



# ST72521M/R/AR

## 8-BIT MCU WITH NESTED INTERRUPTS, FLASH, 10-BIT ADC, FIVE TIMERS, SPI, SCI, I<sup>2</sup>C, CAN INTERFACE

### ■ Memories

- 32K to 60K dual voltage High Density Flash (HDFlash) or ROM with read-out protection capability. In-Application Programming and In-Circuit Programming for HDFlash devices
- 1K to 2K RAM
- HDFlash endurance: 100 cycles, data retention: 20 years at 55°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 or external RC oscillator, clock security system and bypass for external clock
- PLL for 2x frequency multiplication
- Four power saving modes: Halt, Active-Halt, Wait and Slow

### ■ Interrupt Management

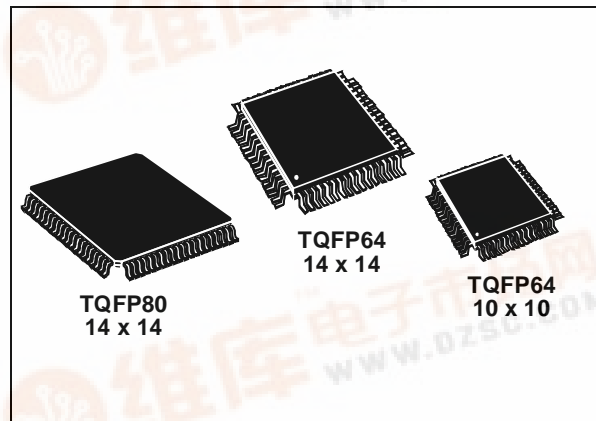
- Nested interrupt controller
- 14 interrupt vectors plus TRAP and RESET
- Top Level Interrupt (TLI) pin
- 15 external interrupt lines (on 4 vectors)

### ■ Up to 64 I/O Ports

- 48 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

### ■ 4 Communications Interfaces

- SPI synchronous serial interface
- SCI asynchronous serial interface (LIN compatible)
- I<sup>2</sup>C multimaster interface
- CAN interface (2.0B Passive)

### ■ Analog peripheral

- 10-bit ADC with 16 input pins

### ■ 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

### Device Summary

| Features               | ST72(F)521(M/R/AR)9                                   | ST72521(R/AR)7                      | ST72(F)521(R/AR)6      |
|------------------------|---|-------------------------------------|------------------------|
| Program memory - bytes | 60K   | 48K                                 | 32K                    |
| RAM (stack) - bytes    | 2048 (256)  | 1536 (256)                          | 1024 (256)             |
| Operating Voltage      | 3.8V to 5.5V  |                                     |                        |
| Temp. Range (ROM)      | up to -40°C to +125°C                                 |                                     |                        |
| Temp. Range (Flash)    | up to -40°C to +125°C                                 | N/A                                 | up to -40°C to +125 °C |
| Package                | TQFP80 14x14 (M), TQFP64 14x14 (R), TQFP64 10x10 (AR) | TQFP64 14x14 (R), TQFP64 10x10 (AR) |                        |

---

## Table of Contents

---

|   |           |
|---|-----------|
| <b>1 INTRODUCTION</b>                           | <b>7</b>  |
| <b>2 PIN DESCRIPTION</b>                        | <b>8</b>  |
| <b>3 REGISTER &amp; MEMORY MAP</b>              | <b>14</b> |
| <b>4 FLASH PROGRAM MEMORY</b>                   | <b>18</b> |
| 4.1 INTRODUCTION                                | 18        |
| 4.2 MAIN FEATURES                               | 18        |
| 4.3 STRUCTURE                                   | 18        |
| 4.3.1 Read-out Protection                       | 18        |
| 4.4 ICC INTERFACE                               | 19        |
| 4.5 ICP (IN-CIRCUIT PROGRAMMING)                | 20        |
| 4.6 IAP (IN-APPLICATION PROGRAMMING)            | 20        |
| 4.6.1 Register Description                      | 20        |
| <b>5 CENTRAL PROCESSING UNIT</b>                | <b>21</b> |
| 5.1 INTRODUCTION                                | 21        |
| 5.2 MAIN FEATURES                               | 21        |
| 5.3 CPU REGISTERS                               | 21        |
| <b>6 SUPPLY, RESET AND CLOCK MANAGEMENT</b>     | <b>24</b> |
| 6.1 PHASE LOCKED LOOP                           | 24        |
| 6.2 MULTI-OSCILLATOR (MO)                       | 25        |
| 6.3 RESET SEQUENCE MANAGER (RSM)                | 26        |
| 6.3.1 Introduction                              | 26        |
| 6.3.2 Asynchronous External RESET pin           | 26        |
| 6.3.3 External Power-On RESET                   | 27        |
| 6.3.4 Internal Low Voltage Detector (LVD) RESET | 27        |
| 6.3.5 Internal Watchdog RESET                   | 27        |
| 6.4 SYSTEM INTEGRITY MANAGEMENT (SI)            | 28        |
| 6.4.1 Low Voltage Detector (LVD)                | 28        |
| 6.4.2 Auxiliary Voltage Detector (AVD)          | 29        |
| 6.4.3 Clock Security System (CSS)               | 31        |
| 6.4.4 Low Power Modes                           | 31        |
| 6.4.5 Register Description                      | 32        |
| <b>7 INTERRUPTS</b>                             | <b>33</b> |
| 7.1 INTRODUCTION                                | 33        |
| 7.2 MASKING AND PROCESSING FLOW                 | 33        |
| 7.3 INTERRUPTS AND LOW POWER MODES              | 35        |
| 7.4 CONCURRENT & NESTED MANAGEMENT              | 35        |
| 7.5 INTERRUPT REGISTER DESCRIPTION              | 36        |
| 7.6 EXTERNAL INTERRUPTS                         | 38        |
| 7.6.1 I/O Port Interrupt Sensitivity            | 38        |
| 7.7 EXTERNAL INTERRUPT CONTROL REGISTER (EICR)  | 40        |
| <b>8 POWER SAVING MODES</b>                     | <b>42</b> |
| 8.1 INTRODUCTION                                | 42        |
| 8.2 SLOW MODE                                   | 42        |
| 8.3 WAIT MODE                                   | 43        |

---

## Table of Contents

---

|           |   |           |
|-----------|---|-----------|
| 8.4       | ACTIVE-HALT AND HALT MODES                                      | 44        |
| 8.4.1     | ACTIVE-HALT MODE  | 44        |
| 8.4.2     | HALT MODE   | 45        |
| <b>9</b>  | <b>I/O PORTS</b>  | <b>47</b> |
| 9.1       | INTRODUCTION  | 47        |
| 9.2       | FUNCTIONAL DESCRIPTION  | 47        |
| 9.2.1     | Input Modes   | 47        |
| 9.2.2     | Output Modes  | 47        |
| 9.2.3     | Alternate Functions   | 47        |
| 9.3       | I/O PORT IMPLEMENTATION   | 50        |
| 9.4       | LOW POWER MODES   | 50        |
| 9.5       | INTERRUPTS  | 50        |
| 9.5.1     | I/O Port Implementation   | 51        |
| <b>10</b> | <b>ON-CHIP PERIPHERALS</b>                                      | <b>53</b> |
| 10.1      | WATCHDOG TIMER (WDG)  | 53        |
| 10.1.1    | Introduction  | 53        |
| 10.1.2    | Main Features   | 53        |
| 10.1.3    | Functional Description  | 53        |
| 10.1.4    | How to Program the Watchdog Timeout                             | 54        |
| 10.1.5    | Low Power Modes   | 56        |
| 10.1.6    | Hardware Watchdog Option  | 56        |
| 10.1.7    | Using Halt Mode with the WDG (WDGHALT option)                   | 56        |
| 10.1.8    | Interrupts  | 56        |
| 10.1.9    | Register Description  | 56        |
| 10.2      | MAIN CLOCK CONTROLLER WITH REAL TIME CLOCK AND BEEPER (MCC/RTC) | 58        |
| 10.2.1    | Programmable CPU Clock Prescaler                                | 58        |
| 10.2.2    | Clock-out Capability  | 58        |
| 10.2.3    | Real Time Clock Timer (RTC)                                     | 58        |
| 10.2.4    | Beeper  | 58        |
| 10.2.5    | Low Power Modes   | 59        |
| 10.2.6    | Interrupts  | 59        |
| 10.2.7    | Register Description  | 59        |
| 10.3      | PWM AUTO-RELOAD TIMER (ART)                                     | 61        |
| 10.3.1    | Introduction  | 61        |
| 10.3.2    | Functional Description  | 62        |
| 10.3.3    | Register Description  | 66        |
| 10.4      | 16-BIT TIMER  | 70        |
| 10.4.1    | Introduction  | 70        |
| 10.4.2    | Main Features   | 70        |
| 10.4.3    | Functional Description  | 70        |
| 10.4.4    | Low Power Modes   | 82        |
| 10.4.5    | Interrupts  | 82        |
| 10.4.6    | Summary of Timer modes  | 82        |
| 10.4.7    | Register Description  | 83        |
| 10.5      | SERIAL PERIPHERAL INTERFACE (SPI)                               | 89        |
| 10.5.1    | Introduction  | 89        |
| 10.5.2    | Main Features   | 89        |

---

## Table of Contents

---

|           |                                       |            |
|-----------|---------------------------------------|------------|
| 10.5.3    | General Description                   | 89         |
| 10.5.4    | Clock Phase and Clock Polarity        | 93         |
| 10.5.5    | Error Flags                           | 94         |
| 10.5.6    | Low Power Modes                       | 96         |
| 10.5.7    | Interrupts                            | 96         |
| 10.5.8    | Register Description                  | 97         |
| 10.6      | SERIAL COMMUNICATIONS INTERFACE (SCI) | 100        |
| 10.6.1    | Introduction                          | 100        |
| 10.6.2    | Main Features                         | 100        |
| 10.6.3    | General Description                   | 100        |
| 10.6.4    | Functional Description                | 102        |
| 10.6.5    | Low Power Modes                       | 107        |
| 10.6.6    | Interrupts                            | 107        |
| 10.6.7    | Register Description                  | 108        |
| 10.7      | I2C BUS INTERFACE (I2C)               | 114        |
| 10.7.1    | Introduction                          | 114        |
| 10.7.2    | Main Features                         | 114        |
| 10.7.3    | General Description                   | 114        |
| 10.7.4    | Functional Description                | 116        |
| 10.7.5    | Low Power Modes                       | 120        |
| 10.7.6    | Interrupts                            | 120        |
| 10.7.7    | Register Description                  | 121        |
| 10.8      | CONTROLLER AREA NETWORK (CAN)         | 127        |
| 10.8.1    | Introduction                          | 127        |
| 10.8.2    | Main Features                         | 128        |
| 10.8.3    | Functional Description                | 128        |
| 10.8.4    | Register Description                  | 134        |
| 10.9      | 10-BIT A/D CONVERTER (ADC)            | 144        |
| 10.9.1    | Introduction                          | 144        |
| 10.9.2    | Main Features                         | 144        |
| 10.9.3    | Functional Description                | 145        |
| 10.9.4    | Low Power Modes                       | 145        |
| 10.9.5    | Interrupts                            | 145        |
| 10.9.6    | Register Description                  | 146        |
| <b>11</b> | <b>INSTRUCTION SET</b>                | <b>148</b> |
| 11.1      | ST7 ADDRESSING MODES                  | 148        |
| 11.1.1    | Inherent                              | 149        |
| 11.1.2    | Immediate                             | 149        |
| 11.1.3    | Direct                                | 149        |
| 11.1.4    | Indexed (No Offset, Short, Long)      | 149        |
| 11.1.5    | Indirect (Short, Long)                | 149        |
| 11.1.6    | Indirect Indexed (Short, Long)        | 150        |
| 11.1.7    | Relative mode (Direct, Indirect)      | 150        |
| 11.2      | INSTRUCTION GROUPS                    | 151        |
| <b>12</b> | <b>ELECTRICAL CHARACTERISTICS</b>     | <b>154</b> |
| 12.1      | PARAMETER CONDITIONS                  | 154        |
| 12.1.1    | Minimum and Maximum values            | 154        |

---

## Table of Contents

---

|   |     |
|---|-----|
| 12.1.2 Typical values                                       | 154 |
| 12.1.3 Typical curves                                       | 154 |
| 12.1.4 Loading capacitor                                    | 154 |
| 12.1.5 Pin input voltage                                    | 154 |
| 12.2 ABSOLUTE MAXIMUM RATINGS                               | 155 |
| 12.2.1 Voltage Characteristics                              | 155 |
| 12.2.2 Current Characteristics                              | 155 |
| 12.2.3 Thermal Characteristics                              | 156 |
| 12.3 OPERATING CONDITIONS                                   | 156 |
| 12.3.1 General Operating Conditions                         | 156 |
| 12.3.2 Operating Conditions with Low Voltage Detector (LVD) | 157 |
| 12.3.3 Auxiliary Voltage Detector (AVD) Thresholds          | 158 |
| 12.3.4 External Voltage Detector (EVD) Thresholds           | 158 |
| 12.4 SUPPLY CURRENT CHARACTERISTICS                         | 159 |
| 12.4.1 RUN and SLOW Modes (Flash devices)                   | 159 |
| 12.4.2 WAIT and SLOW WAIT Modes (Flash devices)             | 160 |
| 12.4.3 RUN and SLOW Modes (ROM devices)                     | 161 |
| 12.4.4 WAIT and SLOW WAIT Modes (ROM devices)               | 161 |
| 12.4.5 HALT and ACTIVE-HALT Modes                           | 162 |
| 12.4.6 Supply and Clock Managers                            | 162 |
| 12.4.7 On-Chip Peripherals                                  | 163 |
| 12.5 CLOCK AND TIMING CHARACTERISTICS                       | 164 |
| 12.5.1 General Timings                                      | 164 |
| 12.5.2 External Clock Source                                | 164 |
| 12.5.3 Crystal and Ceramic Resonator Oscillators            | 165 |
| 12.5.4 RC Oscillators                                       | 167 |
| 12.5.5 Clock Security System (CSS)                          | 168 |
| 12.5.6 PLL Characteristics                                  | 168 |
| 12.6 MEMORY CHARACTERISTICS                                 | 169 |
| 12.6.1 RAM and Hardware Registers                           | 169 |
| 12.6.2 FLASH Memory   | 169 |
| 12.7 EMC CHARACTERISTICS                                    | 170 |
| 12.7.1 Functional EMS                                       | 170 |
| 12.7.2 Electro Magnetic Interference (EMI)                  | 170 |
| 12.7.3 Absolute Electrical Sensitivity                      | 171 |
| 12.7.4 ESD Pin Protection Strategy                          | 173 |
| 12.8 I/O PORT PIN CHARACTERISTICS                           | 175 |
| 12.8.1 General Characteristics                              | 175 |
| 12.8.2 Output Driving Current                               | 176 |
| 12.9 CONTROL PIN CHARACTERISTICS                            | 178 |
| 12.9.1 Asynchronous RESET Pin                               | 178 |
| 12.9.2 ICCSEL/VPP Pin                                       | 179 |
| 12.10 TIMER PERIPHERAL CHARACTERISTICS                      | 179 |
| 12.10.18-Bit PWM-ART Auto-Reload Timer                      | 179 |
| 12.10.216-Bit Timer   | 179 |
| 12.11 COMMUNICATION INTERFACE CHARACTERISTICS               | 180 |
| 12.11.1SPI - Serial Peripheral Interface                    | 180 |

