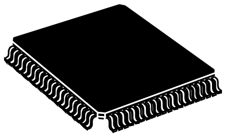
ST72321M6
ST72321M9

80-pin 8-bit MCU with 32 to 60 Kbytes Flash, ADC, five timers, SPI, SCI, I^2C interface

Features

- **Memories**
  - 32 Kbytes to 60 Kbytes dual voltage High Density Flash (HDFlash) with read-out protection capability. In-application programming and In-circuit programming.
  - 1 Kbyte to 2 Kbytes RAM
  - HDFlash endurance: 100 cycles at 85 °C; data retention: 40 years at 85 °C
- **Clock, reset and supply management**
  - Enhanced low voltage supervisor (LVD) for main supply and auxiliary voltage detector (AVD) with interrupt capability
  - Clock sources: crystal/ceramic resonator oscillators, internal RC oscillator and bypass for external clock
  - PLL for 2x frequency multiplication
  - Four power saving modes: Halt, Active-Halt, Wait and Slow
- **Interrupt management**
  - Nested interrupt controller
  - 14 interrupt vectors plus TRAP and RESET
  - Top Level Interrupt (TLI) pin
  - 15 external interrupt lines (on 4 vectors)
- **Up to 64 I/O ports**
  - 64 multifunctional bidirectional I/O lines
  - 34 alternate function lines
  - 16 high sink outputs
- **5 timers**
  - Main clock controller with: Real-time base, Beep and Clock-out capabilities
  - Configurable watchdog timer
  - Two 16-bit timers with: 2 input captures, 2 output compares, external clock input on one timer, PWM and pulse generator modes
  - 8-bit PWM Auto-Reload timer with: 2 input captures, 4 PWM outputs, output compare and time base interrupt, external clock with event detector



LQFP80
14 x 14

- **4 Communication interfaces**
  - SPI synchronous serial interface
  - SCI asynchronous serial interface
  - I^2C multimaster interface (SMbus V1.1 compliant)
- **Analog periperal (low current coupling)**
  - 10-bit ADC with 16 input robust input ports
- **Instruction set**
  - 8-bit data manipulation
  - 63 basic Instructions
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instruction
- **Development tools**
  - Full hardware/software development package
  - In-circuit testing capability

Table 1. Device summary

Features	ST72321M9	ST72321M6
Flash program memory	60 Kbytes	32 Kbytes
RAM (stack) - bytes	2048 bytes (256 bytes)	1024 (256 bytes)
Operating voltage	3.8 V to 5.5 V	
Temperature range	-40 °C to +85 °C	
Package	LQFP80 14x14	

Table of Contents

1 INTRODUCTION 7
2 PIN DESCRIPTION 8
3 REGISTER & MEMORY MAP 13
4 FLASH PROGRAM MEMORY 17
- 4.1 INTRODUCTION 17
- 4.2 MAIN FEATURES 17
- 4.3 STRUCTURE 17
  - 4.3.1 Read-out Protection 17
- 4.4 ICC INTERFACE 18
- 4.5 ICP (IN-CIRCUIT PROGRAMMING) 19
- 4.6 IAP (IN-APPLICATION PROGRAMMING) 19
- 4.7 RELATED DOCUMENTATION 19
  - 4.7.1 Register Description 19

5 CENTRAL PROCESSING UNIT 20
- 5.1 INTRODUCTION 20
- 5.2 MAIN FEATURES 20
- 5.3 CPU REGISTERS 20

6 SUPPLY, RESET AND CLOCK MANAGEMENT 23
- 6.1 PHASE LOCKED LOOP 23
- 6.2 MULTI-OSCILLATOR (MO) 24
- 6.3 RESET SEQUENCE MANAGER (RSM) 25
  - 6.3.1 Introduction 25
  - 6.3.2 Asynchronous External RESET pin 25
  - 6.3.3 External Power-On RESET 26
  - 6.3.4 Internal Low Voltage Detector (LVD) RESET 26
  - 6.3.5 Internal Watchdog RESET 26
- 6.4 SYSTEM INTEGRITY MANAGEMENT (SI) 27
  - 6.4.1 Low Voltage Detector (LVD) 27
  - 6.4.2 Auxiliary Voltage Detector (AVD) 28
  - 6.4.3 Low Power Modes 30
  - 6.4.4 Register Description 31

7 INTERRUPTS 32
- 7.1 INTRODUCTION 32
- 7.2 MASKING AND PROCESSING FLOW 32
- 7.3 INTERRUPTS AND LOW POWER MODES 34
- 7.4 CONCURRENT & NESTED MANAGEMENT 34
- 7.5 INTERRUPT REGISTER DESCRIPTION 35
- 7.6 EXTERNAL INTERRUPTS 37
  - 7.6.1 I/O Port Interrupt Sensitivity 37
- 7.7 EXTERNAL INTERRUPT CONTROL REGISTER (EICR) 39

8 POWER SAVING MODES 41
- 8.1 INTRODUCTION 41
- 8.2 SLOW MODE 41
- 8.3 WAIT MODE 42

Table of Contents

8.4 ACTIVE-HALT AND HALT MODES 43
    8.4.1 ACTIVE-HALT MODE 43
    8.4.2 HALT MODE 44
    8.4.3 I/O Port Implementation 46

9 ON-CHIP PERIPHERALS 48

9.1 WATCHDOG TIMER (WDG) 48
    9.1.1 Introduction 48
    9.1.2 Main Features 48
    9.1.3 Functional Description 48
    9.1.4 How to Program the Watchdog Timeout 49
    9.1.5 Low Power Modes 51
    9.1.6 Hardware Watchdog Option 51
    9.1.7 Using Halt Mode with the WDG (WDGHALT option) 51
    9.1.8 Interrupts 51
    9.1.9 Register Description 51
9.2 MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK AND BEEPER (MCC/RTC) . . 53
    9.2.1 Programmable CPU Clock Prescaler 53
    9.2.2 Clock-out Capability 53
    9.2.3 Real-Time Clock Timer (RTC) 53
    9.2.4 Beeper 53
    9.2.5 Low Power Modes 54
    9.2.6 Interrupts 54
    9.2.7 Register Description 54
9.3 PWM AUTO-RELOAD TIMER (ART) 56
    9.3.1 Introduction 56
    9.3.2 Functional Description 57
    9.3.3 Register Description 61
9.4 16-BIT TIMER 65
    9.4.1 Introduction 65
    9.4.2 Main Features 65
    9.4.3 Functional Description 65
    9.4.4 Low Power Modes 77
    9.4.5 Interrupts 77
    9.4.6 Summary of Timer Modes 77
    9.4.7 Register Description 78
9.5 SERIAL PERIPHERAL INTERFACE (SPI) 84
    9.5.1 Introduction 84
    9.5.2 Main Features 84
    9.5.3 General Description 84
    9.5.4 Clock Phase and Clock Polarity 88
    9.5.5 Error Flags 89
    9.5.6 Low Power Modes 91
    9.5.7 Interrupts 91
    9.5.8 Register Description 92
9.6 SERIAL COMMUNICATIONS INTERFACE (SCI) 95
    9.6.1 Introduction 95
    9.6.2 Main Features 95
    9.6.3 General Description 95

9.6.4 Functional Description . . . 97
9.6.5 Low Power Modes . . . 104
9.6.6 Interrupts . . . 104
9.6.7 Register Description . . . 105
9.7 I2C BUS INTERFACE (I2C) . . . 111
9.7.1 Introduction . . . 111
9.7.2 Main Features . . . 111
9.7.3 General Description . . . 111
9.7.4 Functional Description . . . 113
9.7.5 Low Power Modes . . . 117
9.7.6 Interrupts . . . 117
9.7.7 Register Description . . . 118
9.8 10-BIT A/D CONVERTER (ADC) . . . 124
9.8.1 Introduction . . . 124
9.8.2 Main Features . . . 124
9.8.3 Functional Description . . . 125
9.8.4 Low Power Modes . . . 125
9.8.5 Interrupts . . . 125
9.8.6 Register Description . . . 126
10 INSTRUCTION SET . . . 128
10.1 CPU ADDRESSING MODES . . . 128
10.1.1 Inherent . . . 129
10.1.2 Immediate . . . 129
10.1.3 Direct . . . 129
10.1.4 Indexed (No Offset, Short, Long) . . . 129
10.1.5 Indirect (Short, Long) . . . 129
10.1.6 Indirect Indexed (Short, Long) . . . 130
10.1.7 Relative mode (Direct, Indirect) . . . 130
10.2 INSTRUCTION GROUPS . . . 131
11 ELECTRICAL CHARACTERISTICS . . . 134
11.1 PARAMETER CONDITIONS . . . 134
11.1.1 Minimum and Maximum values . . . 134
11.1.2 Typical values . . . 134
11.1.3 Typical curves . . . 134
11.1.4 Loading capacitor . . . 134
11.1.5 Pin input voltage . . . 134
11.2 ABSOLUTE MAXIMUM RATINGS . . . 135
11.2.1 Voltage Characteristics . . . 135
11.2.2 Current Characteristics . . . 135
11.2.3 Thermal Characteristics . . . 136
11.3 OPERATING CONDITIONS . . . 136
11.3.1 General Operating Conditions . . . 136
11.3.2 Operating Conditions with Low Voltage Detector (LVD) . . . 137
11.3.3 Auxiliary Voltage Detector (AVD) Thresholds . . . 137
11.3.4 External Voltage Detector (EVD) Thresholds . . . 137
11.4 SUPPLY CURRENT CHARACTERISTICS . . . 138
11.4.1 CURRENT CONSUMPTION . . . 138

11.4.2 Supply and Clock Managers 140
11.4.3 On-Chip Peripherals 141
11.5 CLOCK AND TIMING CHARACTERISTICS 142
11.5.1 General Timings 142
11.5.2 External Clock Source 142
11.5.3 Crystal and Ceramic Resonator Oscillators 143
11.5.4 RC Oscillators 145
11.5.5 PLL Characteristics 146
11.6 MEMORY CHARACTERISTICS 147
11.6.1 RAM and Hardware Registers 147
11.6.2 Flash memory 147
11.7 EMC CHARACTERISTICS 148
11.7.1 Functional EMS (Electro Magnetic Susceptibility) 148
11.7.2 Electro Magnetic Interference (EMI) 149
11.7.3 Absolute Maximum Ratings (Electrical Sensitivity) 150
11.8 I/O PORT PIN CHARACTERISTICS 151
11.8.1 General Characteristics 151
11.8.2 Output Driving Current 152
11.9 CONTROL PIN CHARACTERISTICS 154
11.9.1 Asynchronous RESET Pin 154
11.9.2 ICCSEL/VPP Pin 156
11.10TIMER PERIPHERAL CHARACTERISTICS 157
11.10.1 8-Bit PWM-ART Auto-Reload Timer 157
11.10.2 16-Bit Timer 157
11.11COMMUNICATION INTERFACE CHARACTERISTICS 158
11.11.1 SPI - Serial Peripheral Interface 158
11.11.2 I2C - Inter IC Control Interface 160
11.1210-BIT ADC CHARACTERISTICS 162
11.12.1 Analog Power Supply and Reference Pins 164
11.12.2 General PCB Design Guidelines 164
11.12.3 ADC Accuracy 165
12 PACKAGE CHARACTERISTICS 166
12.1 ECOPACK 166
12.2 PACKAGE MECHANICAL DATA 166
12.3 THERMAL CHARACTERISTICS 166
13 ST72321Mx DEVICE CONFIGURATION AND ORDERING INFORMATION 167
13.1 FLASH OPTION BYTES 167
13.2 DEVICE ORDERING INFORMATION 169
14 KNOWN LIMITATIONS 170
14.1 SAFE CONNECTION OF OSC1/OSC2 PINS 170
14.2 RESET PIN PROTECTION WITH LVD ENABLED 170
14.3 UNEXPECTED RESET FETCH 170
14.4 EXTERNAL INTERRUPT MISSED 170
14.5 CLEARING ACTIVE INTERRUPTS OUTSIDE INTERRUPT ROUTINE 171
14.6 SCI WRONG BREAK DURATION 172

14.7 16-BIT TIMER PWM MODE 172
14.8 TIMD SET SIMULTANEOUSLY WITH OC INTERRUPT 172
14.9 I2C MULTIMASTER 172
14.10INTERNAL RC OSCILLATOR WITH LVD 172
14.11I/O BEHAVIOUR DURING ICC MODE ENTRY SEQUENCE 172
14.12READ-OUT PROTECTION WITH LVD 173
15 REVISION HISTORY 174

1 INTRODUCTION

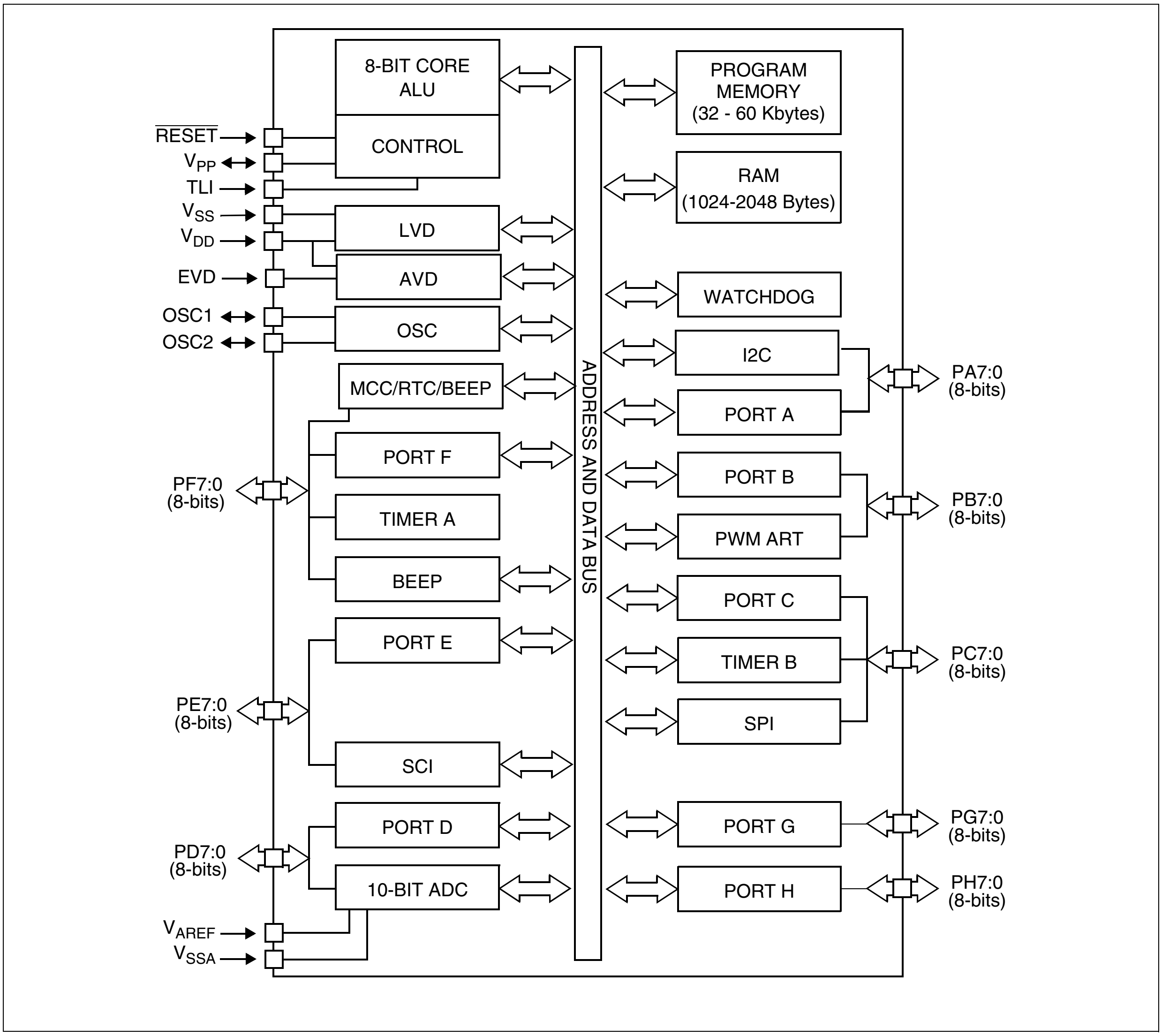
The ST72321Mx Flash devices are members of the ST7 microcontroller family designed for mid-range applications.

All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set and are available with Flash memory.

Under software control, all devices can be placed in Wait, Slow, Active-halt or Halt mode, reducing power consumption when the application is in idle or stand-by state.

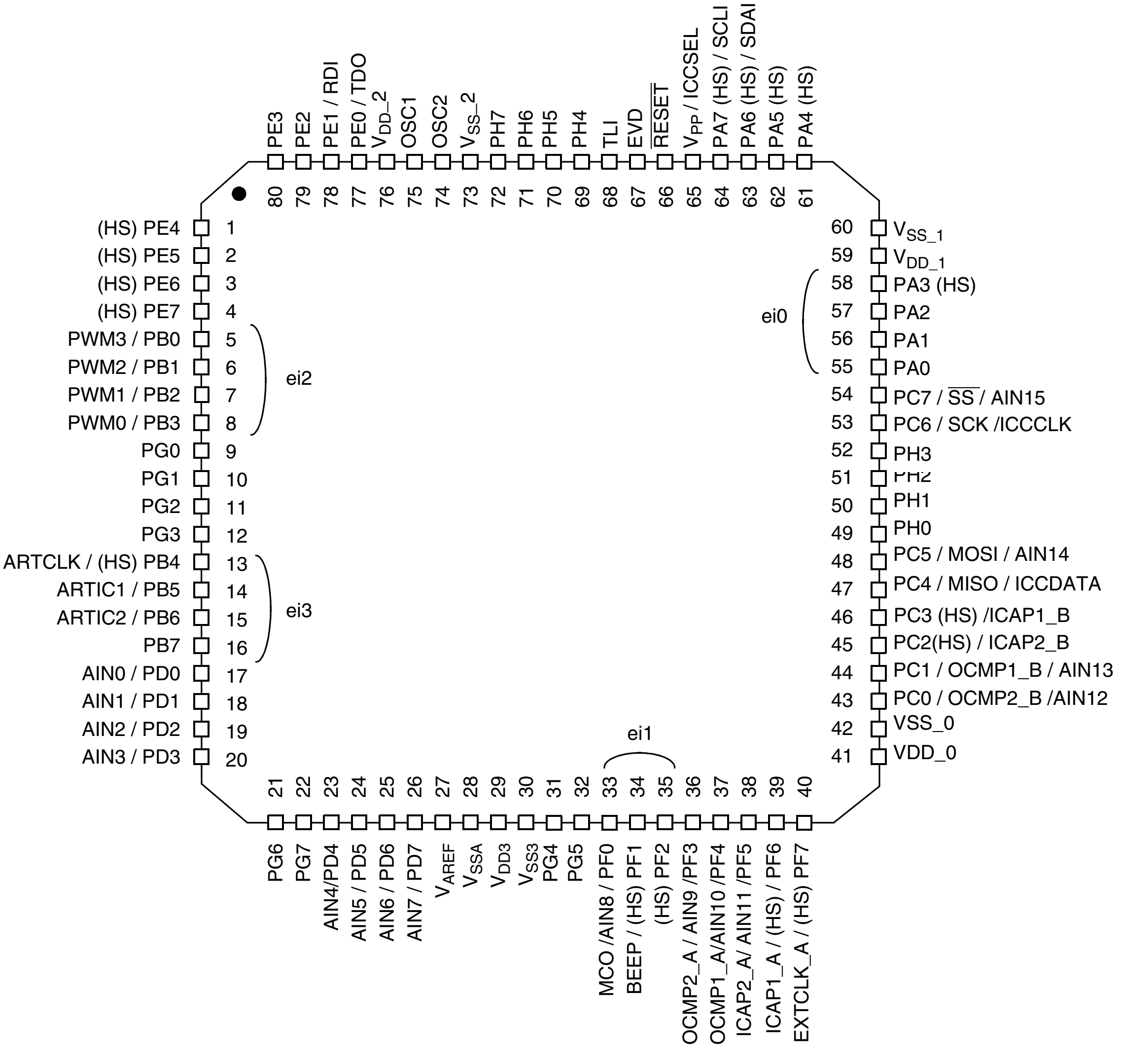
The enhanced instruction set and addressing modes of the ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned multiplication and indirect addressing modes.

Figure 1. Device Block Diagram


8-BIT CORE ALU
CONTROL
RESET
VPP
TLI
VSS
VDD
EVD
OSC1
OSC2
LVD
AVD
OSC
MCC/RTC/BEEP
PORT F
TIMER A
BEEP
PORT E
SCI
PORT D
10-BIT ADC
PF7:0 (8-bits)
PE7:0 (8-bits)
PD7:0 (8-bits)
VAREF
VSSA
ADDRESS AND DATA BUS
PROGRAM MEMORY (32 - 60 Kbytes)
RAM (1024-2048 Bytes)
WATCHDOG
I2C
PORT A
PORT B
PWM ART
PORT C
TIMER B
SPI
PORT G
PORT H
PA7:0 (8-bits)
PB7:0 (8-bits)
PC7:0 (8-bits)
PG7:0 (8-bits)
PH7:0 (8-bits)

2 PIN DESCRIPTION

Figure 2. 80-Pin LQFP 14x14 Package Pinout


PE3 80
PE2 79
PE1 / RDI 78
PE0 / TDO 77
V_{DD_2} 76
OSC1 75
OSC2 74
V_{SS_2} 73
PH7 72
PH6 71
PH5 70
PH4 69
TLI 68
EVD 67
$\overline{RESET}$ 66
V_{PP} / ICCSEL 65
PA7 (HS) / SCLI 64
PA6 (HS) / SDAI 63
PA5 (HS) 62
PA4 (HS) 61
(HS) PE4 1
(HS) PE5 2
(HS) PE6 3
(HS) PE7 4
PWM3 / PB0 5
PWM2 / PB1 6
PWM1 / PB2 7
PWM0 / PB3 8
ei2
PG0 9
PG1 10
PG2 11
PG3 12
ARTCLK / (HS) PB4 13
ARTIC1 / PB5 14
ARTIC2 / PB6 15
PB7 16
ei3
AIN0 / PD0 17
AIN1 / PD1 18
AIN2 / PD2 19
AIN3 / PD3 20
60 V_{SS_1}
59 V_{DD_1}
ei0
58 PA3 (HS)
57 PA2
56 PA1
55 PA0
54 PC7 / $\overline{SS}$ / AIN15
53 PC6 / SCK /ICCCLK
52 PH3
51 PH2
50 PH1
49 PH0
48 PC5 / MOSI / AIN14
47 PC4 / MISO / ICCDATA
46 PC3 (HS) /ICAP1_B
45 PC2(HS) / ICAP2_B
44 PC1 / OCMP1_B / AIN13
43 PC0 / OCMP2_B /AIN12
42 VSS_0
41 VDD_0
ei1
21 PG6
22 PG7
23 AIN4/PD4
24 AIN5 / PD5
25 AIN6 / PD6
26 AIN7 / PD7
27 V_{AREF}
28 V_{SSA}
29 V_{DD3}
30 V_{SS3}
31 PG4
32 PG5
33 MCO /AIN8 / PF0
34 BEEP / (HS) PF1
35 (HS) PF2
36 OCMP2_A / AIN9 /PF3
37 OCMP1_A/AIN10 /PF4
38 ICAP2_A/ AIN11 /PF5
39 ICAP1_A / (HS) / PF6
40 EXTCLK_A / (HS) PF7

(HS) 20 mA high sink capability
eix associated external interrupt vector

PIN DESCRIPTION (Cont'd)

Legend / Abbreviations for Table 2 :

Type: I = input, O = output, S = supply

Input level: A = Dedicated analog input

In/Output level: C = CMOS $0.3V_{DD}/0.7V_{DD}$
C_T= CMOS $0.3V_{DD}/0.7V_{DD}$ with input trigger
T_T= TTL 0.8 V / 2 V with Schmitt trigger

Output level: HS = 20 mA high sink (on N-buffer only)

Port and control configuration:

- Input: float = floating, wpu = weak pull-up, int = interrupt 1), ana = analog
- Output: OD = open drain 2), PP = push-pull

The RESET configuration of each pin is shown in bold. This configuration is valid as long as the device is in reset state.

Table 2. Device Pin Description

Pin n°	Pin Name	Type	Level		Port						Main function (after reset)	Alternate function
LQFP80			Input	Output	Input				Output			
					float	wpu	int	ana	OD	PP		
1	PE4 (HS)	I/O	C_T	HS	**X**	X			X	X	Port E4	
2	PE5 (HS)	I/O	C_T	HS	**X**	X			X	X	Port E5	
3	PE6 (HS)	I/O	C_T	HS	**X**	X			X	X	Port E6	
4	PE7 (HS)	I/O	C_T	HS	**X**	X			X	X	Port E7	
5	PB0/PWM3	I/O	C_T		**X**	ei2			X	X	Port B0	PWM Output 3
6	PB1/PWM2	I/O	C_T		**X**	ei2			X	X	Port B1	PWM Output 2
7	PB2/PWM1	I/O	C_T		**X**	ei2			X	X	Port B2	PWM Output 1
8	PB3/PWM0	I/O	C_T		**X**		ei2		X	X	Port B3	PWM Output 0
9	PG0	I/O	T_T		**X**	X			X	X	Port G0	
10	PG1	I/O	T_T		**X**	X			X	X	Port G1	
11	PG2	I/O	T_T		**X**	X			X	X	Port G2	
12	PG3	I/O	T_T		**X**	X			X	X	Port G3	
13	PB4 (HS)/ARTCLK	I/O	C_T	HS	**X**	ei3			X	X	Port B4	PWM-ART External Clock
14	PB5/ARTIC1	I/O	C_T		**X**	ei3			X	X	Port B5	PWM-ART Input Capture 1
15	PB6/ARTIC2	I/O	C_T		**X**	ei3			X	X	Port B6	PWM-ART Input Capture 2
16	PB7	I/O	C_T		**X**		ei3		X	X	Port B7	
17	PD0 /AIN0	I/O	C_T		**X**	X		X	X	X	Port D0	ADC Analog Input 0
18	PD1/AIN1	I/O	C_T		**X**	X		X	X	X	Port D1	ADC Analog Input 1
19	PD2/AIN2	I/O	C_T		**X**	X		X	X	X	Port D2	ADC Analog Input 2
20	PD3/AIN3	I/O	C_T		**X**	X		X	X	X	Port D3	ADC Analog Input 3
21	PG6	I/O	T_T		**X**	X			X	X	Port G6	
22	PG7	I/O	T_T		**X**	X			X	X	Port G7	
23	PD4/AIN4	I/O	C_T		**X**	X		X	X	X	Port D4	ADC Analog Input 4
24	PD5/AIN5	I/O	C_T		**X**	X		X	X	X	Port D5	ADC Analog Input 5
25	PD6/AIN6	I/O	C_T		**X**	X		X	X	X	Port D6	ADC Analog Input 6

Pin n°	Pin Name	Type	Level		Port						Main function (after reset)	Alternate function	
LQFP80			Input	Output	Input				Output				
					float	wpu	int	ana	OD	PP			
26	PD7/AIN7	I/O	C_T		**X**	X		X	X	X	Port D7	ADC Analog Input 7	
27	V_{AREF} 2)	I									Analog Reference Voltage for ADC		
28	V_{SSA} 2)	S									Analog Ground Voltage		
29	V_{DD_3} 2)	S									Digital Main Supply Voltage		
30	V_{SS_3} 2)	S									Digital Ground Voltage		
31	PG4	I/O	T_T		**X**	X			X	X	Port G4		
32	PG5	I/O	T_T		**X**	X			X	X	Port G5		
33	PF0/MCO/AIN8	I/O	C_T		**X**	ei1		X	X	X	Port F0	Main clock out (f_{CPU})	ADC Analog Input 8
34	PF1 (HS)/BEEP	I/O	C_T	HS	**X**	ei1			X	X	Port F1	Beep signal output	
35	PF2 (HS)	I/O	C_T	HS	**X**		ei1		X	X	Port F2		
36	PF3/OCMP2_A/AIN9	I/O	C_T		**X**	X		X	X	X	Port F3	Timer A Output Compare 2	ADC Analog Input 9
37	PF4/OCMP1_A/AIN10	I/O	C_T		**X**	X		X	X	X	Port F4	Timer A Output Compare 1	ADC Analog Input 10
38	PF5/ICAP2_A/AIN11	I/O	C_T		**X**	X		X	X	X	Port F5	Timer A Input Capture 2	ADC Analog Input 11
39	PF6 (HS)/ICAP1_A	I/O	C_T	HS	**X**	X			X	X	Port F6	Timer A Input Capture 1	
40	PF7 (HS)/EXTCLK_A	I/O	C_T	HS	**X**	X			X	X	Port F7	Timer A External Clock Source	
41	V_{DD_0} 2)	S									Digital Main Supply Voltage		
42	V_{SS_0} 2)	S									Digital Ground Voltage		
43	PC0/OCMP2_B/AIN12	I/O	C_T		**X**	X		X	X	X	Port C0	Timer B Output Compare 2	ADC Analog Input 12
44	PC1/OCMP1_B/AIN13	I/O	C_T		**X**	X		X	X	X	Port C1	Timer B Output Compare 1	ADC Analog Input 13
45	PC2 (HS)/ICAP2_B	I/O	C_T	HS	**X**	X			X	X	Port C2	Timer B Input Capture 2	
46	PC3 (HS)/ICAP1_B	I/O	C_T	HS	**X**	X			X	X	Port C3	Timer B Input Capture 1	
47	PC4/MISO/ICCDATA	I/O	C_T		**X**	X			X	X	Port C4	SPI Master In / Slave Out Data	ICC Data Input
48	PC5/MOSI/AIN14	I/O	C_T		**X**	X		X	X	X	Port C5	SPI Master Out / Slave In Data	ADC Analog Input 14
49	PH0	I/O	T_T		**X**	X			X	X	Port H0		
50	PH1	I/O	T_T		**X**	X			X	X	Port H1		
51	PH2	I/O	T_T		**X**	X			X	X	Port H2		
52	PH3	I/O	T_T		**X**	X			X	X	Port H3		

Pin n°	Pin Name	Type	Level		Port						Main function (after reset)	Alternate function	
LQFP80			Input	Output	Input				Output				
					float	wpu	int	ana	OD	PP			
53	PC6/SCK/ICCCLK	I/O	C_T		**X**	X			X	X	Port C6	SPI Serial Clock **Caution:** Negative current injection not allowed on this pin	ICC Clock Output
54	PC7/$\overline{SS}$/AIN15	I/O	C_T		**X**	X		X	X	X	Port C7	SPI Slave Select (active low)	ADC Analog Input 15
55	PA0	I/O	C_T		**X**	ei0			X	X	Port A0		
56	PA1	I/O	C_T		**X**	ei0			X	X	Port A1		
57	PA2	I/O	C_T		**X**	ei0			X	X	Port A2		
58	PA3 (HS)	I/O	C_T	HS	**X**		ei0		X	X	Port A3		
59	V_{DD_1} 2)	S									Digital Main Supply Voltage		
60	V_{SS_1} 2)	S									Digital Ground Voltage		
61	PA4 (HS)	I/O	C_T	HS	**X**	X			X	X	Port A4		
62	PA5 (HS)	I/O	C_T	HS	**X**	X			X	X	Port A5		
63	PA6 (HS)/SDAI	I/O	C_T	HS	**X**				T		Port A6	I^2C Data 5)	
64	PA7 (HS)/SCLI	I/O	C_T	HS	**X**				T		Port A7	I^2C Clock 5)	
65	V_{PP}/ ICCSEL	I									Must be tied low. In Flash programming mode, this pin acts as the programming voltage input V_{PP}.		
66	$\overline{RESET}$	I/O	C_T								Top priority non maskable interrupt.		
67	EVD										External voltage detector		
68	TLI	I	C_T		**X**		X				Top level interrupt input pin		
69	PH4	I/O	T_T		**X**	X			X	X	Port H4		
70	PH5	I/O	T_T		**X**	X			X	X	Port H5		
71	PH6	I/O	T_T		**X**	X			X	X	Port H6		
72	PH7	I/O	T_T		**X**	X			X	X	Port H7		
73	V_{SS_2} 2)	S									Digital Ground Voltage		
74	OSC2 3)	I/O									Resonator oscillator inverter output		
75	OSC1 3)	I									External clock input or Resonator oscillator inverter input		
76	V_{DD_2} 2)	S									Digital Main Supply Voltage		
77	PE0/TDO	I/O	C_T		**X**	X			X	X	Port E0	SCI Transmit Data Out	
78	PE1/RDI	I/O	C_T		**X**	X			X	X	Port E1	SCI Receive Data In	
79	PE2	I/O	C_T			**X**					Port E2		
80	PE3	I/O	C_T		**X**	X			X	X	Port E3		

Notes:

1. In the interrupt input column, “eiX” defines the associated external interrupt vector. If the weak pull-up column (wpu) is merged with the interrupt column (int), then the I/O configuration is pull-up interrupt input, else the configuration is floating interrupt input.

2. It is mandatory to connect all available V_{DD} and V_{REF} pins to the supply voltage and all V_{SS} and V_{SSA}

pins to ground.

3. OSC1 and OSC2 pins connect a crystal/ceramic resonator, or an external source to the on-chip oscillator.

4. On the chip, each I/O port may have up to 8 pads. Pads that are not bonded to external pins are in input pull-up configuration after reset. The configuration of these pads must be kept at reset state to avoid added current consumption.

5. In the open drain output column, “T” defines a true open drain I/O (P-Buffer and protection diode to V_{DD} are not implemented).

3 REGISTER & MEMORY MAP

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

The available memory locations consist of 128 bytes of register locations, up to 2 Kbytes of RAM and up to 60 Kbytes 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.

IMPORTANT: Memory locations marked as “Reserved” must never be accessed. Accessing a reseved area can have unpredictable effects on the device.

Related Documentation

AN 985: Executing Code in ST7 RAM

Figure 3. Memory Map

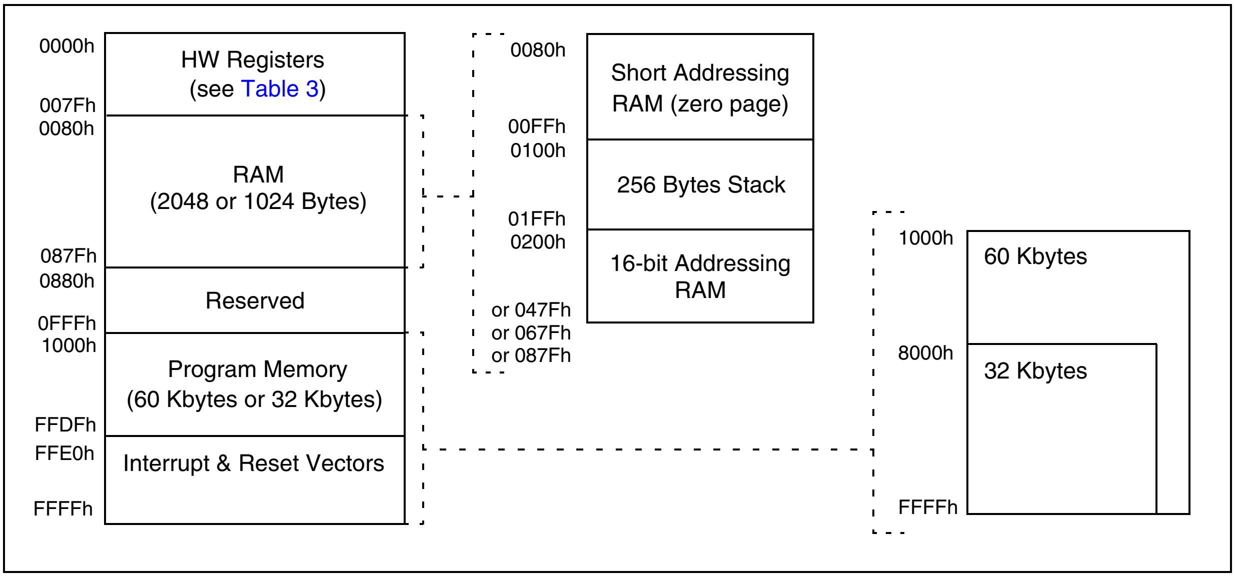

0000h
HW Registers
(see Table 3)
007Fh
0080h
RAM
(2048 or 1024 Bytes)
087Fh
0880h
Reserved
0FFFh
1000h
Program Memory
(60 Kbytes or 32 Kbytes)
FFDFh
FFE0h
Interrupt & Reset Vectors
FFFFh
0080h
Short Addressing
RAM (zero page)
00FFh
0100h
256 Bytes Stack
01FFh
0200h
16-bit Addressing
RAM
or 047Fh
or 067Fh
or 087Fh
1000h
60 Kbytes
8000h
32 Kbytes
FFFFh

Table 3. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h	Port A 2)	PADR	Port A Data Register	00h1)	R/W
0001h		PADDR	Port A Data Direction Register	00h	R/W
0002h		PAOR	Port A Option Register	00h	R/W
0003h	Port B 2)	PBDR	Port B Data Register	00h1)	R/W
0004h		PBDDR	Port B Data Direction Register	00h	R/W
0005h		PBOR	Port B Option Register	00h	R/W
0006h	Port C	PCDR	Port C Data Register	00h1)	R/W
0007h		PCDDR	Port C Data Direction Register	00h	R/W
0008h		PCOR	Port C Option Register	00h	R/W
0009h	Port D 2)	PDDR	Port D Data Register	00h1)	R/W
000Ah		PDDDR	Port D Data Direction Register	00h	R/W
000Bh		PDOR	Port D Option Register	00h	R/W
000Ch	Port E 2)	PEDR	Port E Data Register	00h1)	R/W
000Dh		PEDDR	Port E Data Direction Register	00h	R/W2)
000Eh		PEOR	Port E Option Register	00h	R/W2)
000Fh	Port F 2)	PFDR	Port F Data Register	00h1)	R/W
0010h		PFDDR	Port F Data Direction Register	00h	R/W
0011h		PFOR	Port F Option Register	00h	R/W
0012h	Port G 2)	PGDR	Port G Data Register	00h1)	R/W
0013h		PGDDR	Port G Data Direction Register	00h	R/W
0014h		PGOR	Port G Option Register	00h	R/W
0015h	Port H 2)	PHDR	Port H Data Register	00h1)	R/W
0016h		PHDDR	Port H Data Direction Register	00h	R/W
0017h		PHOR	Port H Option Register	00h	R/W
0018h	I^{2}C	I2CCR	I^{2}C Control Register	00h	R/W
0019h		I2CSR1	I^{2}C Status Register 1	00h	Read Only
001Ah		I2CSR2	I^{2}C Status Register 2	00h	Read Only
001Bh		I2CCCR	I^{2}C Clock Control Register	00h	R/W
001Ch		I2COAR1	I^{2}C Own Address Register 1	00h	R/W
001Dh		I2COAR2	I^{2}C Own Address Register2	00h	R/W
001Eh		I2CDR	I^{2}C Data Register	00h	R/W
001Fh 0020h	Reserved Area (2 Bytes)				
0021h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
0022h		SPICR	SPI Control Register	0xh	R/W
0023h		SPICSR	SPI Control/Status Register	00h	R/W

Address	Block	Register Label	Register Name	Reset Status	Remarks
0024h	ITC	ISPR0	Interrupt Software Priority Register 0	FFh	R/W
0025h		ISPR1	Interrupt Software Priority Register 1	FFh	R/W
0026h		ISPR2	Interrupt Software Priority Register 2	FFh	R/W
0027h		ISPR3	Interrupt Software Priority Register 3	FFh	R/W
0028h		EICR	External Interrupt Control Register	00h	R/W
0029h	Flash	FCSR	Flash Control/Status Register	00h	R/W
002Ah	WATCHDOG	WDGCR	Watchdog Control Register	7Fh	R/W
002Bh		SICSR	System Integrity Control/Status Register	000x 000x b	R/W
002Ch	MCC	MCCSR	Main Clock Control / Status Register	00h	R/W
002Dh		MCCBCR	Main Clock Controller: Beep Control Register	00h	R/W
002Eh to 0030h	Reserved Area (3 Bytes)				
0031h	TIMER A	TACR2	Timer A Control Register 2	00h	R/W
0032h		TACR1	Timer A Control Register 1	00h	R/W
0033h		TACSR	Timer A Control/Status Register	xxxx x0xx b	R/W
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	Reserved Area (1 Byte)				
0041h	TIMER B	TBCR2	Timer B Control Register 2	00h	R/W
0042h		TBCR1	Timer B Control Register 1	00h	R/W
0043h		TBCSR	Timer B Control/Status Register	xxxx x0xx b	R/W
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

Address	Block	Register Label	Register Name	Reset Status	Remarks
0050h	SCI	SCISR	SCI Status Register	C0h	Read Only
0051h		SCIDR	SCI Data Register	xxh	R/W
0052h		SCIBRR	SCI Baud Rate Register	00h	R/W
0053h		SCICR1	SCI Control Register 1	x000 0000b	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 to 006Fh	Reserved Area (24 Bytes)				
0070h	ADC	ADCCSR	Control/Status Register	00h	R/W
0071h		ADCDRH	Data High Register	00h	Read Only
0072h		ADCDRL	Data Low Register	00h	Read Only
0073h	PWM ART	PWMDCR3	PWM AR Timer Duty Cycle Register 3	00h	R/W
0074h		PWMDCR2	PWM AR Timer Duty Cycle Register 2	00h	R/W
0075h		PWMDCR1	PWM AR Timer Duty Cycle Register 1	00h	R/W
0076h		PWMDCR0	PWM AR Timer Duty Cycle Register 0	00h	R/W
0077h		PWMCR	PWM AR Timer Control Register	00h	R/W
0078h		ARTCSR	Auto-Reload Timer Control/Status Register	00h	R/W
0079h		ARTCAR	Auto-Reload Timer Counter Access Register	00h	R/W
007Ah		ARTARR	Auto-Reload Timer Auto-Reload Register	00h	R/W
007Bh		ARTICCSR	AR Timer Input Capture Control/Status Reg.	00h	R/W
007Ch		ARTICR1	AR Timer Input Capture Register 1	00h	Read Only
007Dh		ARTICR2	AR Timer Input Capture Register 1	00h	Read Only
007Eh 007Fh	Reserved Area (2 Bytes)				

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

Notes:

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents.
2. The bits associated with unavailable pins must always keep their reset value.

4 FLASH PROGRAM MEMORY

4.1 Introduction

The ST7 dual voltage High Density Flash (HDFlash) is a non-volatile memory that can be electrically erased as a single block or by individual sectors and programmed on a Byte-by-Byte basis using an external V_{PP} supply.

The HDFlash devices can be programmed and erased off-board (plugged in a programming tool) or on-board using ICP (In-Circuit Programming) or IAP (In-Application Programming).

The array matrix organisation allows each sector to be erased and reprogrammed without affecting other sectors.

4.2 Main Features

- Three Flash programming modes:
  - Insertion in a programming tool. In this mode, all sectors including option bytes can be programmed or erased.
  - ICP (In-Circuit Programming). In this mode, all sectors including option bytes can be programmed or erased without removing the device from the application board.
  - IAP (In-Application Programming) In this mode, all sectors except Sector 0, can be programmed or erased without removing the device from the application board and while the application is running.
- ICT (In-Circuit Testing) for downloading and executing user application test patterns in RAM
- Read-out protection
- Register Access Security System (RASS) to prevent accidental programming or erasing

4.3 Structure

The Flash memory is organised in sectors and can be used for both code and data storage.

Depending on the overall Flash memory size in the microcontroller device, there are up to three user sectors (see Table 4). Each of these sectors can be erased independently to avoid unnecessary erasing of the whole Flash memory when only a partial erasing is required.

The first two sectors have a fixed size of 4 Kbytes (see Figure 4). They are mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

Table 4. Sectors available in Flash devices

Flash Size (Kbytes)	Available Sectors
4	Sector 0
8	Sectors 0,1
> 8	Sectors 0,1, 2

4.3.1 Read-out Protection

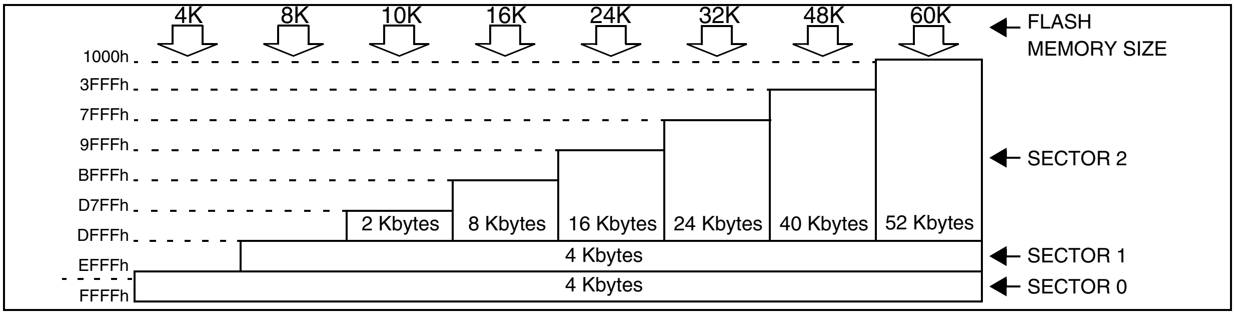
Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller.

In Flash devices, this protection is removed by reprogramming the option. In this case, the entire program memory is first automatically erased and the device can be reprogrammed.

Read-out protection is enabled and removed through the FMP_R bit in the option byte.

Note: The LVD is not supported if read-out protection is enabled.

Figure 4. Memory Map and Sector Address



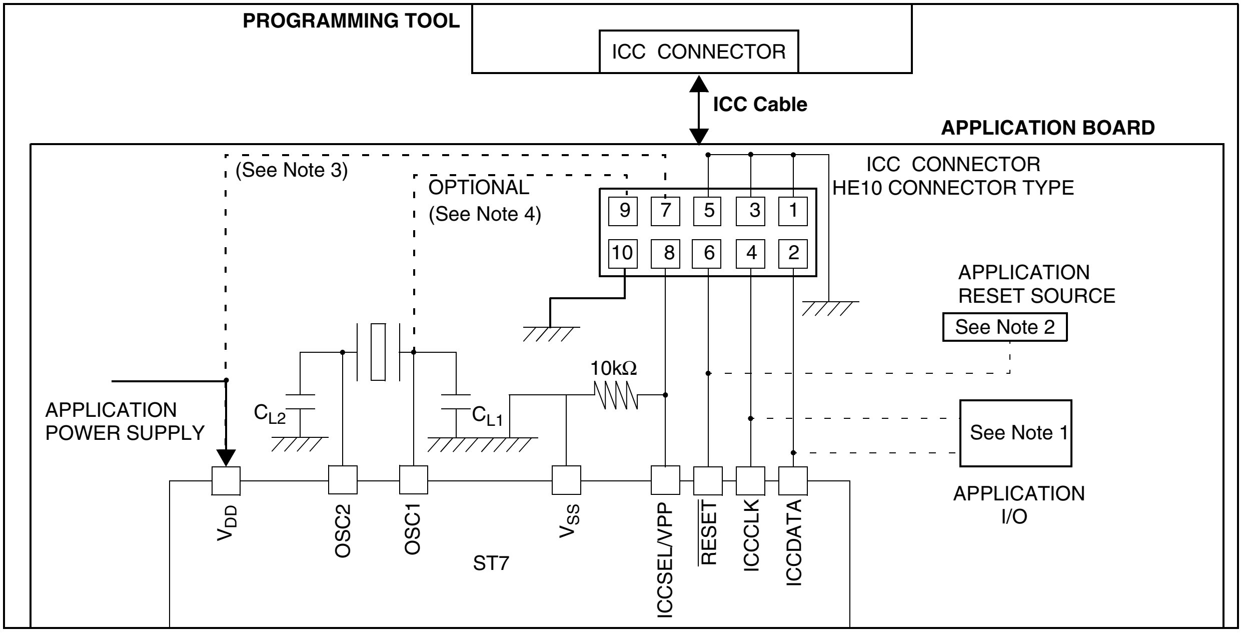
FLASH PROGRAM MEMORY (Cont'd)

4.4 ICC Interface

ICC needs a minimum of 4 and up to 6 pins to be connected to the programming tool (see Figure 5). These pins are:

- RESET: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input/output serial data pin
- ICCSEL/V_{PP}: programming voltage
- OSC1(or OSCIN): main clock input for external source (optional)
- V_{DD}: application board power supply (optional, see Figure 5, Note 3)

Figure 5. Typical ICC Interface


PROGRAMMING TOOL
ICC CONNECTOR
ICC Cable
APPLICATION BOARD
ICC CONNECTOR
HE10 CONNECTOR TYPE
(See Note 3)
OPTIONAL
(See Note 4)
9
7
5
3
1
10
8
6
4
2
APPLICATION
RESET SOURCE
See Note 2
10kΩ
APPLICATION
POWER SUPPLY
C_{L2}
C_{L1}
See Note 1
APPLICATION
I/O
V_{DD}
OSC2
OSC1
V_{SS}
ST7
ICCSEL/VPP
RESET
ICCCLK
ICCDATA

Notes:

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

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

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

4. Pin 9 has to be connected to the OSC1 or OSCIN pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte. ST7 devices with multi-oscillator capability need to have OSC2 grounded in this case.

