

6 MISCELLANEOUS REGISTERS

The Miscellaneous registers allow control over several features such as external interrupts or the I/O alternate functions.

6.1 I/O Port Interrupt Sensitivity Description

The external interrupt sensitivity is controlled by the ISxx bits of the Miscellaneous registers (Figure 24). This control allows to have 2 fully independent external interrupt source sensitivities.

Each external interrupt source can be generated on four different events on the pin:

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

To guaranty the functionality, a modification of the sensitivity in the MISCR registers can be done only when the I1 and I0 bits of the CC register are both set to 1 (level 3). See I/O port register and Miscellaneous register descriptions for more details on the programming.

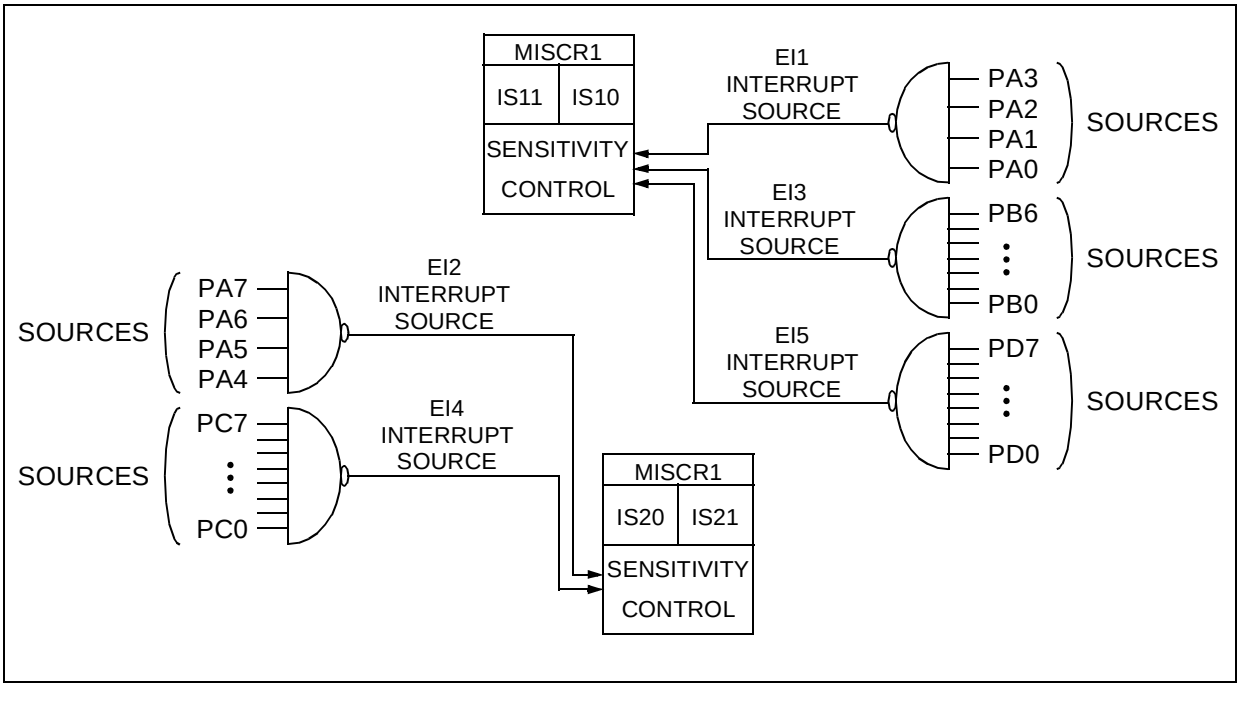
6.2 I/O Port Alternate Functions

The MISCR registers allow to manage three I/O port miscellaneous alternate functions:

- A Beep signal output on PC7 (with three selectable audio frequencies)
- A NMI management on a dedicated pin
- A SPI $\overline{SS}$ pin internal control to use the PD3 I/O port function while the SPI is active.

These functions are described in details in the Section 6.3 Miscellaneous Registers Description.

Figure 24. External Interrupt Sources vs MISCR



MISCELLANEOUS REGISTERS (Cont'd)

6.3 Miscellaneous Registers Description

MISCELLANEOUS REGISTER 1 (MISCR1)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
IS11	IS10	IS21	IS20	0	0	NMIS	NMIE

Bit 7:6 = **IS11-IS10** *EI1,3, 5 Sensitivity*

ISx1	ISx0	External Interrupt Sensitivity
0	0	Falling edge and low level
0	1	Falling edge only
1	0	Rising edge only
1	1	Rising and falling edge

The selection issued from IS11,IS10 combination is applied to the following external interrupts: EI1 (port A3..0) EI3 (port B) and EI5 (port D).
These 2 bits can be written only when the current interrupt software priority in the CC (Condition Code) register is set to level 3 (I1:0=11).

Bit 5:4 = **IS21-IS20** *EI2,4 Sensitivity*
The selection issued from IS21,IS20 combination is applied to the following external interrupts: EI2 (port A7..4) and EI4 (port C).
The functional description is equal to the IS1x one.

Bit 3:2 = **Reserved**, always read as 0.

Bit 1 = **NMIS** *NMI Sensitivity*
This bit allows to toggle the NMI edge sensitivity. It can be set and cleared by software only when NMIE bit is cleared.
0: falling edge
1: rising edge

Bit 0 = **NMIE** *NMI Enable*
This bit allows to enable or disable the NMI capability on the dedicated pin. It is set and cleared by software.
0: NMI disable
1: NMI enable

MISCELLANEOUS REGISTER 2 (MISCR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	MCO	BC1	BC0	0	0	SSM	SSI

Bit 7 = **Reserved.**

Bit 6:4 = **MCO** *Main clock-out control*
BC1-BC0 *Beep Control*
These 3 bits select the PC7 pin configuration. They are set and cleared by software.

MCO	BC1	BC0	PC7 Configuration	
0	0	0	Standard I/O	
1	0	0	f_{OSC}/2 Clock out	
X	0	1	~2-KHz	Output Beep signal w/ f_{OSC}=16MHz ~50% duty cycle
X	1	0	~1-KHz	
X	1	1	~500-Hz	

Note: the clock out and beep capabilities are not available in HALT modes.

Bit 3:2 = **Reserved**, always read as 0.

Bit 1 = **SSM** *$\overline{SS}$ mode selection*
It is set and cleared by software.
0: Normal mode - $\overline{SS}$ uses information coming from the $\overline{SS}$ pin of the SPI.
1: I/O mode, the SPI uses the information stored into bit SSI.

Bit 0 = **SSI** *$\overline{SS}$ internal mode*
This bit replaces pin $\overline{SS}$ of the SPI when bit SSM is set to 1. (see SPI description). It is set and cleared by software.

MISCELLANEOUS REGISTERS (Cont'd)

Table 14. Miscellaneous Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0020h	**MISCR1**	IS11	IS10	IS21	IS20			NMIS	NMI
	Reset Value	0	0	0	0	0	0	0	0
0040h	**MISCR2**		MCO	BC1	BC0			SSM	SSI
	Reset Value	0	0	0	0	0	0	0	0

7 ON-CHIP PERIPHERALS

7.1 LCD DRIVER

7.1.1 Introduction

The LCD driver controls up to 60 segments and 8 backplanes to drive up to 60x8 (480) or 60x4(240) LCD segments.

Two programmable display modes (1/4 and 1/8 duty cycle) with 4 LCD drive frequencies can be selected by software.

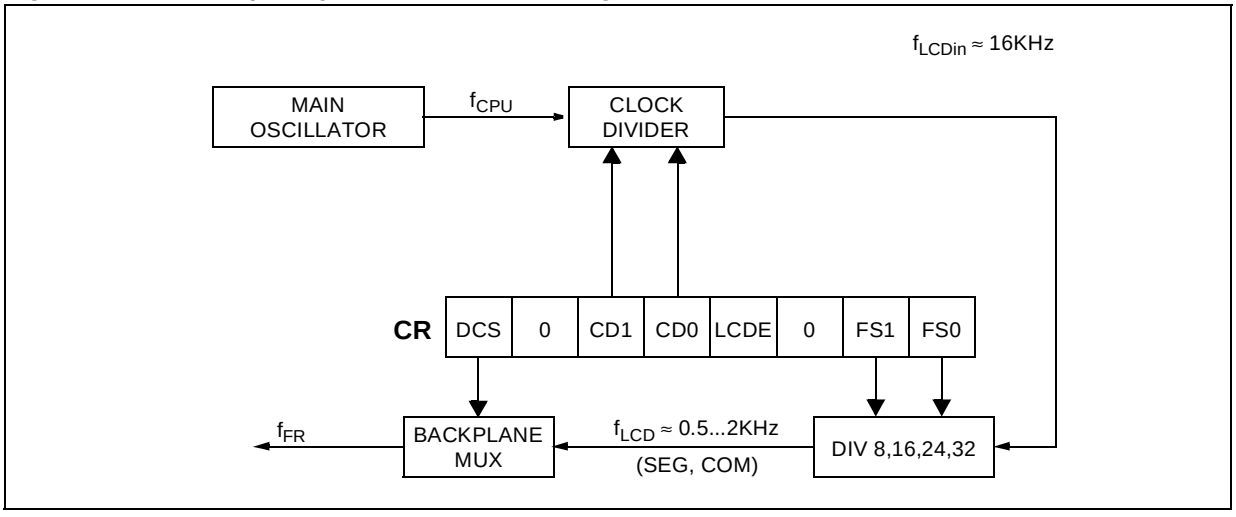
The parameters to display are stored in a 60-byte LCD dual port RAM.

Four different main oscillator clocks can be selected as clock for the peripheral.

– 8, 4, 2, 1 MHz f_{CPU} software selectable.

The peripheral can be switched off by software to reduce the power consumption while it is not used.

Figure 25. LCD Frequency Generator Block Diagram



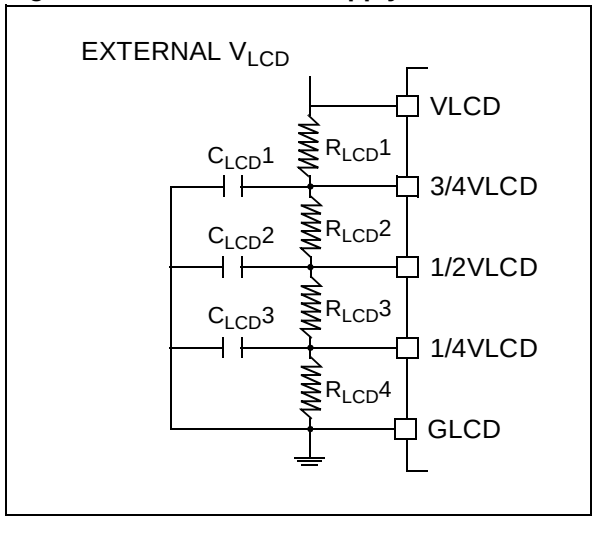
7.1.2 Voltage references

The display voltage levels are supplied by an external resistor chain as shown in Figure 26 This LCD driver needs 5 external voltage references through 5 pins (GLCD, 1/4VLCD, 1/2VLCD, 3/4VLCD, VLCD).

The resistors used must have good tolerance matching within 1% to avoid DC voltage levels on the liquid crystal device. DC levels trigger electrode reactions in the liquid crystal cell, causing a rapid deterioration of the display quality.

Note: To avoid damaging the device, VLCD supply voltages must always be supplied with more than V_{DD} or they have to be left unconnected.

Figure 26. LCD External Supply Network

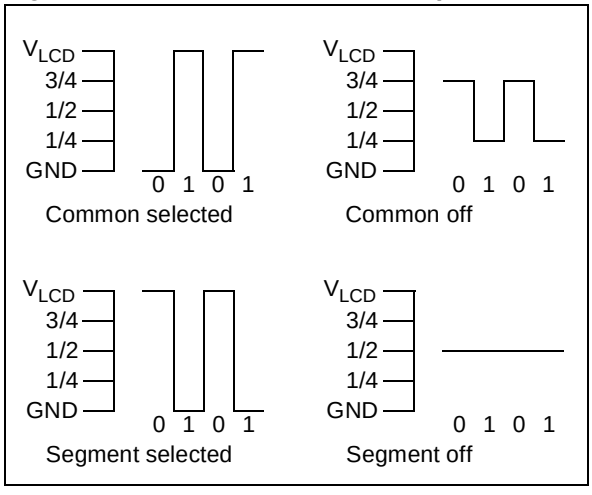


LCD DRIVER (Cont'd)

7.1.3 Segment and Common Output signals

Each dot of the LCD dot matrix panel is turned on when the differential voltage between the segment signal and the common signal increases over a certain threshold, it is turned off when the voltage is below the threshold voltage. The common signals determine the select timing within a frame cycle (see Figure 27). The common signals have similar waveforms to the segments, but different phases.

Figure 27. Waveforms on LCD Outputs

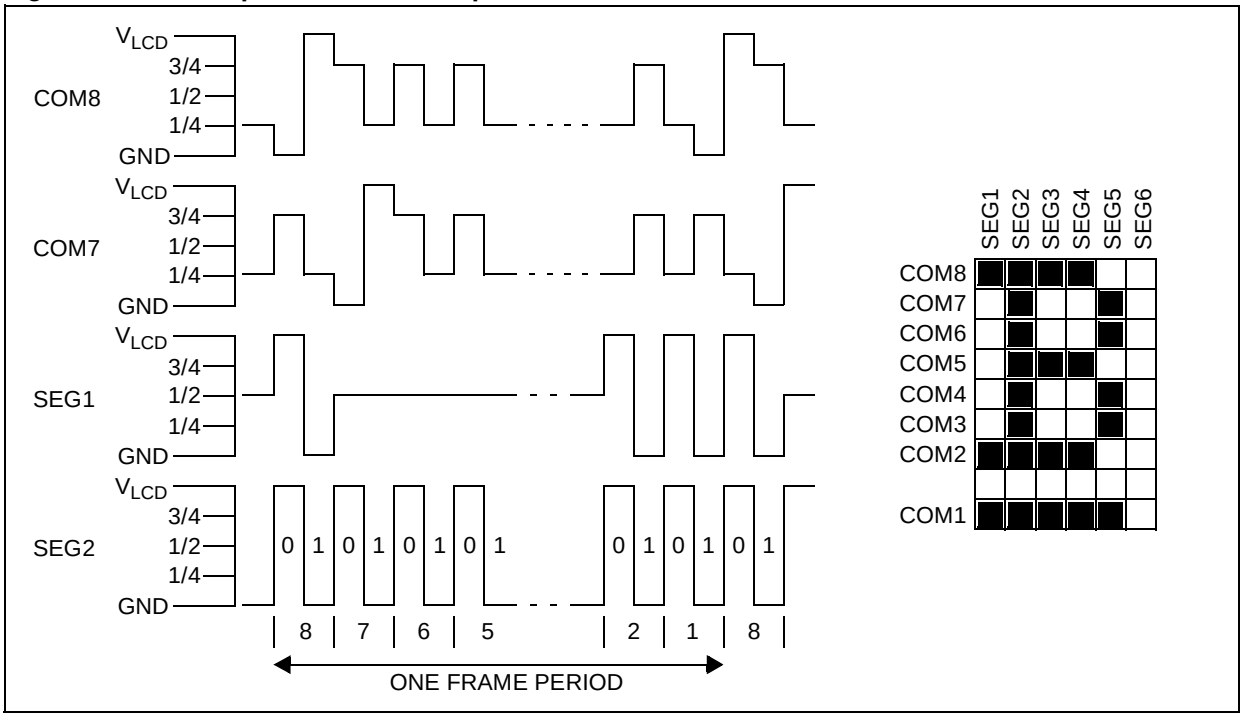


Each common signal shows a high signal amplitude (VLCD-VSS) only at the corresponding section of a frame time. At the other sections of the frame, the signal amplitude is low (3/4VLCD-1/4VLCD). A dot can be turned-on only at phases with high signal amplitude.

In 1/8 duty cycle mode, one frame is divided into 8 sections, and each section is divided into two phases, phase 0 and 1. In 1/4 duty cycle mode, the number of sections is reduced to 4. This means the waveform pattern repeats faster in 1/4 duty cycle mode than 1/8 mode and the average voltage and the ON/OFF duty cycle on a selected pin is higher than in 1/8 mode. This results in a better contrast of the display.

Note: The LCD must be disabled before entering HALT mode or ACTIVE HALT mode.

Figure 28. LCD outputs with 1/8 Multiplex



LCD DRIVER (Cont'd)

7.1.4 Register Description

LCD CONTROL REGISTER (CR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DCS	0	CD1	CD0	LCDE	0	FS1	FS0

Bit 7 = **DCS** *Duty cycle selection*
This bit is set and cleared by software.
0: 1/4 duty cycle selected
1: 1/8 duty cycle selected

Bit 6 = Reserved, always read as 0.

Bit 5:4 = **CD1,CD0** *Clock divider*
These bits allow to tune the f_{LCDin} frequency to ~16KHz based on the selected f_{CPU}.
These bits are set and cleared by software.

f_{LCDin}	f_{CPU}	Divider	CD1	CD0
15625Hz	8-MHz	1/512	0	0
	4-MHz	1/256	1	1
	2-MHz	1/128	1	0
	1-MHz	1/64	0	1

Bit 3 = **LCDE** *LCD enable*
This bit is set and cleared by software.
0: LCD disable
1: LCD enable
While the LCD is disabled (LCDE bit cleared), GLCD is applied to all Segment and Common pins.

Bit 2 = **Reserved**, must be kept cleared.

Bit 1:0 = **FS1,FS0** f_{FR} *Frame Frequency selection*
These two bits allow to select the LCD frame frequency based on the f_{LCDin} frequency and the selected duty cycle.
These bits are set and cleared by software.
The following table gives the possible LCD segment frequency (f_{LCD}) and LCD frame frequency (f_{FR}) according to the selected duty cycle.

With f_{LCDin}=15625Hz (main oscillator)

f_{LCDin} Ratio	f_{LCD}	f_{FR}		FS1	FS0
		1/4 d.c.	1/8 d.c.		
1/8	1953-Hz	488-Hz	244-Hz	0	0
1/16	977-Hz	244-Hz	122-Hz	0	1
1/24	651-Hz	163-Hz	81-Hz	1	0
1/32	488-Hz	122-Hz	61-Hz	1	1

LCD DRIVER (Cont'd)

LCD RAM DESCRIPTION

The LCD RAM is located in the data space in one page of 60 bytes. Each bit of the LCD RAM is mapped to one dot of the LCD matrix. If a bit is set, the corresponding LCD dot is switched on, else the dot is switched off.

After reset, the LCD RAM is not initialized and contains arbitrary information.

LCD RAM Bit position	7	6	5	4	3	2	1	0
LCD Common	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1

The bit position of the selected bit in the LCD RAM byte gives the common data. The segment data is given by the LCD RAM relative address.

LCD RAM Relative address	00h	01h	02h	.....	39h	3Ah	3Bh
LCD Segment	S1	S2	S3		S58	S59	S60

Table 15. LCD Driver Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0058h	**LCDCR**	DCS		CD1	CD0	LCDE		FS1	FS0
	Reset Value	0	0	0	0	0	0	0	0
0480h to 04BBh	**LCDRAM**	Seg X Com8	Seg X Com7	Seg X Com6	Seg X Com5	Seg X Com4	Seg X Com3	Seg X Com2	Seg X Com1
	Reset Value	X	X	X	X	X	X	X	X

7.2 WATCHDOG TIMER (WDG)

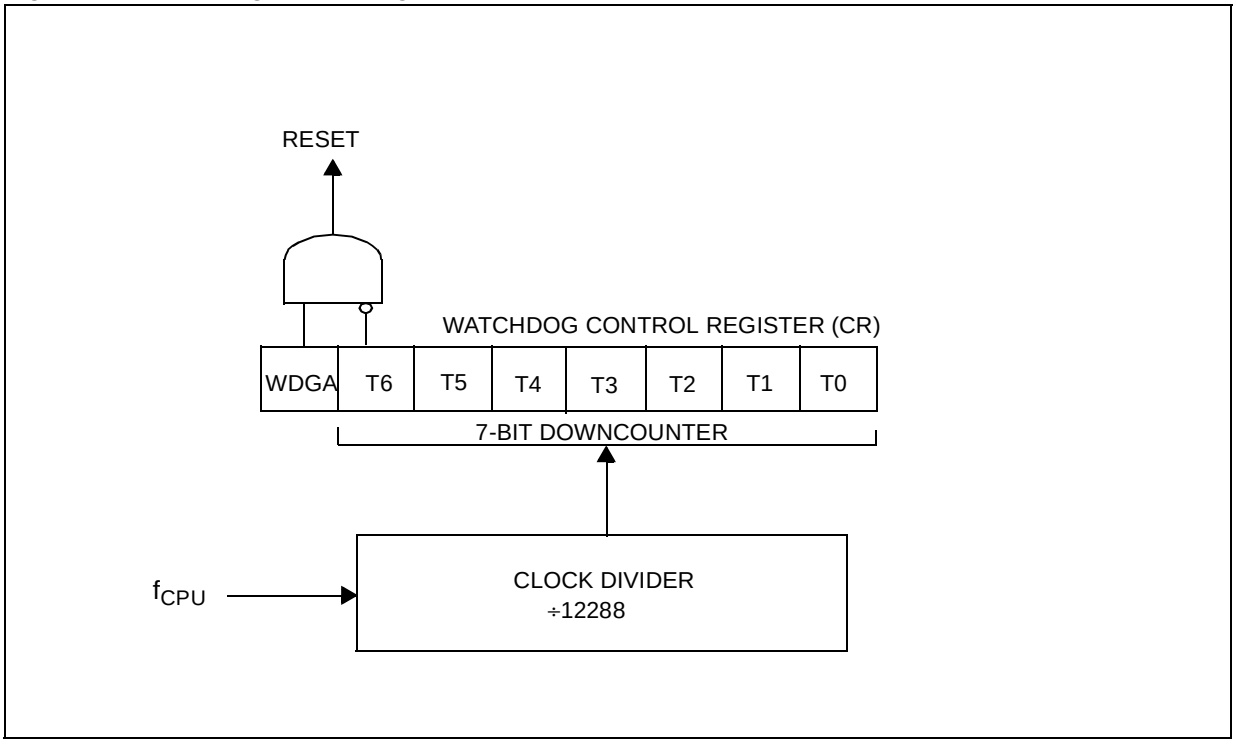
7.2.1 Introduction

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.

7.2.2 Main Features

- Programmable timer (64 increments of 12288 CPU cycles)
- Programmable reset
- Reset (if watchdog activated) after a HALT instruction or when the T6 bit reaches zero

Figure 29. Watchdog Block Diagram



WATCHDOG TIMER (Cont'd)

7.2.3 Functional Description

The counter value stored in the CR register (bits T6:T0), is decremented every 12,288 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 T6:T0) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 500ns.

The application program must write in the CR register at regular intervals during normal operation to prevent an MCU reset. The value to be stored in the CR register must be between FFh and C0h (see Table 16 . Watchdog Timing (fCPU = 8 MHz)):

– The WDGA bit is set (watchdog enabled)

– The T6 bit is set to prevent generating an immediate reset

– The T5:T0 bits contain the number of increments which represents the time delay before the watchdog produces a reset.

Table 16. Watchdog Timing (f_{CPU} = 8 MHz)

	CR Register initial value	WDG timeout period (ms)
Max	FFh	98.304
Min	C0h	1.536

Notes: Following a reset, the watchdog is disabled. Once activated it cannot be disabled, except by a reset.

The T6 bit can be used to generate a software reset (the WDGA bit is set and the T6 bit is cleared).

If the watchdog is activated, the HALT instruction will generate a Reset.

7.2.4 Low Power Modes

Mode	Description
WAIT	No effect on Watchdog.
HALT	Immediate reset generation as soon as the HALT instruction is executed if the Watchdog is activated (WDGA bit is set).

7.2.5 Interrupts

None.

7.2.6 Register Description

CONTROL REGISTER (CR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

Bit 7 = **WDGA** *Activation bit.*
This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset.
0: Watchdog disabled
1: Watchdog enabled

Bit 6:0 = **T[6:0]** *7-bit timer (MSB to LSB).*
These bits contain the decremented value. A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

WATCHDOG TIMER (Cond't)

Table 17. Watchdog Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0024h	**WDGCR**	WDGA	T6	T5	T4	T3	T2	T1	T0
	Reset Value	0	1	1	1	1	1	1	1

7.3 16-BIT TIMER

7.3.1 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 measuring the pulse lengths of up to two input signals (*input capture*) or generating 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).

7.3.2 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 4 times slower than the CPU clock speed) with the choice of active edge
- Output compare functions with:
  - 2 dedicated 16-bit registers
  - 2 dedicated programmable signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Input capture functions 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
- 5 alternate functions on I/O ports (ICAP1, ICAP2, OCMP1, OCMP2, EXTCLK)*

The Block Diagram is shown in Figure 30.

***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'.

7.3.3 Functional Description

7.3.3.1 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 & 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).

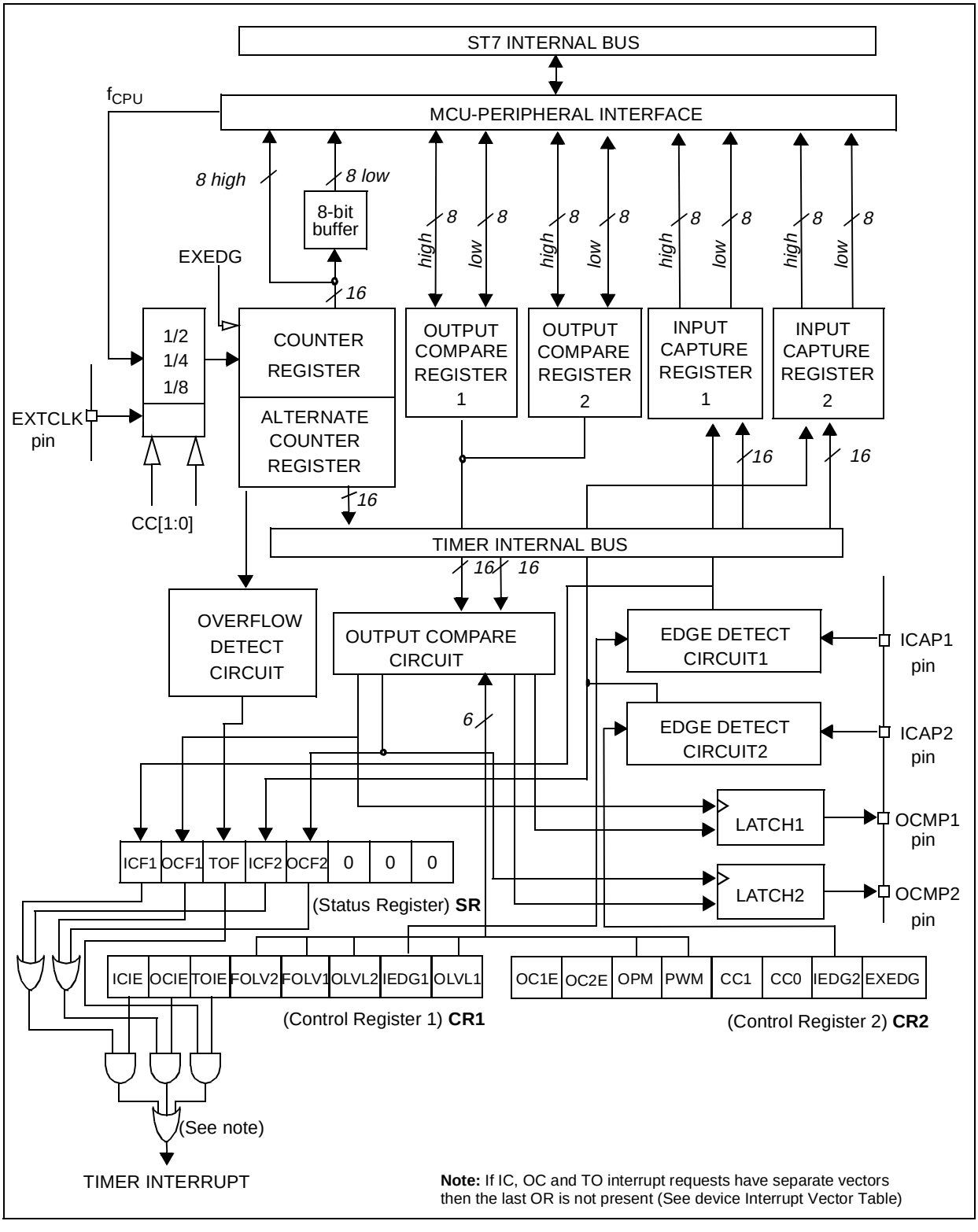
These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register (SR). (See note at the end of paragraph titled 16-bit read sequence).

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 18 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.

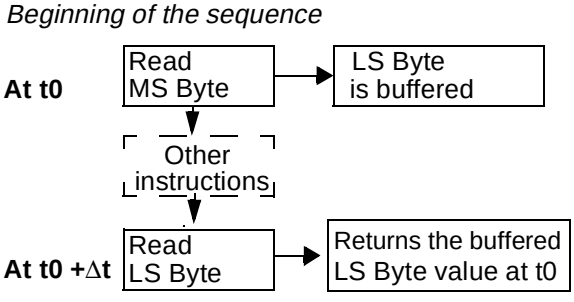
16-BIT TIMER (Cont'd)

Figure 30. Timer Block Diagram



16-BIT TIMER (Cont'd)

16-bit Read Sequence: (from either the Counter Register or the Alternate Counter Register).



The user must read the MS Byte first, then the LS Byte value is buffered automatically.

This buffered value remains unchanged until the 16-bit read sequence is completed, even if the user reads the MS Byte several times.

After a complete reading sequence, if only the CLR register or ACLR register are read, they return the LS Byte of the count value at the time of the read.

Whatever the timer mode used (input capture, output compare, One Pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFFFh to 0000h then:

– The TOF bit of the SR register is set.

– A timer interrupt is generated if:

  – TOIE bit of the CR1 register is set and

  – I bit of the CC register is cleared.

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

Clearing the overflow interrupt request is done in two steps:

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

Note: The TOF bit is not cleared by accessing the ACLR register. The advantage of accessing the ACLR register rather than the CLR register is that it allows simultaneous use of the overflow function and reading the free running counter at random times (for example, to measure elapsed time) without the risk of clearing the TOF bit erroneously.

The timer is not affected by WAIT mode.

In HALT mode, the counter stops counting until the mode is exited. Counting then resumes from the previous count (MCU awakened by an interrupt) or from the reset count (MCU awakened by a Reset).

7.3.3.2 External Clock

The external clock (where available) is selected if CC0=1 and CC1=1 in the CR2 register.

The status of the EXEDG bit in the CR2 register determines the type of level transition on the external clock pin EXTCLK that will trigger the free running counter.

The counter is synchronised with the falling edge of the internal CPU clock.

A minimum of four falling edges of the CPU clock must occur between two consecutive active edges of the external clock; thus the external clock frequency must be less than a quarter of the CPU clock frequency.

16-BIT TIMER (Cont'd)

Figure 31. Counter Timing Diagram, internal clock divided by 2

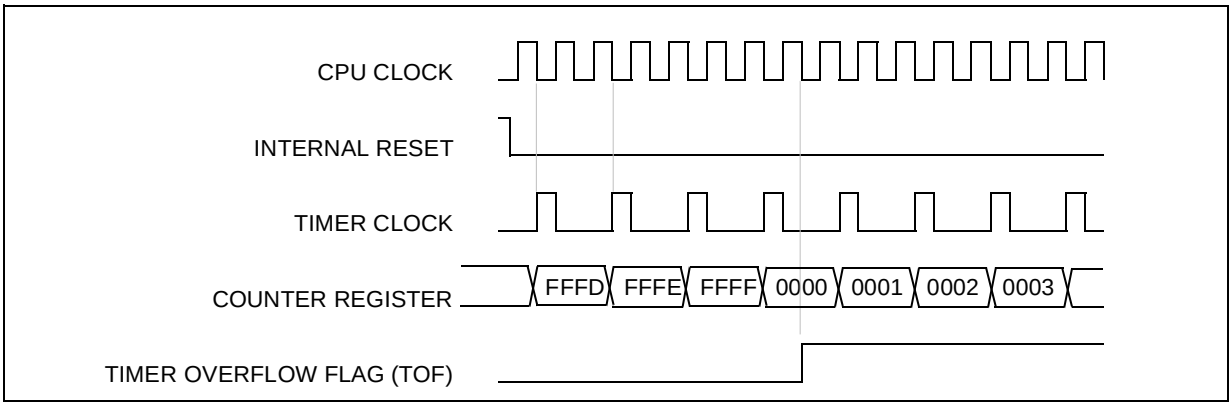


Figure 32. Counter Timing Diagram, internal clock divided by 4

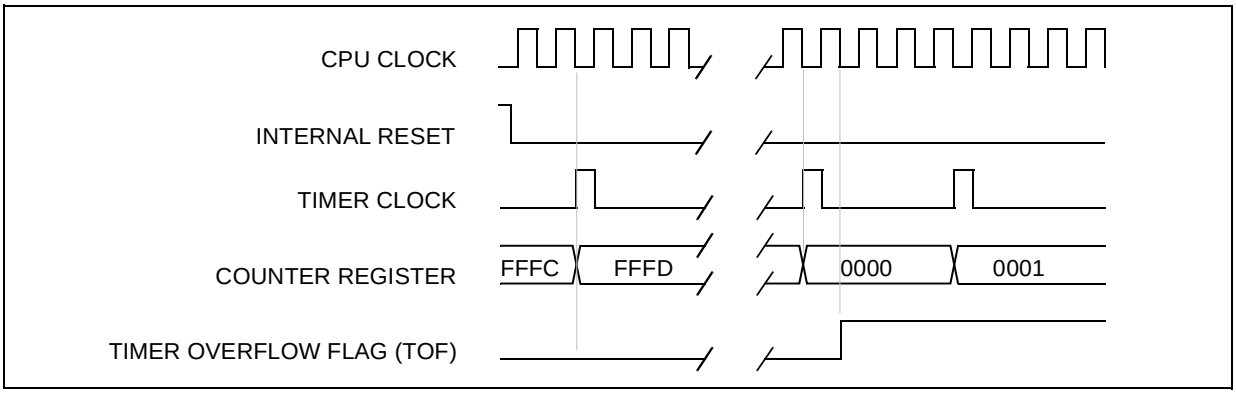
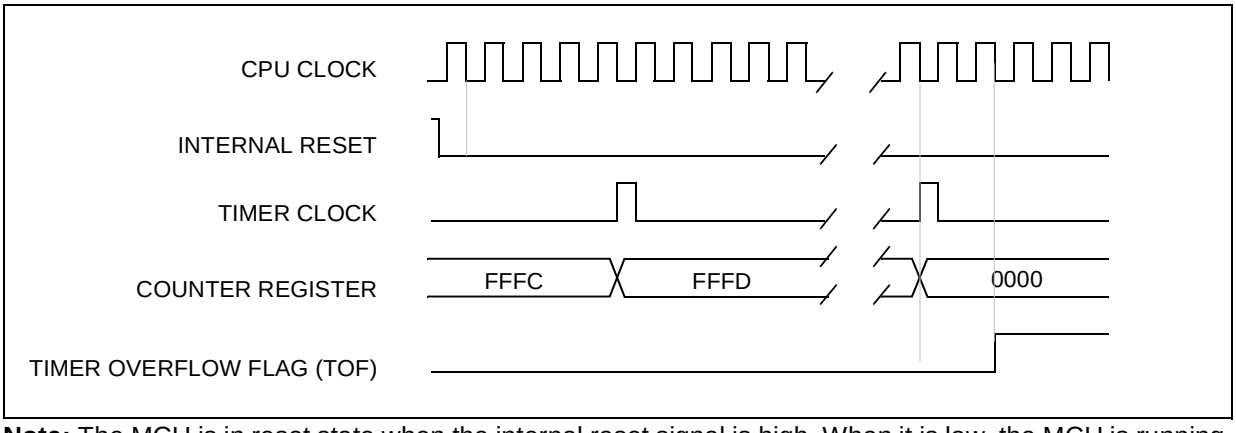


Figure 33. Counter Timing Diagram, internal clock divided by 8



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

16-BIT TIMER (Cont'd)

7.3.3.3 Input Capture

In this section, the index, *i*, may be 1 or 2 because there are 2 input capture functions in the 16-bit timer.