---

## Table of Contents

---

|   |            |
|---|------------|
| 12.11.2CAN - Controller Area Network Interface .....                        | 182        |
| 12.11.3I2C - Inter IC Control Interface .....                               | 183        |
| 12.12 10-BIT ADC CHARACTERISTICS .....                                      | 184        |
| 12.12.1ADC Accuracy .....   | 186        |
| <b>13 PACKAGE CHARACTERISTICS .....</b>                                     | <b>187</b> |
| 13.1 PACKAGE MECHANICAL DATA .....  | 187        |
| 13.2 THERMAL CHARACTERISTICS .....  | 189        |
| 13.3 SOLDERING AND GLUEABILITY INFORMATION .....                            | 190        |
| <b>14 ST72521M/(A)R DEVICE CONFIGURATION AND ORDERING INFORMATION .....</b> | <b>191</b> |
| 14.1 FLASH OPTION BYTES .....   | 191        |
| 14.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE .....        | 193        |
| 14.3 DEVELOPMENT TOOLS .....  | 195        |
| 14.3.1 Socket and Emulator Adapter Information .....                        | 195        |
| 14.4 ST7 APPLICATION NOTES .....  | 196        |
| <b>15 SUMMARY OF CHANGES .....</b>  | <b>198</b> |

### 1 INTRODUCTION

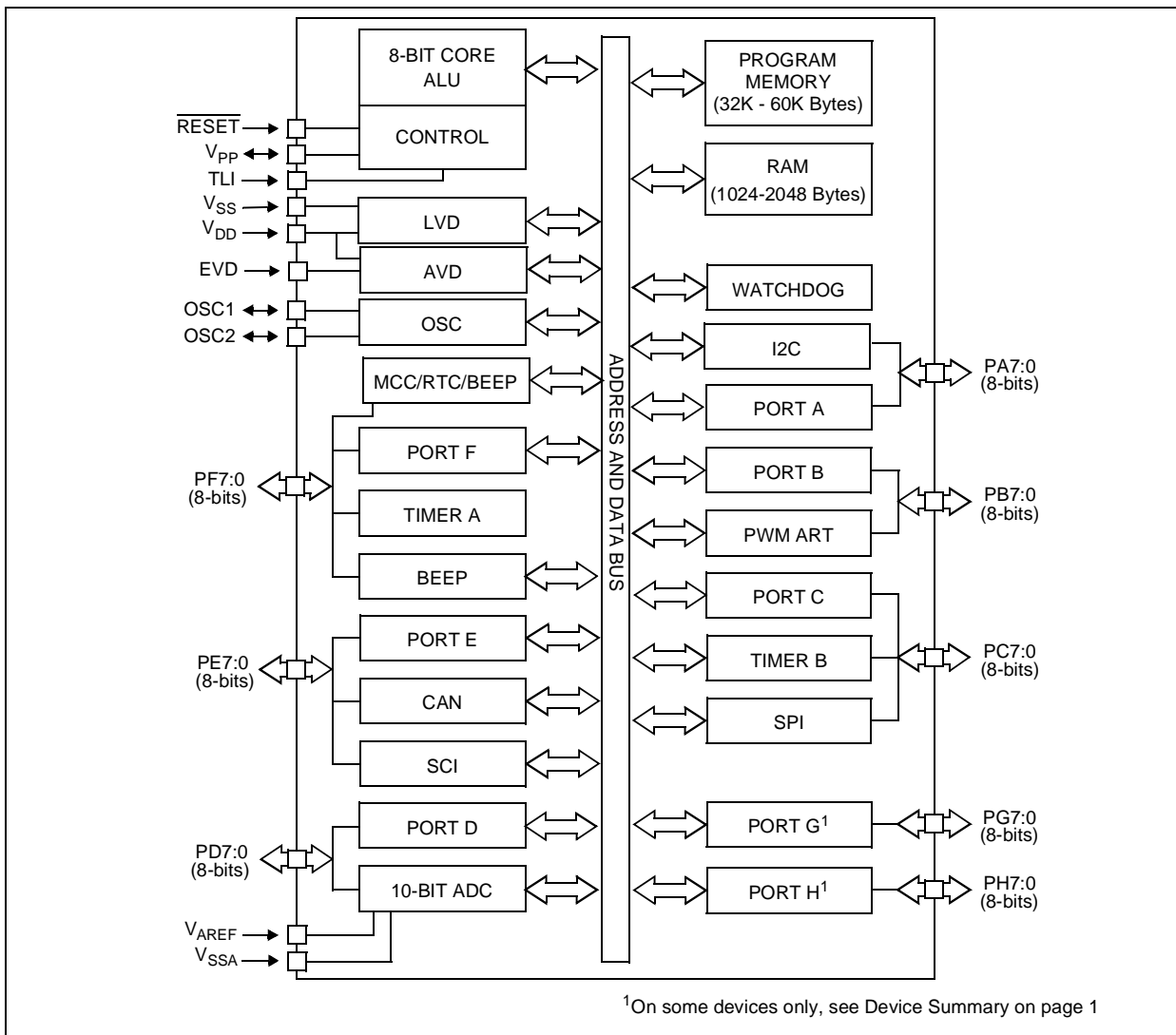
The ST72521(A)R and ST72521M devices are members of the ST7 microcontroller family designed for mid-range applications with a CAN bus interface (Controller Area Network).

All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set and are available with FLASH or ROM program 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.

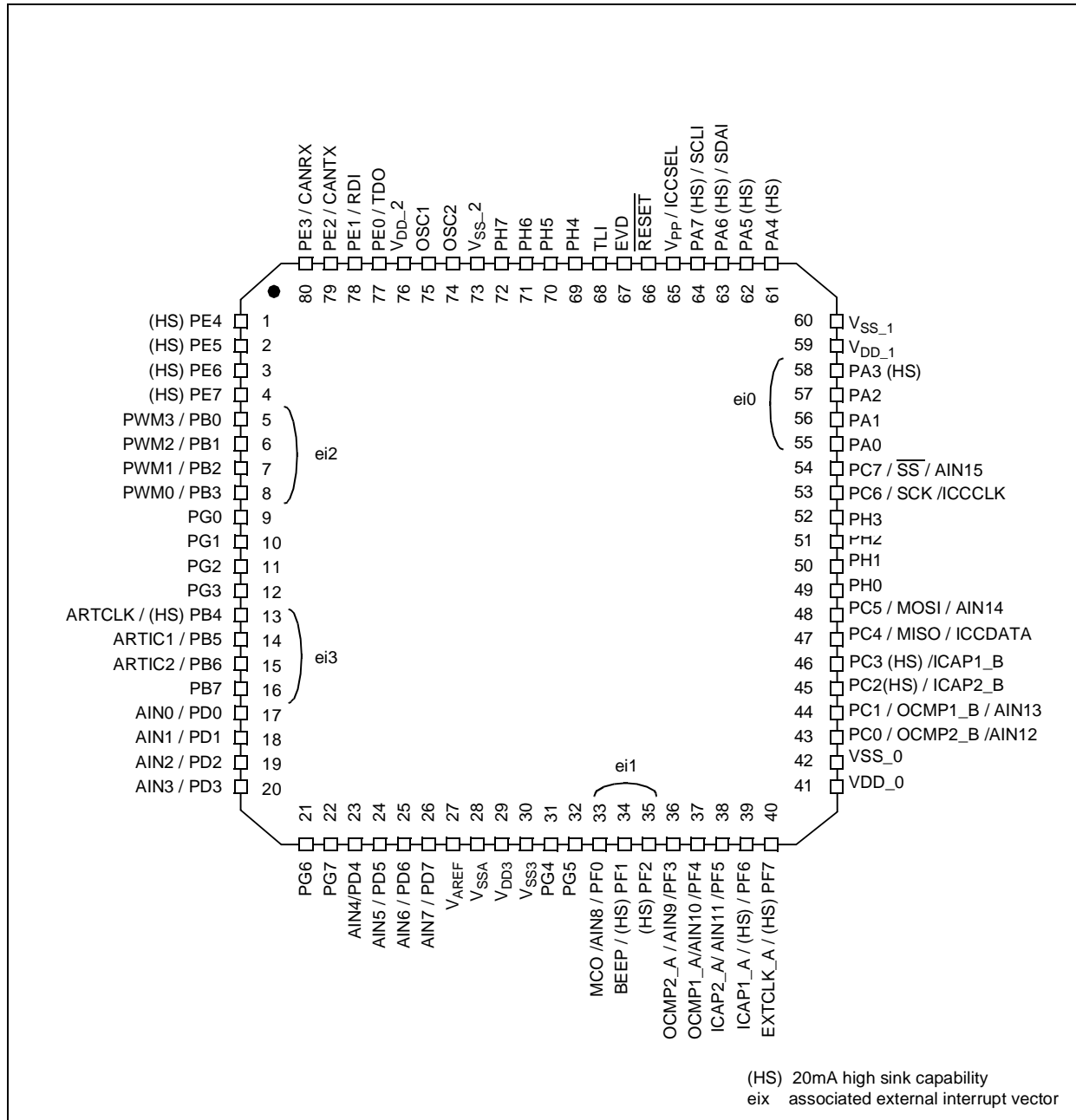
The enhanced instruction set and addressing modes of the ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned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

Figure 1. Device Block Diagram



## 2 PIN DESCRIPTION

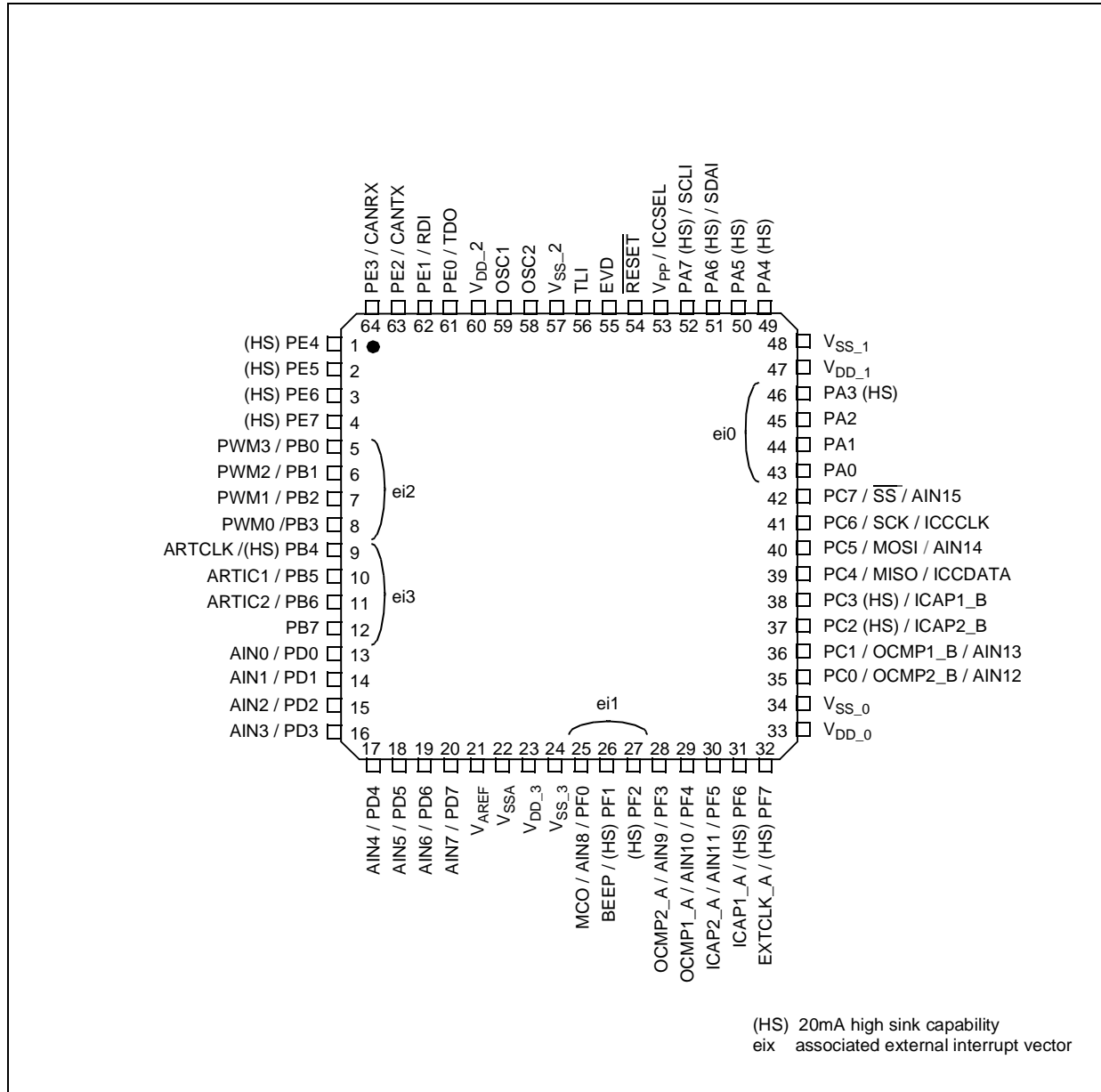
Figure 2. 80-Pin TQFP 14x14 Package Pinout





PIN DESCRIPTION (Cont'd)

Figure 3. 64-Pin TQFP 14x14 and 10x10 Package Pinout



## ST72521M/R/AR

### PIN DESCRIPTION (Cont'd)

For external pin connection guidelines, refer to See “ELECTRICAL CHARACTERISTICS” on page 154.

#### Legend / Abbreviations for Table 1:

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 0.8V / 2V with Schmitt trigger

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

Port and control configuration:

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

Refer to “I/O PORTS” on page 47 for more details on the software configuration of the I/O ports.