FLASH PROGRAM MEMORY (Cont'd)

4.5 ICP (In-Circuit Programming)

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

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

When using an STMicroelectronics or third-party programming tool that supports ICP and the specific microcontroller device, the user needs only to implement the ICP hardware interface on the application board (see Figure 5). For more details on the pin locations, refer to the device pinout description.

4.6 IAP (In-Application Programming)

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

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

4.7 Related Documentation

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

4.7.1 Register Description

FLASH CONTROL/STATUS REGISTER (FCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	0

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

Figure 6. Flash Control/Status Register Address and Reset Value

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0029h	**FCSR** Reset Value	0	0	0	0	0	0	0	0

5 CENTRAL PROCESSING UNIT

5.1 INTRODUCTION

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

5.2 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

5.3 CPU REGISTERS

The six CPU registers shown in Figure 1 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)

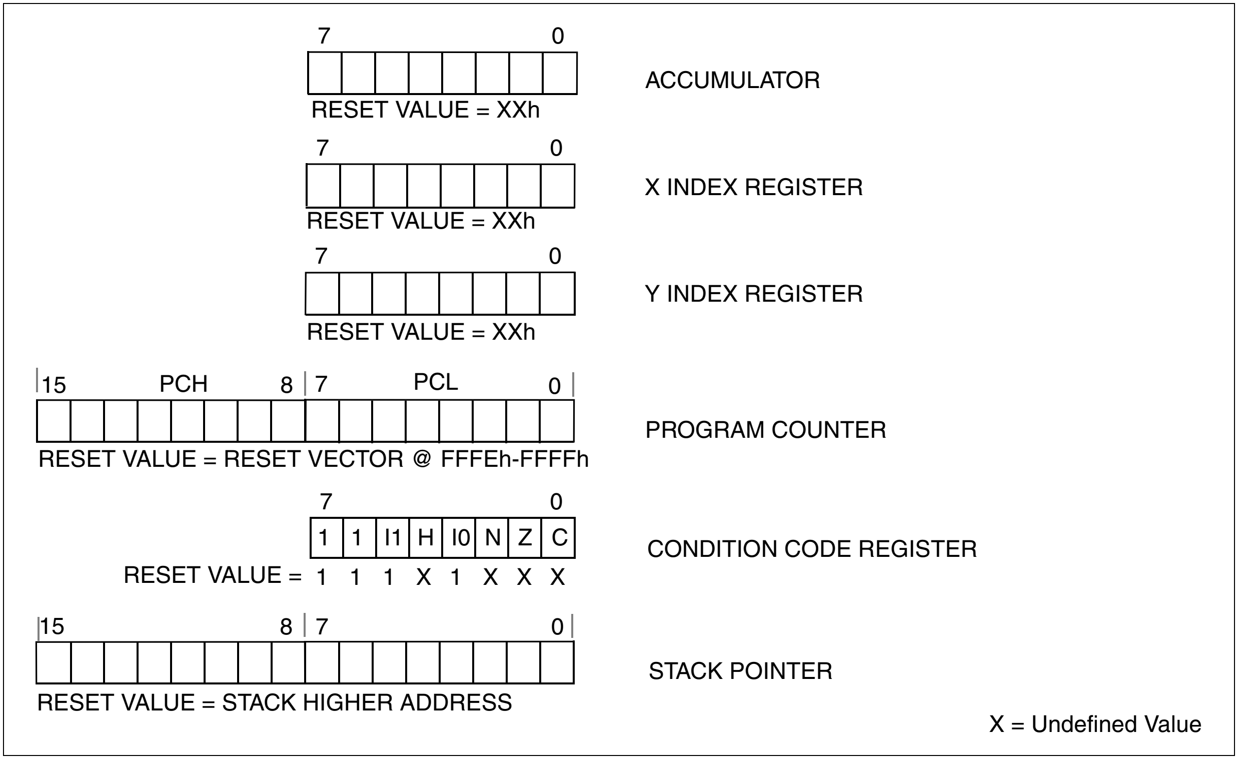
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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 7. CPU Registers


7
0
ACCUMULATOR
RESET VALUE = XXh
7
0
X INDEX REGISTER
RESET VALUE = XXh
7
0
Y INDEX REGISTER
RESET VALUE = XXh
15
PCH
8
7
PCL
0
PROGRAM COUNTER
RESET VALUE = RESET VECTOR @ FFFEh-FFFFh
7
0
1 1 I1 H I0 N Z C
CONDITION CODE REGISTER
RESET VALUE = 1 1 1 X 1 X X X
15
8
7
0
STACK POINTER
RESET VALUE = STACK HIGHER ADDRESS
X = Undefined Value

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 7th bit.
0: The result of the last operation is positive or null.
1: The result of the last operation is negative
(that is, the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

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

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

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

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

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

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

Interrupt Management Bits

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

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

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

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

See the interrupt management chapter for more details.

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 2).

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.

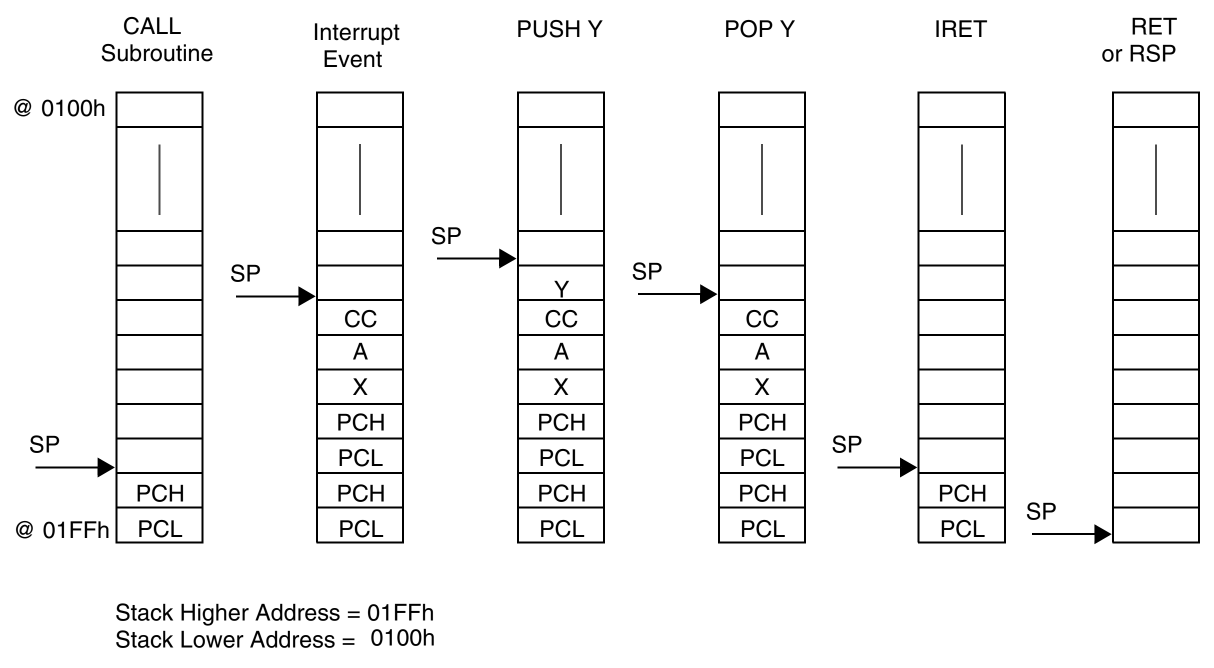
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 2.

– 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 8. Stack Manipulation Example



6 SUPPLY, RESET AND CLOCK MANAGEMENT

The device includes a range of utility features for securing the application in critical situations (for example in case of a power brown-out), and reducing the number of external components. An overview is shown in Figure 10.

For more details, refer to dedicated parametric section.

Main features

- Optional PLL for multiplying the frequency by 2 (not to be used with internal RC oscillator)
- Reset Sequence Manager (RSM)
- Multi-Oscillator Clock Management (MO)
  - 5 Crystal/Ceramic resonator oscillators
  - 1 Internal RC oscillator
- System Integrity Management (SI)
  - Main supply Low voltage detection (LVD)
  - Auxiliary Voltage detector (AVD) with interrupt capability for monitoring the main supply or the EVD pin

6.1 PHASE LOCKED LOOP

If the clock frequency input to the PLL is in the range 2 to 4 MHz, the PLL can be used to multiply the frequency by two to obtain an f_{OSC2} of 4 to 8 MHz. The PLL is enabled by option byte. If the PLL is disabled, then $f_{OSC2} = f_{OSC}/2$.

Caution: The PLL is not recommended for applications where timing accuracy is required. See “PLL Characteristics” on page 146.

Figure 9. PLL Block Diagram

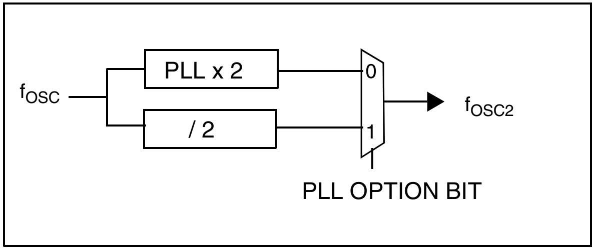
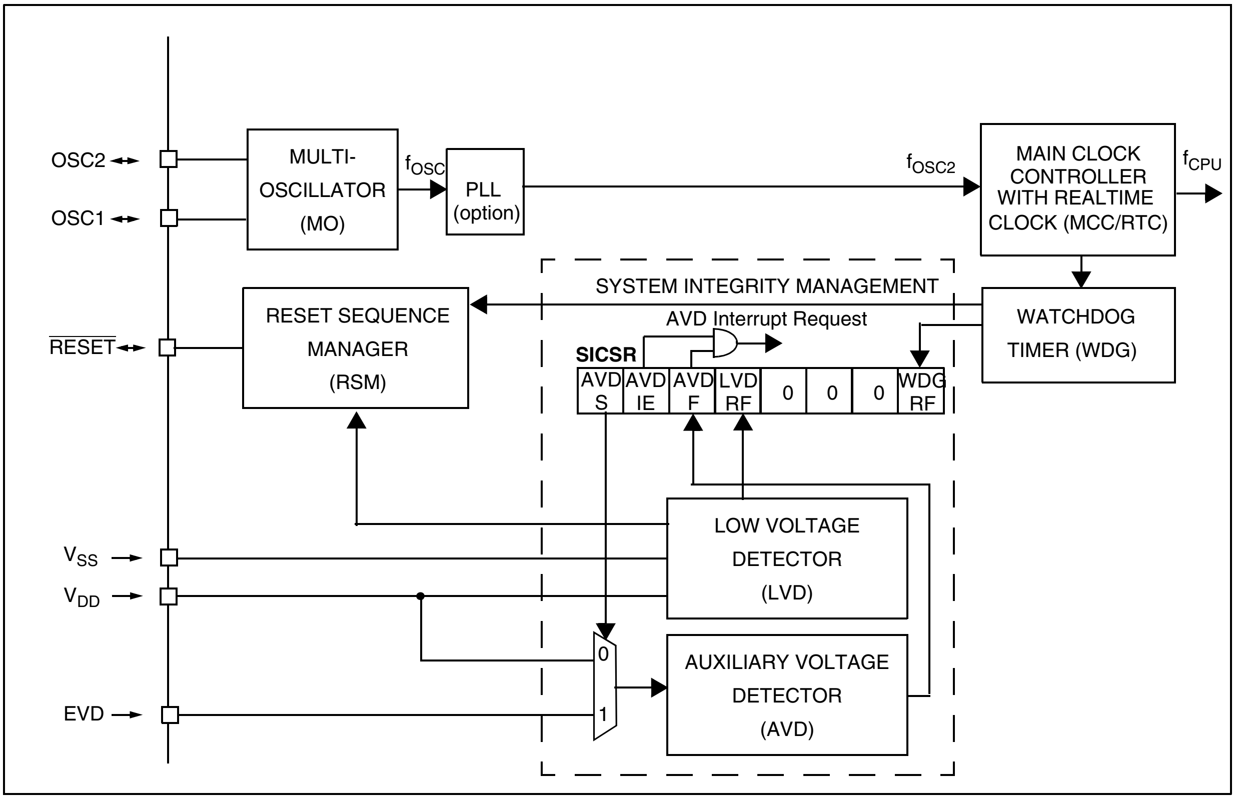

f_{OSC}
PLL x 2
/ 2
0
1
f_{OSC2}
PLL OPTION BIT

Figure 10. Clock, Reset and Supply Block Diagram


OSC2
OSC1
MULTI-OSCILLATOR (MO)
f_{OSC}
PLL (option)
f_{OSC2}
MAIN CLOCK CONTROLLER WITH REALTIME CLOCK (MCC/RTC)
f_{CPU}
SYSTEM INTEGRITY MANAGEMENT
RESET
RESET SEQUENCE MANAGER (RSM)
AVD Interrupt Request
WATCHDOG TIMER (WDG)
SICSR
AVD S
AVD IE
AVD F
LVD RF
0
0
0
WDG RF
LOW VOLTAGE DETECTOR (LVD)
V_{SS}
V_{DD}
0
1
AUXILIARY VOLTAGE DETECTOR (AVD)
EVD

6.2 MULTI-OSCILLATOR (MO)

The main clock of the ST7 can be generated by three different source types coming from the multi-oscillator block:

- an external source
- 4 crystal or ceramic resonator oscillators
- an internal high frequency RC oscillator

Each oscillator is optimized for a given frequency range in terms of consumption and is selectable through the option byte. The associated hardware configurations are shown in Table 5. Refer to the electrical characteristics section for more details.

Caution: The OSC1 and/or OSC2 pins must not be left unconnected. For the purposes of Failure Mode and Effect Analysis, it should be noted that if the OSC1 and/or OSC2 pins are left unconnected, the ST7 main oscillator may start and, in this configuration, could generate an f_{OSC} clock frequency in excess of the allowed maximum (>16 MHz.), putting the ST7 in an unsafe/undefined state. The product behaviour must therefore be considered undefined when the OSC pins are left unconnected.

External Clock Source

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

Crystal/Ceramic Oscillators

This family of oscillators has the advantage of producing a very accurate rate on the main clock of the ST7. The selection within a list of 4 oscillators with different frequency ranges has to be done by option byte in order to reduce consumption (refer to section 13.1 on page 167 for more details on the frequency ranges). In this mode of the multi-oscillator, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. The loading capacitance values must be adjusted according to the selected oscillator.

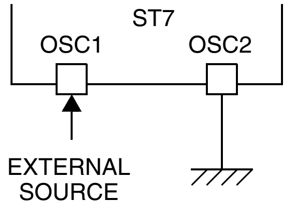
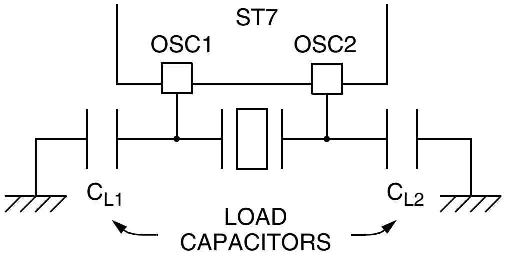
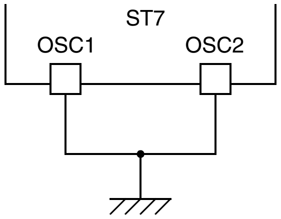
These oscillators are not stopped during the RESET phase to avoid losing time in the oscillator start-up phase.

Internal RC Oscillator

This oscillator allows a low cost solution for the main clock of the ST7 using only an internal resistor and capacitor. Internal RC oscillator mode has the drawback of a lower frequency accuracy and should not be used in applications that require accurate timing.

In this mode, the two oscillator pins have to be tied to ground.

Table 5. ST7 Clock Sources

	Hardware Configuration
External Clock	
Crystal/Ceramic Resonators	 ST7 OSC1 OSC2 C_{L1} C_{L2} LOAD CAPACITORS
Internal RC Oscillator	

6.3 RESET SEQUENCE MANAGER (RSM)

6.3.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 12:

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

These sources act on the $\overline{\text{RESET}}$ pin and it is always kept low during the delay phase.

The RESET service routine vector is fixed at addresses FFFEh-FFFFh in the ST7 memory map.

The basic RESET sequence consists of 3 phases as shown in Figure 11:

- Active Phase depending on the RESET source
- 256 or 4096 CPU clock cycle delay (selected by option byte)
- RESET vector fetch

The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay should be selected by option byte to correspond to the stabilization time of the external oscillator used in the application (see section 13.1 on page 167).

The RESET vector fetch phase duration is 2 clock cycles.

Figure 11. RESET Sequence Phases

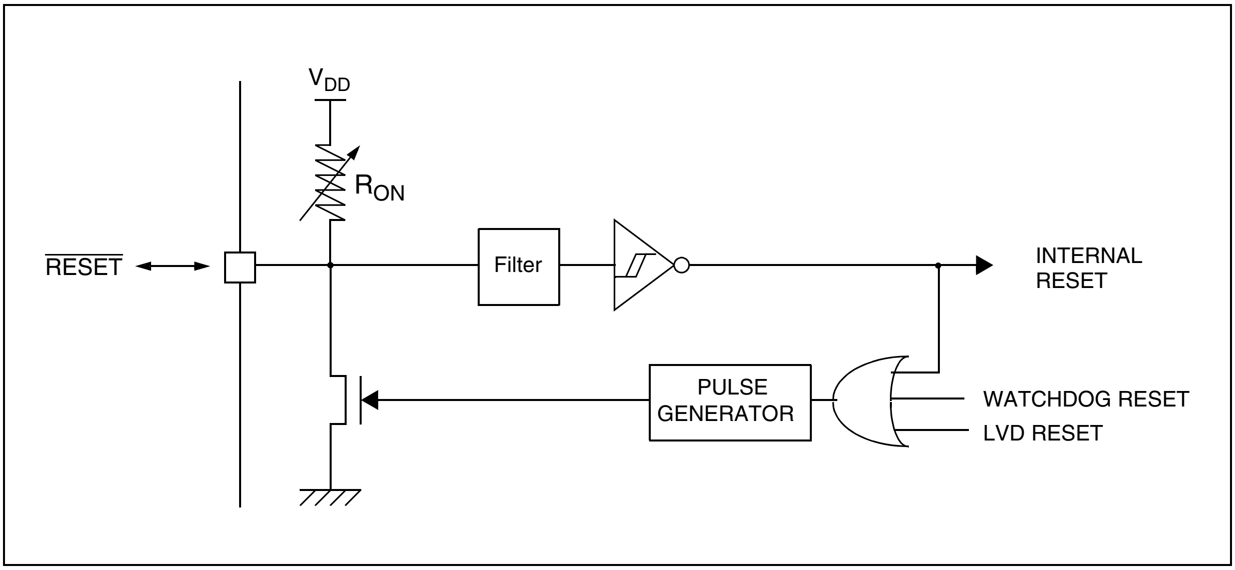
RESET		
Active Phase	INTERNAL RESET 256 or 4096 CLOCK CYCLES	FETCH VECTOR

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

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device. See "CONTROL PIN CHARACTERISTICS" on page 154 for more details.

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see Figure 13). This detection is asynchronous and therefore the MCU can enter reset state even in HALT mode.

Figure 12. Reset Block Diagram



RESET SEQUENCE MANAGER (Cont'd)

The $\overline{\text{RESET}}$ pin is an asynchronous signal which plays a major role in EMS performance. In a noisy environment, it is recommended to follow the guidelines mentioned in the electrical characteristics section.

If the external $\overline{\text{RESET}}$ pulse is shorter than $t_{w(RSTL)out}$ (see short ext. Reset in Figure 13), the signal on the $\overline{\text{RESET}}$ pin may be stretched. Otherwise the delay will not be applied (see long ext. Reset in Figure 13). Starting from the external RESET pulse recognition, the device $\overline{\text{RESET}}$ pin acts as an output that is pulled low during at least $t_{w(RSTL)out}$.

6.3.3 External Power-On RESET

If the LVD is disabled by option byte, to start up the microcontroller correctly, the user must ensure by means of an external reset circuit that the reset signal is held low until V_{DD} is over the minimum level specified for the selected f_{OSC} frequency. (see "OPERATING CONDITIONS" on page 136)

A proper reset signal for a slow rising V_{DD} supply can generally be provided by an external RC network connected to the $\overline{\text{RESET}}$ pin.

6.3.4 Internal Low Voltage Detector (LVD) RESET

Two different RESET sequences caused by the internal LVD circuitry can be distinguished:

- Power-On RESET
- Voltage Drop RESET

The device $\overline{\text{RESET}}$ pin acts as an output that is pulled low when $V_{DD}<V_{IT+}$ (rising edge) or $V_{DD}<V_{IT-}$ (falling edge) as shown in Figure 13.

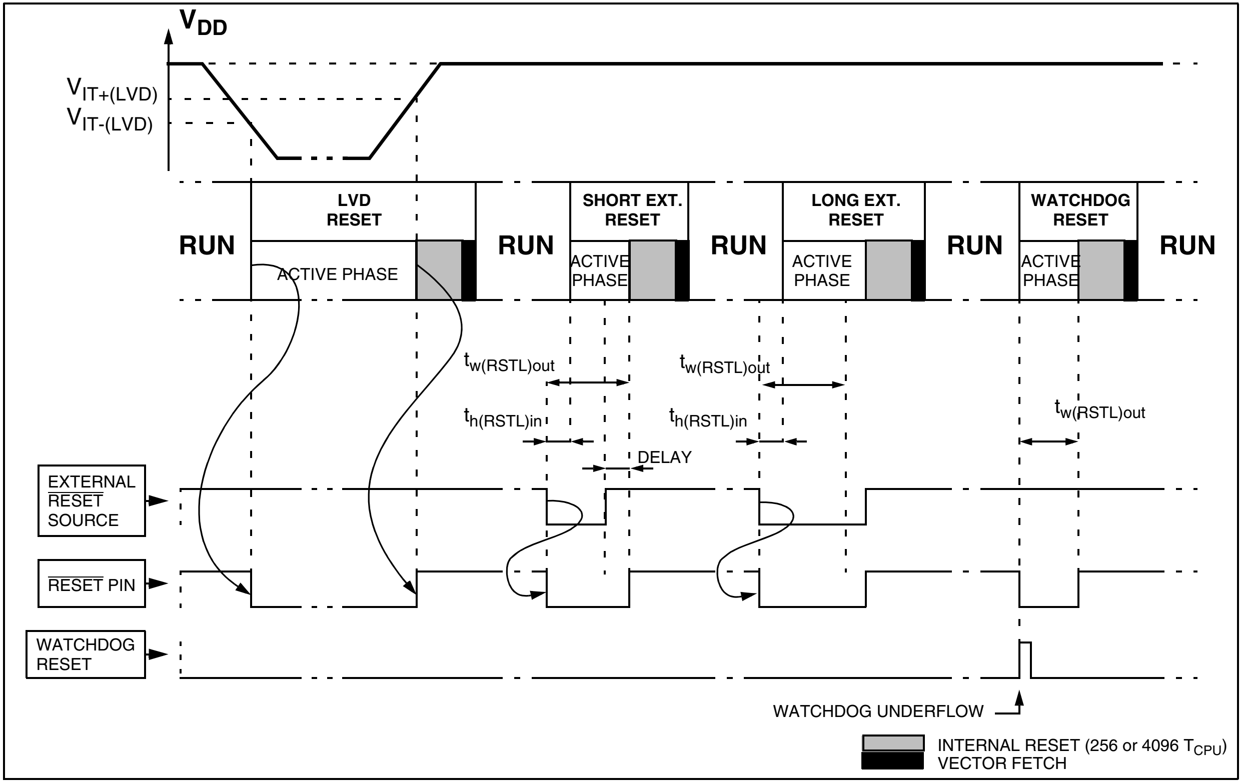
The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

6.3.5 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 13.

Starting from the Watchdog counter underflow, the device $\overline{\text{RESET}}$ pin acts as an output that is pulled low during at least $t_{w(RSTL)out}$.

Figure 13. RESET Sequences


V_DD
V_IT+(LVD)
V_IT-(LVD)
RUN
LVD RESET
ACTIVE PHASE
RUN
SHORT EXT. RESET
ACTIVE PHASE
RUN
LONG EXT. RESET
ACTIVE PHASE
RUN
WATCHDOG RESET
ACTIVE PHASE
RUN
t_w(RSTL)out
t_h(RSTL)in
t_w(RSTL)out
t_h(RSTL)in
DELAY
t_w(RSTL)out
EXTERNAL RESET SOURCE
RESET PIN
WATCHDOG RESET
WATCHDOG UNDERFLOW
INTERNAL RESET (256 or 4096 T_CPU)
VECTOR FETCH

6.4 SYSTEM INTEGRITY MANAGEMENT (SI)

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

6.4.1 Low Voltage Detector (LVD)

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

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

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

– V_{IT+} when V_{DD} is rising
– V_{IT-} when V_{DD} is falling

The LVD function is illustrated in Figure 14.

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

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

– under full software control
– in static safe reset

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

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

Notes:

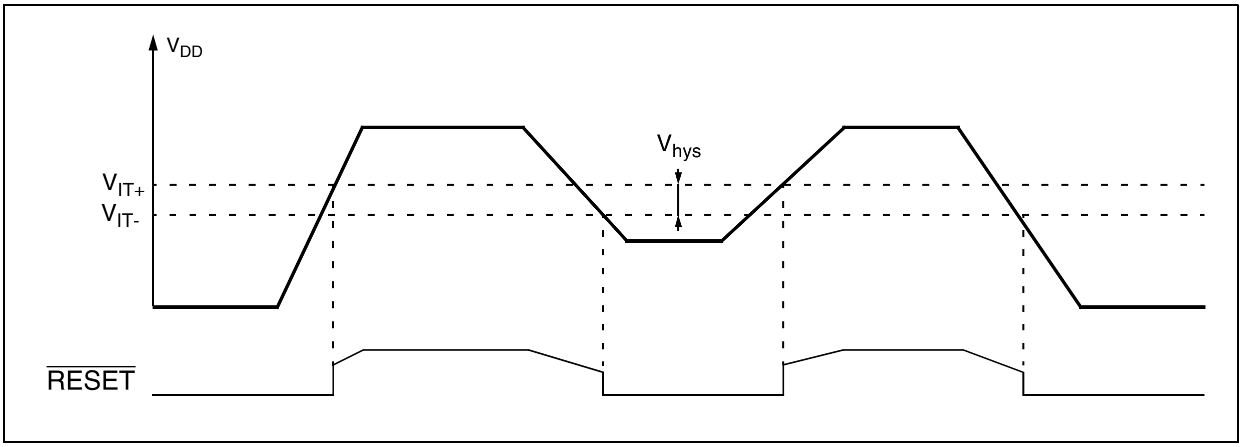
The LVD allows the device to be used without any external RESET circuitry.

If the medium or low thresholds are selected, the detection may occur outside the specified operating voltage range. Below 3.8 V, device operation is not guaranteed.

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

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

Figure 14. Low Voltage Detector vs Reset



SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.4.2 Auxiliary Voltage Detector (AVD)

The Voltage Detector function (AVD) is based on an analog comparison between a $V_{IT-(AVD)}$ and $V_{IT+(AVD)}$ reference value and the V_{DD} main supply or the external EVD pin voltage level (V_{EVD}). The V_{IT-} reference value for falling voltage is lower than the V_{IT+} reference value for rising voltage in order to avoid parasitic detection (hysteresis).

The output of the AVD comparator is directly readable by the application software through a real-time status bit (AVDF) in the SICSR register. This bit is read only.

Caution: The AVD function is active only if the LVD is enabled through the option byte.

6.4.2.1 Monitoring the V_{DD} Main Supply

This mode is selected by clearing the AVDS bit in the SICSR register.

The AVD voltage threshold value is relative to the selected LVD threshold configured by option byte (see section 13.1 on page 167).

If the AVD interrupt is enabled, an interrupt is generated when the voltage crosses the $V_{IT+(AVD)}$ or $V_{IT-(AVD)}$ threshold (AVDF bit toggles).

In the case of a drop in voltage, the AVD interrupt acts as an early warning, allowing software to shut down safely before the LVD resets the microcontroller. See Figure 15.

The interrupt on the rising edge is used to inform the application that the V_{DD} warning state is over.

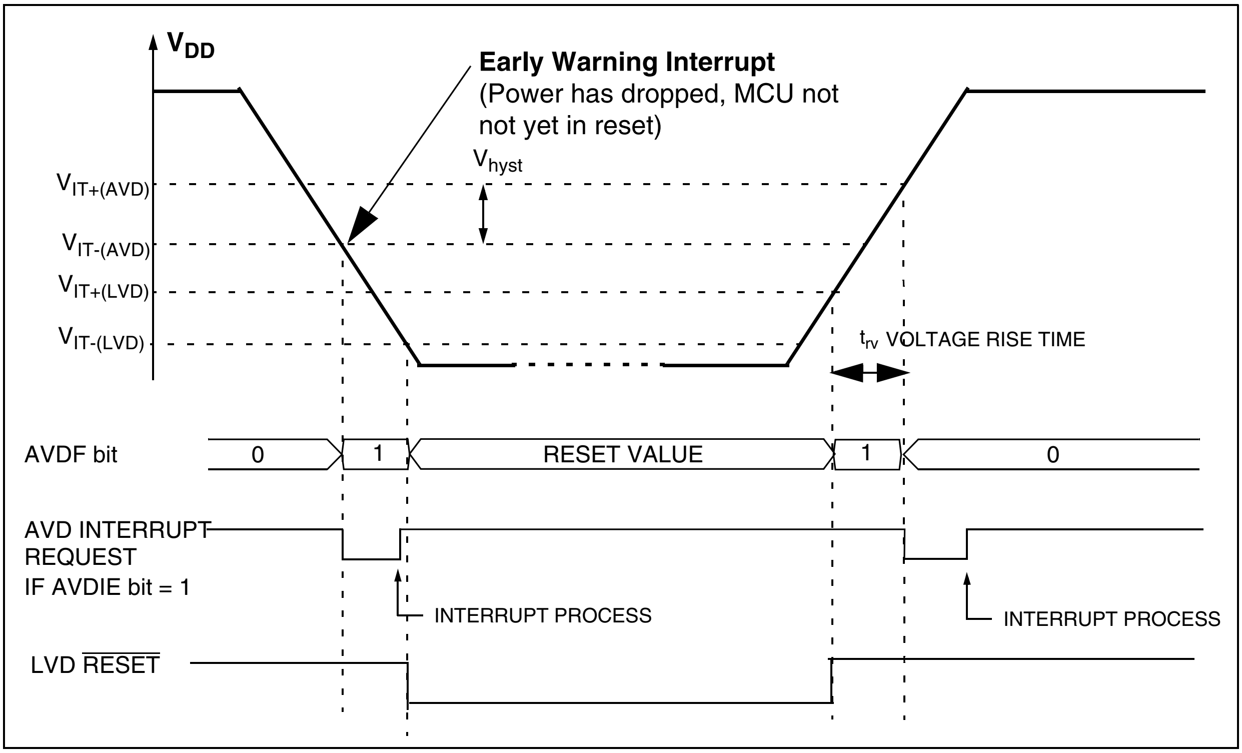
If the voltage rise time t_{rv} is less than 256 or 4096 CPU cycles (depending on the reset delay selected by option byte), no AVD interrupt will be generated when $V_{IT+(AVD)}$ is reached.

If t_{rv} is greater than 256 or 4096 cycles then:

– If the AVD interrupt is enabled before the $V_{IT+(AVD)}$ threshold is reached, then 2 AVD interrupts will be received: the first when the AVDIE bit is set, and the second when the threshold is reached.

– If the AVD interrupt is enabled after the $V_{IT+(AVD)}$ threshold is reached then only one AVD interrupt will occur.

Figure 15. Using the AVD to Monitor V_{DD} (AVDS bit=0)


V_DD
Early Warning Interrupt
(Power has dropped, MCU not
not yet in reset)
V_hyst
V_IT+(AVD)
V_IT-(AVD)
V_IT+(LVD)
V_IT-(LVD)
t_rv VOLTAGE RISE TIME
AVDF bit
0
1
RESET VALUE
1
0
AVD INTERRUPT
REQUEST
IF AVDIE bit = 1
INTERRUPT PROCESS
INTERRUPT PROCESS
LVD RESET

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.4.2.2 Monitoring a Voltage on the EVD pin

This mode is selected by setting the AVDS bit in the SICSR register.

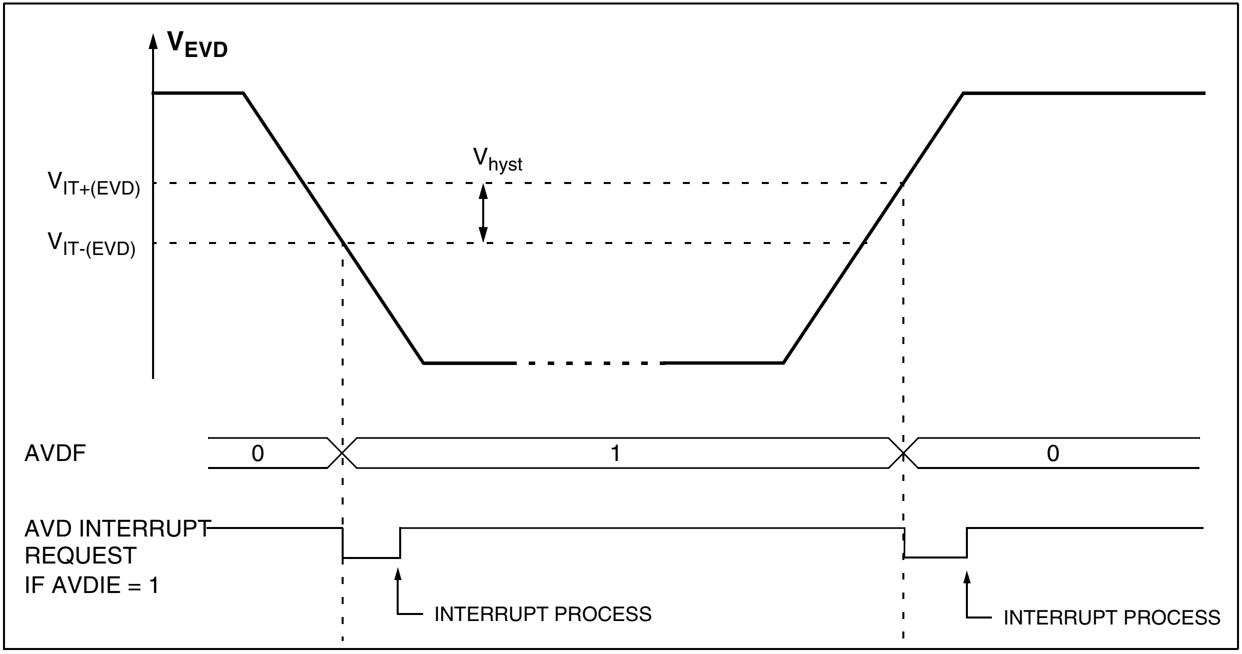
The AVD circuitry can generate an interrupt when the AVDIE bit of the SICSR register is set. This interrupt is generated on the rising and falling edges of the comparator output. This means it is generated when either one of these two events occur:

– V_{EVD} rises up to $V_{IT+(EVD)}$
– V_{EVD} falls down to $V_{IT-(EVD)}$

The EVD function is illustrated in Figure 16.

For more details, refer to the Electrical Characteristics section.

Figure 16. Using the Voltage Detector to Monitor the EVD pin (AVDS bit=1)


V_EVD
V_hyst
V_IT+(EVD)
V_IT-(EVD)
AVDF
0
1
0
AVD INTERRUPT
REQUEST
IF AVDIE = 1
INTERRUPT PROCESS
INTERRUPT PROCESS

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.4.3 Low Power Modes

Mode	Description
WAIT	No effect on SI. AVD interrupts cause the device to exit from Wait mode.
HALT	The CRSR register is frozen.

6.4.3.1 Interrupts

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

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

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.4.4 Register Description

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

Read/Write

Reset Value: 000x 000x (00h)

7							0
AVDS	AVDIE	AVDF	LVDRF	0	0	0	WDGRF

Bit 7 = **AVDS** *Voltage Detection selection*
This bit is set and cleared by software. Voltage Detection is available only if the LVD is enabled by option byte.
0: Voltage detection on V_{DD} supply
1: Voltage detection on EVD pin

Bit 6 = **AVDIE** *Voltage Detector interrupt enable*
This bit is set and cleared by software. It enables an interrupt to be generated when the AVDF flag changes (toggles). The pending interrupt information is automatically cleared when software enters the AVD interrupt routine.
0: AVD interrupt disabled
1: AVD interrupt enabled

Bit 5 = **AVDF** *Voltage Detector flag*
This read-only bit is set and cleared by hardware. If the AVDIE bit is set, an interrupt request is generated when the AVDF bit changes value. Refer to Figure 15 and to Section 6.4.2.1 for additional details.
0: V_{DD} or V_{EVD} over $V_{IT+(AVD)}$ threshold
1: V_{DD} or V_{EVD} under $V_{IT-(AVD)}$ threshold

Bit 4 = **LVDRF** *LVD reset flag*
This bit indicates that the last Reset was generated by the LVD block. It is set by hardware (LVD reset) and cleared by software (writing zero). See WDGRF flag description for more details. When the LVD is disabled by OPTION BYTE, the LVDRF bit value is undefined.

Bits 3:1 = Reserved, must be kept cleared.

Bit 0 = **WDGRF** *Watchdog reset flag*
This bit indicates that the last Reset was generated by the Watchdog peripheral. It is set by hardware (watchdog reset) and cleared by software (writing zero) or an LVD Reset (to ensure a stable cleared state of the WDGRF flag when CPU starts).
Combined with the LVDRF flag information, the flag description is given by the following table.

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

Application notes

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

CAUTION: When the LVD is not activated with the associated option byte, the WDGRF flag can not be used in the application.

7 INTERRUPTS

7.1 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
  - 2 non maskable events: RESET, TRAP
  - 1 maskable Top Level event: TLI

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.

7.2 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 6). The processing flow is shown in Figure 17

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).

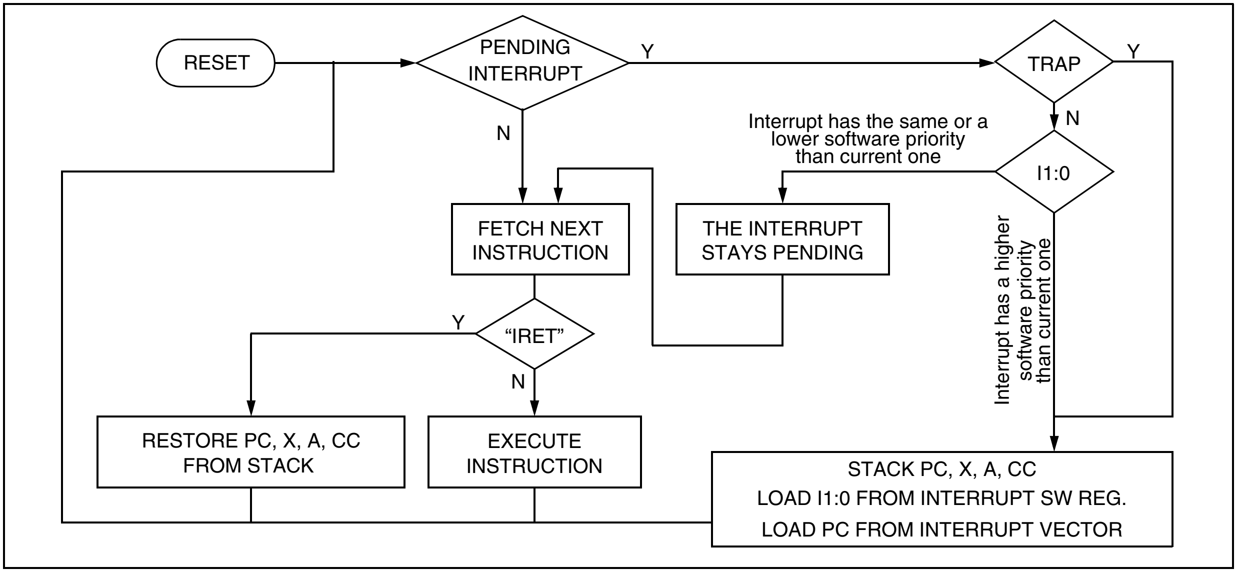
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

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 6. 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 17. Interrupt Processing Flowchart


RESET
PENDING INTERRUPT
Y
TRAP
Y
N
N
Interrupt has the same or a lower software priority than current one
I1:0
FETCH NEXT INSTRUCTION
THE INTERRUPT STAYS PENDING
Interrupt has a higher software priority than current one
Y
“IRET”
N
RESTORE PC, X, A, CC FROM STACK
EXECUTE INSTRUCTION
STACK PC, X, A, CC
LOAD I1:0 FROM INTERRUPT SW REG.
LOAD PC FROM INTERRUPT VECTOR

INTERRUPTS (Cont'd)

Servicing Pending Interrupts

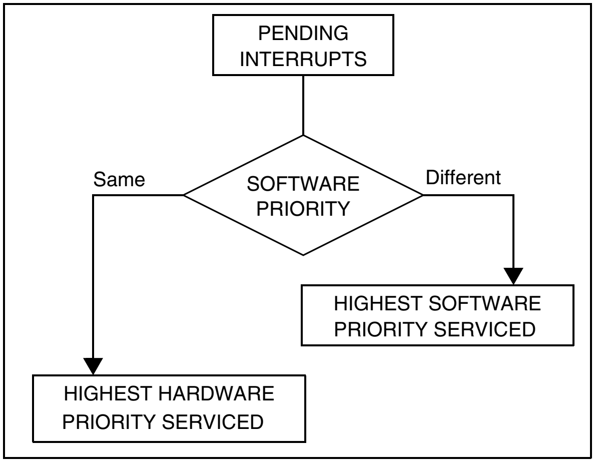
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– 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 18 describes this decision process.

Figure 18. Priority Decision Process


PENDING INTERRUPTS
Same
SOFTWARE PRIORITY
Different
HIGHEST SOFTWARE PRIORITY SERVICED
HIGHEST HARDWARE PRIORITY SERVICED

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: TLI, RESET and TRAP 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, 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 17). 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.

- TRAP (Non Maskable Software Interrupt)

This software interrupt is serviced when the TRAP instruction is executed. It will be serviced according to the flowchart in Figure 17.

Caution: TRAP can be interrupted by a TLI.

- 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.

- TLI (Top Level Hardware Interrupt)

This hardware interrupt occurs when a specific edge is detected on the dedicated TLI pin. It will be serviced according to the flowchart in Figure 17 as a trap.
Caution: A TRAP instruction must not be used in a TLI service routine.

- External Interrupts

External interrupts allow the processor to exit from HALT low power mode. External interrupt sensitivity is software selectable through the External Interrupt Control register (EICR).
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)

7.3 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 from 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 18.

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.

7.4 CONCURRENT & NESTED MANAGEMENT

The following Figure 19 and Figure 20 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 20. The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0, TLI. The software priority is given for each interrupt.

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

Figure 19. Concurrent Interrupt Management

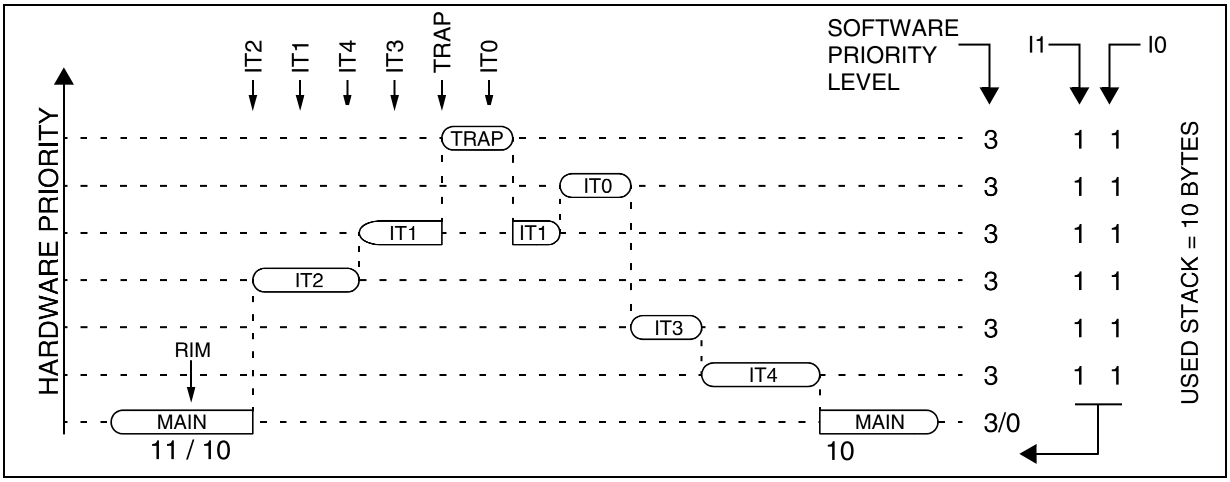
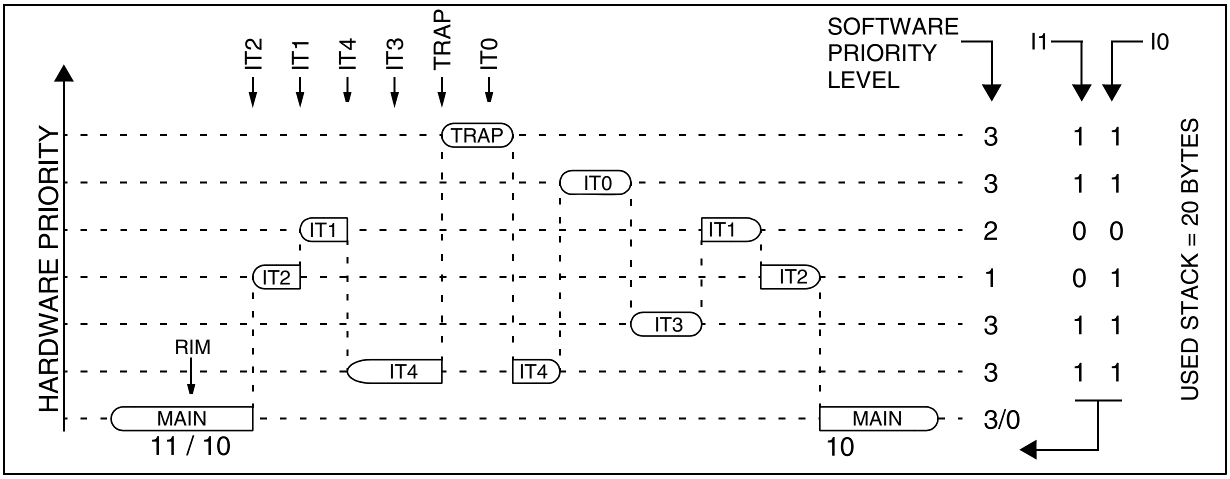

IT2
IT1
IT4
IT3
TRAP
IT0
SOFTWARE PRIORITY LEVEL
I1
I0
HARDWARE PRIORITY
TRAP
IT0
IT1
IT1
IT2
IT3
IT4
RIM
MAIN
MAIN
11 / 10
10
3 1 1
3 1 1
3 1 1
3 1 1
3 1 1
3 1 1
3/0
USED STACK = 10 BYTES

Figure 20. Nested Interrupt Management


IT2
IT1
IT4
IT3
TRAP
IT0
SOFTWARE PRIORITY LEVEL
I1
I0
HARDWARE PRIORITY
TRAP
IT0
IT1
IT1
IT2
IT2
IT3
IT4
IT4
RIM
MAIN
MAIN
11 / 10
10
3 1 1
3 1 1
2 0 0
1 0 1
3 1 1
3 1 1
3/0
USED STACK = 20 BYTES

INTERRUPTS (Cont'd)

7.5 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**: TLI, TRAP and RESET events can interrupt a level 3 program.