The two input capture 16-bit registers (IC1R and IC2R) are used to latch the value of the free running counter after a transition is detected by the ICAP*i* pin (see figure 5).

	MS Byte	LS Byte
ICiR	IC*i*HR	IC*i*LR

The IC*i*R register is a read-only register.

The active transition is software programmable through the IEDG*i* bit of Control Registers (CR*i*).

Timing resolution is one count of the free running counter: (f_{CPU}/CC[1:0]).

Procedure:

To use the input capture function, select the following in the CR2 register:

– Select the timer clock (CC[1:0]) (see Table 18 Clock Control Bits).

– Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit (the ICAP2 pin must be configured as a floating input or input with pull-up without interrupt if this configuration is available).

And select the following in the CR1 register:

– Set the ICIE bit to generate an interrupt after an input capture coming from either the ICAP1 pin or the ICAP2 pin

– Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1 pin must be configured as a floating input or input with pull-up without interrupt if this configuration is available).

When an input capture occurs:

– The ICF*i* bit is set.

– The IC*i*R register contains the value of the free running counter on the active transition on the ICAP*i* pin (see Figure 35).

– A timer interrupt is generated if the ICIE bit is set and the I bit is cleared in the CC register. Otherwise, the interrupt remains pending until both conditions become true.

Clearing the Input Capture interrupt request (i.e. 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.

Notes:

1. After reading the IC*i*HR register, the transfer of input capture data is inhibited and ICF*i* will never be set until the IC*i*LR register is also read.
2. The IC*i*R register contains the free running counter value which corresponds to the most recent input capture.
3. The 2 input capture functions can be used together even if the timer also uses the 2 output compare functions.
4. In One Pulse mode and PWM mode only the input capture 2 function can be used.
5. The alternate inputs (ICAP1 & ICAP2) are always directly connected to the timer. So any transitions on these pins activate the input capture function.
   Moreover if one of the ICAP*i* pin is configured as an input and the second one as an output, an interrupt can be generated if the user toggles the output pin and if the ICIE bit is set.
   This can be avoided if the input capture function *i* is disabled by reading the IC*i*HR (see note 1).
6. The TOF bit can be used with an interrupt in order to measure events that exceed the timer range (FFFFh).

16-BIT TIMER (Cont'd)

Figure 34. Input Capture Block Diagram

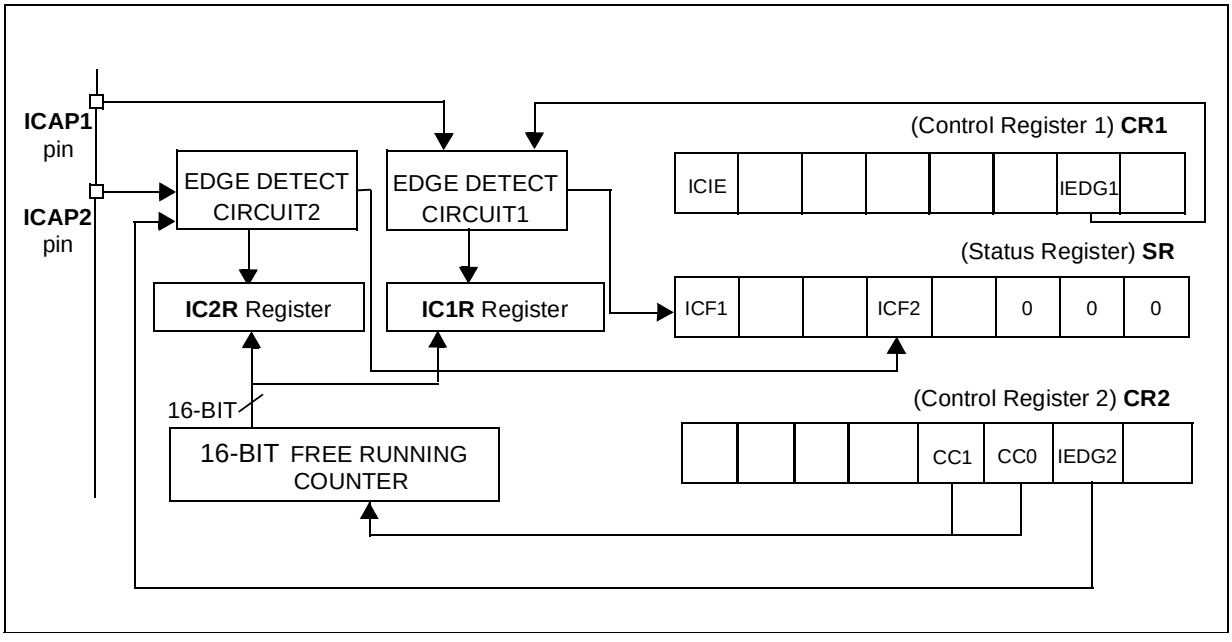
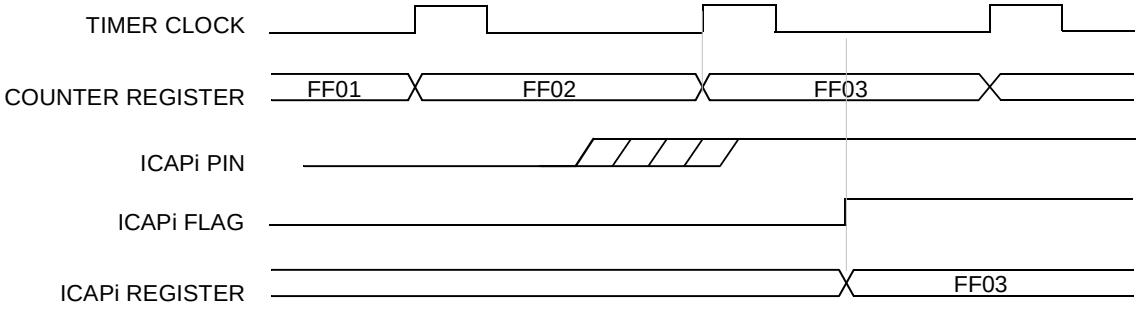


Figure 35. Input Capture Timing Diagram



Note: Active edge is rising edge.

16-BIT TIMER (Cont'd)

7.3.3.4 Output Compare

In this section, the index, *i*, may be 1 or 2 because there are 2 output compare functions in the 16-bit timer.

This function can be used to control an output waveform or indicate when a period of time has elapsed.

When a match is found between the Output Compare register and the free running counter, the output compare function:

- Assigns pins with a programmable value if the OC*i*E bit is set
- Sets a flag in the status register
- Generates an interrupt if enabled

Two 16-bit registers Output Compare Register 1 (OC1R) and Output Compare Register 2 (OC2R) contain the value to be compared to the counter register each timer clock cycle.

	MS Byte	LS Byte
OC*i*R	OC*i*HR	OC*i*LR

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC*i*R value to 8000h.

Timing resolution is one count of the free running counter: ($f_{CPU/CC[1:0]}$).

Procedure:

To use the output compare function, select the following in the CR2 register:

- Set the OC*i*E bit if an output is needed then the OCMP*i* pin is dedicated to the output compare *i* signal.
- Select the timer clock (CC[1:0]) (see Table 18 Clock Control Bits).

And select the following in the CR1 register:

- Select the OLVL*i* bit to applied to the OCMP*i* pins after the match occurs.
- Set the OCIE bit to generate an interrupt if it is needed.

When a match is found between OCRi register and CR register:

- OCF*i* bit is set.
- The OCMP*i* pin takes OLVL*i* bit value (OCMP*i* pin latch is forced low during reset).
- A timer interrupt is generated if the OCIE bit is set in the CR1 register and the I bit is cleared in the CC register (CC).

The OC*i*R register value required for a specific timing application can be calculated using the following formula:

$$\Delta \text{ OC}i\text{R} = \frac{\Delta t * f_{CPU}}{PRESC}$$

Where:

Δt = Output compare period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 18 Clock Control Bits)

If the timer clock is an external clock, the formula is:

$$\Delta \text{ OC}i\text{R} = \Delta t * f_{EXT}$$

Where:

Δt = Output compare period (in seconds)

f_{EXT} = External timer clock frequency (in hertz)

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

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

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

- Write to the OC*i*HR register (further compares are inhibited).
- Read the SR register (first step of the clearance of the OCF*i* bit, which may be already set).
- Write to the OC*i*LR register (enables the output compare function and clears the OCF*i* bit).

16-BIT TIMER (Cont'd)

Notes:

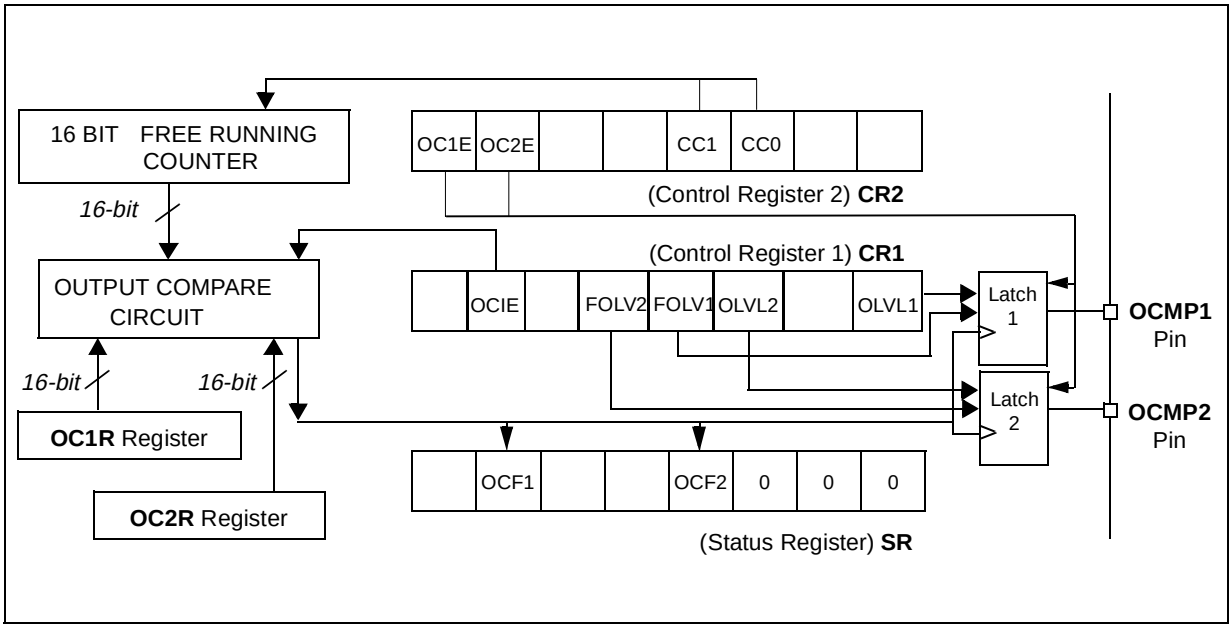
1. After a processor write cycle to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.
2. If the OC*i*E bit is not set, the OCMP*i* pin is a general I/O port and the OLVL*i* bit will not appear when a match is found but an interrupt could be generated if the OCIE bit is set.
3. When the timer clock is $f_{CPU}/2$, OCF*i* and OCMP*i* are set while the counter value equals the OC*i*R register value (see Figure 37). This behaviour is the same in OPM or PWM mode. When the timer clock is $f_{CPU}/4$, $f_{CPU}/8$ or in external clock mode, OCF*i* and OCMP*i* are set while the counter value equals the OC*i*R register value plus 1 (see Figure 38).
4. The output compare functions can be used both for generating external events on the OCMP*i* pins even if the input capture mode is also used.
5. The value in the 16-bit OC*i*R register and the OLV*i* bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout.

Forced Compare Output capability

When the FOLV*i* bit is set by software, the OLVL*i* bit is copied to the OCMP*i* pin. The OLV*i* bit has to be toggled in order to toggle the OCMP*i* pin when it is enabled (OC*i*E bit=1). The OCF*i* bit is then not set by hardware, and thus no interrupt request is generated.

FOLVL*i* bits have no effect in either One-Pulse mode or PWM mode.

Figure 36. Output Compare Block Diagram



16-BIT TIMER (Cont'd)

Figure 37. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/2$

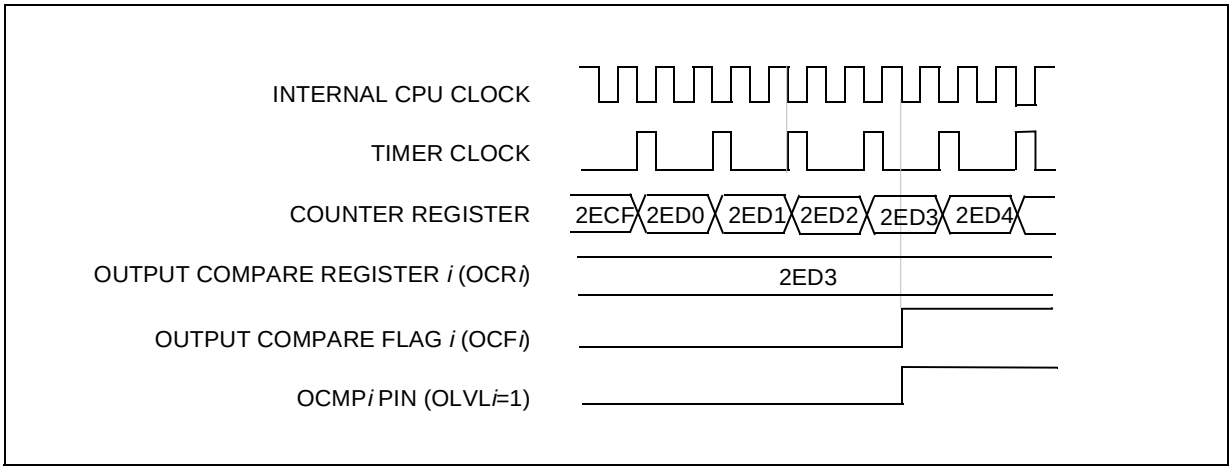
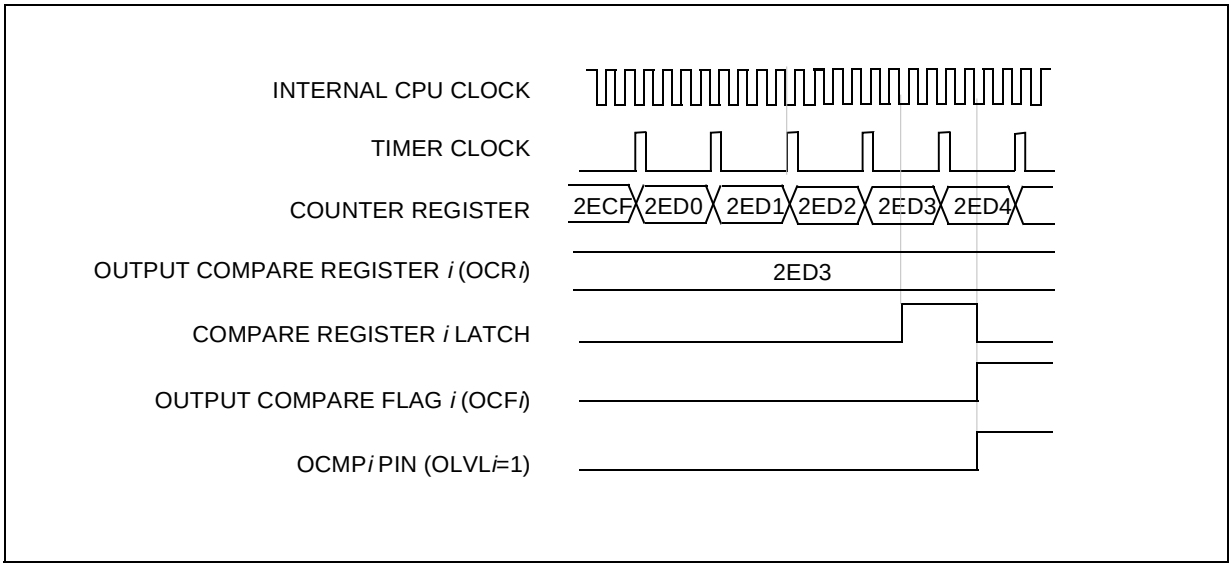


Figure 38. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/4$



16-BIT TIMER (Cont'd)

7.3.3.5 One Pulse Mode

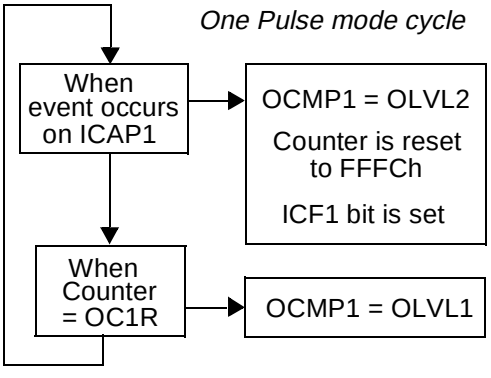
One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

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 18 Clock Control Bits).



Then, on a valid event on the ICAP1 pin, the counter is initialized to FFFCh and the 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 (i.e. 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} = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on the CC[1:0] bits, see Table 18 Clock Control Bits)

If the timer clock is an external clock the formula is:

$$OCiR = t * f_{EXT} - 5$$

Where:

t = Pulse period (in seconds)

f_{EXT} = External timer clock frequency (in hertz)

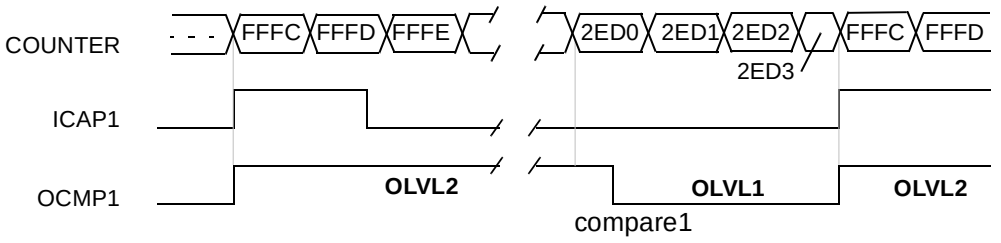
When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin (see Figure 39).

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 that a period of time has elapsed but cannot generate an output waveform because the OLVL2 level is dedicated to One Pulse mode.

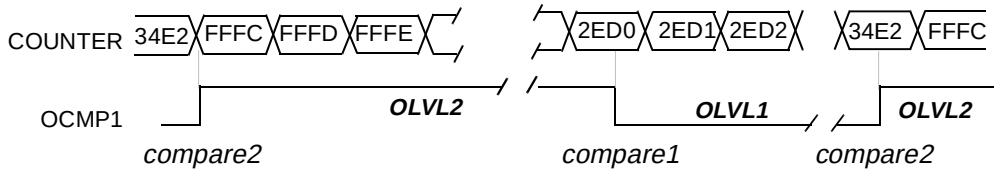
16-BIT TIMER (Cont'd)

Figure 39. One Pulse Mode Timing Example



Note: IEDG1=1, OC1R=2ED0h, OLVL1=0, OLVL2=1

Figure 40. Pulse Width Modulation Mode Timing Example



Note: OC1R=2ED0h, OC2R=34E2, OLVL1=0, OLVL2= 1

16-BIT TIMER (Cont'd)

7.3.3.6 Pulse Width Modulation Mode

Pulse Width Modulation (PWM) mode enables the generation of a signal with a frequency and pulse length determined by the value of the OC1R and OC2R registers.

The Pulse Width Modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so these functions cannot be used when the PWM mode is activated.

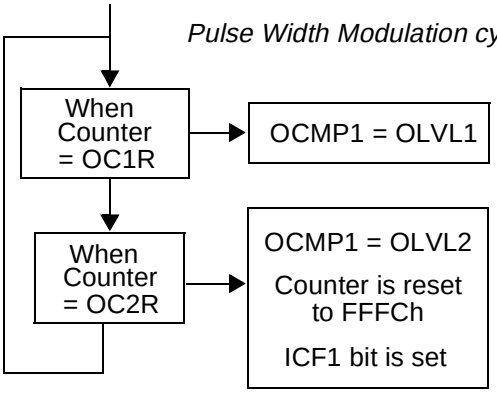
Procedure

To use Pulse Width Modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if OLVL1=0 and OLVL2=1, using the formula in the opposite column.
3. Select the following in the CR1 register:
   - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC1R register.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC2R register.
4. Select the following in the CR2 register:
   - Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.
   - Set the PWM bit.
   - Select the timer clock (CC[1:0]) (see Table 18 Clock Control Bits).

If OLVL1=1 and OLVL2=0, the length of the positive pulse is the difference between the OC2R and OC1R registers.

If OLVL1=OLVL2 a continuous signal will be seen on the OCMP1 pin.



The OC*i*R register value required for a specific timing application can be calculated using the following formula:

$$\text{OC}i\text{R Value} = \frac{t * f_{CPU}}{PRESC} - 5$$

Where:

t = Signal or pulse period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 18 Clock Control Bits)

If the timer clock is an external clock the formula is:

$$\text{OC}i\text{R} = t * f_{EXT} - 5$$

Where:

t = Signal or pulse period (in seconds)

f_{EXT} = External timer clock frequency (in hertz)

The Output Compare 2 event causes the counter to be initialized to FFFCh (See Figure 40)

Notes:

1. After a write instruction to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.
2. The OCF1 and OCF2 bits cannot be set by hardware in PWM mode, therefore the Output Compare interrupt is inhibited.
3. The ICF1 bit is set by hardware when the counter reaches the OC2R value and can produce a timer interrupt if the ICIE bit is set and the I bit is cleared.
4. In PWM mode the ICAP1 pin can not be used to perform input capture because it is disconnected from the timer. 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 after each period and ICF1 can also generate an interrupt if ICIE is set.
5. When the Pulse Width Modulation (PWM) and One Pulse mode (OPM) bits are both set, the PWM mode is the only active one.

16-BIT TIMER (Cont'd)

7.3.4 Low Power Modes

Mode	Description
WAIT	No effect on 16-bit Timer. Timer interrupts cause the device to exit from WAIT mode.
HALT	16-bit Timer registers are frozen. In HALT mode, the counter stops counting until Halt mode is exited. Counting resumes from the previous count when the MCU is woken up by an interrupt with "exit from HALT mode" capability or from the counter reset value when the MCU is woken up by a RESET. If an input capture event occurs on the ICAP*i* pin, the input capture detection circuitry is armed. Consequently, when the MCU is woken up by an interrupt with "exit from HALT mode" capability, the ICF*i* bit is set, and the counter value present when exiting from HALT mode is captured into the IC*i*R register.

7.3.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Input Capture 1 event/Counter reset in PWM mode	ICF1	ICIE	Yes	No
Input Capture 2 event	ICF2		Yes	No
Output Compare 1 event (not available in PWM mode)	OCF1	OCIE	Yes	No
Output Compare 2 event (not available in PWM mode)	OCF2		Yes	No
Timer Overflow event	TOF	TOIE	Yes	No

Note: The 16-bit Timer interrupt events are connected to the same interrupt vector (see Interrupts chapter). These events generate an interrupt if the corresponding Enable Control Bit is set and the interrupt mask in the CC register is reset (RIM instruction).

7.3.6 Summary of Timer modes

MODES	AVAILABLE RESOURCES			
	Input Capture 1	Input Capture 2	Output Compare 1	Output Compare 2
Input Capture (1 and/or 2)	Yes	Yes	Yes	Yes
Output Compare (1 and/or 2)	Yes	Yes	Yes	Yes
One Pulse mode	No	Not Recommended[1)]	No	Partially [2)]
PWM Mode	No	Not Recommended[3)]	No	No

1) See note 4 in Section 7.3.3.5 One Pulse Mode

2) See note 5 in Section 7.3.3.5 One Pulse Mode

3) See note 4 in Section 7.3.3.6 Pulse Width Modulation Mode

16-BIT TIMER (Cont'd)

7.3.7 Register Description

Each Timer is associated with three control and status registers, and with six pairs of data registers (16-bit values) relating to the two input captures, the two output compares, the counter and the alternate counter.

CONTROL REGISTER 1 (CR1)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1

Bit 7 = **ICIE** *Input Capture Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the ICF1 or ICF2 bit of the SR register is set.

Bit 6 = **OCIE** *Output Compare Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the OCF1 or OCF2 bit of the SR register is set.

Bit 5 = **TOIE** *Timer Overflow Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is enabled whenever the TOF bit of the SR register is set.

Bit 4 = **FOLV2** *Forced Output Compare 2.*
This bit is set and cleared by software.
0: No effect on the OCMP2 pin.
1: Forces the OLVL2 bit to be copied to the OCMP2 pin, if the OC2E bit is set and even if there is no successful comparison.

Bit 3 = **FOLV1** *Forced Output Compare 1.*
This bit is set and cleared by software.
0: No effect on the OCMP1 pin.
1: Forces OLVL1 to be copied to the OCMP1 pin, if the OC1E bit is set and even if there is no successful comparison.

Bit 2 = **OLVL2** *Output Level 2.*
This bit is copied to the OCMP2 pin whenever a successful comparison occurs with the OC2R register and OCxE is set in the CR2 register. This value is copied to the OCMP1 pin in One Pulse mode and Pulse Width Modulation mode.

Bit 1 = **IEDG1** *Input Edge 1.*
This bit determines which type of level transition on the ICAP1 pin will trigger the capture.
0: A falling edge triggers the capture.
1: A rising edge triggers the capture.

Bit 0 = **OLVL1** *Output Level 1.*
The OLVL1 bit is copied to the OCMP1 pin whenever a successful comparison occurs with the OC1R register and the OC1E bit is set in the CR2 register.

16-BIT TIMER (Cont'd)

CONTROL REGISTER 2 (CR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG

Bit 7 = **OC1E** *Output Compare 1 Pin Enable.*
This bit is used only to output the signal from the timer on the OCMP1 pin (OLV1 in Output Compare mode, both OLV1 and OLV2 in PWM and one-pulse mode). Whatever the value of the OC1E bit, the internal Output Compare 1 function of the timer remains active.
0: OCMP1 pin alternate function disabled (I/O pin free for general-purpose I/O).
1: OCMP1 pin alternate function enabled.

Bit 6 = **OC2E** *Output Compare 2 Pin Enable.*
This bit is used only to output the signal from the timer on the OCMP2 pin (OLV2 in Output Compare mode). Whatever the value of the OC2E bit, the internal Output Compare 2 function of the timer remains active.
0: OCMP2 pin alternate function disabled (I/O pin free for general-purpose I/O).
1: OCMP2 pin alternate function enabled.

Bit 5 = **OPM** *One Pulse mode.*
0: One Pulse mode is not active.
1: One Pulse mode is active, the ICAP1 pin can be used to trigger one pulse on the OCMP1 pin; the active transition is given by the IEDG1 bit. The length of the generated pulse depends on the contents of the OC1R register.

Bit 4 = **PWM** *Pulse Width Modulation.*
0: PWM mode is not active.
1: PWM mode is active, the OCMP1 pin outputs a programmable cyclic signal; the length of the pulse depends on the value of OC1R register; the period depends on the value of OC2R register.

Bits 3:2 = **CC[1:0]** *Clock Control.*

The timer clock mode depends on these bits:

Table 18. Clock Control Bits

Timer Clock	CC1	CC0
f_{CPU} / 4	0	0
f_{CPU} / 2	0	1
f_{CPU} / 8	1	0
External Clock (where available)	1	1

Note: If the external clock pin is not available, programming the external clock configuration stops the counter.

Bit 1 = **IEDG2** *Input Edge 2.*
This bit determines which type of level transition on the ICAP2 pin will trigger the capture.
0: A falling edge triggers the capture.
1: A rising edge triggers the capture.

Bit 0 = **EXEDG** *External Clock Edge.*
This bit determines which type of level transition on the external clock pin (EXTCLK) will trigger the counter register.
0: A falling edge triggers the counter register.
1: A rising edge triggers the counter register.

16-BIT TIMER (Cont'd)

STATUS REGISTER (SR)

Read Only

Reset Value: 0000 0000 (00h)

The three least significant bits are not used.

7							0
ICF1	OCF1	TOF	ICF2	OCF2	0	0	0

Bit 7 = **ICF1** *Input Capture Flag 1.*
0: No input capture (reset value).
1: An input capture has occurred on the ICAP1 pin or the counter has reached the OC2R value in PWM mode. To clear this bit, first read the SR register, then read or write the low byte of the IC1R (IC1LR) register.

Bit 6 = **OCF1** *Output Compare Flag 1.*
0: No match (reset value).
1: The content of the free running counter matches the content of the OC1R register. To clear this bit, first read the SR register, then read or write the low byte of the OC1R (OC1LR) register.

Bit 5 = **TOF** *Timer Overflow Flag.*
0: No timer overflow (reset value).
1: The free running counter has rolled over from FFFFh to 0000h. To clear this bit, first read the SR register, then read or write the low byte of the CR (CLR) register.

Note: Reading or writing the ACLR register does not clear TOF.

Bit 4 = **ICF2** *Input Capture Flag 2.*
0: No input capture (reset value).
1: An input capture has occurred on the ICAP2 pin. To clear this bit, first read the SR register, then read or write the low byte of the IC2R (IC2LR) register.

Bit 3 = **OCF2** *Output Compare Flag 2.*
0: No match (reset value).
1: The content of the free running counter matches the content of the OC2R register. To clear this bit, first read the SR register, then read or write the low byte of the OC2R (OC2LR) register.

Bit 2-0 = Reserved, forced by hardware to 0.

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

16-BIT TIMER (Cont'd)

OUTPUT COMPARE 2 HIGH REGISTER (OC2HR)

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 2 LOW REGISTER (OC2LR)

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

COUNTER HIGH REGISTER (CHR)

Read Only
Reset Value: 1111 1111 (FFh)

This is an 8-bit register that contains the high part of the counter value.

7							0
MSB							LSB

COUNTER LOW REGISTER (CLR)

Read Only
Reset Value: 1111 1100 (FCh)

This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after accessing the SR register clears the TOF bit.

7							0
MSB							LSB

ALTERNATE COUNTER HIGH REGISTER (ACHR)

Read Only
Reset Value: 1111 1111 (FFh)

This is an 8-bit register that contains the high part of the counter value.

7							0
MSB							LSB

ALTERNATE COUNTER LOW REGISTER (ACLR)

Read Only
Reset Value: 1111 1100 (FCh)

This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after an access to SR register does not clear the TOF bit in SR register.

7							0
MSB							LSB

INPUT CAPTURE 2 HIGH REGISTER (IC2HR)

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 2 event).

7							0
MSB							LSB

INPUT CAPTURE 2 LOW REGISTER (IC2LR)

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 2 event).

7							0
MSB							LSB

16-BIT TIMER (Cont'd)

Table 19. 16-Bit Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Timer A: 32 Timer B: 42	**CR1** Reset Value	ICIE 0	OCIE 0	TOIE 0	FOLV2 0	FOLV1 0	OLVL2 0	IEDG1 0	OLVL1 0
Timer A: 31 Timer B: 41	**CR2** Reset Value	OC1E 0	OC2E 0	OPM 0	PWM 0	CC1 0	CC0 0	IEDG2 0	EXEDG 0
Timer A: 33 Timer B: 43	**SR** Reset Value	ICF1 0	OCF1 0	TOF 0	ICF2 0	OCF2 0	- 0	- 0	- 0
Timer A: 34 Timer B: 44	**ICHR1** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 35 Timer B: 45	**ICLR1** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 36 Timer B: 46	**OCHR1** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 37 Timer B: 47	**OCLR1** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3E Timer B: 4E	**OCHR2** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3F Timer B: 4F	**OCLR2** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 38 Timer B: 48	**CHR** Reset Value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 39 Timer B: 49	**CLR** Reset Value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3A Timer B: 4A	**ACHR** Reset Value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 3B Timer B: 4B	**ACLR** Reset Value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3C Timer B: 4C	**ICHR2** Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3D Timer B: 4D	**ICLR2** Reset Value	MSB -	-	-	-	-	-	-	LSB -

7.4 PWM/BRM GENERATOR (DAC)

7.4.1 Introduction

This PWM/BRM peripheral includes a 6-bit Pulse Width Modulator (PWM) and a 4-bit Binary Rate Multiplier (BRM) Generator. It allows the digital to analog conversion (DAC) when used with external filtering.

Note: The number of PWM and BRM channels available depends on the device. Refer to the device pin description and register map.

7.4.2 Main Features

- Fixed frequency: $f_{CPU}/64$
- Resolution: T_{CPU}
- Steps of $V_{DD}/2^{10}$ (5mV if V_{DD}=5V)

7.4.3 Functional Description

The 10 bits of the 10-bit PWM/BRM are distributed as 6 PWM bits and 4 BRM bits. The generator consists of a 10-bit counter (common for all channels), a comparator and the PWM/BRM generation logic.

PWM Generation

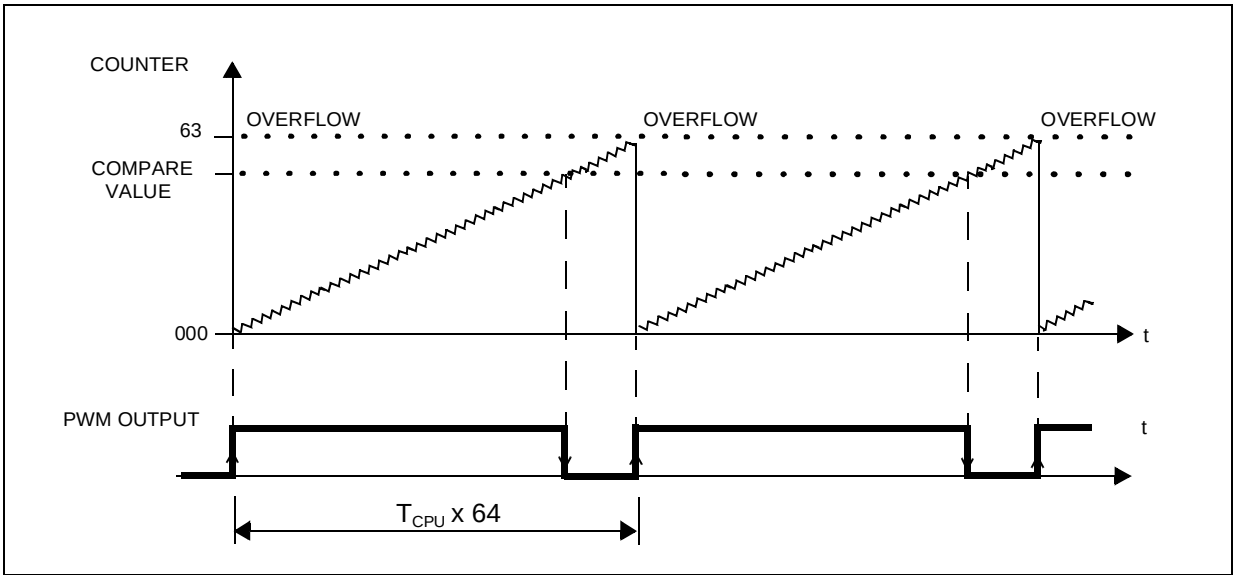
The counter increments continuously, clocked at internal CPU clock. Whenever the 6 least significant bits of the counter (defined as the PWM counter) overflow, the output level for all active channels is set.

The state of the PWM counter is continuously compared to the PWM binary weight for each channel, as defined in the relevant PWM register, and when a match occurs the output level for that channel is reset.

This Pulse Width modulated signal must be filtered, using an external RC network placed as close as possible to the associated pin. This provides an analog voltage proportional to the average charge passed to the external capacitor. Thus for a higher mark/space ratio (high time much greater than low time) the average output voltage is higher. The external components of the RC network should be selected for the filtering level required for control of the system variable.

Each output may individually have its polarity inverted by software, and can also be used as a logical output.

Figure 41. PWM Generation



PWM/BRM GENERATOR (Cont'd)

PWM/BRM Outputs

The PWM/BRM outputs are assigned to dedicated pins.
In these pins, the PWM/BRM outputs are connected to a serial resistor which must be taken into account to calculate the RC filter (see Figure 42).
In any case, the RC filter time must be higher than T_{CPU}x64.