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

**Table 1. Device Pin Description**

| Pin n° |        | Pin Name        | Type | Level |        | Port  |     |     |     |        |    | Main function (after reset) | Alternate function      |
|--------|--------|-----------------|------|-------|--------|-------|-----|-----|-----|--------|----|-----------------------------|-------------------------|
| TQFP80 | TQFP64 |                 |      | Input | Output | Input |     |     |     | Output |    |                             |                         |
|        |        |                 |      |       |        | float | wpu | int | ana | OD     | PP |                             |                         |
| 1      | 1      | PE4 (HS)        | I/O  | $C_T$ | HS     | X     | X   |     |     | X      | X  | Port E4                     |                         |
| 2      | 2      | PE5 (HS)        | I/O  | $C_T$ | HS     | X     | X   |     |     | X      | X  | Port E5                     |                         |
| 3      | 3      | PE6 (HS)        | I/O  | $C_T$ | HS     | X     | X   |     |     | X      | X  | Port E6                     |                         |
| 4      | 4      | PE7 (HS)        | I/O  | $C_T$ | HS     | X     | X   |     |     | X      | X  | Port E7                     |                         |
| 5      | 5      | PB0/PWM3        | I/O  | $C_T$ |        | X     |     | ei2 |     | X      | X  | Port B0                     | PWM Output 3            |
| 6      | 6      | PB1/PWM2        | I/O  | $C_T$ |        | X     |     | ei2 |     | X      | X  | Port B1                     | PWM Output 2            |
| 7      | 7      | PB2/PWM1        | I/O  | $C_T$ |        | X     |     | ei2 |     | X      | X  | Port B2                     | PWM Output 1            |
| 8      | 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     | 9      | PB4 (HS)/ARTCLK | I/O  | $C_T$ | HS     | X     |     | ei3 |     | X      | X  | Port B4                     | PWM-ART External Clock  |
| 14     | 10     | PB5/ARTIC1      | I/O  | $C_T$ |        | X     |     | ei3 |     | X      | X  | Port B5                     | PWM-ART Input Capture 1 |
| 15     | 11     | PB6/ARTIC2      | I/O  | $C_T$ |        | X     |     | ei3 |     | X      | X  | Port B6                     | PWM-ART Input Capture 2 |
| 16     | 12     | PB7             | I/O  | $C_T$ |        | X     |     | ei3 |     | X      | X  | Port B7                     |                         |
| 17     | 13     | PD0 /AIN0       | I/O  | $C_T$ |        | X     | X   |     | X   | X      | X  | Port D0                     | ADC Analog Input 0      |
| 18     | 14     | PD1/AIN1        | I/O  | $C_T$ |        | X     | X   |     | X   | X      | X  | Port D1                     | ADC Analog Input 1      |
| 19     | 15     | PD2/AIN2        | I/O  | $C_T$ |        | X     | X   |     | X   | X      | X  | Port D2                     | ADC Analog Input 2      |
| 20     | 16     | 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     | 17     | PD4/AIN4        | I/O  | $C_T$ |        | X     | X   |     | X   | X      | X  | Port D4                     | ADC Analog Input 4      |

| Pin n° | TQFP80 | TQFP64 | Pin Name          | Type | Level          |        | Port  |     |     |     |        |    | Main function (after reset)      | Alternate function   |  |
|--------|--------|--------|-------------------|------|----------------|--------|-------|-----|-----|-----|--------|----|----------------------------------|--|--|
|        |        |        |                   |      | Input          | Output | Input |     |     |     | Output |    |                                  |  |  |
|        |        |        |                   |      |                |        | float | wpu | int | ana | OD     | PP |                                  |  |  |
| 24     | 18     |        | PD5/AIN5          | I/O  | C <sub>T</sub> |        | X     | X   |     | X   | X      | X  | Port D5                          | ADC Analog Input 5   |  |
| 25     | 19     |        | PD6/AIN6          | I/O  | C <sub>T</sub> |        | X     | X   |     | X   | X      | X  | Port D6                          | ADC Analog Input 6   |  |
| 26     | 20     |        | PD7/AIN7          | I/O  | C <sub>T</sub> |        | X     | X   |     | X   | X      | X  | Port D7                          | ADC Analog Input 7   |  |
| 27     | 21     |        | V <sub>AREF</sub> | I    |                |        |       |     |     |     |        |    | Analog Reference Voltage for ADC |  |  |
| 28     | 22     |        | V <sub>SSA</sub>  | S    |                |        |       |     |     |     |        |    | Analog Ground Voltage            |  |  |
| 29     | 23     |        | V <sub>DD_3</sub> | S    |                |        |       |     |     |     |        |    | Digital Main Supply Voltage      |  |  |
| 30     | 24     |        | V <sub>SS_3</sub> | S    |                |        |       |     |     |     |        |    | Digital Ground Voltage           |  |  |
| 31     | -      |        | PG4               | I/O  | T <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port G4                          |  |  |
| 32     | -      |        | PG5               | I/O  | T <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port G5                          |  |  |
| 33     | 25     |        | PF0/MCO/AIN8      | I/O  | C <sub>T</sub> |        | X     |     | ei1 |     | X      | X  | Port F0                          | Main clock out (f <sub>OSC</sub> /2)<br>ADC Analog Input 8 |  |
| 34     | 26     |        | PF1 (HS)/BEEP     | I/O  | C <sub>T</sub> | HS     | X     |     | ei1 |     | X      | X  | Port F1                          | Beep signal output   |  |
| 35     | 27     |        | PF2 (HS)          | I/O  | C <sub>T</sub> | HS     | X     |     | ei1 |     | X      | X  | Port F2                          |  |  |
| 36     | 28     |        | PF3/OCMP2_A/AIN9  | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port F3                          | Timer A Output Compare 2<br>ADC Analog Input 9             |  |
| 37     | 29     |        | PF4/OCMP1_A/AIN10 | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port F4                          | Timer A Output Compare 1<br>ADC Analog Input 10            |  |
| 38     | 30     |        | PF5/ICAP2_A/AIN11 | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port F5                          | Timer A Input Capture 2<br>ADC Analog Input 11             |  |
| 39     | 31     |        | PF6 (HS)/ICAP1_A  | I/O  | C <sub>T</sub> | HS     | X     | X   |     |     | X      | X  | Port F6                          | Timer A Input Capture 1                                    |  |
| 40     | 32     |        | PF7 (HS)/EXTCLK_A | I/O  | C <sub>T</sub> | HS     | X     | X   |     |     | X      | X  | Port F7                          | Timer A External Clock Source                              |  |
| 41     | 33     |        | V <sub>DD_0</sub> | S    |                |        |       |     |     |     |        |    | Digital Main Supply Voltage      |  |  |
| 42     | 34     |        | V <sub>SS_0</sub> | S    |                |        |       |     |     |     |        |    | Digital Ground Voltage           |  |  |
| 43     | 35     |        | PC0/OCMP2_B/AIN12 | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port C0                          | Timer B Output Compare 2<br>ADC Analog Input 12            |  |
| 44     | 36     |        | PC1/OCMP1_B/AIN13 | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port C1                          | Timer B Output Compare 1<br>ADC Analog Input 13            |  |
| 45     | 37     |        | PC2 (HS)/ICAP2_B  | I/O  | C <sub>T</sub> | HS     | X     | X   |     |     | X      | X  | Port C2                          | Timer B Input Capture 2                                    |  |
| 46     | 38     |        | PC3 (HS)/ICAP1_B  | I/O  | C <sub>T</sub> | HS     | X     | X   |     |     | X      | X  | Port C3                          | Timer B Input Capture 1                                    |  |
| 47     | 39     |        | PC4/MISO/ICCDATA  | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port C4                          | SPI Master In / Slave Out Data<br>ICC Data Input           |  |
| 48     | 40     |        | PC5/MOSI/AIN14    | I/O  | C <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port C5                          | SPI Master Out / Slave In Data<br>ADC Analog Input 14      |  |
| 49     | -      |        | PH0               | I/O  | T <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port H0                          |  |  |
| 50     | -      |        | PH1               | I/O  | T <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port H1                          |  |  |
| 51     | -      |        | PH2               | I/O  | T <sub>T</sub> |        | X     | X   |     |     | X      | X  | Port H2                          |  |  |

## ST72521M/R/AR

| Pin n° |        | Pin Name                    | Type | Level          |        | Port  |     |     |     |        |    | Main function (after reset) | Alternate function   |                                      |                     |
|--------|--------|-----------------------------|------|----------------|--------|-------|-----|-----|-----|--------|----|-----------------------------|--|--------------------------------------|---------------------|
| TQFP80 | TQFP64 |                             |      | Input          | Output | Input |     |     |     | Output |    |                             |  |                                      |                     |
|        |        |                             |      |                |        | float | wpu | int | ana | OD     | PP |                             |  |                                      |                     |
| 52     | -      | PH3                         | I/O  | T <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port H3  |                                      |                     |
| 53     | 41     | PC6/SCK/ICCCCLK             | I/O  | C <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port C6  | SPI Serial Clock                     | ICC Clock Output    |
| 54     | 42     | PC7/ $\overline{SS}$ /AIN15 | I/O  | C <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port C7  | SPI Slave Select (active low)        | ADC Analog Input 15 |
| 55     | 43     | PA0                         | I/O  | C <sub>T</sub> |        | X     | ei0 |     |     |        | X  | X                           | Port A0  |                                      |                     |
| 56     | 44     | PA1                         | I/O  | C <sub>T</sub> |        | X     | ei0 |     |     |        | X  | X                           | Port A1  |                                      |                     |
| 57     | 45     | PA2                         | I/O  | C <sub>T</sub> |        | X     | ei0 |     |     |        | X  | X                           | Port A2  |                                      |                     |
| 58     | 46     | PA3 (HS)                    | I/O  | C <sub>T</sub> | HS     | X     |     | ei0 |     |        | X  | X                           | Port A3  |                                      |                     |
| 59     | 47     | V <sub>DD_1</sub>           | S    |                |        |       |     |     |     |        |    |                             | Digital Main Supply Voltage  |                                      |                     |
| 60     | 48     | V <sub>SS_1</sub>           | S    |                |        |       |     |     |     |        |    |                             | Digital Ground Voltage   |                                      |                     |
| 61     | 49     | PA4 (HS)                    | I/O  | C <sub>T</sub> | HS     | X     | X   |     |     |        | X  | X                           | Port A4  |                                      |                     |
| 62     | 50     | PA5 (HS)                    | I/O  | C <sub>T</sub> | HS     | X     | X   |     |     |        | X  | X                           | Port A5  |                                      |                     |
| 63     | 51     | PA6 (HS)/SDAI               | I/O  | C <sub>T</sub> | HS     | X     |     |     |     |        | T  |                             | Port A6  | I <sup>2</sup> C Data <sup>1)</sup>  |                     |
| 64     | 52     | PA7 (HS)/SCLI               | I/O  | C <sub>T</sub> | HS     | X     |     |     |     |        | T  |                             | Port A7  | I <sup>2</sup> C Clock <sup>1)</sup> |                     |
| 65     | 53     | V <sub>pp</sub> / ICCSEL    | I    |                |        |       |     |     |     |        |    |                             | Must be tied low. In flash programming mode, this pin acts as the programming voltage input V <sub>pp</sub> . See Section 12.9.2 for more details. High voltage must not be applied to ROM devices |                                      |                     |
| 66     | 54     | $\overline{RESET}$          | I/O  | C <sub>T</sub> |        |       |     |     |     |        |    |                             | Top priority non maskable interrupt.   |                                      |                     |
| 67     | 55     | EVD                         |      |                |        |       |     |     |     |        |    |                             | External voltage detector  |                                      |                     |
| 68     | 56     | TLI                         | I    | C <sub>T</sub> |        | X     |     | X   |     |        |    |                             | Top level interrupt input pin  |                                      |                     |
| 69     | -      | PH4                         | I/O  | T <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port H4  |                                      |                     |
| 70     | -      | PH5                         | I/O  | T <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port H5  |                                      |                     |
| 71     | -      | PH6                         | I/O  | T <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port H6  |                                      |                     |
| 72     | -      | PH7                         | I/O  | T <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port H7  |                                      |                     |
| 73     | 57     | V <sub>SS_2</sub>           | S    |                |        |       |     |     |     |        |    |                             | Digital Ground Voltage   |                                      |                     |
| 74     | 58     | OSC2 <sup>3)</sup>          | I/O  |                |        |       |     |     |     |        |    |                             | Resonator oscillator inverter output or capacitor input for RC oscillator  |                                      |                     |
| 75     | 59     | OSC1 <sup>3)</sup>          | I    |                |        |       |     |     |     |        |    |                             | External clock input or Resonator oscillator inverter input or resistor input for RC oscillator  |                                      |                     |
| 76     | 60     | V <sub>DD_2</sub>           | S    |                |        |       |     |     |     |        |    |                             | Digital Main Supply Voltage  |                                      |                     |
| 77     | 61     | PE0/TDO                     | I/O  | C <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port E0  | SCI Transmit Data Out                |                     |
| 78     | 62     | PE1/RDI                     | I/O  | C <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port E1  | SCI Receive Data In                  |                     |
| 79     | 63     | PE2/CANTX                   | I/O  | C <sub>T</sub> |        |       | X   |     |     |        |    |                             | Port E2  | CAN Transmit Data Output             |                     |
| 80     | 64     | PE3/CANRX                   | I/O  | C <sub>T</sub> |        | X     | X   |     |     |        | X  | X                           | Port E3  | CAN Receive Data Input               |                     |

### 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. 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). See "I/O PORTS" on page 47. and Section 12.8 I/O PORT PIN CHARACTERISTICS for more details.

3. OSC1 and OSC2 pins connect a crystal/ceramic resonator, an RC oscillator, or an external source to the on-chip oscillator; see Section 1 INTRODUCTION and Section 12.5 CLOCK AND TIMING CHARACTERISTICS for more details.

4. 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.

### 3 REGISTER & MEMORY MAP

As shown in Figure 4, the MCU is capable of addressing 64K bytes of memories and I/O registers. The available memory locations consist of 128 bytes of register locations, up to 2Kbytes of RAM and up to 60Kbytes of user program memory. The

RAM space includes up to 256 bytes for the stack from 0100h to 01FFh.