INTERRUPT SOFTWARE PRIORITY REGISTERS (ISPRX)

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

Reset Value: 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 correspondance 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 TLI, RESET, and TRAP 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 TLI 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 7. Dedicated Interrupt Instruction Set

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

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

INTERRUPTS (Cont'd)

Table 8. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT/ Active-halt3)	Address Vector
	RESET	Reset	N/A	Higher Priority ↓ Lower Priority	yes	FFFEh-FFFFh
	TRAP	Software interrupt			no	FFFCh-FFFDh
0	TLI	External top level interrupt	EICR		yes	FFFAh-FFFBh
1	MCC/RTC	Main clock controller time base interrupt	MCCSR		yes	FFF8h-FFF9h
2	ei0	External interrupt port A3..0	N/A		yes	FFF6h-FFF7h
3	ei1	External interrupt port F2..0			yes	FFF4h-FFF5h
4	ei2	External interrupt port B3..0			yes	FFF2h-FFF3h
5	ei3	External interrupt port B7..4			yes	FFF0h-FFF1h
6	Not used					FFEEh-FFEFh
7	SPI	SPI peripheral interrupts	SPICSR		yes[1]	FFECh-FFEDh
8	TIMER A	TIMER A peripheral interrupts	TASR		no	FFEAh-FFEBh
9	TIMER B	TIMER B peripheral interrupts	TBSR		no	FFE8h-FFE9h
10	SCI	SCI Peripheral interrupts	SCISR		no	FFE6h-FFE7h
11	AVD	Auxiliary Voltage detector interrupt	SICSR		no	FFE4h-FFE5h
12	I2C	I2C Peripheral interrupts	(see periph)		no	FFE2h-FFE3h
13	PWM ART	PWM ART interrupt	ARTCSR		yes[2]	FFE0h-FFE1h

Notes:

1. Exit from HALT possible when SPI is in slave mode.
2. Exit from HALT possible when PWM ART is in external clock mode.
3. In Flash devices only a RESET or MCC/RTC interrupt can be used to wake-up from Active-halt mode.

7.6 EXTERNAL INTERRUPTS

7.6.1 I/O Port Interrupt Sensitivity

The external interrupt sensitivity is controlled by the IPA, IPB and ISxx bits of the EICR register (Figure 21). This control allows to have up to 4 fully independent external interrupt source sensitivities.

Each external interrupt source can be generated on four (or five) different events on the pin:

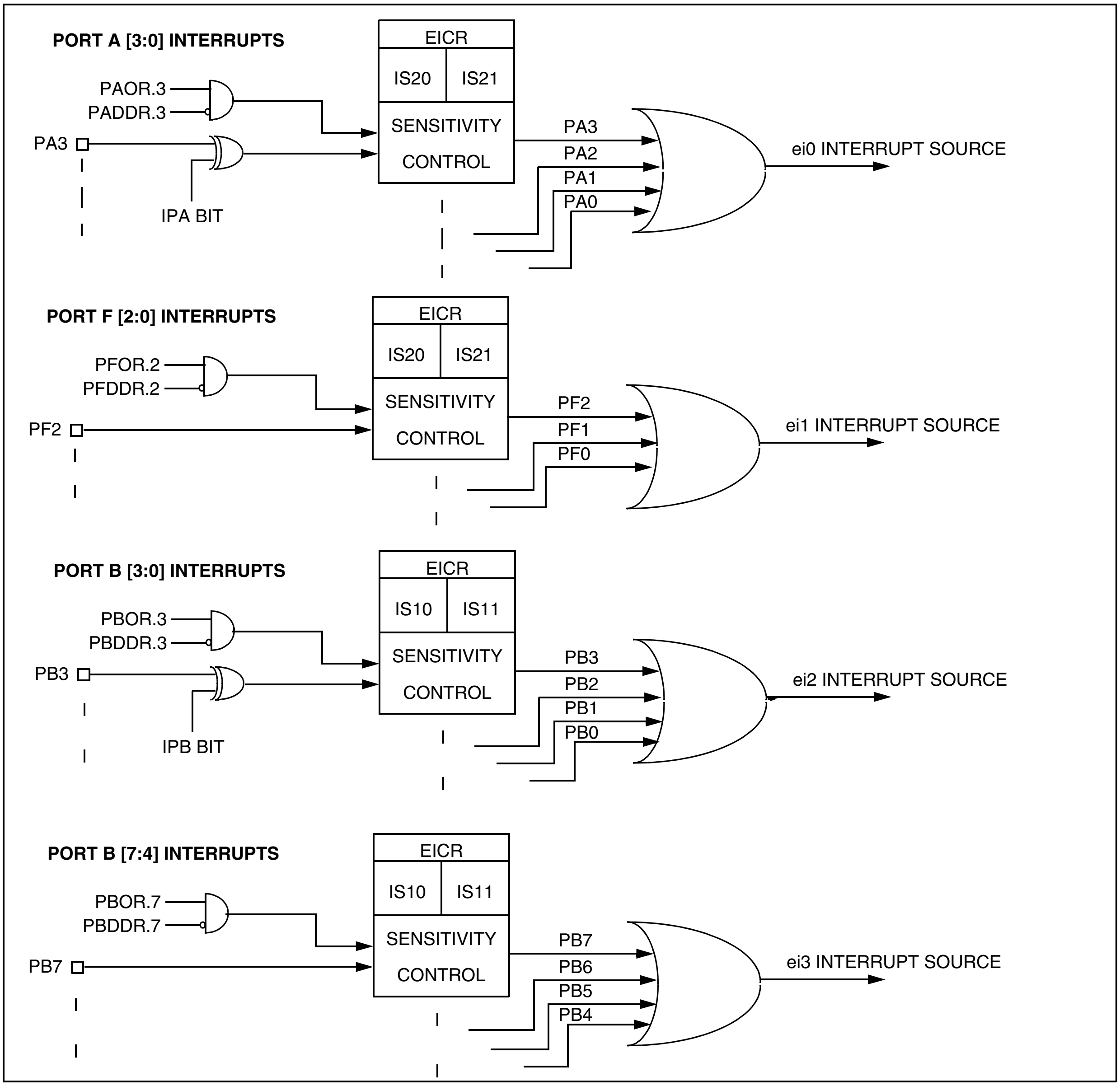
- Falling edge
- Rising edge
- Falling and rising edge
- Falling edge and low level
- Rising edge and high level (only for ei0 and ei2)

To guarantee correct functionality, the sensitivity bits in the EICR register can be modified only when the I1 and I0 bits of the CC register are both set to 1 (level 3). This means that interrupts must be disabled before changing sensitivity.

The pending interrupts are cleared by writing a different value in the ISx[1:0], IPA or IPB bits of the EICR.

INTERRUPTS (Cont'd)

Figure 21. External Interrupt Control bits


PORT A [3:0] INTERRUPTS
EICR
IS20
IS21
SENSITIVITY CONTROL
PAOR.3
PADDR.3
PA3
IPA BIT
PA3
PA2
PA1
PA0
ei0 INTERRUPT SOURCE
PORT F [2:0] INTERRUPTS
EICR
IS20
IS21
SENSITIVITY CONTROL
PFOR.2
PFDDR.2
PF2
PF2
PF1
PF0
ei1 INTERRUPT SOURCE
PORT B [3:0] INTERRUPTS
EICR
IS10
IS11
SENSITIVITY CONTROL
PBOR.3
PBDDR.3
PB3
IPB BIT
PB3
PB2
PB1
PB0
ei2 INTERRUPT SOURCE
PORT B [7:4] INTERRUPTS
EICR
IS10
IS11
SENSITIVITY CONTROL
PBOR.7
PBDDR.7
PB7
PB7
PB6
PB5
PB4
ei3 INTERRUPT SOURCE

7.7 EXTERNAL INTERRUPT CONTROL REGISTER (EICR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
IS11	IS10	IPB	IS21	IS20	IPA	TLIS	TLIE

Bit 7:6 = **IS1[1:0]** *ei2 and ei3 sensitivity*
The interrupt sensitivity, defined using the IS1[1:0] bits, is applied to the following external interrupts:
- ei2 (port B3..0)

IS11	IS10	External Interrupt Sensitivity	
		IPB bit =0	**IPB bit =1**
0	0	Falling edge & low level	Rising edge & high level
0	1	Rising edge only	Falling edge only
1	0	Falling edge only	Rising edge only
1	1	Rising and falling edge	

- ei3 (port B7..4)

IS11	IS10	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

These 2 bits can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

Bit 5 = **IPB** *Interrupt polarity for port B*
This bit is used to invert the sensitivity of the port B [3:0] external interrupts. It can be set and cleared by software only when I1 and I0 of the CC register are both set to 1 (level 3).
0: No sensitivity inversion
1: Sensitivity inversion

Bit 4:3 = **IS2[1:0]** *ei0 and ei1 sensitivity*
The interrupt sensitivity, defined using the IS2[1:0] bits, is applied to the following external interrupts:
- ei0 (port A3..0)

IS21	IS20	External Interrupt Sensitivity	
		IPA bit =0	**IPA bit =1**
0	0	Falling edge & low level	Rising edge & high level
0	1	Rising edge only	Falling edge only
1	0	Falling edge only	Rising edge only
1	1	Rising and falling edge	

- ei1 (port F2..0)

IS21	IS20	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

These 2 bits can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

Bit 2 = **IPA** *Interrupt polarity for port A*
This bit is used to invert the sensitivity of the port A [3:0] external interrupts. It can be set and cleared by software only when I1 and I0 of the CC register are both set to 1 (level 3).
0: No sensitivity inversion
1: Sensitivity inversion

Bit 1 = **TLIS** *TLI sensitivity*
This bit allows to toggle the TLI edge sensitivity. It can be set and cleared by software only when TLIE bit is cleared.
0: Falling edge
1: Rising edge

Bit 0 = **TLIE** *TLI enable*
This bit allows to enable or disable the TLI capability on the dedicated pin. It is set and cleared by software.
0: TLI disabled
1: TLI enabled
Note: a parasitic interrupt can be generated when clearing the TLIE bit.

INTERRUPTS (Cont'd)

Table 9. Nested Interrupts Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0024h		ei1		ei0		MCC		TLI	
	ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1		
	Reset Value	1	1	1	1	1	1	1	1
0025h		SPI				ei3		ei2	
	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
0026h		AVD		SCI		TIMER B		TIMER A	
	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
0027h						PWMART		I2C	
	ISPR3					I1_13	I0_13	I1_12	I0_12
	Reset Value	1	1	1	1	1	1	1	1
0028h	**EICR**	IS11	IS10	IPB	IS21	IS20	IPA	TLIS	TLIE
	Reset Value	0	0	0	0	0	0	0	0

8 POWER SAVING MODES

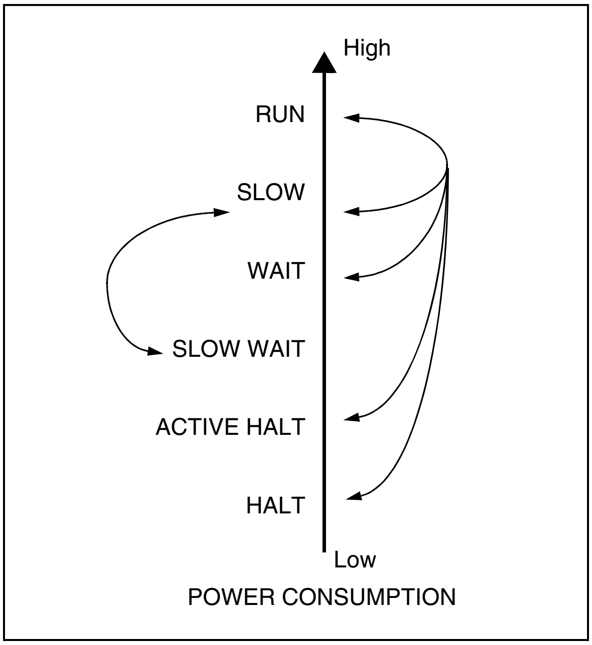
8.1 INTRODUCTION

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 (see Figure 22): Slow, Wait (Slow wait), Active-halt and Halt.

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 or multiplied by 2 (f_{OSC2}).

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 22. Power Saving Mode Transitions


High
RUN
SLOW
WAIT
SLOW WAIT
ACTIVE HALT
HALT
Low
POWER CONSUMPTION

8.2 SLOW MODE

This mode has two targets:

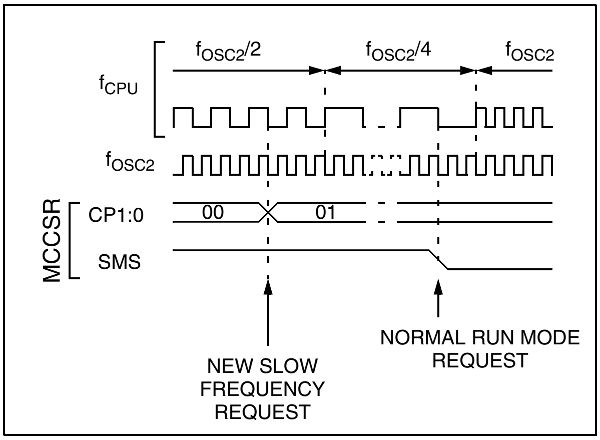
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

Slow mode is controlled by three bits in the MCCSR 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 master clock frequency (f_{OSC2}) can be divided by 2, 4, 8 or 16. The CPU and peripherals are clocked at this lower frequency (f_{CPU}).

Note: Slow-wait mode is activated when entering the WAIT mode while the device is already in Slow mode.

Figure 23. Slow Mode Clock Transitions


$f_{OSC2}/2$
$f_{OSC2}/4$
f_{OSC2}
f_{CPU}
f_{OSC2}
MCCSR
CP1:0
00
01
SMS
NEW SLOW FREQUENCY REQUEST
NORMAL RUN MODE REQUEST

POWER SAVING MODES (Cont'd)

8.3 WAIT MODE

WAIT mode places the MCU in a low power consumption mode by stopping the CPU.

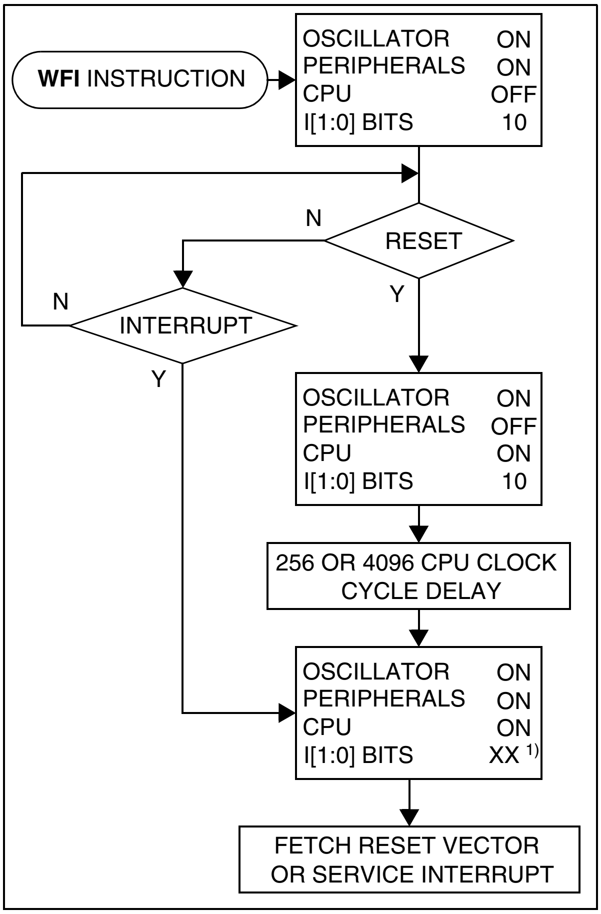
This power saving mode is selected by calling the 'WFI' instruction.

All peripherals remain active. During WAIT mode, the I[1:0] bits of the CC register are forced to '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 24.

Figure 24. WAIT Mode Flow-chart


WFI INSTRUCTION
OSCILLATOR ON
PERIPHERALS ON
CPU OFF
I[1:0] BITS 10
N
RESET
Y
N
INTERRUPT
Y
OSCILLATOR ON
PERIPHERALS OFF
CPU ON
I[1:0] BITS 10
256 OR 4096 CPU CLOCK CYCLE DELAY
OSCILLATOR ON
PERIPHERALS ON
CPU ON
I[1:0] BITS XX 1)
FETCH RESET VECTOR OR SERVICE INTERRUPT

Note:

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

POWER SAVING MODES (Cont'd)

8.4 ACTIVE-HALT AND HALT MODES

Active-halt and Halt modes are the two lowest power consumption modes of the MCU. They are both entered by executing the 'HALT' instruction. The decision to enter either in Active-halt or Halt mode is given by the MCC/RTC interrupt enable flag (OIE bit in MCCSR register).

MCCSR OIE bit	Power Saving Mode entered when HALT instruction is executed
0	Halt mode
1	ACTIVE-HALT mode

8.4.1 ACTIVE-HALT MODE

ACTIVE-HALT mode is the lowest power consumption mode of the MCU with a real-time clock available. It is entered by executing the 'HALT' instruction when the OIE bit of the Main Clock Controller Status register (MCCSR) is set (see section 9.2 on page 53 for more details on the MCCSR register).

The MCU can exit Active-halt mode on reception of an MCC/RTC interrupt or a RESET. When exiting Active-halt mode by means of an interrupt, no 256 or 4096 CPU cycle delay occurs. The CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 26).

When entering Active-halt mode, the I[1:0] bits in the CC register are forced to '10b' to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.

In Active-halt mode, only the main oscillator and its associated counter (MCC/RTC) are running to keep a wake-up time base. All other peripherals are not clocked except those which get their clock supply from another clock generator (such as external or auxiliary oscillator).

The safeguard against staying locked in Active-halt mode is provided by the oscillator interrupt.

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

CAUTION: When exiting Active-halt mode following an MCC/RTC interrupt, OIE bit of MCCSR register must not be cleared before t_{DELAY} after the interrupt occurs (t_{DELAY} = 256 or 4096 t_{CPU} delay depending on option byte). Otherwise, the ST7 enters HALT mode for the remaining t_{DELAY} period.

Figure 25. Active-halt Timing Overview

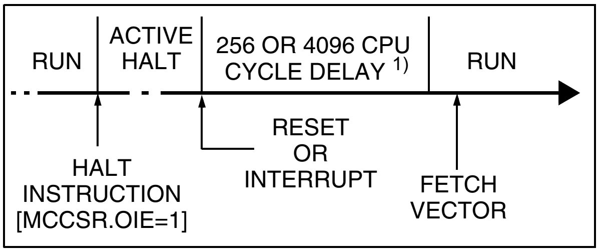
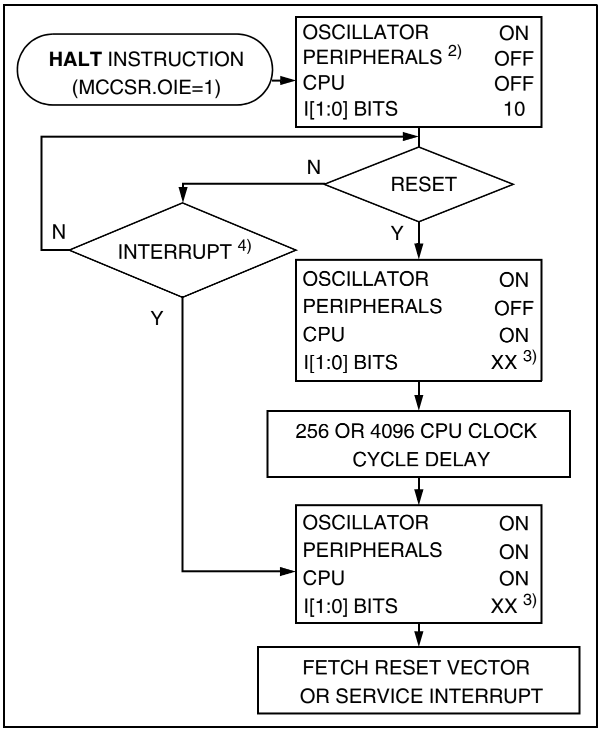

RUN
ACTIVE HALT
256 OR 4096 CPU CYCLE DELAY 1)
RUN
HALT INSTRUCTION [MCCSR.OIE=1]
RESET OR INTERRUPT
FETCH VECTOR

Figure 26. Active-halt Mode Flow-chart


HALT INSTRUCTION (MCCSR.OIE=1)
OSCILLATOR ON
PERIPHERALS 2) OFF
CPU OFF
I[1:0] BITS 10
N
RESET
N
INTERRUPT 4)
Y
Y
OSCILLATOR ON
PERIPHERALS OFF
CPU ON
I[1:0] BITS XX 3)
256 OR 4096 CPU CLOCK CYCLE DELAY
OSCILLATOR ON
PERIPHERALS ON
CPU ON
I[1:0] BITS XX 3)
FETCH RESET VECTOR OR SERVICE INTERRUPT

Notes:

1. This delay occurs only if the MCU exits Active-halt mode by means of a RESET.
2. Peripheral clocked with an external clock source can still be active.
3. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and restored when the CC register is popped.
4. In Flash devices only the MCC/RTC interrupt can exit the MCU from Active-halt mode.

POWER SAVING MODES (Cont'd)

8.4.2 HALT MODE

The Halt mode is the lowest power consumption mode of the MCU. It is entered by executing the 'HALT' instruction when the OIE bit of the Main Clock Controller Status register (MCCSR) is cleared (see section 9.2 on page 53 for more details on the MCCSR register).

The MCU can exit Halt mode on reception of either a specific interrupt (see Table 8, "Interrupt Mapping," on page 37) or a RESET. When exiting Halt mode by means of a RESET or an interrupt, the oscillator is immediately turned on and the 256 or 4096 CPU cycle delay is used to stabilize the oscillator. After the start up delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 28).

When entering Halt mode, the I[1:0] bits in the CC register are forced to '10b'to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.

In Halt 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 or auxiliary oscillator).

The compatibility of Watchdog operation with Halt mode is configured by the "WDGHALT" option bit of the option byte. The HALT instruction when executed while the Watchdog system is enabled, can generate a Watchdog RESET (see section 13.1 on page 167 for more details).

Figure 27. Halt Timing Overview

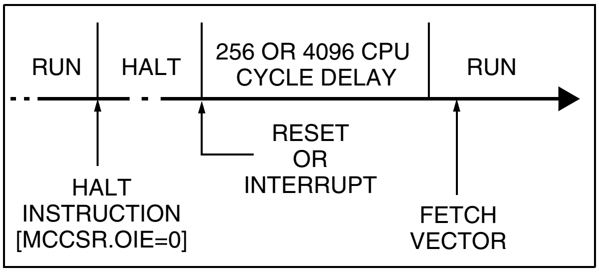
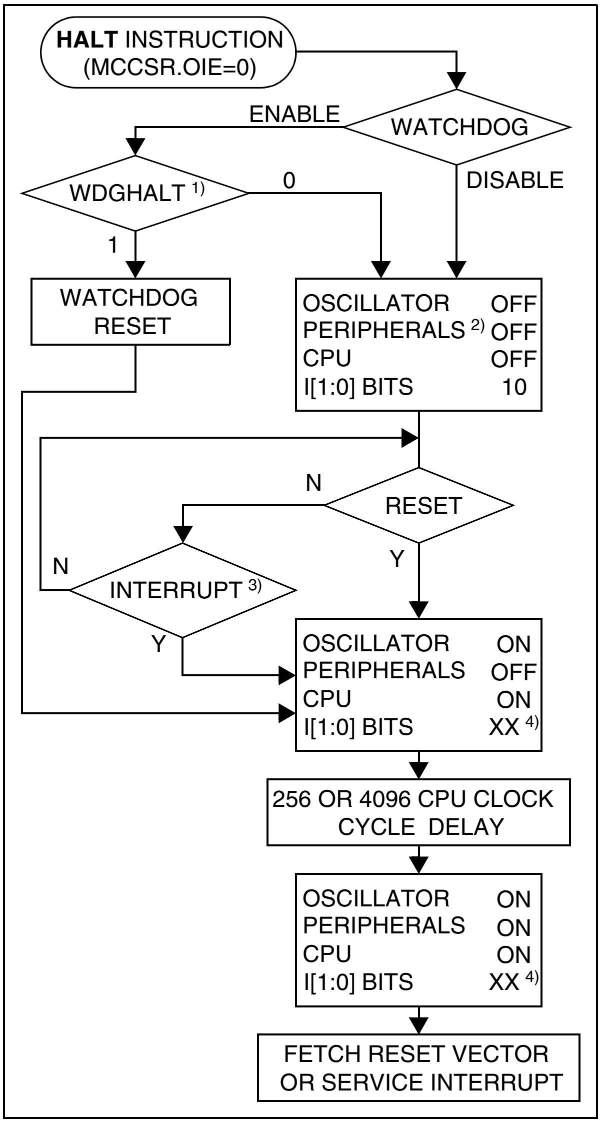

RUN
HALT
256 OR 4096 CPU CYCLE DELAY
RUN
RESET OR INTERRUPT
HALT INSTRUCTION [MCCSR.OIE=0]
FETCH VECTOR

Figure 28. Halt Mode Flow-chart


HALT INSTRUCTION (MCCSR.OIE=0)
WATCHDOG
ENABLE
DISABLE
WDGHALT 1)
0
1
WATCHDOG RESET
OSCILLATOR OFF
PERIPHERALS 2) OFF
CPU OFF
I[1:0] BITS 10
N
RESET
Y
N
INTERRUPT 3)
Y
OSCILLATOR ON
PERIPHERALS OFF
CPU ON
I[1:0] BITS XX 4)
256 OR 4096 CPU CLOCK CYCLE DELAY
OSCILLATOR ON
PERIPHERALS ON
CPU ON
I[1:0] BITS XX 4)
FETCH RESET VECTOR OR SERVICE INTERRUPT

Notes:

1. WDGHALT is an option bit. See option byte section for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to Table 8, "Interrupt Mapping," on page 37 for more details.
4. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped.

POWER SAVING MODES (Cont'd)

8.4.2.1 Halt Mode Recommendations

- Make sure that an external event is available to wake up the microcontroller from Halt mode.
- When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as "Input Pull-up with Interrupt" before executing the HALT instruction. The main reason for this is that the I/O may be wrongly configured due to external interference or by an unforeseen logical condition.
- For the same reason, reinitialize the level sensitiveness of each external interrupt as a precautionary measure.
- The opcode for the HALT instruction is 0x8E. To avoid an unexpected HALT instruction due to a program counter failure, it is advised to clear all occurrences of the data value 0x8E from memory. For example, avoid defining a constant in ROM with the value 0x8E.
- As the HALT instruction clears the interrupt mask in the CC register to allow interrupts, the user may choose to clear all pending interrupt bits before executing the HALT instruction. This avoids entering other peripheral interrupt routines after executing the external interrupt routine corresponding to the wake-up event (reset or external interrupt).

Related Documentation

AN 980: ST7 Keypad Decoding Techniques, Implementing Wake-Up on Keystroke

AN1014: How to Minimize the ST7 Power Consumption

AN1605: Using an active RC to wakeup the ST7LITE0 from power saving mode

I/O PORTS (Cont'd)

8.4.3 I/O Port Implementation

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

Standard Ports

PA5:4, PC7:0, PD7:0, PE7:34, PE1:0, PF7:3, PG7:0, PH7:0

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

Interrupt Ports

PA2:0, PB6:5, PB4, PB2:0, PF1:0 (with pull-up)

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

PA3, PB7, PB3, PF2 (without pull-up)

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

True Open Drain Ports

PA7:6

MODE	DDR
floating input	0
open drain (high sink ports)	1

Pull-up Input Port

PE2

MODE
pull-up input

Table 10. Port Configuration

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7:6	floating		true open-drain	
	PA5:4	floating	pull-up	open drain	push-pull
	PA3	floating	floating interrupt	open drain	push-pull
	PA2:0	floating	pull-up interrupt	open drain	push-pull
Port B	PB7, PB3	floating	floating interrupt	open drain	push-pull
	PB6:5, PB4, PB2:0	floating	pull-up interrupt	open drain	push-pull
Port C	PC7:0	floating	pull-up	open drain	push-pull
Port D	PD7:0	floating	pull-up	open drain	push-pull
Port E	PE7:3, PE1:0	floating	pull-up	open drain	push-pull
	PE2	pull-up input only			
Port F	PF7:3	floating	pull-up	open drain	push-pull
	PF2	floating	floating interrupt	open drain	push-pull
	PF1:0	floating	pull-up interrupt	open drain	push-pull
Port G	PG7:0	floating	pull-up	open drain	push-pull
Port H	PH7:0	floating	pull-up	open drain	push-pull

I/O PORTS (Cont'd)

Table 11. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all I/O port registers		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PAOR**								
0003h	**PBDR**	MSB							LSB
0004h	**PBDDR**								
0005h	**PBOR**								
0006h	**PCDR**	MSB							LSB
0007h	**PCDDR**								
0008h	**PCOR**								
0009h	**PDDR**	MSB							LSB
000Ah	**PDDDR**								
000Bh	**PDOR**								
000Ch	**PEDR**	MSB							LSB
000Dh	**PEDDR**								
000Eh	**PEOR**								
000Fh	**PFDR**	MSB							LSB
0010h	**PFDDR**								
0011h	**PFOR**								
0012h	**PGDR**	MSB							LSB
0013h	**PGDDR**								
0014h	**PGOR**								
0015h	**PHDR**	MSB							LSB
0016h	**PHDDR**								
0017h	**PHOR**								

Related Documentation

AN 970: SPI Communication between ST7 and EEPROM

AN1045: S/W implementation of I2C bus master

AN1048: Software LCD driver

9 ON-CHIP PERIPHERALS

9.1 WATCHDOG TIMER (WDG)

9.1.1 Introduction

The Watchdog timer is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The Watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the counter's contents before the T6 bit becomes cleared.

9.1.2 Main Features

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

9.1.3 Functional Description

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

If the watchdog is activated (the WDGA bit is set) and when the 7-bit timer (bits T[6:0]) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling the reset pin low for typically 30 µs.

The application program must write in the WDGCR register at regular intervals during normal operation to prevent an MCU reset. This downcounter is free-running: it counts down even if the watchdog is disabled. The value to be stored in the WDGCR register must be between FFh and C0h:

– The WDGA bit is set (watchdog enabled)

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

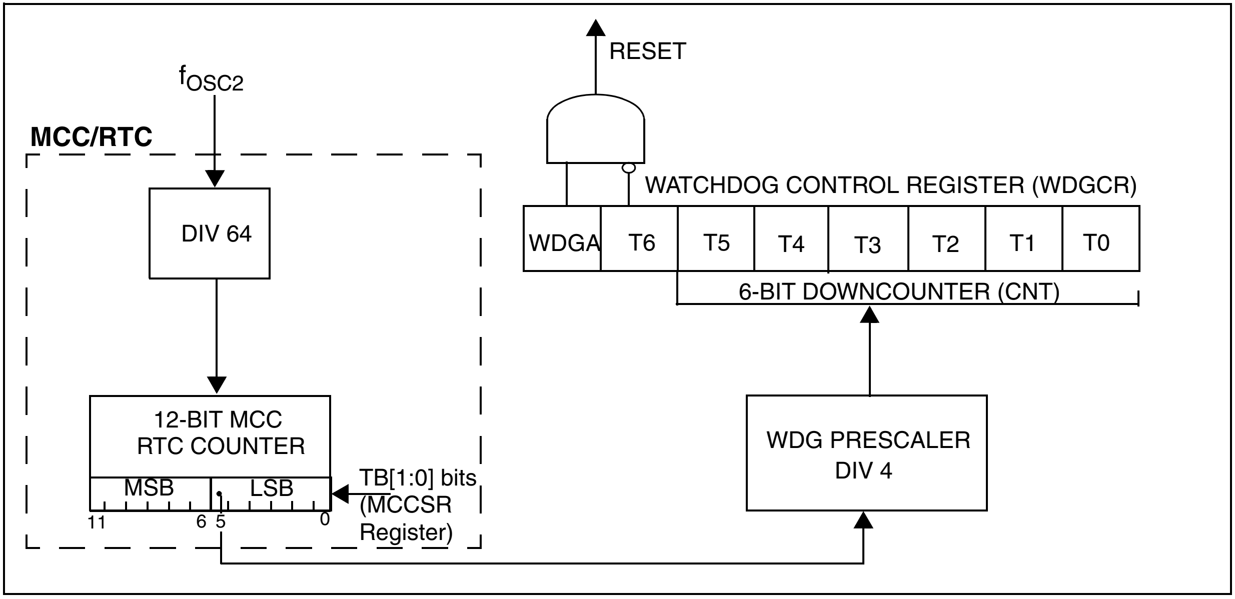
– The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset (see Figure 30. Approximate Timeout Duration). The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WDGCR register (see Figure 31).

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.

Figure 29. Watchdog Block Diagram



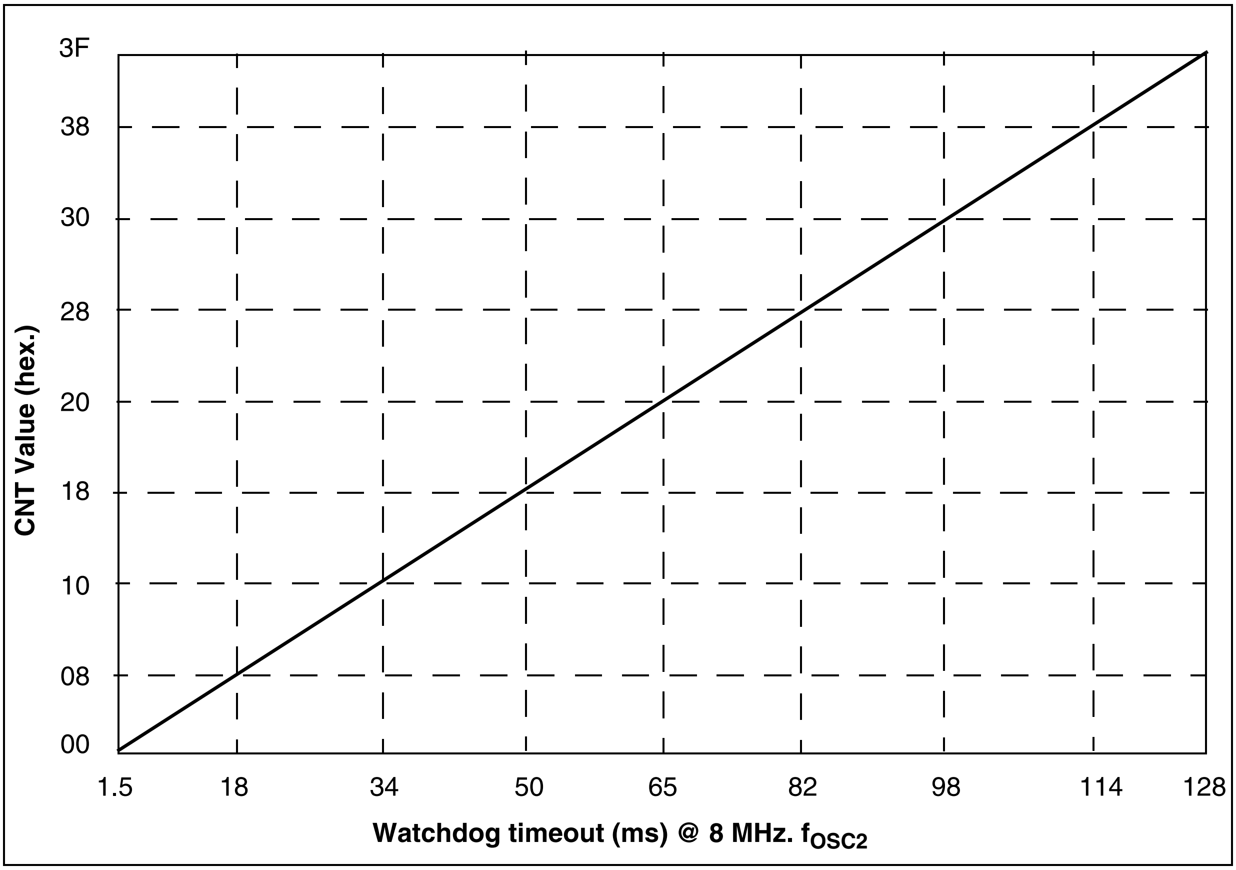
WATCHDOG TIMER (Cont'd)

9.1.4 How to Program the Watchdog Timeout

Figure 30 shows the linear relationship between the 6-bit value to be loaded in the Watchdog Counter (CNT) and the resulting timeout duration in milliseconds. This can be used for a quick calculation without taking the timing variations into account. If more precision is needed, use the formulae in Figure 31.

Caution: When writing to the WDGCR register, always write 1 in the T6 bit to avoid generating an immediate reset.

Figure 30. Approximate Timeout Duration



WATCHDOG TIMER (Cont'd)

Figure 31. Exact Timeout Duration (t_{min} and t_{max})

WHERE:

t_{min0} = (LSB + 128) x 64 x t_{OSC2}
t_{max0} = 16384 x t_{OSC2}
t_{OSC2} = 125 ns if f_{OSC2}=8 MHz

CNT = Value of T[5:0] bits in the WDGCR register (6 bits)
MSB and LSB are values from the table below depending on the timebase selected by the TB[1:0] bits in the MCCSR register

TB1 Bit (MCCSR Reg.)	TB0 Bit (MCCSR Reg.)	Selected MCCSR Timebase	MSB	LSB
0	0	2ms	4	59
0	1	4ms	8	53
1	0	10ms	20	35
1	1	25ms	49	54

To calculate the minimum Watchdog Timeout (t_{min}):

IF $\mathrm{CNT} < \left[\frac{\mathrm{MSB}}{4}\right]$ **THEN** $t_{min} = t_{min0} + 16384 \times \mathrm{CNT} \times t_{osc2}$

ELSE $t_{min} = t_{min0} + \left[16384 \times \left(\mathrm{CNT} - \left[\frac{4\mathrm{CNT}}{\mathrm{MSB}}\right]\right) + (192 + \mathrm{LSB}) \times 64 \times \left[\frac{4\mathrm{CNT}}{\mathrm{MSB}}\right]\right] \times t_{osc2}$

To calculate the maximum Watchdog Timeout (t_{max}):

IF $\mathrm{CNT} \leq \left[\frac{\mathrm{MSB}}{4}\right]$ **THEN** $t_{max} = t_{max0} + 16384 \times \mathrm{CNT} \times t_{osc2}$

ELSE $t_{max} = t_{max0} + \left[16384 \times \left(\mathrm{CNT} - \left[\frac{4\mathrm{CNT}}{\mathrm{MSB}}\right]\right) + (192 + \mathrm{LSB}) \times 64 \times \left[\frac{4\mathrm{CNT}}{\mathrm{MSB}}\right]\right] \times t_{osc2}$

Note: In the above formulae, division results must be rounded down to the next integer value.

Example:

With 2 ms timeout selected in MCCSR register

Value of T[5:0] Bits in WDGCR Register (Hex.)	Min. Watchdog Timeout (ms) t_{min}	Max. Watchdog Timeout (ms) t_{max}
00	1.496	2.048
3F	128	128.552

WATCHDOG TIMER (Cont'd)

9.1.5 Low Power Modes

Mode	Description		
SLOW	No effect on Watchdog.		
WAIT	No effect on Watchdog.		
HALT	**OIE bit in MCCSR register**	**WDGHALT bit in Option Byte**	
	0	0	No Watchdog reset is generated. The MCU enters Halt mode. The Watchdog counter is decremented once and then stops counting and is no longer able to generate a watchdog reset until the MCU receives an external interrupt or a reset. If an external interrupt is received, the Watchdog restarts counting after 256 or 4096 CPU clocks. If a reset is generated, the Watchdog is disabled (reset state) unless Hardware Watchdog is selected by option byte. For application recommendations see Section 9.1.7 below.
	0	1	A reset is generated.
	1	x	No reset is generated. The MCU enters Active Halt mode. The Watchdog counter is not decremented. It stop counting. When the MCU receives an oscillator interrupt or external interrupt, the Watchdog restarts counting immediately. When the MCU receives a reset the Watchdog restarts counting after 256 or 4096 CPU clocks.

9.1.6 Hardware Watchdog Option

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

9.1.7 Using Halt Mode with the WDG (WDGHALT option)

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

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

9.1.8 Interrupts

None.

9.1.9 Register Description

CONTROL REGISTER (WDGCR)

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

Note: This bit is not used if the hardware watchdog option is enabled by option byte.

Bit 6:0 = **T[6:0]** *7-bit counter (MSB to LSB).*
These bits contain the value of the watchdog counter. It is decremented every 16384 f_{OSC2} cycles (approx.). A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

Table 12. Watchdog Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
002Ah	**WDGCR** Reset Value	WDGA 0	T6 1	T5 1	T4 1	T3 1	T2 1	T1 1	T0 1

9.2 MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK AND BEEPER (MCC/RTC)

The Main Clock Controller consists of three different functions:

- a programmable CPU clock prescaler
- a clock-out signal to supply external devices
- a real-time clock timer with interrupt capability

Each function can be used independently and simultaneously.

9.2.1 Programmable CPU Clock Prescaler

The programmable CPU clock prescaler supplies the clock for the ST7 CPU and its internal peripherals. It manages Slow power saving mode (See Section 8.2 SLOW MODE for more details).

The prescaler selects the f_{CPU} main clock frequency and is controlled by three bits in the MCCSR register: CP[1:0] and SMS.

9.2.2 Clock-out Capability

The clock-out capability is an alternate function of an I/O port pin that outputs a f_{CPU} clock to drive external devices. It is controlled by the MCO bit in the MCCSR register.

CAUTION: When selected, the clock out pin suspends the clock during Active-halt mode.

9.2.3 Real-Time Clock Timer (RTC)

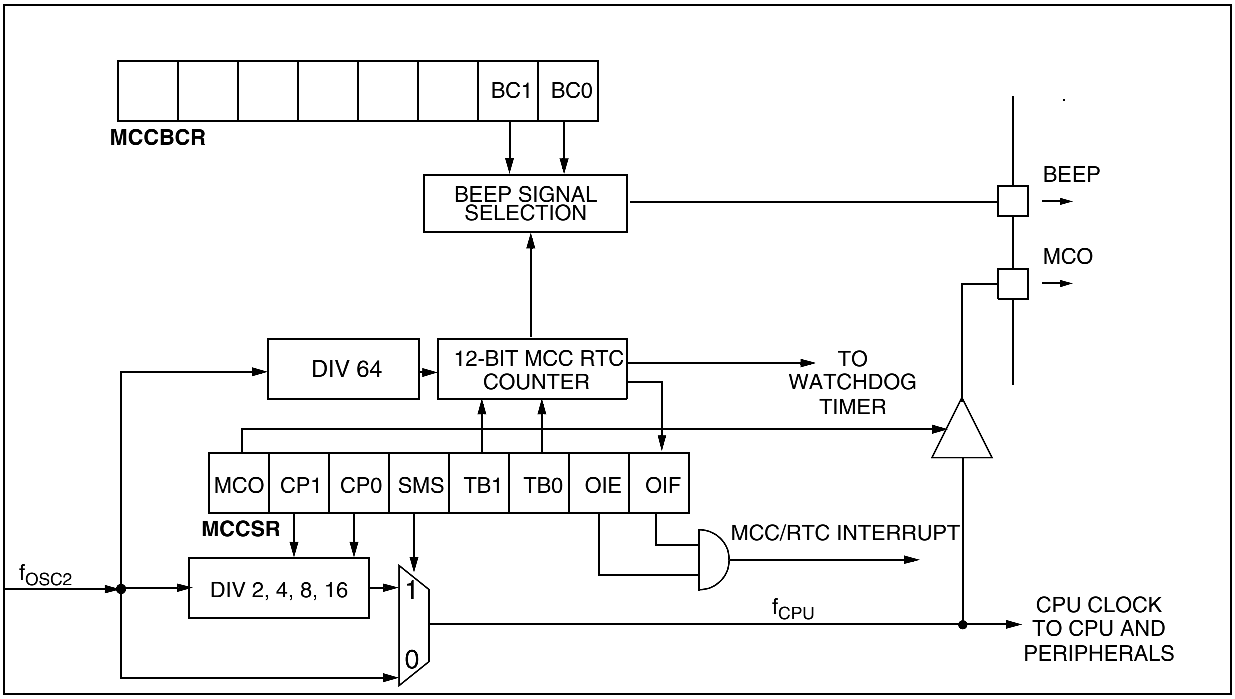
The counter of the real-time clock timer allows an interrupt to be generated based on an accurate real-time clock. Four different time bases depending directly on f_{OSC2} are available. The whole functionality is controlled by four bits of the MCCSR register: TB[1:0], OIE and OIF.

When the RTC interrupt is enabled (OIE bit set), the ST7 enters Active-halt mode when the HALT instruction is executed. See Section 8.4 ACTIVE-HALT AND HALT MODES for more details.

9.2.4 Beeper

The beep function is controlled by the MCCBCR register. It can output three selectable frequencies on the BEEP pin (I/O port alternate function).

Figure 32. Main Clock Controller (MCC/RTC) Block Diagram


BC1
BC0
MCCBCR
BEEP SIGNAL SELECTION
BEEP
MCO
DIV 64
12-BIT MCC RTC COUNTER
TO WATCHDOG TIMER
MCO
CP1
CP0
SMS
TB1
TB0
OIE
OIF
MCCSR
MCC/RTC INTERRUPT
f_{OSC2}
DIV 2, 4, 8, 16
1
0
f_{CPU}
CPU CLOCK TO CPU AND PERIPHERALS

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

9.2.5 Low Power Modes

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

9.2.6 Interrupts

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

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

Note:

The MCC/RTC interrupt wakes up the MCU from Active-halt mode, not from Halt mode.

9.2.7 Register Description

MCC CONTROL/STATUS REGISTER (MCCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
MCO	CP1	CP0	SMS	TB1	TB0	OIE	OIF

Bit 7 = **MCO** *Main clock out selection*
This bit enables the MCO alternate function on the PF0 I/O port. It is set and cleared by software.
0: MCO alternate function disabled (I/O pin free for general-purpose I/O)
1: MCO alternate function enabled (f_{CPU} on I/O port)

Note: To reduce power consumption, the MCO function is not active in Active-halt mode.

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

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

Bit 4 = **SMS** *Slow mode select*
This bit is set and cleared by software.
0: Normal mode. $f_{CPU} = f_{OSC2}$
1: Slow mode. f_{CPU} is given by CP1, CP0
See Section 8.2 SLOW MODE and Section 9.2 MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK AND BEEPER (MCC/RTC) for more details.

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

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

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

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

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

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

Bit 0 = **OIF** *Oscillator interrupt flag*
This bit is set by hardware and cleared by software reading the MCCSR register. It indicates when set that the main oscillator has reached the selected elapsed time (TB1:0).
0: Timeout not reached
1: Timeout reached

CAUTION: The BRES and BSET instructions must not be used on the MCCSR register to avoid unintentionally clearing the OIF bit.

MCC BEEP CONTROL REGISTER (MCCBCR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	BC1	BC0

Bit 7:2 = Reserved, must be kept cleared.

Bit 1:0 = **BC[1:0]** *Beep control*
These 2 bits select the PF1 pin beep capability.

BC1	BC0	Beep mode with f_{OSC2}=8 MHz	
0	0	Off	
0	1	~2 kHz	Output Beep signal ~50% duty cycle
1	0	~1 kHz	
1	1	~500 Hz	

The beep output signal is available in Active-halt mode but has to be disabled to reduce the consumption.