Figure 42. Typical PWM Output Filter

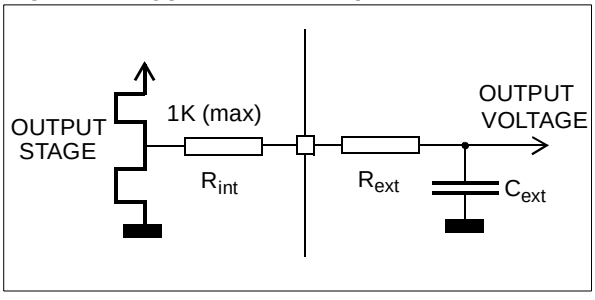


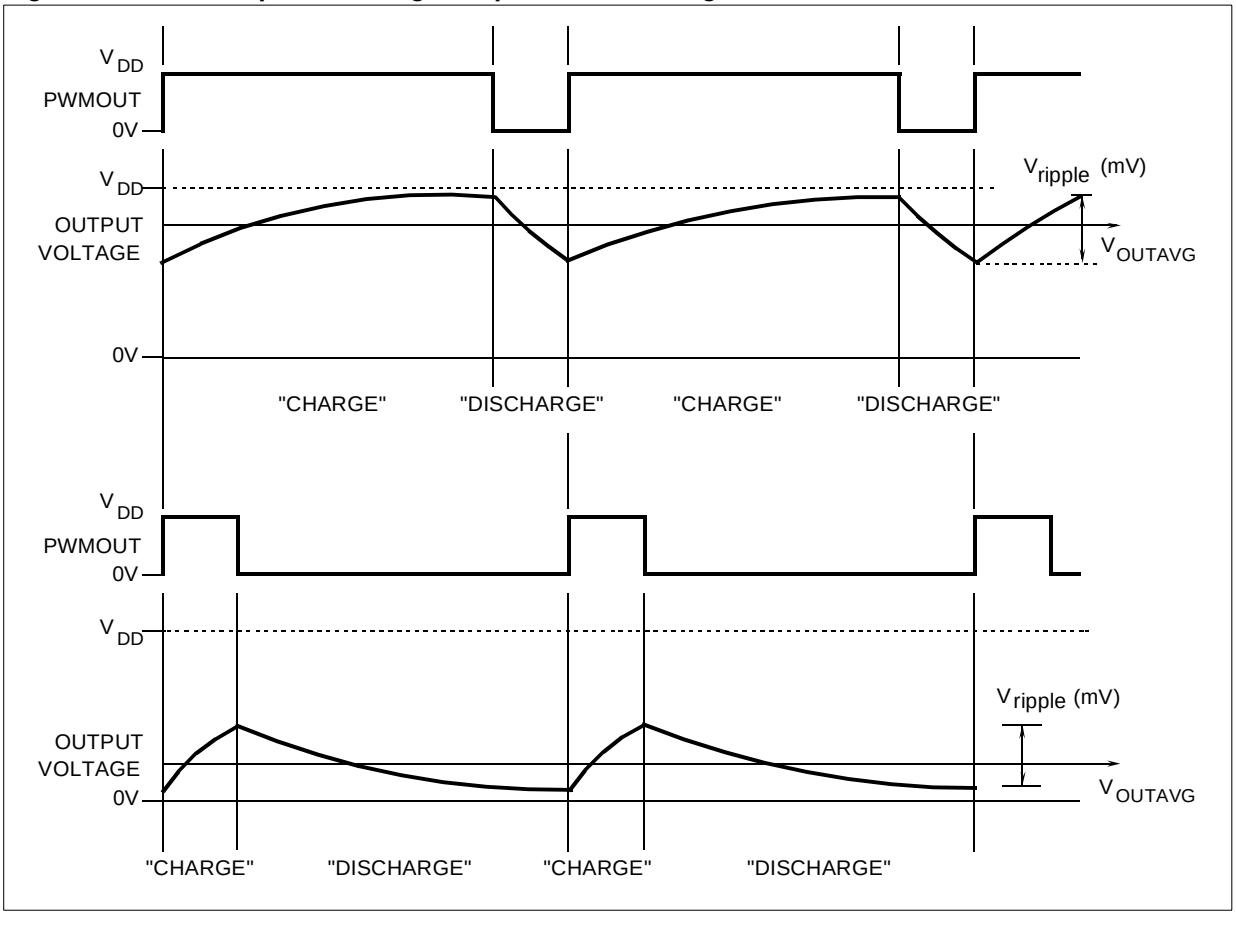
Table 20. 6-Bit PWM Ripple After Filtering

Cext (µF)	V RIPPLE (mV)
0.128	78
1.28	7.8
12.8	0.78

With RC filter (R=1KΩ),
f_{CPU} = 8 MHz
V_{DD} = 5V
PWM Duty Cycle 50%
$R=R_{int}+R_{ext}$ (Rext is optional).

Note: after a reset these pins are tied low by default and are not in a high impedance state.

Figure 43. PWM Simplified Voltage Output After Filtering



PWM/BRM GENERATOR (Cont'd)

BRM Generation

The BRM bits allow the addition of a pulse to widen a standard PWM pulse for specific PWM cycles. This has the effect of "fine-tuning" the PWM Duty cycle (without modifying the base duty cycle), thus, with the external filtering, providing additional fine voltage steps.

The incremental pulses (with duration of T_{CPU}) are added to the beginning of the original PWM pulse. The PWM intervals which are added to are specified in the 4-bit BRM register and are encoded as shown in the following table. The BRM values shown may be combined together to provide a summation of the incremental pulse intervals specified.

The pulse increment corresponds to the PWM resolution.

For example,if

– Data 18h is written to the PWM register

– Data 06h (00000110b) is written to the BRM register

– with a 8MHz internal clock (125ns resolution)

Then 3.0 µs-long pulse will be output at 8 µs intervals, except for cycles numbered 2,4,6,10,12,14, where the pulse is broadened to 3.125 µs.

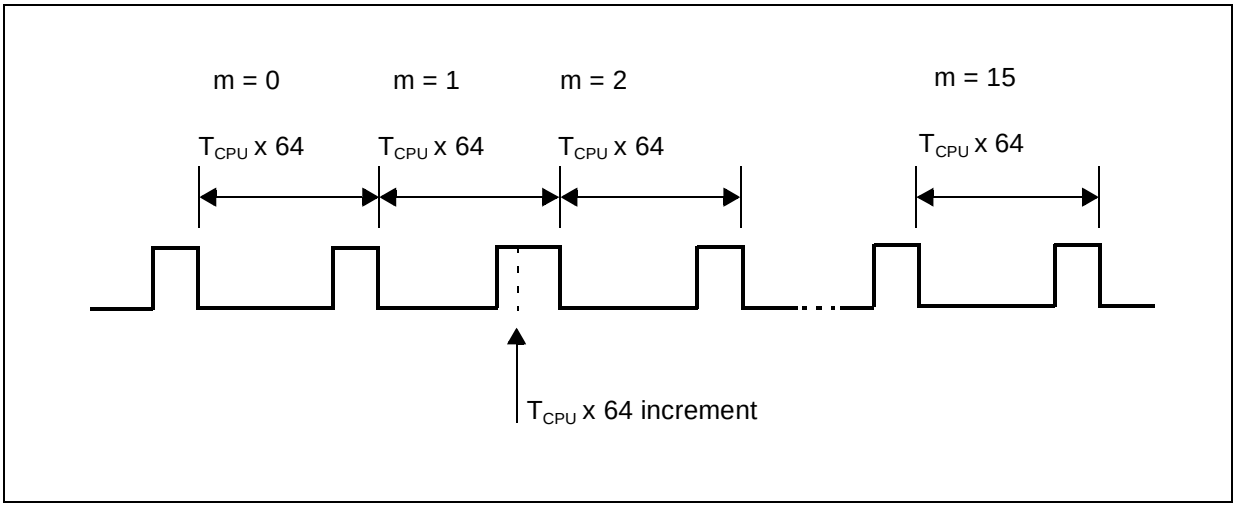
Note. If 00h is written to both PWM and BRM registers, the generator output will remain at "0". Conversely, if both registers hold data 3Fh and 0Fh, respectively, the output will remain at "1" for all intervals 1 to 15, but it will return to zero at interval 0 for an amount of time corresponding to the PWM resolution (T_{CPU}).

An output can be set to a continuous "1" level by clearing the PWM and BRM values and setting POL = "1" (inverted polarity) in the PWM register. This allows a PWM/BRM channel to be used as an additional I/O pin if the DAC function is not required.

Table 21. Bit BRM Added Pulse Intervals (Interval #0 not selected).

BRM 4 - Bit Data	Incremental Pulse Intervals
0000	none
0001	i = 8
0010	i = 4,12
0100	i = 2,6,10,14
1000	i = 1,3,5,7,9,11,13,15

Figure 44. BRM pulse addition (PWM > 0)



PWM/BRM GENERATOR (Cont'd)

Figure 45. Simplified Filtered Voltage Output Schematic with BRM Added

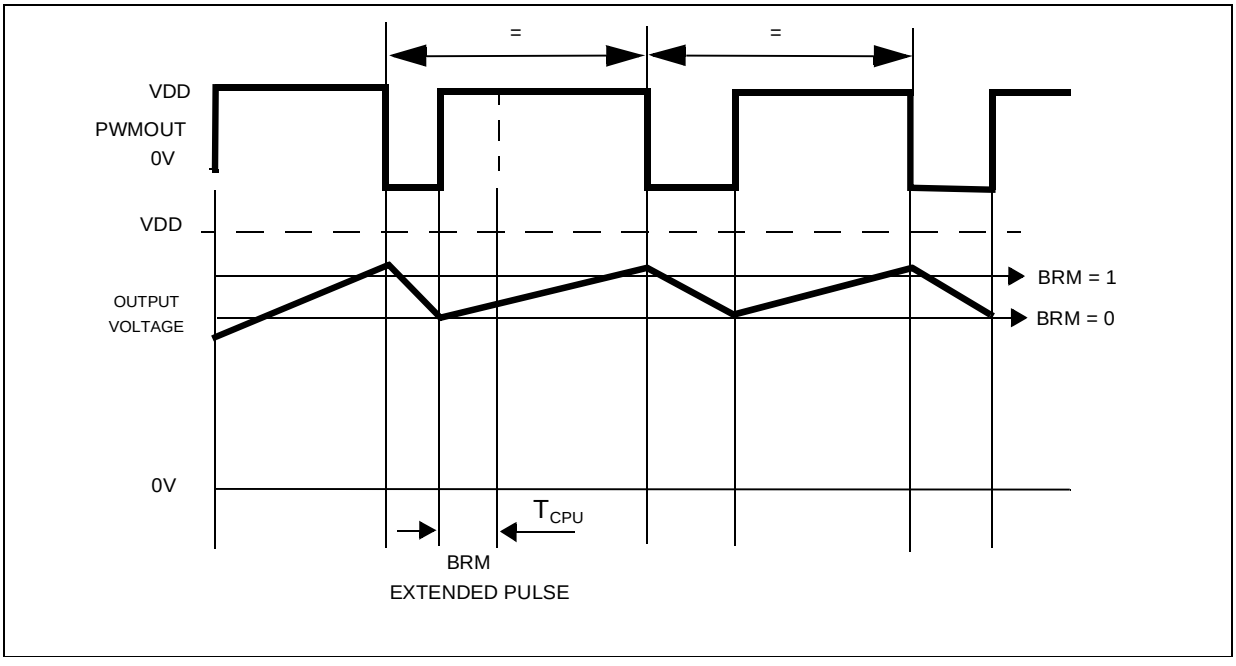
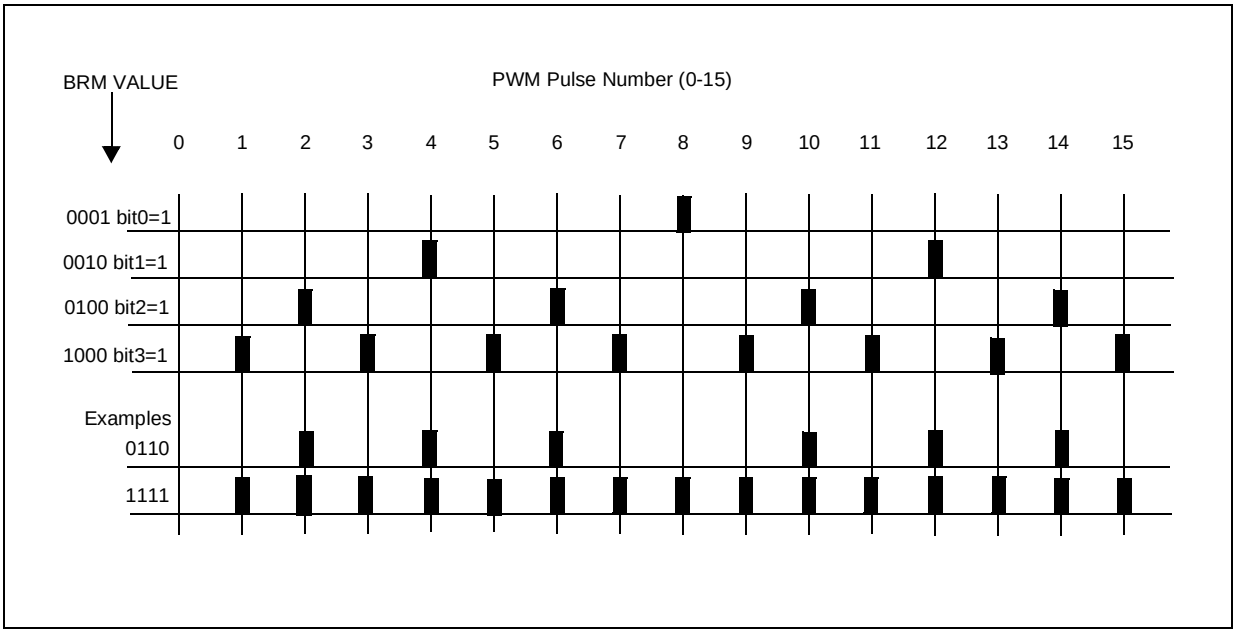
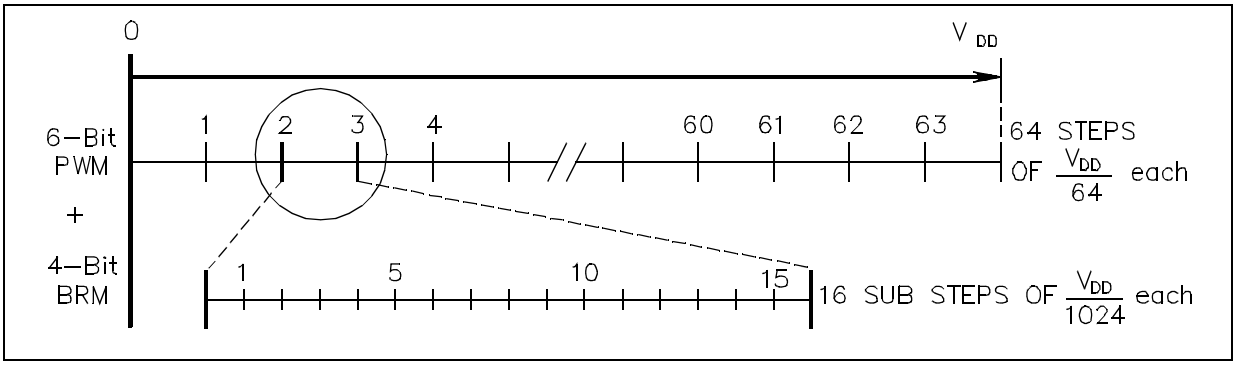


Figure 46. Graphical Representation of 4-Bit BRM Added Pulse Positions



PWM/BRM GENERATOR (Cont'd)

Figure 47. Precision for PWM/BRM Tuning for VOUTEFF (After filtering)



7.4.4 Register Description

On a channel basis, the 10 bits are separated into two data registers:

Note: The number of PWM and BRM channels available depends on the device. Refer to the device pin description and register map.

PULSE BINARY WEIGHT REGISTERS (PWMi)

Read / Write
Reset Value 1000 0000 (80h)

7							0
1	POL	P5	P4	P3	P2	P1	P0

Bit 7 = Reserved (Forced by hardware to "1")

Bit 6 = **POL** *Polarity Bit for channel i.*
0: The channel *i* outputs a "1" level during the binary pulse and a "0" level after.
1: The channel *i* outputs a "0" level during the binary pulse and a "1" level after.

Bit 5:0 = **P[5:0]** *PWM Pulse Binary Weight for channel i.*

This register contains the binary value of the pulse.

BRM REGISTERS

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

7							0
B7	B6	B5	B4	B3	B2	B1	B0

These registers define the intervals where an incremental pulse is added to the beginning of the original PWM pulse. Two BRM channel values share the same register.

Bit 7:4 = **B[7:4]** *BRM Bits (channel i+1).*

Bit 3:0 = **B[3:0]** *BRM Bits (channel i)*

Note: From the programmer's point of view, the PWM and BRM registers can be regarded as being combined to give one data value.

For example :

1	POL	P	P	P	P	P	P	+	B	B	B	B

Effective (with external RC filtering) DAC value

1	POL	P	P	P	P	P	P	B	B	B	B

PWM/BRM GENERATOR (Cond't)

Table 22. PWM Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
74	**PWM0**		POL	P5	P4	P3	P2	P1	P0
	Reset Value	1	0	0	0	0	0	0	0
75	**BRM10**	B7	B6	B5	B4	B3	B2	B1	B0
	Reset Value	0	0	0	0	0	0	0	0
76	**PWM1**		POL	P5	P4	P3	P2	P1	P0
	Reset Value	1	0	0	0	0	0	0	0
77	**PWM2**		POL	P5	P4	P3	P2	P1	P0
	Reset Value	1	0	0	0	0	0	0	0
78	**BRM32**	B7	B6	B5	B4	B3	B2	B1	B0
	Reset Value	0	0	0	0	0	0	0	0
79	**PWM3**		POL	P5	P4	P3	P2	P1	P0
	Reset Value	1	0	0	0	0	0	0	0

7.5 SERIAL PERIPHERAL INTERFACE (SPI)

7.5.1 Introduction

The Serial Peripheral Interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

The SPI is normally used for communication between the microcontroller and external peripherals or another microcontroller.

Refer to the Pin Description chapter for the device-specific pin-out.

7.5.2 Main Features

- Full duplex, three-wire synchronous transfers
- Master or slave operation
- Four master mode frequencies
- Maximum slave mode frequency = $f_{CPU}/4$.
- Four programmable master bit rates
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision flag protection
- Master mode fault protection capability.

7.5.3 General description

The SPI is connected to external devices through 4 alternate pins:

- MISO: Master In Slave Out pin
- MOSI: Master Out Slave In pin
- SCK: Serial Clock pin
- $\overline{SS}$: Slave select pin

A basic example of interconnections between a single master and a single slave is illustrated on Figure 48.

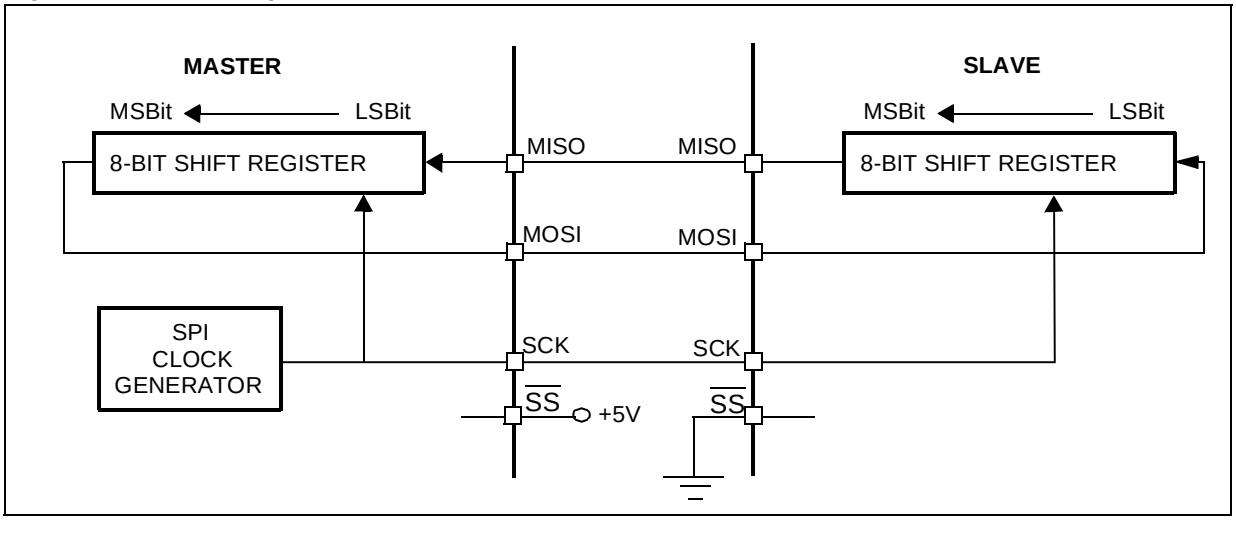
The MOSI pins are connected together as are MISO pins. In this way data is transferred serially between master and slave (most significant bit first).

When the master device transmits data to a slave device via MOSI pin, the slave device responds by sending data to the master device via the MISO pin. This implies full duplex transmission with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

Thus, the byte transmitted is replaced by the byte received and eliminates the need for separate transmit-empty and receiver-full bits. A status flag is used to indicate that the I/O operation is complete.

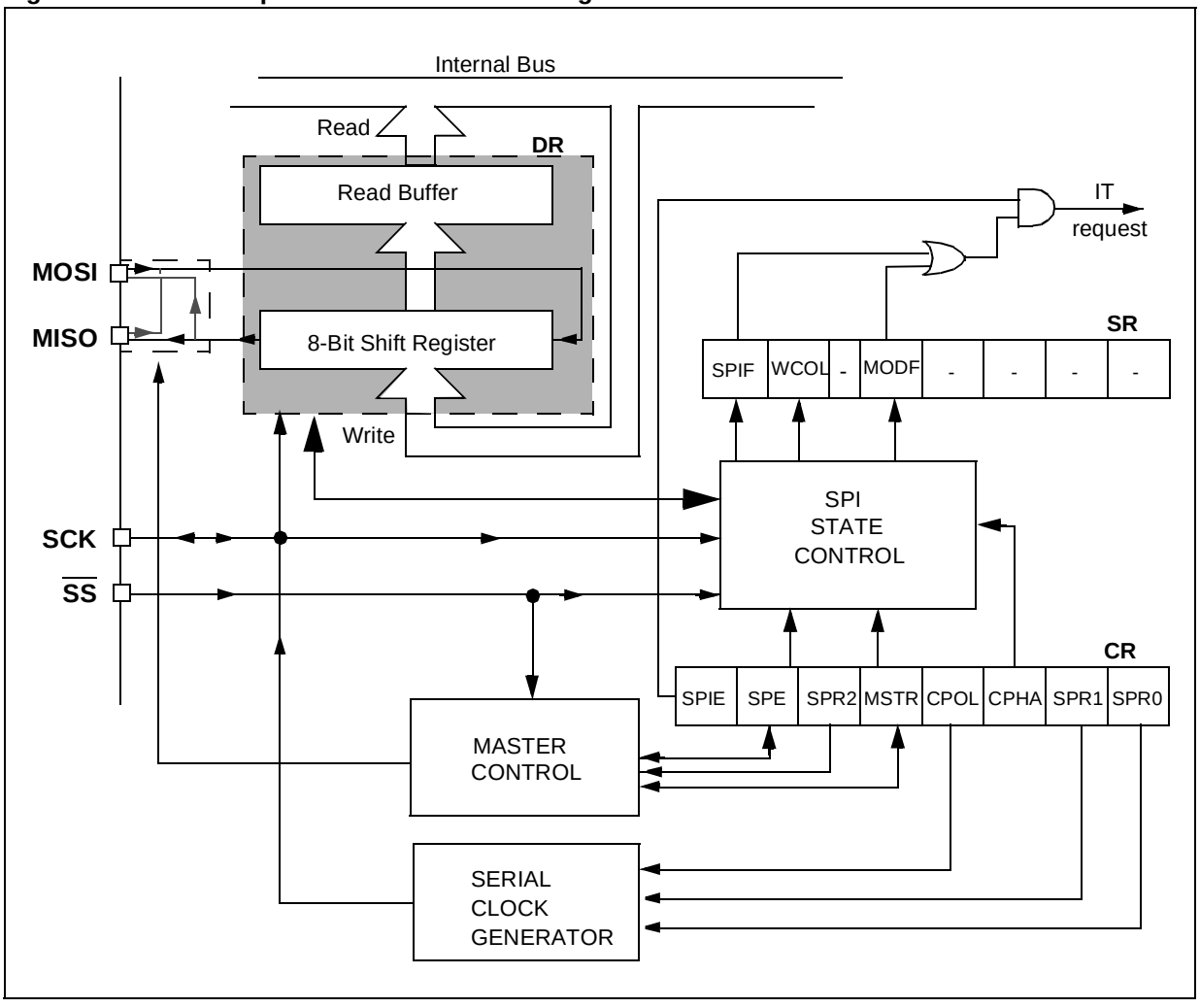
Four possible data/clock timing relationships may be chosen (see Figure 51) but master and slave must be programmed with the same timing mode.

Figure 48. Serial Peripheral Interface Master/Slave



SERIAL PERIPHERAL INTERFACE (Cont'd)

Figure 49. Serial Peripheral Interface Block Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.4 Functional Description

Figure 48 shows the serial peripheral interface (SPI) block diagram.

This interface contains 3 dedicated registers:

- A Control Register (CR)
- A Status Register (SR)
- A Data Register (DR)

Refer to the CR, SR and DR registers in Section 7.5.7for the bit definitions.

7.5.4.1 Master Configuration

In a master configuration, the serial clock is generated on the SCK pin.

Procedure

- Select the SPR0 & SPR1 bits to define the serial clock baud rate (see CR register).
- Select the CPOL and CPHA bits to define one of the four relationships between the data transfer and the serial clock (see Figure 51).
- The $\overline{SS}$ pin must be connected to a high level signal during the complete byte transmit sequence.
- The MSTR and SPE bits must be set (they remain set only if the $\overline{SS}$ pin is connected to a high level signal).

In this configuration the MOSI pin is a data output and to the MISO pin is a data input.

Transmit sequence

The transmit sequence begins when a byte is written the DR register.

The data byte is parallel loaded into the 8-bit shift register (from the internal bus) during a write cycle and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware
- An interrupt is generated if the SPIE bit is set and the I bit in the CCR register is cleared.

During the last clock cycle the SPIF bit is set, a copy of the data byte received in the shift register is moved to a buffer. When the DR register is read, the SPI peripheral returns this buffered value.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SR register while the SPIF bit is set
2. A read to the DR register.

Note: While the SPIF bit is set, all writes to the DR register are inhibited until the SR register is read.

SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.4.2 Slave Configuration

In slave configuration, the serial clock is received on the SCK pin from the master device.

The value of the SPR0 & SPR1 bits is not used for the data transfer.

Procedure

– For correct data transfer, the slave device must be in the same timing mode as the master device (CPOL and CPHA bits). See Figure 51.

– The $\overline{SS}$ pin must be connected to a low level signal during the complete byte transmit sequence.

– Clear the MSTR bit and set the SPE bit to assign the pins to alternate function.

In this configuration the MOSI pin is a data input and the MISO pin is a data output.

Transmit Sequence

The data byte is parallel loaded into the 8-bit shift register (from the internal bus) during a write cycle and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

– The SPIF bit is set by hardware

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

During the last clock cycle the SPIF bit is set, a copy of the data byte received in the shift register is moved to a buffer. When the DR register is read, the SPI peripheral returns this buffered value.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SR register while the SPIF bit is set.

2.A read to the DR register.

Notes: While the SPIF bit is set, all writes to the DR register are inhibited until the SR register is read.

The SPIF bit can be cleared during a second transmission; however, it must be cleared before the second SPIF bit in order to prevent an overrun condition (see Section 7.5.4.6).

Depending on the CPHA bit, the $\overline{SS}$ pin has to be set to write to the DR register between each data byte transfer to avoid a write collision (see Section 7.5.4.4).

SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.4.3 Data Transfer Format

During an SPI transfer, data is simultaneously transmitted (shifted out serially) and received (shifted in serially). The serial clock is used to synchronize the data transfer during a sequence of eight clock pulses.

The $\overline{SS}$ pin allows individual selection of a slave device; the other slave devices that are not selected do not interfere with the SPI transfer.

Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits.

The CPOL (clock polarity) bit controls the steady state value of the clock when no data is being transferred. This bit affects both master and slave modes.

The combination between the CPOL and CPHA (clock phase) bits selects the data capture clock edge.

Figure 51, shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin, the MOSI pin are directly connected between the master and the slave device.

The $\overline{SS}$ pin is the slave device select input and can be driven by the master device.

The master device applies data to its MOSI pin-clock edge before the capture clock edge.

CPHA bit is set

The second edge on the SCK pin (falling edge if the CPOL bit is reset, rising edge if the CPOL bit is set) is the MSBit capture strobe. Data is latched on the occurrence of the second clock transition.

No write collision should occur even if the $\overline{SS}$ pin stays low during a transfer of several bytes (see Figure 50).

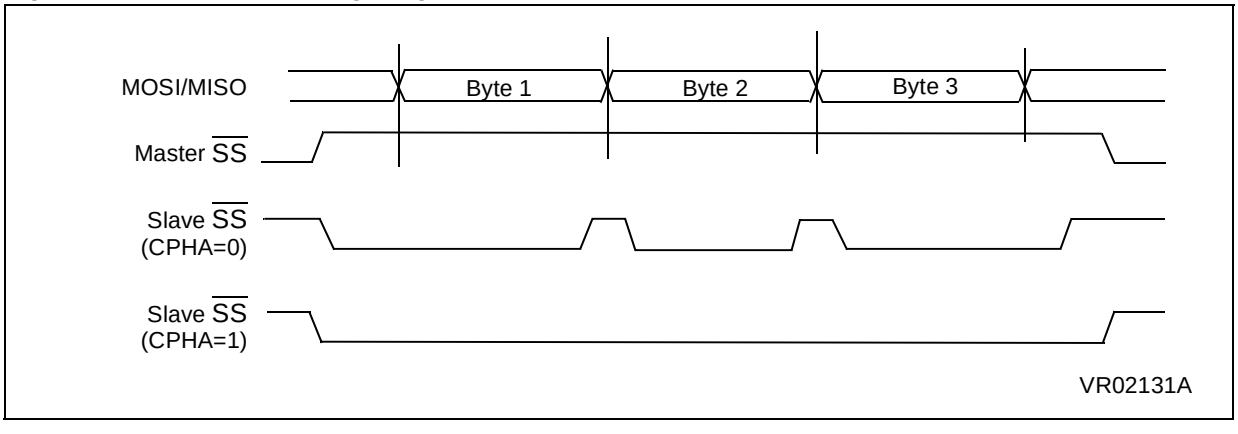
CPHA bit is reset

The first edge on the SCK pin (falling edge if CPOL bit is set, rising edge if CPOL bit is reset) is the MSBit capture strobe. Data is latched on the occurrence of the first clock transition.

The $\overline{SS}$ pin must be toggled high and low between each byte transmitted (see Figure 50).

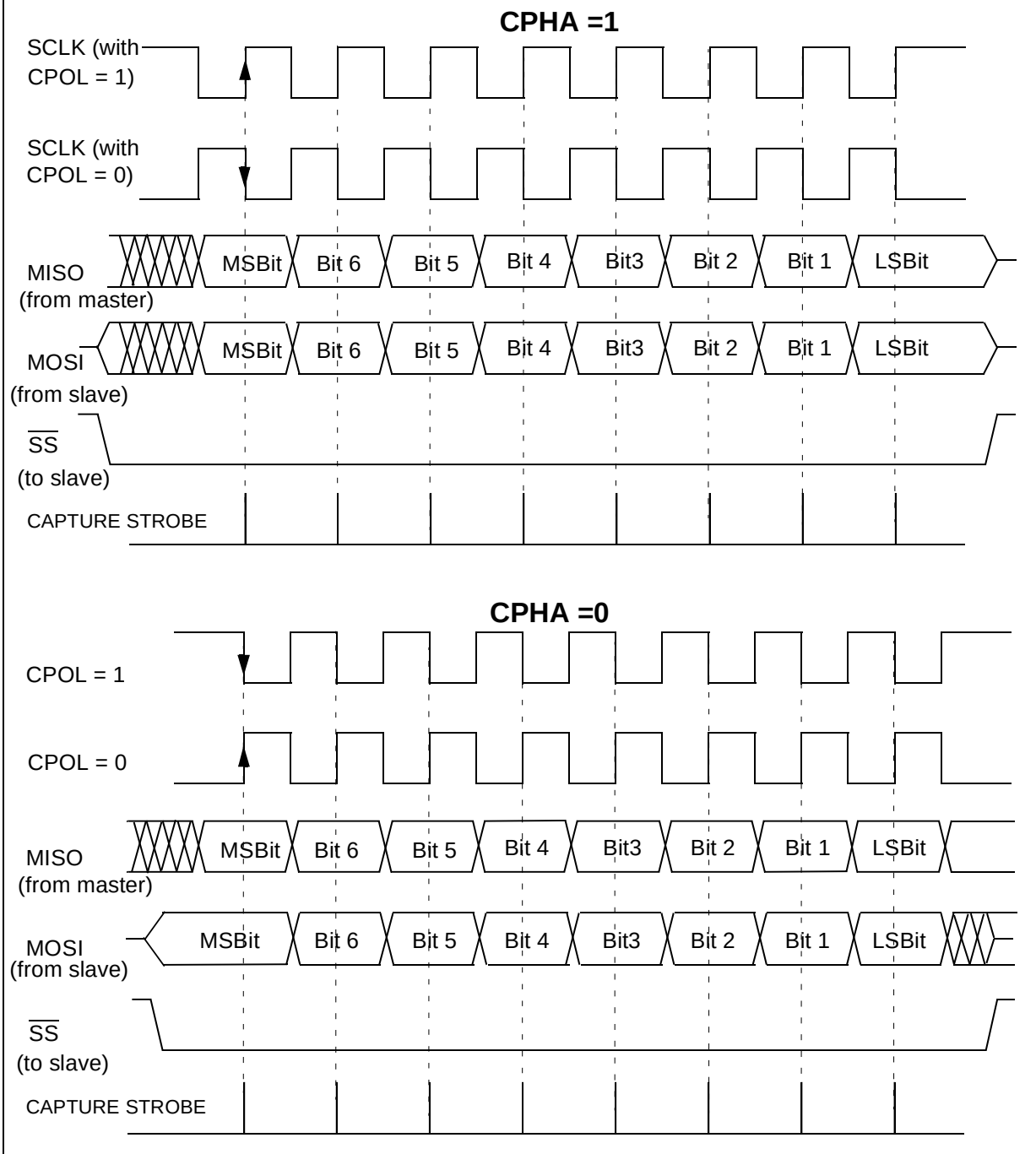
To protect the transmission from a write collision a low value on the $\overline{SS}$ pin of a slave device freezes the data in its DR register and does not allow it to be altered. Therefore the $\overline{SS}$ pin must be high to write a new data byte in the DR without producing a write collision.

Figure 50. CPHA / $\overline{SS}$ Timing Diagram



SERIAL PERIPHERAL INTERFACE (Cont'd)

Figure 51. Data Clock Timing Diagram



Note: This figure should not be used as a replacement for parametric information. Refer to the Electrical Characteristics chapter.

VR02131B

SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.4.4 Write Collision Error

A write collision occurs when the software tries to write to the DR 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.

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 MCU operation.

In Slave mode

When the CPHA bit is set:

The slave device will receive a clock (SCK) edge prior to the latch of the first data transfer. This first clock edge will freeze the data in the slave device DR register and output the MSBit on to the external MISO pin of the slave device.

The $\overline{SS}$ pin low state enables the slave device but the output of the MSBit onto the MISO pin does not take place until the first data transfer clock edge.

When the CPHA bit is reset:

Data is latched on the occurrence of the first clock transition. The slave device does not have any way of knowing when that transition will occur; therefore, the slave device collision occurs when software attempts to write the DR register after its $\overline{SS}$ pin has been pulled low.