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

**Figure 4. Memory Map**

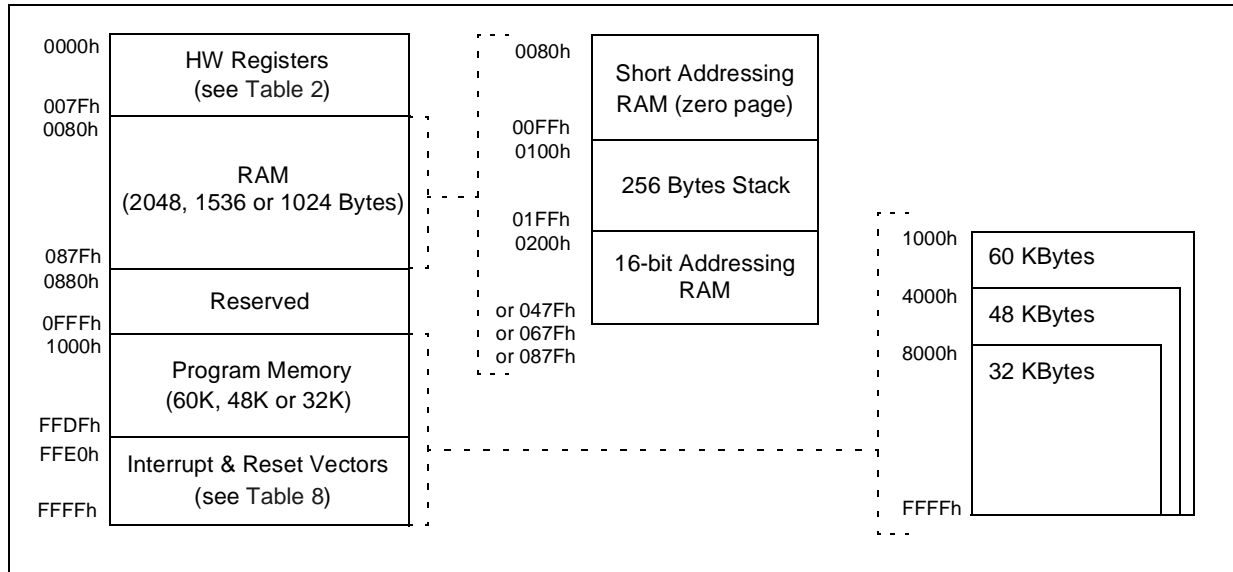


Table 2. Hardware Register Map

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

**ST72521M/R/AR**

| Address   | Block    | Register Label   | Register Name   | Reset Status  | Remarks  |  |
|---|----------|--|---|---|--|--|
| 0024h<br>0025h<br>0026h<br>0027h  | ITC      | ISPR0<br>ISPR1<br>ISPR2<br>ISPR3   | Interrupt Software Priority Register 0<br>Interrupt Software Priority Register 1<br>Interrupt Software Priority Register 2<br>Interrupt Software Priority Register 3  | FFh<br>FFh<br>FFh<br>FFh  | R/W<br>R/W<br>R/W<br>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<br>002Dh  | MCC      | MCCSR<br>MCCBCR  | Main Clock Control / Status Register<br>Main Clock Controller: Beep Control Register  | 00h<br>00h  | R/W<br>R/W   |  |
| 002Eh<br>to<br>0030h  |          | Reserved Area (3 Bytes)  |   |   |  |  |
| 0031h<br>0032h<br>0033h<br>0034h<br>0035h<br>0036h<br>0037h<br>0038h<br>0039h<br>003Ah<br>003Bh<br>003Ch<br>003Dh<br>003Eh<br>003Fh | TIMER A  | TACR2<br>TACR1<br>TACSR<br>TAIC1HR<br>TAIC1LR<br>TAOC1HR<br>TAOC1LR<br>TACHR<br>TACL<br>TAACHR<br>TAACLR<br>TAIC2HR<br>TAIC2LR<br>TAOC2HR<br>TAOC2LR | Timer A Control Register 2<br>Timer A Control Register 1<br>Timer A Control/Status Register<br>Timer A Input Capture 1 High Register<br>Timer A Input Capture 1 Low Register<br>Timer A Output Compare 1 High Register<br>Timer A Output Compare 1 Low Register<br>Timer A Counter High Register<br>Timer A Counter Low Register<br>Timer A Alternate Counter High Register<br>Timer A Alternate Counter Low Register<br>Timer A Input Capture 2 High Register<br>Timer A Input Capture 2 Low Register<br>Timer A Output Compare 2 High Register<br>Timer A Output Compare 2 Low Register | 00h<br>00h<br>xxh<br>xxh<br>xxh<br>80h<br>00h<br>FFh<br>FCh<br>FFh<br>FCh<br>xxh<br>xxh<br>80h<br>00h   | R/W<br>R/W<br>R/W<br>Read Only<br>Read Only<br>R/W<br>R/W<br>Read Only<br>Read Only<br>Read Only<br>Read Only<br>Read Only<br>R/W<br>R/W |  |
| 0040h   |          | Reserved Area (1 Byte)   |   |   |  |  |
| 0041h<br>0042h<br>0043h<br>0044h<br>0045h<br>0046h<br>0047h<br>0048h<br>0049h<br>004Ah<br>004Bh<br>004Ch<br>004Dh<br>004Eh<br>004Fh |          | TIMER B  | TBCR2<br>TBCR1<br>TBCSR<br>TBIC1HR<br>TBIC1LR<br>TBOC1HR<br>TBOC1LR<br>TBCHR<br>TBCLR<br>TBACHR<br>TBACLR<br>TBIC2HR<br>TBIC2LR<br>TBOC2HR<br>TBOC2LR   | Timer B Control Register 2<br>Timer B Control Register 1<br>Timer B Control/Status Register<br>Timer B Input Capture 1 High Register<br>Timer B Input Capture 1 Low Register<br>Timer B Output Compare 1 High Register<br>Timer B Output Compare 1 Low Register<br>Timer B Counter High Register<br>Timer B Counter Low Register<br>Timer B Alternate Counter High Register<br>Timer B Alternate Counter Low Register<br>Timer B Input Capture 2 High Register<br>Timer B Input Capture 2 Low Register<br>Timer B Output Compare 2 High Register<br>Timer B Output Compare 2 Low Register | 00h<br>00h<br>xxh<br>xxh<br>xxh<br>80h<br>00h<br>FFh<br>FCh<br>FFh<br>FCh<br>xxh<br>xxh<br>80h<br>00h                                    | R/W<br>R/W<br>R/W<br>Read Only<br>Read Only<br>R/W<br>R/W<br>Read Only<br>Read Only<br>Read Only<br>Read Only<br>Read Only<br>R/W<br>R/W |



| Address   | Block                   | Register Label  | Register Name   | Reset Status  | Remarks  |
|---|-------------------------|---|---|---|--|
| 0050h<br>0051h<br>0052h<br>0053h<br>0054h<br>0055h<br>0056h<br>0057h                            | SCI                     | SCISR<br>SCIDR<br>SCIBRR<br>SCICR1<br>SCICR2<br>SCIERPR<br>SCIETPR  | SCI Status Register<br>SCI Data Register<br>SCI Baud Rate Register<br>SCI Control Register 1<br>SCI Control Register 2<br>SCI Extended Receive Prescaler Register<br>Reserved area<br>SCI Extended Transmit Prescaler Register  | C0h<br>xxh<br>00xx xxxxb<br>xxh<br>00h<br>00h<br>---                      | Read Only<br>R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>R/W                            |
| 0058h<br>0059h  | Reserved Area (2 Bytes) |   |   |   |  |
| 005Ah<br>005Bh<br>005Ch<br>005Dh<br>005Eh<br>005Fh<br>0060h<br>to<br>006Fh                      | CAN                     | CANISR<br>CANICR<br>CANCSR<br>CANBRPR<br>CANBTR<br>CANPSR   | CAN Interrupt Status Register<br>CAN Interrupt Control Register<br>CAN Control / Status Register<br>CAN Baud Rate Prescaler Register<br>CAN Bit Timing Register<br>CAN Page Selection Register<br>First address<br>to<br>Last address of CAN page x   | 00h<br>00h<br>00h<br>00h<br>23h<br>00h<br>--                              | R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>See CAN<br>Description               |
| 0070h<br>0071h<br>0072h   | ADC                     | ADCCSR<br>ADCDRH<br>ADCURL  | Control/Status Register<br>Data High Register<br>Data Low Register  | 00h<br>xxh<br>0000 00xxb  | R/W<br>Read Only<br>Read Only  |
| 0073h<br>0074h<br>0075h<br>0076h<br>0077h<br>0078h<br>0079h<br>007Ah<br>007Bh<br>007Ch<br>007Dh | PWM ART                 | PWMDCR3<br>PWMDCR2<br>PWMDCR1<br>PWMDCR0<br>PWMCR<br>ARTCSR<br>ARTCAR<br>ARTARR<br>ARTICCSR<br>ARTICR1<br>ARTICR2 | PWM AR Timer Duty Cycle Register 3<br>PWM AR Timer Duty Cycle Register 2<br>PWM AR Timer Duty Cycle Register 1<br>PWM AR Timer Duty Cycle Register 0<br>PWM AR Timer Control Register<br>Auto-Reload Timer Control/Status Register<br>Auto-Reload Timer Counter Access Register<br>Auto-Reload Timer Auto-Reload Register<br>AR Timer Input Capture Control/Status Reg.<br>AR Timer Input Capture Register 1<br>AR Timer Input Capture Register 1 | 00h<br>00h<br>00h<br>00h<br>00h<br>00h<br>00h<br>00h<br>00h<br>00h<br>00h | R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>R/W<br>Read Only<br>Read Only |
| 007Eh<br>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<sub>PP</sub> 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.

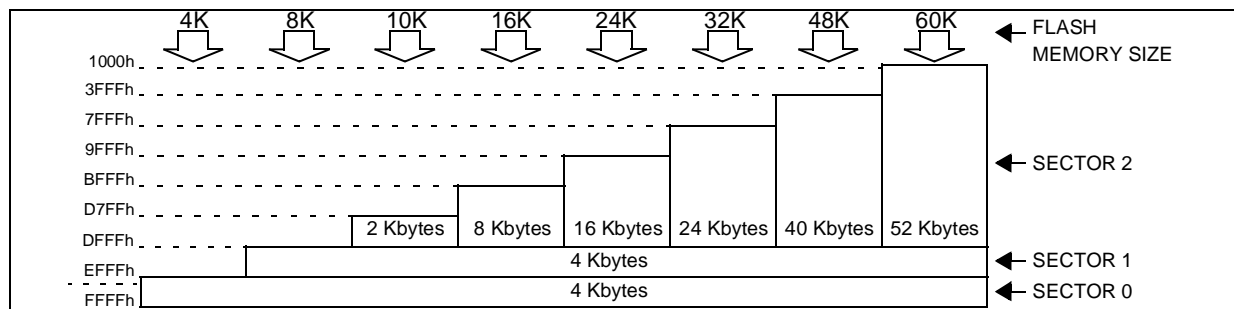
### 4.2 Main Features

- 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 against piracy
- 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.

Figure 5. Memory Map and Sector Address



Depending on the overall Flash memory size in the microcontroller device, there are up to three user sectors (see Table 3). 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 5). 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 3. Sectors available in Flash devices

| Flash Size (bytes) | Available Sectors |
|--------------------|-------------------|
| 4K                 | Sector 0          |
| 8K                 | Sectors 0,1       |
| > 8K               | Sectors 0,1, 2    |

### 4.3.1 Read-out Protection

Read-out protection, when selected, makes it impossible to extract the memory content from the microcontroller, thus preventing piracy. Even ST cannot access the user code.

In flash devices, this protection is removed by reprogramming the option. In this case, the entire program memory is first automatically erased.

Read-out protection selection depends on the device type:

- In Flash devices it is enabled and removed through the FMP\_R bit in the option byte.
- In ROM devices it is enabled by mask option specified in the Option List.

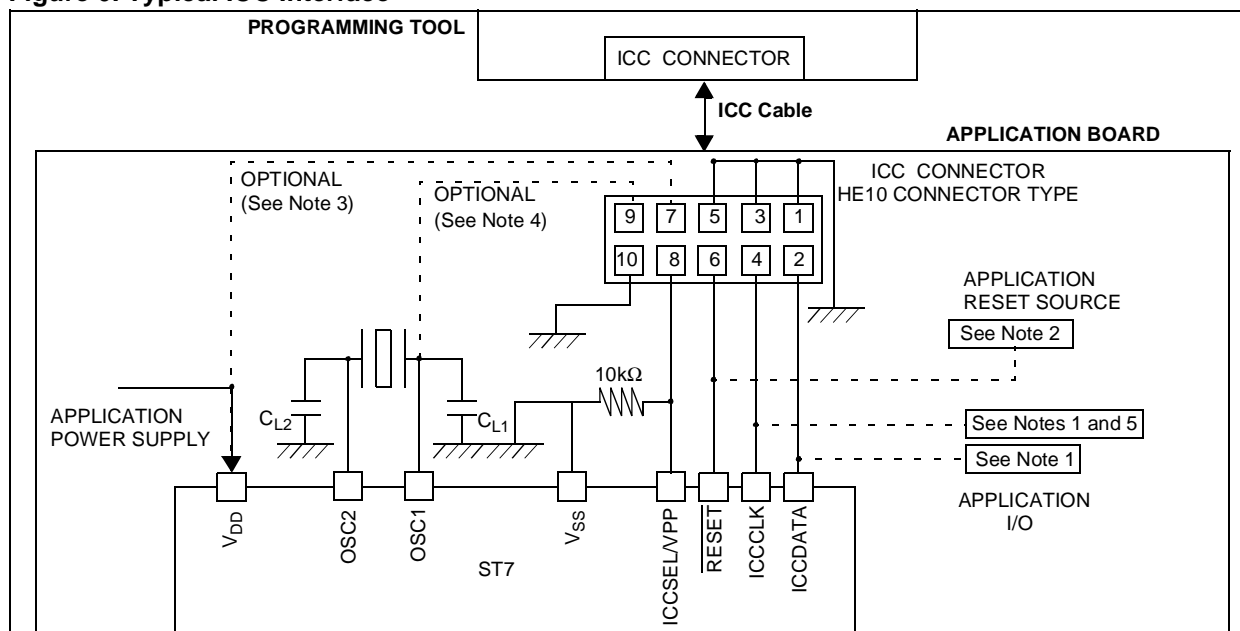
## FLASH PROGRAM MEMORY (Cont'd)

## 4.4 ICC Interface

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

- RESET: device reset
- V<sub>SS</sub>: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input/output serial data pin
- ICCSEL/V<sub>PP</sub>: programming voltage
- OSC1(or OSCIN): main clock input for external source (optional)
- V<sub>DD</sub>: application board power supply (optional, see Figure 6, Note 3)

Figure 6. Typical ICC Interface



## Notes:

1. If the ICCCLK or ICCDATA pins are only used as outputs in the application, no signal isolation is necessary. As soon as the Programming Tool is plugged to the board, even if an ICC session is not in progress, the ICCCLK and ICCDATA pins are not available for the application. If they are used as inputs by the application, isolation such as a serial resistor has to be implemented 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 < 1K). A Schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with  $R > 1K$  or a reset management IC with open drain output and pull-up resistor  $> 1K$ , no additional components are needed. In all cases the user must ensure that no external

reset is generated by the application during the ICC session.

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

4. Pin 9 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.

5. In the application, when the RESET pin is low, the ICCCLK pin must always be in pull-up or high impedance state. For instance, it must never be forced to ground or connected to an external pull-down. This is to avoid entering ICC mode unexpectedly during normal application operation.