Table 13. Main Clock Controller Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
002Bh	**SICSR** Reset Value	AVDS 0	AVDIE 0	AVDF 0	LVDRF x	0	0	0	WDGRF x
002Ch	**MCCSR** Reset Value	MCO 0	CP1 0	CP0 0	SMS 0	TB1 0	TB0 0	OIE 0	OIF 0
002Dh	**MCCBCR** Reset Value	0	0	0	0	0	0	BC1 0	BC0 0

9.3 PWM AUTO-RELOAD TIMER (ART)

9.3.1 Introduction

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

These resources allow five possible operating modes:

– Generation of up to 4 independent PWM signals

– Output compare and Time base interrupt

– Up to two input capture functions

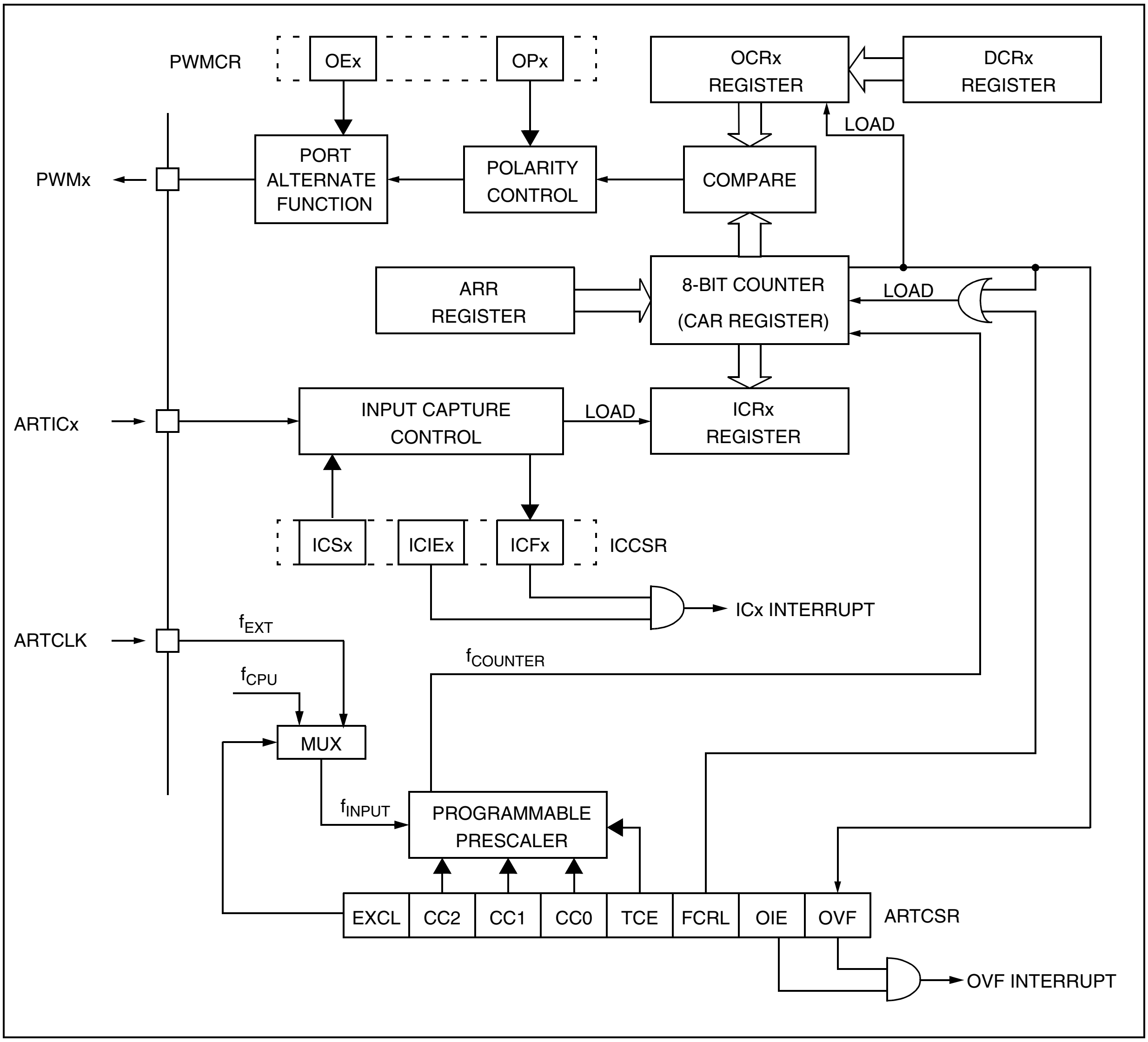
– External event detector

– Up to two external interrupt sources

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

The timer can be used to wake up the MCU from Wait and Halt modes.

Figure 33. PWM Auto-Reload Timer Block Diagram


PWMCR
OEx
OPx
OCRx REGISTER
DCRx REGISTER
LOAD
PWMx
PORT ALTERNATE FUNCTION
POLARITY CONTROL
COMPARE
ARR REGISTER
8-BIT COUNTER (CAR REGISTER)
LOAD
ARTICx
INPUT CAPTURE CONTROL
LOAD
ICRx REGISTER
ICSx
ICIEx
ICFx
ICCSR
ICx INTERRUPT
ARTCLK
f_{EXT}
f_{CPU}
$f_{COUNTER}$
MUX
f_{INPUT}
PROGRAMMABLE PRESCALER
EXCL
CC2
CC1
CC0
TCE
FCRL
OIE
OVF
ARTCSR
OVF INTERRUPT

ON-CHIP PERIPHERALS (Cont'd)

9.3.2 Functional Description

Counter

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

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

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

Counter clock and prescaler

The counter clock frequency is given by:

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

The timer counter's input clock (f_{INPUT}) feeds the 7-bit programmable prescaler, which selects one of the 8 available taps of the prescaler, as defined by CC[2:0] bits in the Control/Status Register (ARTCSR). Thus the division factor of the prescaler can be set to 2^n (where n = 0, 1,..7).

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

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

Counter and Prescaler Initialization

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

The counter can be initialized by:

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

– Writing to the ARTCAR counter access register,

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

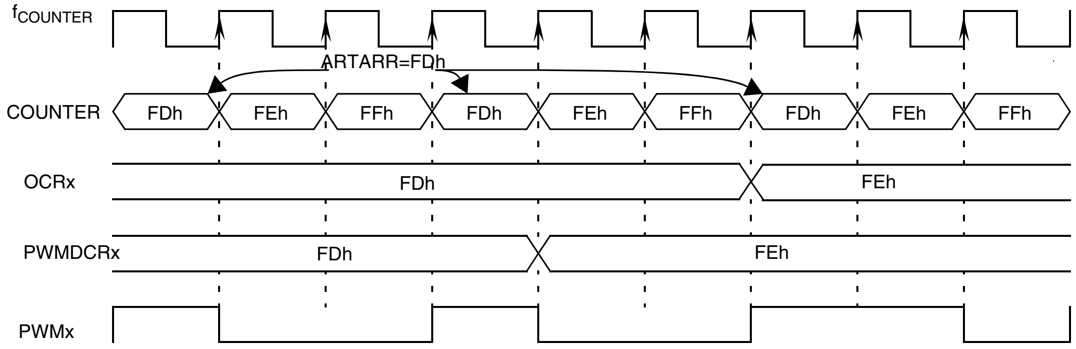
Direct access to the prescaler is not possible.

Output compare control

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

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

Figure 34. Output compare control



ON-CHIP PERIPHERALS (Cont'd)

Independent PWM signal generation

This mode allows up to four Pulse Width Modulated signals to be generated on the PWMx output pins with minimum core processing overhead. This function is stopped during HALT mode.

Each PWMx output signal can be selected independently using the corresponding OEx bit in the PWM Control register (PWMCR). When this bit is set, the corresponding I/O pin is configured as output push-pull alternate function.

The PWM signals all have the same frequency which is controlled by the counter period and the ARTARR register value.

$$f_{PWM} = f_{COUNTER} / (256 - ARTARR)$$

When a counter overflow occurs, the PWMx pin level is changed depending on the corresponding OPx (output polarity) bit in the PWMCR register.

When the counter reaches the value contained in one of the output compare register (OCRx) the corresponding PWMx pin level is restored.

It should be noted that the reload values will also affect the value and the resolution of the duty cycle of the PWM output signal. To obtain a signal on a PWMx pin, the contents of the OCRx register must be greater than the contents of the ARTARR register.

The maximum available resolution for the PWMx duty cycle is:

$$Resolution = 1 / (256 - ARTARR)$$

Note: To get the maximum resolution (1/256), the ARTARR register must be 0. With this maximum resolution, 0% and 100% can be obtained by changing the polarity.

Figure 35. PWM Auto-reload Timer Function

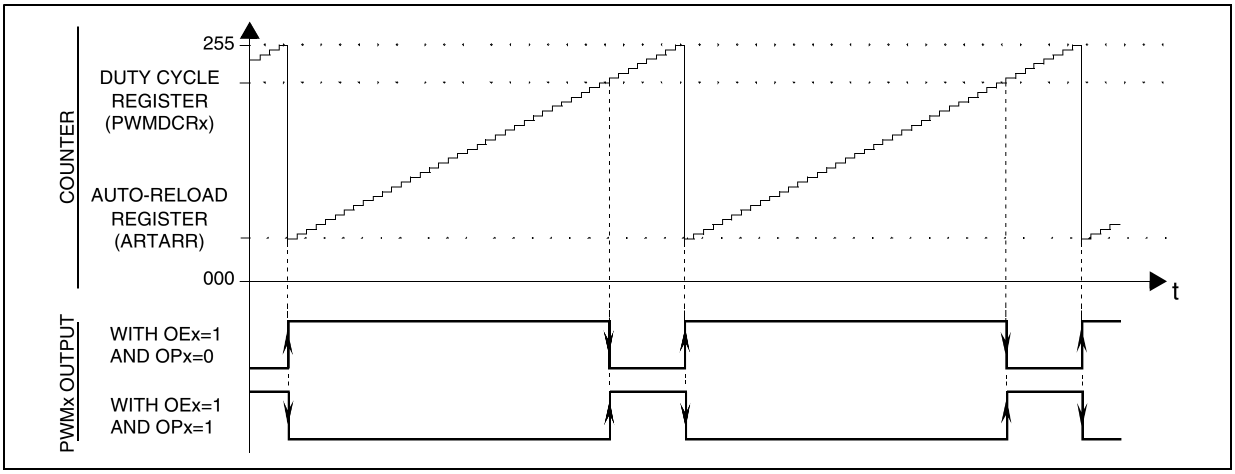
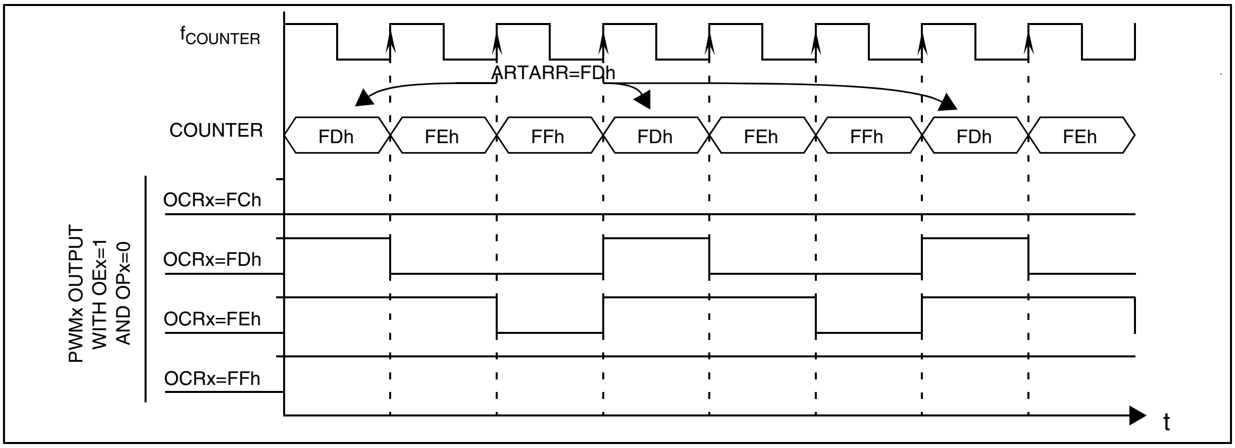

COUNTER
255
DUTY CYCLE REGISTER (PWMDCRx)
AUTO-RELOAD REGISTER (ARTARR)
000
t
PWMx OUTPUT
WITH OEx=1 AND OPx=0
WITH OEx=1 AND OPx=1

Figure 36. PWM Signal from 0% to 100% Duty Cycle


$f_{COUNTER}$
ARTARR=FDh
COUNTER
FDh
FEh
FFh
FDh
FEh
FFh
FDh
FEh
PWMx OUTPUT WITH OEx=1 AND OPx=0
OCRx=FCh
OCRx=FDh
OCRx=FEh
OCRx=FFh
t

ON-CHIP PERIPHERALS (Cont'd)

Output compare and Time base interrupt

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

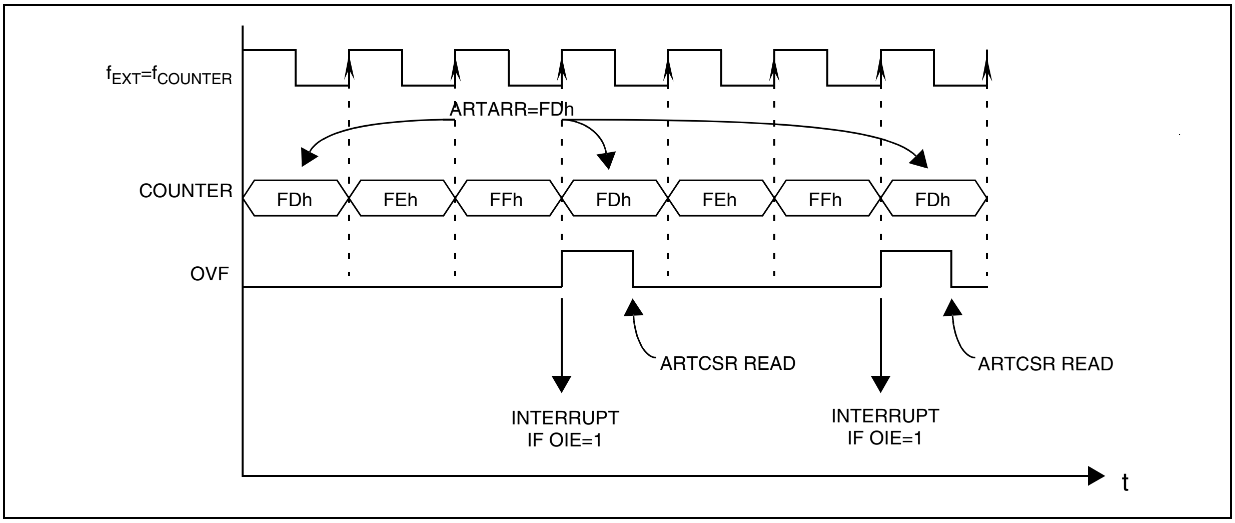
External clock and event detector mode

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

$$n_{EVENT} = 256 - ARTARR$$

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

Figure 37. External Event Detector Example (3 counts)


fEXT=fCOUNTER
ARTARR=FDh
COUNTER
FDh
FEh
FFh
FDh
FEh
FFh
FDh
OVF
ARTCSR READ
ARTCSR READ
INTERRUPT IF OIE=1
INTERRUPT IF OIE=1
t

ON-CHIP PERIPHERALS (Cont'd)

Input capture function

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

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

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

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

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

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

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

Note: During HALT mode, if both input capture and external clock are enabled, the ARTICRx register value is not guaranteed if the input capture pin and the external clock change simultaneously.

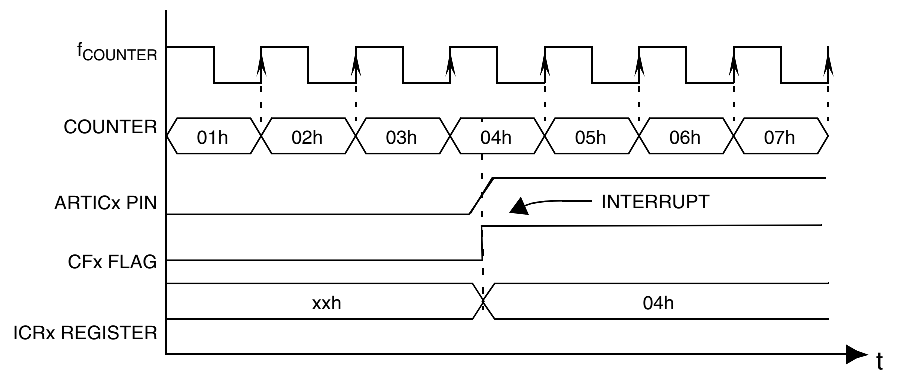
External interrupt capability

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

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

During HALT mode, the external interrupts can be used to wake up the micro (if the CIEx bit is set).

Figure 38. Input Capture Timing Diagram



ON-CHIP PERIPHERALS (Cont'd)

9.3.3 Register Description

CONTROL / STATUS REGISTER (ARTCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
EXCL	CC2	CC1	CC0	TCE	FCRL	OIE	OVF

Bit 7 = **EXCL** *External Clock*
This bit is set and cleared by software. It selects the input clock for the 7-bit prescaler.
0: CPU clock.
1: External clock.

Bit 6:4 = **CC[2:0]** *Counter Clock Control*
These bits are set and cleared by software. They determine the prescaler division ratio from f_{INPUT}.

$f_{COUNTER}$	With f_{INPUT}=8 MHz	**CC2**	**CC1**	**CC0**
f_{INPUT}	8 MHz	0	0	0
f_{INPUT} / 2	4 MHz	0	0	1
f_{INPUT} / 4	2 MHz	0	1	0
f_{INPUT} / 8	1 MHz	0	1	1
f_{INPUT} / 16	500 kHz	1	0	0
f_{INPUT} / 32	250 kHz	1	0	1
f_{INPUT} / 64	125 kHz	1	1	0
f_{INPUT} / 128	62.5 kHz	1	1	1

Bit 3 = **TCE** *Timer Counter Enable*
This bit is set and cleared by software. It puts the timer in the lowest power consumption mode.
0: Counter stopped (prescaler and counter frozen).
1: Counter running.

Bit 2 = **FCRL** *Force Counter Re-Load*
This bit is write-only and any attempt to read it will yield a logical zero. When set, it causes the contents of ARTARR register to be loaded into the counter, and the content of the prescaler register to be cleared in order to initialize the timer before starting to count.

Bit 1 = **OIE** *Overflow Interrupt Enable*
This bit is set and cleared by software. It allows to enable/disable the interrupt which is generated when the OVF bit is set.
0: Overflow Interrupt disable.
1: Overflow Interrupt enable.

Bit 0 = **OVF** *Overflow Flag*
This bit is set by hardware and cleared by software reading the ARTCSR register. It indicates the transition of the counter from FFh to the ARTARR value.
0: New transition not yet reached
1: Transition reached

COUNTER ACCESS REGISTER (ARTCAR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0

Bit 7:0 = **CA[7:0]** *Counter Access Data*

These bits can be set and cleared either by hardware or by software. The ARTCAR register is used to read or write the auto-reload counter "on the fly" (while it is counting).

AUTO-RELOAD REGISTER (ARTARR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0

Bit 7:0 = **AR[7:0]** *Counter Auto-Reload Data*

These bits are set and cleared by software. They are used to hold the auto-reload value which is automatically loaded in the counter when an overflow occurs. At the same time, the PWM output levels are changed according to the corresponding OPx bit in the PWMCR register.

This register has two PWM management functions:

- Adjusting the PWM frequency
- Setting the PWM duty cycle resolution

PWM Frequency vs Resolution:

ARTARR value	**Resolution**	f_{PWM}	
		Min	**Max**
0	8-bit	~0.244 kHz	31.25 kHz
[0..127]	> 7-bit	~0.244 kHz	62.5 kHz
[128..191]	> 6-bit	~0.488 kHz	125 kHz
[192..223]	> 5-bit	~0.977 kHz	250 kHz
[224..239]	> 4-bit	~1.953 kHz	500 kHz

ON-CHIP PERIPHERALS (Cont'd)

PWM CONTROL REGISTER (PWMCR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OE3	OE2	OE1	OE0	OP3	OP2	OP1	OP0

Bit 7:4 = **OE[3:0]** *PWM Output Enable*
These bits are set and cleared by software. They enable or disable the PWM output channels independently acting on the corresponding I/O pin.
0: PWM output disabled.
1: PWM output enabled.

Bit 3:0 = **OP[3:0]** *PWM Output Polarity*
These bits are set and cleared by software. They independently select the polarity of the four PWM output signals.

PWMx output level		OPx
Counter <= OCRx	**Counter > OCRx**	
1	0	0
0	1	1

Note: When an OPx bit is modified, the PWMx output signal polarity is immediately reversed.

DUTY CYCLE REGISTERS (PWMDCRx)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0

Bit 7:0 = **DC[7:0]** *Duty Cycle Data*

These bits are set and cleared by software.

A PWMDCRx register is associated with the OCRx register of each PWM channel to determine the second edge location of the PWM signal (the first edge location is common to all channels and given by the ARTARR register). These PWMDCR registers allow the duty cycle to be set independently for each PWM channel.

ON-CHIP PERIPHERALS (Cont'd)

INPUT CAPTURE CONTROL / STATUS REGISTER (ARTICCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	CS2	CS1	CIE2	CIE1	CF2	CF1

Bit 7:6 = Reserved, always read as 0.

Bit 5:4 = **CS[2:1]** *Capture Sensitivity*
These bits are set and cleared by software. They determine the trigger event polarity on the corresponding input capture channel.
0: Falling edge triggers capture on channel x.
1: Rising edge triggers capture on channel x.

Bit 3:2 = **CIE[2:1]** *Capture Interrupt Enable*
These bits are set and cleared by software. They enable or disable the Input capture channel interrupts independently.
0: Input capture channel x interrupt disabled.
1: Input capture channel x interrupt enabled.

Bit 1:0 = **CF[2:1]** *Capture Flag*
These bits are set by hardware and cleared by software reading the corresponding ARTICRx register. Each CFx bit indicates that an input capture x has occurred.
0: No input capture on channel x.
1: An input capture has occurred on channel x.

INPUT CAPTURE REGISTERS (ARTICRx)

Read only

Reset Value: 0000 0000 (00h)

7							0
IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0

Bit 7:0 = **IC[7:0]** *Input Capture Data*

These read only bits are set and cleared by hardware. An ARTICRx register contains the 8-bit auto-reload counter value transferred by the input capture channel x event.

PWM AUTO-RELOAD TIMER (Cont'd)

Table 14. PWM Auto-Reload Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0073h	**PWMDCR3** Reset Value	DC7 0	DC6 0	DC5 0	DC4 0	DC3 0	DC2 0	DC1 0	DC0 0
0074h	**PWMDCR2** Reset Value	DC7 0	DC6 0	DC5 0	DC4 0	DC3 0	DC2 0	DC1 0	DC0 0
0075h	**PWMDCR1** Reset Value	DC7 0	DC6 0	DC5 0	DC4 0	DC3 0	DC2 0	DC1 0	DC0 0
0076h	**PWMDCR0** Reset Value	DC7 0	DC6 0	DC5 0	DC4 0	DC3 0	DC2 0	DC1 0	DC0 0
0077h	**PWMCR** Reset Value	OE3 0	OE2 0	OE1 0	OE0 0	OP3 0	OP2 0	OP1 0	OP0 0
0078h	**ARTCSR** Reset Value	EXCL 0	CC2 0	CC1 0	CC0 0	TCE 0	FCRL 0	RIE 0	OVF 0
0079h	**ARTCAR** Reset Value	CA7 0	CA6 0	CA5 0	CA4 0	CA3 0	CA2 0	CA1 0	CA0 0
007Ah	**ARTARR** Reset Value	AR7 0	AR6 0	AR5 0	AR4 0	AR3 0	AR2 0	AR1 0	AR0 0
007Bh	**ARTICCSR** Reset Value	0	0	CS2 0	CS1 0	CIE2 0	CIE1 0	CF2 0	CF1 0
007Ch	**ARTICR1** Reset Value	IC7 0	IC6 0	IC5 0	IC4 0	IC3 0	IC2 0	IC1 0	IC0 0
007Dh	**ARTICR2** Reset Value	IC7 0	IC6 0	IC5 0	IC4 0	IC3 0	IC2 0	IC1 0	IC0 0

9.4 16-BIT TIMER

9.4.1 Introduction

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

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

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

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

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

9.4.2 Main Features

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

The Block Diagram is shown in Figure 39.

***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’.

9.4.3 Functional Description

9.4.3.1 Counter

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

Counter Register (CR):

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

Alternate Counter Register (ACR)

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

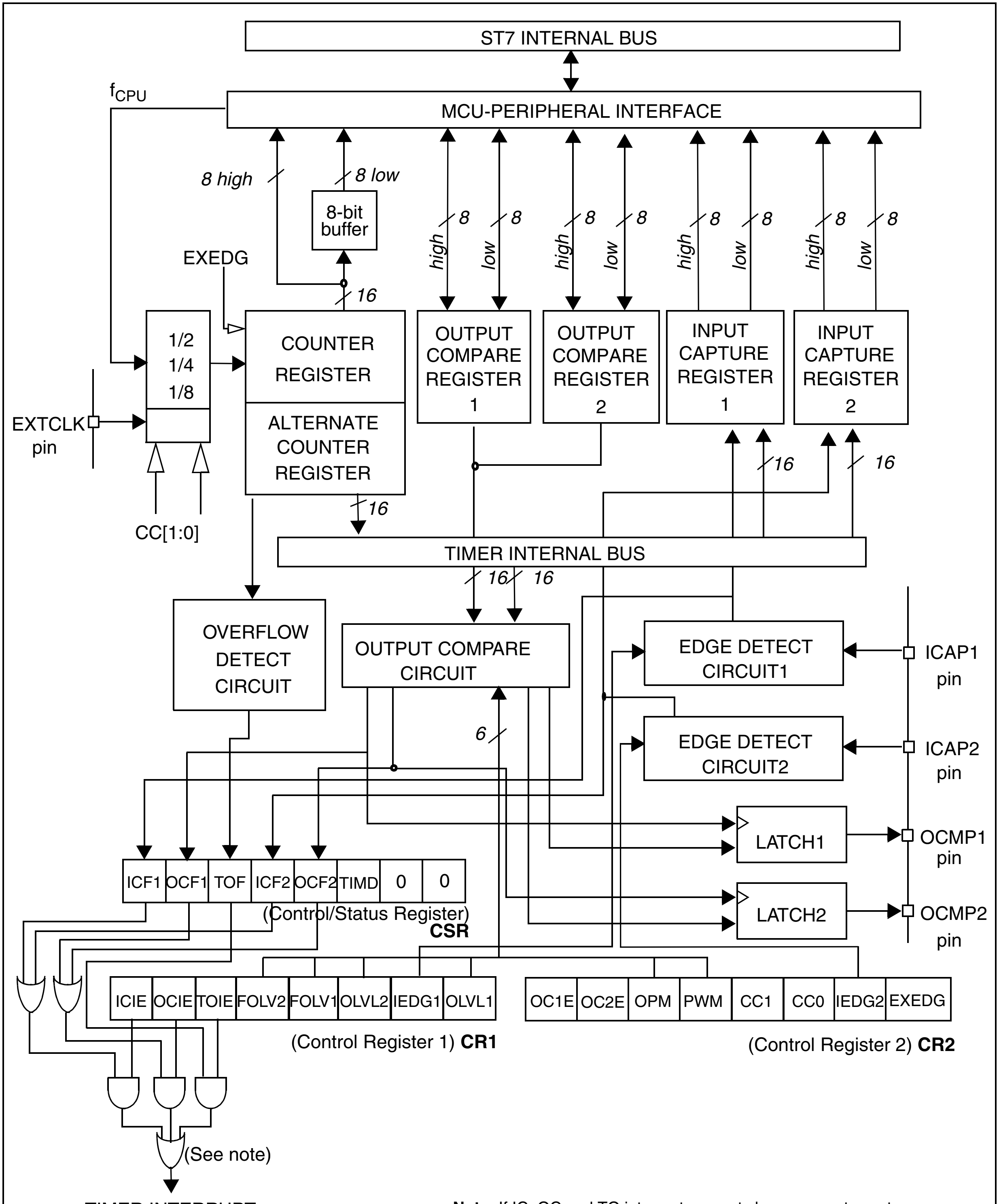
These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR), (see note at the end of paragraph titled 16-bit read sequence).

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

The timer clock depends on the clock control bits of the CR2 register, as illustrated in Table 15 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)

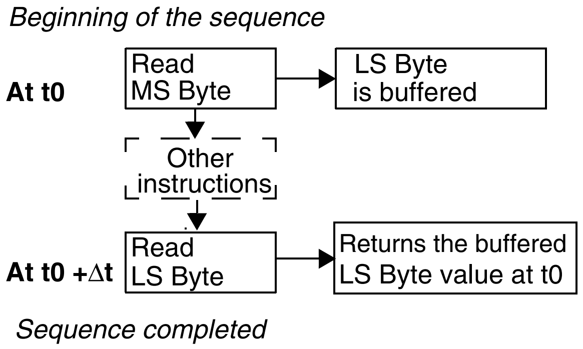
Figure 39. Timer Block Diagram


ST7 INTERNAL BUS
MCU-PERIPHERAL INTERFACE
f_{CPU}
8 high
8 low
8-bit buffer
EXEDG
16
high 8
low 8
high 8
low 8
high 8
low 8
high 8
low 8
1/2
1/4
1/8
COUNTER REGISTER
ALTERNATE COUNTER REGISTER
OUTPUT COMPARE REGISTER 1
OUTPUT COMPARE REGISTER 2
INPUT CAPTURE REGISTER 1
INPUT CAPTURE REGISTER 2
EXTCLK pin
CC[1:0]
16
16
16
TIMER INTERNAL BUS
16
16
OVERFLOW DETECT CIRCUIT
OUTPUT COMPARE CIRCUIT
EDGE DETECT CIRCUIT1
EDGE DETECT CIRCUIT2
ICAP1 pin
ICAP2 pin
6
LATCH1
LATCH2
OCMP1 pin
OCMP2 pin
ICF1
OCF1
TOF
ICF2
OCF2
TIMD
0
0
(Control/Status Register) CSR
ICIE
OCIE
TOIE
FOLV2
FOLV1
OLVL2
IEDG1
OLVL1
OC1E
OC2E
OPM
PWM
CC1
CC0
IEDG2
EXEDG
(Control Register 1) CR1
(Control Register 2) CR2
(See note)
TIMER INTERRUPT

Note: If IC, OC and TO interrupt requests have separate vectors then the last OR is not present (See device Interrupt Vector Table)

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.

Notes: The TOF bit is not cleared by accesses to 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).

9.4.3.2 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 synchronized 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 40. Counter Timing Diagram, Internal Clock Divided by 2

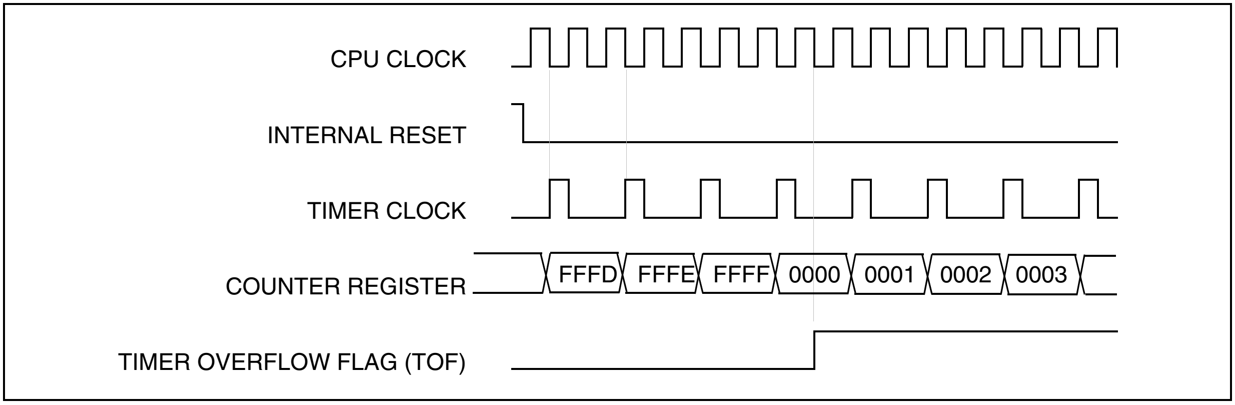

CPU CLOCK
INTERNAL RESET
TIMER CLOCK
COUNTER REGISTER
FFFD
FFFE
FFFF
0000
0001
0002
0003
TIMER OVERFLOW FLAG (TOF)

Figure 41. Counter Timing Diagram, Internal Clock Divided by 4

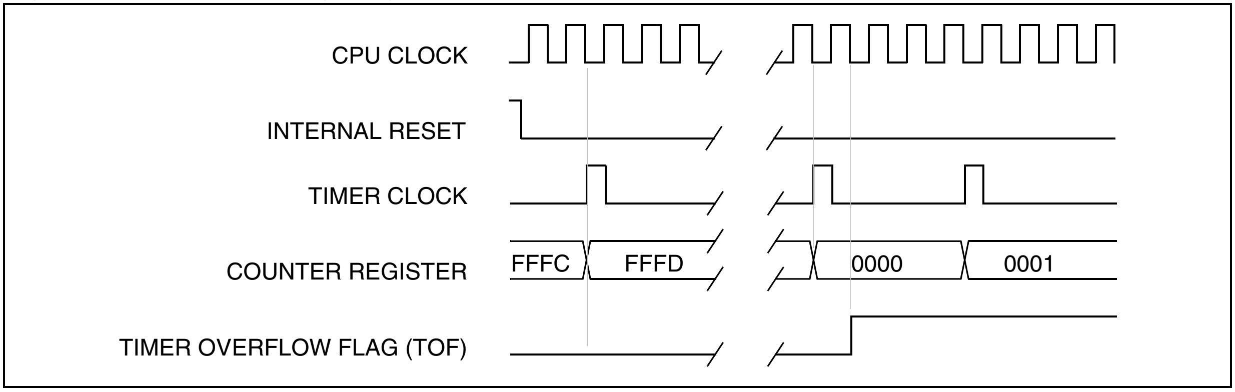
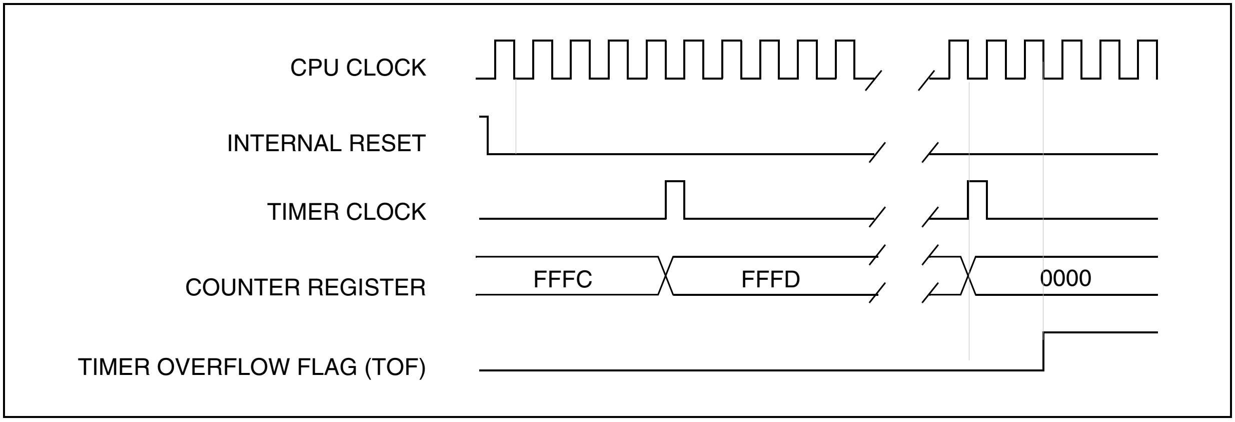

CPU CLOCK
INTERNAL RESET
TIMER CLOCK
COUNTER REGISTER
FFFC
FFFD
0000
0001
TIMER OVERFLOW FLAG (TOF)

Figure 42. Counter Timing Diagram, Internal Clock Divided By 8


CPU CLOCK
INTERNAL RESET
TIMER CLOCK
COUNTER REGISTER
FFFC
FFFD
0000
TIMER OVERFLOW FLAG (TOF)

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)

9.4.3.3 Input Capture

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

The two 16-bit input capture registers (IC1R and IC2R) are used to latch the value of the free running counter after a transition is detected on the ICAP*i* pin (see Figure 43).

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

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 15 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 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 ICAP1pin must be configured as floating input or input with pull-up without interrupt if this configuration is available).

When an input capture occurs:

- 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 44).
- 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 (that is, clearing the ICF*i* bit) is done in two steps:

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

Notes:

1. After reading the IC*i*HR register, 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 two input capture functions can be used together even if the timer also uses the two output compare functions.
4. In One Pulse mode and PWM mode only Input Capture 2 can be used.
5. The alternate inputs (ICAP1 and ICAP2) are always directly connected to the timer. So any transitions on these pins activates the input capture function.
   Moreover if one of the ICAP*i* pins 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 interrupt generation in order to measure events that go beyond the timer range (FFFFh).

16-BIT TIMER (Cont'd)

Figure 43. Input Capture Block Diagram

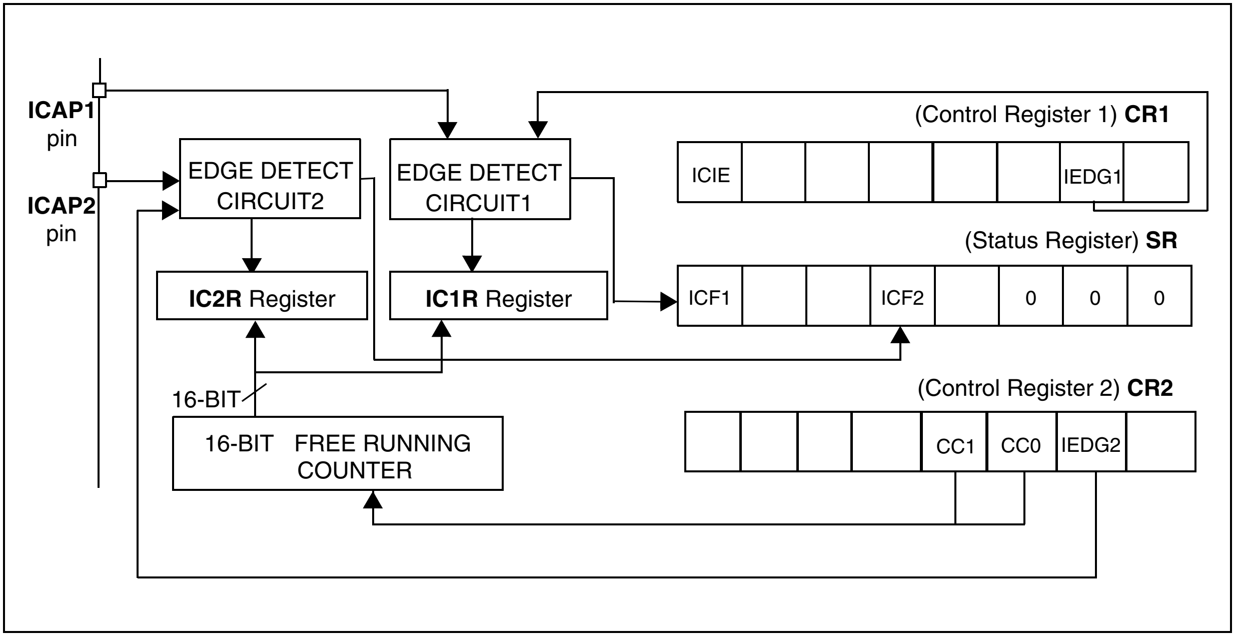
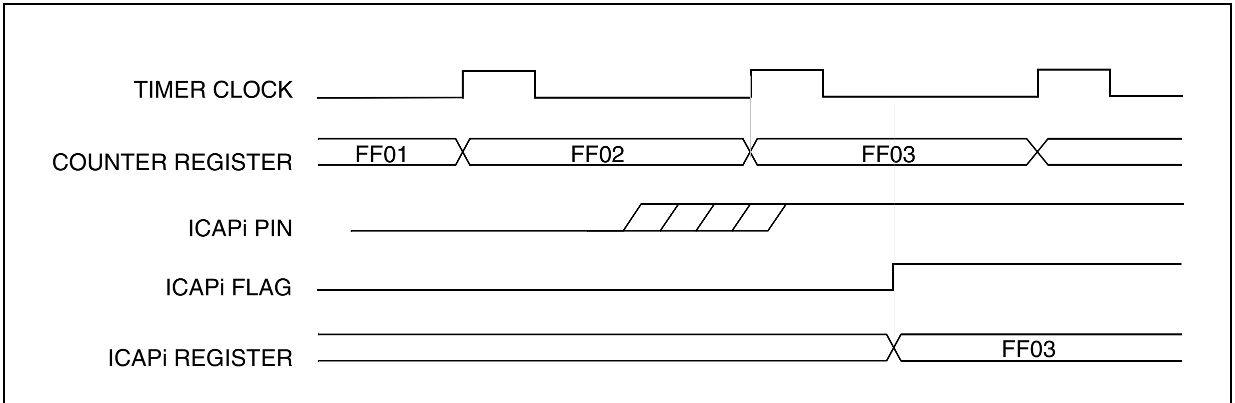

ICAP1
pin
ICAP2
pin
EDGE DETECT CIRCUIT2
EDGE DETECT CIRCUIT1
IC2R Register
IC1R Register
16-BIT
16-BIT FREE RUNNING COUNTER
(Control Register 1) CR1
ICIE
IEDG1
(Status Register) SR
ICF1
ICF2
0
0
0
(Control Register 2) CR2
CC1
CC0
IEDG2

Figure 44. Input Capture Timing Diagram


TIMER CLOCK
COUNTER REGISTER
FF01
FF02
FF03
ICAPi PIN
ICAPi FLAG
ICAPi REGISTER
FF03

Note: The rising edge is the active edge.

16-BIT TIMER (Cont'd)

9.4.3.4 Output Compare

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

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

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

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

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

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

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

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

Procedure:

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

- Set the OC*i*E bit if an output is needed then the OCMP*i* pin is dedicated to the output compare *i* signal.
- Select the timer clock (CC[1:0]) (see Table 15 Clock Control Bits).

And select the following in the CR1 register:

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

When a match is found between OC*i*R register and CR register:

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

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

$$\Delta \, OCiR = \frac{\Delta t * f_{CPU}}{PRESC}$$

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 15 Clock Control Bits)

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

$$\Delta \, OCiR = \Delta t * f_{EXT}$$

Where:

Δt = Output compare period (in seconds)

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

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

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

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

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

16-BIT TIMER (Cont'd)

Notes:

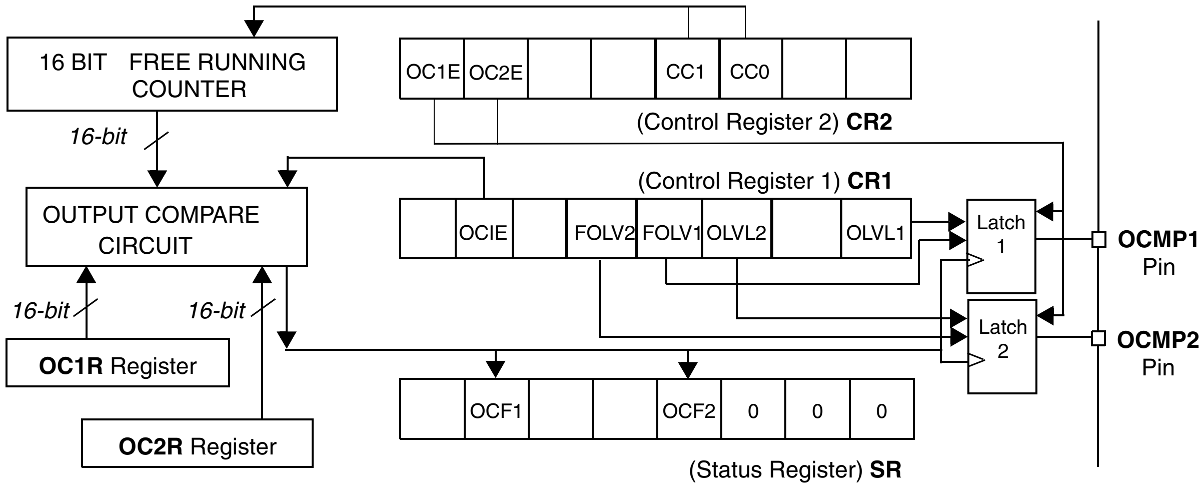
1. After a processor write cycle to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.
2. If the OC*i*E bit is not set, the OCMP*i* pin is a general I/O port and the OLVL*i* bit will not appear when a match is found but an interrupt could be generated if the OCIE bit is set.
3. In both internal and external clock modes, OCFi and OCMPi are set while the counter value equals the OCiR register value (see Figure 46 on page 73 for an example with $f_{CPU}/2$ and Figure 47 on page 73 for an example with $f_{CPU}/4$). This behavior is the same in OPM or PWM mode.
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.

The FOLVL*i* bits have no effect in both One Pulse mode and PWM mode.

Figure 45. Output Compare Block Diagram


16 BIT FREE RUNNING COUNTER
16-bit
OUTPUT COMPARE CIRCUIT
16-bit
16-bit
OC1R Register
OC2R Register
OC1E
OC2E
CC1
CC0
(Control Register 2) CR2
(Control Register 1) CR1
OCIE
FOLV2
FOLV1
OLVL2
OLVL1
Latch 1
Latch 2
OCMP1 Pin
OCMP2 Pin
OCF1
OCF2
0
0
0
(Status Register) SR

16-BIT TIMER (Cont'd)

Figure 46. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/2$

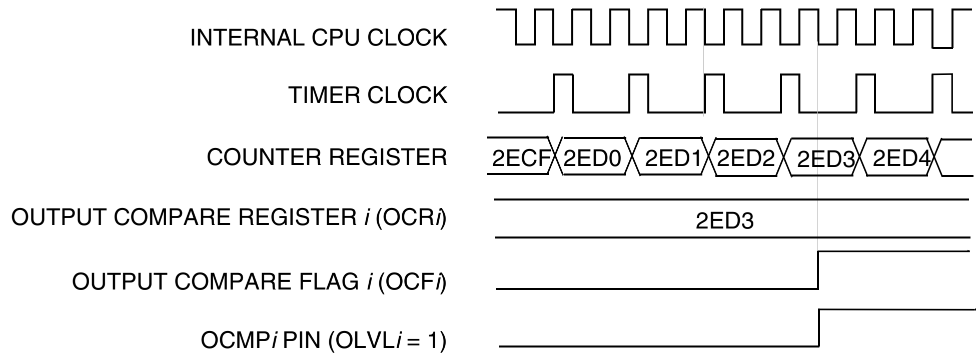
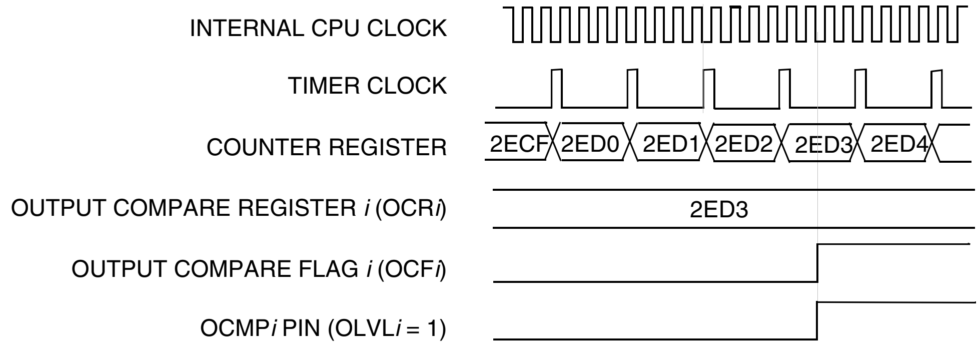

INTERNAL CPU CLOCK
TIMER CLOCK
COUNTER REGISTER
2ECF
2ED0
2ED1
2ED2
2ED3
2ED4
OUTPUT COMPARE REGISTER *i* (OCR*i*)
2ED3
OUTPUT COMPARE FLAG *i* (OCF*i*)
OCMP*i* PIN (OLVL*i* = 1)

Figure 47. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/4$


INTERNAL CPU CLOCK
TIMER CLOCK
COUNTER REGISTER
2ECF
2ED0
2ED1
2ED2
2ED3
2ED4
OUTPUT COMPARE REGISTER *i* (OCR*i*)
2ED3
OUTPUT COMPARE FLAG *i* (OCF*i*)
OCMP*i* PIN (OLVL*i* = 1)

16-BIT TIMER (Cont'd)

9.4.3.5 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.

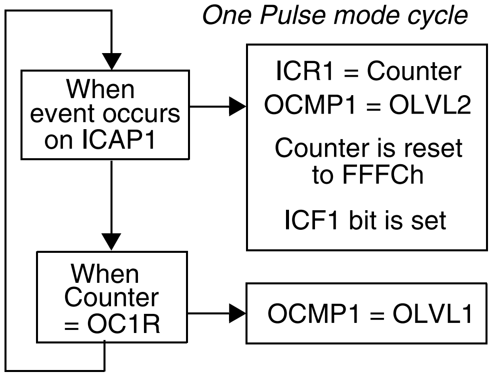
The One Pulse mode uses the Input Capture1 function and the Output Compare1 function.