For this reason, the $\overline{SS}$ pin must be high, between each data byte transfer, to allow the CPU to write in the DR register without generating a write collision.

In Master mode

Collision in the master device is defined as a write of the DR register while the internal serial clock (SCK) is in the process of transfer.

The $\overline{SS}$ pin signal must be always high on the master device.

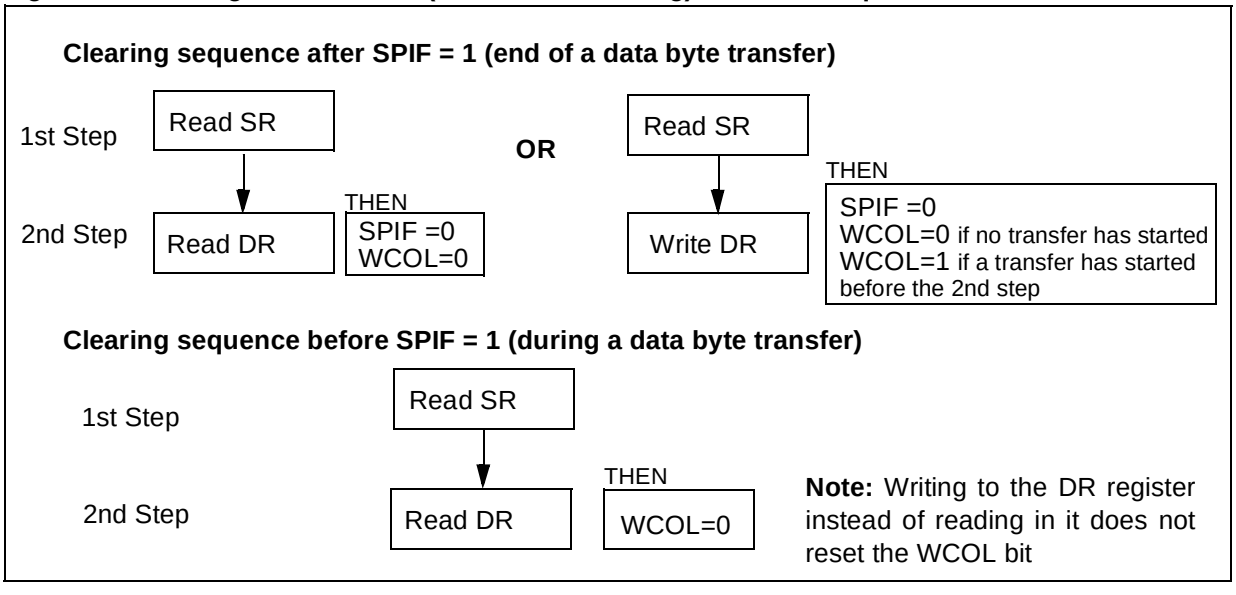
WCOL bit

The WCOL bit in the SR 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 52).

Figure 52. Clearing the WCOL bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.4.5 Master Mode Fault

Master mode fault occurs when the master device has its SS pin pulled low, then the MODF bit is set.

Master mode fault affects the SPI peripheral in the following ways:

- The MODF bit is set and an SPI interrupt 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 or write access to the SR register while the MODF bit is set.
2. A write to the CR register.

Notes: To avoid any multiple slave conflicts in the case of a system comprising several MCUs, the SS pin must be pulled high during the clearing sequence of the MODF bit. 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 multi master configuration the device can be in slave mode with this MODF bit set.

The MODF bit indicates that there might have been a multi-master conflict for system control and allows a proper exit from system operation to a reset or default system state using an interrupt routine.

7.5.4.6 Overrun Condition

An overrun condition occurs when the master device has sent several data bytes and the slave device has not cleared the SPIF bit issuing from the previous data byte transmitted.

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

This condition is not detected by the SPI peripheral.

SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.4.7 Single Master and Multimaster Configurations

There are two types of SPI systems:

– Single Master System

– Multimaster System

Single Master System

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see Figure 53).

The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices.

The $\overline{SS}$ pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

Note: To prevent a bus conflict on the MISO line the master allows only one active slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written its DR register.

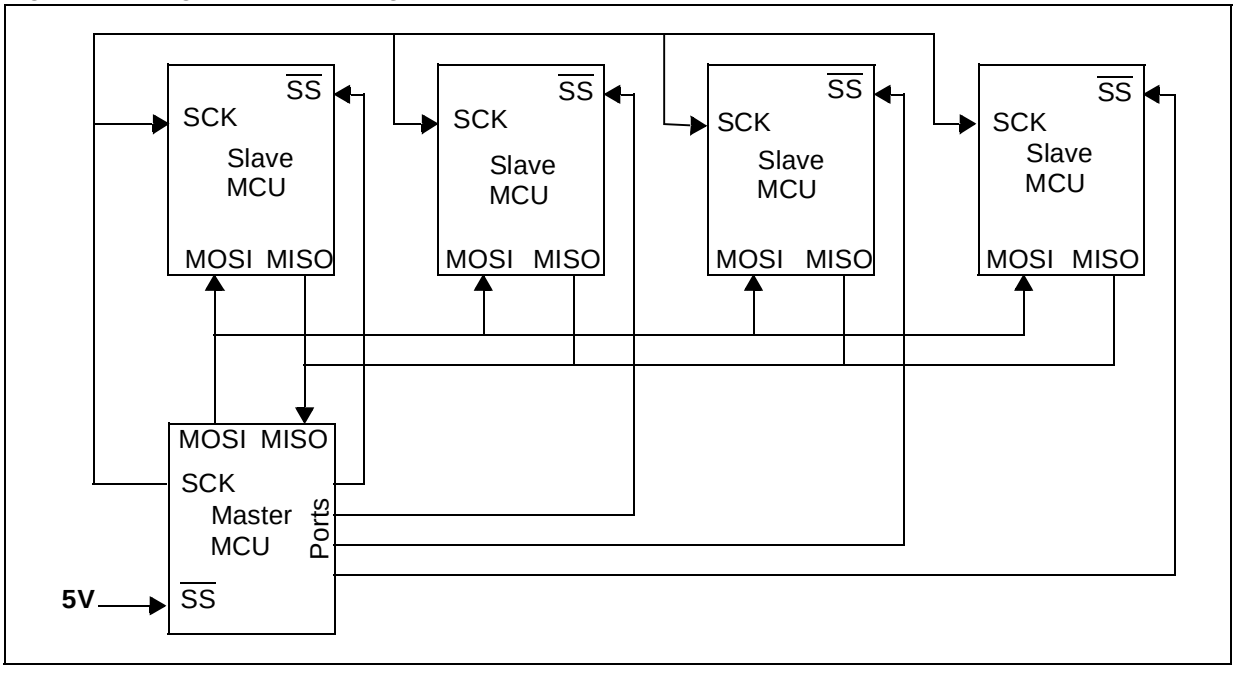
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Multi-master System

A multi-master system may also be configured by the user. Transfer of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system.

The multi-master system is principally handled by the MSTR bit in the CR register and the MODF bit in the SR register.

Figure 53. Single Master Configuration



SERIAL PERIPHERAL INTERFACE (Cont'd)

7.5.5 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 MCU is woken up by an interrupt with "exit from HALT mode" capability.

7.5.6 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
SPI End of Transfer Event	SPIF	SPIE	Yes	No
Master Mode Fault Event	MODF		Yes	No

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)

7.5.7 Register Description

CONTROL REGISTER (CR)

Read/Write

Reset Value: 0000xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPIE** *Serial peripheral interrupt enable.*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SPI interrupt is generated whenever SPIF=1 or MODF=1 in the SR register

Bit 6 = **SPE** *Serial peripheral output enable.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, SS=0 (see Section 7.5.4.5 Master Mode Fault).
0: I/O port connected to pins
1: SPI alternate functions connected to pins

The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.

Bit 5 = **SPR2** *Divider Enable*.

this bit is set and cleared by software and it is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 23.
0: Divider by 2 enabled
1: Divider by 2 disabled

Bit 4 = **MSTR** *Master.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, SS=0 (see Section 7.5.4.5 Master Mode Fault).
0: Slave mode is selected
1: Master mode is selected, the function of the SCK pin changes from an input to an output and the functions of the MISO and MOSI pins are reversed.

Bit 3 = **CPOL** *Clock polarity.*
This bit is set and cleared by software. This bit determines the steady state of the serial Clock. The CPOL bit affects both the master and slave modes.
0: The steady state is a low value at the SCK pin.
1: The steady state is a high value at the SCK pin.

Bit 2 = **CPHA** *Clock phase.*
This bit is set and cleared by software.
0: The first clock transition is the first data capture edge.
1: The second clock transition is the first capture edge.

Bit 1:0 = **SPR[1:0]** *Serial peripheral rate.*
These bits are set and cleared by software.Used with the SPR2 bit, they select one of six baud rates to be used as the serial clock when the device is a master.

These 2 bits have no effect in slave mode.

Table 23. Serial Peripheral Baud Rate

Serial Clock	SPR2	SPR1	SPR0
$f_{CPU}/4$	1	0	0
$f_{CPU}/8$	0	0	0
$f_{CPU}/16$	0	0	1
$f_{CPU}/32$	1	1	0
$f_{CPU}/64$	0	1	0
$f_{CPU}/128$	0	1	1

SERIAL PERIPHERAL INTERFACE (Cont'd)

STATUS REGISTER (SR)

Read Only

Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	-	MODF	-	-	-	-

Bit 7 = **SPIF** *Serial Peripheral data transfer flag.*
This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE=1 in the CR register. It is cleared by a software sequence (an access to the SR register followed by a read or write to the DR register).
0: Data transfer is in progress or has been approved by a clearing sequence.
1: Data transfer between the device and an external device has been completed.

Note: While the SPIF bit is set, all writes to the DR register are inhibited.

Bit 6 = **WCOL** *Write Collision status.*

This bit is set by hardware when a write to the DR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 52).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = Unused.

Bit 4 = **MODF** *Mode Fault flag.*
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 7.5.4.5 Master Mode Fault). An SPI interrupt can be generated if SPIE=1 in the CR register. This bit is cleared by a software sequence (An access to the SR register while MODF=1 followed by a write to the CR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bits 3-0 = Unused.

DATA I/O REGISTER (DR)

Read/Write

Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The DR register is used to transmit and receive data on the serial bus. In the master device only a write to this register will initiate transmission/reception of another byte.

Notes: During the last clock cycle the SPIF bit is set, a copy of the received data byte in the shift register is moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read.

Warning:

A write to the DR register places data directly into the shift register for transmission.

A read to the the DR register returns the value located in the buffer and not the contents of the shift register (See Figure 49).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 24. SPI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0021h	**SPIDR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
0022h	**SPICR**	SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0
	Reset Value	0	0	0	0	x	x	x	x
0023h	**SPISR**	SPIF	WCOL		MODF				
	Reset Value	0	0	0	0	0	0	0	0

7.6 SERIAL COMMUNICATIONS INTERFACE (SCI)

7.6.1 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.

7.6.2 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 250K baud using conventional baud rate generator and up to 500K baud using the extended baud rate generator.
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- Two receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- LIN compatible (if MCU clock frequency tolerance ≤2%)
- Separate enable bits for Transmitter and Receiver
- Three error detection flags:
  - Overrun error
  - Noise error
  - Frame error
- Five interrupt sources with flags:
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error detected

7.6.3 General Description

The interface is externally connected to another device by two pins (see Figure 2.):

– 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 this 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,

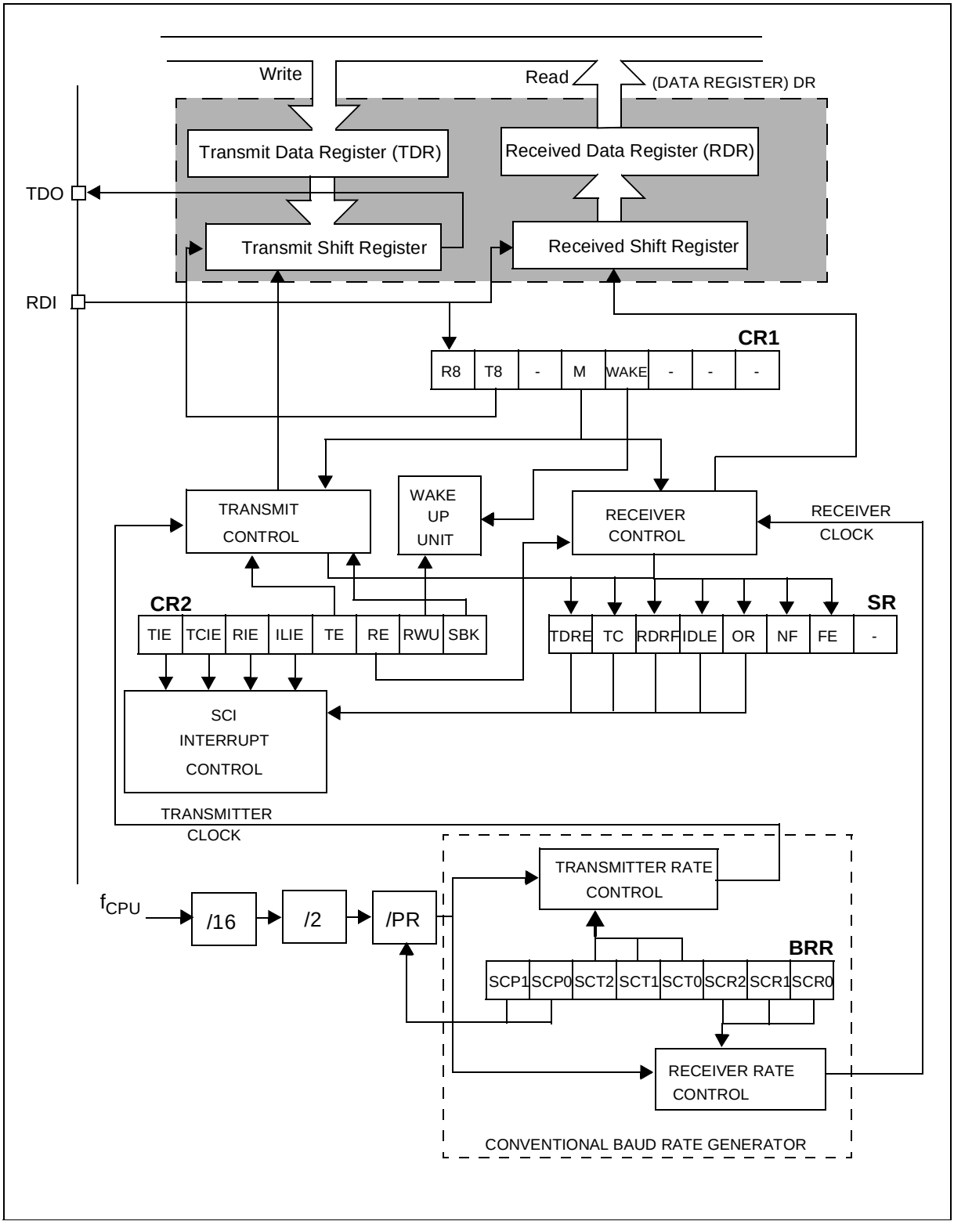
– An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies.

7.6.4 LIN Protocol support

For LIN applications where resynchronization is not required (application clock tolerance less than or equal to 2%) the LIN protocol can be efficiently implemented with this standard SCI.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 54. SCI Block Diagram



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

7.6.5 Functional Description

The block diagram of the Serial Control Interface, is shown in Figure 1.. It contains 6 dedicated registers:

– Two control registers (CR1 & CR2)

– A status register (SR)

– A baud rate register (BRR)

– An extended prescaler receiver register (ERPR)

– An extended prescaler transmitter register (ETPR)

Refer to the register descriptions in Section 0.1.8 for the definitions of each bit.

7.6.5.1 Serial Data Format

Word length may be selected as being either 8 or 9 bits by programming the M bit in the CR1 register (see Figure 1.).

The TDO pin is in low state during the start bit.

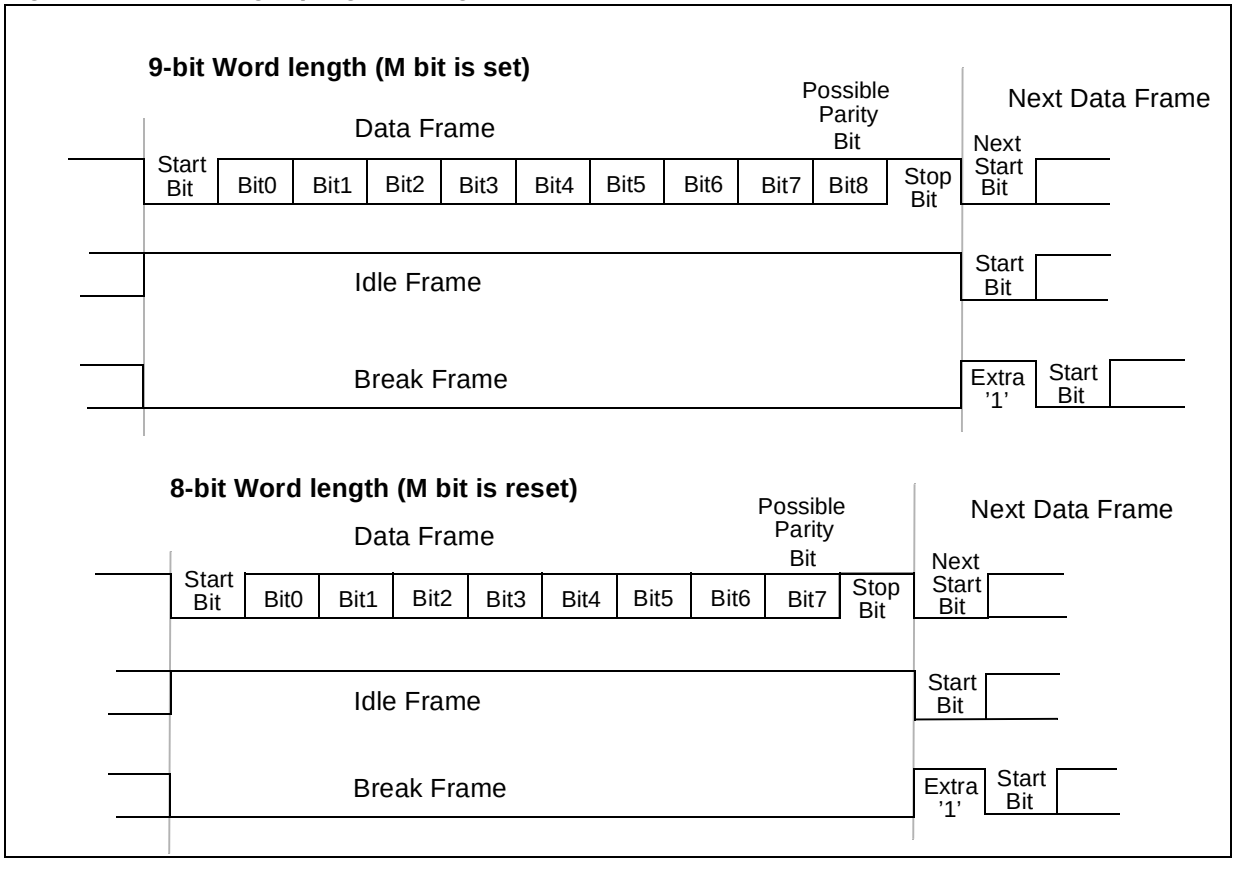
The TDO pin is in high state during the stop bit.

An Idle character is interpreted as an entire frame of "1"s followed by the start bit of the next frame which contains data.

A Break character is interpreted on receiving "0"s for some multiple of the frame period. At the end of the last break frame the transmitter inserts an extra "1" bit to acknowledge the start bit.

Transmission and reception are driven by their own baud rate generator.

Figure 55. Word length programming



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

7.6.5.2 Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bit status. When the M bit is set, word length is 9 bits and the 9th bit (the MSB) has to be stored in the T8 bit in the CR1 register.

Character Transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the DR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 1.).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the BRR and the ETPR registers.

– Set the TE bit to assign the TDO pin to the alternate function and to send a idle frame as first transmission.

– Access the SR register and write the data to send in the DR register (this sequence clears the TDRE bit). Repeat this sequence for each data to be transmitted.

Clearing the TDRE bit is always performed by the following software sequence:
1. An access to the SR register
2. A write to the DR register

The TDRE bit is set by hardware and it indicates:

– The TDR register is empty.

– The data transfer is beginning.

– The next data can be written in the DR register without overwriting the previous data.

This flag generates an interrupt if the TIE bit is set and the I bit is cleared in the CCR register.

When a transmission is taking place, a write instruction to the DR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the DR register places the data directly in the shift register, the data transmission starts, and the TDRE bit is immediately set.

When a frame transmission is complete (after the stop bit or after the break frame) the TC bit is set and an interrupt is generated if the TCIE is set and the I bit is cleared in the CCR register.

Clearing the TC bit is performed by the following software sequence:
1. An access to the SR register
2. A write to the DR register

Note: The TDRE and TC bits are cleared by the same software sequence.

Break Characters

Setting the SBK bit loads the shift register with a break character. The break frame length depends on the M bit (see Figure 2.).

As long as the SBK bit is set, the SCI send break frames to the TDO pin. After clearing this bit by software the SCI insert a logic 1 bit at the end of the last break frame to guarantee the recognition of the start bit of the next frame.

Idle Characters

Setting the TE bit drives the SCI to send an idle frame before the first data frame.

Clearing and then setting the TE bit during a transmission sends an idle frame after the current word.

Note: Resetting and setting the TE bit causes the data in the TDR register to be lost. Therefore the best time to toggle the TE bit is when the TDRE bit is set i.e. before writing the next byte in the DR.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

7.6.5.3 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 CR1 register.

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, DR register consists in a buffer (RDR) between the internal bus and the received shift register (see Figure 1.).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the BRR and the ERPR 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 SR register

2. A read to the DR 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 a 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 can not be transferred from the shift register to the TDR register as long as the RDRF bit is not cleared.

When a overrun error occurs:

– The OR bit is set.

– The RDR content will not be lost.

– The shift register will be 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 SR register followed by a DR 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 DR 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 SR register read operation followed by a DR 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

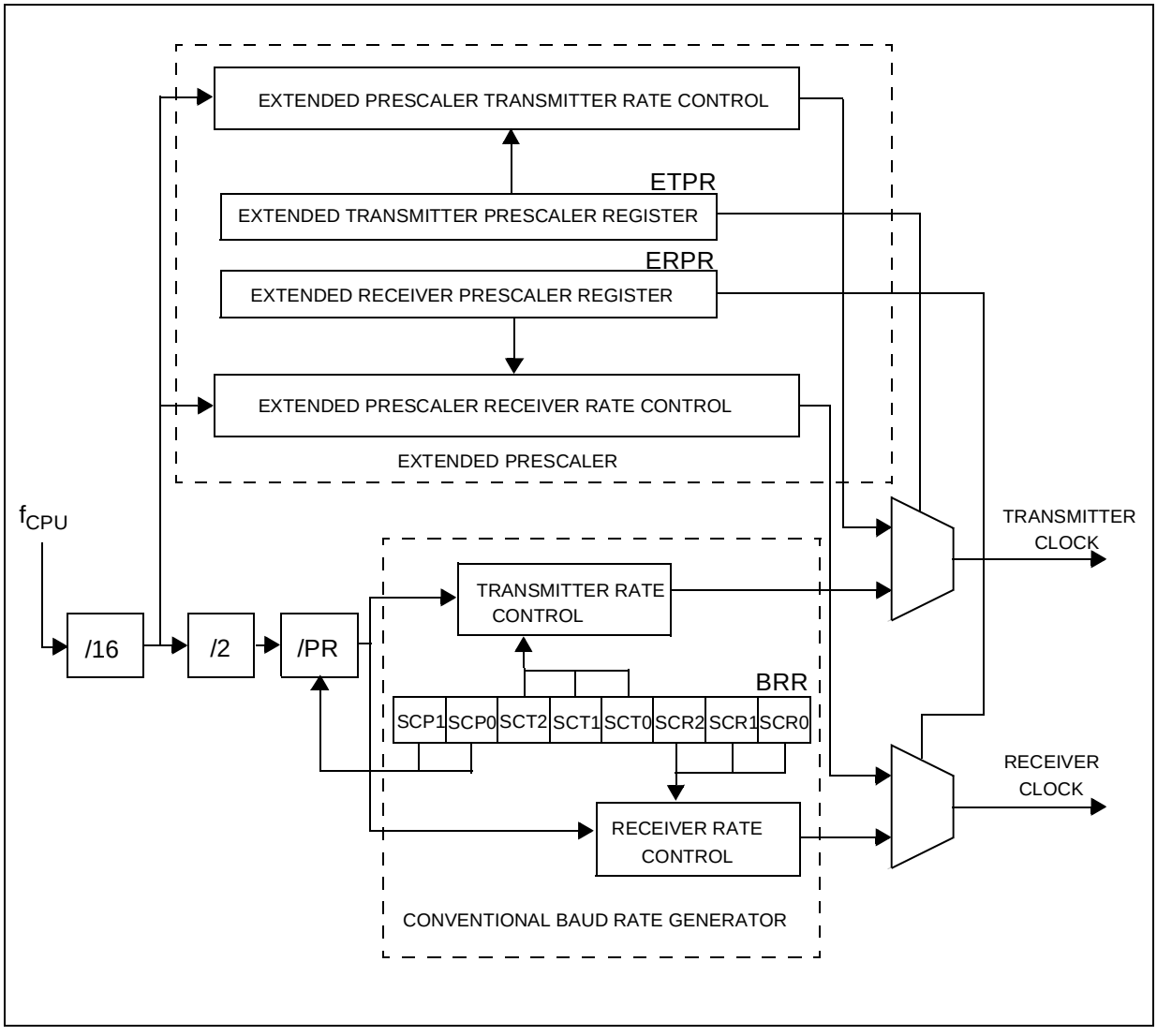
– Data is transferred from the Shift register to the DR register.

– 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 SR register read operation followed by a DR register read operation.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 56. SCI Baud Rate and Extended Prescaler Block Diagram



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

7.6.5.4 Conventional Baud Rate Generation

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

$$Tx = \frac{f_{CPU}}{(32*PR)*TR} \qquad Rx = \frac{f_{CPU}}{(32*PR)*RR}$$

with:

PR = 1, 3, 4 or 13 (see SCP0 & SCP1 bits)

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

(see SCT0, SCT1 & SCT2 bits)

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

(see SCR0,SCR1 & SCR2 bits)

All this bits are in the BRR register.

Example: If f_{CPU} is 8 MHz (normal mode) and if PR=13 and TR=RR=1, the transmit and receive baud rates are 19200 baud.

Caution: The baud rate register (SCIBRR) MUST NOT be written to (changed or refreshed) while the transmitter or the receiver is enabled.

7.6.5.5 Extended Baud Rate Generation

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

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

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

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

$$Tx = \frac{f_{CPU}}{16*ETPR} \qquad Rx = \frac{f_{CPU}}{16*ERPR}$$

with:

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

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

7.6.5.6 Receiver Muting and Wake-up Feature

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant SCI service overhead for all non addressed receivers.

The non addressed devices may be placed in sleep mode by means of the muting function.

Setting the RWU bit by software puts the SCI in sleep mode:

All the reception status bits can not be set.

All the receive interrupt are inhibited.

A muted receiver may be awakened by one of the following two ways:

– by Idle Line detection if the WAKE bit is reset,

– by Address Mark detection if the WAKE bit is set.

Receiver wakes-up by Idle Line detection when the Receive line has recognised an Idle Frame. Then the RWU bit is reset by hardware but the IDLE bit is not set.

Receiver wakes-up by Address Mark detection when it received a "1" as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

7.6.6 Low Power Modes

Mode	Description
WAIT	No effect on SCI. SCI interrupts cause the device to exit from Wait mode.
HALT	SCI registers are frozen. In Halt mode, the SCI stops transmitting/receiving until Halt mode is exited.

7.6.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Transmit Data Register Empty	TDRE	TIE	Yes	No
Transmission Complete	TC	TCIE	Yes	No
Received Data Ready to be Read	RDRF	RIE	Yes	No
Overrrun Error Detected	OR		Yes	No
Idle Line Detected	IDLE	ILIE	Yes	No

The SCI interrupt events are connected to the same interrupt vector (see Interrupts chapter).

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

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

7.6.8 Register Description

STATUS REGISTER (SR)

Read Only

Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	-

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 =1 in the CR2 register. It is cleared by a software sequence (an access to the SR register followed by a write to the DR register).
0: Data is not transferred to the shift register
1: Data is transferred to the shift register

Note: data will not be transferred to the shift register as long as the TDRE bit is not reset.

Bit 6 = **TC** *Transmission complete.*
This bit is set by hardware when transmission of a frame containing Data, a Preamble or a Break is complete. An interrupt is generated if TCIE=1 in the CR2 register. It is cleared by a software sequence (an access to the SR register followed by a write to the DR register).
0: Transmission is not complete
1: Transmission is complete

Bit 5 = **RDRF** *Received data ready flag.*
This bit is set by hardware when the content of the RDR register has been transferred into the DR register. An interrupt is generated if RIE=1 in the CR2 register. It is cleared by a software sequence (an access to the SR register followed by a read to the DR 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 a Idle Line is detected. An interrupt is generated if the ILIE=1 in the CR2 register. It is cleared by a software sequence (an access to the SR register followed by a read to the DR register).
0: No Idle Line is detected
1: Idle Line is detected

Note: The IDLE bit will not be set again until the RDRF bit has been set itself (i.e. a new idle line occurs). This bit is not set by an idle line when the receiver wakes up from wake-up mode.

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 CR2 register. It is cleared by a software sequence (an access to the SR register followed by a read to the DR register).
0: No Overrun error
1: Overrun error is detected

Note: When this bit is set RDR register content will not be lost but the shift register will be 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 SR register followed by a read to the DR 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 SR register followed by a read to the DR register).
0: No Framing error is detected
1: Framing error or break character 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. If the word currently being transferred causes both frame error and overrun error, it will be transferred and only the OR bit will be set.

Bit 0 = Unused.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CONTROL REGISTER 1 (CR1)

Read/Write

Reset Value: Undefined

7							0
R8	T8	-	M	WAKE	-	-	-

Bit 7 = **R8** *Receive data bit 8.*
This bit is used to store the 9th bit of the received word when M=1.

Bit 6 = **T8** *Transmit data bit 8.*
This bit is used to store the 9th bit of the transmitted word when M=1.

Bit 4 = **M** *Word length.*
This bit determines the word length. It is set or cleared by software.
0: 1 Start bit, 8 Data bits, 1 Stop bit
1: 1 Start bit, 9 Data bits, 1 Stop bit

Bit 3 = **WAKE** *Wake-Up method.*
This bit determines the SCI Wake-Up method, it is set or cleared by software.
0: Idle Line
1: Address Mark

CONTROL REGISTER 2 (CR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

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

Bit 6 = **TCIE** *Transmission complete interrupt enable*

This bit is set and cleared by software.
0: interrupt is inhibited
1: An SCI interrupt is generated whenever TC=1 in the SR register

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

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

Bit 3 = **TE** *Transmitter enable.*
This bit enables the transmitter and assigns the TDO pin to the alternate function. It is set and cleared by software.
0: Transmitter is disabled, the TDO pin is back to the I/O port configuration.
1: Transmitter is enabled

Note: during transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble after the current word.

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

Bit 1 = **RWU** *Receiver wake-up.*
This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.
0: Receiver in active mode
1: Receiver in mute mode

Bit 0 = **SBK** *Send break.*
This bit set is used to send break characters. It is set and cleared by software.
0: No break character is transmitted
1: Break characters are transmitted

Note: If the SBK bit is set to "1" and then to "0", the transmitter will send a BREAK word at the end of the current word.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

DATA REGISTER (DR)

Read/Write

Reset Value: Undefined

Contains the Received or Transmitted data character, depending on whether it is read from or written to.

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).
The TDR register provides the parallel interface between the internal bus and the output shift register (see Figure 1.).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 1.).

BAUD RATE REGISTER (BRR)

Read/Write

Reset Value: 00xx xxxx (XXh)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

Bit 7:6= **SCP[1:0]** *First SCI Prescaler*
These 2 prescaling bits allow several standard clock division ranges:

PR Prescaling factor	SCP1	SCP0
1	0	0
3	0	1
4	1	0
13	1	1

Bit 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*

These 3 bits, in conjunction with the SCP1 & SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional Baud Rate Generator mode.

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Note: this TR factor is used only when the ETPR fine tuning factor is equal to 00h; otherwise, TR is replaced by the ETPR dividing factor.

Bit 2:0 = **SCR[2:0]** *SCI Receiver rate divisor.*

These 3 bits, in conjunction with the SCP1 & SCP0 bits define the total division applied to the bus clock to yield the receive rate clock in conventional Baud Rate Generator mode.

RR dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2	0	0	1
4	0	1	0
8	0	1	1
16	1	0	0
32	1	0	1
64	1	1	0
128	1	1	1

Note: this RR factor is used only when the ERPR fine tuning factor is equal to 00h; otherwise, RR is replaced by the ERPR dividing factor.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

EXTENDED RECEIVE PRESCALER DIVISION REGISTER (ERPR)

Read/Write

Reset Value: 0000 0000 (00h)

Allows setting of the Extended Prescaler rate division factor for the receive circuit.

7							0
ERPR 7	ERPR 6	ERPR 5	ERPR 4	ERPR 3	ERPR 2	ERPR 1	ERPR 0

Bit 7:1 = **ERPR[7:0]** *8-bit Extended Receive Prescaler Register.*

The extended Baud Rate Generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see Figure 3.) is divided by the binary factor set in the ERPR register (in the range 1 to 255).

The extended baud rate generator is not used after a reset.

EXTENDED TRANSMIT PRESCALER DIVISION REGISTER (ETPR)

Read/Write

Reset Value:0000 0000 (00h)

Allows setting of the External Prescaler rate division factor for the transmit circuit.

7							0
ETPR 7	ETPR 6	ETPR 5	ETPR 4	ETPR 3	ETPR 2	ETPR 1	ETPR 0

Bit 7:1 = **ETPR[7:0]** *8-bit Extended Transmit Prescaler Register.*

The extended Baud Rate Generator is activated when a value different from 00h is stored in this register. Therefore the clock frequency issued from the 16 divider (see Figure 3.) is divided by the binary factor set in the ETPR register (in the range 1 to 255).

The extended baud rate generator is not used after a reset.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Table 25. SCI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
50	**SCISR**	TDRE	TC	RDRF	IDLE	OR	NF	FE	
	Reset Value	1	1	0	0	0	0	0	0
51	**SCIDR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
52	**SCIBRR**	SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0
	Reset Value	0	0	x	x	x	x	x	x
53	**SCICR1**	R8	T8		M	WAKE			
	Reset Value	x	x	0	x	x	0	0	0
54	**SCICR2**	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	Reset Value	0	0	0	0	0	0	0	0
55	**SCIPBRR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
57	**SCIPBRT**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0

7.7 I²C BUS INTERFACE (I2C)

7.7.1 Introduction

The I²C Bus Interface serves as an interface between the microcontroller and the serial I²C bus. It provides both multimaster and slave functions, and controls all I²C bus-specific sequencing, protocol, arbitration and timing. It supports fast I²C mode (400kHz).

7.7.2 Main Features

- Parallel-bus/I²C protocol converter
- Multi-master capability
- 7-bit/10-bit Addressing
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection

I²C Master Features:

- Clock generation
- I²C bus busy flag
- Arbitration Lost Flag
- End of byte transmission flag
- Transmitter/Receiver Flag
- Start bit detection flag
- Start and Stop generation

I²C Slave Features:

- Stop bit detection
- I²C bus busy flag
- Detection of misplaced start or stop condition
- Programmable I²C Address detection
- Transfer problem detection
- End-of-byte transmission flag
- Transmitter/Receiver flag

7.7.3 General Description

In addition to receiving and transmitting data, this interface converts it from serial to parallel format and vice versa, using either an interrupt or polled handshake. The interrupts are enabled or disabled by software. The interface is connected to the I²C bus by a data pin (SDAI) and by a clock pin (SCLI). It can be connected both with a standard I²C bus and a Fast I²C bus. This selection is made by software.