**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 6). 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.

**4.6.1 Register Description**

**FLASH CONTROL/STATUS REGISTER (FCSR)**

Read/Write

Reset Value: 0000 0000 (00h)

|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 7 |   |   |   |   |   |   |   | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

This register is reserved for use by Programming Tool software. It controls the Flash programming and erasing operations. For details on customizing Flash programming methods and In-Circuit Testing, refer to the ST7 Flash Programming Reference Manual.

**Table 4. Flash Control/Status Register Address and Reset Value**

| Address (Hex.) | Register Label      | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---------------------|---|---|---|---|---|---|---|---|
| 0029h          | FCSR<br>Reset Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



**CENTRAL PROCESSING UNIT (Cont'd)**

**Condition Code Register (CC)**

Read/Write

Reset Value: 111x1xxx

|   |   |    |   |    |   |   |   |
|---|---|----|---|----|---|---|---|
| 7 |   |    |   |    |   |   | 0 |
| 1 | 1 | I1 | H | I0 | N | Z | C |

The 8-bit Condition Code register contains the interrupt masks and four flags representative of the result of the instruction just executed. This register can also be handled by the PUSH and POP instructions.

These bits can be individually tested and/or controlled by specific instructions.

**Arithmetic Management Bits**

Bit 4 = **H** *Half carry*.

This bit is set by hardware when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instructions. It is reset by hardware during the same instructions.

- 0: No half carry has occurred.
- 1: A half carry has occurred.

This bit is tested using the JRH or JRNH instruction. The H bit is useful in BCD arithmetic subroutines.

Bit 2 = **N** *Negative*.

This bit is set and cleared by hardware. It is representative of the result sign of the last arithmetic, logical or data manipulation. It's a copy of the result 7<sup>th</sup> bit.

- 0: The result of the last operation is positive or null.
- 1: The result of the last operation is negative (i.e. the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

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

This bit is set and cleared by hardware. This bit indicates that the result of the last arithmetic, logical or data manipulation is zero.

- 0: The result of the last operation is different from zero.
- 1: The result of the last operation is zero.

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

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

This bit is set and cleared by hardware and software. It indicates an overflow or an underflow has occurred during the last arithmetic operation.

- 0: No overflow or underflow has occurred.
- 1: An overflow or underflow has occurred.

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

**Interrupt Management Bits**

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

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

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

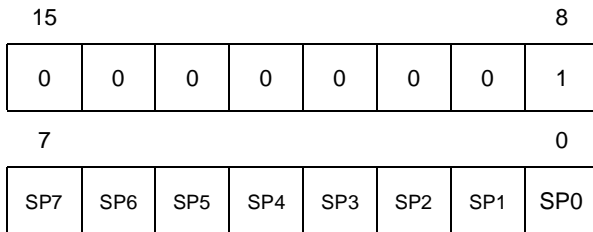
These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (IxSPR). They can be also set/cleared by software with the RIM, SIM, IRET, HALT, WFI and PUSH/POP instructions.

See the interrupt management chapter for more details.

**CENTRAL PROCESSING UNIT (Cont'd)**

**Stack Pointer (SP)**

Read/Write  
Reset Value: 01 FFh



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 8).

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.

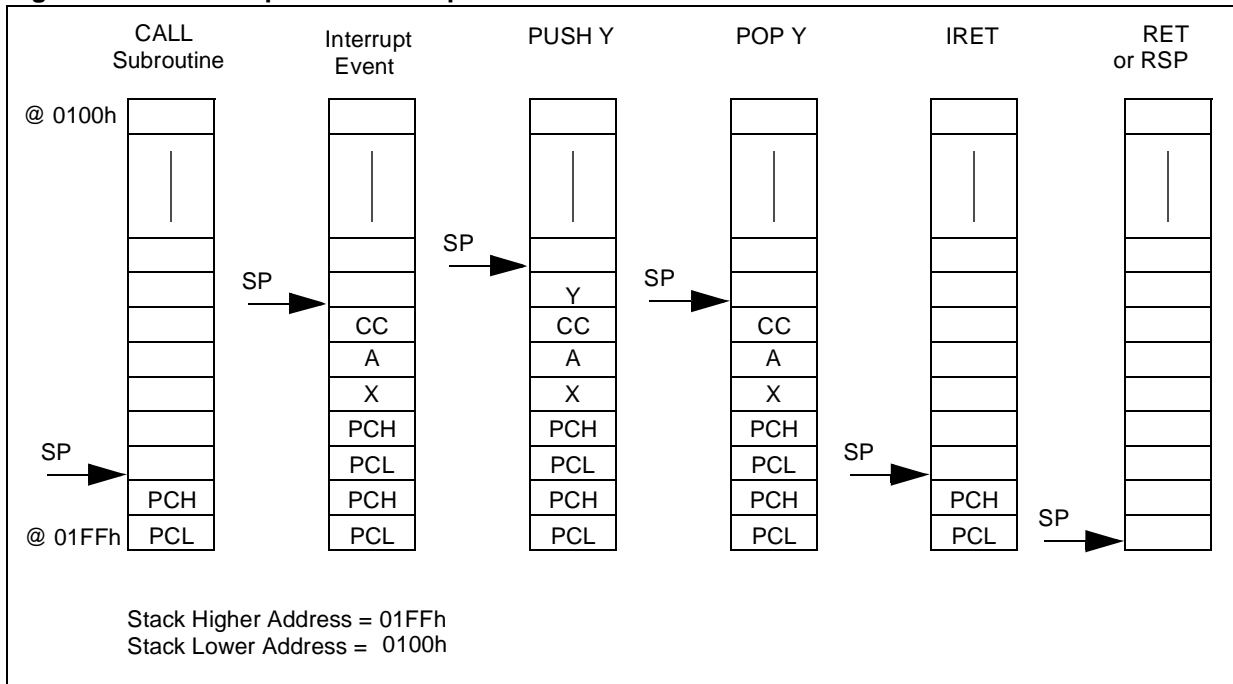
**Note:** When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

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 8.

- 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 External RC oscillator
  - 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
  - Clock Security System (CSS) with Clock Filter

and Backup Safe Oscillator (enabled by option byte)

### 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 168.

Figure 9. PLL Block Diagram

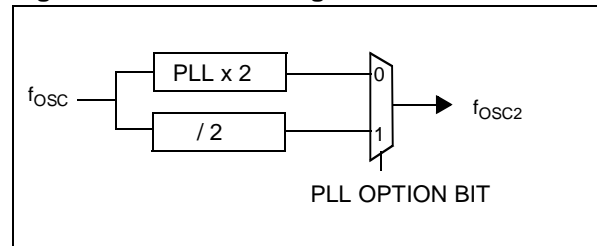
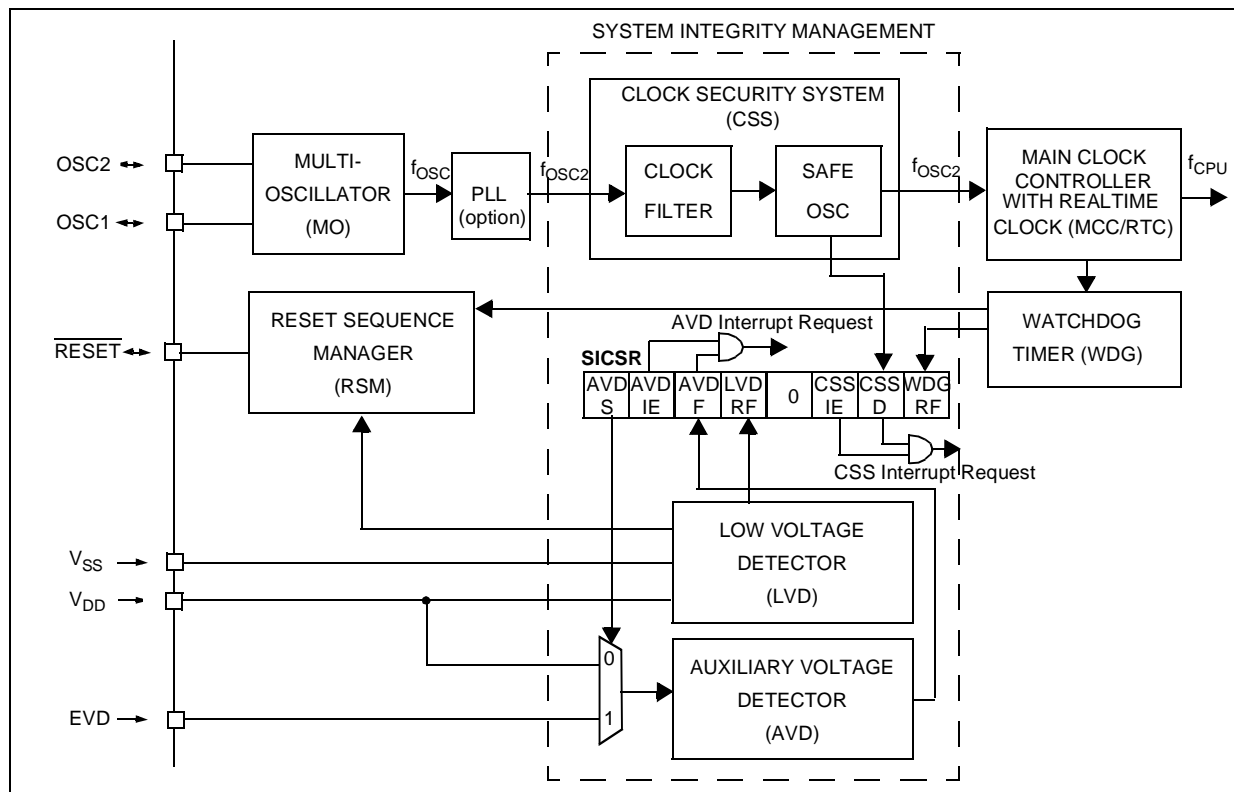


Figure 10. Clock, Reset and Supply Block Diagram





## 6.2 MULTI-OSCILLATOR (MO)

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

- an external source
- 4 crystal or ceramic resonator oscillators
- an external RC oscillator
- 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.

### 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 14.1 on page 191 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.

### External RC Oscillator

This oscillator allows a low cost solution for the main clock of the ST7 using only an external resistor and an external capacitor. The frequency of the external RC oscillator (in the range of some MHz.) is fixed by the resistor and the capacitor values. Consequently in this MO mode, the accuracy of the clock is directly linked to the accuracy of the discrete components. The corresponding formula is  $f_{OSC} = 5 / (R_{EX} C_{EX})$ .

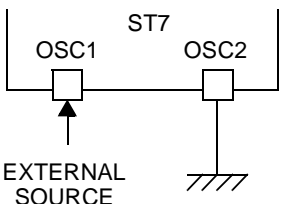
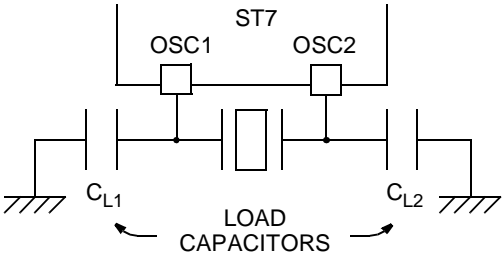
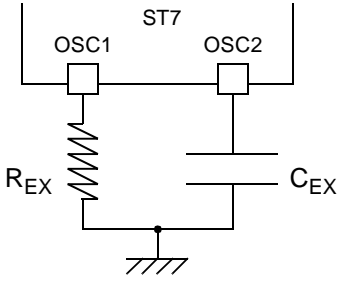
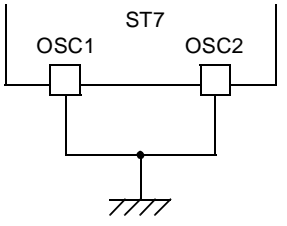
### Internal RC Oscillator

The internal RC oscillator mode is based on the same principle as the external RC oscillator includ-

ing the resistance and the capacitance of the device. This mode is the most cost effective one with the drawback of a lower frequency accuracy. Its frequency is in the range of several MHz.

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 |   |
| External RC Oscillator     |   |
| 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.

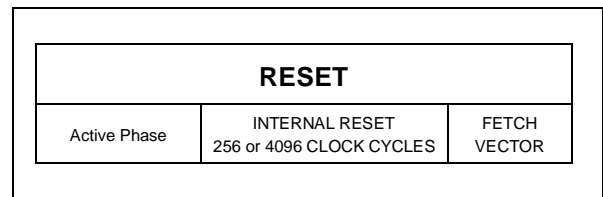
The basic RESET sequence consists of 3 phases as shown in Figure 11:

- 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 14.1 on page 191).

The RESET vector fetch phase duration is 2 clock cycles.

Figure 11. RESET Sequence Phases

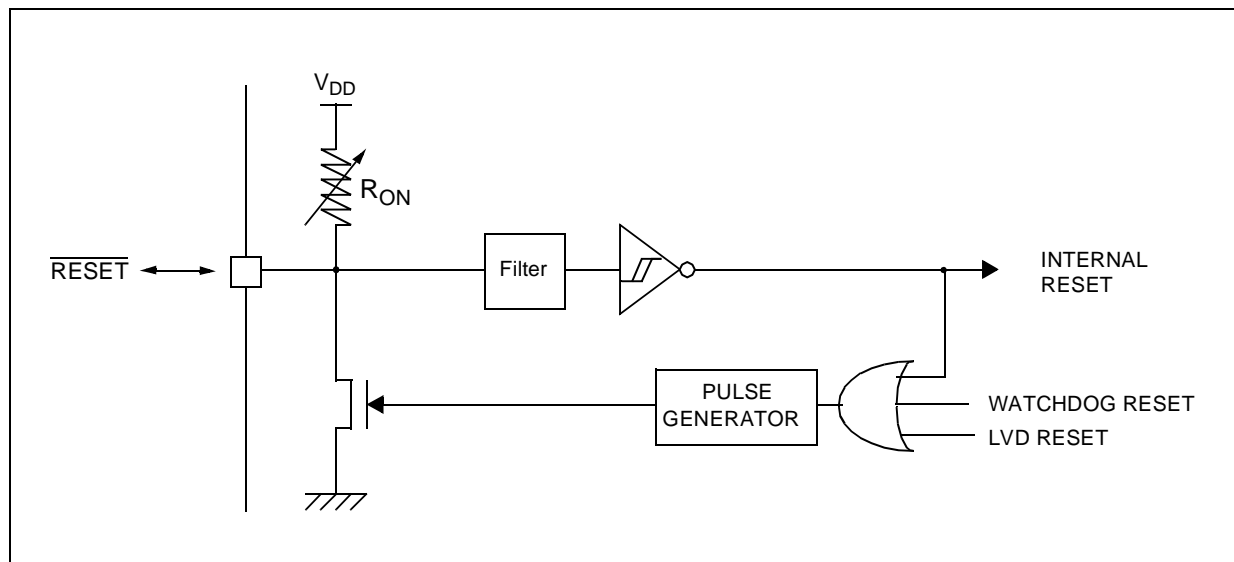


#### 6.3.2 Asynchronous External $\overline{\text{RESET}}$ pin

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 178 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  $t_{w(\text{RSTL})\text{out}}$  (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  $\overline{\text{RESET}}$  pulse recognition, the device  $\overline{\text{RESET}}$  pin acts as an output that is pulled low during at least  $t_{w(\text{RSTL})\text{out}}$ .