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 15 Clock Control Bits).

One Pulse mode cycle



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

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

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

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

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

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

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 15 Clock Control Bits)

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

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

Where:

t = Pulse period (in seconds)

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

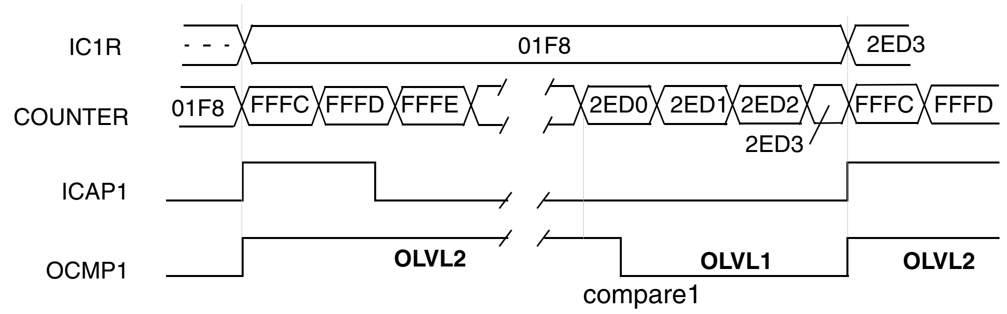
When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin, (See Figure 48).

Notes:

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

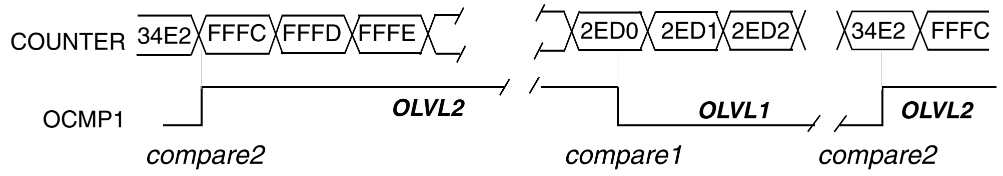
16-BIT TIMER (Cont'd)

Figure 48. One Pulse Mode Timing Example


IC1R
- - -
01F8
2ED3
COUNTER
01F8
FFFC
FFFD
FFFE
2ED0
2ED1
2ED2
2ED3
FFFC
FFFD
ICAP1
OCMP1
OLVL2
OLVL1
OLVL2
compare1

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

Figure 49. Pulse Width Modulation Mode Timing Example with 2 Output Compare Functions


COUNTER
34E2
FFFC
FFFD
FFFE
2ED0
2ED1
2ED2
34E2
FFFC
OCMP1
OLVL2
OLVL1
OLVL2
compare2
compare1
compare2

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

Note: On timers with only one Output Compare register, a fixed frequency PWM signal can be generated using the output compare and the counter overflow to define the pulse length.

16-BIT TIMER (Cont'd)

9.4.3.6 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.

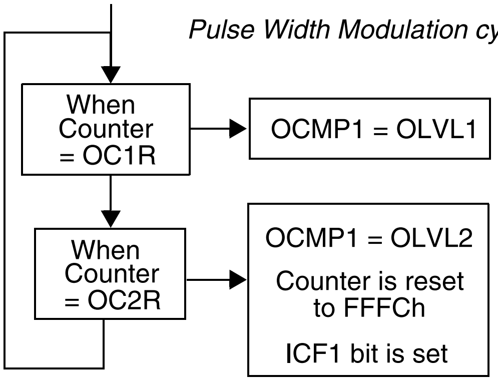
Pulse Width Modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

In PWM mode, double buffering is implemented on the output compare registers. Any new values written in the OC1R and OC2R registers are taken into account only at the end of the PWM period (OC2) to avoid spikes on the PWM output pin (OCMP1).

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 the OC1R register.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with the 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 15 Clock Control Bits).



Pulse Width Modulation cycle

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:

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

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 15 Clock Control Bits)

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

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

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 49)

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 to 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 each period and ICF1 can also generates 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)

9.4.4 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.

9.4.5 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			
Output Compare 1 event (not available in PWM mode)	OCF1	OCIE		
Output Compare 2 event (not available in PWM mode)	OCF2			
Timer Overflow event	TOF	TOIE		

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).

9.4.6 Summary of Timer Modes

MODES	TIMER 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)				
One Pulse Mode	No	Not Recommended 1)	No	Partially 2)
PWM Mode		Not Recommended 3)		No

1) See note 4 in Section 9.4.3.5 One Pulse Mode

2) See note 5 in Section 9.4.3.5 One Pulse Mode

3) See note 4 in Section 9.4.3.6 Pulse Width Modulation Mode

16-BIT TIMER (Cont'd)

9.4.7 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 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 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.

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

The timer clock mode depends on these bits:

Table 15. Clock Control Bits

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

CONTROL/STATUS REGISTER (CSR)

Read/Write (bits 7:3 read only)

Reset Value: xxxx x0xx (xxh)

7							0
ICF1	OCF1	TOF	ICF2	OCF2	TIMD	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 has matched 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 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 has matched 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 = **TIMD** *Timer disable.*
This bit is set and cleared by software. When set, it freezes the timer prescaler and counter and disabled the output functions (OCMP1 and OCMP2 pins) to reduce power consumption. Access to the timer registers is still available, allowing the timer configuration to be changed, or the counter reset, while it is disabled.
0: Timer enabled
1: Timer prescaler, counter and outputs disabled

Bits 1:0 = Reserved, must be kept cleared.

16-BIT TIMER (Cont'd)

INPUT CAPTURE 1 HIGH REGISTER (IC1HR)

Read Only
Reset Value: Undefined

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

7							0
MSB							LSB

INPUT CAPTURE 1 LOW REGISTER (IC1LR)

Read Only
Reset Value: Undefined

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

7							0
MSB							LSB

OUTPUT COMPARE 1 HIGH REGISTER (OC1HR)

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

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

7							0
MSB							LSB

OUTPUT COMPARE 1 LOW REGISTER (OC1LR)

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

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

7							0
MSB							LSB

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 CSR 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 CSR register does not clear the TOF bit in the CSR 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 16. 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	**CSR** Reset Value	ICF1 x	OCF1 x	TOF x	ICF2 x	OCF2 x	TIMD 0	- x	- x
Timer A: 34 Timer B: 44	**IC1HR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
Timer A: 35 Timer B: 45	**IC1LR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
Timer A: 36 Timer B: 46	**OC1HR** Reset Value	MSB 1	0	0	0	0	0	0	LSB 0
Timer A: 37 Timer B: 47	**OC1LR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0
Timer A: 3E Timer B: 4E	**OC2HR** Reset Value	MSB 1	0	0	0	0	0	0	LSB 0
Timer A: 3F Timer B: 4F	**OC2LR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0
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	**IC2HR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
Timer A: 3D Timer B: 4D	**IC2LR** Reset Value	MSB x	x	x	x	x	x	x	LSB x

Related Documentation

AN 973: SCI software communications using 16-bit timer

AN 974: Real-Time Clock with ST7 Timer Output Compare

AN 976: Driving a buzzer through the ST7 Timer PWM function

AN1041: Using ST7 PWM signal to generate analog input (sinusoid)

AN1046: UART emulation software

AN1078: PWM duty cycle switch implementing true 0 or 100 per cent duty cycle

AN1504: Starting a PWM signal directly at high level using the ST7 16-Bit timer

9.5 SERIAL PERIPHERAL INTERFACE (SPI)

9.5.1 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 however the SPI interface can not be a master in a multi-master system.

9.5.2 Main Features

- Full duplex synchronous transfers (on 3 lines)
- Simplex synchronous transfers (on 2 lines)
- Master or slave operation
- Six master mode frequencies (f_{CPU}/4 max.)
- f_{CPU}/2 max. slave mode frequency (see note)
- $\overline{SS}$ Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

9.5.3 General Description

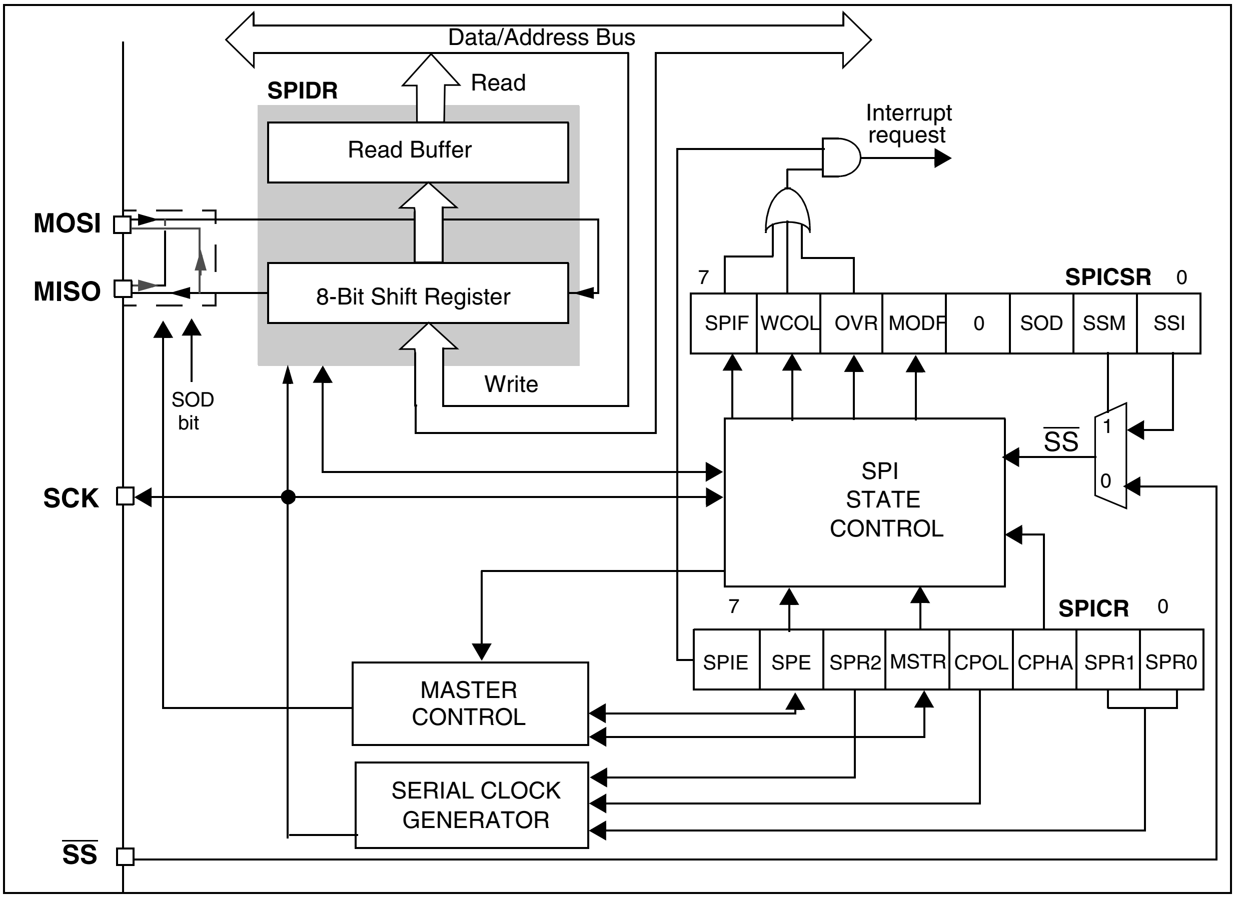
Figure 50 shows the serial peripheral interface (SPI) block diagram. There are 3 registers:

- SPI Control Register (SPICR)
- SPI Control/Status Register (SPICSR)
- SPI Data Register (SPIDR)

The SPI is connected to external devices through 4 pins:

- MISO: Master In / Slave Out data
- MOSI: Master Out / Slave In data
- SCK: Serial Clock out by SPI masters and input by SPI slaves

Figure 50. Serial Peripheral Interface Block Diagram


Data/Address Bus
SPIDR
Read
Read Buffer
MOSI
MISO
8-Bit Shift Register
Write
SOD
bit
SCK
$\overline{SS}$
Interrupt
request
7
SPICSR
0
SPIF
WCOL
OVR
MODF
0
SOD
SSM
SSI
1
0
$\overline{SS}$
SPI
STATE
CONTROL
7
SPICR
0
SPIE
SPE
SPR2
MSTR
CPOL
CPHA
SPR1
SPR0
MASTER
CONTROL
SERIAL CLOCK
GENERATOR

SERIAL PERIPHERAL INTERFACE (Cont'd)

– $\overline{SS}$: Slave select:
  This input signal acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave $\overline{SS}$ inputs can be driven by standard I/O ports on the master MCU.

9.5.3.1 Functional Description

A basic example of interconnections between a single master and a single slave is illustrated in Figure 51.

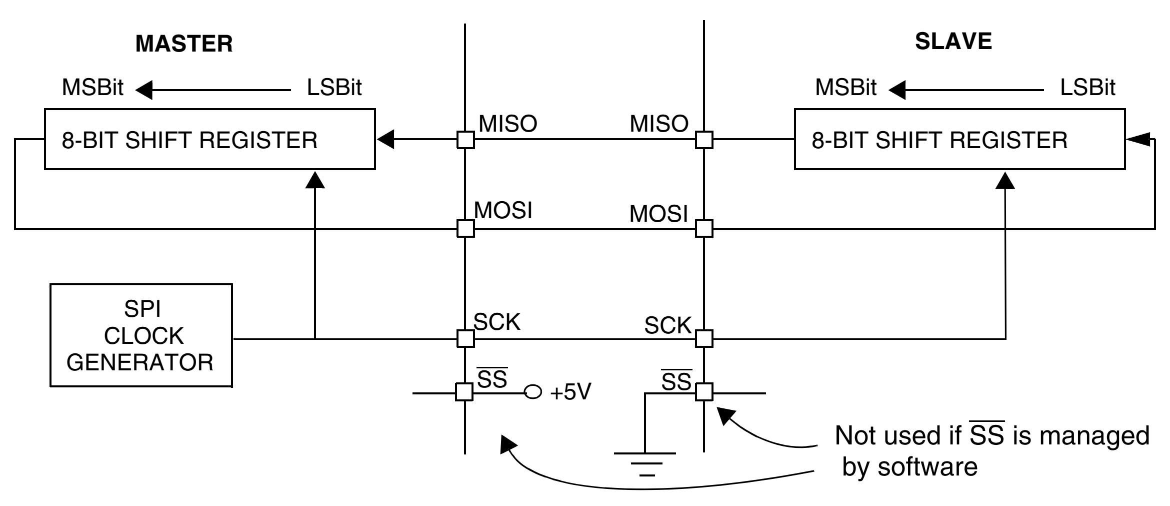
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

The communication is always initiated by the master. 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 communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

To use a single data line, the MISO and MOSI pins must be connected at each node (in this case only simplex communication is possible).

Four possible data/clock timing relationships may be chosen (see Figure 54) but master and slave must be programmed with the same timing mode.

Figure 51. Single Master/ Single Slave Application


MASTER
MSBit
LSBit
8-BIT SHIFT REGISTER
SPI CLOCK GENERATOR
MISO
MOSI
SCK
$\overline{SS}$
+5V
MISO
MOSI
SCK
$\overline{SS}$
SLAVE
MSBit
LSBit
8-BIT SHIFT REGISTER

Not used if $\overline{SS}$ is managed by software

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.3.2 Slave Select Management

As an alternative to using the $\overline{SS}$ pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 53)

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

In Master mode:

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

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 52):

If CPHA=1 (data latched on 2nd clock edge):

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

If CPHA=0 (data latched on 1st clock edge):

– $\overline{SS}$ internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If $\overline{SS}$ is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 9.5.5.3).

Figure 52. Generic $\overline{SS}$ Timing Diagram

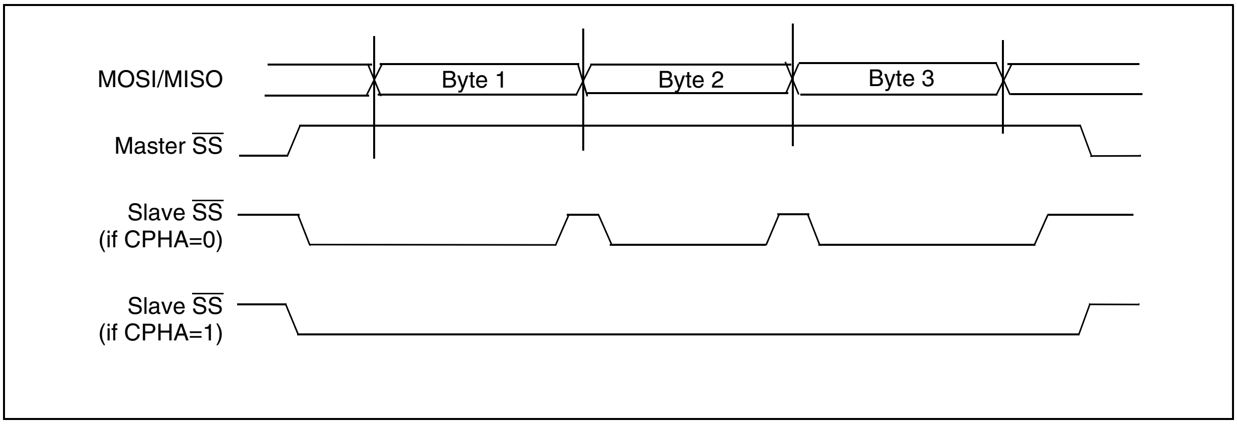
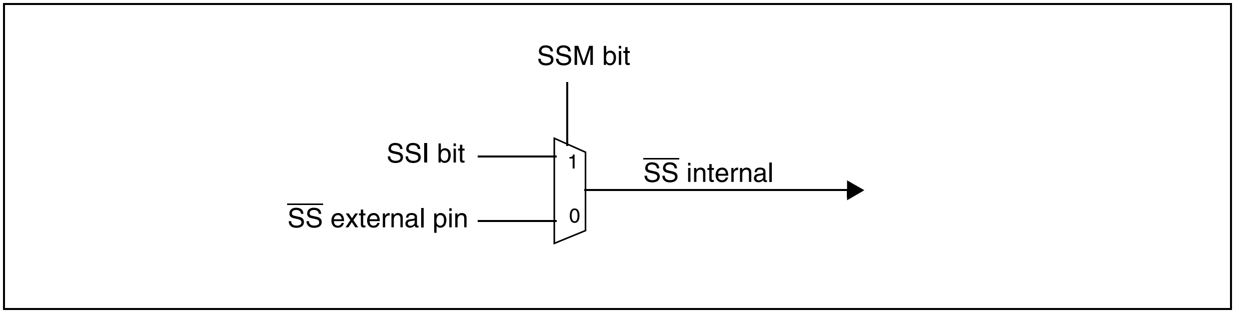


Figure 53. Hardware/Software Slave Select Management



SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.3.3 Master Mode Operation

In master mode, the serial clock is output on the SCK pin. The clock frequency, polarity and phase are configured by software (refer to the description of the SPICSR register).

Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

To operate the SPI in master mode, perform the following steps in order (if the SPICSR register is not written first, the SPICR register setting (MSTR bit) may be not taken into account):

1. Write to the SPICR register:
   - Select the clock frequency by configuring the SPR[2:0] bits.
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits. Figure 54 shows the four possible configurations.
     Note: The slave must have the same CPOL and CPHA settings as the master.
2. Write to the SPICSR register:
   - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the $\overline{\text{SS}}$ pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
   - Set the MSTR and SPE bits
     Note: MSTR and SPE bits remain set only if $\overline{\text{SS}}$ is high).

The transmit sequence begins when software writes a byte in the SPIDR register.

9.5.3.4 Master Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register 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 request is generated if the SPIE bit is set and the interrupt mask in the CCR register is cleared.

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

1. An access to the SPICSR register while the SPIF bit is set
2. A read to the SPIDR register.

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

9.5.3.5 Slave Mode Operation

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

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see Figure 54).
     Note: The slave must have the same CPOL and CPHA settings as the master.
   - Manage the $\overline{\text{SS}}$ pin as described in Section 9.5.3.2 and Figure 52. If CPHA=1 $\overline{\text{SS}}$ must be held low continuously. If CPHA=0 $\overline{\text{SS}}$ must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

9.5.3.6 Slave Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register 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 request is generated if SPIE bit is set and interrupt mask in the CCR register is cleared.

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

1. An access to the SPICSR register while the SPIF bit is set.
2. A write or a read to the SPIDR register.

Notes: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR 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 9.5.5.2).

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.4 Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 54).

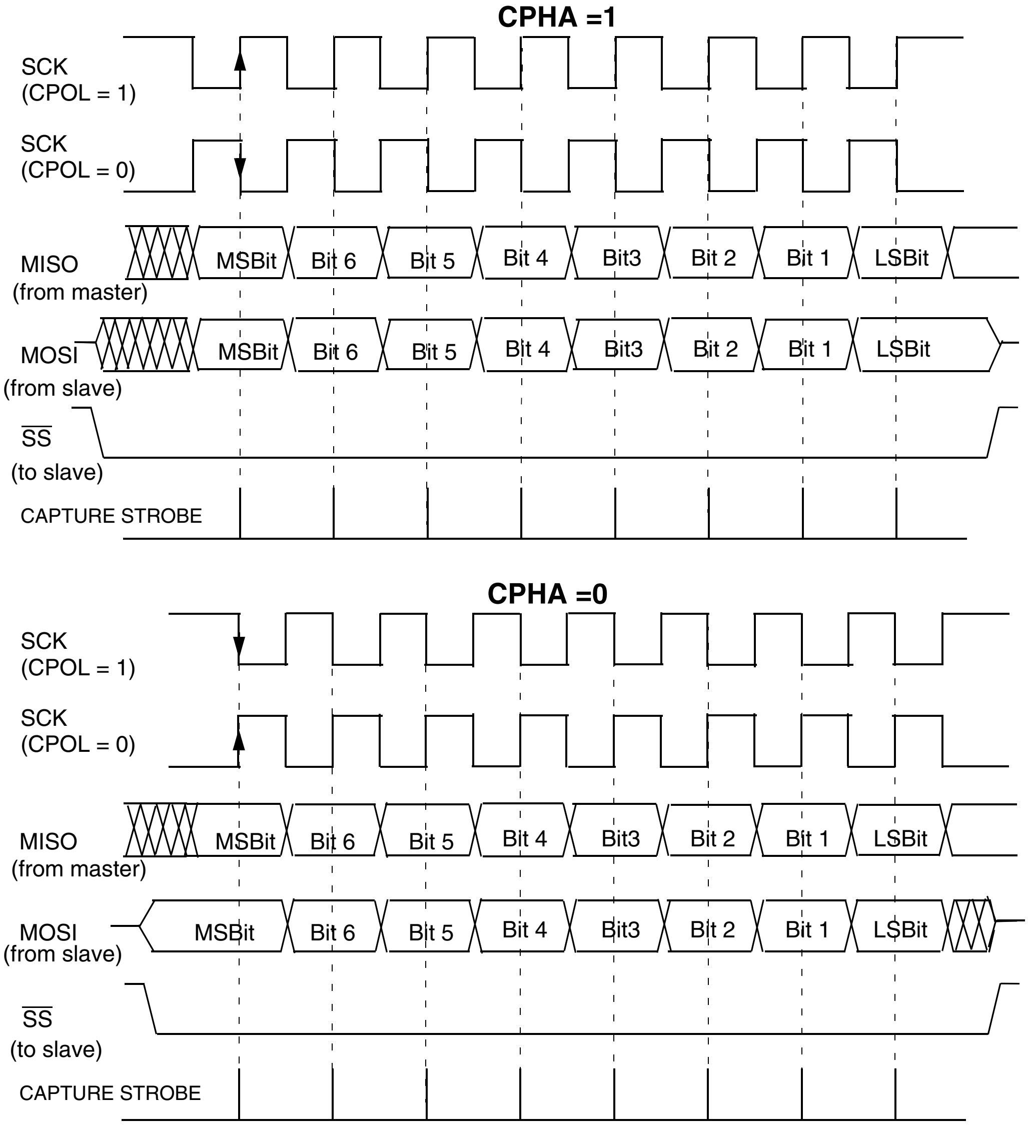
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

The combination of the CPOL clock polarity and CPHA (clock phase) bits selects the data capture clock edge

Figure 54, 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.

Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.

Figure 54. Data Clock Timing Diagram


CPHA =1
SCK
(CPOL = 1)
SCK
(CPOL = 0)
MISO
(from master)
MSBit
Bit 6
Bit 5
Bit 4
Bit3
Bit 2
Bit 1
LSBit
MOSI
(from slave)
MSBit
Bit 6
Bit 5
Bit 4
Bit3
Bit 2
Bit 1
LSBit
SS
(to slave)
CAPTURE STROBE
CPHA =0
SCK
(CPOL = 1)
SCK
(CPOL = 0)
MISO
(from master)
MSBit
Bit 6
Bit 5
Bit 4
Bit3
Bit 2
Bit 1
LSBit
MOSI
(from slave)
MSBit
Bit 6
Bit 5
Bit 4
Bit3
Bit 2
Bit 1
LSBit
SS
(to slave)
CAPTURE STROBE

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

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.5 Error Flags

9.5.5.1 Master Mode Fault (MODF)

Master mode fault occurs when the master device has its $\overline{SS}$ pin pulled low.

When a Master mode fault occurs:

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

Clearing the MODF bit is done through a software sequence:

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

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

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

9.5.5.2 Overrun Condition (OVR)

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

When an Overrun occurs:

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

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

The OVR bit is cleared by reading the SPICSR register.

9.5.5.3 Write Collision Error (WCOL)

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

Write collisions can occur both in master and slave mode. See also Section 9.5.3.2 Slave Select Management.

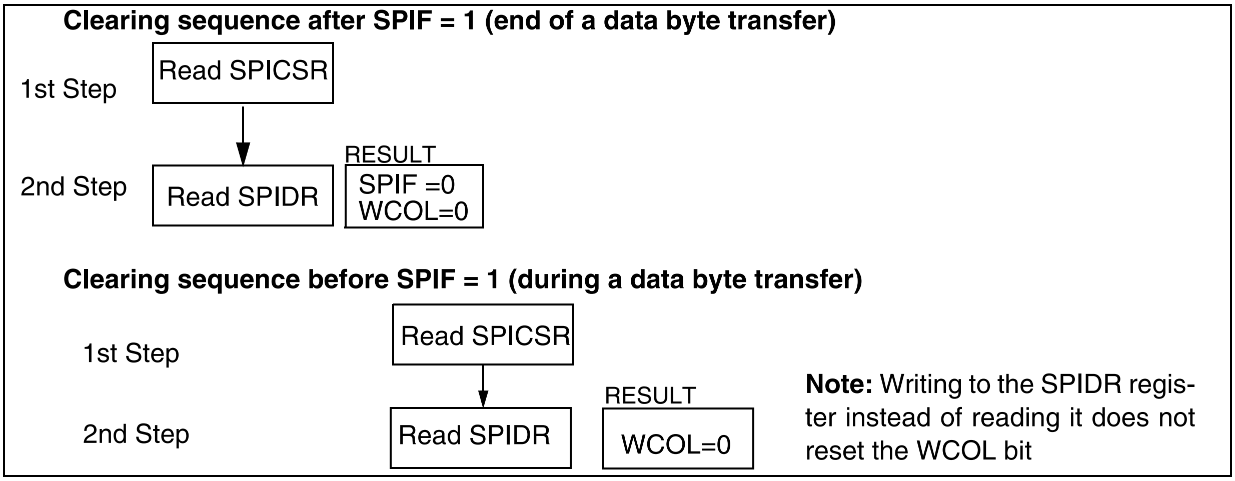
Note: a "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation.

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

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

Clearing the WCOL bit is done through a software sequence (see Figure 55).

Figure 55. Clearing the WCOL bit (Write Collision Flag) Software Sequence


Clearing sequence after SPIF = 1 (end of a data byte transfer)
1st Step
Read SPICSR
2nd Step
Read SPIDR
RESULT
SPIF =0
WCOL=0
Clearing sequence before SPIF = 1 (during a data byte transfer)
1st Step
Read SPICSR
2nd Step
Read SPIDR
RESULT
WCOL=0
Note: Writing to the SPIDR register instead of reading it does not reset the WCOL bit

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.5.4 Single Master Systems

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see Figure 56).

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.

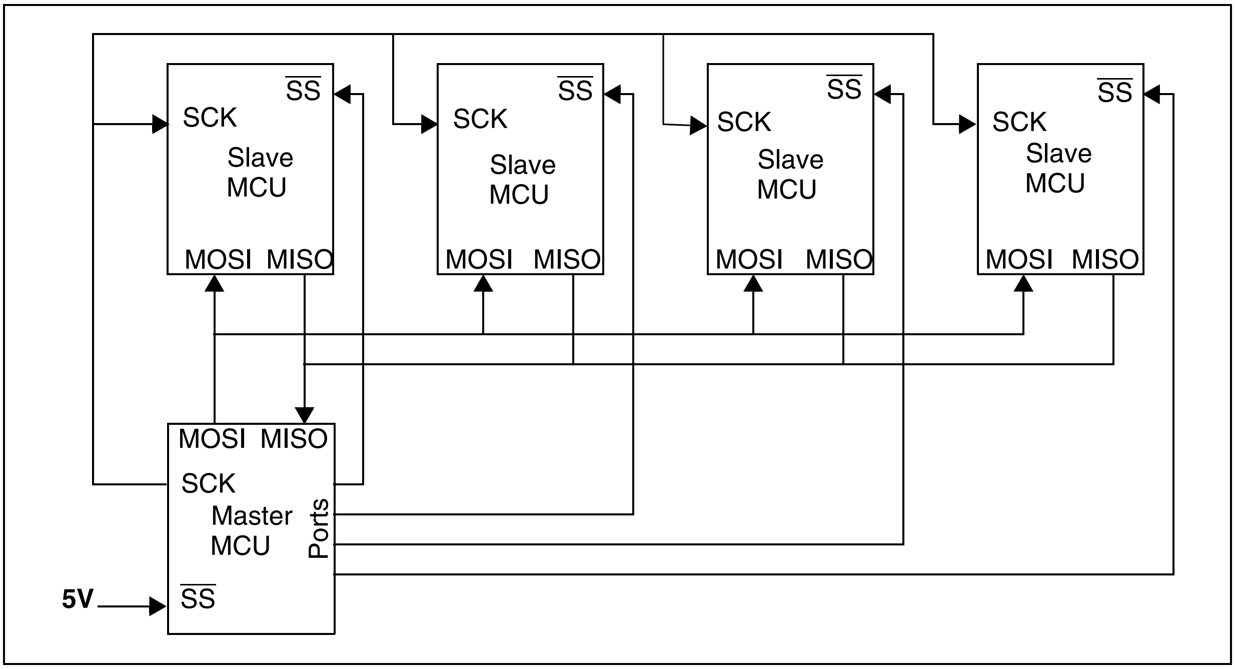
The $\overline{SS}$ pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

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 to its SPIDR register.

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

Figure 56. Single Master / Multiple Slave Configuration


$\overline{SS}$
SCK
Slave MCU
MOSI MISO
$\overline{SS}$
SCK
Slave MCU
MOSI MISO
$\overline{SS}$
SCK
Slave MCU
MOSI MISO
$\overline{SS}$
SCK
Slave MCU
MOSI MISO
MOSI MISO
SCK
Master MCU
Ports
5V
$\overline{SS}$

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.6 Low Power Modes

Mode	Description
Wait	No effect on SPI. SPI interrupt events cause the device to exit from Wait mode.
Halt	SPI registers are frozen. In Halt mode, the SPI is inactive. SPI operation resumes when the MCU is woken up by an interrupt with "exit from Halt mode" capability. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetching). If several data are received before the wake-up event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the device.

9.5.6.1 Using the SPI to wakeup the MCU from Halt mode

In slave configuration, the SPI is able to wakeup the ST7 device from Halt mode through a SPIF interrupt. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetch). If multiple data transfers have been performed before software clears the SPIF bit, then the OVR bit is set by hardware.

Note: When waking up from Halt mode, if the SPI remains in Slave mode, it is recommended to perform an extra communications cycle to bring the SPI from Halt mode state to normal state. If the SPI exits from Slave mode, it returns to normal state immediately.

Caution: The SPI can wake up the ST7 from Halt mode only if the Slave Select signal (external $\overline{\text{SS}}$ pin or the SSI bit in the SPICSR register) is low when the ST7 enters Halt mode. So if Slave selection is configured as external (see Section 9.5.3.2), make sure the master drives a low level on the $\overline{\text{SS}}$ pin when the slave enters Halt mode.

9.5.7 Interrupts

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

SERIAL PERIPHERAL INTERFACE (Cont'd)

9.5.8 Register Description

CONTROL REGISTER (SPICR)

Read/Write

Reset Value: 0000 xxxx (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, MODF=1 or OVR=1 in the SPICSR 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, $\overline{SS}$=0 (see Section 9.5.5.1 Master Mode Fault (MODF)). The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.
0: I/O pins free for general purpose I/O
1: SPI I/O pin alternate functions enabled

Bit 5 = **SPR2** *Divider Enable*.
This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 17 SPI Master mode SCK Frequency.
0: Divider by 2 enabled
1: Divider by 2 disabled

Note: This bit has no effect in slave mode.

Bit 4 = **MSTR** *Master Mode.*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$=0 (see Section 9.5.5.1 Master Mode Fault (MODF)).
0: Slave mode
1: Master mode. 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 idle state of the serial Clock. The CPOL bit affects both the master and slave modes.
0: SCK pin has a low level idle state
1: SCK pin has a high level idle state

Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.

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.

Note: The slave must have the same CPOL and CPHA settings as the master.

Bits 1:0 = **SPR[1:0]** *Serial Clock Frequency.*
These bits are set and cleared by software. Used with the SPR2 bit, they select the baud rate of the SPI serial clock SCK output by the SPI in master mode.

Note: These 2 bits have no effect in slave mode.

Table 17. SPI Master mode SCK Frequency

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)

CONTROL/STATUS REGISTER (SPICSR)

Read/Write (some bits Read Only)
Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI

Bit 7 = **SPIF** *Serial Peripheral Data Transfer Flag (Read only).*
This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE=1 in the SPICR register. It is cleared by a software sequence (an access to the SPICSR register followed by a write or a read to the SPIDR register).
0: Data transfer is in progress or the flag has been cleared.
1: Data transfer between the device and an external device has been completed.

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

Bit 6 = **WCOL** *Write Collision status (Read only).*
This bit is set by hardware when a write to the SPIDR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 55).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = **OVR** S*PI Overrun error (Read only).*
This bit is set by hardware when the byte currently being received in the shift register is ready to be transferred into the SPIDR register while SPIF = 1 (See Section 9.5.5.2). An interrupt is generated if SPIE = 1 in SPICR register. The OVR bit is cleared by software reading the SPICSR register.
0: No overrun error
1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only).*
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 9.5.5.1 Master Mode Fault (MODF)). An SPI interrupt can be generated if SPIE=1 in the SPICSR register. This bit is cleared by a software sequence (An access to the SPICR register while MODF=1 followed by a write to the SPICR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **SOD** *SPI Output Disable.*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode / MISO in slave mode)
0: SPI output enabled (if SPE=1)
1: SPI output disabled

Bit 1 = **SSM** *$\overline{SS}$ Management.*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI $\overline{SS}$ pin and uses the SSI bit value instead. See Section 9.5.3.2 Slave Select Management.
0: Hardware management ($\overline{SS}$ managed by external pin)
1: Software management (internal $\overline{SS}$ signal controlled by SSI bit. External $\overline{SS}$ pin free for general-purpose I/O)

Bit 0 = **SSI** *$\overline{SS}$ Internal Mode.*
This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the $\overline{SS}$ slave select signal when the SSM bit is set.
0 : Slave selected
1 : Slave deselected

DATA I/O REGISTER (SPIDR)

Read/Write
Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The SPIDR register is used to transmit and receive data on the serial bus. In a master device, 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.

While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Warning: A write to the SPIDR register places data directly into the shift register for transmission.

A read to the SPIDR register returns the value located in the buffer and not the content of the shift register (see Figure 50).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 18. SPI Register Map and Reset Values

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

9.6 SERIAL COMMUNICATIONS INTERFACE (SCI)

9.6.1 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.

9.6.2 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 500 Kbaud
- 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
- Separate enable bits for Transmitter and Receiver
- Four error detection flags:
  - Overrun error
  - Noise error
  - Frame error
  - Parity error
- Five interrupt sources with flags:
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error detected
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode

9.6.3 General Description

The interface is externally connected to another device by two pins (see Figure 58):

- TDO: Transmit Data Output. When the transmitter and the receiver are disabled, the output pin returns to its I/O port configuration. When the transmitter and/or the receiver are enabled and nothing is to be transmitted, the TDO pin is at high level.
- RDI: Receive Data Input is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

Through these pins, serial data is transmitted and received as frames comprising:

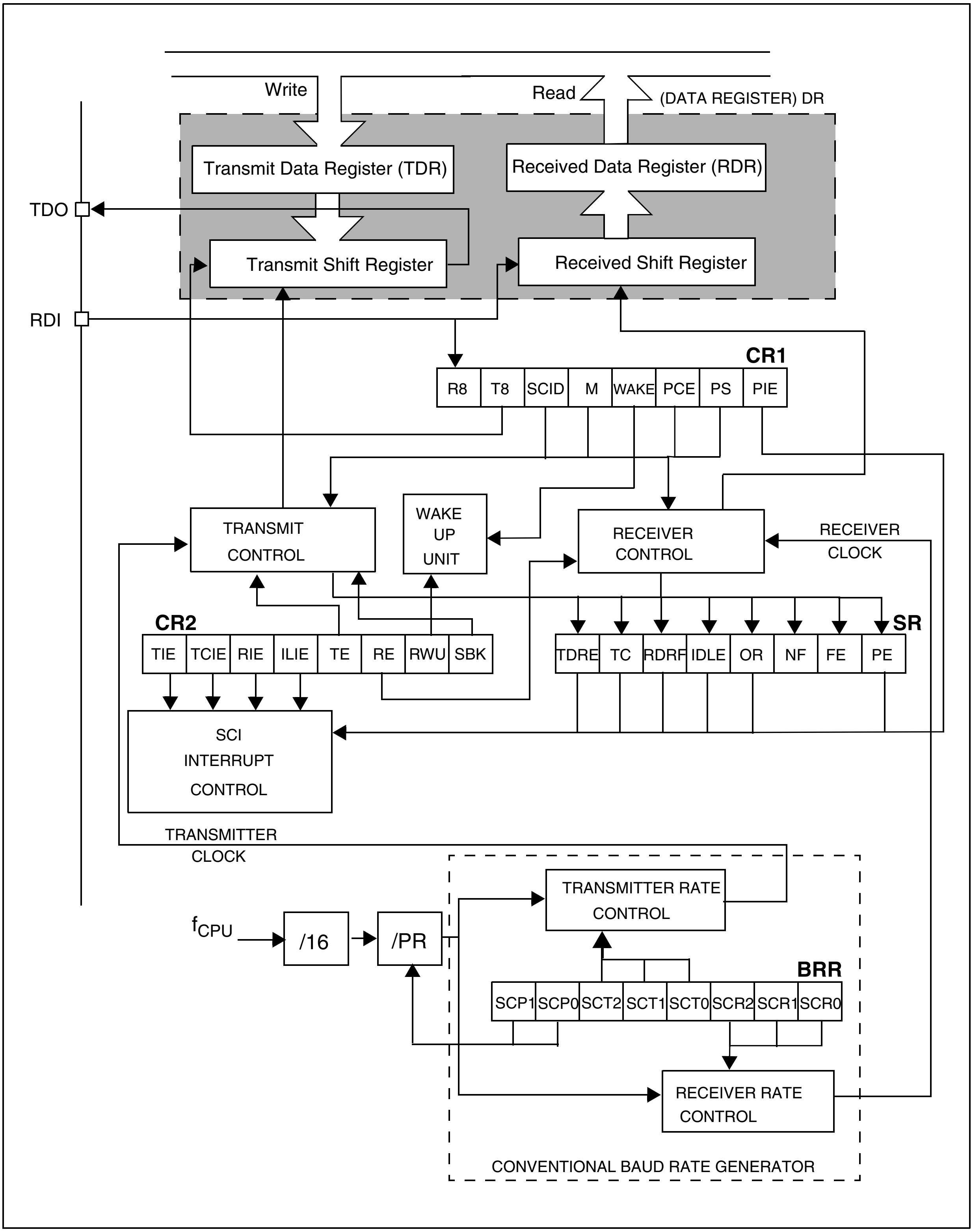
- An Idle Line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) least significant bit first
- A Stop bit indicating that the frame is complete

This interface uses two types of baud rate generator:

- A conventional type for commonly-used baud rates
- An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 57. SCI Block Diagram


Write
Read
(DATA REGISTER) DR
Transmit Data Register (TDR)
Received Data Register (RDR)
TDO
Transmit Shift Register
Received Shift Register
RDI
CR1
R8
T8
SCID
M
WAKE
PCE
PS
PIE
TRANSMIT CONTROL
WAKE UP UNIT
RECEIVER CONTROL
RECEIVER CLOCK
CR2
TIE
TCIE
RIE
ILIE
TE
RE
RWU
SBK
SR
TDRE
TC
RDRF
IDLE
OR
NF
FE
PE
SCI INTERRUPT CONTROL
TRANSMITTER CLOCK
TRANSMITTER RATE CONTROL
f_CPU
/16
/PR
BRR
SCP1
SCP0
SCT2
SCT1
SCT0
SCR2
SCR1
SCR0
RECEIVER RATE CONTROL
CONVENTIONAL BAUD RATE GENERATOR

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.4 Functional Description

The block diagram of the Serial Control Interface, is shown in Figure 57 It contains six dedicated registers:

– Two control registers (SCICR1 & SCICR2)

– A status register (SCISR)

– A baud rate register (SCIBRR)

– An extended prescaler receiver register (SCIERPR)

– An extended prescaler transmitter register (SCIETPR)

Refer to the register descriptions in Section 9.6.7for the definitions of each bit.

9.6.4.1 Serial Data Format

Word length may be selected as being either 8 or 9 bits by programming the M bit in the SCICR1 register (see Figure 57).

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

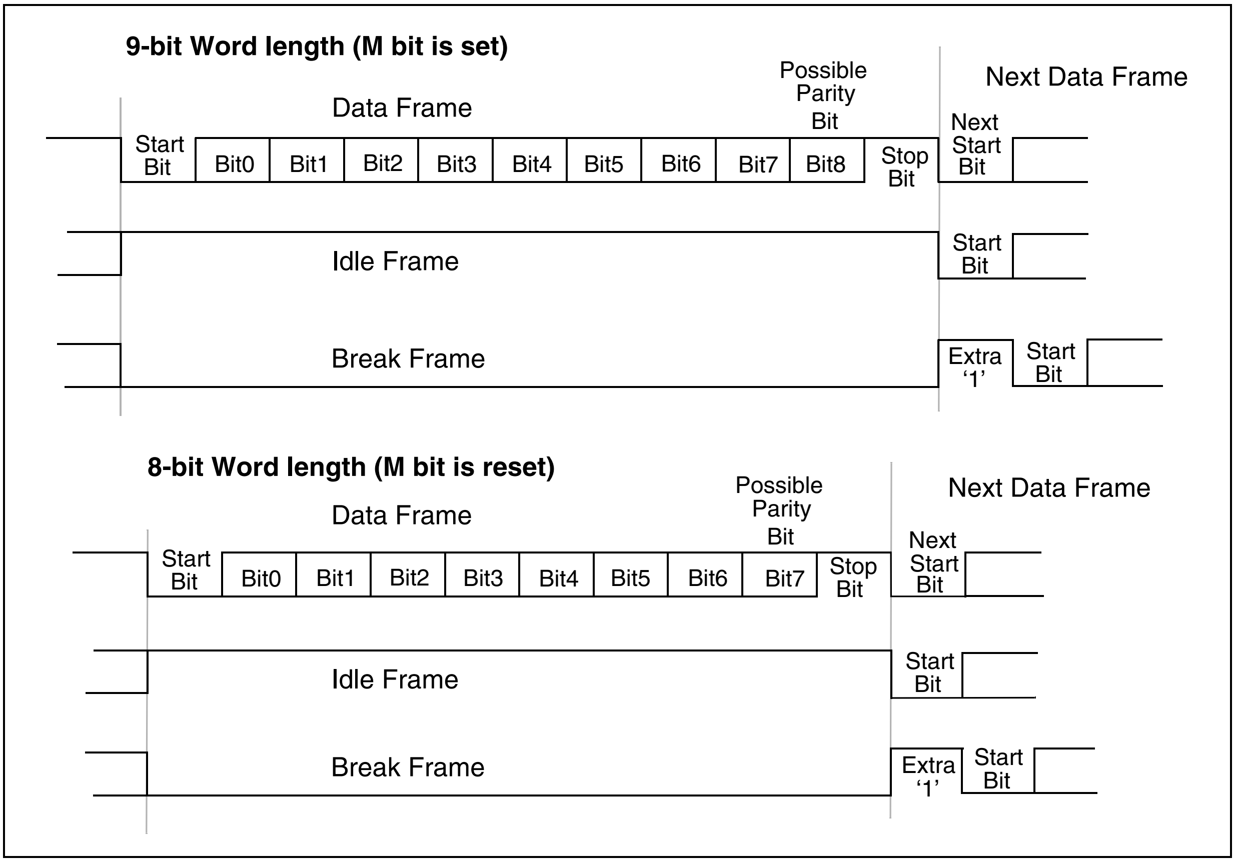
The TDO pin is in high state during the stop bit.

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 58. Word Length Programming


9-bit Word length (M bit is set)
Data Frame
Possible Parity Bit
Next Data Frame
Start Bit
Bit0
Bit1
Bit2
Bit3
Bit4
Bit5
Bit6
Bit7
Bit8
Stop Bit
Next Start Bit
Idle Frame
Start Bit
Break Frame
Extra ‘1’
Start Bit
8-bit Word length (M bit is reset)
Data Frame
Possible Parity Bit
Next Data Frame
Start Bit
Bit0
Bit1
Bit2
Bit3
Bit4
Bit5
Bit6
Bit7
Stop Bit
Next Start Bit
Idle Frame
Start Bit
Break Frame
Extra ‘1’
Start Bit

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.4.2 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 SCICR1 register.

Character Transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 57).

Procedure

- Select the M bit to define the word length.
- Select the desired baud rate using the SCIBRR and the SCIETPR 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 SCISR register and write the data to send in the SCIDR 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 SCISR register
2. A write to the SCIDR 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 SCIDR 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 SCIDR 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 SCIDR 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) 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 SCISR register
2. A write to the SCIDR 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 58).

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, that is, before writing the next byte in the SCIDR.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.4.3 Receiver

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

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists or a buffer (RDR) between the internal bus and the received shift register (see Figure 57).

Procedure

- Select the M bit to define the word length.
- Select the desired baud rate using the SCIBRR and the SCIERPR registers.
- Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

- The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR.
- An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.
- The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

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

1. An access to the SCISR register
2. A read to the SCIDR register.

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

Break Character

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

Idle Character

When 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 RDR register as long as the RDRF bit is not cleared.

When an overrun error occurs:

- The OR bit is set.
- The RDR content is not lost.
- The shift register is overwritten.
- An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

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

Noise Error

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise. Normal data bits are considered valid if three consecutive samples (8th, 9th, 10th) have the same bit value, otherwise the NF flag is set. In the case of start bit detection, the NF flag is set on the basis of an algorithm combining both valid edge detection and three samples (8th, 9th, 10th). Therefore, to prevent the NF flag getting set during start bit reception, there should be a valid edge detection as well as three valid samples.