Mode Selection

The interface can operate in the four following modes:

– Slave transmitter/receiver

– Master transmitter/receiver

By default, it operates in slave mode.

The interface automatically switches from slave to master after it generates a START condition and from master to slave in case of arbitration loss or a STOP generation, allowing then Multi-Master capability.

Communication Flow

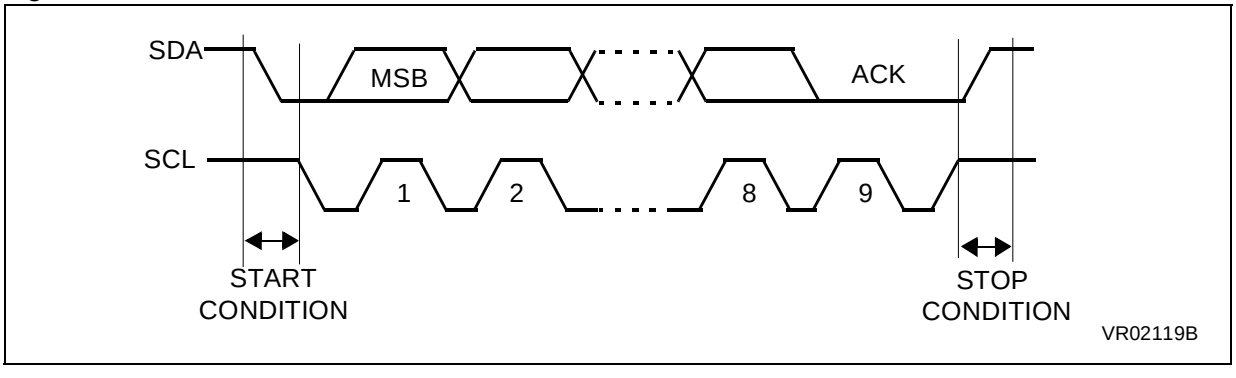
In Master mode, it initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

In Slave mode, the interface is capable of recognising its own address (7 or 10-bit), and the General Call address. The General Call address detection may be enabled or disabled by software.

Data and addresses are transferred as 8-bit bytes, MSB first. The first byte(s) following the start condition contain the address (one in 7-bit mode, two in 10-bit mode). The address is always transmitted in Master mode.

A 9th clock pulse follows the 8 clock cycles of a byte transfer, during which the receiver must send an acknowledge bit to the transmitter. Refer to Figure 57.

Figure 57. I²C BUS Protocol



I²C BUS INTERFACE (Cont'd)

Acknowledge may be enabled and disabled by software.

The I²C interface address and/or general call address can be selected by software.

The speed of the I²C interface may be selected between Standard (0-100KHz) and Fast I²C (100-400KHz).

SDA/SCL Line Control

Transmitter mode: the interface holds the clock line low before transmission to wait for the microcontroller to write the byte in the Data Register.

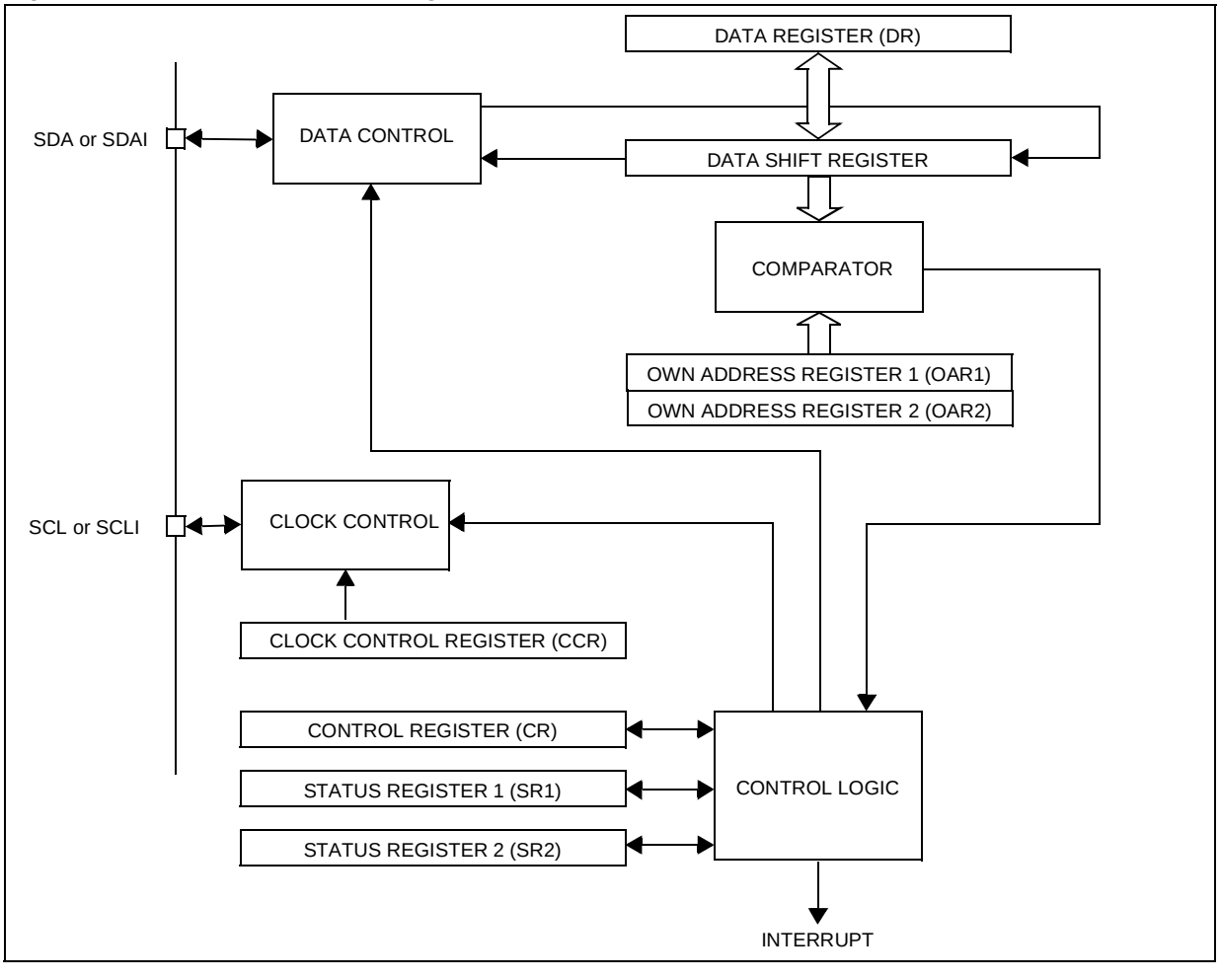
Receiver mode: the interface holds the clock line low after reception to wait for the microcontroller to read the byte in the Data Register.

The SCL frequency (F_{SCL}) is controlled by a programmable clock divider which depends on the I²C bus mode.

When the I²C cell is enabled, the SDA and SCL ports must be configured as floating inputs. In this case, the value of the external pull-up resistor used depends on the application.

When the I²C cell is disabled, the SDA and SCL ports revert to being standard I/O port pins.

Figure 58. I²C Interface Block Diagram



I²C BUS INTERFACE (Cont'd)

7.7.4 Functional Description

Refer to the CR, SR1 and SR2 registers in Section 7.7.7. for the bit definitions.

By default the I²C interface operates in Slave mode (M/SL bit is cleared) except when it initiates a transmit or receive sequence.

First the interface frequency must be configured using the FRi bits in the OAR2 register.

7.7.4.1 Slave Mode

As soon as a start condition is detected, the address is received from the SDA line and sent to the shift register; then it is compared with the address of the interface or the General Call address (if selected by software).

Note: In 10-bit addressing mode, the comparision includes the header sequence (11110xx0) and the two most significant bits of the address.

Header matched (10-bit mode only): the interface generates an acknowledge pulse if the ACK bit is set.

Address not matched: the interface ignores it and waits for another Start condition.

Address matched: the interface generates in sequence:

– Acknowledge pulse if the ACK bit is set.

– EVF and ADSL bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV1).
Next, in 7-bit mode read the DR register to determine from the least significant bit (Data Direction Bit) if the slave must enter Receiver or Transmitter mode.

In 10-bit mode, after receiving the address sequence the slave is always in receive mode. It will enter transmit mode on receiving a repeated Start condition followed by the header sequence with matching address bits and the least significant bit set (11110xx1) .

Slave Receiver

Following the address reception and after SR1 register has been read, the slave receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

– Acknowledge pulse if the ACK bit is set

– EVF and BTF bits are set with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV2).

Slave Transmitter

Following the address reception and after SR1 register has been read, the slave sends bytes from the DR register to the SDA line via the internal shift register.

The slave waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV3).

When the acknowledge pulse is received:

– The EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Closing slave communication

After the last data byte is transferred a Stop Condition is generated by the master. The interface detects this condition and sets:

– EVF and STOPF bits with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR2 register (see Figure 59 Transfer sequencing EV4).

Error Cases

– **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and the BERR bits are set with an interrupt if the ITE bit is set.
If it is a Stop then the interface discards the data, released the lines and waits for another Start condition.
If it is a Start then the interface discards the data and waits for the next slave address on the bus.

– **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set with an interrupt if the ITE bit is set.

Note: In both cases, SCL line is not held low; however, SDA line can remain low due to possible «0» bits transmitted last. It is then necessary to release both lines by software.

I²C BUS INTERFACE (Cont'd)

How to release the SDA / SCL lines

Set and subsequently clear the STOP bit while BTF is set. The SDA/SCL lines are released after the transfer of the current byte.

7.7.4.2 Master Mode

To switch from default Slave mode to Master mode a Start condition generation is needed.

Start condition

Setting the START bit while the BUSY bit is cleared causes the interface to switch to Master mode (M/SL bit set) and generates a Start condition.

Once the Start condition is sent:

– The EVF and SB bits are set by hardware with an interrupt if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register with the Slave address, **holding the SCL line low** (see Figure 59 Transfer sequencing EV5).

Slave address transmission

Then the slave address is sent to the SDA line via the internal shift register.

In 7-bit addressing mode, one address byte is sent.

In 10-bit addressing mode, sending the first byte including the header sequence causes the following event:

– The EVF bit is set by hardware with interrupt generation if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV9).

Then the second address byte is sent by the interface.

After completion of this transfer (and acknowledge from the slave if the ACK bit is set):

– The EVF bit is set by hardware with interrupt generation if the ITE bit is set.

Then the master waits for a read of the SR1 register followed by a write in the CR register (for example set PE bit), **holding the SCL line low** (see Figure 59 Transfer sequencing EV6).

Next the master must enter Receiver or Transmitter mode.

Note: In 10-bit addressing mode, to switch the master to Receiver mode, software must generate a repeated Start condition and resend the header sequence with the least significant bit set (11110xx1).

Master Receiver

Following the address transmission and after SR1 and CR registers have been accessed, the master receives bytes from the SDA line into the DR register via the internal shift register. After each byte the interface generates in sequence:

– Acknowledge pulse if if the ACK bit is set

– EVF and BTF bits are set by hardware with an interrupt if the ITE bit is set.

Then the interface waits for a read of the SR1 register followed by a read of the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV7).

To close the communication: before reading the last byte from the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

Note: In order to generate the non-acknowledge pulse after the last received data byte, the ACK bit must be cleared just before reading the second last data byte.

I²C BUS INTERFACE (Cont'd)

Master Transmitter

Following the address transmission and after SR1 register has been read, the master sends bytes from the DR register to the SDA line via the internal shift register.

The master waits for a read of the SR1 register followed by a write in the DR register, **holding the SCL line low** (see Figure 59 Transfer sequencing EV8).

When the acknowledge bit is received, the interface sets:

– EVF and BTF bits with an interrupt if the ITE bit is set.

To close the communication: after writing the last byte to the DR register, set the STOP bit to generate the Stop condition. The interface goes automatically back to slave mode (M/SL bit cleared).

Error Cases

– **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and BERR bits are set by hardware with an interrupt if ITE is set.

– **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set by hardware with an interrupt if the ITE bit is set. To resume, set the START or STOP bit.

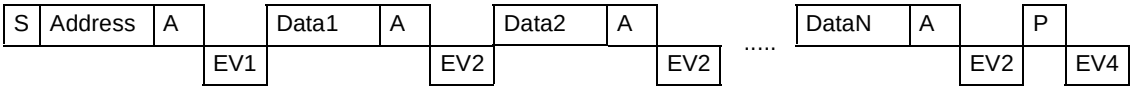
– **ARLO:** Detection of an arbitration lost condition. In this case the ARLO bit is set by hardware (with an interrupt if the ITE bit is set and the interface goes automatically back to slave mode (the M/SL bit is cleared).

Note: In all these cases, the SCL line is not held low; however, the SDA line can remain low due to possible «0» bits transmitted last. It is then necessary to release both lines by software.

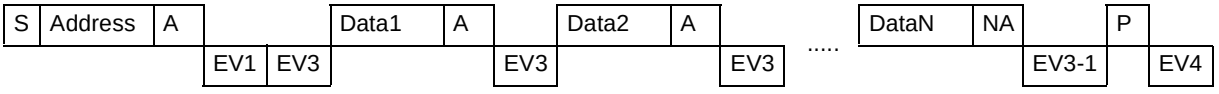
I²C BUS INTERFACE (Cont'd)

Figure 59. Transfer Sequencing

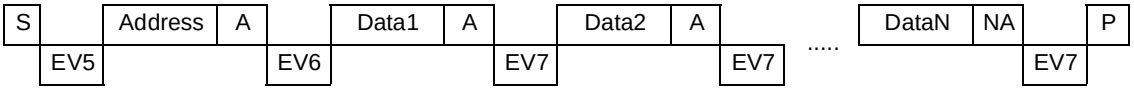
7-bit Slave receiver:



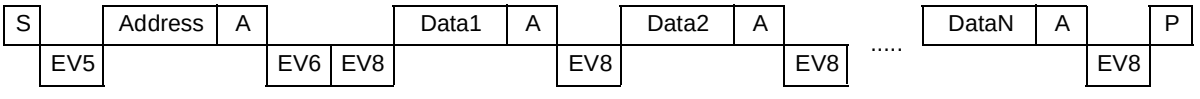
7-bit Slave transmitter:



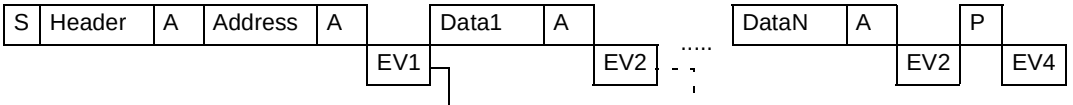
7-bit Master receiver:



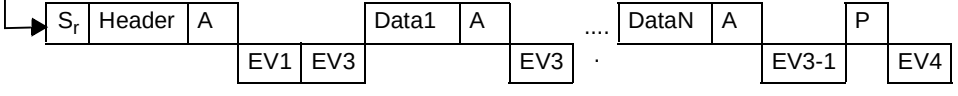
7-bit Master transmitter:



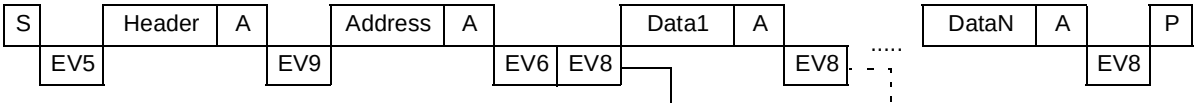
10-bit Slave receiver:



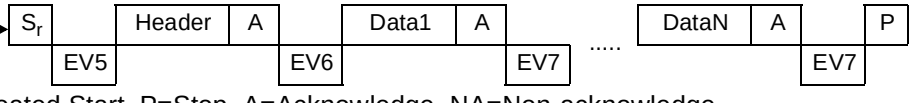
10-bit Slave transmitter:



10-bit Master transmitter



10-bit Master receiver:



Legend: S=Start, S_r = Repeated Start, P=Stop, A=Acknowledge, NA=Non-acknowledge, EVx=Event (with interrupt if ITE=1)

EV1: EVF=1, ADSL=1, cleared by reading SR1 register.

EV2: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

EV3: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

EV3-1: EVF=1, AF=1, BTF=1; AF is cleared by reading SR1 register. BTF is cleared by releasing the lines (STOP=1, STOP=0) or by writing DR register (DR=FFh). **Note:** If lines are released by STOP=1, STOP=0, the subsequent EV4 is not seen.

EV4: EVF=1, STOPF=1, cleared by reading SR2 register.

EV5: EVF=1, SB=1, cleared by reading SR1 register followed by writing DR register.

EV6: EVF=1, cleared by reading SR1 register followed by writing CR register (for example PE=1).

EV7: EVF=1, BTF=1, cleared by reading SR1 register followed by reading DR register.

EV8: EVF=1, BTF=1, cleared by reading SR1 register followed by writing DR register.

EV9: EVF=1, ADD10=1, cleared by reading SR1 register followed by writing DR register.

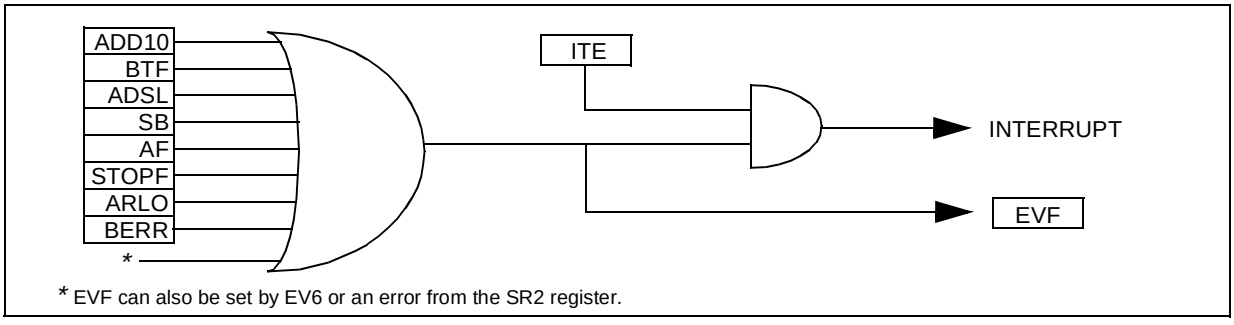
I²C BUS INTERFACE (Cont'd)

7.7.5 Low Power Modes

Mode	Description
WAIT	No effect on I²C interface. I²C interrupts cause the device to exit from WAIT mode.
HALT	I²C registers are frozen. In HALT mode, the I²C interface is inactive and does not acknowledge data on the bus. The I²C interface resumes operation when the MCU is woken up by an interrupt with "exit from HALT mode" capability.

7.7.6 Interrupts

Figure 60. Event Flags and Interrupt Generation



* EVF can also be set by EV6 or an error from the SR2 register.

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
10-bit Address Sent Event (Master mode)	ADD10	ITE	Yes	No
End of Byte Transfer Event	BTF		Yes	No
Address Matched Event (Slave mode)	ADSEL		Yes	No
Start Bit Generation Event (Master mode)	SB		Yes	No
Acknowledge Failure Event	AF		Yes	No
Stop Detection Event (Slave mode)	STOPF		Yes	No
Arbitration Lost Event (Multimaster configuration)	ARLO		Yes	No
Bus Error Event	BERR		Yes	No

Note: The I²C 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 I-bit in the CC register is reset (RIM instruction).

I²C BUS INTERFACE (Cont'd)

7.7.7 Register Description

I²C CONTROL REGISTER (CR)

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

7							0
0	0	PE	ENGC	START	ACK	STOP	ITE

Bit 7:6 = Reserved. Forced to 0 by hardware.

Bit 5 = **PE** *Peripheral enable.*
This bit is set and cleared by software.
0: Peripheral disabled
1: Master/Slave capability
Notes:
– When PE=0, all the bits of the CR register and the SR register except the Stop bit are reset. All outputs are released while PE=0
– When PE=1, the corresponding I/O pins are selected by hardware as alternate functions.
– To enable the I²C interface, write the CR register **TWICE** with PE=1 as the first write only activates the interface (only PE is set).

Bit 4 = **ENGC** *Enable General Call.*
This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0). The 00h General Call address is acknowledged (01h ignored).
0: General Call disabled
1: General Call enabled

Bit 3 = **START** *Generation of a Start condition*.
This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0) or when the Start condition is sent (with interrupt generation if ITE=1).

– In master mode:
  0: No start generation
  1: Repeated start generation
– In slave mode:
  0: No start generation
  1: Start generation when the bus is free

Bit 2 = **ACK** *Acknowledge enable.*
This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0).
0: No acknowledge returned
1: Acknowledge returned after an address byte or a data byte is received

Bit 1 = **STOP** *Generation of a Stop condition*.
This bit is set and cleared by software. It is also cleared by hardware in master mode. Note: This bit is not cleared when the interface is disabled (PE=0).

– In master mode:
  0: No stop generation
  1: Stop generation after the current byte transfer or after the current Start condition is sent. The STOP bit is cleared by hardware when the Stop condition is sent.
– In slave mode:
  0: No stop generation
  1: Release the SCL and SDA lines after the current byte transfer (BTF=1). In this mode the STOP bit has to be cleared by software.

Bit 0 = **ITE** *Interrupt enable.*
This bit is set and cleared by software and cleared by hardware when the interface is disabled (PE=0).
0: Interrupts disabled
1: Interrupts enabled
Refer to Figure 60 for the relationship between the events and the interrupt.
SCL is held low when the ADD10, SB, BTF or ADSL flags or an EV6 event (See Figure 59) is detected.

I²C BUS INTERFACE (Cont'd)

I²C STATUS REGISTER 1 (SR1)

Read Only

Reset Value: 0000 0000 (00h)

7							0
EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB

Bit 7 = **EVF** *Event flag.*
This bit is set by hardware as soon as an event occurs. It is cleared by software reading SR2 register in case of error event or as described in Figure 59. It is also cleared by hardware when the interface is disabled (PE=0).
0: No event
1: One of the following events has occurred:

- BTF=1 (Byte received or transmitted)
- ADSL=1 (Address matched in Slave mode while ACK=1)
- SB=1 (Start condition generated in Master mode)
- AF=1 (No acknowledge received after byte transmission)
- STOPF=1 (Stop condition detected in Slave mode)
- ARLO=1 (Arbitration lost in Master mode)
- BERR=1 (Bus error, misplaced Start or Stop condition detected)
- ADD10=1 (Master has sent header byte)
- Address byte successfully transmitted in Master mode.

Bit 6 = **ADD10** *10-bit addressing in Master mode*.
This bit is set by hardware when the master has sent the first byte in 10-bit address mode. It is cleared by software reading SR2 register followed by a write in the DR register of the second address byte. It is also cleared by hardware when the peripheral is disabled (PE=0).

0: No ADD10 event occurred.
1: Master has sent first address byte (header)

Bit 5 = **TRA** *Transmitter/Receiver.*
When BTF is set, TRA=1 if a data byte has been transmitted. It is cleared automatically when BTF is cleared. It is also cleared by hardware after detection of Stop condition (STOPF=1), loss of bus arbitration (ARLO=1) or when the interface is disabled (PE=0).
0: Data byte received (if BTF=1)
1: Data byte transmitted

Bit 4 = **BUSY** *Bus busy*.
This bit is set by hardware on detection of a Start condition and cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. This information is still updated when the interface is disabled (PE=0).
0: No communication on the bus
1: Communication ongoing on the bus

Bit 3 = **BTF** *Byte transfer finished.*
This bit is set by hardware as soon as a byte is correctly received or transmitted with interrupt generation if ITE=1. It is cleared by software reading SR1 register followed by a read or write of DR register. It is also cleared by hardware when the interface is disabled (PE=0).

- Following a byte transmission, this bit is set after reception of the acknowledge clock pulse. In case an address byte is sent, this bit is set only after the EV6 event (See Figure 59). BTF is cleared by reading SR1 register followed by writing the next byte in DR register.
- Following a byte reception, this bit is set after transmission of the acknowledge clock pulse if ACK=1. BTF is cleared by reading SR1 register followed by reading the byte from DR register.

The SCL line is held low while BTF=1.

0: Byte transfer not done
1: Byte transfer succeeded

Bit 2 = **ADSL** *Address matched (Slave mode).*
This bit is set by hardware as soon as the received slave address matched with the OAR register content or a general call is recognized. An interrupt is generated if ITE=1. It is cleared by software reading SR1 register or by hardware when the interface is disabled (PE=0).

The SCL line is held low while ADSL=1.

0: Address mismatched or not received
1: Received address matched

I²C BUS INTERFACE (Cont'd)

Bit 1 = **M/SL** *Master/Slave.*
This bit is set by hardware as soon as the interface is in Master mode (writing START=1). It is cleared by hardware after detecting a Stop condition on the bus or a loss of arbitration (ARLO=1). It is also cleared when the interface is disabled (PE=0).
0: Slave mode
1: Master mode

Bit 0 = **SB** *Start bit (Master mode).*
This bit is set by hardware as soon as the Start condition is generated (following a write START=1). An interrupt is generated if ITE=1. It is cleared by software reading SR1 register followed by writing the address byte in DR register. It is also cleared by hardware when the interface is disabled (PE=0).
0: No Start condition
1: Start condition generated

I²C STATUS REGISTER 2 (SR2)
Read Only
Reset Value: 0000 0000 (00h)

7							0
0	0	0	AF	STOPF	ARLO	BERR	GCAL

Bit 7:5 = Reserved. Forced to 0 by hardware.

Bit 4 = **AF** *Acknowledge failure*.
This bit is set by hardware when no acknowledge is returned. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while AF=1.

0: No acknowledge failure
1: Acknowledge failure

Bit 3 = **STOPF** *Stop detection (Slave mode).*
This bit is set by hardware when a Stop condition is detected on the bus after an acknowledge (if ACK=1). An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while STOPF=1.

0: No Stop condition detected
1: Stop condition detected

Bit 2 = **ARLO** *Arbitration lost*.
This bit is set by hardware when the interface loses the arbitration of the bus to another master. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

After an ARLO event the interface switches back automatically to Slave mode (M/SL=0).

The SCL line is not held low while ARLO=1.

0: No arbitration lost detected
1: Arbitration lost detected

Bit 1 = **BERR** *Bus error.*
This bit is set by hardware when the interface detects a misplaced Start or Stop condition. An interrupt is generated if ITE=1. It is cleared by software reading SR2 register or by hardware when the interface is disabled (PE=0).

The SCL line is not held low while BERR=1.

0: No misplaced Start or Stop condition
1: Misplaced Start or Stop condition

Bit 0 = **GCAL** *General Call (Slave mode).*
This bit is set by hardware when a general call address is detected on the bus while ENGC=1. It is cleared by hardware detecting a Stop condition (STOPF=1) or when the interface is disabled (PE=0).

0: No general call address detected on bus
1: general call address detected on bus

I²C BUS INTERFACE (Cont'd)

I²C CLOCK CONTROL REGISTER (CCR)

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

7							0
FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0

Bit 7 = **FM/SM** *Fast/Standard I²C mode.*
This bit is set and cleared by software. It is not cleared when the interface is disabled (PE=0).
0: Standard I²C mode
1: Fast I²C mode

Bit 6:0 = **CC[6:0]** *7-bit clock divider.*
These bits select the speed of the bus (F_{SCL}) depending on the I²C mode. They are not cleared when the interface is disabled (PE=0).

– Standard mode (FM/SM=0): F_{SCL} <= 100kHz

$F_{SCL} = F_{CPU}/(2x([CC6..CC0]+2))$

– Fast mode (FM/SM=1): F_{SCL} > 100kHz

$F_{SCL} = F_{CPU}/(3x([CC6..CC0]+2))$

Note: The programmed F_{SCL} assumes no load on SCL and SDA lines.

I²C DATA REGISTER (DR)

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

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **D[7:0]** *8-bit Data Register.*
These bits contain the byte to be received or transmitted on the bus.

– Transmitter mode: Byte transmission start automatically when the software writes in the DR register.

– Receiver mode: the first data byte is received automatically in the DR register using the least significant bit of the address.
Then, the following data bytes are received one by one after reading the DR register.

I²C BUS INTERFACE (Cont'd)

I²C OWN ADDRESS REGISTER (OAR1)

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

7							0
ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0

7-bit Addressing Mode

Bit 7:1 = **ADD[7:1]** *Interface address*.
These bits define the I²C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

Bit 0 = **ADD0** *Address direction bit.*
This bit is don't care, the interface acknowledges either 0 or 1. It is not cleared when the interface is disabled (PE=0).

Note: Address 01h is always ignored.

10-bit Addressing Mode

Bit 7:0 = **ADD[7:0]** *Interface address*.
These are the least significant bits of the I²C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

I²C OWN ADDRESS REGISTER (OAR2)

Read / Write
Reset Value: 0100 0000 (40h)

7							0
FR1	FR0	0	0	0	ADD9	ADD8	0

Bit 7:6 = **FR[1:0]** *Frequency bits.*

These bits are set by software only when the interface is disabled (PE=0). To configure the interface to I²C specifed delays select the value corresponding to the microcontroller frequency F_{CPU}.

f_{CPU}	FR1	FR0
< 6 MHz	0	0
6 to 8 MHz	0	1

Bit 5:3 = Reserved

Bit 2:1 = **ADD[9:8]** *Interface address*.
These are the most significant bits of the I²C bus address of the interface (10-bit mode only). They are not cleared when the interface is disabled (PE=0).

Bit 0 = Reserved.

I²C BUS INTERFACE (Cont'd)

Table 26. I²C Register Map and Reset Values

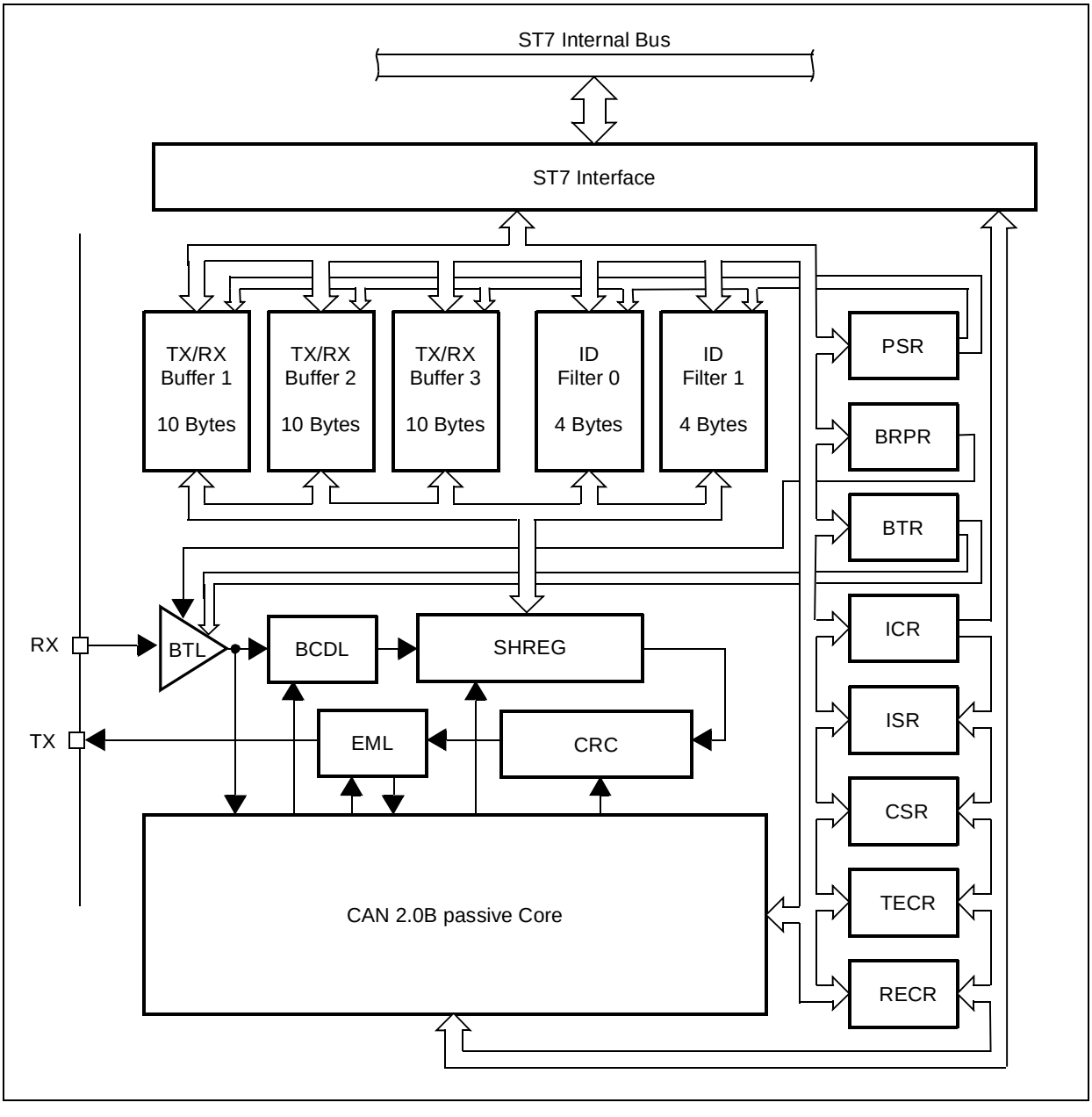
Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
28	**I2CCR**			PE	ENGC	START	ACK	STOP	ITE
	Reset Value	0	0	0	0	0	0	0	0
29	**I2CSR1**	EVF	ADD10	TRA	BUSY	BTF	ADSL	M/SL	SB
	Reset Value	0	0	0	0	0	0	0	0
2A	**I2CSR2**				AF	STOPF	ARLO	BERR	GCAL
	Reset Value	0	0	0	0	0	0	0	0
2B	**I2CCCR**	FM/SM	CC6	CC5	CC4	CC3	CC2	CC1	CC0
	Reset Value	0	0	0	0	0	0	0	0
2C	**I2COAR1**	ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
	Reset Value	0	0	0	0	0	0	0	0
2D	**I2COAR2**	FR1	FR0				ADD9	ADD8	
	Reset Value	0	1	0	0	0	0	0	0
2E	**I2CDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0

7.8 CONTROLLER AREA NETWORK (CAN)

7.8.1 Introduction

This peripheral is designed to support serial data exchanges using a multi-master contention based priority scheme as described in CAN specification Rev. 2.0 part A. It can also be connected to a 2.0 B network without problems, since extended frames are checked for correctness and acknowledged accordingly although such frames cannot be transmitted nor received. The same applies to overload frames which are recognized but never initiated.

Figure 61. CAN Block Diagram



CONTROLLER AREA NETWORK (Cont'd)

7.8.2 Main Features

– Support of CAN specification 2.0A and 2.0B passive
– Three prioritized 10-byte Transmit/Receive message buffers
– Two programmable global 12-bit message acceptance filters
– Programmable baud rates up to 1 MBit/s
– Buffer flip-flopping capability in transmission
– Maskable interrupts for transmit, receive (one per buffer), error and wake-up
– Automatic low-power mode after 20 recessive bits or on demand (standby mode)
– Interrupt-driven wake-up from standby mode upon reception of dominant pulse
– Optional dominant pulse transmission on leaving standby mode
– Automatic message queuing for transmission upon writing of data byte 7
– Programmable loop-back mode for self-test operation
– Advanced error detection and diagnosis functions
– Software-efficient buffer mapping at a unique address space
– Scalable architecture.

7.8.3 Functional Description

7.8.3.1 Frame Formats

A summary of all the CAN frame formats is given in Figure 62 for reference. It covers only the standard frame format since the extended one is only acknowledged.

A message begins with a start bit called Start Of Frame (SOF). This bit is followed by the arbitration field which contains the 11-bit identifier (ID) and the Remote Transmission Request bit (RTR). The RTR bit indicates whether it is a data frame or a remote request frame. A remote request frame does not have any data byte.

The control field contains the Identifier Extension bit (IDE), which indicates standard or extended format, a reserved bit (ro) and, in the last four bits, a count of the data bytes (DLC). The data field ranges from zero to eight bytes and is followed by the Cyclic Redundancy Check (CRC) used as a frame integrity check for detecting bit errors.