**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  $V_{\text{DD}}$  is over the minimum level specified for the selected  $f_{\text{OSC}}$  frequency. (see "OPERATING CONDITIONS" on page 156)

A proper reset signal for a slow rising  $V_{\text{DD}}$  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  $V_{\text{DD}} < V_{\text{IT}+}$  (rising edge) or  $V_{\text{DD}} < V_{\text{IT}-}$  (falling edge) as shown in Figure 13.

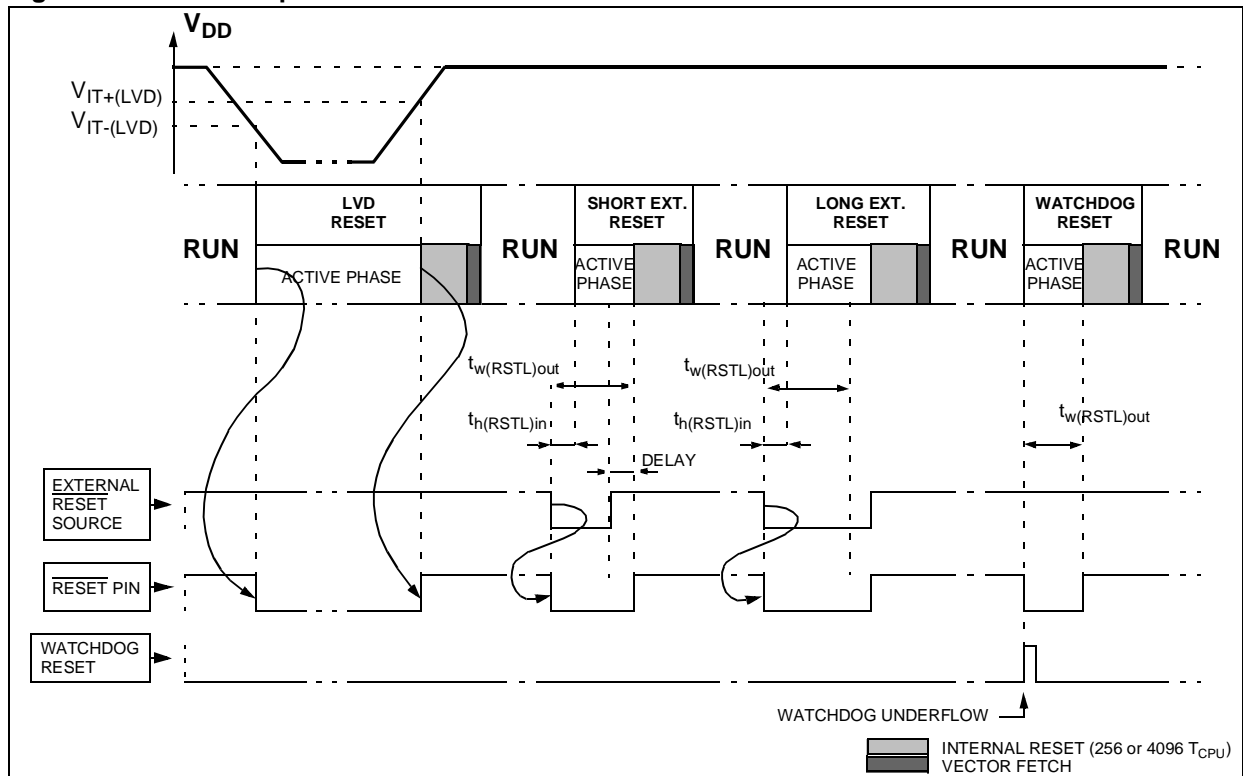
The LVD filters spikes on  $V_{\text{DD}}$  larger than  $t_{\text{g}}(V_{\text{DD}})$  to avoid parasitic resets.

**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  $t_{w(\text{RSTL})\text{out}}$ .

**Figure 13. RESET Sequences**



**6.4 SYSTEM INTEGRITY MANAGEMENT (SI)**

The System Integrity Management block contains the Low Voltage Detector (LVD), Auxiliary Voltage Detector (AVD) and Clock Security System (CSS) 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.

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.

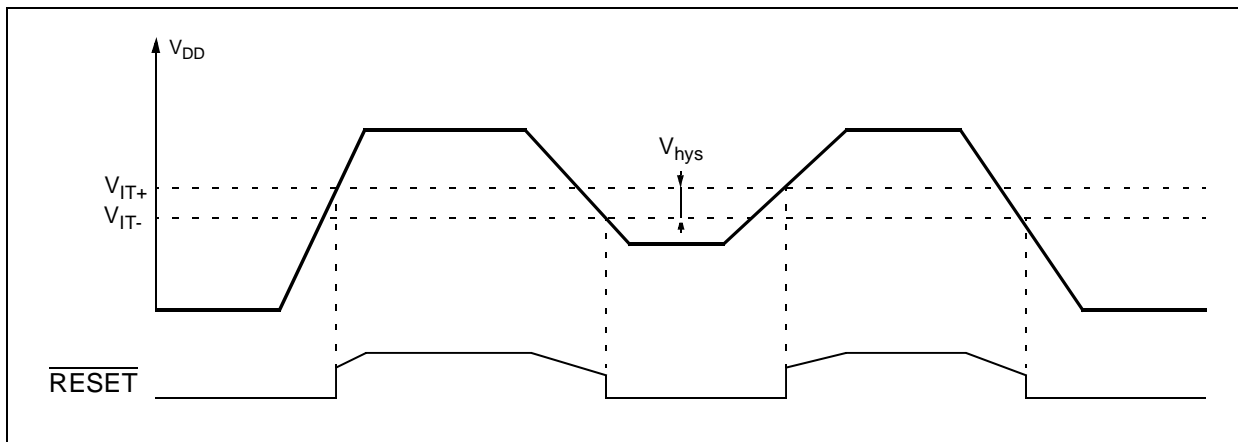
During a Low Voltage Detector Reset, the  $\overline{RESET}$  pin is held low, thus permitting the MCU to reset other devices.

**Notes:**

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

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

**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 14.1 on page 191).

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.

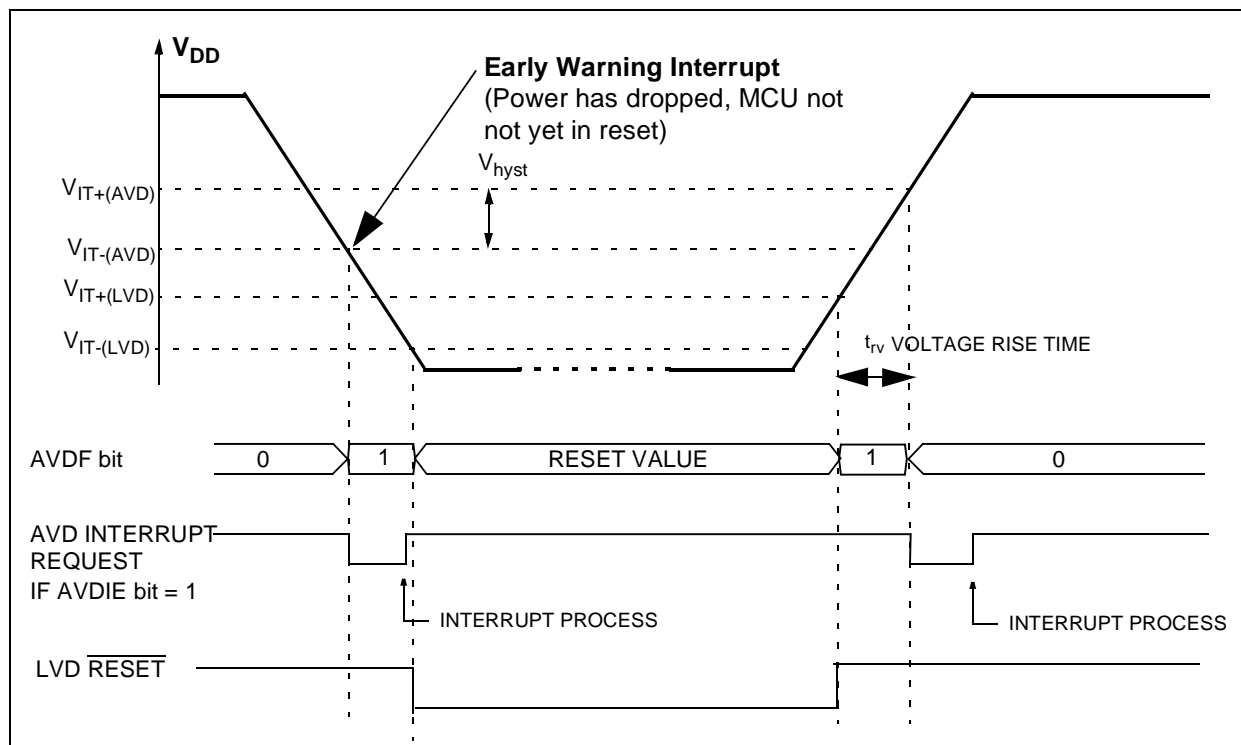
The interrupt on the rising edge is used to inform the application that the  $V_{DD}$  warning state is over.

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)



**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

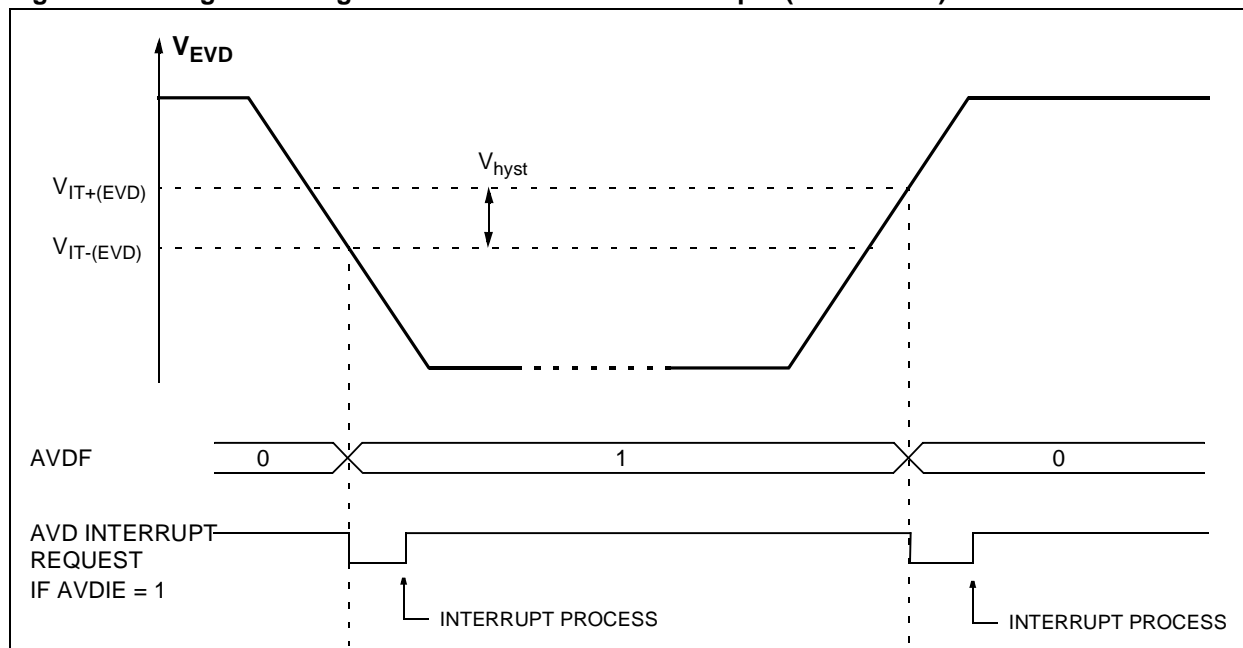
of the comparator output. This means it is generated when either one of these two events occur:

- $V_{EVD}$  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)**



**SYSTEM INTEGRITY MANAGEMENT (Cont'd)**

**6.4.3 Clock Security System (CSS)**

The Clock Security System (CSS) protects the ST7 against breakdowns, spikes and overfrequencies occurring on the main clock source ( $f_{OSC}$ ). It is based on a clock filter and a clock detection control with an internal safe oscillator ( $f_{SFOSC}$ ).

**6.4.3.1 Clock Filter Control**

The PLL has an integrated glitch filtering capability making it possible to protect the internal clock from overfrequencies created by individual spikes. This feature is available only when the PLL is enabled. If glitches occur on  $f_{OSC}$  (for example, due to loose connection or noise), the CSS filters these automatically, so the internal CPU frequency ( $f_{CPU}$ ) continues deliver a glitch-free signal (see Figure 17).

**6.4.3.2 Clock detection Control**

If the clock signal disappears (due to a broken or disconnected resonator...), the safe oscillator delivers a low frequency clock signal ( $f_{SFOSC}$ ) which allows the ST7 to perform some rescue operations.

Automatically, the ST7 clock source switches back from the safe oscillator ( $f_{SFOSC}$ ) if the main clock source ( $f_{OSC}$ ) recovers.

When the internal clock ( $f_{CPU}$ ) is driven by the safe oscillator ( $f_{SFOSC}$ ), the application software is notified by hardware setting the CSSD bit in the SICSR register. An interrupt can be generated if the

CSSIE bit has been previously set. These two bits are described in the SICSR register description.