When noise is detected in a frame:

- The NF flag is set at the rising edge of the RDRF bit.
- Data is transferred from the Shift register to the SCIDR register.
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

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

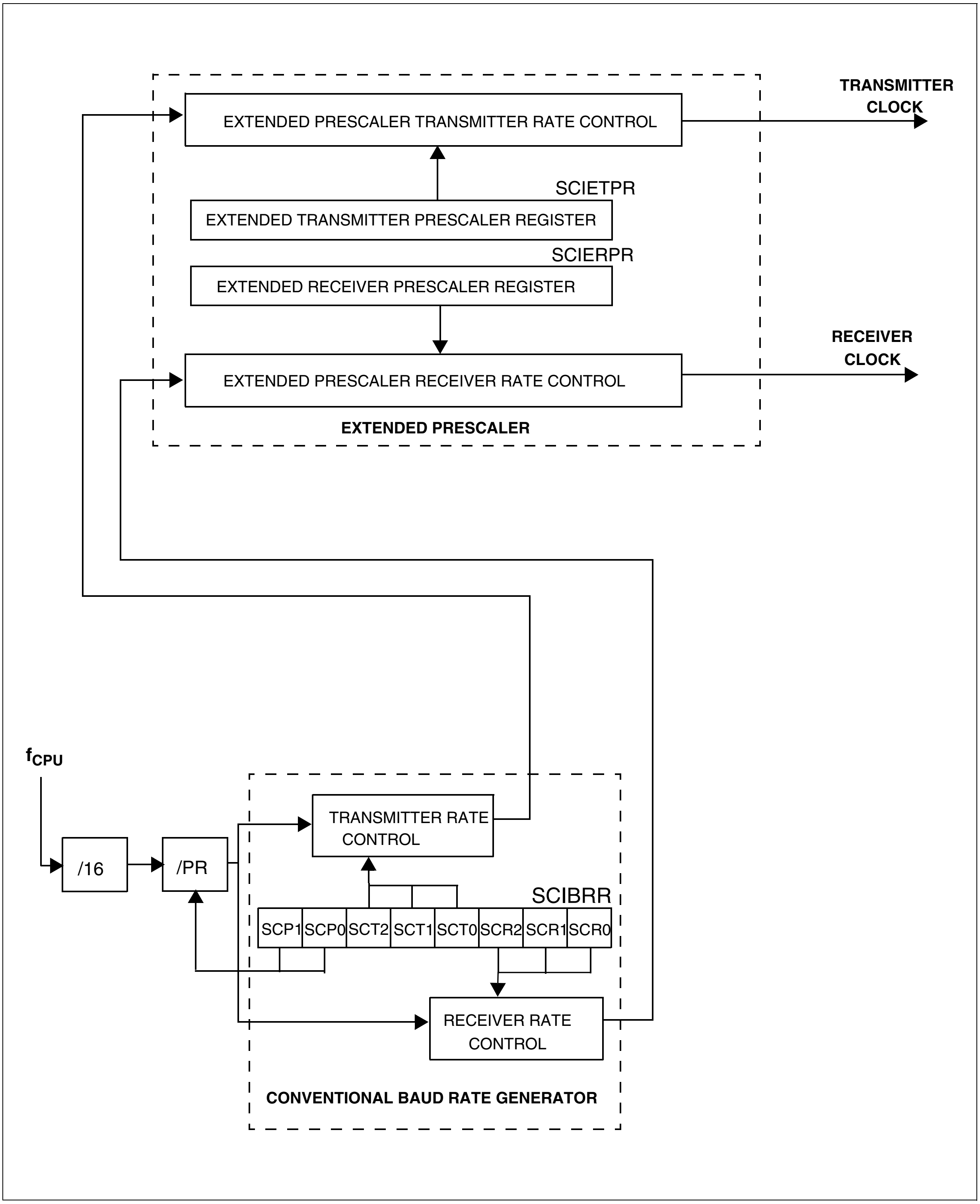
During reception, if a false start bit is detected (e.g. 8th, 9th, 10th samples are 011,101,110), the frame is discarded and the receiving sequence is not started for this frame. There is no RDRF bit set for this frame and the NF flag is set internally (not accessible to the user). This NF flag is accessible along with the RDRF bit when a next valid frame is received.

Note: If the application Start Bit is not long enough to match the above requirements, then the NF Flag may get set due to the short Start Bit. In this case, the NF flag may be ignored by the application software when the first valid byte is received.

See also Section 9.6.4.10.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 59. SCI Baud Rate and Extended Prescaler Block Diagram


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

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Framing Error

A framing error is detected when:

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

When the framing error is detected:

- the FE bit is set by hardware
- Data is transferred from the Shift register to the SCIDR register.
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

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

9.6.4.4 Conventional Baud Rate Generation

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

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

with:

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

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

(see SCT[2:0] bits)

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

(see SCR[2:0] bits)

All these bits are in the SCIBRR register.

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

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

9.6.4.5 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 59

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

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

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

with:

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

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

9.6.4.6 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 interrupts 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 recognized 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.

CAUTION: In Mute mode, do not write to the SCICR2 register. If the SCI is in Mute mode during the read operation (RWU = 1) and a address mark wake up event occurs (RWU is reset) before the write operation, the RWU bit is set again by this write operation. Consequently the address byte is lost and the SCI is not woken up from Mute mode.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.4.7 Parity Control

Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the SCICR1 register. Depending on the frame length defined by the M bit, the possible SCI frame formats are as listed in Table 19.

Table 19. Frame Formats

M bit	PCE bit	SCI frame
0	0	I SB I 8 bit data I STB I
0	1	I SB I 7-bit data I PB I STB I
1	0	I SB I 9-bit data I STB I
1	1	I SB I 8-bit data PB I STB I

Legend: SB = Start Bit, STB = Stop Bit,
PB = Parity Bit

Note: In case of wake up by an address mark, the MSB bit of the data is taken into account and not the parity bit

Even parity: the parity bit is calculated to obtain an even number of “1s” inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit is 0 if even parity is selected (PS bit = 0).

Odd parity: the parity bit is calculated to obtain an odd number of “1s” inside the frame made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit is 1 if odd parity is selected (PS bit = 1).

Transmission mode: If the PCE bit is set then the MSB bit of the data written in the data register is not transmitted but is changed by the parity bit.

Reception mode: If the PCE bit is set then the interface checks if the received data byte has an even number of “1s” if even parity is selected (PS = 0) or an odd number of “1s” if odd parity is selected (PS = 1). If the parity check fails, the PE flag is set in the SCISR register and an interrupt is generated if PIE is set in the SCICR1 register.

9.6.4.8 SCI Clock Tolerance

During reception, each bit is sampled 16 times. The majority of the 8th, 9th and 10th samples is considered as the bit value. For a valid bit detection, all the three samples should have the same value otherwise the noise flag (NF) is set. For example: If the 8th, 9th and 10th samples are 0, 1 and 1 respectively, then the bit value is “1”, but the Noise Flag bit is set because the three samples values are not the same.

Consequently, the bit length must be long enough so that the 8th, 9th and 10th samples have the desired bit value. This means the clock frequency should not vary more than 6/16 (37.5%) within one bit. The sampling clock is resynchronized at each start bit, so that when receiving 10 bits (one start bit, 1 data byte, 1 stop bit), the clock deviation must not exceed 3.75%.

Note: The internal sampling clock of the microcontroller samples the pin value on every falling edge. Therefore, the internal sampling clock and the time the application expects the sampling to take place may be out of sync. For example: If the baud rate is 15.625 Kbaud (bit length is 64µs), then the 8th, 9th and 10th samples are at 28µs, 32µs and 36µs respectively (the first sample starting ideally at 0µs). But if the falling edge of the internal clock occurs just before the pin value changes, the samples would then be out of sync by ~4us. This means the entire bit length must be at least 40µs (36µs for the 10th sample + 4µs for synchronization with the internal sampling clock).

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.4.9 Clock Deviation Causes

The causes which contribute to the total deviation are:

- D_{TRA}: Deviation due to transmitter error (Local oscillator error of the transmitter or the transmitter is transmitting at a different baud rate).
- D_{QUANT}: Error due to the baud rate quantization of the receiver.
- D_{REC}: Deviation of the local oscillator of the receiver: This deviation can occur during the reception of one complete SCI message assuming that the deviation has been compensated at the beginning of the message.
- D_{TCL}: Deviation due to the transmission line (generally due to the transceivers)

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

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

9.6.4.10 Noise Error Causes

See also description of Noise error in Section 9.6.4.3.

Start bit

The noise flag (NF) is set during start bit reception if one of the following conditions occurs:

1. A valid falling edge is not detected. A falling edge is considered to be valid if the 3 consecutive samples before the falling edge occurs are detected as '1' and, after the falling edge occurs, during the sampling of the 16 samples, if one of the samples numbered 3, 5 or 7 is detected as a “1”.
2. During sampling of the 16 samples, if one of the samples numbered 8, 9 or 10 is detected as a “1”.

Therefore, a valid Start Bit must satisfy both the above conditions to prevent the Noise Flag getting set.

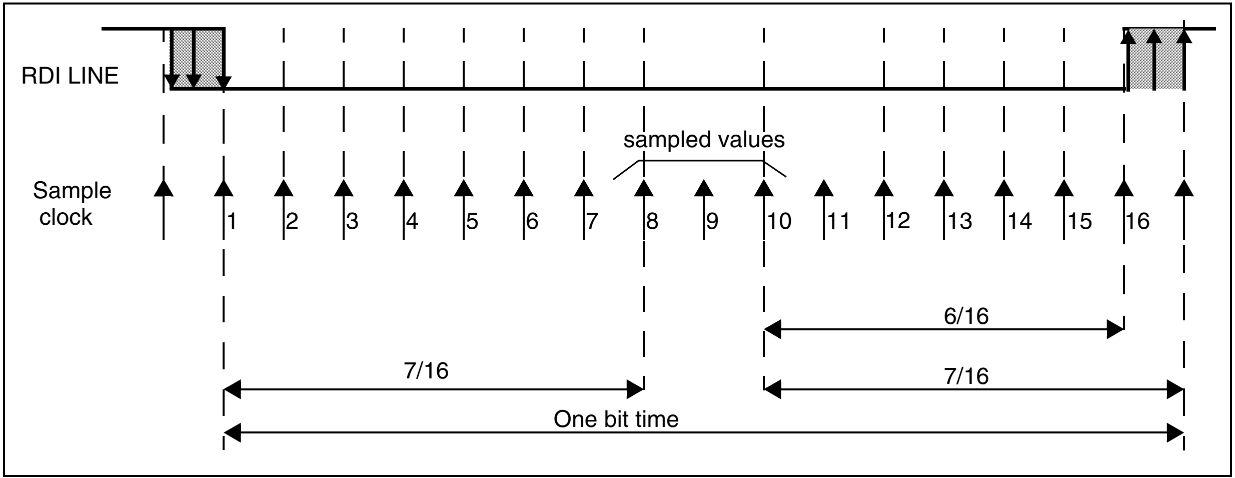
Data Bits

The noise flag (NF) is set during normal data bit reception if the following condition occurs:

- During the sampling of 16 samples, if all three samples numbered 8, 9 and10 are not the same. The majority of the 8th, 9th and 10th samples is considered as the bit value.

Therefore, a valid Data Bit must have samples 8, 9 and 10 at the same value to prevent the Noise Flag getting set.

Figure 60. Bit Sampling in Reception Mode



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.5 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.

9.6.6 Interrupts

The SCI interrupt events are connected to the same interrupt vector.

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).

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
Overrun Error Detected	OR		Yes	No
Idle Line Detected	IDLE	ILIE	Yes	No
Parity Error	PE	PIE	Yes	No

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

9.6.7 Register Description

STATUS REGISTER (SCISR)

Read Only
Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR	NF	FE	PE

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

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

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

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

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

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

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

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

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

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

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

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

Note: This bit does not generate 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 = **PE** *Parity error.*
This bit is set by hardware when a parity error occurs in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.
0: No parity error
1: Parity error

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CONTROL REGISTER 1 (SCICR1)

Read/Write

Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE

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 5 = **SCID** *Disabled for low power consumption*
When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption.This bit is set and cleared by software.
0: SCI enabled
1: SCI prescaler and outputs disabled

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

Note: The M bit must not be modified during a data transfer (both transmission and reception).

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

Bit 2 = **PCE** *Parity control enable.*
This bit selects the hardware parity control (generation and detection). When the parity control is enabled, the computed parity is inserted at the MSB position (9th bit if M = 1; 8th bit if M = 0) and parity is checked on the received data. This bit is set and cleared by software. Once it is set, PCE is active after the current byte (in reception and in transmission).
0: Parity control disabled
1: Parity control enabled

Bit 1 = **PS** *Parity selection.*
This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity is selected after the current byte.
0: Even parity
1: Odd parity

Bit 0 = **PIE** *Parity interrupt enable.*
This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). It is set and cleared by software.
0: Parity error interrupt disabled
1: Parity error interrupt enabled.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

CONTROL REGISTER 2 (SCICR2)

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 SCISR 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 SCISR 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 SCISR 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 SCISR register.

Bit 3 = **TE** *Transmitter enable.*
This bit enables the transmitter. It is set and cleared by software.
0: Transmitter is disabled
1: Transmitter is enabled

Notes:

– During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word.

– When TE is set there is a 1 bit-time delay before the transmission starts.

CAUTION: The TDO pin is free for general purpose I/O only when the TE and RE bits are both cleared (or if TE is never set).

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

Note: Before selecting Mute mode (setting the RWU bit), the SCI must receive some data first, otherwise it cannot function in Mute mode with wake-up by idle line detection.

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 sends a BREAK word at the end of the current word.

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

DATA REGISTER (SCIDR)

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 57).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 57).

BAUD RATE REGISTER (SCIBRR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

Bits 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

Bits 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

Bits 2:0 = **SCR[2:0]** *SCI Receiver rate divisor.*
These 3 bits, in conjunction with the SCP[1:0] 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

SERIAL COMMUNICATIONS INTERFACE (Cont'd)

EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERPR)

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

Bits 7:0 = **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 59) is divided by the binary factor set in the SCIERPR register (in the range 1 to 255).

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

EXTENDED TRANSMIT PRESCALER DIVISION REGISTER (SCIETPR)

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

Bits 7:0 = **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 59) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255).

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

Table 20. Baudrate Selection

Symbol	Parameter	Conditions			Standard	Baud Rate	Unit
		f_{CPU}	Accuracy vs Standard	Prescaler			
f_{Tx} f_{Rx}	Communication frequency	8 MHz	~0.16%	Conventional Mode TR (or RR)=128, PR=13	300	~300.48	Hz
				TR (or RR)= 32, PR=13	1200	~1201.92	
				TR (or RR)= 16, PR=13	2400	~2403.84	
				TR (or RR)= 8, PR=13	4800	~4807.69	
				TR (or RR)= 4, PR=13	9600	~9615.38	
				TR (or RR)= 16, PR= 3	10400	~10416.67	
				TR (or RR)= 2, PR=13	19200	~19230.77	
				TR (or RR)= 1, PR=13	38400	~38461.54	
			~0.79%	Extended Mode ETPR (or ERPR) = 35, TR (or RR)= 1, PR=1	14400	~14285.71	

SERIAL COMMUNICATION INTERFACE (Cont'd)

Table 21. SCI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0050h	**SCISR** Reset Value	TDRE 1	TC 1	RDRF 0	IDLE 0	OR 0	NF 0	FE 0	PE 0
0051h	**SCIDR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
0052h	**SCIBRR** Reset Value	SCP1 0	SCP0 0	SCT2 0	SCT1 0	SCT0 0	SCR2 0	SCR1 0	SCR0 0
0053h	**SCICR1** Reset Value	R8 x	T8 0	SCID 0	M 0	WAKE 0	PCE 0	PS 0	PIE 0
0054h	**SCICR2** Reset Value	TIE 0	TCIE 0	RIE 0	ILIE 0	TE 0	RE 0	RWU 0	SBK 0
0055h	**SCIERPR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0
0057h	**SCIPETPR** Reset Value	MSB 0	0	0	0	0	0	0	LSB 0

9.7 I^{2}C BUS INTERFACE (I2C)

9.7.1 Introduction

The I^{2}C Bus Interface serves as an interface between the microcontroller and the serial I^{2}C bus. It provides both multimaster and slave functions, and controls all I^{2}C bus-specific sequencing, protocol, arbitration and timing. It supports fast I^{2}C mode (400 kHz).

9.7.2 Main Features

- Parallel-bus/I^{2}C protocol converter
- Multi-master capability
- 7-bit/10-bit Addressing
- SMBus V1.1 Compliant
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection

I^{2}C Master Features:

- Clock generation
- I^{2}C bus busy flag
- Arbitration Lost Flag
- End of byte transmission flag
- Transmitter/Receiver Flag
- Start bit detection flag
- Start and Stop generation

I^{2}C Slave Features:

- Stop bit detection
- I^{2}C bus busy flag
- Detection of misplaced start or stop condition
- Programmable I^{2}C Address detection
- Transfer problem detection
- End-of-byte transmission flag
- Transmitter/Receiver flag

9.7.3 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 I^{2}C bus by a data pin (SDAI) and by a clock pin (SCLI). It can be connected both with a standard I^{2}C bus and a Fast I^{2}C 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

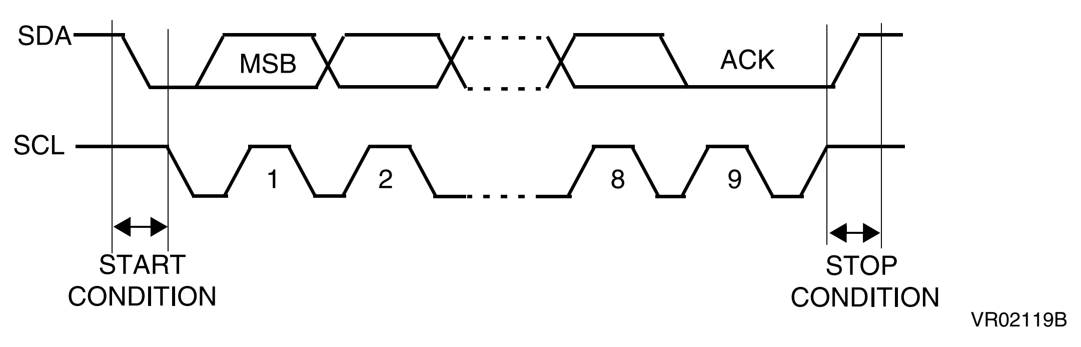
In Master mode, it initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

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 61.

Figure 61. I^{2}C BUS Protocol



I^{2}C BUS INTERFACE (Cont'd)

Acknowledge may be enabled and disabled by software.

The I^{2}C interface address and/or general call address can be selected by software.

The speed of the I^{2}C interface may be selected between Standard (up to 100 kHz) and Fast I^{2}C (up to 400 kHz).

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.

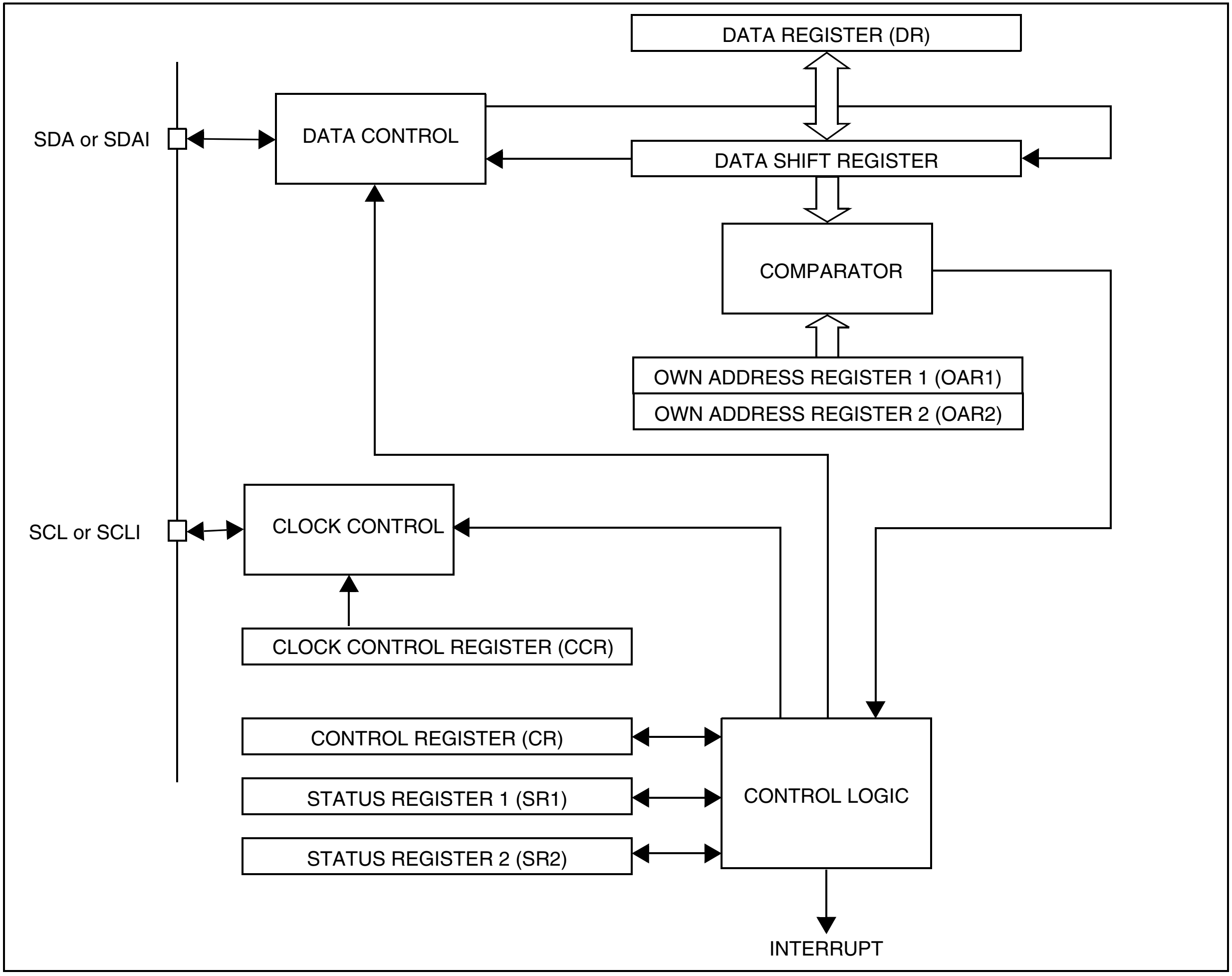
Receiver mode: the interface holds the clock line low after reception to wait for the microcontroller to read the byte in the Data Register.

The SCL frequency (F_{scl}) is controlled by a programmable clock divider which depends on the I^{2}C bus mode.

When the I^{2}C 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 I^{2}C cell is disabled, the SDA and SCL ports revert to being standard I/O port pins.

Figure 62. I^{2}C Interface Block Diagram


DATA REGISTER (DR)
SDA or SDAI
DATA CONTROL
DATA SHIFT REGISTER
COMPARATOR
OWN ADDRESS REGISTER 1 (OAR1)
OWN ADDRESS REGISTER 2 (OAR2)
SCL or SCLI
CLOCK CONTROL
CLOCK CONTROL REGISTER (CCR)
CONTROL REGISTER (CR)
STATUS REGISTER 1 (SR1)
STATUS REGISTER 2 (SR2)
CONTROL LOGIC
INTERRUPT

I^{2}C BUS INTERFACE (Cont'd)

9.7.4 Functional Description

Refer to the CR, SR1 and SR2 registers in Section 9.7.7. for the bit definitions.

By default the I^{2}C 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.

9.7.4.1 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 comparison 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 63 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 63 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 63 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 63 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.
The AF bit is cleared by reading the I2CSR2 register. However, if read before the completion of the transmission, the AF flag will be set again, thus possibly generating a new interrupt. Software must ensure either that the SCL line is back at 0 before reading the SR2 register, or be able to correctly handle a second interrupt during the 9th pulse of a transmitted byte.

Note: In case of errors, SCL line is not held low; however, the SDA line can remain low if the last bits transmitted are all 0. While AF=1, the SCL line may be held low due to SB or BTF flags that are set at the same time. It is then necessary to release both lines by software.

I^2C 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.

SMBus Compatibility

ST7 I^2C is compatible with SMBus V1.1 protocol. It supports all SMBus adressing modes, SMBus bus protocols and CRC-8 packet error checking. Refer to AN1713: SMBus Slave Driver For ST7 I^2C Peripheral.

9.7.4.2 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 63 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 63 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 63 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 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 63 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.

I^{2}C 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 63 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.
Note that BERR will not be set if an error is detected during the first or second pulse of each 9-bit transaction:
Single Master Mode
If a Start or Stop is issued during the first or second pulse of a 9-bit transaction, the BERR flag will not be set and transfer will continue however the BUSY flag will be reset. To work around this, slave devices should issue a NACK when they receive a misplaced Start or Stop. The reception of a NACK or BUSY by the master in the middle of communication gives the possibility to reinitiate transmission.
Multimaster Mode
Normally the BERR bit would be set whenever unauthorized transmission takes place while transfer is already in progress. However, an issue will arise if an external master generates an unauthorized Start or Stop while the I^{2}C master is on the first or second pulse of a 9-bit transaction. It is possible to work around this by polling the BUSY bit during I^{2}C master mode transmission. The resetting of the BUSY bit can then be handled in a similar manner as the BERR flag being set.

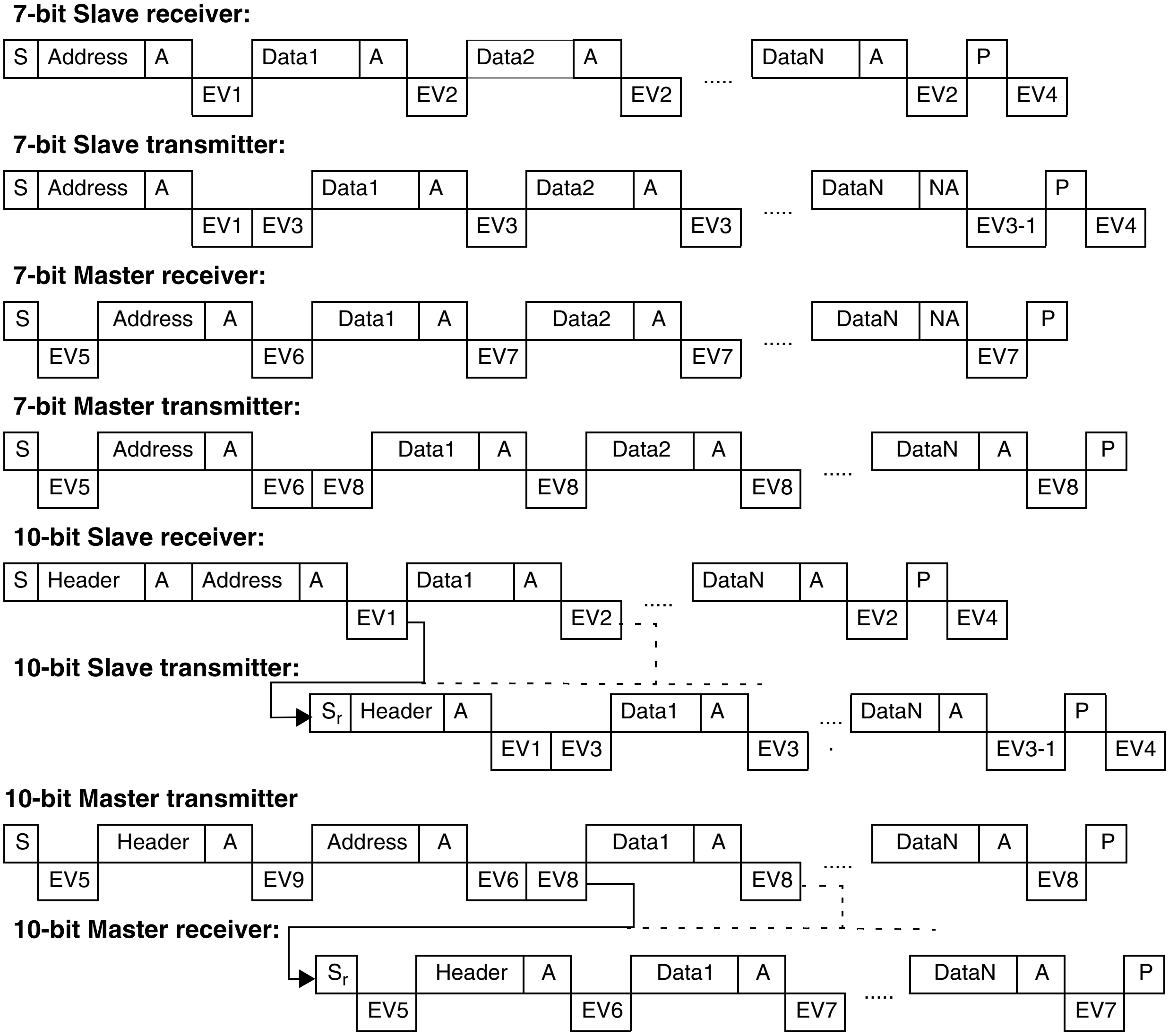
– **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set by hardware with an interrupt if the ITE bit is set. To resume, set the Start or Stop bit.
The AF bit is cleared by reading the I2CSR2 register. However, if read before the completion of the transmission, the AF flag will be set again, thus possibly generating a new interrupt. Software must ensure either that the SCL line is back at 0 before reading the SR2 register, or be able to correctly handle a second interrupt during the 9th pulse of a transmitted byte.

– **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 63. Transfer Sequencing


7-bit Slave receiver:
S Address A Data1 A Data2 A ..... DataN A P
EV1 EV2 EV2 EV2 EV4
7-bit Slave transmitter:
S Address A Data1 A Data2 A ..... DataN NA P
EV1 EV3 EV3 EV3 EV3-1 EV4
7-bit Master receiver:
S Address A Data1 A Data2 A ..... DataN NA P
EV5 EV6 EV7 EV7 EV7
7-bit Master transmitter:
S Address A Data1 A Data2 A ..... DataN A P
EV5 EV6 EV8 EV8 EV8 EV8
10-bit Slave receiver:
S Header A Address A Data1 A ..... DataN A P
EV1 EV2 EV2 EV4
10-bit Slave transmitter:
Sr Header A Data1 A ..... DataN A P
EV1 EV3 EV3 EV3-1 EV4
10-bit Master transmitter
S Header A Address A Data1 A ..... DataN A P
EV5 EV9 EV6 EV8 EV8 EV8
10-bit Master receiver:
Sr Header A Data1 A ..... DataN A P
EV5 EV6 EV7 EV7

Legend: S=Start, Sr = 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.

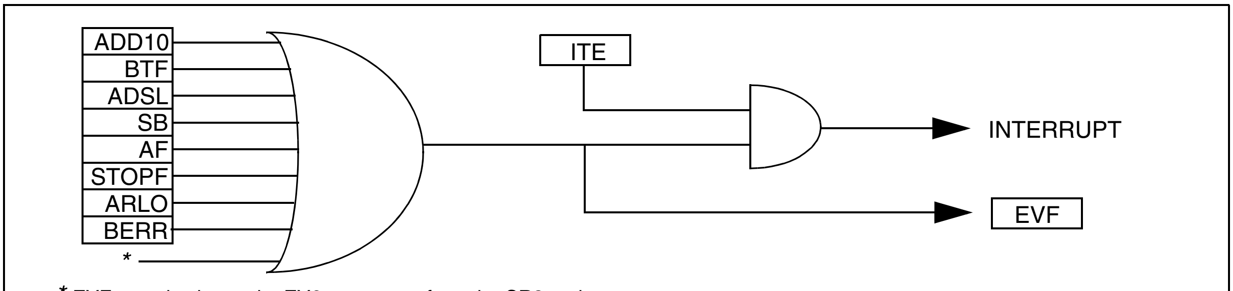
I²C BUS INTERFACE (Cont'd)

9.7.5 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.

9.7.6 Interrupts

Figure 64. 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).

I^2C BUS INTERFACE (Cont'd)

9.7.7 Register Description

I^2C 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 I^2C 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

Note: In accordance with the I2C standard, when GCAL addressing is enabled, an I2C slave can only receive data. It will not transmit data to the master.

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 64 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 63) is detected.

I^{2}C BUS INTERFACE (Cont'd)

I^{2}C 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 63. 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. The BUSY flag of the I2CSR1 register is cleared if a Bus Error occurs.
0: No communication on the bus
1: Communication ongoing on the bus
Note:

- The BUSY flag is NOT updated when the interface is disabled (PE=0). This can have consequences when operating in Multimaster mode; i.e. a second active I^{2}C master commencing a transfer with an unset BUSY bit can cause a conflict resulting in lost data. A software workaround consists of checking that the I^{2}C is not busy before enabling the I^{2}C Multimaster cell.

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 63). 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

I^{2}C 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

I^{2}C 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 but by other flags (SB or BTF) that are set at the same time.

0: No acknowledge failure
1: Acknowledge failure
Note:
– When an AF event occurs, the SCL line is not held low; however, the SDA line can remain low if the last bits transmitted are all 0. It is then necessary to release both lines by software.

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
Note:
– In a Multimaster environment, when the interface is configured in Master Receive mode it does not perform arbitration during the reception of the Acknowledge Bit. Mishandling of the ARLO bit from the I2CSR2 register may occur when a second master simultaneously requests the same data from the same slave and the I^{2}C master does not acknowledge the data. The ARLO bit is then left at 0 instead of being set.

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
Note:
– If a Bus Error occurs, a Stop or a repeated Start condition should be generated by the Master to re-synchronize communication, get the transmission acknowledged and the bus released for further communication

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

I^2C BUS INTERFACE (Cont'd)

I^2C 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 I^2C mode.*
This bit is set and cleared by software. It is not cleared when the interface is disabled (PE=0).
0: Standard I^2C mode
1: Fast I^2C mode

Bit 6:0 = **CC[6:0]** *7-bit clock divider.*
These bits select the speed of the bus (F_{SCL}) depending on the I^2C mode. They are not cleared when the interface is disabled (PE=0).

Refer to the Electrical Characteristics section for the table of values.

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

I^2C 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.

I^2C BUS INTERFACE (Cont'd)

I^2C 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 I^2C 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 I^2C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

I^2C 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 I^2C specified 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 I^2C 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 22. I^{2}C Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0018h	**I2CCR** Reset Value	 0	 0	PE 0	ENGC 0	START 0	ACK 0	STOP 0	ITE 0
0019h	**I2CSR1** Reset Value	EVF 0	ADD10 0	TRA 0	BUSY 0	BTF 0	ADSL 0	M/SL 0	SB 0
001Ah	**I2CSR2** Reset Value	 0	 0	 0	AF 0	STOPF 0	ARLO 0	BERR 0	GCAL 0
001Bh	**I2CCCR** Reset Value	FM/SM 0	CC6 0	CC5 0	CC4 0	CC3 0	CC2 0	CC1 0	CC0 0
001Ch	**I2COAR1** Reset Value	ADD7 0	ADD6 0	ADD5 0	ADD4 0	ADD3 0	ADD2 0	ADD1 0	ADD0 0
001Dh	**I2COAR2** Reset Value	FR1 0	FR0 1	 0	 0	 0	ADD9 0	ADD8 0	 0
001Eh	**I2CDR** Reset Value	MSB 0	 0	 0	 0	 0	 0	 0	LSB 0

9.8 10-BIT A/D CONVERTER (ADC)

9.8.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 16 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 16 different sources.

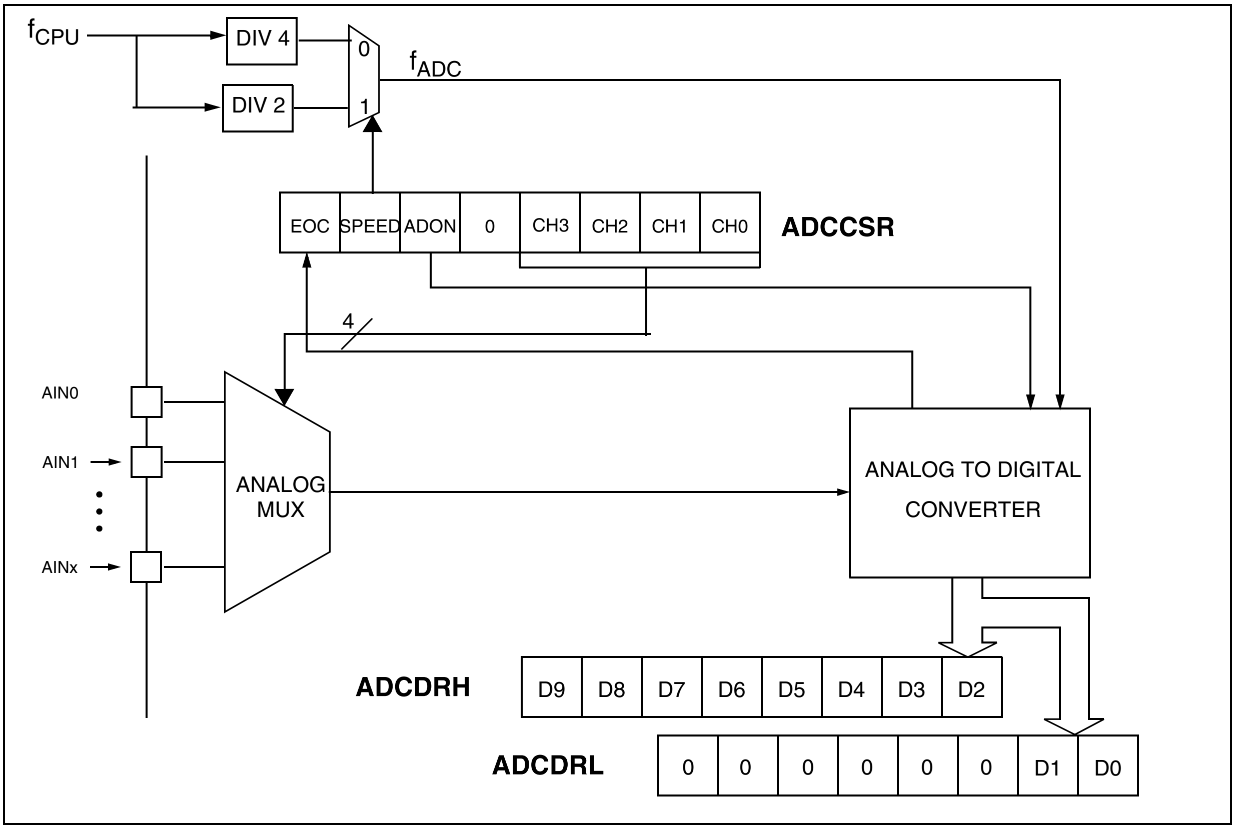
The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

9.8.2 Main Features

- 10-bit conversion
- Up to 16 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 65.

Figure 65. ADC Block Diagram



f_{CPU}
DIV 4
DIV 2
0
1
f_{ADC}
EOC
SPEED
ADON
0
CH3
CH2
CH1
CH0
ADCCSR
4
AIN0
AIN1
AINx
ANALOG MUX
ANALOG TO DIGITAL CONVERTER
ADCDRH
D9
D8
D7
D6
D5
D4
D3
D2
ADCDRL
0
0
0
0
0
0
D1
D0

10-BIT A/D CONVERTER (ADC) (Cont'd)

9.8.3 Functional Description

The conversion is monotonic, meaning that the result never decreases if the analog input does not and never increases if the analog input does not.

If the input voltage (V_{AIN}) is greater than V_{AREF} (high-level voltage reference) then the conversion result is FFh in the ADCDRH register and 03h in the ADCDRL register (without overflow indication).

If the input voltage (V_{AIN}) is lower than V_{SSA} (low-level voltage reference) then the conversion result in the ADCDRH and ADCDRL registers is 00 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDRH and ADCDRL registers. The accuracy of the conversion is described in the Electrical Characteristics Section.

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.

9.8.3.1 A/D Converter Configuration

The analog input ports 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 ADCCSR register:

- Select the CS[3:0] bits to assign the analog channel to convert.

9.8.3.2 Starting the Conversion

In the ADCCSR register:

- Set the ADON bit to enable the A/D converter and to start the conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

When a conversion is complete:

- The EOC bit is set by hardware.
- The result is in the ADCDR registers.

A read to the ADCDRH or a write to any bit of the ADCCSR register resets the EOC bit.

To read the 10 bits, perform the following steps:

1. Poll the EOC bit
2. Read the ADCDRL register
3. Read the ADCDRH register. This clears EOC automatically.

Note: The data is not latched, so both the low and the high data register must be read before the next conversion is complete, so it is recommended to disable interrupts while reading the conversion result.

To read only 8 bits, perform the following steps:

1. Poll the EOC bit
2. Read the ADCDRH register. This clears EOC automatically.

9.8.3.3 Changing the conversion channel

The application can change channels during conversion. When software modifies the CH[3:0] bits in the ADCCSR register, the current conversion is stopped, the EOC bit is cleared, and the A/D converter starts converting the newly selected channel.

9.8.4 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 and between single shot conversions.

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 stabilization time t_{STAB} (see Electrical Characteristics) before accurate conversions can be performed.

9.8.5 Interrupts

None.

10-BIT A/D CONVERTER (ADC) (Cont'd)

9.8.6 Register Description

CONTROL/STATUS REGISTER (ADCCSR)

Read/Write (Except bit 7 read only)

Reset Value: 0000 0000 (00h)

7							0
EOC	SPEED	ADON	0	CH3	CH2	CH1	CH0

Bit 7 = **EOC** *End of Conversion*
This bit is set by hardware. It is cleared by hardware when software reads the ADCDRH register or writes to any bit of the ADCCSR register.
0: Conversion is not complete
1: Conversion complete

Bit 6 = **SPEED** *ADC clock selection*
This bit is set and cleared by software.
0: $f_{ADC} = f_{CPU}/4$
1: $f_{ADC} = f_{CPU}/2$

Bit 5 = **ADON** *A/D Converter on*
This bit is set and cleared by software.
0: Disable ADC and stop conversion
1: Enable ADC and start conversion

Bit 4 = **Reserved.** Must be kept cleared.

Bit 3:0 = **CH[3:0]** *Channel Selection*
These bits are set and cleared by software. They select the analog input to convert.

Channel Pin*	CH3	CH2	CH1	CH0
AIN0	0	0	0	0
AIN1	0	0	0	1
AIN2	0	0	1	0
AIN3	0	0	1	1
AIN4	0	1	0	0
AIN5	0	1	0	1
AIN6	0	1	1	0
AIN7	0	1	1	1
AIN8	1	0	0	0
AIN9	1	0	0	1
AIN10	1	0	1	0
AIN11	1	0	1	1
AIN12	1	1	0	0
AIN13	1	1	0	1
AIN14	1	1	1	0
AIN15	1	1	1	1

*The number of channels is device dependent. Refer to the device pinout description.

DATA REGISTER (ADCDRH)

Read Only

Reset Value: 0000 0000 (00h)

7							0
D9	D8	D7	D6	D5	D4	D3	D2

Bit 7:0 = **D[9:2]** *MSB of Converted Analog Value*

DATA REGISTER (ADCDRL)

Read Only

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	D1	D0

Bit 7:2 = Reserved. Forced by hardware to 0.

Bit 1:0 = **D[1:0]** *LSB of Converted Analog Value*

10-BIT A/D CONVERTER (Cont'd)

Table 23. ADC Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0070h	**ADCCSR** Reset Value	EOC 0	SPEED 0	ADON 0	0	CH3 0	CH2 0	CH1 0	CH0 0
0071h	**ADCDRH** Reset Value	D9 0	D8 0	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0
0072h	**ADCDRL** Reset Value	0	0	0	0	0	0	D1 0	D0 0

10 INSTRUCTION SET

10.1 CPU ADDRESSING MODES

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

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

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

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

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

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

Table 24. 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)

10.1.1 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

10.1.2 Immediate

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

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

10.1.3 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.

10.1.4 Indexed (No Offset, Short, Long)

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

The indirect addressing mode consists of three submodes:

Indexed (No Offset)

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

Indexed (Short)

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

Indexed (long)

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

10.1.5 Indirect (Short, Long)

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

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

Indirect (short)

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

Indirect (long)

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

INSTRUCTION SET OVERVIEW (Cont'd)

10.1.6 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 submodes:

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 25. 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

10.1.7 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)

10.2 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 prebyte

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 ext. INT pin = 1	(ext. INT pin high)								
JRIL	Jump if ext. INT pin = 0	(ext. INT pin low)								
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	

11 ELECTRICAL CHARACTERISTICS

11.1 PARAMETER CONDITIONS

Unless otherwise specified, all voltages are referred to V_{SS}.

11.1.1 Minimum and Maximum values

Unless otherwise specified the minimum and maximum values are guaranteed in the worst conditions of ambient temperature, supply voltage and frequencies by tests in production on 100% of the devices with an ambient temperature at T_A=25 °C and T_A=$T_A$max (given by the selected temperature range).

Data based on characterization results, design simulation and/or technology characteristics are indicated in the table footnotes and are not tested in production. Based on characterization, the minimum and maximum values refer to sample tests and represent the mean value plus or minus three times the standard deviation (mean±3Σ).

11.1.2 Typical values

Unless otherwise specified, typical data are based on T_A= 25 °C, V_{DD}= 5 V.They are given only as design guidelines and are not tested.

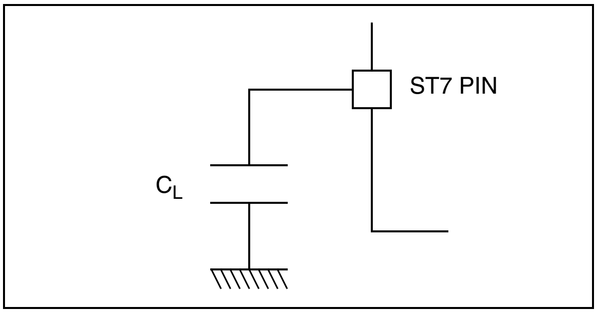
11.1.3 Typical curves

Unless otherwise specified, all typical curves are given only as design guidelines and are not tested.

11.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in Figure 66.

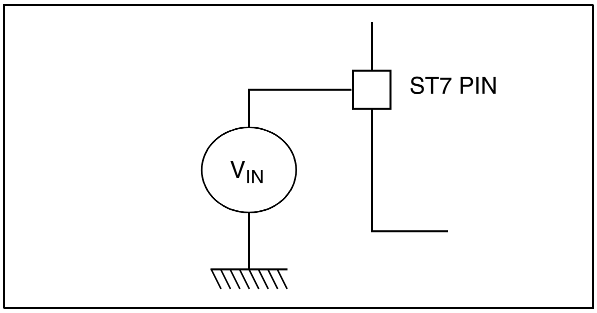
Figure 66. Pin loading conditions



11.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 67.

Figure 67. Pin input voltage



11.2 ABSOLUTE MAXIMUM RATINGS

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 under these conditions is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

11.2.1 Voltage Characteristics

Symbol	Ratings	Maximum value	Unit
$V_{DD} - V_{SS}$	Supply voltage	6.5	V
$V_{PP} - V_{SS}$	Programming Voltage	13	
V_{IN} 1) & 2)	Input Voltage on true open drain pin	V_{SS}-0.3 to 6.5	
	Input voltage on any other pin	V_{SS}-0.3 to V_{DD}+0.3	
$\|\Delta V_{DDx}\|$ and $\|\Delta V_{SSx}\|$	Variations between different digital power pins	50	mV
$\|V_{SSA} - V_{SSx}\|$	Variations between digital and analog ground pins	50	
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)	see section 11.7.3 on page 150	
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine Model)		

11.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) 3)	150	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink) 3)	150	
I_{IO}	Output current sunk by any standard I/O and control pin	25	mA
	Output current sunk by any high sink I/O pin	50	
	Output current source by any I/Os and control pin	- 25	
$I_{INJ(PIN)}$ 2) & 4)	Injected current on V_{PP} pin	+ 5	mA
	Injected current on $\overline{\text{RESET}}$ pin	+ 5	
	Injected current on OSC1 and OSC2 pins	+ 5	
	Injected current on any other pin 5) & 6)	± 5	
$\Sigma I_{INJ(PIN)}$ 2)	Total injected current (sum of all I/O and control pins) 5)	± 25	

Notes:

1. Directly connecting the $\overline{\text{RESET}}$ and I/O pins to V_{DD} or V_{SS} could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection has to be done through a pull-up or pull-down resistor (typical: 4.7 kΩ for $\overline{\text{RESET}}$, 10 kΩ for I/Os). For the same reason, unused I/O pins must not be directly tied to V_{DD} or V_{SS}.
2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$. For true open-drain pads, there is no positive injection current, and the corresponding V_{IN} maximum must always be respected
3. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
4. Negative injection disturbs the analog performance of the device. See note in “ADC Accuracy” on page 165.
For best reliability, it is recommended to avoid negative injection of more than 1.6 mA.
5. When several inputs are submitted to a current injection, the maximum $\Sigma I_{INJ(PIN)}$ is the absolute sum of the positive and negative injected currents (instantaneous values). These results are based on characterisation with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.
6. True open drain I/O port pins do not accept positive injection.