The acknowledgement (ACK) field comprises the ACK slot and the ACK delimiter. The bit in the ACK slot is placed on the bus by the transmitter as a recessive bit (logical 1). It is overwritten as a dominant bit (logical 0) by those receivers which have at this time received the data correctly. In this way, the transmitting node can be assured that at least one receiver has correctly received its message. Note that messages are acknowledged by the receivers regardless of the outcome of the acceptance test.

The end of the message is indicated by the End Of Frame (EOF). The intermission field defines the minimum number of bit periods separating consecutive messages. If there is no subsequent bus access by any station, the bus remains idle.

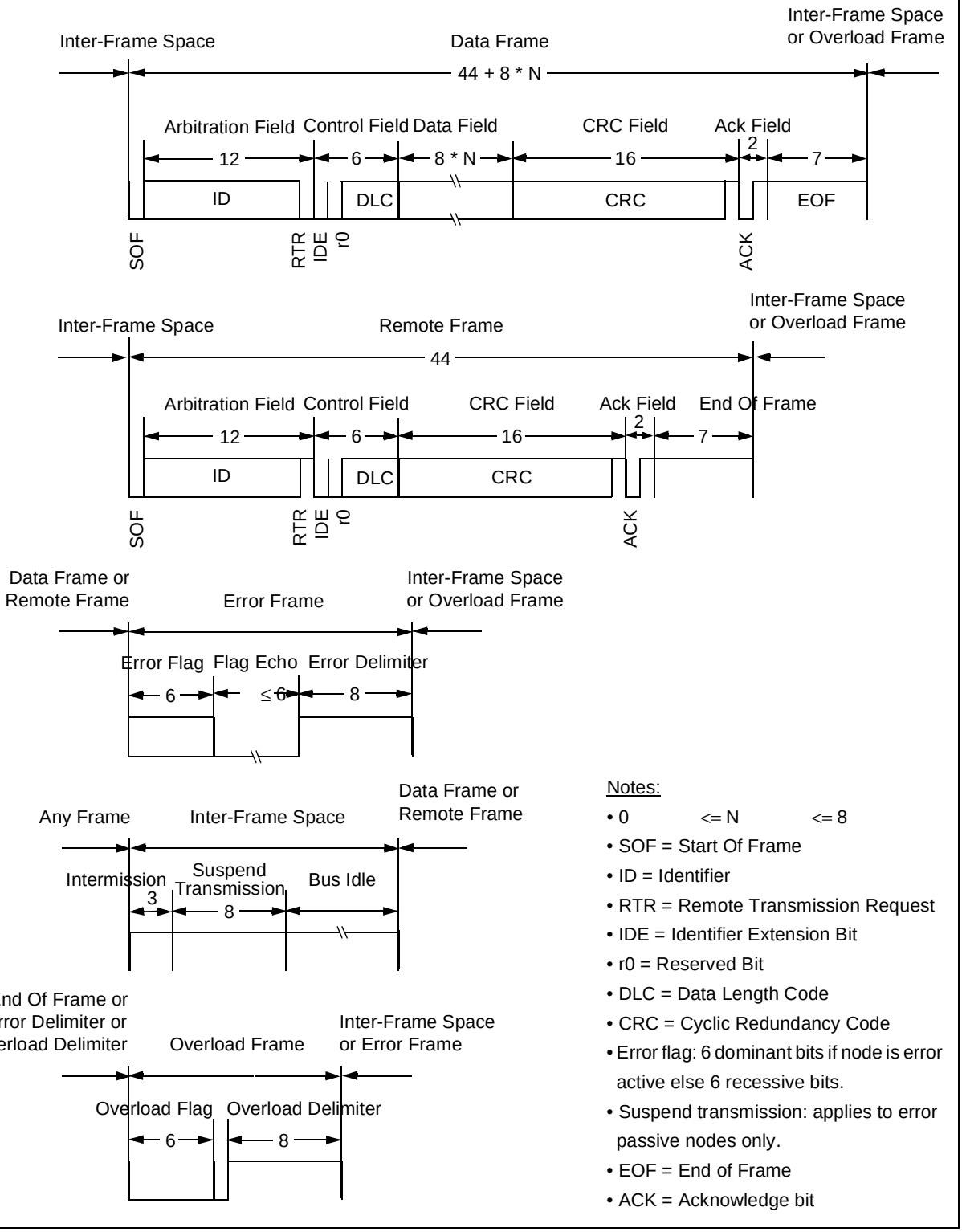
7.8.3.2 Hardware Blocks

The CAN controller contains the following functional blocks (refer to Figure 61):

– ST7 Interface: buffering of the ST7 internal bus and address decoding of the CAN registers.
– TX/RX Buffers: three 10-byte buffers for transmission and reception of maximum length messages.
– ID Filters: two 12-bit compare and don't care masks for message acceptance filtering.
– PSR: page selection register (see memory map).
– BRPR: clock divider for different data rates.
– BTR: bit timing register.
– ICR: interrupt control register.
– ISR: interrupt status register.
– CSR: general purpose control/status register.
– TECR: transmit error counter register.
– RECR: receive error counter register.
– BTL: bit timing logic providing programmable bit sampling and bit clock generation for synchronization of the controller.
– BCDL: bit coding logic generating a NRZ-coded datastream with stuff bits.
– SHREG: 8-bit shift register for serialization of data to be transmitted and parallelisation of received data.
– CRC: 15-bit CRC calculator and checker.
– EML: error detection and management logic.
– CAN Core: CAN 2.0B passive protocol controller.

CONTROLLER AREA NETWORK (Cont'd)

Figure 62. CAN Frames



Notes:

- 0 <= N <= 8
- SOF = Start Of Frame
- ID = Identifier
- RTR = Remote Transmission Request
- IDE = Identifier Extension Bit
- r0 = Reserved Bit
- DLC = Data Length Code
- CRC = Cyclic Redundancy Code
- Error flag: 6 dominant bits if node is error active else 6 recessive bits.
- Suspend transmission: applies to error passive nodes only.
- EOF = End of Frame
- ACK = Acknowledge bit

CONTROLLER AREA NETWORK (Cont'd)

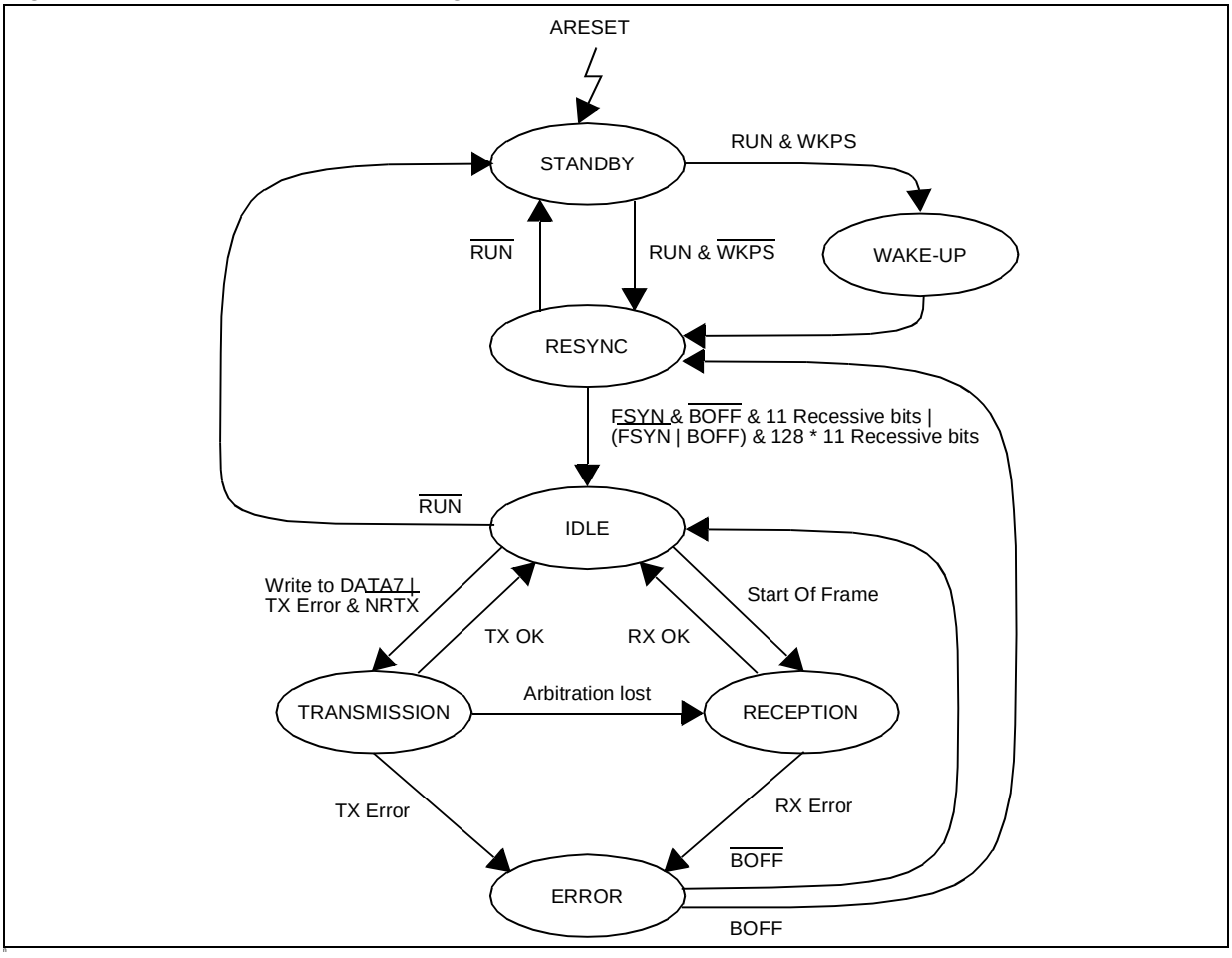
7.8.3.3 Modes of Operation

The CAN Core unit assumes one of the seven states described below:

– **STANDBY**. Standby mode is entered either on a chip reset or on resetting the RUN bit in the Control/Status Register (CSR). Any on-going transmission or reception operation is not interrupted and completes normally before the Bit Time Logic and the clock prescaler are turned off for minimum power consumption. This state is signalled by the RUN bit being read-back as 0.
Once in standby, the only event monitored is the reception of a dominant bit which causes a wake-up interrupt if the SCIE bit of the Interrupt Control Register (ICR) is set.
The STANDBY mode is left by setting the RUN bit. If the WKPS bit is set in the CSR register, then the controller passes through WAKE-UP otherwise it enters RESYNC directly.
It is important to note that the wake-up mechanism is software-driven and therefore carries a significant time overhead. All messages received after the wake-up bit and before the controller is set to run and has completed synchronization are ignored.

– **WAKE-UP**. The CAN bus line is forced to dominant for one bit time signalling the wake-up condition to all other bus members.

Figure 63. CAN Controller State Diagram



CONTROLLER AREA NETWORK (Cont'd)

– **RESYNC**. The resynchronization mode is used to find the correct entry point for starting transmission or reception after the node has gone asynchronous either by going into the STANDBY or bus-off states.
Resynchronization is achieved when 128 sequences of 11 recessive bits have been monitored unless the node is not bus-off and the FSYN bit in the CSR register is set in which case a single sequence of 11 recessive bits needs to be monitored.

– **IDLE**. The CAN controller looks for one of the following events: the RUN bit is reset, a Start Of Frame appears on the CAN bus or the DATA7 register of the currently active page is written to.

– **TRANSMISSION**. Once the LOCK bit of a Buffer Control/Status Register (BCSRx) has been set and read back as such, a transmit job can be submitted by writing to the DATA7 register. The message with the highest priority will be transmitted as soon as the CAN bus becomes idle. Among those messages with a pending transmission request, the highest priority is given to Buffer 3 then 2 and 1. If the transmission fails due to a lost arbitration or to an error while the NRTX bit of the CSR register is reset, then a new transmission attempt is performed . This goes on until the transmission ends successfully or until the job is cancelled by unlocking the buffer, by setting the NRTX bit or if the node ever enters bus-off or if a higher priority message becomes pending. The RDY bit in the BCSRx register, which was set since the job was submitted, gets reset. When a transmission is in progress, the BUSY bit in the BCSRx register is set. If it ends successfully then the TXIF bit in the Interrupt Status Register (ISR) is set, else the TEIF bit is set. An interrupt is generated in either case provided the TXIE and TEIE bits of the ICR register are set. The ETX bit in the same register is used to get an early transmit interrupt and to automatically unlock the transmitting buffer upon successful completion of its job. This enables the CPU to get a new transmit job pending by the end of the current transmission while always leaving two buffers available for reception. An uninterrupted stream of messages may be transmitted in this way at no overrun risk.

Note 1: Setting the SRTE bit of the CSR register allows transmitted messages to be simultaneously received when they pass the acceptance filtering. This is particularly useful for checking the integrity of the communication path.
Note 2: When the ETX bit is reset, the buffer with the highest priority and with a pending transmission request is always transmitted. When the ETX bit is set, once a buffer participates in the arbitration phase, it is sent until it wins the arbitration even if another transmission is requested from a buffer with a higher priority.

– **RECEPTION**. Once the CAN controller has synchronized itself onto the bus activity, it is ready for reception of new messages. Every incoming message gets its identifier compared to the acceptance filters. If the bitwise comparison of the selected bits ends up with a match for at least one of the filters then that message is elected for reception and a target buffer is searched for. This buffer will be the first one - order is 1 to 3 - that has the LOCK and RDY bits of its BCSRx register reset.

  – When no such buffer exists then an overrun interrupt is generated if the ORIE bit of the ICR register has been set. In this case the identifier of the last message is made available in the Last Identifier Register (LIDHR and LIDLR) at least until it gets overwritten by a new identifier picked-up from the bus.

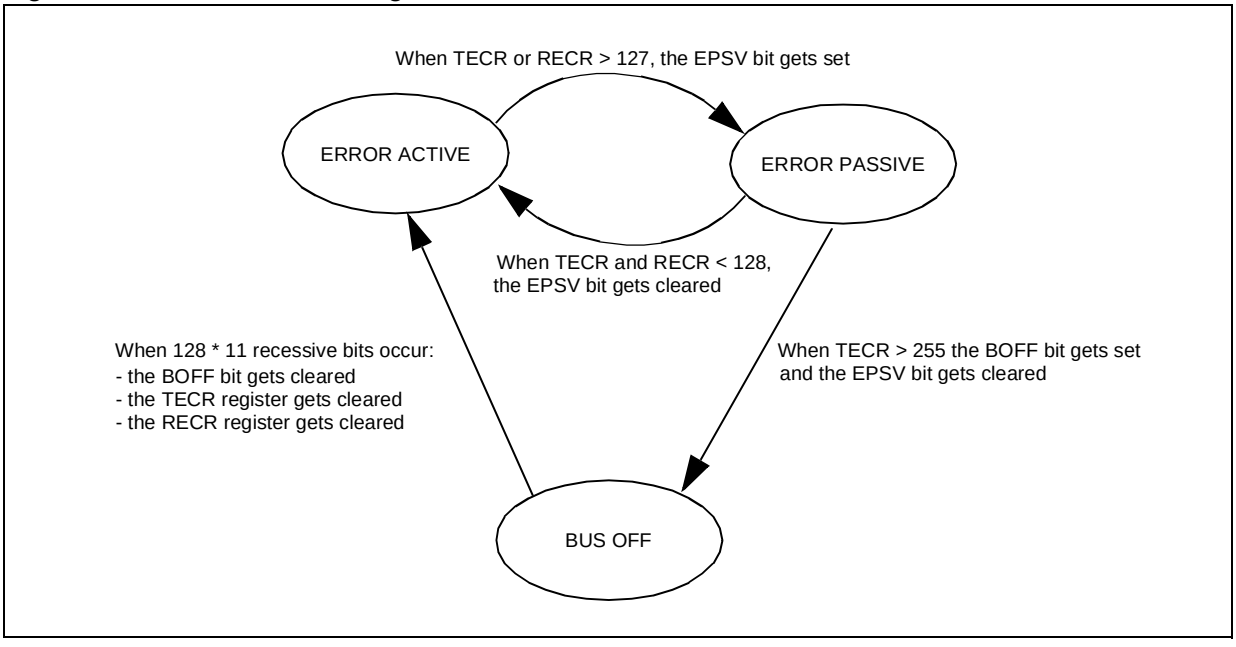
  – When a buffer does exist, the accepted message gets written into it, the ACC bit in the BCSRx register gets the number of the matching filter, the RDY and RXIF bits get set and an interrupt is generated if the RXIE bit in the ISR register is set.

  Up to three messages can be automatically received without intervention from the CPU because each buffer has its own set of status bits, greatly reducing the reactiveness requirements in the processing of the receive interrupts.

CONTROLLER AREA NETWORK (Cont'd)

– **ERROR**. The error management as described in the CAN protocol is completely handled by hardware using 2 error counters which get incremented or decremented according to the error condition. Both of them may be read by the application to determine the stability of the network. Moreover, as one of the node status bits (EPSV or BOFF of the CSR register) changes, an interrupt is generated if the SCIE bit is set in the ICR Register. Refer to Figure 64.

Figure 64. CAN Error State Diagram



CONTROLLER AREA NETWORK (Cont'd)

7.8.3.4 Bit Timing Logic

The bit timing logic monitors the serial bus-line and performs sampling and adjustment of the sample point by synchronizing on the start-bit edge and resynchronizing on following edges.

Its operation may be explained simply when the nominal bit time is divided into three segments as follows:

– **Synchronisation segment (SYNC_SEG)**: a bit change is expected to lie within this time segment. It has a fixed length of one time quanta (1 x t_{CAN}).

– **Bit segment 1 (BS1)**: defines the location of the sample point. It includes the PROP_SEG and PHASE_SEG1 of the CAN standard. Its duration is programmable between 1 and 16 time quanta but may be automatically lengthened to compensate for positive phase drifts due to differences in the frequency of the various nodes of the network.

– **Bit segment 2 (BS2)**: defines the location of the transmit point. It represents the PHASE_SEG2 of the CAN standard. Its duration is programmable between 1 and 8 time quanta but may also be automatically shortened to compensate for negative phase drifts.

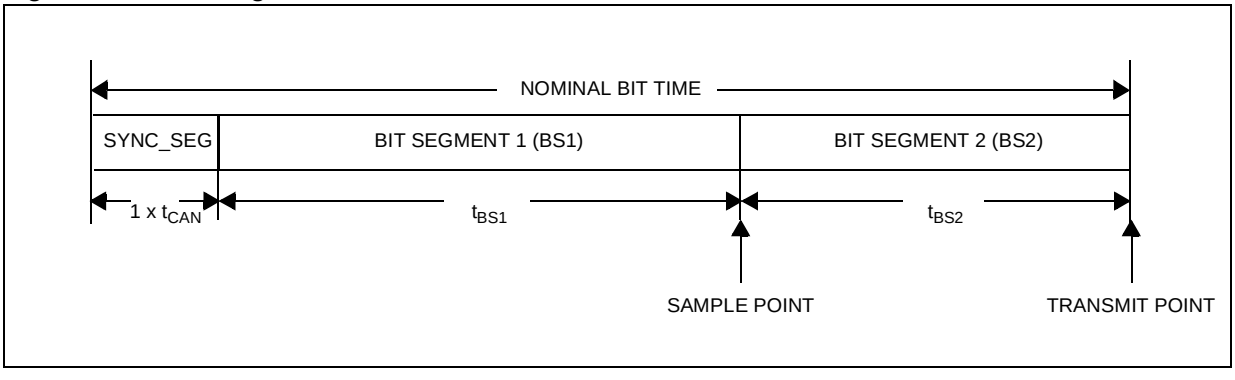
– **Resynchronization Jump Width (RJW)**: defines an upper bound to the amount of lengthening or shortening of the bit segments. It is programmable between 1 and 4 time quanta.

To guarantee the correct behaviour of the CAN controller, SYNC_SEG + BS1 + BS2 must be greater than or equal to 5 time quanta.

For a detailed description of the CAN resynchronization mechanism and other bit timing configuration constraints, please refer to the Bosch CAN standard 2.0.

As a safeguard against programming errors, the configuration of the Bit Timing Register (BTR) is only possible while the device is in STANDBY mode.

Figure 65. Bit Timing



CONTROLLER AREA NETWORK (Cont'd)

7.8.4 Register Description

The CAN registers are organized as 6 general purpose registers plus 5 pages of 16 registers spanning the same address space and primarily used for message and filter storage. The page actually selected is defined by the content of the Page Selection Register. Refer to Figure 66.

7.8.4.1 General Purpose Registers

INTERRUPT STATUS REGISTER (ISR)

Read/Write

Reset Value: 00h

7							0
RXIF3	RXIF2	RXIF1	TXIF	SCIF	ORIF	TEIF	EPND

Bit 7 = **RXIF3** *Receive Interrupt Flag for Buffer 3*
— Read/Clear
Set by hardware to signal that a new error-free message is available in buffer 3.
Cleared by software to release buffer 3.
Also cleared by resetting bit RDY of BCSR3.

Bit 6 = **RXIF2** *Receive Interrupt Flag for Buffer 2*
— Read/Clear
Set by hardware to signal that a new error-free message is available in buffer 2.
Cleared by software to release buffer 2.
Also cleared by resetting bit RDY of BCSR2.

Bit 5 = **RXIF1** *Receive Interrupt Flag for Buffer 1*
— Read/Clear
Set by hardware to signal that a new error-free message is available in buffer 1.
Cleared by software to release buffer 1.
Also cleared by resetting bit RDY of BCSR1.

Bit 4 = **TXIF** *Transmit Interrupt Flag*
— Read/Clear
Set by hardware to signal that the highest priority message queued for transmission has been successfully transmitted (ETX = 0) or that it has passed successfully the arbitration (ETX = 1).
Cleared by software.

Bit 3 = **SCIF** *Status Change Interrupt Flag*
— Read/Clear
Set by hardware to signal the reception of a dominant bit while in standby or a change from error active to error passive and bus-off while in run. Also signals any receive error when ESCI = 1.
Cleared by software.

Bit 2 = **ORIF** *Overrun Interrupt Flag*
— Read/Clear
Set by hardware to signal that a message could not be stored because no receive buffer was available.
Cleared by software.

Bit 1 = **TEIF** *Transmit Error Interrupt Flag*
— Read/Clear
Set by hardware to signal that an error occurred during the transmission of the highest priority message queued for transmission.
Cleared by software.

Bit 0 = **EPND** *Error Interrupt Pending*
— Read Only
Set by hardware when at least one of the three error interrupt flags SCIF, ORIF or TEIF is set.
Reset by hardware when all error interrupt flags have been cleared.

Caution;

Interrupt flags are reset by writing a "0" to the corresponding bit position. The appropriate way consists in writing an immediate mask or the one's complement of the register content initially read by the interrupt handler. Bit manipulation instruction BRES should never be used due to its read-modify-write nature.

CONTROLLER AREA NETWORK (Cont'd)

INTERRUPT CONTROL REGISTER (ICR)

Read/Write

Reset Value: 00h

7							0
0	ESCI	RXIE	TXIE	SCIE	ORIE	TEIE	ETX

Bit 6 = **ESCI** *Extended Status Change Interrupt*
— Read/Set/Clear
Set by software to specify that SCIF is to be set on receive errors also.
Cleared by software to set SCIF only on status changes and wake-up but not on all receive errors.

Bit 5 = **RXIE** *Receive Interrupt Enable*
— Read/Set/Clear
Set by software to enable an interrupt request whenever a message has been received free of errors.
Cleared by software to disable receive interrupt requests.

Bit 4 = TXIE *Transmit Interrupt Enable*
— Read/Set/Clear
Set by software to enable an interrupt request whenever a message has been successfully transmitted.
Cleared by software to disable transmit interrupt requests.

Bit 3 = **SCIE** *Status Change Interrupt Enable*
— Read/Set/Clear
Set by software to enable an interrupt request whenever the node's status changes in run mode or whenever a dominant pulse is received in standby mode.
Cleared by software to disable status change interrupt requests.

Bit 2 = **ORIE** *Overrun Interrupt Enable*
— Read/Set/Clear
Set by software to enable an interrupt request whenever a message should be stored and no receive buffer is avalaible.
Cleared by software to disable overrun interrupt requests.

Bit 1 = **TEIE** *Transmit Error Interrupt Enable*
— Read/Set/Clear
Set by software to enable an interrupt whenever an error has been detected during transmission of a message.
Cleared by software to disable transmit error interrupts.

Bit 0 = **ETX** *Early Transmit Interrupt*
— Read/Set/Clear
Set by software to request the transmit interrupt to occur as soon as the arbitration phase has been passed successfully.
Cleared by software to request the transmit interrupt to occur at the completion of the transfer.

CONTROLLER AREA NETWORK (Cont'd)

CONTROL/STATUS REGISTER (CSR)

Read/Write

Reset Value: 00h

7							0
0	BOFF	EPSV	SRTE	NRTX	FSYN	WKPS	RUN

Bit 6 = **BOFF** *Bus-Off State*
— Read Only
Set by hardware to indicate that the node is in bus-off state, i.e. the Transmit Error Counter exceeds 255.
Reset by hardware to indicate that the node is involved in bus activities.

Bit 5 = **EPSV** *Error Passive State*
— Read Only
Set by hardware to indicate that the node is error passive.
Reset by hardware to indicate that the node is either error active (BOFF = 0) or bus-off.

Bit 4 = **SRTE** *Simultaneous Receive/Transmit Enable* — Read/Set/Clear
Set by software to enable simultaneous transmission and reception of a message passing the acceptance filtering. Allows to check the integrity of the communication path.
Reset by software to discard all messages transmitted by the node. Allows remote and data frames to share the same identifier.

Bit 3 = **NRTX** *No Retransmission*
— Read/Set/Clear
Set by software to disable the retransmission of unsuccessful messages.
Cleared by software to enable retransmission of messages until success is met.

Bit 2 = **FSYN** *Fast Synchronization*
— Read/Set/Clear
Set by software to enable a fast resynchronization when leaving standby mode, i.e. wait for only 11 recessive bits in a row.
Cleared by software to enable the standard resynchronization when leaving standby mode, i.e. wait for 128 sequences of 11 recessive bits.

Bit 1 = **WKPS** *Wake-up Pulse*
— Read/Set/Clear
Set by software to generate a dominant pulse when leaving standby mode.
Cleared by software for no dominant wake-up pulse.

Bit 0 = **RUN** *CAN Enable*
— Read/Set/Clear

Set by software to leave standby mode after 128 sequences of 11 recessive bits or just 11 recessive bits if FSYN is set.
Cleared by software to request a switch to the standby or low-power mode as soon as any on-going transfer is complete. Read-back as 1 in the meantime to enable proper signalling of the standby state. The CPU clock may therefore be safely switched OFF whenever RUN is read as 0.

CONTROLLER AREA NETWORK (Cont'd)

BAUD RATE PRESCALER REGISTER (BRPR)

Read/Write in Standby mode

Reset Value: 00h

7							0
RJW1	RJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0

RJW[1:0] determine the maximum number of time quanta by which a bit period may be shortened or lengthened to achieve resynchronization.
$t_{RJW} = t_{CAN} * (RJW + 1)$

BRP[5:0] determine the CAN system clock cycle time or time quanta which is used to build up the individual bit timing.
$t_{CAN} = t_{CPU} * (BRP + 1)$

Where t_{CPU} = time period of the CPU clock.
The resulting baud rate can be computed by the formula:

$$BR = \frac{1}{t_{CPU} \times (BRP + 1) \times (BS1 + BS2 + 3)}$$

Note: Writing to this register is allowed only in Standby mode to prevent any accidental CAN protocol violation through programming errors.

BIT TIMING REGISTER (BTR)

Read/Write in Standby mode

Reset Value: 23h

7							0
0	BS22	BS21	BS20	BS13	BS12	BS11	BS10

BS2[2:0] determine the length of Bit Segment 2.
$t_{BS2} = t_{CAN} * (BS2 + 1)$

BS1[3:0] determine the length of Bit Segment 1.
$t_{BS1} = t_{CAN} * (BS1 + 1)$

Note: Writing to this register is allowed only in Standby mode to prevent any accidental CAN protocol violation through programming errors.

PAGE SELECTION REGISTER (PSR)

Read/Write

Reset Value: 00h

7							0
0	0	0	0	0	PAGE 2	PAGE 1	PAGE 0

PAGE[2:0] determine which buffer or filter page is mapped at addresses 0010h to 001Fh.

PAGE2	PAGE1	PAGE0	Page Title
0	0	0	Diagnosis
0	0	1	Buffer 1
0	1	0	Buffer 2
0	1	1	Buffer 3
1	0	0	Filters
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

CONTROLLER AREA NETWORK (Cont'd)

7.8.4.2 Paged Registers

LAST IDENTIFIER HIGH REGISTER (LIDHR)

Read/Write

Reset Value: Undefined

7							0
LID10	LID9	LID8	LID7	LID6	LID5	LID4	LID3

LID[10:3] are the most significant 8 bits of the last Identifier read on the CAN bus.

LAST IDENTIFIER LOW REGISTER (LIDLR)

Read/Write

Reset Value: Undefined

7							0
LID2	LID1	LID0	LRTR	LDLC 3	LDLC 2	LDLC 1	LDLC 0

LID[2:0] are the least significant 3 bits of the last Identifier read on the CAN bus.

LRTR is the last Remote Transmission Request bit read on the CAN bus.

LDLC[3:0] is the last Data Length Code read on the CAN bus.

TRANSMIT ERROR COUNTER REG. (TECR)

Read Only

Reset Value: 00h

7							0
TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0

TEC[7:0] is the least significant byte of the 9-bit Transmit Error Counter implementing part of the fault confinement mechanism of the CAN protocol. In case of an error during transmission, this counter is incremented by 8. It is decremented by 1 after every successful transmission. When the counter value exceeds 127, the CAN controller enters the error passive state. When a value of 256 is reached, the CAN controller is disconnected from the bus.

RECEIVE ERROR COUNTER REG. (RECR)

Page: 00h — Read Only

Reset Value: 00h

7							0
REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0

REC[7:0] is the Receive Error Counter implementing part of the fault confinement mechanism of the CAN protocol. In case of an error during reception, this counter is incremented by 1 or by 8 depending on the error condition as defined by the CAN standard. After every successful reception the counter is decremented by 1 or reset to 120 if its value was higher than 128. When the counter value exceeds 127, the CAN controller enters the error passive state.

IDENTIFIER HIGH REGISTERS (IDHRx)

Read/Write

Reset Value: Undefined

7							0
ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3

ID[10:3] are the most significant 8 bits of the 11-bit message identifier.The identifier acts as the message's name, used for bus access arbitration and acceptance filtering.

CONTROLLER AREA NETWORK (Cont'd)

IDENTIFIER LOW REGISTERS (IDLRx)

Read/Write

Reset Value: Undefined

7							0
ID2	ID1	ID0	RTR	DLC3	DLC2	DLC1	DLC0

ID[2:0] are the least significant 3 bits of the 11-bit message identifier.

RTR is the Remote Transmission Request bit. It is set to indicate a remote frame and reset to indicate a data frame.

DLC[3:0] is the Data Length Code. It gives the number of bytes in the data field of the message.The valid range is 0 to 8.

DATA REGISTERS (DATA0-7x)

Read/Write

Reset Value: Undefined

7							0
DATA 7	DATA 6	DATA 5	DATA 4	DATA 3	DATA 2	DATA 1	DATA 0

DATA[7:0] is a message data byte. Up to eight such bytes may be part of a message. Writing to byte DATA7 initiates a transmit request and should always be done even when DATA7 is not part of the message.

BUFFER CONTROL/STATUS REGs. (BCSRx)

Read/Write

Reset Value: 00h

7							0
0	0	0	0	ACC	RDY	BUSY	LOCK

Bit 3 = **ACC** *Acceptance Code*
— Read Only
Set by hardware with the id of the highest priority filter which accepted the message stored in the buffer.
ACC = 0: Match for Filter/Mask0. Possible match for Filter/Mask1.
ACC = 1: No match for Filter/Mask0 and match for Filter/Mask1.

Reset by hardware when either RDY or RXIF gets reset.

Bit 2 = **RDY** *Message Ready*
— Read/Clear
Set by hardware to signal that a new error-free message is available (LOCK = 0) or that a transmission request is pending (LOCK = 1).
Cleared by software when LOCK = 0 to release the buffer and to clear the corresponding RXIF bit in the Interrupt Status Register.
Cleared by hardware when LOCK = 1 to indicate that the transmission request has been serviced or cancelled.

Bit 1 = **BUSY** *Busy Buffer*
— Read Only
Set by hardware when the buffer is being filled (LOCK = 0) or emptied (LOCK = 1).
Reset by hardware when the buffer is not accessed by the CAN core for transmission nor reception purposes.

Bit 0 = **LOCK** *Lock Buffer*
— Read/Set/Clear
Set by software to lock a buffer. No more message can be received into the buffer thus preserving its content and making it available for transmission.
Cleared by software to make the buffer available for reception. Cancels any pending transmission request.
Cleared by hardware once a message has been successfully transmitted provided the early transmit interrupt mode is on. Left untouched otherwise.

Note that in order to prevent any message corruption or loss of context, LOCK cannot be set nor reset while BUSY is set. Trying to do so will result in LOCK not changing state.

CONTROLLER AREA NETWORK (Cont'd)

FILTER HIGH REGISTERS (FHRx)

Read/Write

Reset Value: Undefined

7							0
FIL11	FIL10	FIL9	FIL8	FIL7	FIL6	FIL5	FIL4

FIL[11:3] are the most significant 8 bits of a 12-bit message filter. The acceptance filter is compared bit by bit with the identifier and the RTR bit of the incoming message. If there is a match for the set of bits specified by the acceptance mask then the message is stored in a receive buffer.

FILTER LOW REGISTERS (FLRx)

Read/Write

Reset Value: Undefined

7							0
FIL3	FIL2	FIL1	FIL0	0	0	0	0

FIL[3:0] are the least significant 4 bits of a 12-bit message filter.

MASK HIGH REGISTERS (MHRx)

Read/Write

Reset Value: Undefined

7							0
MSK11	MSK10	MSK9	MSK8	MSK7	MSK6	MSK5	MSK4

MSK[11:3] are the most significant 8 bits of a 12-bit message mask. The acceptance mask defines which bits of the acceptance filter should match the identifier and the RTR bit of the incoming message.
MSK_i = 0: don't care.
MSK_i = 1: match required.

MASK LOW REGISTERS (MLRx)

Read/Write

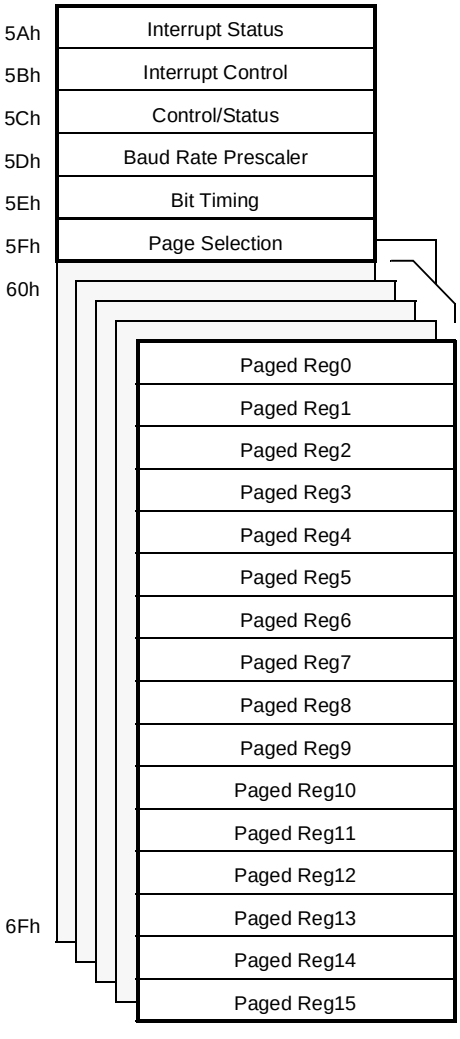
Reset Value: Undefined

7							0
MSK3	MSK2	MSK1	MSK0	0	0	0	0

MSK[3:0] are the least significant 4 bits of a 12-bit message mask.