**6.4.4 Low Power Modes**

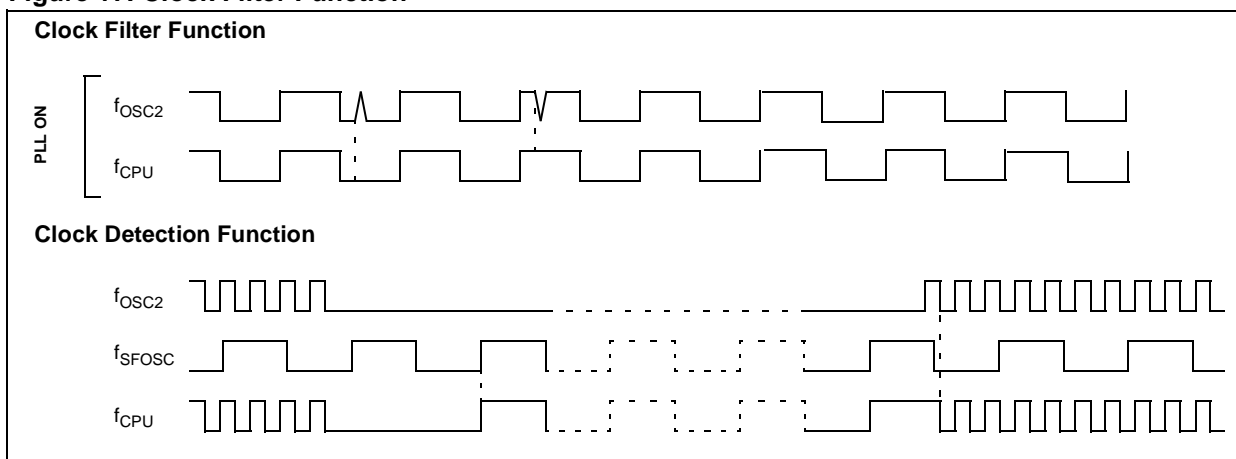
| Mode | Description  |
|------|--|
| WAIT | No effect on SI. CSS and AVD interrupts cause the device to exit from Wait mode.   |
| HALT | The CRSR register is frozen. The CSS (including the safe oscillator) is disabled until HALT mode is exited. The previous CSS configuration resumes when the MCU is 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. |

**6.4.4.1 Interrupts**

The CSS or AVD interrupt events generate an interrupt if the corresponding Enable Control Bit (CSSIE or 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 |
|---|------------|--------------------|----------------|----------------|
| CSS event detection (safe oscillator activated as main clock) | CSSD       | CSSIE              | Yes            | No             |
| AVD event   | AVDF       | AVDIE              | Yes            | No             |

**Figure 17. Clock Filter Function**



**SYSTEM INTEGRITY MANAGEMENT (Cont'd)**

**6.4.5 Register Description**

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

Read/Write

Reset Value: 000x 000x (00h)

|          |           |          |           |   |           |          |           |
|----------|-----------|----------|-----------|---|-----------|----------|-----------|
| 7        |           |          |           |   |           |          | 0         |
| AVD<br>S | AVD<br>IE | AVD<br>F | LVD<br>RF | 0 | CSS<br>IE | CSS<br>D | WDG<br>RF |

**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<sub>DD</sub> 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<sub>DD</sub> or V<sub>EVD</sub> over V<sub>IT+(AVD)</sub> threshold  
 1: V<sub>DD</sub> or V<sub>EVD</sub> under V<sub>IT-(AVD)</sub> 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.

Bit 3 = Reserved, must be kept cleared.

**Bit 2 = CSSIE Clock security syst interrupt enable**  
 This bit enables the interrupt when a disturbance

is detected by the Clock Security System (CSSD bit set). It is set and cleared by software.  
 0: Clock security system interrupt disabled  
 1: Clock security system interrupt enabled  
 When the CSS is disabled by OPTION BYTE, the CSSIE bit has no effect.

**Bit 1 = CSSD Clock security system detection**  
 This bit indicates that the safe oscillator of the Clock Security System block has been selected by hardware due to a disturbance on the main clock signal (f<sub>OSC</sub>). It is set by hardware and cleared by reading the SICSR register when the original oscillator recovers.  
 0: Safe oscillator is not active  
 1: Safe oscillator has been activated  
 When the CSS is disabled by OPTION BYTE, the CSSD bit value is forced to 0.

**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 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
  - 3 non maskable events: TLI, RESET, TRAP

This interrupt management is based on:

- Bit 5 and bit 3 of the CPU CC register (I1:0),
- Interrupt software priority registers (ISPRx),
- Fixed interrupt vector addresses located at the high addresses of the memory map (FFE0h to FFFFh) sorted by hardware priority order.

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

### 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 18

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).

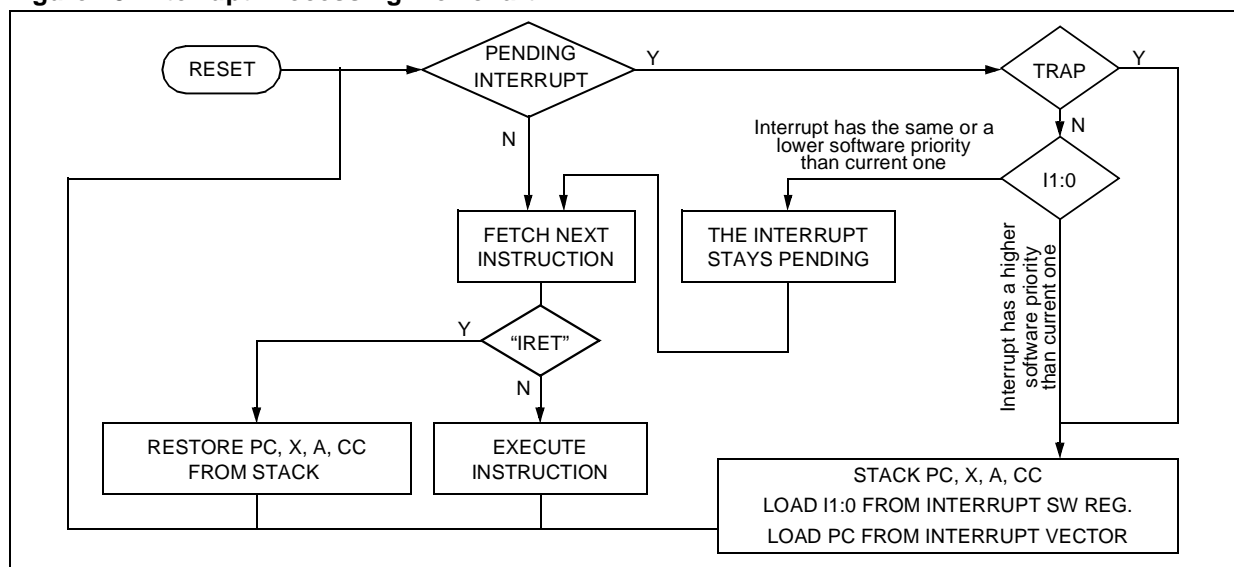
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

**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 18. Interrupt Processing Flowchart**



## INTERRUPTS (Cont'd)

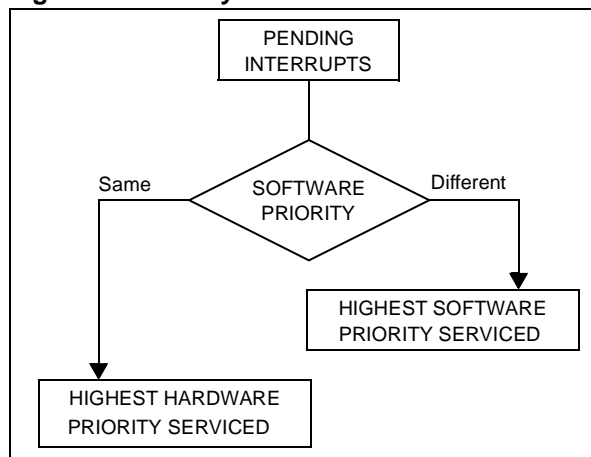
### Servicing Pending Interrupts

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

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

Figure 19 describes this decision process.

**Figure 19. Priority Decision Process**



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

**Note 1:** The hardware priority is exclusive while the software one is not. This allows the previous process to succeed with only one interrupt.

**Note 2:** TLI, RESET and TRAP are non maskable and they can be considered as having the highest software priority in the decision process.

### Different Interrupt Vector Sources

Two interrupt source types are managed by the ST7 interrupt controller: the non-maskable type (RESET, TLI, 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 18). 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.

#### ■ 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 18 as a trap.

**Caution:** A TRAP instruction must not be used in a TLI service routine.

#### ■ 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 18.

**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.

#### ■ 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 19.

**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 20 and Figure 21 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 21. 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 20. Concurrent Interrupt Management

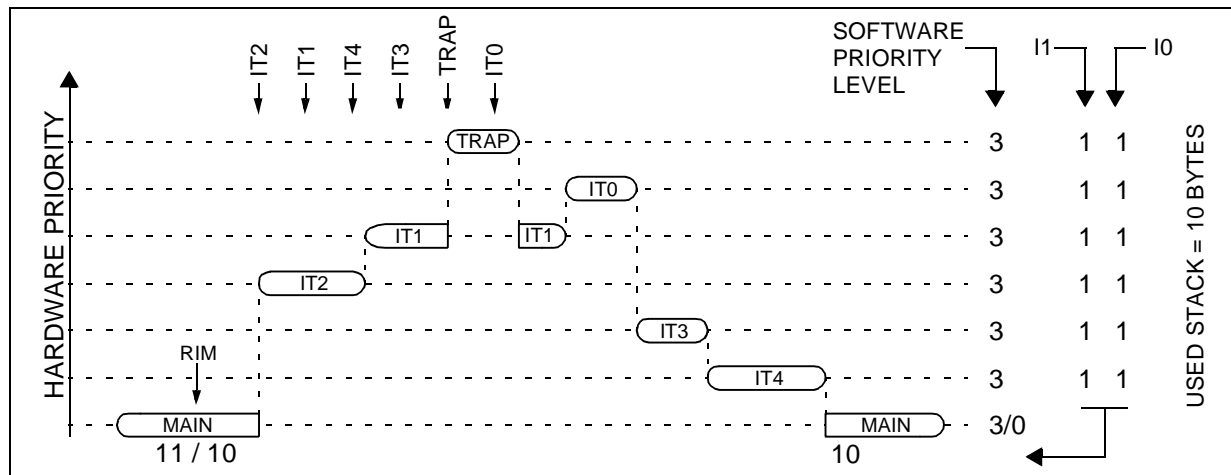
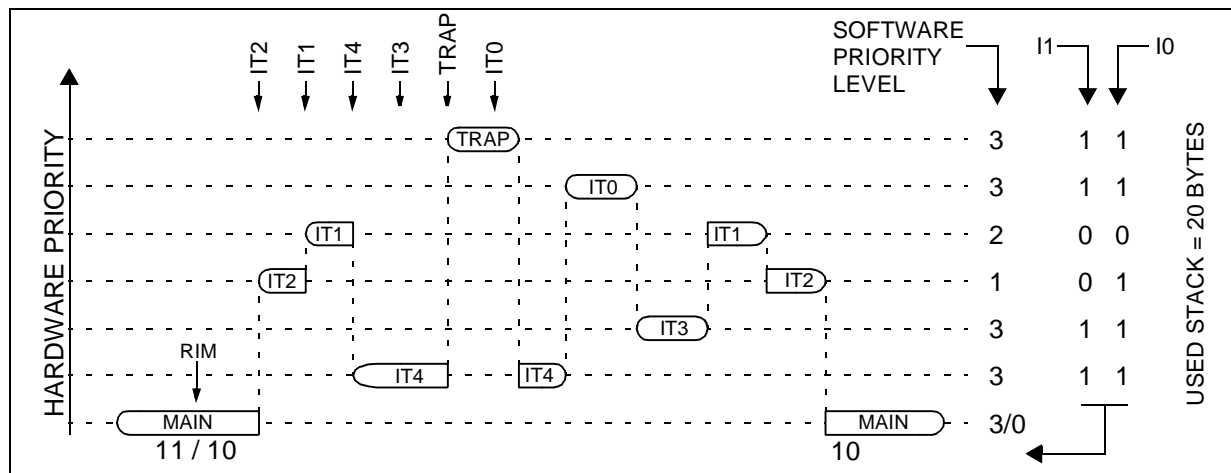


Figure 21. Nested Interrupt Management



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 are non maskable sources and can interrupt a level 3 program.

INTERRUPT SOFTWARE PRIORITY REGISTERS (ISPRX)

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

Reset 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 I0 in I1:0 of CC | 1  |   | 0  |   |   |   |
| SIM         | Disable interrupt (level 3 set) | Load I1 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 <sup>1)</sup> | Address Vector |
|----|--------------|---|----------------|--|------------------------------|----------------|
|    | RESET        | Reset   | N/A            |  | yes                          | FFFEh-FFFFh    |
|    | TRAP         | Software interrupt  |                |  | no                           | FFFC h-FFFDh   |
| 0  | TLI          | External top level interrupt  | EICR           |  | yes                          | FFFAh-FFFBh    |
| 1  | MCC/RTC CSS  | Main clock controller time base interrupt<br>Safe oscillator activation interrupt | MCCSR<br>SICSR | Higher Priority<br><br>↓<br><br>Lower Priority | 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  | CAN          | CAN peripheral interrupts   | CANISR         |  | yes                          | FFEEh-FFEFh    |
| 7  | SPI          | SPI peripheral interrupts   | SPICSR         |  | yes                          | FFEC h-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                          | FFE0h-FFE1h    |

Notes:

1. Valid for HALT and ACTIVE-HALT modes except for the MCC/RTC or CSS interrupt source which exits from ACTIVE-HALT mode only.