11.2.3 Thermal Characteristics

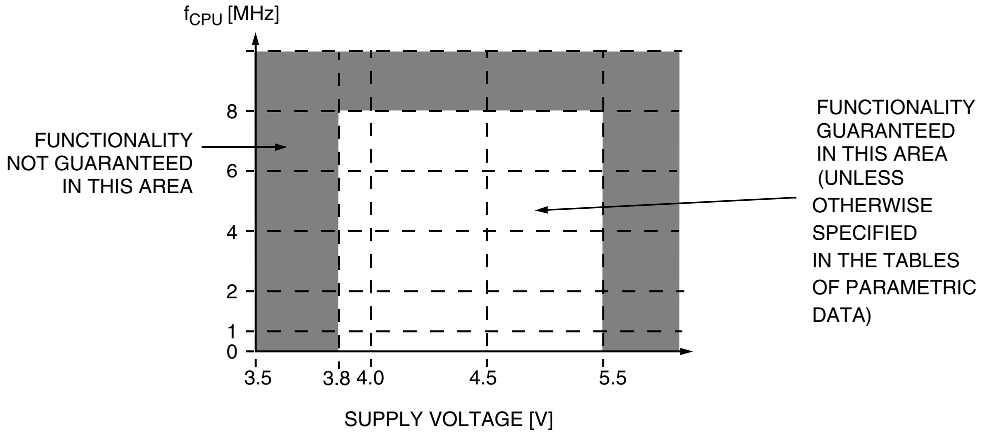
Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature (see Section 12.3 THERMAL CHARACTERISTICS)		

11.3 OPERATING CONDITIONS

11.3.1 General Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
f_{CPU}	Internal clock frequency		0	8	MHz
V_{DD}	Standard voltage range (except Flash Write/Erase)		3.8	5.5	V
	Operating Voltage for Flash Write/Erase	V_{PP} = 11.4 to 12.6 V	4.5	5.5	
T_A	Ambient temperature range	1 Suffix Version	0	70	°C
		5 Suffix Version	-10	85	
		6 or A Suffix Versions	-40	85	
		7 or B Suffix Versions	-40	105	
		3 or C Suffix Version	-40	125	

Figure 68. f_{CPU} Max Versus V_{DD}



Note: Some temperature ranges are only available with a specific package and memory size. Refer to Ordering Information.

OPERATING CONDITIONS (Cont'd)

11.3.2 Operating Conditions with Low Voltage Detector (LVD)

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(LVD)}$	Reset release threshold (V_{DD} rise)	VD level = High in option byte	4.0[1]	4.2	4.5	V
		VD level = Med. in option byte[2] VD level = Low in option byte[2]	3.55[1] 2.95[1]	3.75 3.15	4.0[1] 3.35[1]	
$V_{IT-(LVD)}$	Reset generation threshold (V_{DD} fall)	VD level = High in option byte	3.8	4.0	4.25[1]	
		VD level = Med. in option byte[2] VD level = Low in option byte[2]	3.35[1] 2.8[1]	3.55 3.0	3.75[1)] 3.15 [1]	
$V_{hys(LVD)}$	LVD voltage threshold hysteresis	$V_{IT+(LVD)}$-$V_{IT-(LVD)}$		200		mV
Vt_{POR}	V_{DD} rise time [3][2]	LVD enabled	6 µs/V		100 ms/V	
$t_{g(VDD)}$	V_{DD} glitches filtered (not detected) by LVD [3]				40	ns

Notes:

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

2. If the medium or low thresholds are selected, the detection may occur outside the specified operating voltage range. Below 3.8 V, device operation is not guaranteed.

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

3. When Vt_{POR} is faster than 100 µs/V, the Reset signal is released after a delay of max. 42 µs after V_{DD} crosses the $V_{IT+(LVD)}$ threshold.

11.3.3 Auxiliary Voltage Detector (AVD) Thresholds

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(AVD)}$	1 ⇒0 AVDF flag toggle threshold (V_{DD} rise)	VD level = High in option byte	4.4[1]	4.6	4.9[1]	V
		VD level = Med. in option byte VD level = Low in option byte	3.95 [1] 3.4 [1]	4.15 3.6	4.4[1] 3.8[1]	
$V_{IT-(AVD)}$	0 ⇒1 AVDF flag toggle threshold (V_{DD} fall)	VD level = High in option byte	4.2[1]	4.4	4.65 [1]	
		VD level = Med. in option byte VD level = Low in option byte	3.75[1] 3.2[1]	4.0 3.4	4.2 [1] 3.6[1]	
$V_{hys(AVD)}$	AVD voltage threshold hysteresis	$V_{IT+(AVD)}$-$V_{IT-(AVD)}$		200		mV
ΔV_{IT-}	Voltage drop between AVD flag set and LVD reset activated	$V_{IT-(AVD)}$-$V_{IT-(LVD)}$		450		mV

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

11.3.4 External Voltage Detector (EVD) Thresholds

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(EVD)}$	1⇒0 AVDF flag toggle threshold (V_{DD} rise)[1]		1.15	1.26	1.35	V
$V_{IT-(EVD)}$	0 ⇒1 AVDF flag toggle threshold (V_{DD} fall)[1]		1.1	1.2	1.3	
$V_{hys(EVD)}$	EVD voltage threshold hysteresis	$V_{IT+(EVD)}$-$V_{IT-(EVD)}$		200		mV

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

11.4 SUPPLY CURRENT CHARACTERISTICS

The following current consumption specified for the ST7 functional operating modes over temperature range does not take into account the clock source current consumption. To get the total device consumption, the two current values must be added (except for Halt mode for which the clock is stopped).

11.4.1 CURRENT CONSUMPTION

Symbol	Parameter	Conditions	Typ	Max 1)	Unit
I_{DD}	Supply current in RUN mode 2)	f_{OSC}=2 MHz, f_{CPU}=1 MHz f_{OSC}=4 MHz, f_{CPU}=2 MHz f_{OSC}=8 MHz, f_{CPU}=4 MHz f_{OSC}=16 MHz, f_{CPU}=8 MHz	1.3 2.0 3.6 7.1	3.0 5.0 8.0 15.0	mA
	Supply current in Slow mode 2)	f_{OSC}=2 MHz, f_{CPU}=62.5 kHz f_{OSC}=4 MHz, f_{CPU}=125 kHz f_{OSC}=8 MHz, f_{CPU}=250 kHz f_{OSC}=16 MHz, f_{CPU}=500 kHz	600 700 800 1100	2700 3000 3600 4000	µA
	Supply current in Wait mode 2)	f_{OSC}=2 MHz, f_{CPU}=1 MHz f_{OSC}=4 MHz, f_{CPU}=2 MHz f_{OSC}=8 MHz, f_{CPU}=4 MHz f_{OSC}=16 MHz, f_{CPU}=8 MHz	1.0 1.5 2.5 4.5	3.0 4.0 5.0 7.0	mA
	Supply current in Slow wait mode 2)	f_{OSC}=2 MHz, f_{CPU}=62.5 kHz f_{OSC}=4 MHz, f_{CPU}=125 kHz f_{OSC}=8 MHz, f_{CPU}=250 kHz f_{OSC}=16 MHz, f_{CPU}=500 kHz	580 650 770 1050	1200 1300 1800 2000	µA
	Supply current in Halt mode 3)	-40 °C $\leq T_A \leq$+85 °C	<1	10	µA
I_{DD}	Supply current in Active-halt mode 4)	f_{OSC}=2 MHz f_{OSC}=4 MHz f_{OSC}=8 MHz f_{OSC} =16 MHz	80 160 325 650	No max. guaranteed	µA

Notes:

1. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
2. Measurements are done in the following conditions:

- Program executed from RAM, CPU running with RAM access. The increase in consumption when executing from Flash is 50%.
- All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load)
- All peripherals in reset state.
- LVD disabled.
- Clock input (OSC1) driven by external square wave.
- In Slow and Slow wait mode, f_{CPU} is based on f_{OSC} divided by 32.

To obtain the total current consumption of the device, add the clock source (Section 11.4.2) and the peripheral power consumption (Section 11.4.3).

3. All I/O pins in push-pull 0 mode (when applicable) with a static value at V_{DD} or VSS (no load), LVD disabled. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
4. Data based on characterisation results, not tested in production. All I/O pins in push-pull 0 mode (when applicable) with a static value at V_{DD} or V_{SS} (no load); clock input (OSC1) driven by external square wave, LVD disabled. To obtain the total current consumption of the device, add the clock source consumption (Section 11.4.2).

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

11.4.1.1 Power Consumption vs f_{CPU}

Figure 69. Typical I_{DD} in RUN mode

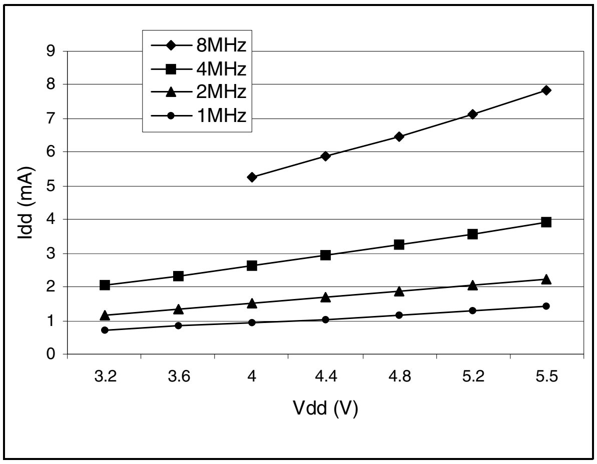


Figure 70. Typical I_{DD} in Slow mode

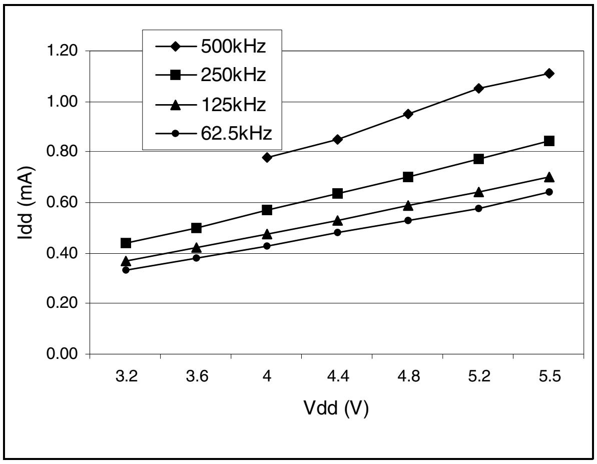


Figure 71. Typical I_{DD} in Wait mode

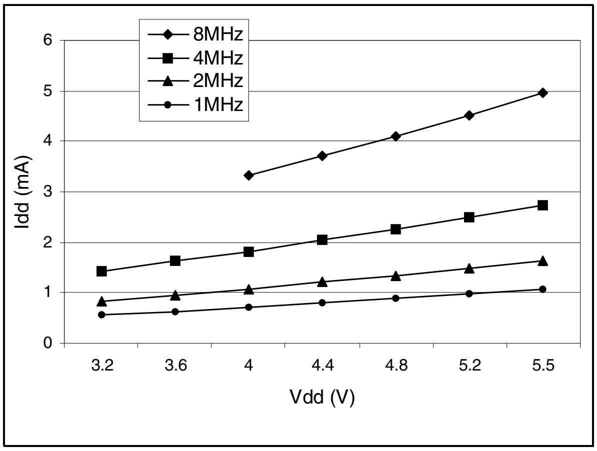
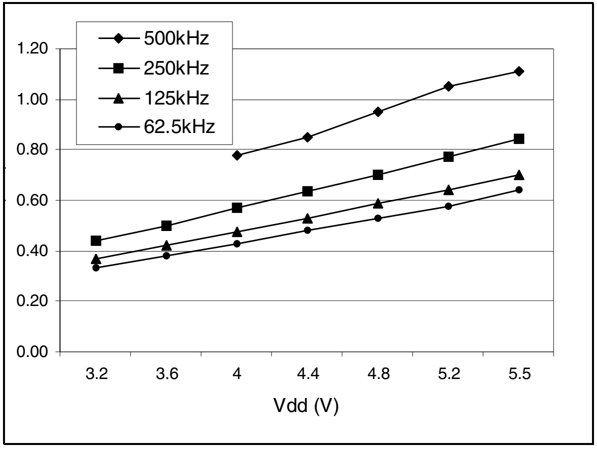


Figure 72. Typ. I_{DD} in Slow-wait mode



SUPPLY CURRENT CHARACTERISTICS (Cont'd)

11.4.2 Supply and Clock Managers

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

Symbol	Parameter	Conditions	Typ	Max	Unit
$I_{DD(RCINT)}$	Supply current of internal RC oscillator		625		µA
$I_{DD(RES)}$	Supply current of resonator oscillator 1) & 2)		see section 11.5.3 on page 143		
$I_{DD(PLL)}$	PLL supply current	V_{DD}= 5 V	360		
$I_{DD(LVD)}$	LVD supply current	V_{DD}= 5 V	150	300	

Notes:

1.. Data based on characterization results done with the external components specified in Section 11.5.3, not tested in production.

2. As the oscillator is based on a current source, the consumption does not depend on the voltage.

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

11.4.3 On-Chip Peripherals

Measured on LQFP64 generic board T_A = 25 °C f_{CPU}=4 MHz.

Symbol	Parameter	Conditions	Typ	Unit
$I_{DD(TIM)}$	16-bit Timer supply current [1]	V_{DD}=5.0 V	50	µA
$I_{DD(ART)}$	ART PWM supply current[2]	V_{DD}=5.0 V	75	µA
$I_{DD(SPI)}$	SPI supply current [3]	V_{DD}=5.0 V	400	µA
$I_{DD(SCI)}$	SCI supply current [4]	V_{DD}=5.0 V	400	µA
$I_{DD(I2C)}$	I2C supply current [5]	V_{DD}=5.0 V	175	µA
$I_{DD(ADC)}$	ADC supply current when converting [6]	V_{DD}=5.0 V	400	µA

Notes:

1. Data based on a differential I_{DD} measurement between reset configuration (timer counter running at f_{CPU}/4) and timer counter stopped (only TIMD bit set). Data valid for one timer.
2. Data based on a differential I_{DD} measurement between reset configuration (timer stopped) and timer counter enabled (only TCE bit set).
3. Data based on a differential I_{DD} measurement between reset configuration (SPI disabled) and a permanent SPI master communication at maximum speed (data sent equal to 55h).This measurement includes the pad toggling consumption.
4. Data based on a differential I_{DD} measurement between SCI low power state (SCID=1) and a permanent SCI data transmit sequence.
5. Data based on a differential I_{DD} measurement between reset configuration (I2C disabled) and a permanent I2C master communication at 100 kHz (data sent equal to 55h). This measurement include the pad toggling consumption (27 kOhm external pull-up on clock and data lines).
6. Data based on a differential I_{DD} measurement between reset configuration and continuous A/D conversions.

11.5 CLOCK AND TIMING CHARACTERISTICS

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A.

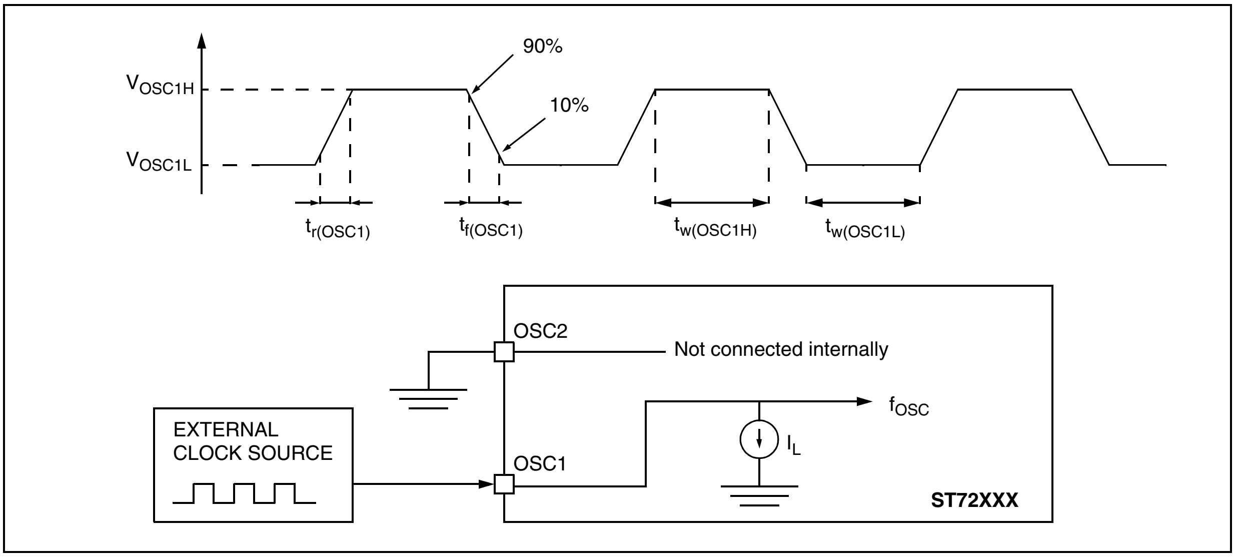
11.5.1 General Timings

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
$t_{c(INST)}$	Instruction cycle time		2	3	12	t_{CPU}
		f_{CPU}=8 MHz	250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time [2] $t_{v(IT)} = \Delta t_{c(INST)} + 10$		10		22	t_{CPU}
		f_{CPU}=8 MHz	1.25		2.75	µs

11.5.2 External Clock Source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSC1H}	OSC1 input pin high level voltage	see Figure 73	$0.7 \times V_{DD}$		V_{DD}	V
V_{OSC1L}	OSC1 input pin low level voltage		V_{SS}		$0.3 \times V_{DD}$	
$t_{w(OSC1H)}$ $t_{w(OSC1L)}$	OSC1 high or low time [3]		5			ns
$t_{r(OSC1)}$ $t_{f(OSC1)}$	OSC1 rise or fall time [3]				15	
I_L	OSC1 Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			±1	µA

Figure 73. Typical Application with an External Clock Source


V_{OSC1H}
V_{OSC1L}
90%
10%
$t_{r(OSC1)}$
$t_{f(OSC1)}$
$t_{w(OSC1H)}$
$t_{w(OSC1L)}$
OSC2
Not connected internally
f_{OSC}
I_L
EXTERNAL CLOCK SOURCE
OSC1
ST72XXX

Notes:

1. Data based on typical application software.
2. Time measured between interrupt event and interrupt vector fetch. $\Delta t_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.
3. Data based on design simulation and/or technology characteristics, not tested in production.

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

11.5.3 Crystal and Ceramic Resonator Oscillators

The ST7 internal clock can be supplied with four different Crystal/Ceramic resonator oscillators. All the information given in this paragraph is based on characterization results with specified typical external components. In the application, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. Refer to the crystal/ceramic resonator manufacturer for more details (frequency, package, accuracy...).

Symbol	Parameter	Conditions		Min	Max	Unit
f_{OSC}	Oscillator Frequency 1)	LP: Low power oscillator		1	2	MHz
		MP: Medium power oscillator		>2	4	
		MS: Medium speed oscillator		>4	8	
		HS: High speed oscillator		>8	16	
R_F	Feedback resistor[2]			20	40	kΩ
C_{L1} C_{L2}	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator (R_S)	R_S=200 Ω	LP oscillator	22	56	pF
		R_S=200 Ω	MP oscillator	22	46	
		R_S=200 Ω	MS oscillator	18	33	
		R_S=100 Ω	HS oscillator	15	33	

Symbol	Parameter	Conditions		Typ	Max	Unit
i_2	OSC2 driving current	V_{DD}=5 V	LP oscillator	80	150	µA
		V_{IN}=V_{SS}	MP oscillator	160	250	
			MS oscillator	310	460	
			HS oscillator	610	910	

Notes:

1. The oscillator selection can be optimized in terms of supply current using an high quality resonator with small R_S value. Refer to crystal/ceramic resonator manufacturer for more details.

2. Data based on characterisation results, not tested in production.

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

Supplier	f_{OSC} (MHz)	Typical Ceramic Resonators[1]	
		Reference[2]	Recommended OSCRANGE Option bit configuration
Murata	2	CSTCC2M00G56A-R0	MP Mode[3]
	4	CSTCR4M00G55B-R0	MS Mode
	8	CSTCE8M00G55A-R0	HS Mode
	16	CSTCE16M0G53A-R0	HS Mode

Notes:

1. Resonator characteristics given by the ceramic resonator manufacturer.
2. SMD = [-R0: Plastic tape package (∅ =180 mm), -B0: Bulk]
   LEAD = [-A0: Flat pack package (Radial taping Ho= 18 mm), -B0: Bulk]
3. LP mode is not recommended for 2 MHz resonator because the peak to peak amplitude is too small (>0.8 V)

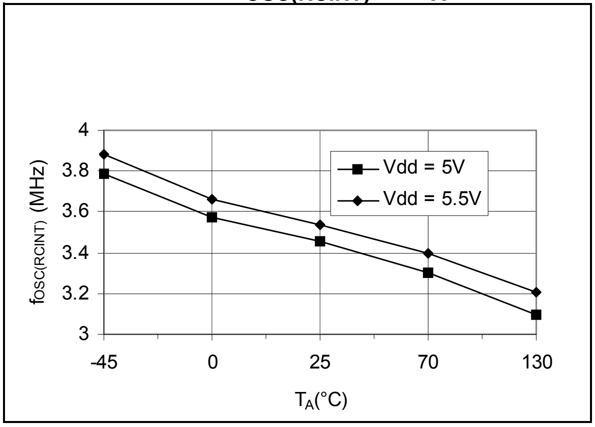
For more information on these resonators, please consult www.murata.com

CLOCK CHARACTERISTICS (Cont'd)

11.5.4 RC Oscillators

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$f_{OSC\ (RCINT)}$	Internal RC oscillator frequency See Figure 74	T_A=25 °C, V_{DD}=5 V	2	3.5	5.6	MHz

Figure 74. Typical $f_{OSC(RCINT)}$ vs T_A



Note: To reduce disturbance to the RC oscillator, it is recommended to place decoupling capacitors between V_{DD} and V_{SS} as shown in Figure 94

CLOCK CHARACTERISTICS (Cont'd)

Note:

1. Data based on characterization results.

11.5.5 PLL Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	PLL input frequency range		2		4	MHz
$\Delta\ f_{CPU}/\ f_{CPU}$	Instantaneous PLL jitter [1]	f_{OSC} = 4 MHz.		1.0	2.5	%
		f_{OSC} = 2 MHz.		2.5	4.0	

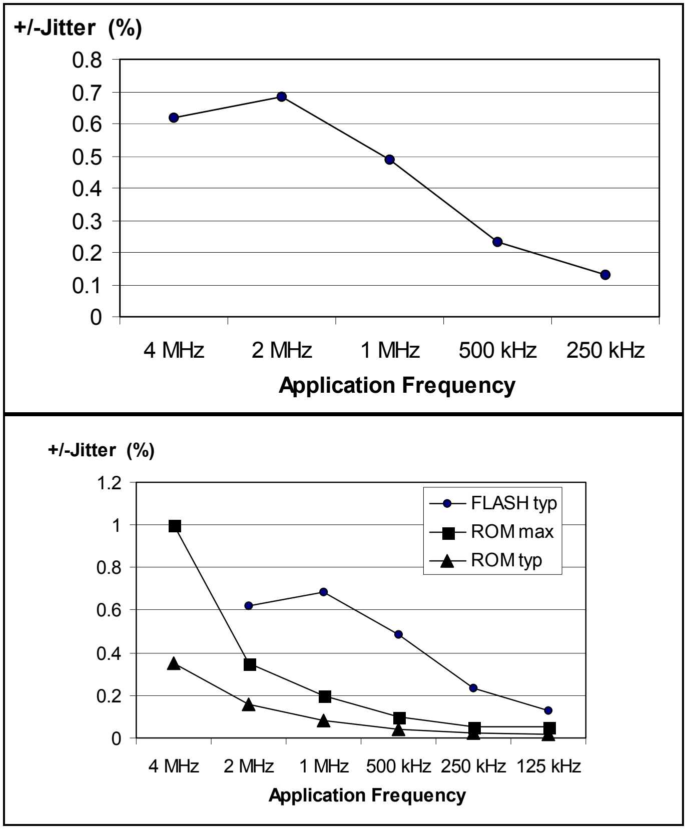
Note:

1. Data characterized but not tested.

The user must take the PLL jitter into account in the application (for example in serial communication or sampling of high frequency signals). The PLL jitter is a periodic effect, which is integrated over several CPU cycles. Therefore the longer the period of the application signal, the less it will be impacted by the PLL jitter.

Figure 75 shows the PLL jitter integrated on application signals in the range 125 kHz to 4 MHz. At frequencies of less than 125 kHz, the jitter is negligible.

Figure 75. Integrated PLL Jitter vs signal frequency[1]



Note 1: Measurement conditions: f_{CPU} = 8 MHz.

11.6 MEMORY CHARACTERISTICS

11.6.1 RAM and Hardware Registers

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

11.6.2 Flash memory

DUAL VOLTAGE HDFlash MEMORY						
Symbol	**Parameter**	**Conditions**	**Min 2)**	**Typ**	**Max 2)**	**Unit**
f_{CPU}	Operating frequency	Read mode	0		8	MHz
		Write / Erase mode	1		8	
V_{PP}	Programming voltage 3)	4.5 V $\leq V_{DD} \leq$ 5.5 V	11.4		12.6	V
I_{DD}	Supply current4)	RUN mode (f_{CPU} = 4 MHz)			3	mA
		Write / Erase		0		
		Power down mode / Halt		1	10	µA
I_{PP}	V_{PP} current4)	Read (V_{PP}=12 V)			200	
		Write / Erase			30	mA
t_{VPP}	Internal V_{PP} stabilization time			10		µs
t_{RET}	Data retention	T_A=85 °C	40			years
		T_A=105 °C	15			
		T_A=125 °C	7			
N_{RW}	Write erase cycles	T_A= 55 °C	1000			cycles
		T_A= 85 °C	100			cycles
T_{PROG} T_{ERASE}	Programming or erasing temperature range		-40	25	85	°C

Notes:

1. Minimum V_{DD} supply voltage without losing data stored in RAM (in Halt mode or under RESET) or in hardware registers (only in Halt mode). Not tested in production.
2. Data based on characterization results, not tested in production.
3. V_{PP} must be applied only during the programming or erasing operation and not permanently for reliability reasons.
4. Data based on simulation results, not tested in production.

Warning: Do not connect 12 V to V_{PP} before V_{DD} is powered on, as this may damage the device.

11.7 EMC CHARACTERISTICS

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

11.7.1 Functional EMS (Electro Magnetic Susceptibility)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electro magnetic events until a failure occurs (indicated by the LEDs).

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

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

11.7.1.1 Designing hardened software to avoid noise problems

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

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

Software recommendations:

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

– Corrupted program counter

– Unexpected reset

– Critical Data corruption (control registers...)

Prequalification trials:

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

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

Symbol	Parameter	Conditions	Level/ Class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD}=5 V, T_A=+25 °C, LQFP64, f_{OSC}=8 MHz, conforms to IEC 1000-4-2	4B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100 pF on V_{DD} and V_{DD} pins to induce a functional disturbance	V_{DD}=5 V, T_A=+25 °C, f_{OSC}=8 MHz, LQFP64, conforms to IEC 1000-4-4	3B

EMC CHARACTERISTICS (Cont'd)

11.7.2 Electro Magnetic Interference (EMI)

Based on a simple application running on the product (toggling 2 LEDs through the I/O ports), the product is monitored in terms of emission. This emission test is in line with the norm SAE J 1752/3 which specifies the board and the loading of each pin.

Symbol	Parameter	Conditions	Monitored Frequency Band	Max vs. [f_{OSC}/f_{CPU}][1]		Unit
				8/4 MHz	16/8 MHz	
S_{EMI}	Peak level	60 Kbytes Flash Devices [2]: V_{DD}=5 V, T_A=+25 °C, LQFP64 package conforming to SAE J 1752/3	0.1 MHz to 30 MHz	15	15	dBμV
			30 MHz to 130 MHz	20	27	
			130 MHz to 1 GHz	0	5	
			SAE EMI Level	2.5	3	-

Notes:

1. Data based on characterization results, not tested in production.
2. Refer to Application Note AN1709 for data on other package types.

EMC CHARACTERISTICS (Cont'd)

11.7.3 Absolute Maximum Ratings (Electrical Sensitivity)

Based on three different tests (ESD and LU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. For more details, refer to the application note AN1181.

11.7.3.1 Electro-Static Discharge (ESD)

Electro-Static Discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends on the number of supply pins in the device (3 parts*(n+1) supply pin). Two models can be simulated: Human Body Model and Machine Model. This test conforms to the JESD22-A114A/A115A standard.

Absolute Maximum Ratings

Symbol	Ratings	Conditions	Maximum value [1]	Unit
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)	T_A=+25 °C	2000	V

Notes:

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

11.7.3.2 Static and Dynamic Latch-Up

- **LU**: 2 complementary static tests are required on 6 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard.

Electrical Sensitivities

Symbol	Parameter	Conditions	Class
LU	Static latch-up class	T_A=+125 °C conforming to JESD 78	II level A

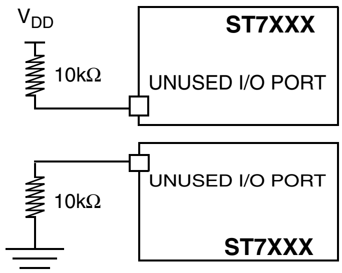
11.8 I/O PORT PIN CHARACTERISTICS

11.8.1 General Characteristics

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

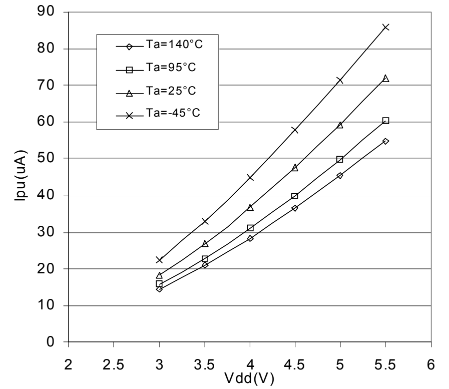
Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IL}	Input low level voltage 1)	CMOS ports			0.3xV_{DD}	V
V_{IH}	Input high level voltage 1)		0.7xV_{DD}			
V_{hys}	Schmitt trigger voltage hysteresis 2)			0.7		
V_{IL}	Input low level voltage 1)	TTL ports			0.8	
V_{IH}	Input high level voltage 1)		2			
V_{hys}	Schmitt trigger voltage hysteresis 2)			1		
$I_{INJ(PIN)}$ 3)	Injected Current on PC6	V_{DD}=5 V	0		+4	mA
	Injected Current on an I/O pin				± 4	
$\Sigma I_{INJ(PIN)}$ 3)	Total injected current (sum of all I/O and control pins)				± 25	
I_L	Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			±1	µA
I_S	Static current consumption	Floating input mode 4)		400		
R_{PU}	Weak pull-up equivalent resistor 5)	V_{IN}=V_{SS} V_{DD}=5 V	50	120	250	kΩ
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time 1)	C_L=50 pF Between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time 1)			25		
$t_{w(IT)in}$	External interrupt pulse time 6)		1			t_{CPU}

Figure 76. Unused I/O Pins configured as input



Note: I/O can be left unconnected if it is configured as output (0 or 1) by the software. This has the advantage of greater EMC robustness and lower cost.

Figure 77. Typical I_{PU} vs. V_{DD} with V_{IN}=V_{SS}



Notes:

1. Data based on characterization results, not tested in production.
2. Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.
3. When the current limitation is not possible, the V_{IN} maximum must be respected, otherwise refer to $I_{INJ(PIN)}$ specification. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$. Refer to section 11.2.2 on page 135 for more details.
4. Configuration not recommended, all unused pins must be kept at a fixed voltage: using the output mode of the I/O for example and leaving the I/O unconnected on the board or an external pull-up or pull-down resistor (see Figure 76). Data based on design simulation and/or technology characteristics, not tested in production.
5. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in Figure 77).
6. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

I/O PORT PIN CHARACTERISTICS (Cont'd)

11.8.2 Output Driving Current

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions		Min	Max	Unit
V_{OL} 1)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time (see Figure 78)	V_{DD}=5 V	I_{IO}=+5 mA		1.2	V
			I_{IO}=+2 mA		0.5	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see Figure 79 and Figure 81)		I_{IO}=+20 mA,T_A≤85 °C T_A≥85 °C		1.3 1.5	
			I_{IO}=+8 mA		0.6	
V_{OH} 2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 80 and Figure 83)		I_{IO}=-5 mA, T_A≤85 °C T_A≥85 °C	V_{DD}-1.4 V_{DD}-1.6		
			I_{IO}=-2 mA	V_{DD}-0.7		

Figure 78. Typical V_{OL} at V_{DD}= 5 V (standard)

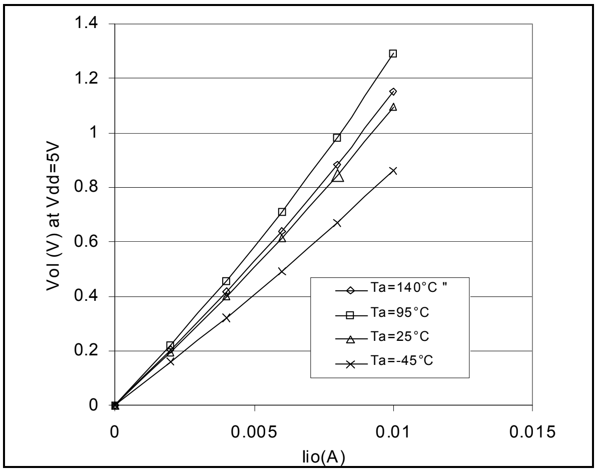


Figure 80. Typical V_{OH} at V_{DD}= 5 V

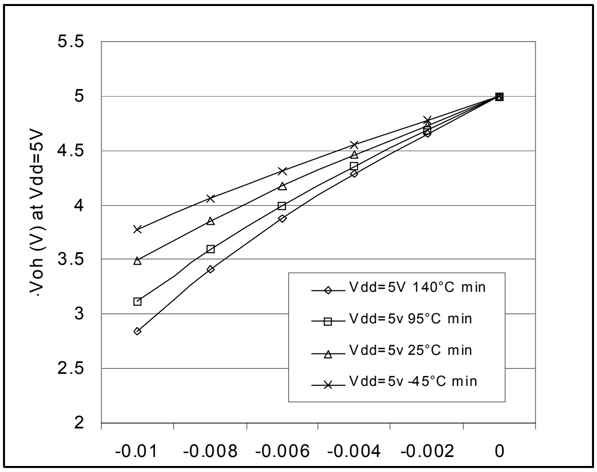
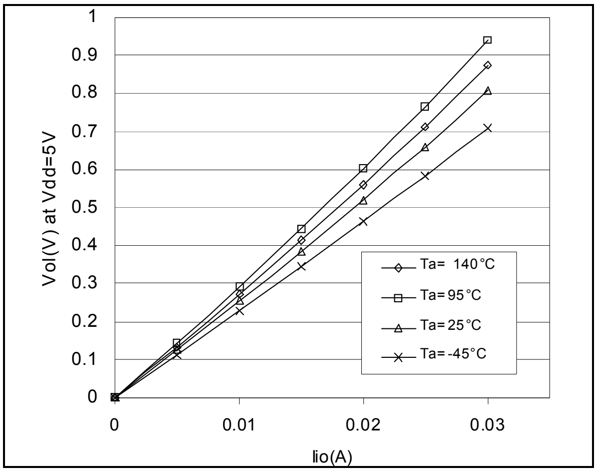


Figure 79. Typical V_{OL} at V_{DD}= 5 V (high-sink)



Notes:

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 11.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.

2. The I_{IO} current sourced must always respect the absolute maximum rating specified in Section 11.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}. True open drain I/O pins do not have V_{OH}.

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 81. Typical V_{OL} vs. V_{DD} (standard)

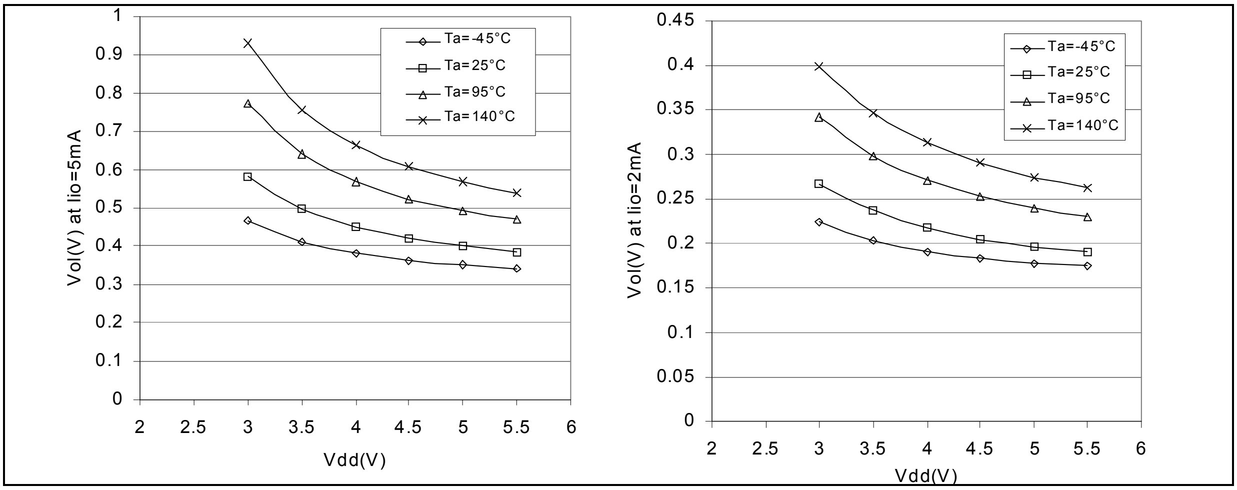


Figure 82. Typical V_{OL} vs. V_{DD} (high-sink)

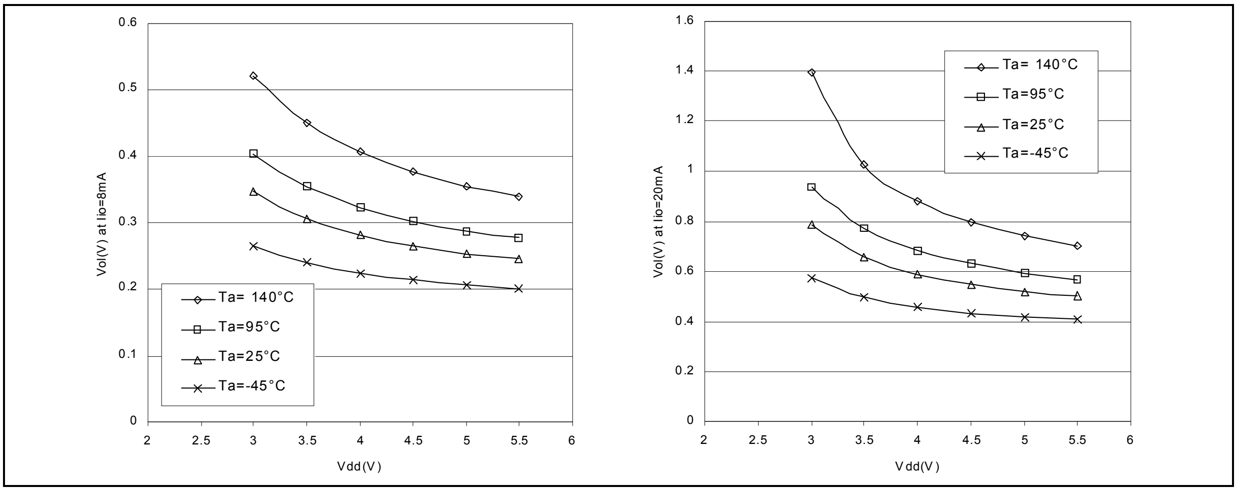
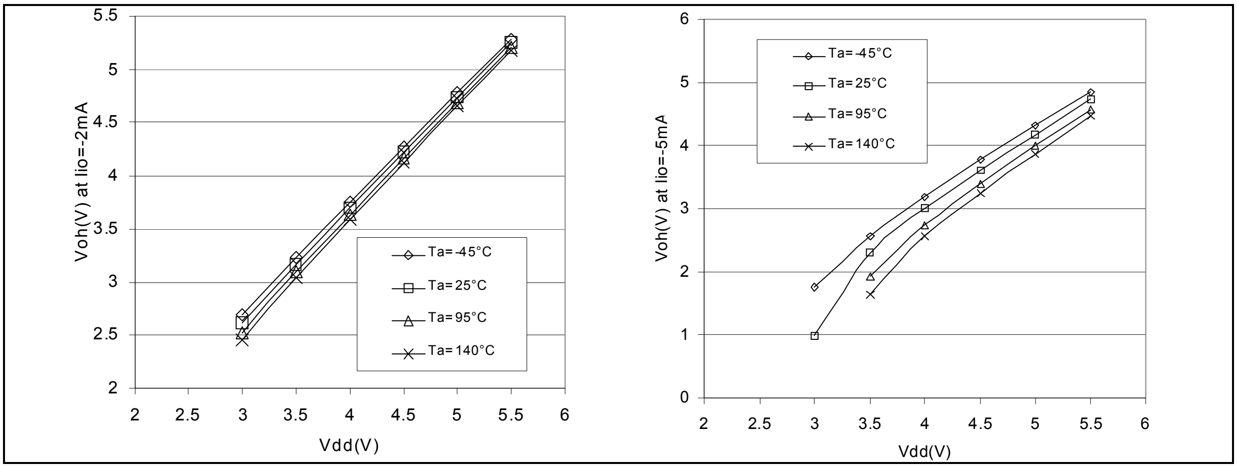


Figure 83. Typical V_{DD}-V_{OH} vs. V_{DD}



11.9 CONTROL PIN CHARACTERISTICS

11.9.1 Asynchronous $\overline{\text{RESET}}$ Pin

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IL}	Input low level voltage [1]				$0.16 \times V_{DD}$	V
V_{IH}	Input high level voltage [1]		$0.85 \times V_{DD}$			
V_{hys}	Schmitt trigger voltage hysteresis [2]			2.5		V
V_{OL}	Output low level voltage [3]	V_{DD}=5 V I_{IO}=+2 mA		0.2	0.5	
I_{IO}	Input current on $\overline{\text{RESET}}$ pin			2		mA
R_{ON}	Weak pull-up equivalent resistor		20	30	120	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Stretch applied on external pulse	0		42[6]	µs
		Internal reset sources	20	30	42[6]	µs
$t_{h(RSTL)in}$	External reset pulse hold time [4]		2.5			µs
$t_{g(RSTL)in}$	Filtered glitch duration [5]			200		ns

Notes:

1. Data based on characterization results, not tested in production.
2. Hysteresis voltage between Schmitt trigger switching levels.
3. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 11.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
4. To guarantee the reset of the device, a minimum pulse has to be applied to the $\overline{\text{RESET}}$ pin. All short pulses applied on the RESET pin with a duration below $t_{h(RSTL)in}$ can be ignored.
5. The reset network (the resistor and two capacitors) protects the device against parasitic resets, especially in noisy environments.
6. Data guaranteed by design, not tested in production.

CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 84. $\overline{\text{RESET}}$ pin protection when LVD is enabled.1)2)3)4)

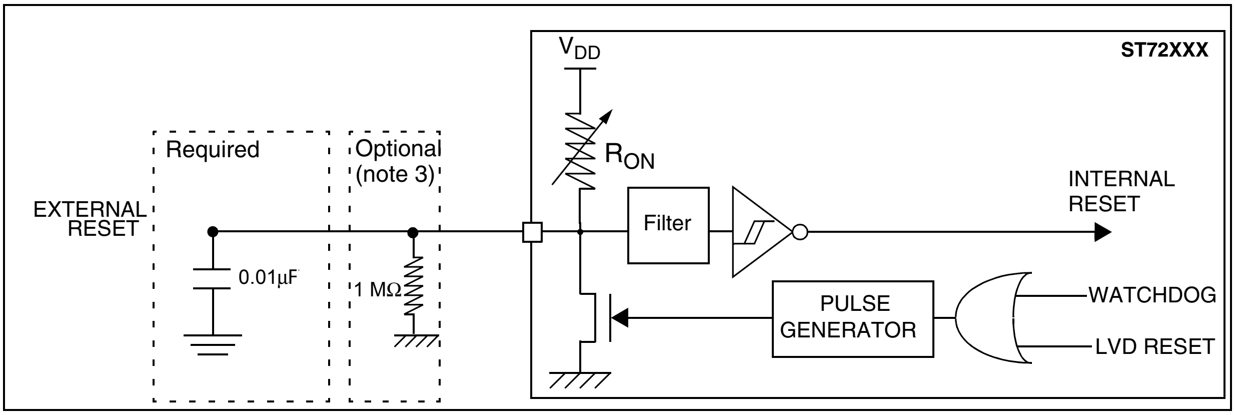
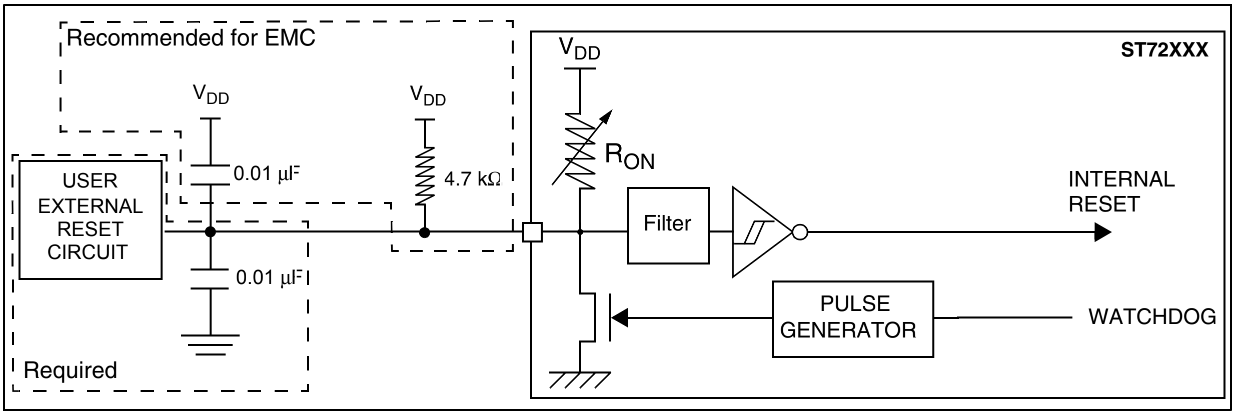

Required
Optional
(note 3)
EXTERNAL
RESET
0.01µF
1 MΩ
V_{DD}
ST72XXX
R_{ON}
Filter
INTERNAL
RESET
PULSE
GENERATOR
WATCHDOG
LVD RESET

Figure 85. $\overline{\text{RESET}}$ pin protection when LVD is disabled.1)


Recommended for EMC
V_{DD}
V_{DD}
0.01 µF
4.7 kΩ
USER
EXTERNAL
RESET
CIRCUIT
0.01 µF
Required
V_{DD}
ST72XXX
R_{ON}
Filter
INTERNAL
RESET
PULSE
GENERATOR
WATCHDOG

Note 1:

- The reset network protects the device against parasitic resets.
- The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).
- Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{\text{RESET}}$ pin can go below the V_{IL} max. level specified in section 11.9.1 on page 154. Otherwise the reset will not be taken into account internally.
- Because the reset circuit is designed to allow the internal RESET to be output in the $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in section 11.2.2 on page 135.

Note 2: When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

Note 3: In case a capacitive power supply is used, it is recommended to connect a 1 MΩ pull-down resistor to the $\overline{\text{RESET}}$ pin to discharge any residual voltage induced by the capacitive effect of the power supply (this will add 5 µA to the power consumption of the MCU).

Note 4: Tips when using the LVD:

- 1. Check that all recommendations related to reset circuit have been applied (see notes above).
- 2. Check that the power supply is properly decoupled (100 nF + 10 µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100 nF + 1 MΩ pull-down on the $\overline{\text{RESET}}$ pin.
- 3. The capacitors connected on the RESET pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the $\overline{\text{RESET}}$ pin with a 5 µF to 20 µF capacitor.