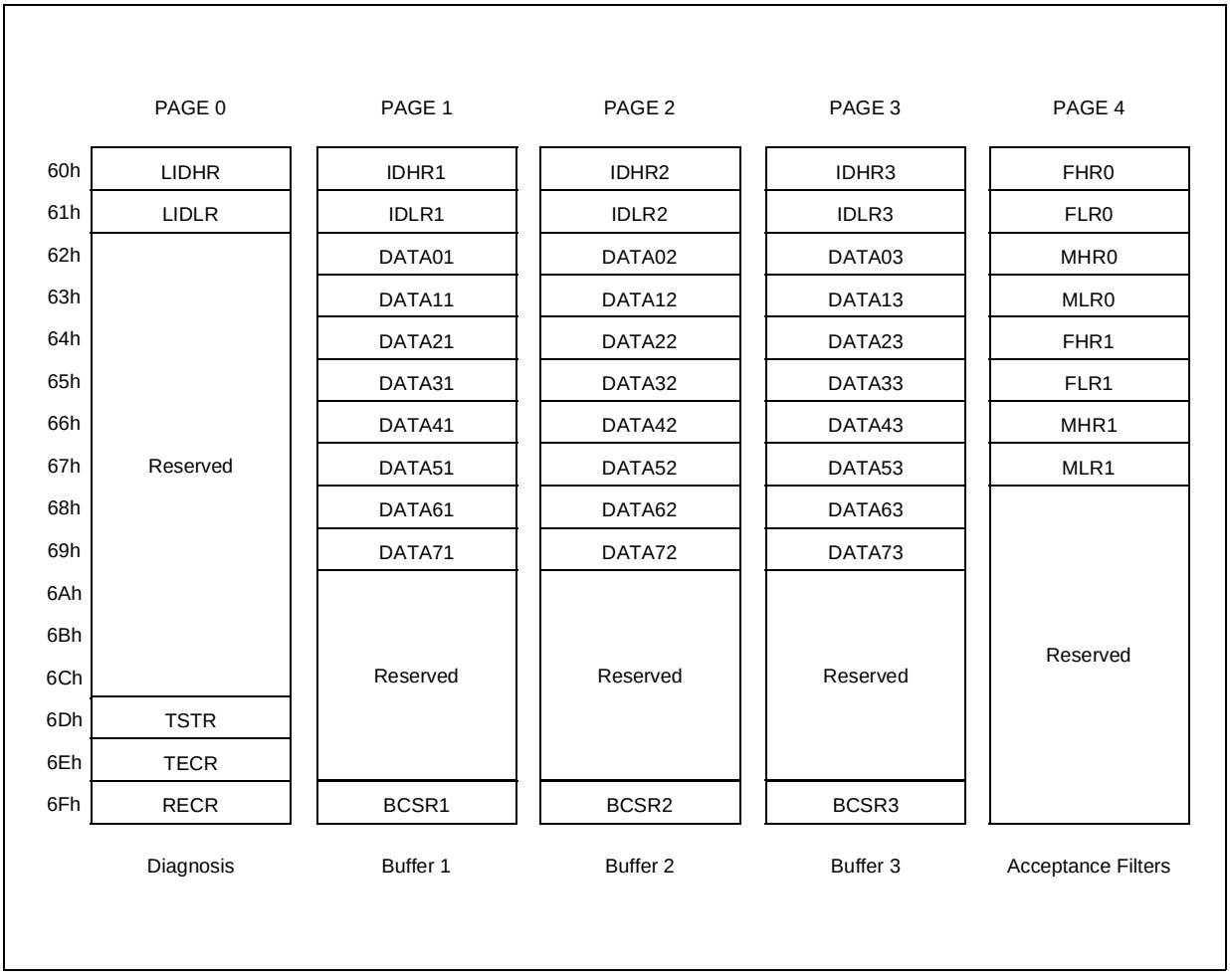
CONTROLLER AREA NETWORK (Cont'd)

Figure 66. CAN Register Map



CONTROLLER AREA NETWORK (Cont'd)

Figure 67. Page Maps



CONTROLLER AREA NETWORK (Cont'd)

Table 27. CAN Register Map and Reset Values

Address (Hex.)	Page	Register Label	7	6	5	4	3	2	1	0
5A		**CANISR**	RXIF3	RXIF2	RXIF1	TXIF	SCIF	ORIF	TEIF	EPND
		Reset Value	0	0	0	0	0	0	0	0
5B		**CANICR**		ESCI	RXIE	TXIE	SCIE	ORIE	TEIE	ETX
		Reset Value	0	0	0	0	0	0	0	0
5C		**CANCSR**		BOFF	EPSV	SRTE	NRTX	FSYN	WKPS	RUN
		Reset Value	0	0	0	0	0	0	0	0
5D		**CANBRPR**	RJW1	RJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
		Reset Value	0	0	0	0	0	0	0	0
5E		**CANBTR**		BS22	BS21	BS20	BS13	BS12	BS11	BS10
		Reset Value	0	0	1	0	0	0	1	1
5F		**CANPSR**						PAGE2	PAGE1	PAGE0
		Reset Value	0	0	0	0	0	0	0	0
60	0	**CANLIDHR**	LID10	LID9	LID8	LID7	LID6	LID5	LID4	LID3
		Reset Value	x	x	x	x	x	x	x	x
	1 to 3	**CANIDHRx**	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
		Reset Value	x	x	x	x	x	x	x	x
60, 64	4	**CANFHRx**	FIL11	FIL10	FIL9	FIL8	FIL7	FIL6	FIL5	FIL4
		Reset Value	x	x	x	x	x	x	x	x
61	0	**CANLIDLR**	LID2	LID1	LID0	LRTR	LDLC3	LDLC2	LDLC1	LDLC0
		Reset Value	x	x	x	x	x	x	x	x
	1 to 3	**CANIDLRx**	ID2	ID1	ID0	RTR	DLC3	DLC2	DLC1	DLC0
		Reset Value	x	x	x	x	x	x	x	x
61, 65	4	**CANFLRx**	FIL3	FIL2	FIL1	FIL0				
		Reset Value	x	x	x	x	0	0	0	0
62 to 69	1 to 3	**CANDRx**	MSB							LSB
		Reset Value	x	x	x	x	x	x	x	x
62, 66	4	**CANMHRx**	MSK11	MSK10	MSK9	MSK8	MSK7	MSK6	MSK5	MSK4
		Reset Value	x	x	x	x	x	x	x	x
63, 67	4	**CANMLRx**	MSK3	MSK2	MSK1	MSK0				
		Reset Value	x	x	x	x	0	0	0	0
6E	0	**CANTECR**	MSB							LSB
		Reset Value	0	0	0	0	0	0	0	0
6F		**CANRECR**	MSB							LSB
		Reset Value	0	0	0	0	0	0	0	0
	1 to 3	**CANBCSRx**					ACC	RDY	BUSY	LOCK
		Reset Value	0	0	0	0	0	0	0	0

7.9 8-BIT A/D CONVERTER (ADC)

7.9.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 8-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 8 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 8 different sources.

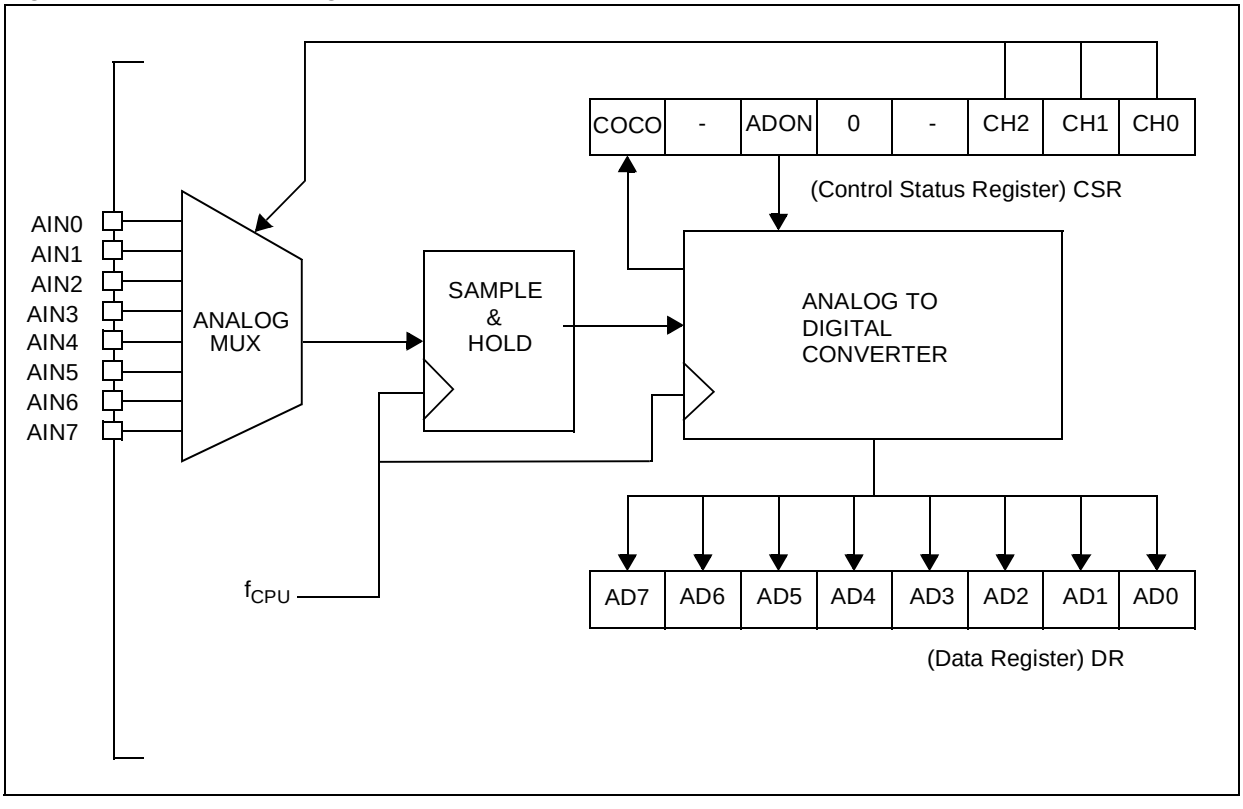
The result of the conversion is stored in a 8-bit Data Register. The A/D converter is controlled through a Control/Status Register.

7.9.2 Main Features

- 8-bit conversion
- Up to 8 channels with multiplexed input
- Linear successive approximation
- Data register (DR) which contains the results
- Conversion complete status flag
- On/Off bit (to reduce consumption)

The block diagram is shown in Figure 68.

Figure 68. ADC Block Diagram



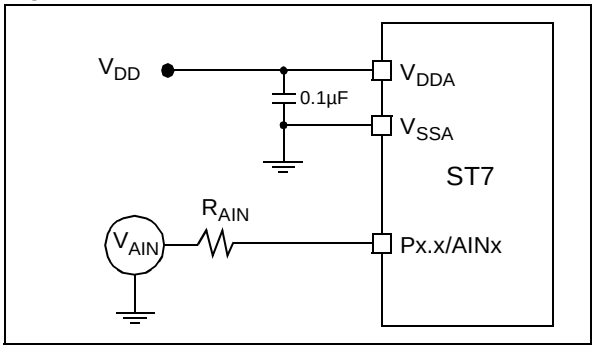
8-BIT A/D CONVERTER (ADC) (Cont'd)

7.9.3 Functional Description

The high level reference voltage V_{DDA} must be connected externally to the V_{DD} pin. The low level reference voltage V_{SSA} must be connected externally to the V_{SS} pin. In some devices (refer to device pin out description) high and low level reference voltages are internally connected to the V_{DD} and V_{SS} pins.

Conversion accuracy may therefore be degraded by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines.

Figure 69. Recommended Ext. Connections



Characteristics:

The conversion is monotonic, meaning the result never decreases if the analog input does not and never increases if the analog input does not.

If input voltage is greater than or equal to V_{DD} (voltage reference high) then results = FFh (full scale) without overflow indication.

If input voltage ≤ V_{SS} (voltage reference low) then the results = 00h.

The conversion time is 64 CPU clock cycles including a sampling time of 31.5 CPU clock cycles.

R_{AIN} is the maximum recommended impedance for an analog input signal. If the impedance is too high, this will result in a loss of accuracy due to leakage and sampling not being completed in the alloted time.

The A/D converter is linear and the digital result of the conversion is given by the formula:

$$\text{Digital result} = \frac{255 \times \text{Input Voltage}}{\text{Reference Voltage}}$$

Where Reference Voltage is V_{DD} - V_{SS}.

The accuracy of the conversion is described in the Electrical Characteristics Section.

Procedure:

Refer to the CSR and DR register description section for the bit definitions.

Each analog input pin must be configured as input, no pull-up, no interrupt. Refer to the "I/O Ports" chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

In the CSR register:

– Select the CH2 to CH0 bits to assign the analog channel to convert. Refer to Table 28 Channel Selection.

– Set the ADON bit. Then the A/D converter is enabled after a stabilization time (typically 30 µs). It then performs a continuous conversion of the selected channel.

When a conversion is complete

– The COCO bit is set by hardware.

– No interrupt is generated.

– The result is in the DR register.

A write to the CSR register aborts the current conversion, resets the COCO bit and starts a new conversion.

7.9.4 Low Power Modes

Note: The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed.

Mode	Description
WAIT	No effect on A/D Converter
HALT	A/D Converter disabled. After wakeup from Halt mode, the A/D Converter requires a stabilisation time before accurate conversions can be performed.

7.9.5 Interrupts

None.

8-BIT A/D CONVERTER (ADC) (Cont'd)

7.9.6 Register Description

CONTROL/STATUS REGISTER (CSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
COCO	-	ADON	0	-	CH2	CH1	CH0

Bit 7 = **COCO** *Conversion Complete*

This bit is set by hardware. It is cleared by software reading the result in the DR register or writing to the CSR register.
0: Conversion is not complete.
1: Conversion can be read from the DR register.

Bit 6 = **Reserved**. Must always be cleared.

Bit 5 = **ADON** *A/D converter On*

This bit is set and cleared by software.
0: A/D converter is switched off.
1: A/D converter is switched on.

Note: A typical 30 µs delay time is necessary for the ADC to stabilize when the ADON bit is set.

Bit 4 = **Reserved**. Forced by hardware to 0.

Bit 3 = **Reserved**. Must always be cleared.

Bits 2:0: **CH[2:0]** *Channel Selection*

These bits are set and cleared by software. They select the analog input to convert.

Table 28. Channel Selection

Pin*	CH2	CH1	CH0
AIN0	0	0	0
AIN1	0	0	1
AIN2	0	1	0
AIN3	0	1	1
AIN4	1	0	0
AIN5	1	0	1
AIN6	1	1	0
AIN7	1	1	1

***IMPORTANT NOTE:** The number of pins AND the channel selection vary according to the device. REFER TO THE DEVICE PINOUT).

DATA REGISTER (DR)

Read Only

Reset Value: 0000 0000 (00h)

7							0
AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0

Bit 7:0 = **AD[7:0]** *Analog Converted Value*

This register contains the converted analog value in the range 00h to FFh.

Reading this register resets the COCO flag.

8-BIT A/D CONVERTER (ADC) (Cont'd)

Table 29. ADC Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0070h	**ADCDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
0071h	**ADCCSR**	COCO		ADON			CH2	CH1	CH0
	Reset Value	0	0	0	0	0	0	0	0

8 INSTRUCTION SET

8.1 CPU ADDRESSING MODES

The CPU features 17 different addressing modes which can be classified in 7 main groups:

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

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

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

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

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

Table 30. CPU Addressing Mode Overview

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

INSTRUCTION SET OVERVIEW (Cont'd)

8.1.1 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

8.1.2 Immediate

Immediate instructions have two 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

8.1.3 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.

8.1.4 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 sub-modes:

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.

8.1.5 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 sub-modes:

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.

INSTRUCTION SET OVERVIEW (Cont'd)

8.1.6 Indirect Indexed (Short, Long)

This is a combination of indirect and short indexed addressing modes. The operand is referenced by its memory address, which is defined by the unsigned addition of an index register value (X or Y) with a pointer value located in memory. The pointer address follows the opcode.

The indirect indexed addressing mode consists of two sub-modes:

Indirect Indexed (Short)

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

Indirect Indexed (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.

Table 31. Instructions Supporting Direct, Indexed, Indirect and Indirect Indexed Addressing Modes

Long and Short Instructions	Function
LD	Load
CP	Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Additions/Substractions operations
BCP	Bit Compare

Short Instructions Only	Function
CLR	Clear
INC, DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
BSET, BRES	Bit Operations
BTJT, BTJF	Bit Test and Jump Operations
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles
CALL, JP	Call or Jump subroutine

8.1.7 Relative mode (Direct, Indirect)

This addressing mode is used to modify the PC register value, by adding an 8-bit signed offset to it.

Available Relative Direct/Indirect Instructions	Function
JRxx	Conditional Jump
CALLR	Call Relative

The relative addressing mode consists of two submodes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

The offset is defined in memory, which address follows the opcode.

INSTRUCTION SET OVERVIEW (Cont'd)

8.2 INSTRUCTION GROUPS

The ST7 family devices use an Instruction Set consisting of 63 instructions. The instructions may be subdivided into 13 main groups as illustrated in the following table:

Load and Transfer	LD	CLR						
Stack operation	PUSH	POP	RSP					
Increment/Decrement	INC	DEC						
Compare and Tests	CP	TNZ	BCP					
Logical operations	AND	OR	XOR	CPL	NEG			
Bit Operation	BSET	BRES						
Conditional Bit Test and Branch	BTJT	BTJF						
Arithmetic operations	ADC	ADD	SUB	SBC	MUL			
Shift and Rotates	SLL	SRL	SRA	RLC	RRC	SWAP	SLA	
Unconditional Jump or Call	JRA	JRT	JRF	JP	CALL	CALLR	NOP	RET
Conditional Branch	JRxx							
Interruption management	TRAP	WFI	HALT	IRET				
Condition Code Flag modification	SIM	RIM	SCF	RCF				

Using a pre-byte

The instructions are described with one to four opcodes.

In order to extend the number of available opcodes for an 8-bit CPU (256 opcodes), three different prebyte opcodes are defined. These prebytes modify the meaning of the instruction they precede.

The whole instruction becomes:

PC-2 End of previous instruction

PC-1 Prebyte

PC opcode

PC+1 Additional word (0 to 2) according to the number of bytes required to compute the effective address

These prebytes enable instruction in Y as well as indirect addressing modes to be implemented. They precede the opcode of the instruction in X or the instruction using direct addressing mode. The prebytes are:

PDY 90 Replace an X based instruction using immediate, direct, indexed, or inherent addressing mode by a Y one.

PIX 92 Replace an instruction using direct, direct bit, or direct relative addressing mode to an instruction using the corresponding indirect addressing mode.
It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.

PIY 91 Replace an instruction using X indirect indexed addressing mode by a Y one.

INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src	I1	H	I0	N	Z	C
ADC	Add with Carry	A = A + M + C	A	M		H		N	Z	C
ADD	Addition	A = A + M	A	M		H		N	Z	C
AND	Logical And	A = A . M	A	M				N	Z	
BCP	Bit compare A, Memory	tst (A . M)	A	M				N	Z	
BRES	Bit Reset	bres Byte, #3	M							
BSET	Bit Set	bset Byte, #3	M							
BTJF	Jump if bit is false (0)	btjf Byte, #3, Jmp1	M							C
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M							C
CALL	Call subroutine									
CALLR	Call subroutine relative									
CLR	Clear		reg, M					0	1	
CP	Arithmetic Compare	tst(Reg - M)	reg	M				N	Z	C
CPL	One Complement	A = FFH-A	reg, M					N	Z	1
DEC	Decrement	dec Y	reg, M					N	Z	
HALT	Halt				1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC			I1	H	I0	N	Z	C
INC	Increment	inc X	reg, M					N	Z	
JP	Absolute Jump	jp [TBL.w]								
JRA	Jump relative always									
JRT	Jump relative									
JRF	Never jump	jrf *								
JRIH	Jump if Port B INT pin = 1	(no Port B Interrupts)								
JRIL	Jump if Port B INT pin = 0	(Port B interrupt)								
JRH	Jump if H = 1	H = 1 ?								
JRNH	Jump if H = 0	H = 0 ?								
JRM	Jump if I1:0 = 11	I1:0 = 11 ?								
JRNM	Jump if I1:0 <> 11	I1:0 <> 11 ?								
JRMI	Jump if N = 1 (minus)	N = 1 ?								
JRPL	Jump if N = 0 (plus)	N = 0 ?								
JREQ	Jump if Z = 1 (equal)	Z = 1 ?								
JRNE	Jump if Z = 0 (not equal)	Z = 0 ?								
JRC	Jump if C = 1	C = 1 ?								
JRNC	Jump if C = 0	C = 0 ?								
JRULT	Jump if C = 1	Unsigned <								
JRUGE	Jump if C = 0	Jmp if unsigned >=								
JRUGT	Jump if (C + Z = 0)	Unsigned >								

INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src	I1	H	I0	N	Z	C
JRULE	Jump if (C + Z = 1)	Unsigned <=								
LD	Load	dst <= src	reg, M	M, reg				N	Z	
MUL	Multiply	X,A = X * A	A, X, Y	X, Y, A		0				0
NEG	Negate (2's compl)	neg $10	reg, M					N	Z	C
NOP	No Operation									
OR	OR operation	A = A + M	A	M				N	Z	
POP	Pop from the Stack	pop reg	reg	M						
		pop CC	CC	M	I1	H	I0	N	Z	C
PUSH	Push onto the Stack	push Y	M	reg, CC						
RCF	Reset carry flag	C = 0								0
RET	Subroutine Return									
RIM	Enable Interrupts	I1:0 = 10 (level 0)			1		0			
RLC	Rotate left true C	C <= A <= C	reg, M					N	Z	C
RRC	Rotate right true C	C => A => C	reg, M					N	Z	C
RSP	Reset Stack Pointer	S = Max allowed								
SBC	Substract with Carry	A = A - M - C	A	M				N	Z	C
SCF	Set carry flag	C = 1								1
SIM	Disable Interrupts	I1:0 = 11 (level 3)			1		1			
SLA	Shift left Arithmetic	C <= A <= 0	reg, M					N	Z	C
SLL	Shift left Logic	C <= A <= 0	reg, M					N	Z	C
SRL	Shift right Logic	0 => A => C	reg, M					0	Z	C
SRA	Shift right Arithmetic	A7 => A => C	reg, M					N	Z	C
SUB	Substraction	A = A - M	A	M				N	Z	C
SWAP	SWAP nibbles	A7-A4 <=> A3-A0	reg, M					N	Z	
TNZ	Test for Neg & Zero	tnz lbl1						N	Z	
TRAP	S/W trap	S/W interrupt			1		1			
WFI	Wait for Interrupt				1		0			
XOR	Exclusive OR	A = A XOR M	A	M				N	Z	

9 ELECTRICAL CHARACTERISTICS

9.1 ABSOLUTE MAXIMUM RATINGS

This product contains devices to protect the inputs against damage due to high static voltages, however it is advisable to take normal precaution to avoid application of any voltage higher than the specified maximum rated voltages.

For proper operation it is recommended that V_I and V_O be higher than V_{SS} and lower than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriate logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, T_J, in Celsius can be obtained from:

T_J = TA + PD x RthJA

Where: T_A = Ambient Temperature.
RthJA = Package thermal resistance (junction-to ambient).
P_D = P_{INT} + P_{PORT}.
P_{INT} = I_{DD} x V_{DD} (chip internal power).
P_{PORT} = Port power dissipation determined by the user)

Symbol	Ratings	Value	Unit
V_{DD} - V_{SS}	Supply Voltage	6.5	V
V_{DDA} - V_{SSA}	Analog Reference Voltage $V_{DDA} > V_{SS}$	6.5	V
V_{LCD}	Max. Display Voltage	10.5	V
$\|V_{DD\_i} - V_{DD\_j}\|$ $\|V_{DD\_i} - V_{DDA}\|$	Max. variations (Power Line)	50	mV
$\|V_{SS\_i} - V_{SS\_j}\|$ $\|V_{SS\_i} - V_{SSA}\|$ $\|G_{LCD} - V_{SS\_i}\|$	Max. variations (Ground Line)	50	mV
V_{IN}	Input Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_{OUT}	Output Voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
ESD	ESD susceptibility	2000	V
I_{VDD_i}	Total current into V_{DD_i} (source)	80	mA
I_{VSS_i}	Total current out of V_{SS_i} (sink)	80	
I_{INJ}	Total injected current	+/- 5	

Note: Stresses above those listed as “absolute maximum ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

General Warning: Direct connection to V_{DD} or V_{SS} of the RESET and I/O pins could damage the device in case of unwanted internal reset generation or program counter corruption (due to unwanted change of the I/O configuration). To guarantee safe conditions, this connection has to be done through a typical 10KΩ pull-up or pull-down resistor.

Thermal Characteristics

Symbol	Ratings		Value	Unit
R_{thJA}	Package thermal resistance	PQFP128 EQFP128	42 na	°C/W
T_{Jmax}	Max. Junction Temperature		150	°C
T_{STG}	Storage Temperature Range		-65 to +150	°C
PD	Power Dissipation		500	mW

9.2 RECOMMENDED OPERATING CONDITIONS

GENERAL						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{DD}	Supply Voltage	f_{OSC} = 16MHz	4.5	5.0	5.5	V
f_{OSC}	Oscillator frequency	4.5V < V_{DD} < 5.5V	8		16	MHz
T_A	Ambient Temperature Range		-40		+85	°C

CURRENT INJECTION ON I/O PORT AND CONTROL PINS						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
I_{INJ+}	Total positive injected current (1)	$V_{EXTERNAL}$ > V_{DD}			5	mA
I_{INJ-}	Total negative injected current (2)	$V_{EXTERNAL}$ < V_{SS} Digital pins Analog pins			 1.6 0.8	mA

Note 1: *Positive injection*

The I_{inj+} is done through protection diodes insulated from the substrate of the die.

The pins which have a high voltage capability like true open-drain do not accept positive injection. In this case the maximum voltage rating must be followed.

Note 2: *ADC accuracy reduced by negative injection*

The I_{inj-} is done through protection diodes NOT INSULATED from the substrate of the die. The drawback is a small leakage (few µA) induced inside the die when a negative injection is performed. This leakage is tolerated by the digital structure, but it acts on the analog line according to the impedance versus a leakage current of few µA (if the MCU has an AD converter). The effect depends on the pin which is submitted to the injection. Of course, external digital signals applied to the component must have a maximum impedance close to 50KΩ.

Location of the negative current injection:

- The pins with analog input capability are the more sensitive. I_{inj-} maximum is 0.8 mA (assuming that the impedance of the analog voltage is lower than 25KΩ)
- The pure digital pins can tolerate 1.6mA. In addition, the best choice is to inject the current as far as possible from the analog input pins.

General Note: When several inputs are submitted to a current injection, the maximum I_{inj} is the sum of the positive and negative currents respectively (instantaneous values).

9.3 TIMING CHARACTERISTICS

(Operating conditions T_A = -40 to +85°C unless otherwise specified)

Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{OSC}	External Oscillator Frequency	V_{DD} = 4.5V	8		16	MHz
t_{DOG}	Watchdog Time-out	f_{CPU} = 8MHz	12,288 1.54		786,432 98.3	t_{CPU} ms
t_{INST}	Instruction Time		2		12	t_{CPU}
t_{IRT}	Interrupt Reaction Time	t_{IRT} = Δt_{INST} + 10*	10		22	t_{CPU}

* Δt_{INST} is the number of t_{CPU} to finish the current instruction execution.

9.4 ELECTRICAL CHARACTERISTICS

(T_A=-40 to +85°C, V_{DD}-V_{SS}=5V unless otherwise specified)

ST72E/T589BW5

Symbol	Parameter	Conditions	Min	Typ.	Max	Unit
V_{DD}	Operating Supply Voltage	f_{OSC}=16MHz, f_{CPU}=8MHz	4.5		5.5	V
I_{DD}	Supply Current in RUN mode 1)	f_{OSC}=16MHz f_{CPU}=8MHz			20	mA
	Supply Current in SLOW mode 1)	f_{OSC}=16MHz f_{CPU}=500KHz		1	3.5	
	Supply Current in WAIT mode 2)	f_{OSC}=16MHz f_{CPU}=8MHz		6	10	
	Supply Current in SLOW WAIT mode 3)	f_{OSC}=16MHz f_{CPU}=500KHz		2	2.5	
	Supply Current in ACTIVE-HALT mode 4)			0.7		
	Supply Current in HALT mode 4)	T_A<25°C		1	10	µA
		25°C<T_A<85°C			50	
V_{RM}	Data Retention Mode	HALT mode	2			V

(T_A=-40 to +85°C, V_{DD}-V_{SS}=5V unless otherwise specified)

ROM devices

Symbol	Parameter	Conditions	Min	Typ.	Max	Unit
V_{DD}	Operating Supply Voltage	f_{OSC}=16MHz, f_{CPU}=8MHz	4.5		5.5	V
I_{DD}	Supply Current in RUN mode 1)	f_{OSC}=16MHz f_{CPU}=8MHz			15	mA
	Supply Current in SLOW mode 1)	f_{OSC}=16MHz f_{CPU}=500KHz		1	3	
	Supply Current in WAIT mode 2)	f_{OSC}=16MHz f_{CPU}=8MHz		5	8	
	Supply Current in SLOW WAIT mode 3)	f_{OSC}=16MHz f_{CPU}=500KHz		2	2.5	
	Supply Current in ACTIVE-HALT mode 4)			0.7		
	Supply Current in HALT mode 4)			1	10	µA
V_{RM}	Data Retention Mode	HALT mode	2			V

Notes:
1) CPU running with memory access, all I/O pins in input mode with a static value at V_{DD} or V_{SS}; clock input (OSC1) driven by external square wave.
2) All I/O pins in input mode with a static value at V_{DD} or V_{SS}; clock input (OSC1) driven by external square wave.
3) WAIT Mode with SLOW Mode selected. Based on characterisation results, not tested.
4) All I/O pins in input mode with a static value at V_{DD} or V_{SS}.

9.5 I/O PORTS CHARACTERISTICS

(T_A=-40 to +85°C, voltages are referred to V_{SS} unless otherwise specified)

I/O PORT PINS (ST72E/T589BW5)						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{IL}	Input Low Level Voltage				$0.3xV_{DD}$	V
V_{IH}	Input High Level Voltage		$0.7xV_{DD}$			V
V_{HYS}	Schmitt Trigger Voltage Hysteresis			400*		mV
V_{OL}	Output Low Level Voltage	I=-5mA			1.3	V
		I=-2mA			0.4	V
V_{OH}	Output High Level Voltage	I=5mA	V_{DD}-1.3			V
		I=2mA	V_{DD}-0.4			V
I_L	Input Leakage Current	$V_{SS}<V_{PIN}<V_{DD}$			100	nA
I_{SV}	Static Current Consumption	Floating input mode			200	µA
R_{PU}	Pull-up Equivalent Resistance	V_{DD}=5V	10		50	KΩ
t_{ohl}	Output H-L Fall Time	C_l=50pF	14.8	25	45.6	ns
t_{olh}	Output L-H Rise Time	C_l=50pF	14.4	25	45.9	ns
t_{itext}	External Interrupt Pulse Time		1			t_{CPU}

Note: * Based on characterization results. Not tested.

(T_A=-40 to +85°C, voltages are referred to V_{SS} unless otherwise specified)

I/O PORT PINS (ROM devices)						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{IL}	Input Low Level Voltage				$0.3xV_{DD}$	V
V_{IH}	Input High Level Voltage		$0.7xV_{DD}$			V
V_{OL}	Output Low Level Voltage	I=-5mA			1.3	V
		I=-2mA			0.6	V
V_{OH}	Output High Level Voltage	I=5mA	V_{DD}-1.6			V
		I=2mA	V_{DD}-0.6			V
I_L	Input Leakage Current	$V_{SS}<V_{PIN}<V_{DD}$			100	nA
I_{SV}	Static Current Consumption	Floating input mode			200	µA
R_{PU}	Pull-up Equivalent Resistance	V_{DD}=5V	10		50	KΩ
t_{ohl}	Output H-L Fall Time	C_l=50pF	14.8	25	45.6	ns
t_{olh}	Output L-H Rise Time	C_l=50pF	14.4	25	45.9	ns
t_{itext}	External Interrupt Pulse Time		1			t_{CPU}

9.6 SUPPLY, RESET and CLOCK CHARACTERISTICS

The values given in the specifications are generally not applicable for all chips. Therefore, only the limits listed below are valid for the product.

The values below substitute the corresponding values in the specifications of dedicated functions.

(T_A=-40 to +85°C, V_{DD}-V_{SS}=5V unless otherwise specified)

RESET						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
R_{ON}	Reset weak pull-up resistance	$V_{IN} > V_{IH}$ $V_{IN} < V_{IL}$	20 60	40 120	80 240	KΩ
t_{PULSE}	External $\overline{RESET}$ pin Pulse time		1.5			t_{CPU}

POWER ON RESET						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{TN}	Re-initialization level *			0.8		V
V_{POR}	Reset generation level **		2.0	2.6	3.4	V

Notes: * V_{DD} has to fall down to V_{TN} to re-arm the POR function
** POR function reset the device through a pulse generation at V_{POR}.

MAIN EXTERNAL CLOCK SOURCE						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{OSC}	Main oscillator frequency	Square signal with 50% Duty Cycle			16	MHz
V_{OSC}	OSC1 pin voltage			V_{DD}		V

MAIN OSCILLATOR						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{OSC}	Main oscillator frequency		8		16	MHz
C_{Li}	Load Capacitances	R_{Smax}=100Ω *	10		20	pF
t_{START}	Oscillator start-up time	Depends on resonator quality. A typical value is 10ms				

Note: R_{Smax} is the equivalent serial resistance of the crystal or ceramic resonator.

9.7 MEMORY AND PERIPHERAL CHARACTERISTICS

(T_A=-40 to +85°C, V_{DD}-V_{SS}=5V unless otherwise specified)

EPROM						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
W_{ERASE}	UV lamp	Lamp wavelength 2537Å		15		Watt-sec/cm^2
t_{ERASE}	Erase Time	UV lamp is placed 1 inch from the device window without any interposed filters	15		20	min

LCD DRIVER						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{FR}	Frame frequency	f_{LCD}=16384Hz	64		512	Hz
V_{DCRC}	DC Residual Component	V_{LCD}=V_{DD} no load		100		mV
V_{COH}	COM high level, Output voltage	I=50µA, V_{LCD}=5V I=100µA, V_{LCD}=10V	4.5 9.5			V
V_{COL}	COM low level, Output voltage	I=100µA, V_{LCD}=5..10V			0.5	
V_{SOH}	SEG high level, Output voltage	I=25µA, V_{LCD}=5V I=25µA, V_{LCD}=10V	4.5 9.5			
V_{SOL}	SEG low level, Output voltage	I=25µA, V_{LCD}=5..10V			0.5	
V_{LCD}	Display voltage		3		10	
R_{LCDi}	Voltage divider resistances	1% accuracy		10		KΩ
C_{LCDi}	Voltage divider coupling capacitances			100		nF
C_{LOAD}	LCD dot load			50		pF

Notes: If V_{LCD}=$V_{LCDnominal}$ and V_{DD}<4.5V then V_{LCD} DC level is applied on SEG and COM outputs.

MEMORY AND PERIPHERAL CHARACTERISTICS - SCI, CAN

SCI Serial Communication Interface					
Symbol	**Parameter**	**Conditions**		**Typ**	**Unit**
f_{Tx} or f_{Rx}	Communication frequency (precision vs. standard ~0.16%)	f_{CPU}=8MHz	Standard Mode TR (resp.RR)=64, PR=13 TR (resp.RR)=16, PR=13 TR (resp.RR)= 8, PR=13 TR (resp.RR)= 4, PR=13 TR (resp.RR)= 2, PR=13 TR (resp.RR)= 8, PR= 3 TR (resp.RR)= 1, PR=13	 ~300.48 ~1201.92 ~2403.84 ~4807.69 ~9615.38 ~10416.67 ~19230,77	Hz
			Extended Mode ETPR (resp.ERPR) = 13	~38461.54	
See “STANDARD I/O PORT PINS” description for more details.					