CONTROL PIN CHARACTERISTICS (Cont'd)

11.9.2 ICCSEL/V_{PP} Pin

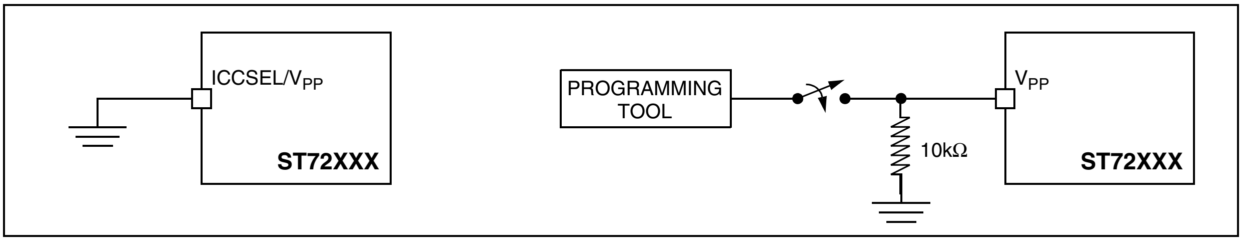
Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max[1]	Unit
V_{IL}	Input low level voltage 1)		V_{SS}	0.2	V
V_{IH}	Input high level voltage 1)		V_{DD}-0.1	12.6	
I_L	Input leakage current	$V_{IN}=V_{SS}$		±1	µA

Note:

1. Data based on design simulation and/or technology characteristics, not tested in production.

Figure 86. Two typical Applications with ICCSEL/V_{PP} Pin 1)



Note:

1. When ICC mode is not required by the application ICCSEL/V_{PP} pin must be tied to V_{SS}.

11.10 TIMER PERIPHERAL CHARACTERISTICS

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 (output compare, input capture, external clock, PWM output...).

11.10.1 8-Bit PWM-ART Auto-Reload Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{res(PWM)}$	PWM resolution time		1			t_{CPU}
		f_{CPU}=8 MHz	125			ns
f_{EXT}	ART external clock frequency		0		$f_{CPU}/2$	MHz
f_{PWM}	PWM repetition rate		0		$f_{CPU}/2$	
Res_{PWM}	PWM resolution				8	bit
V_{OS}	PWM/DAC output step voltage	V_{DD}=5 V, Res=8-bits		20		mV

11.10.2 16-Bit Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{w(ICAP)in}$	Input capture pulse time		1			t_{CPU}
$t_{res(PWM)}$	PWM resolution time		2			t_{CPU}
		f_{CPU}=8 MHz	250			ns
f_{EXT}	Timer external clock frequency		0		$f_{CPU}/4$	MHz
f_{PWM}	PWM repetition rate		0		$f_{CPU}/4$	MHz
Res_{PWM}	PWM resolution				16	bit

11.11 COMMUNICATION INTERFACE CHARACTERISTICS

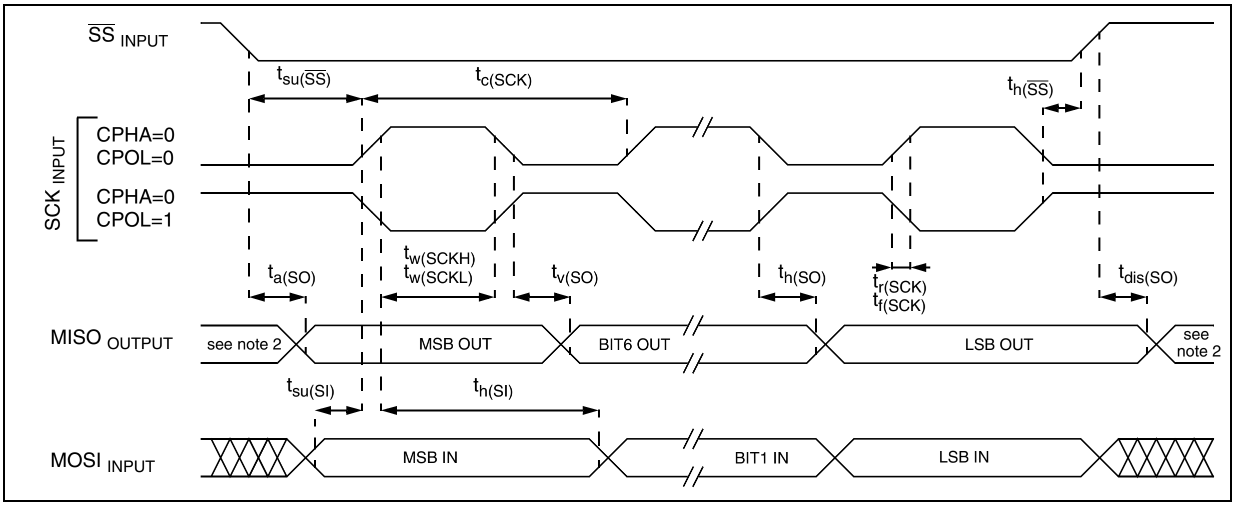
11.11.1 SPI - Serial Peripheral Interface

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics ($\overline{SS}$, SCK, MOSI, MISO).

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCK} $1/t_{c(SCK)}$	SPI clock frequency	Master f_{CPU}=8 MHz	f_{CPU}/128 0.0625	f_{CPU}/4 2	MHz
		Slave f_{CPU}=8 MHz	0	f_{CPU}/2 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see I/O port pin description		
$t_{su(\overline{SS})}$	$\overline{SS}$ setup time	Slave	120		ns
$t_{h(\overline{SS})}$	$\overline{SS}$ hold time	Slave	120		
$t_{w(SCKH)}$ $t_{w(SCKL)}$	SCK high and low time	Master Slave	100 90		
$t_{su(MI)}$ $t_{su(SI)}$	Data input setup time	Master Slave	100 100		
$t_{h(MI)}$ $t_{h(SI)}$	Data input hold time	Master Slave	100 100		
$t_{a(SO)}$	Data output access time	Slave	0	120	
$t_{dis(SO)}$	Data output disable time	Slave		240	
$t_{v(SO)}$	Data output valid time	Slave (after enable edge)		90	
$t_{h(SO)}$	Data output hold time		0		
$t_{v(MO)}$	Data output valid time	Master (before capture edge)	0.25		t_{CPU}
$t_{h(MO)}$	Data output hold time		0.25		

Figure 87. SPI Slave Timing Diagram with CPHA=0 [3)]


$\overline{SS}$ INPUT
$t_{su(\overline{SS})}$
$t_{c(SCK)}$
$t_{h(\overline{SS})}$
SCK INPUT
CPHA=0
CPOL=0
CPHA=0
CPOL=1
$t_{a(SO)}$
$t_{w(SCKH)}$
$t_{w(SCKL)}$
$t_{v(SO)}$
$t_{h(SO)}$
$t_{r(SCK)}$
$t_{f(SCK)}$
$t_{dis(SO)}$
MISO OUTPUT
see note 2
MSB OUT
BIT6 OUT
LSB OUT
see note 2
$t_{su(SI)}$
$t_{h(SI)}$
MOSI INPUT
MSB IN
BIT1 IN
LSB IN

Notes:

1. Data based on design simulation and/or characterisation results, not tested in production.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.
3. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

Figure 88. SPI Slave Timing Diagram with CPHA=1[1)]

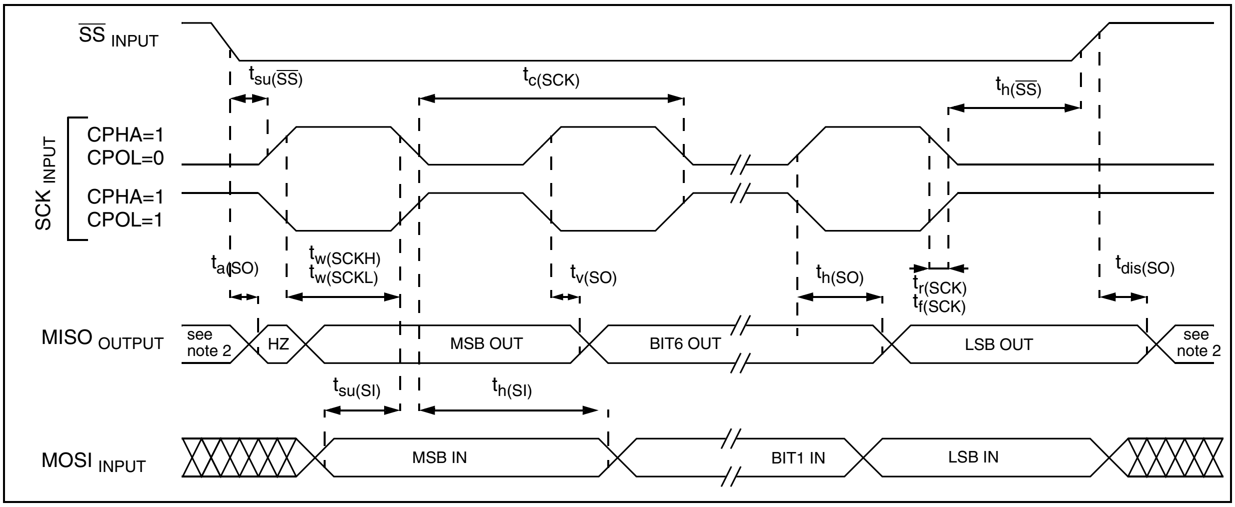
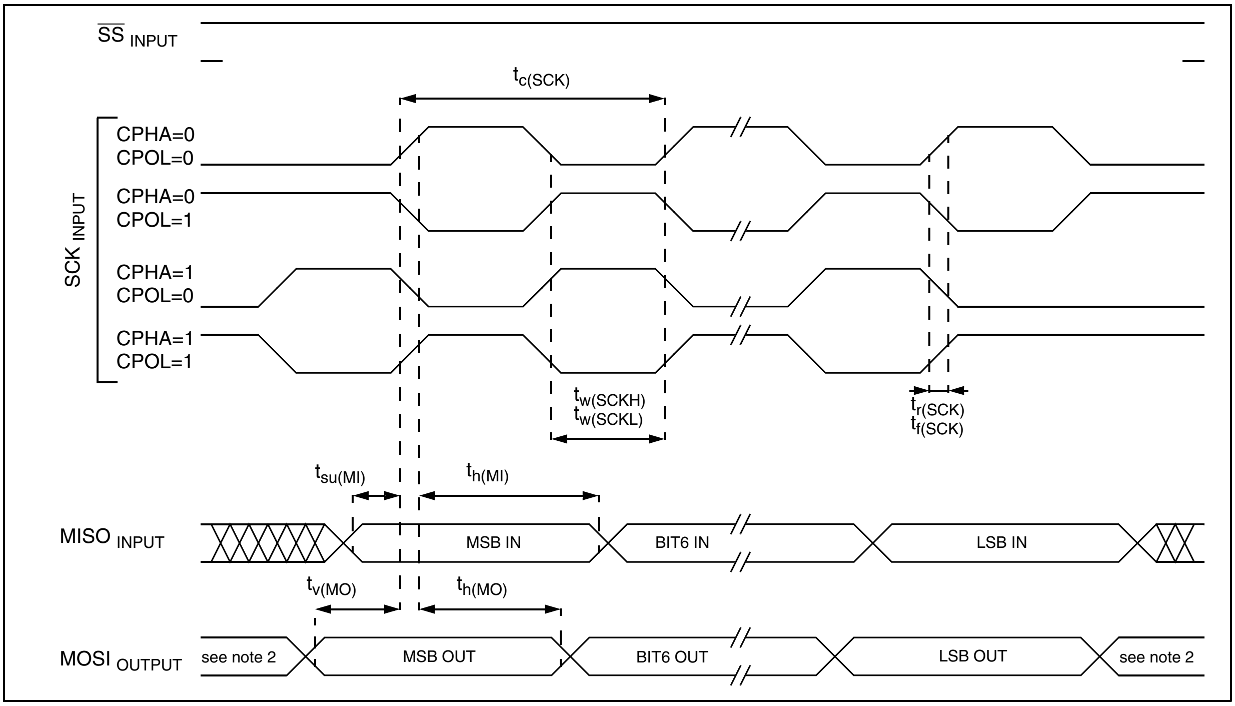

SS INPUT
tsu(SS)
tc(SCK)
th(SS)
SCK INPUT
CPHA=1
CPOL=0
CPHA=1
CPOL=1
ta(SO)
tw(SCKH)
tw(SCKL)
tv(SO)
th(SO)
tr(SCK)
tf(SCK)
tdis(SO)
MISO OUTPUT
see note 2
HZ
MSB OUT
BIT6 OUT
LSB OUT
see note 2
tsu(SI)
th(SI)
MOSI INPUT
MSB IN
BIT1 IN
LSB IN

Figure 89. SPI Master Timing Diagram [1)]


SS INPUT
tc(SCK)
SCK INPUT
CPHA=0
CPOL=0
CPHA=0
CPOL=1
CPHA=1
CPOL=0
CPHA=1
CPOL=1
tw(SCKH)
tw(SCKL)
tr(SCK)
tf(SCK)
tsu(MI)
th(MI)
MISO INPUT
MSB IN
BIT6 IN
LSB IN
tv(MO)
th(MO)
MOSI OUTPUT
see note 2
MSB OUT
BIT6 OUT
LSB OUT
see note 2

Notes:

1. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

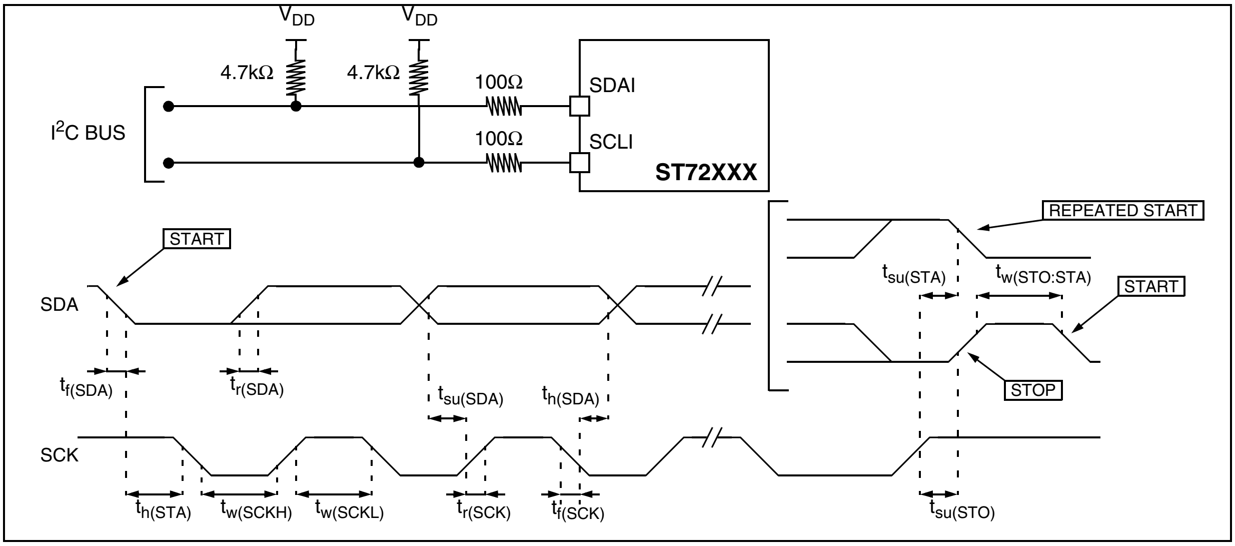
11.11.2 I²C - Inter IC Control Interface

Subject to general operating conditions for V_{DD}, f_{CPU}, 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 I²C interface meets the requirements of the Standard I²C communication protocol described in the following table.

Symbol	Parameter	Standard mode I²C		Fast mode I²C 5)		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		µs
$t_{w(STO:STA)}$	STOP to START condition time (bus free)	4.7		1.3		µs
C_b	Capacitive load for each bus line		400		400	pF

Figure 90. Typical Application with I²C Bus and Timing Diagram 4)



Notes:

1. Data based on standard I²C protocol requirement, not tested in production.
2. The device must internally provide a hold time of at least 300 ns 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 $0.7xV_{DD}$.
5. At 4 MHz f_{CPU}, max.I²C speed (400 kHz) is not achievable. In this case, max. I²C speed will be approximately 260 kHz.

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

The following table gives the values to be written in the I2CCCR register to obtain the required I^2C SCL line frequency.

Table 26. SCL Frequency Table

f_{SCL} (kHz)	I2CCCR Value							
	f_{CPU}=4 MHz.				f_{CPU}=8 MHz.			
	V_{DD} = 4.1 V		V_{DD} = 5 V		V_{DD} = 4.1 V		V_{DD} = 5 V	
	R_P=3.3 kΩ	R_P=4.7 kΩ	R_P=3.3 kΩ	R_P=4.7 kΩ	R_P=3.3 kΩ	R_P=4.7 kΩ	R_P=3.3 kΩ	R_P=4.7 kΩ
400	NA	NA	NA	NA	83h	83	83h	83h
300	NA	NA	NA	NA	85h	85h	85h	85h
200	83h	83h	83h	83h	8Ah	89h	8Ah	8Ah
100	10h	10h	10h	10h	24h	23h	24h	23h
50	24h	24h	24h	24h	4Ch	4Ch	4Ch	4Ch
20	5Fh	5Fh	5Fh	5Fh	FFh	FFh	FFh	FFh

Legend:

R_P = External pull-up resistance

f_{SCL} = I^2C speed

NA = Not achievable

Note:

– For speeds around 200 kHz, achieved speed can have ±5% tolerance

– For other speed ranges, achieved speed can have ±2% tolerance

The above variations depend on the accuracy of the external components used.

11.12 10-BIT ADC CHARACTERISTICS

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{ADC}	ADC clock frequency		0.4		2	MHz
V_{AREF}	Analog reference voltage	$0.7*V_{DD} \le V_{AREF} \le V_{DD}$	3.8		V_{DD}	V
V_{AIN}	Conversion voltage range 1)		V_{SSA}		V_{AREF}	
I_{lkg}	Positive input leakage current for analog input	-40 °C≤T_A≤ 85 °C range			±250	nA
		Other T_A ranges			±1	µA
	Negative input leakage current on robust analog pins [2]	$V_{IN}<V_{SS}$, $\lvert I_{IN} \rvert < 400$ µA on adjacent robust analog pin	See note 2			
R_{AIN}	External input impedance				see Figure 91 and Figure 92	kΩ
C_{AIN}	External capacitor on analog input					pF
f_{AIN}	Variation freq. of analog input signal					Hz
C_{ADC}	Internal sample and hold capacitor			12		pF
t_{ADC}	Conversion time (Sample+Hold) f_{CPU}=8 MHz, SPEED=0 f_{ADC}=2 MHz		7.5			µs
t_{ADC}	- No of sample capacitor loading cycles - No. of Hold conversion cycles		4 11			$1/f_{ADC}$

Notes:

1. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10 kΩ). Data based on characterization results, not tested in production.

2. Injecting negative current on any of the analog input pins significantly reduces the accuracy of any conversion being performed on any analog input. Analog pins of Flash devices can be protected against negative injection by adding a Schottky diode (pin to ground). Injecting negative current on digital input pins degrades ADC accuracy especially if performed on a pin close to the analog input pins. Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in Section 11.8 does not affect the ADC accuracy.

ADC CHARACTERISTICS (Cont'd)

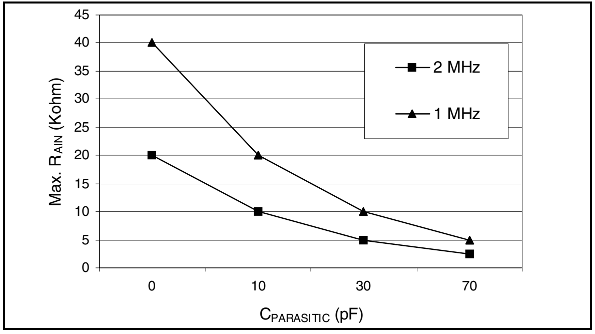


Figure 91. R_{AIN} max. vs f_{ADC} with C_{AIN}=0pF[1)]

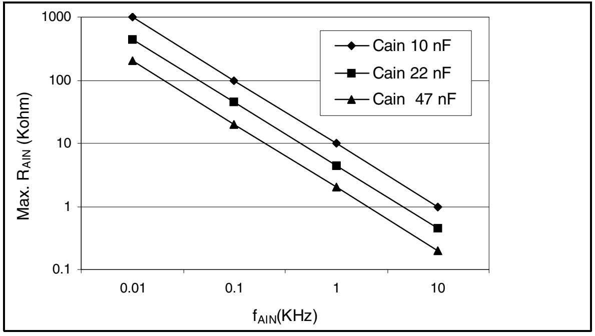
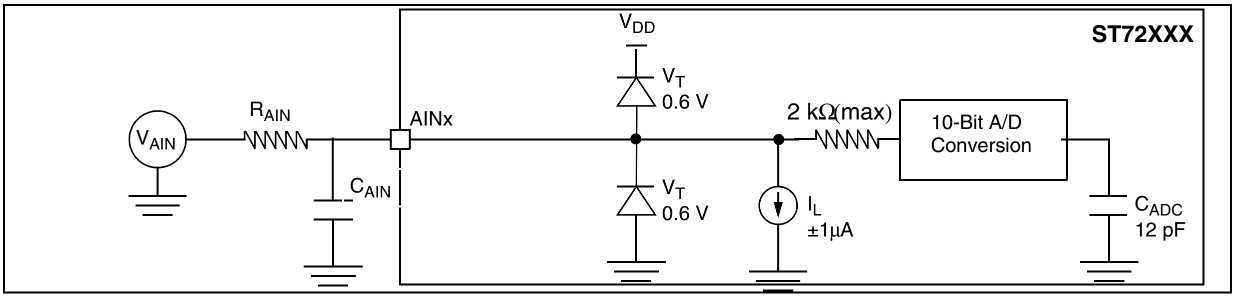


Figure 92. Recommended C_{AIN} & R_{AIN} values.[2)]

Figure 93. Typical A/D Converter Application


ST72XXX
VDD
VT
0.6 V
VT
0.6 V
RAIN
VAIN
CAIN
AINx
2 kΩ(max)
10-Bit A/D Conversion
IL
±1µA
CADC
12 pF

Notes:

1. $C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3 pF). A high $C_{PARASITIC}$ value will downgrade conversion accuracy. To remedy this, f_{ADC} should be reduced.

2. This graph shows that depending on the input signal variation (f_{AIN}), C_{AIN} can be increased for stabilization time and decreased to allow the use of a larger serial resistor (R_{AIN}).

ADC CHARACTERISTICS (Cont'd)

11.12.1 Analog Power Supply and Reference Pins

Depending on the MCU pin count, the package may feature separate V_{AREF} and V_{SSA} analog power supply pins. These pins supply power to the A/D converter cell and function as the high and low reference voltages for the conversion.

Separation of the digital and analog power pins allow board designers to improve A/D performance. Conversion accuracy can be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines (see Section 11.12.2 General PCB Design Guidelines).

11.12.2 General PCB Design Guidelines

To obtain best results, some general design and layout rules should be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

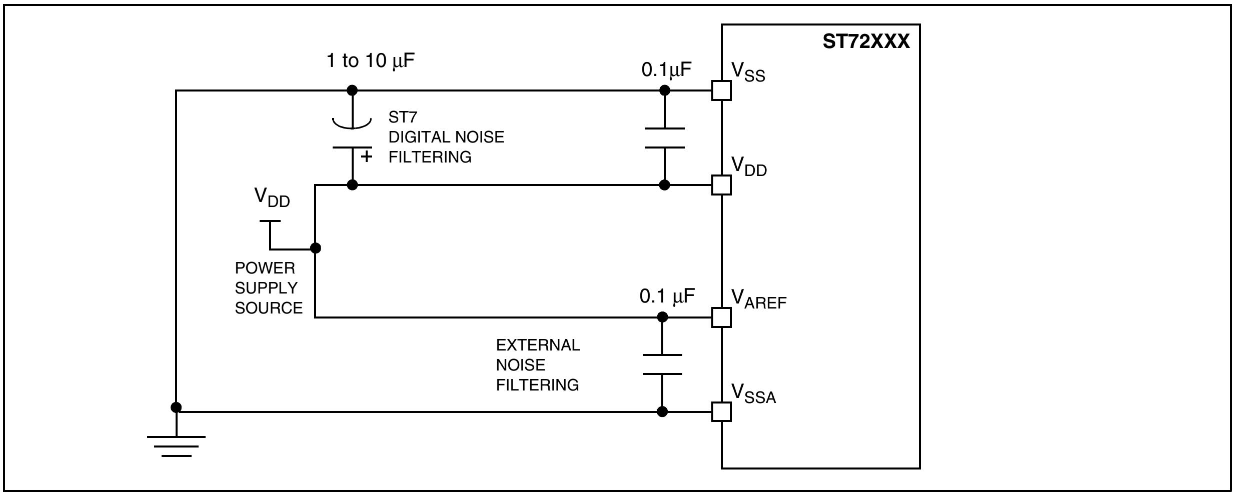
– Use separate digital and analog planes. The analog ground plane should be connected to the digital ground plane via a single point on the PCB.

– Filter power to the analog power planes. It is recommended to connect capacitors, with good high frequency characteristics, between the power and ground lines, placing 0.1µF and optionally, if needed 10 pF capacitors as close as possible to the ST7 power supply pins and a 1 to 10µF capacitor close to the power source (see Figure 94).

– The analog and digital power supplies should be connected in a star network. Do not use a resistor, as V_{AREF} is used as a reference voltage by the A/D converter and any resistance would cause a voltage drop and a loss of accuracy.

– Properly place components and route the signal traces on the PCB to shield the analog inputs. Analog signals paths should run over the analog ground plane and be as short as possible. Isolate analog signals from digital signals that may switch while the analog inputs are being sampled by the A/D converter. Do not toggle digital outputs on the same I/O port as the A/D input being converted.

Figure 94. Power Supply Filtering



10-BIT ADC CHARACTERISTICS (Cont'd)

11.12.3 ADC Accuracy

Conditions: V_{DD}=5 V 1)

Symbol	Parameter	Conditions	Typ	Max2)	Unit
$\lvert E_T \rvert$	Total unadjusted error 1)		3	4	LSB
$\lvert E_O \rvert$	Offset error 1)		2	3	
$\lvert E_G \rvert$	Gain Error 1)		0.5	3	
$\lvert E_D \rvert$	Differential linearity error 1)	CPU in run mode @ f_{ADC} 2 MHz.	1	2	
$\lvert E_L \rvert$	Integral linearity error 1)	CPU in run mode @ f_{ADC} 2 MHz.	1	2	

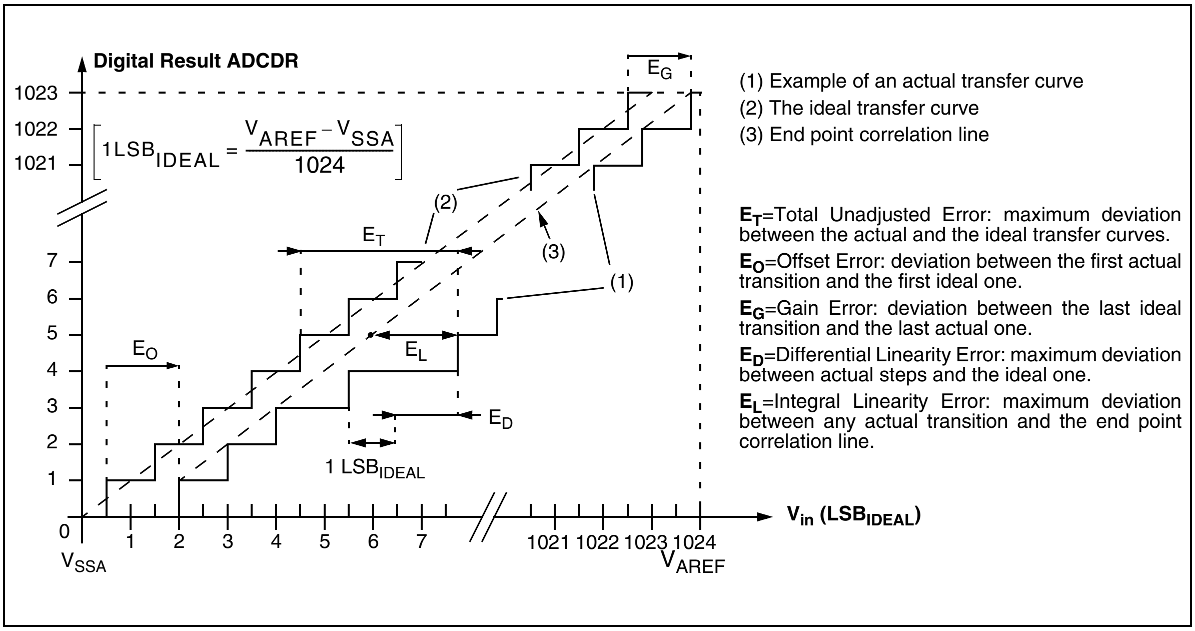
Notes:

1. ADC Accuracy vs. Negative Injection Current: Injecting negative current may reduce the accuracy of the conversion being performed on another analog input. The effect of negative injection current on robust pins is specified in Section 11.12.

Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in Section 11.8 does not affect the ADC accuracy.

2. Data based on characterization results, monitored in production to guarantee 99.73 % within ± max value from -40 °C to 125 °C (± 3σ distribution limits).

Figure 95. ADC Accuracy Characteristics



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

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

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

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

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

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

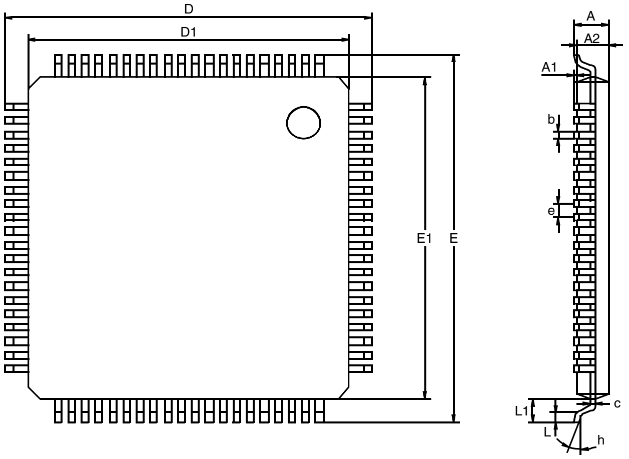
12 PACKAGE CHARACTERISTICS

12.1 ECOPACK

In order to meet environmental requirements, ST offers these devices in different grades of ECOPACK® packages, depending on their level of environmental compliance. ECOPACK® specifications, grade definitions and product status are available at: www.st.com. ECOPACK® is an ST trademark.

12.2 PACKAGE MECHANICAL DATA

Figure 96. 80-Pin Low Profile Quad Flat Package



Dim.	mm			inches[1)]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.22	0.32	0.38	0.0087	0.0126	0.0150
C	0.09		0.20	0.0035		0.0079
D		16.00			0.6299	
D1		14.00			0.5512	
E		16.00			0.6299	
E1		14.00			0.5512	
e		0.65			0.0256	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	80					

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

12.3 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient) TQFP80 14x14	55	°C/W
P_D	Power dissipation [1)]	500	mW
T_{Jmax}	Maximum junction temperature [2)]	150	°C

Notes:

1. The power dissipation is obtained from the formula $P_D=P_{INT}+P_{PORT}$ where P_{INT} is the chip internal power ($I_{DD}xV_{DD}$) and P_{PORT} is the port power dissipation determined by the user.
2. The average chip-junction temperature can be obtained from the formula $T_J = T_A + P_D$ x RthJA.

13 ST72321Mx DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in user programmable versions (Flash).

Flash devices are shipped to customers with a default content. This implies that Flash devices have to be configured by the customer using the Option Bytes.

13.1 FLASH OPTION BYTES

	STATIC OPTION BYTE 0								STATIC OPTION BYTE 1							
	7							0	7							0
	WDG		Res.	VD		Reserved	PKG0	FMP_R	PKG1	RSTC	OSCTYPE		OSCRANGE			PLLOFF
	HALT	SW		1	0						1	0	2	1	0	
Default	1	1	1	0	0	1	1	1	1	1	1	0	1	1	1	1

The option bytes allow the hardware configuration of the microcontroller to be selected. They have no address in the memory map and can be accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the Flash is fixed to FFh. To program the Flash devices directly using ICP, Flash devices are shipped to customers with the internal RC clock source.

OPTION BYTE 0

OPT7= **WDG HALT** *Watchdog and Halt mode*
This option bit determines if a RESET is generated when entering Halt mode while the Watchdog is active.
0: No Reset generation when entering Halt mode
1: Reset generation when entering Halt mode

OPT6= **WDG SW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware (watchdog always enabled)
1: Software (watchdog to be enabled by software)

OPT5 = Reserved, must be kept at default value.

OPT4:3= **VD[1:0]** *Voltage detection*
These option bits enable the voltage detection block (LVD, and AVD) with a selected threshold for the LVD and AVD (EVD+AVD).

Selected Low Voltage Detector	VD1	VD0
LVD and AVD Off	1	1
Lowest Threshold: (V_{DD}~3 V)	1	0
Med. Threshold (V_{DD}~3.5 V)	0	1
Highest Threshold (V_{DD}~4 V)	0	0

Caution: If the medium or low thresholds are selected, the detection may occur outside the specified operating voltage range. Below 3.8V, device operation is not guaranteed.

ST72321Mx DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

OPT2 = Reserved, must be kept at default value.

OPT1= **PKG0** *Package selection bit 0*
This option bit is used to select the package (see table in PKG1 option bit description).

OPT0= **FMP_R** *Flash memory read-out protection*
Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory.

Erasing the option bytes when the FMP_R option is selected causes the whole user memory to be erased first, and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual for more details.
Note: Readout protection is not supported if LVD is enabled.
0: Read-out protection enabled
1: Read-out protection disabled

OPTION BYTE 1

OPT7= **PKG1** *Package selection bit 1*
This option bit, with the PKG0 bit, selects the package.

Version	Selected Package	PKG 1	PKG 0
M	LQFP80	1	1

Note: On the chip, each I/O port has 8 pads. Pads that are not bonded to external pins are in input pull-up configuration after reset. The configuration of these pads must be kept at reset state to avoid added current consumption.

OPT6 = **RSTC** *RESET clock cycle selection*
This option bit selects the number of CPU cycles applied during the RESET phase and when exiting Halt mode. For resonator oscillators, it is advised to select 4096 due to the long crystal stabilization time.
0: Reset phase with 4096 CPU cycles
1: Reset phase with 256 CPU cycles

OPT5:4 = **OSCTYPE[1:0]** *Oscillator Type*
These option bits select the ST7 main clock source type.

Clock Source	OSCTYPE	
	1	0
Resonator Oscillator	0	0
Reserved	0	1
Internal RC Oscillator	1	0
External Source	1	1

OPT3:1 = **OSCRANGE[2:0]** *Oscillator range*
When the resonator oscillator type is selected, these option bits select the resonator oscillator current source corresponding to the frequency range of the used resonator. Otherwise, these bits are used to select the normal operating frequency range.

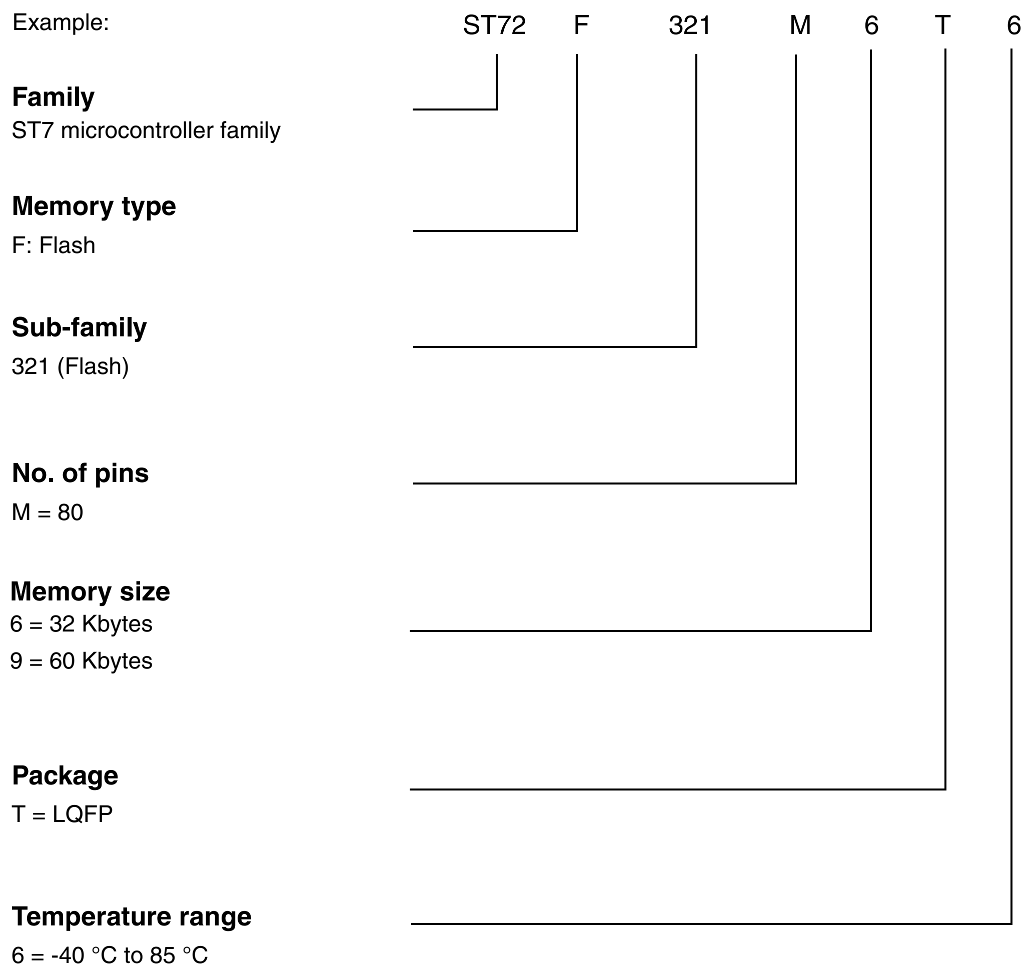
Typ. Freq. Range		OSCRANGE		
		2	1	0
LP	1~2 MHz	0	0	0
MP	2~4 MHz	0	0	1
MS	4~8 MHz	0	1	0
HS	8~16 MHz	0	1	1

OPT0 = **PLLOFF** *PLL activation*
This option bit activates the PLL which allows multiplication by two of the main input clock frequency. The PLL must not be used with the internal RC oscillator or with external clock source. The PLL is guaranteed only with an input frequency between 2 and 4 MHz.
0: PLL x2 enabled
1: PLL x2 disabled
CAUTION: the PLL can be enabled only if the "OSC RANGE" (OPT3:1) bits are configured to "MP - 2~4 MHz". Otherwise, the device functionality is not guaranteed.

ST72321xx DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

13.2 DEVICE ORDERING INFORMATION

Figure 97. Ordering information scheme


Example:
ST72
F
321
M
6
T
6
Family
ST7 microcontroller family
Memory type
F: Flash
Sub-family
321 (Flash)
No. of pins
M = 80
Memory size
6 = 32 Kbytes
9 = 60 Kbytes
Package
T = LQFP
Temperature range
6 = -40 °C to 85 °C

For a list of available options (e.g. memory size, package) and orderable part numbers or for further information on any aspect of this device, please contact the ST Sales Office nearest to you.

14 KNOWN LIMITATIONS

14.1 Safe Connection of OSC1/OSC2 Pins

The OSC1 and/or OSC2 pins must not be left unconnected otherwise the ST7 main oscillator may start and, in this configuration, could generate an f_{OSC} clock frequency in excess of the allowed maximum (>16 MHz.), putting the ST7 in an unsafe/undefined state. Refer to section 6.2 on page 24.

14.2 Reset pin protection with LVD Enabled

As mentioned in note 2 below Figure 84 on page 155, when the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10 nF pull-down capacitor is required to filter noise on the reset line.

14.3 Unexpected Reset Fetch

If an interrupt request occurs while a “POP CC” instruction is executed, the interrupt controller does not recognise the source of the interrupt and, by default, passes the RESET vector address to the CPU.

Workaround

To solve this issue, a “POP CC” instruction must always be preceded by a “SIM” instruction.

14.4 External interrupt missed

To avoid any risk if generating a parasitic interrupt, the edge detector is automatically disabled for one clock cycle during an access to either DDR and OR. Any input signal edge during this period will not be detected and will not generate an interrupt.

This case can typically occur if the application refreshes the port configuration registers at intervals during runtime.

Workaround

The workaround is based on software checking the level on the interrupt pin before and after writing to the PxOR or PxDDR registers. If there is a level change (depending on the sensitivity programmed for this pin) the interrupt routine is invoked using the call instruction with three extra PUSH instructions before executing the interrupt routine (this is to make the call compatible with the IRET instruction at the end of the interrupt service routine).

But detection of the level change does not make sure that edge occurs during the critical 1 cycle duration and the interrupt has been missed. This may lead to occurrence of same interrupt twice (one hardware and another with software call).

To avoid this, a semaphore is set to '1' before checking the level change. The semaphore is changed to level '0' inside the interrupt routine. When a level change is detected, the semaphore status is checked and if it is '1' this means that the last interrupt has been missed. In this case, the interrupt routine is invoked with the call instruction.

There is another possible case i.e. if writing to PxOR or PxDDR is done with global interrupts disabled (interrupt mask bit set). In this case, the semaphore is changed to '1' when the level change is detected. Detecting a missed interrupt is done after the global interrupts are enabled (interrupt mask bit reset) and by checking the status of the semaphore. If it is '1' this means that the last interrupt was missed and the interrupt routine is invoked with the call instruction.

To implement the workaround, the following software sequence is to be followed for writing into the PxOR/PxDDR registers. The example is for for Port PF1 with falling edge interrupt sensitivity. The software sequence is given for both cases (global interrupt disabled/enabled).

Case 1: Writing to PxOR or PxDDR with Global Interrupts Enabled:

```
LD A,#01
LD sema,A      ; set the semaphore to '1'
LD A,PFDR
AND A,#02
LD  X,A        ; store the level before writing to PxOR/PxDDR
LD A,#$90
LD PFDDR,A  ; Write to PFDDR
LD A,#$ff
LD  PFOR,A     ; Write to PFOR
LD A,PFDR
AND A,#02
LD Y,A         ; store the level after writing to PxOR/PxDDR
LD A,X         ; check for falling edge
cp A,#02
jrne OUT
TNZ Y
jrne OUT
```

```
LD A,sema     ; check the semaphore status if edge is detected
CP A,#01
jrne OUT
call call_routine; call the interrupt routine
OUT:LD A,#00
LD sema,A
.call_routine     ; entry to call_routine
PUSH A
PUSH X
PUSH CC
.ext1_rt          ; entry to interrupt routine
LD A,#00
LD sema,A
IRET
```

Case 2: Writing to PxOR or PxDDR with Global Interrupts Disabled:

```
SIM           ; set the interrupt mask
LD A,PFDR
AND A,#$02
LD X,A        ; store the level before writing to PxOR/PxDDR
LD A,#$90
LD PFDDR,A; Write into PFDDR
LD A,#$ff
LD PFOR,A           ; Write to PFOR
LD A,PFDR
AND A,#$02
LD Y,A        ; store the level after writing to PxOR/PxDDR
LD A,X        ; check for falling edge
cp A,#$02
jrne OUT
TNZ Y
jrne OUT
LD A,#$01
LD sema,A  ; set the semaphore to '1' if edge is detected
RIM            ; reset the interrupt mask
LD A,sema  ; check the semaphore status
CP A,#$01
jrne OUT
call call_routine; call the interrupt routine
RIM
OUT:          RIM
JP while_loop
.call_routine ; entry to call_routine
PUSH A
PUSH X
PUSH CC
.ext1_rt         ; entry to interrupt routine
LD A,#$00
LD sema,A
IRET
```

14.5 Clearing active interrupts outside interrupt routine

When an active interrupt request occurs at the same time as the related flag is being cleared, an unwanted reset may occur.

Note: clearing the related interrupt mask will not generate an unwanted reset

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, i.e.

when:

- The interrupt flag is cleared within its own interrupt routine
- The interrupt flag is cleared within any interrupt routine
- The interrupt flag is cleared in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request.

Example:

```
  SIM
  reset interrupt flag
  RIM
```

Nested interrupt context:

The symptom does not occur when the interrupts are handled normally, i.e.

when:

- The interrupt flag is cleared within its own interrupt routine
- The interrupt flag is cleared within any interrupt routine with higher or identical priority level

– The interrupt flag is cleared in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

PUSH CC

SIM

reset interrupt flag

POP CC

14.6 SCI Wrong Break duration

Description

A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 20 bits instead of 10 bits if M=0
- 22 bits instead of 11 bits if M=1.

In the same way, as long as the SBK bit is set, break characters are sent to the TDO pin. This may lead to generate one break more than expected.

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (f_{CPU}=8 MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Workaround

If this wrong duration is not compliant with the communication protocol in the application, software can request that an Idle line be generated before the break character. In this case, the break duration is always correct assuming the application is not doing anything between the idle and the break. This can be ensured by temporarily disabling interrupts.

The exact sequence is:

- Disable interrupts
- Reset and Set TE (IDLE request)
- Set and Reset SBK (Break Request)
- Re-enable interrupts

14.7 16-bit Timer PWM Mode

In PWM mode, the first PWM pulse is missed after writing the value FFFCh in the OC1R register (OC1HR, OC1LR). It leads to either full or no PWM during a period, depending on the OLVL1 and OLVL2 settings.

14.8 TIMD set simultaneously with OC interrupt

If the 16-bit timer is disabled at the same time the output compare event occurs then output compare flag gets locked and cannot be cleared before the timer is enabled again.

Impact on the application

If output compare interrupt is enabled, then the output compare flag cannot be cleared in the timer interrupt routine. Consequently the interrupt service routine is called repeatedly.

Workaround

Disable the timer interrupt before disabling the timer. Again while enabling, first enable the timer then the timer interrupts.

Perform the following to disable the timer:

TACR1 or TBCR1 = 0x00h; // Disable the compare interrupt

TACSR | or TBCSR | = 0x40; // Disable the timer

Perform the following to enable the timer again:

TACSR & or TBCSR &= ~0x40; // Enable the timer

TACR1 or TBCR1 = 0x40; // Enable the compare interrupt

14.9 I2C Multimaster

In multimaster configurations, if the ST7 I2C receives a START condition from another I2C master after the START bit is set in the I2CCR register and before the START condition is generated by the ST7 I2C, it may ignore the START condition from the other I2C master. In this case, the ST7 master will receive a NACK from the other device. On reception of the NACK, ST7 can send a re-start and Slave address to re-initiate communication

14.10 Internal RC Oscillator with LVD

The internal RC can only be used if LVD is enabled.

14.11 I/O behaviour during ICC mode entry sequence

Symptom

Both Port G and H are forced to output push-pull during ICC mode entry sequence.

Details

To enable programming of all Flash sectors, the device must leave USER mode and be configured in ICC mode. Once in ICC mode, the ICC protocol enables an ST7 microcontroller to communicate with an external controller (such as a PC). ICC mode is entered by applying 39 pulses on the ICCDATA signal during reset. To enter ICC mode, the device goes through other modes, some modes are critical because the I/Os PG[7:0] and PH[7:0] are forced to output push-pull.

Impact on the Application

The PG and PH I/O ports are forced to output push-pull during three pulses on ICCDATA. In certain circumstances, this behaviour can lead to a short-circuit between the I/O signals and V_{DD}, V_{SS} or an output signal of another application component.

In addition, switching these I/Os to output mode can cause the application to leave reset state, disturbing the ICC communication and preventing the user from programming the Flash.

14.12 Read-out protection with LVD

The LVD is not supported if Readout protection is enabled.

15 REVISION HISTORY

Table 27. Revision History

Date	Revision	Description of Changes
26-Sep-2006	1	Initial release
04-May-2009	2	Modified title of the document Removed references to ROM devices Modified HDFlash endurance and data retention on first page Modified Table 1 on page 1 Added note 2 to Table 2 on page 9 Modified “Starting the Conversion” on page 125 Modified t_{RET} and N_{RW} values in “Flash memory” on page 147 Modified “Absolute Maximum Ratings (Electrical Sensitivity)” on page 150 (removed $V_{ESD(MM)}$) Modified “Electro Magnetic Interference (EMI)” on page 149 Modified conditions for V_{FESD} and V_{FFTB} in “Functional EMS (Electro Magnetic Susceptibility)” on page 148 (LQFP64 added) Added Section 12.1 “ECOPACK” on page 166 Values in inches rounded to 4 decimal digits (instead of 3) in “PACKAGE MECHANICAL DATA” on page 166 Added “TIMD set simultaneously with OC interrupt” on page 172 Modified Section 13 ST72321Mx DEVICE CONFIGURATION AND ORDERING INFORMATION Removed products with a -40 to 125 °C temperature range Removed option list

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2009 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com