CAN Controller Area Network						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{CPU}	Frequency of Operation	V_{DD} = 4.5V	DC		8	MHz

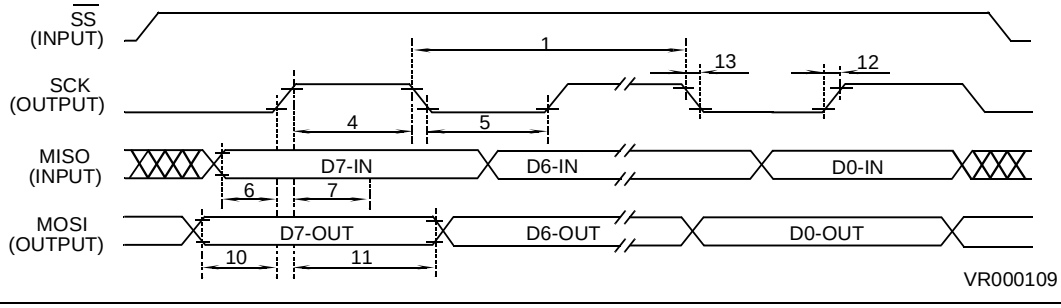
MEMORY AND PERIPHERAL CHARACTERISTICS - SPI

SPI Serial Peripheral Interface

Ref.	Symbol	Parameter	Condition	Value		Unit
				Min.	Max.	
	f_{SPI}	SPI frequency	Master Slave	1/128 dc	1/4 1/2	f_{CPU}
1	t_{SPI}	SPI clock period	Master Slave	4 2		t_{CPU}
2	t_{Lead}	Enable lead time	Slave	120		ns
3	t_{Lag}	Enable lag time	Slave	120		ns
4	t_{SPI_H}	Clock (SCK) high time	Master Slave	100 90		ns
5	t_{SPI_L}	Clock (SCK) low time	Master Slave	100 90		ns
6	t_{SU}	Data set-up time	Master Slave	100 100		ns
7	t_H	Data hold time (inputs)	Master Slave	100 100		ns
8	t_A	Access time (time to data active from high impedance state)	Slave	0	120	ns
9	t_{Dis}	Disable time (hold time to high impedance state)			240	ns
10	t_V	Data valid	Master (before capture edge) Slave (after enable edge)	0.25	 120	t_{CPU} ns
11	t_{Hold}	Data hold time (outputs)	Master (before capture edge) Slave (after enable edge)	0.25 0		t_{CPU} ns
12	t_{Rise}	Rise time (20% V_{DD} to 70% V_{DD}, C_L = 200pF)	Outputs: SCK,MOSI,MISO Inputs: SCK,MOSI,MISO,$\overline{SS}$		100 100	ns µs
13	t_{Fall}	Fall time (70% V_{DD} to 20% V_{DD}, C_L = 200pF)	Outputs: SCK,MOSI,MISO Inputs: SCK,MOSI,MISO,$\overline{SS}$		100 100	ns µs

Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH} in the SPI Timing Diagram

Figure 70. SPI Master Timing Diagram CPHA=0, CPOL=0



MEMORY AND PERIPHERAL CHARACTERISTICS- SPI (Cont'd)

Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH} in the SPI Timing Diagram

Figure 71. SPI Master Timing Diagram CPHA=0, CPOL=1

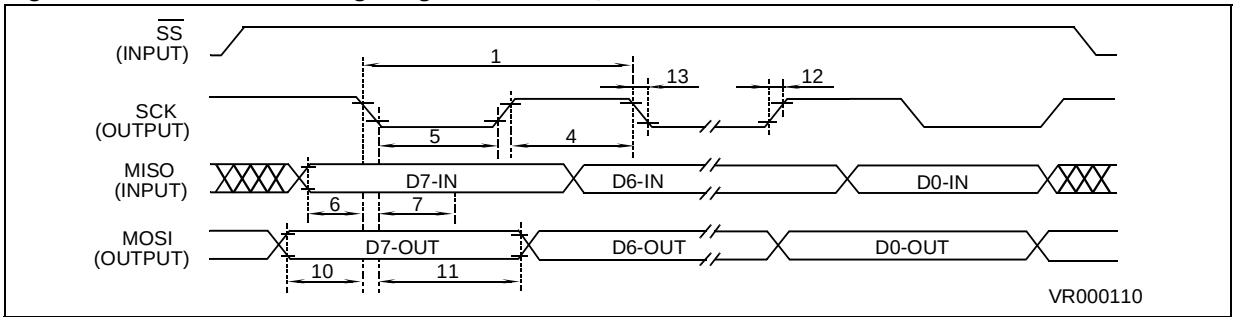


Figure 72. SPI Master Timing Diagram CPHA=1, CPOL=0

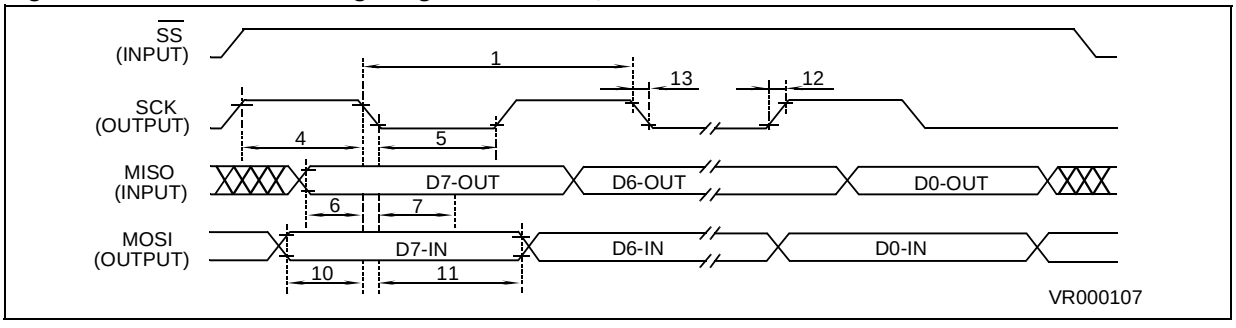
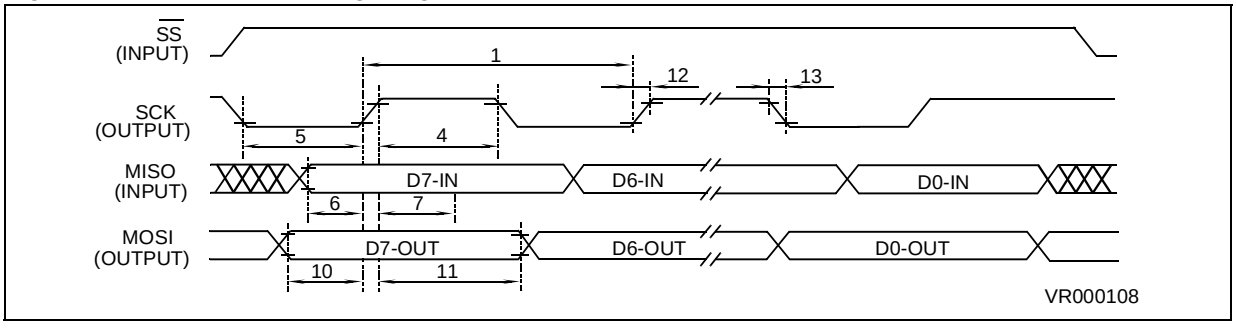


Figure 73. SPI Master Timing Diagram CPHA=1, CPOL=1



MEMORY AND PERIPHERAL CHARACTERISTICS - SPI (Cont'd)

Measurement points are V_{OL}, V_{OH}, V_{IL} and V_{IH} in the SPI Timing Diagram

Figure 74. SPI Slave Timing Diagram CPHA=0, CPOL=0

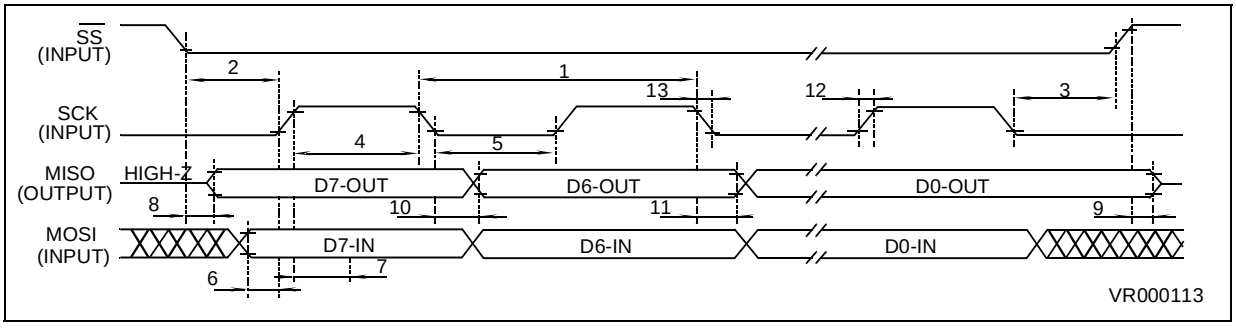


Figure 75. SPI Slave Timing Diagram CPHA=0, CPOL=1

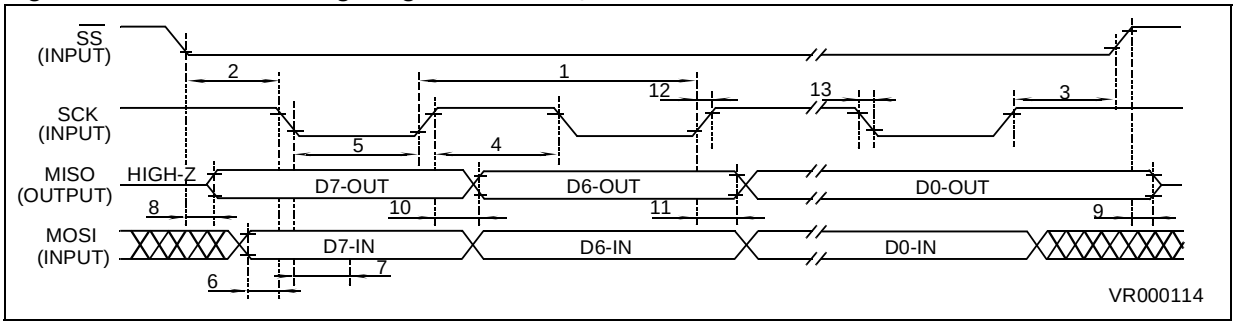


Figure 76. SPI Slave Timing Diagram CPHA=1, CPOL=0

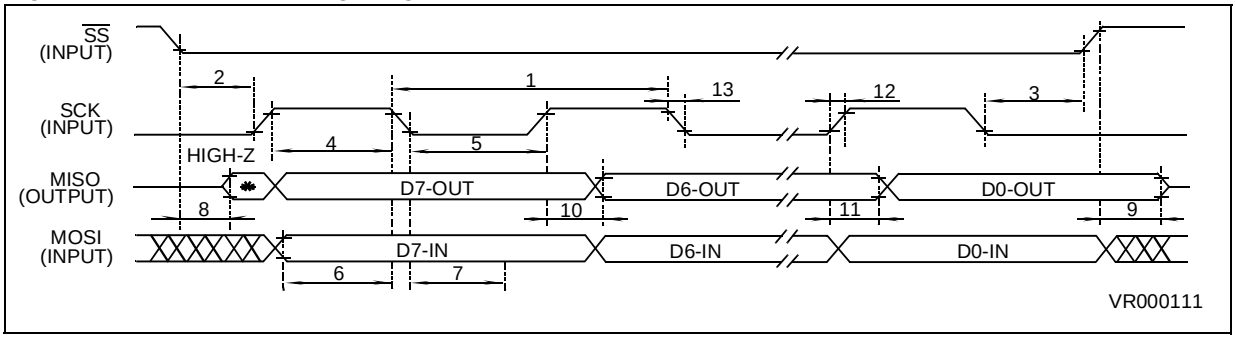
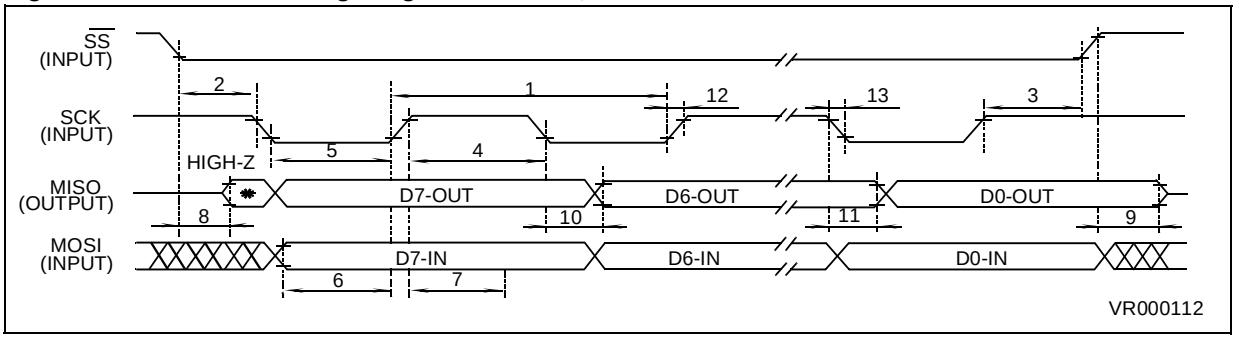


Figure 77. SPI Slave Timing Diagram CPHA=1, CPOL=1



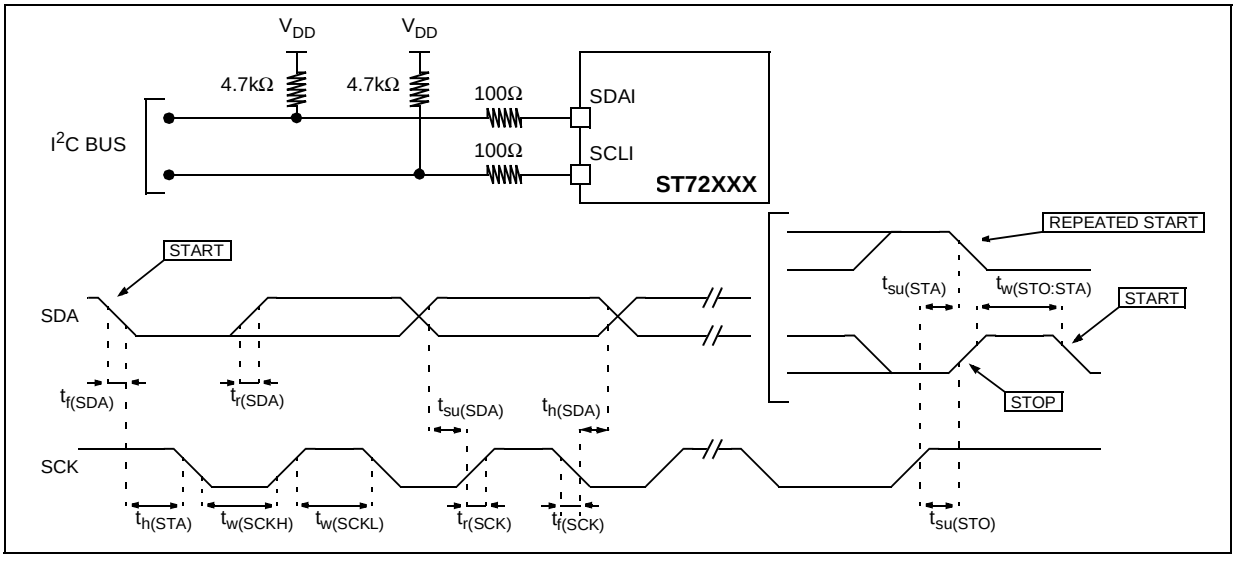
MEMORY AND PERIPHERAL CHARACTERISTICS - I2C

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (SDAI and SCLI). The ST7 I^2C interface meets the requirements of the Standard I^2C communication protocol described in the following table.

Symbol	Parameter	Standard mode I^2C		Fast mode I^2C		Unit
		Min 1)	Max 1)	Min 1)	Max 1)	
$t_{w(SCLL)}$	SCL clock low time	4.7		1.3		µs
$t_{w(SCLH)}$	SCL clock high time	4.0		0.6		
$t_{su(SDA)}$	SDA setup time	250		100		ns
$t_{h(SDA)}$	SDA data hold time	0 3)		0 2)	900 3)	
$t_{r(SDA)}$ $t_{r(SCL)}$	SDA and SCL rise time		1000	20+0.1C_b	300	
$t_{f(SDA)}$ $t_{f(SCL)}$	SDA and SCL fall time		300	20+0.1C_b	300	
$t_{h(STA)}$	START condition hold time	4.0		0.6		µs
$t_{su(STA)}$	Repeated START condition setup time	4.7		0.6		
$t_{su(STO)}$	STOP condition setup time	4.0		0.6		ns
$t_{w(STO:STA)}$	STOP to START condition time (bus free)	4.7		1.3		ms
C_b	Capacitive load for each bus line		400		400	pF

Figure 78. Typical Application with I^2C Bus and Timing Diagram 4)

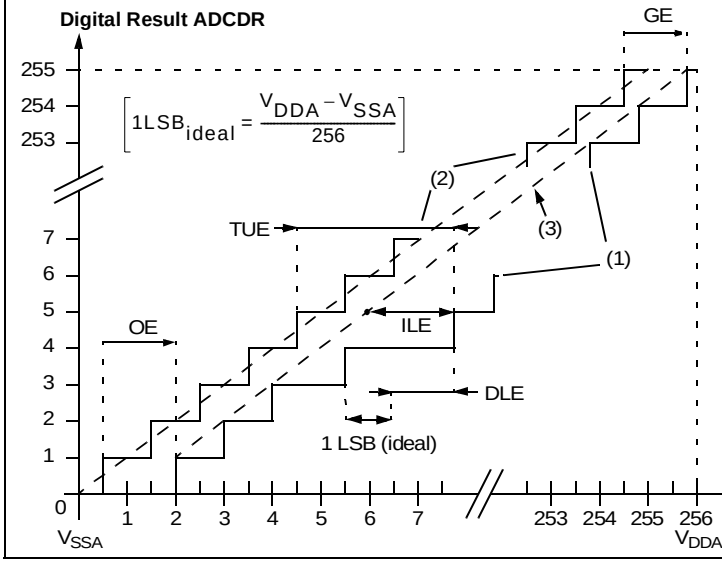


Notes:

1. Data based on standard I^2C protocol requirement, not tested in production.
2. The device must internally provide a hold time of at least 300ns for the SDA signal in order to bridge the undefined region of the falling edge of SCL.
3. The maximum hold time of the START condition has only to be met if the interface does not stretch the low period of SCL signal.
4. Measurement points are done at CMOS levels: 0.3xV_{DD} and 0.7xV_{DD}.

MEMORY AND PERIPHERAL CHARACTERISTICS - ADC

WARNING ON ADC: The ST72E/T589BW5 EPROM devices have their ADC which differs from the ROM devices in term of accuracy (less precision), timing (slower stabilization and conversion time) and external components. Care must be taken at software and hardware application levels when transferring a code from the EPROM device to the ROM device.



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

TUE=Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.
OE=Offset Error: deviation between the first actual transition and the first ideal one.
GE=Gain Error: deviation between the last ideal transition and the last actual one.
DLE=Differential Linearity Error: maximum deviation between actual steps and the ideal one.
ILE=Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

ADC Analog to Digital Converter (8-bit) for ST72E/T589BW5 devices

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
|TUE|	Total unadjusted error*	$f_{ADC}=f_{CPU}$=4MHz $V_{DD}=V_{DDA}$=5V			2	LSB
OE	Offset error*		-1		1	
GE	Gain Error*		-2		2	
|DLE|	Differential linearity error*				1	
|ILE|	Integral linearity error*				2	
V_{AIN}	Conversion range voltage		V_{SSA}		V_{DDA}	V
I_{ADC}	A/D conversion supply current	$f_{ADC}=f_{CPU}$=4MHz $V_{DD}=V_{DDA}$=5V		1		mA
t_{STAB}	Stabilization time after enable ADC				30	µs
t_{LOAD}	Sample capacitor loading time		8 32			µs $1/f_{ADC}$
t_{CONV}	Hold conversion timeval		8 32			µs $1/f_{ADC}$
R_{AIN}	External input resistor				20	KΩ
R_{ADC}	Internal input resistor				18	KΩ
C_{SAMPLE}	Sample capacitor				22	pF

MEMORY AND PERIPHERAL CHARACTERISTICS - ADC (Cont'd)

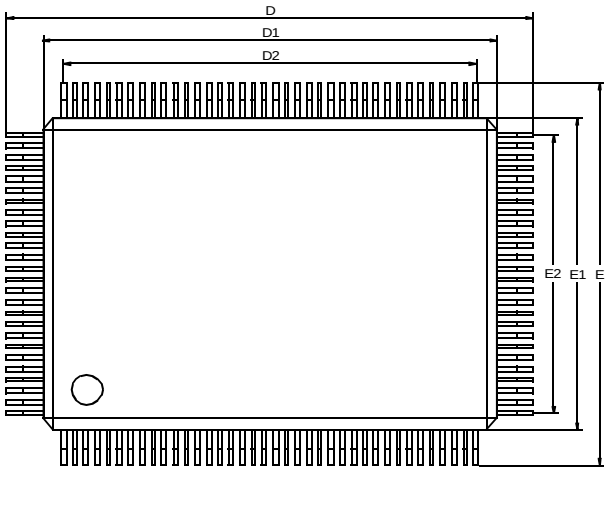
ADC Analog to Digital Converter (8-bit) for ROM devices						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{ADC}	Analog control frequency	$f_{ADC} = f_{CPU}/2$			4	MHz
|TUE|	Total unadjusted error*	f_{CPU}=8MHz, f_{ADC}=4MHz V_{DD}=V_{DDA}=5V			1.5	LSB
OE	Offset error*		-1		1	
GE	Gain Error*		-0.6		0.6	
|DLE|	Differential linearity error*				1	
|ILE|	Integral linearity error*				1.2	
V_{AIN}	Conversion range voltage		V_{SSA}		V_{DDA}	V
I_{ADC}	A/D conversion supply current	f_{CPU}=8MHz, f_{ADC}=4MHz V_{DD}=V_{DDA}=5V		1		mA
t_{STAB}	Stabilization time after enable ADC				1	µs
t_{LOAD}	Sample capacitor loading time			1 4		µs $1/f_{ADC}$
t_{CONV}	Hold conversion time			2 8		µs $1/f_{ADC}$
R_{AIN}	External input resistor				15	KΩ
R_{ADC}	Internal input resistor			1.5		KΩ
C_{SAMPLE}	Sample capacitor			6		pF

*__Note__: *ADC Accuracy vs. Negative Injection Current*:
For I_{inj-}=0.8mA, the typical leakage induced inside the die is 1.6µA and the effect on the ADC accuracy is a loss of 1 LSB by 10KΩ increase of the external analog source impedance.
These measurements results and recommendations are done in the worst condition of injection:
- negative injection
- injection to an Input with analog capability,adjacent to the enabled Analog Input
- at 5V V_{DD} supply, and worse temperature case.

10 PACKAGE CHARACTERISTICS

10.1 PACKAGE MECHANICAL DATA

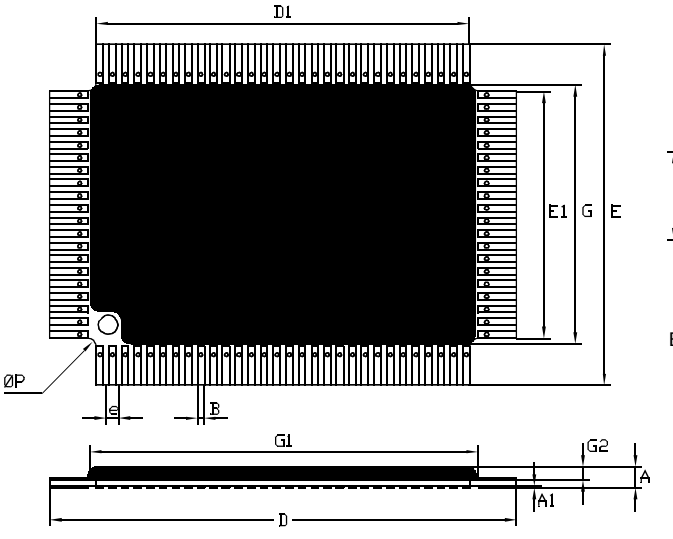
Figure 79. 128-Pin Plastic Quad Flat Package



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A		3.04	3.40		0.120	0.134
A1	0.25	0.33		0.010	0.013	
A2	2.57	2.71	2.87	0.101	0.107	0.113
B	0.13		0.28	0.005		0.011
C	0.13		0.23	0.005		0.009
D		23.20			0.913	
D1		20.00			0.787	
D3		18.50			0.728	
E		17.20			0.677	
E1		14.00			0.551	
E3		12.50			0.492	
e		0.50			0.020	
K	0°		7°	0°		7°
L	0.73	0.88	1.03	0.029	0.035	0.041
L1		1.60			0.063	
	Number of Pins					
N	128					

PACKAGE MECHANICAL DATA (Cont'd)

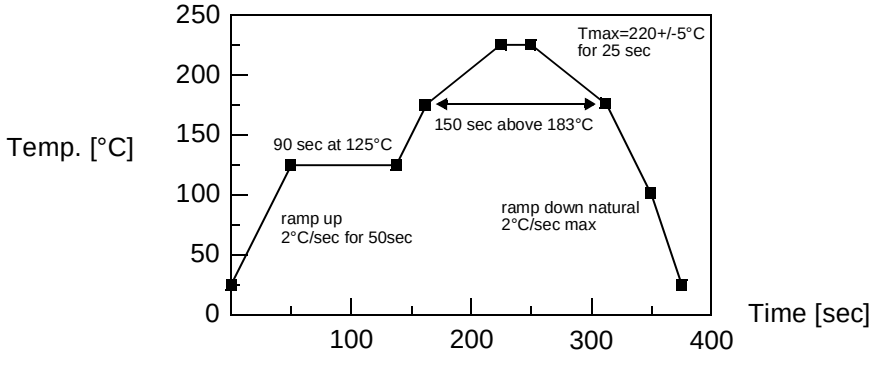
Figure 80. 128-Pin Economic Quad Flat Package



Dim	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			2.40			0.094
A1		0.03			0.001	
B	0.17	0.22	0.27	0.007	0.009	0.011
D	23.00	23.20	23.40	0.906	0.913	0.921
D1	18.57	18.72	18.97	0.731	0.737	0.747
E	17.00	17.20	17.40	0.669	0.677	0.685
E1	12.57	12.72	12.97	0.495	0.501	0.511
e		0.50			0.020	
G		13.60			0.535	
G1		19.60			0.772	
G2			1.80			0.071
L		1.40			0.055	
L1		1.60			0.063	
Øn			0.35			0.014
ØP			1.10			0.043
	Number of Pins					
N	128 (ND=38 / NE=26)					

Note: "QUALIFICATION OR VOLUME PRODUCTION OF DEVICES USING EPOXY PACKAGES (ESO/EDIL/EQFP) IS NOT AUTHORIZED It is expressly specified that qualification and/or volume production of devices using the package E.... in any applications is not authorized. Usage in any application is strictly restricted to development purpose.
Similar devices are available in plastic package mechanically compatible to the epoxy package for qualification and volume production."

Figure 81. Recommended Reflow Oven Profile (MID JEDEC)



11 DEVICE CONFIGURATION AND ORDERING INFORMATION

11.1 ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

Customer code is made up of the FASTROM or ROM contents and the list of the selected options (if any). The FASTROM or ROM contents are to be sent on diskette, or by electronic means, with the hexadecimal file in .S19 format generated by the development tool. All unused bytes must be set to FFh.

The selected options are communicated to STMicroelectronics using the correctly completed OPTION LIST appended. See page 155.

The STMicroelectronics Sales Organization will be pleased to provide detailed information on contractual points.

Table 32. Ordering Information

Sales Type [1)]	Program Memory (bytes)	RAM (bytes)	Package
ST72389BW4/xxx	16K ROM	512	PQFP128
ST72P589BW5/xxx	24K FASTROM	1024	
ST72T589BW5	24K OTP		

Note 1. /xxx stands for the ROM or FASTROM code name assigned by STMicroelectronics.

MICROCONTROLLER OPTION LIST

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM or FASTROM code*: . . . . . .
*The ROM or FASTROM code name is assigned by STMicroelectronics.
ROM or FASTROM code must be sent in .S19 format. .Hex extension cannot be processed.

STMicroelectronics references:
Device: [ ] ST72P589B (24 KB)
[ ] ST72389B (16 KB)

Temperature Range:
[ ] 0° C to +70° C
[ ] -40° C to +85° C

Package: [ ] PQFP128: [ ] Tape & Reel
[ ] Tray

Marking: [ ] Standard marking:
[ ] Special marking (ROM only):
PQFP128 (14 char. max) : _ _ _ _ _ _ _ _ _ _ _ _ _ _

Authorized characters are letters, digits, '.', '-', '/' and spaces only.
Please consult your local STMicroelectronics sales office for other marking details if required.

Comments:

Notes: . . . . . .

Date: . . . . . .

Signature: . . . . . .

11.2 ST7 APPLICATION NOTES

IDENTIFICATION	DESCRIPTION
EXAMPLE DRIVERS	
AN 969	SCI COMMUNICATION BETWEEN ST7 AND PC
AN 970	SPI COMMUNICATION BETWEEN ST7 AND EEPROM
AN 971	I²C COMMUNICATING BETWEEN ST7 AND M24CXX EEPROM
AN 972	ST7 SOFTWARE SPI MASTER COMMUNICATION
AN 973	SCI SOFTWARE COMMUNICATION WITH A PC USING ST72251 16-BIT TIMER
AN 974	REAL TIME CLOCK WITH ST7 TIMER OUTPUT COMPARE
AN 976	DRIVING A BUZZER THROUGH ST7 TIMER PWM FUNCTION
AN 979	DRIVING AN ANALOG KEYBOARD WITH THE ST7 ADC
AN 980	ST7 KEYPAD DECODING TECHNIQUES, IMPLEMENTING WAKE-UP ON KEYSTROKE
AN1017	USING THE ST7 UNIVERSAL SERIAL BUS MICROCONTROLLER
AN1041	USING ST7 PWM SIGNAL TO GENERATE ANALOG OUTPUT (SINUSOID)
AN1042	ST7 ROUTINE FOR I²C SLAVE MODE MANAGEMENT
AN1044	MULTIPLE INTERRUPT SOURCES MANAGEMENT FOR ST7 MCUS
AN1045	ST7 S/W IMPLEMENTATION OF I²C BUS MASTER
AN1046	UART EMULATION SOFTWARE
AN1047	MANAGING RECEPTION ERRORS WITH THE ST7 SCI PERIPHERALS
AN1048	ST7 SOFTWARE LCD DRIVER
AN1078	PWM DUTY CYCLE SWITCH IMPLEMENTING TRUE 0% & 100% DUTY CYCLE
AN1082	DESCRIPTION OF THE ST72141 MOTOR CONTROL PERIPHERAL REGISTERS
AN1083	ST72141 BLDC MOTOR CONTROL SOFTWARE AND FLOWCHART EXAMPLE
AN1105	ST7 PCAN PERIPHERAL DRIVER
AN1129	PERMANENT MAGNET DC MOTOR DRIVE.
AN1130	AN INTRODUCTION TO SENSORLESS BRUSHLESS DC MOTOR DRIVE APPLICATIONS WITH THE ST72141
AN1148	USING THE ST7263 FOR DESIGNING A USB MOUSE
AN1149	HANDLING SUSPEND MODE ON A USB MOUSE
AN1180	USING THE ST7263 KIT TO IMPLEMENT A USB GAME PAD
AN1276	BLDC MOTOR START ROUTINE FOR THE ST72141 MICROCONTROLLER
AN1321	USING THE ST72141 MOTOR CONTROL MCU IN SENSOR MODE
AN1325	USING THE ST7 USB LOW-SPEED FIRMWARE V4.X
AN1445	USING THE ST7 SPI TO EMULATE A 16-BIT SLAVE
AN1475	DEVELOPING AN ST7265X MASS STORAGE APPLICATION
AN1504	STARTING A PWM SIGNAL DIRECTLY AT HIGH LEVEL USING THE ST7 16-BIT TIMER
PRODUCT EVALUATION	
AN 910	PERFORMANCE BENCHMARKING
AN 990	ST7 BENEFITS VERSUS INDUSTRY STANDARD
AN1077	OVERVIEW OF ENHANCED CAN CONTROLLERS FOR ST7 AND ST9 MCUS
AN1086	U435 CAN-DO SOLUTIONS FOR CAR MULTIPLEXING
AN1150	BENCHMARK ST72 VS PC16
AN1151	PERFORMANCE COMPARISON BETWEEN ST72254 & PC16F876
AN1278	LIN (LOCAL INTERCONNECT NETWORK) SOLUTIONS
PRODUCT MIGRATION	
AN1131	MIGRATING APPLICATIONS FROM ST72511/311/214/124 TO ST72521/321/324
AN1322	MIGRATING AN APPLICATION FROM ST7263 REV.B TO ST7263B
AN1365	GUIDELINES FOR MIGRATING ST72C254 APPLICATION TO ST72F264
PRODUCT OPTIMIZATION	

IDENTIFICATION	DESCRIPTION
AN 982	USING ST7 WITH CERAMIC RESONATOR
AN1014	HOW TO MINIMIZE THE ST7 POWER CONSUMPTION
AN1015	SOFTWARE TECHNIQUES FOR IMPROVING MICROCONTROLLER EMC PERFORMANCE
AN1040	MONITORING THE VBUS SIGNAL FOR USB SELF-POWERED DEVICES
AN1070	ST7 CHECKSUM SELF-CHECKING CAPABILITY
AN1324	CALIBRATING THE RC OSCILLATOR OF THE ST7FLITE0 MCU USING THE MAINS
AN1477	EMULATED DATA EEPROM WITH XFLASH MEMORY
AN1502	EMULATED DATA EEPROM WITH ST7 HDFLASH MEMORY
AN1529	EXTENDING THE CURRENT & VOLTAGE CAPABILITY ON THE ST7265 VDDF SUPPLY
AN1530	ACCURATE TIMEBASE FOR LOW-COST ST7 APPLICATIONS WITH INTERNAL RC OSCILLATOR
PROGRAMMING AND TOOLS	
AN 978	KEY FEATURES OF THE STVD7 ST7 VISUAL DEBUG PACKAGE
AN 983	KEY FEATURES OF THE COSMIC ST7 C-COMPILER PACKAGE
AN 985	EXECUTING CODE IN ST7 RAM
AN 986	USING THE INDIRECT ADDRESSING MODE WITH ST7
AN 987	ST7 SERIAL TEST CONTROLLER PROGRAMMING
AN 988	STARTING WITH ST7 ASSEMBLY TOOL CHAIN
AN 989	GETTING STARTED WITH THE ST7 HIWARE C TOOLCHAIN
AN1039	ST7 MATH UTILITY ROUTINES
AN1064	WRITING OPTIMIZED HIWARE C LANGUAGE FOR ST7
AN1071	HALF DUPLEX USB-TO-SERIAL BRIDGE USING THE ST72611 USB MICROCONTROLLER
AN1106	TRANSLATING ASSEMBLY CODE FROM HC05 TO ST7
AN1179	PROGRAMMING ST7 FLASH MICROCONTROLLERS IN REMOTE ISP MODE (IN-SITU PROGRAMMING)
AN1446	USING THE ST72521 EMULATOR TO DEBUG A ST72324 TARGET APPLICATION
AN1478	PORTING AN ST7 PANTA PROJECT TO CODEWARRIOR IDE
AN1527	DEVELOPING A USB SMARTCARD READER WITH ST7SCR
AN1575	ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUS

12 SUMMARY OF CHANGES

Description of the changes between the current release of the specification and the previous one.

Rev.	Main changes	Date
2.7	Changed output configuration for PD4 and PD5 in Table 1, “Device Pin Description,” on page 6 Removed references to 32-kHz auxiliary oscillator Added notes: “I2C, PWM-BRM and CAN available on ST72589 version only” Added V_{HYS} in section 9.5 on page 142	June 03