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 22). 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

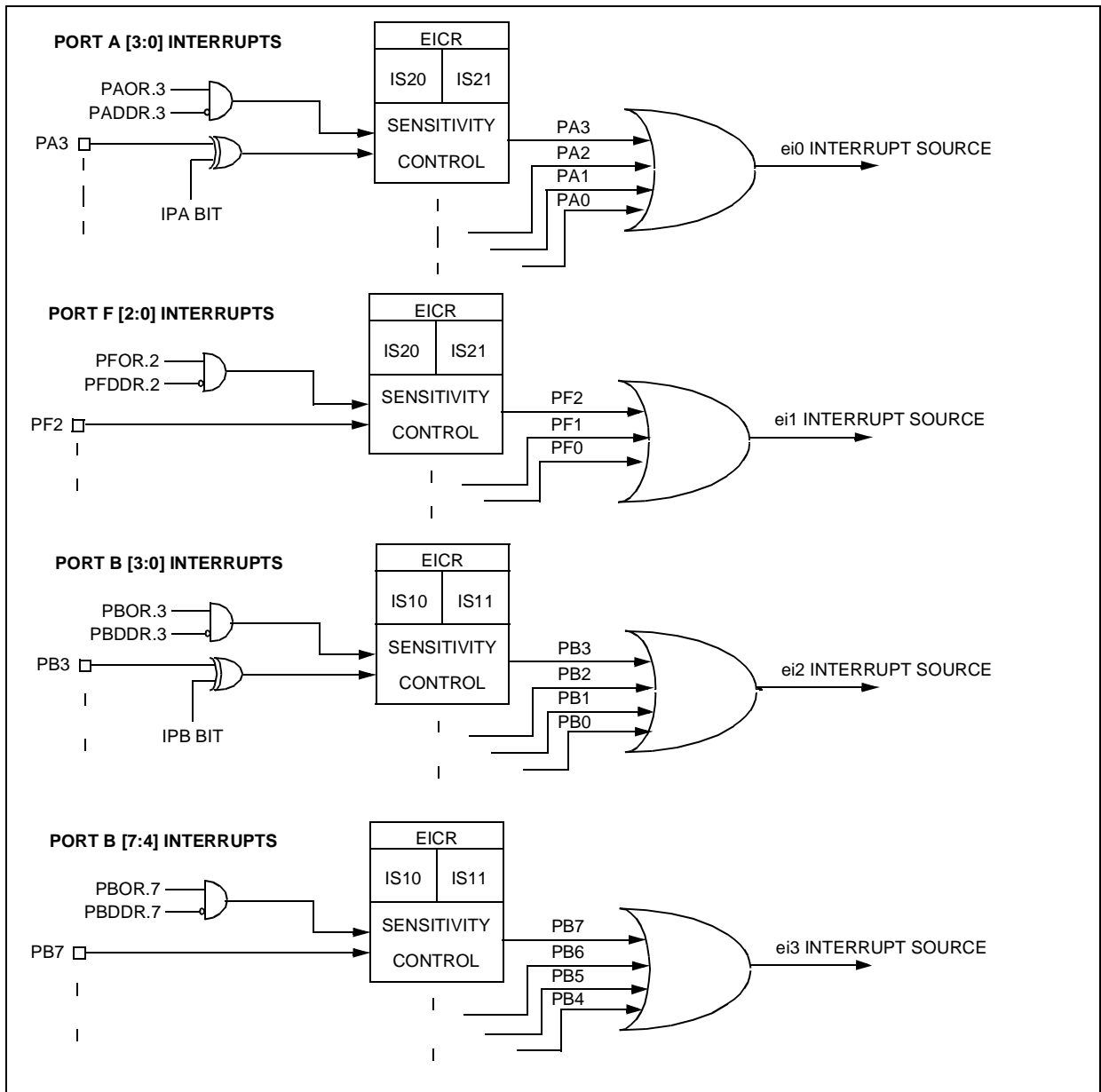
- Falling edge and low level
- Rising edge and high level (only for ei0 and ei2)

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 22. External Interrupt Control bits



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



## 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          | ISPR0<br>Reset Value | ei1        |            | ei0        |            | MCC + SI   |            | TLI        |            |
|                |                      | I1_3<br>1  | I0_3<br>1  | I1_2<br>1  | I0_2<br>1  | I1_1<br>1  | I0_1<br>1  | 1          | 1          |
| 0025h          | ISPR1<br>Reset Value | SPI        |            | CAN        |            | ei3        |            | ei2        |            |
|                |                      | I1_7<br>1  | I0_7<br>1  | I1_6<br>1  | I0_6<br>1  | I1_5<br>1  | I0_5<br>1  | I1_4<br>1  | I0_4<br>1  |
| 0026h          | ISPR2<br>Reset Value | AVD        |            | SCI        |            | TIMER B    |            | TIMER A    |            |
|                |                      | I1_11<br>1 | I0_11<br>1 | I1_10<br>1 | I0_10<br>1 | I1_9<br>1  | I0_9<br>1  | I1_8<br>1  | I0_8<br>1  |
| 0027h          | ISPR3<br>Reset Value |            |            |            |            | PWMART     |            | I2C        |            |
|                |                      | 1          | 1          | 1          | 1          | I1_13<br>1 | I0_13<br>1 | I1_12<br>1 | I0_12<br>1 |
| 0028h          | EICR<br>Reset Value  | IS11<br>0  | IS10<br>0  | IPB<br>0   | IS21<br>0  | IS20<br>0  | IPA<br>0   | TLIS<br>0  | TLIE<br>0  |

## 8 POWER SAVING MODES

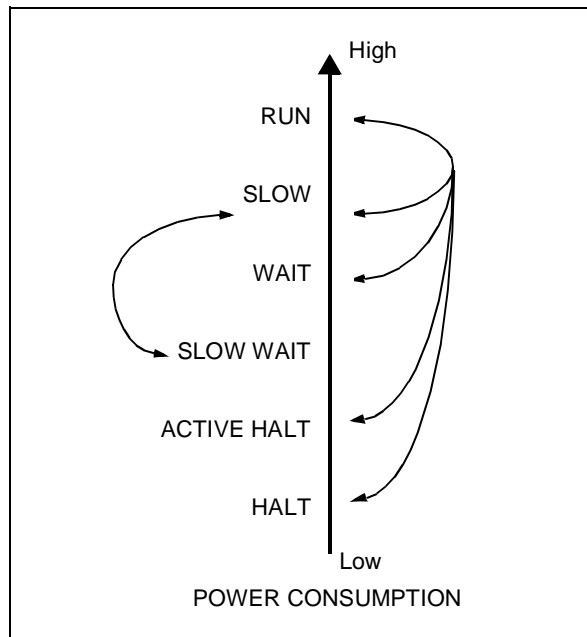
### 8.1 INTRODUCTION

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 23): 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 23. Power Saving Mode Transitions



### 8.2 SLOW MODE

This mode has two targets:

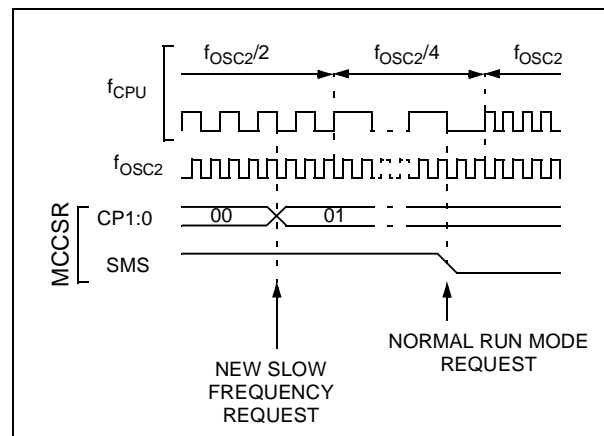
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency ( $f_{CPU}$ ) to the available supply voltage.

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 24. SLOW Mode Clock Transitions



## POWER SAVING MODES (Cont'd)

## 8.3 WAIT MODE

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

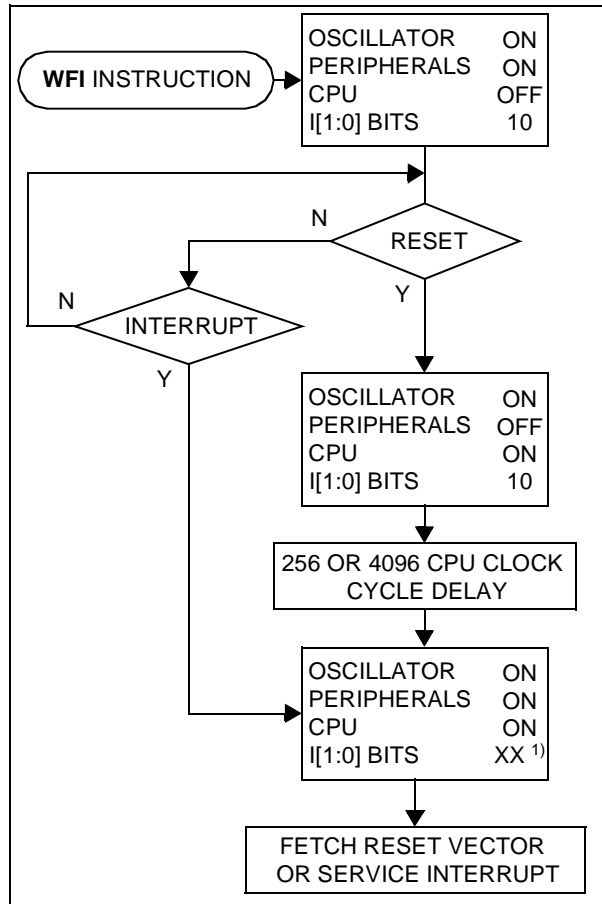
This power saving mode is selected by calling the 'WFI' instruction.

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 25.

Figure 25. WAIT Mode Flow-chart

**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 10.2 on page 58 for more details on the MCCSR register).

The MCU can exit ACTIVE-HALT mode on reception of either an MCC/RTC interrupt, a specific interrupt (see Table 8, "Interrupt Mapping," on page 38) 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 27).

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 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 de-

pending on option byte). Otherwise, the ST7 enters HALT mode for the remaining  $t_{DELAY}$  period.

Figure 26. ACTIVE-HALT Timing Overview

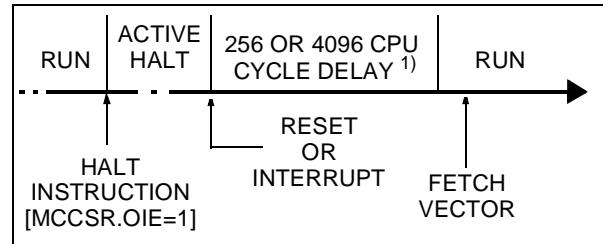
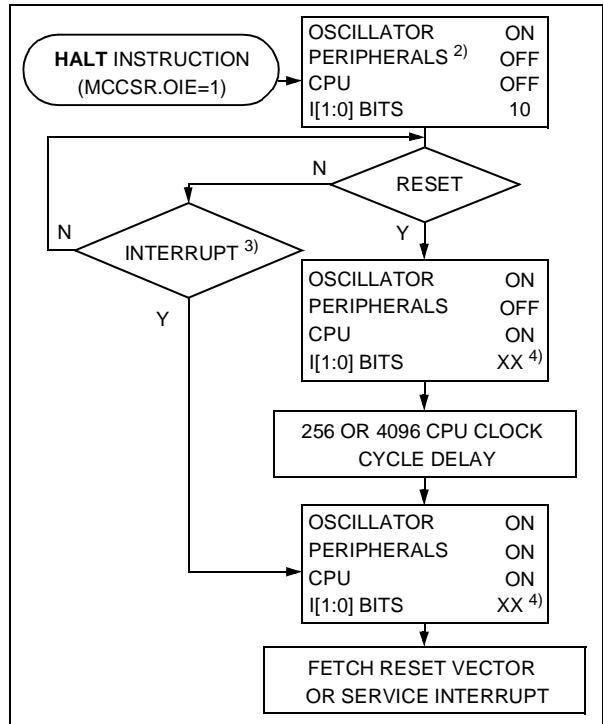


Figure 27. ACTIVE-HALT Mode Flow-chart



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. Only the MCC/RTC interrupt and some specific interrupts can exit the MCU from ACTIVE-HALT mode (such as external interrupt). Refer to Table 8, "Interrupt Mapping," on page 38 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 restored when the CC register is popped.

## 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 10.2 on page 58 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 38) 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 29).

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 14.1 on page 191 for more details).

Figure 28. HALT Timing Overview

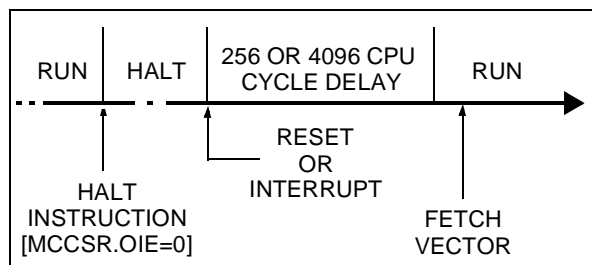
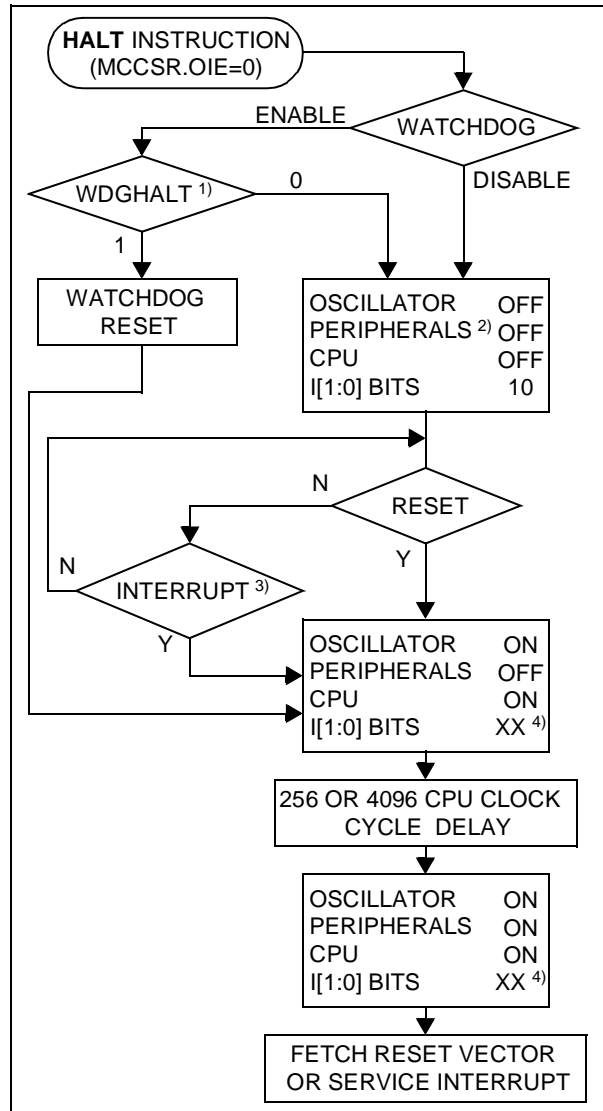


Figure 29. HALT Mode Flow-chart



## Notes:

1. WDGHALT is an option bit. See option byte section for more details.
2. Peripheral clocked with an external clock source can still be active.
3. Only some specific interrupts can exit the MCU from HALT mode (such as external interrupt). Refer to Table 8, "Interrupt Mapping," on page 38 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 sensitivity 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).

## 9 I/O PORTS

### 9.1 INTRODUCTION

The I/O ports offer different functional modes:

- transfer of data through digital inputs and outputs and for specific pins:
- external interrupt generation
- alternate signal input/output for the on-chip peripherals.

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

### 9.2 FUNCTIONAL DESCRIPTION

Each port has 2 main registers:

- Data Register (DR)
- Data Direction Register (DDR)

and one optional register:

- Option Register (OR)

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

The following description takes into account the OR register, (for specific ports which do not provide this register refer to the I/O Port Implementation section). The generic I/O block diagram is shown in Figure 30

#### 9.2.1 Input Modes

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

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

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

#### Notes:

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

#### External interrupt function

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

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

Each external interrupt vector is linked to a dedicated group of I/O port pins (see pinout description and interrupt section). If several input pins are selected simultaneously as interrupt sources, these are first detected according to the sensitivity bits in the EICR register and then logically ORed.

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

#### 9.2.2 Output Modes

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

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

DR register value and output pin status:

| DR | Push-pull       | Open-drain      |
|----|-----------------|-----------------|
| 0  | V <sub>SS</sub> | V <sub>SS</sub> |
| 1  | V <sub>DD</sub> | Floating        |

#### 9.2.3 Alternate Functions

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

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

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

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

I/O PORTS (Cont'd)

Figure 30. I/O Port General Block Diagram

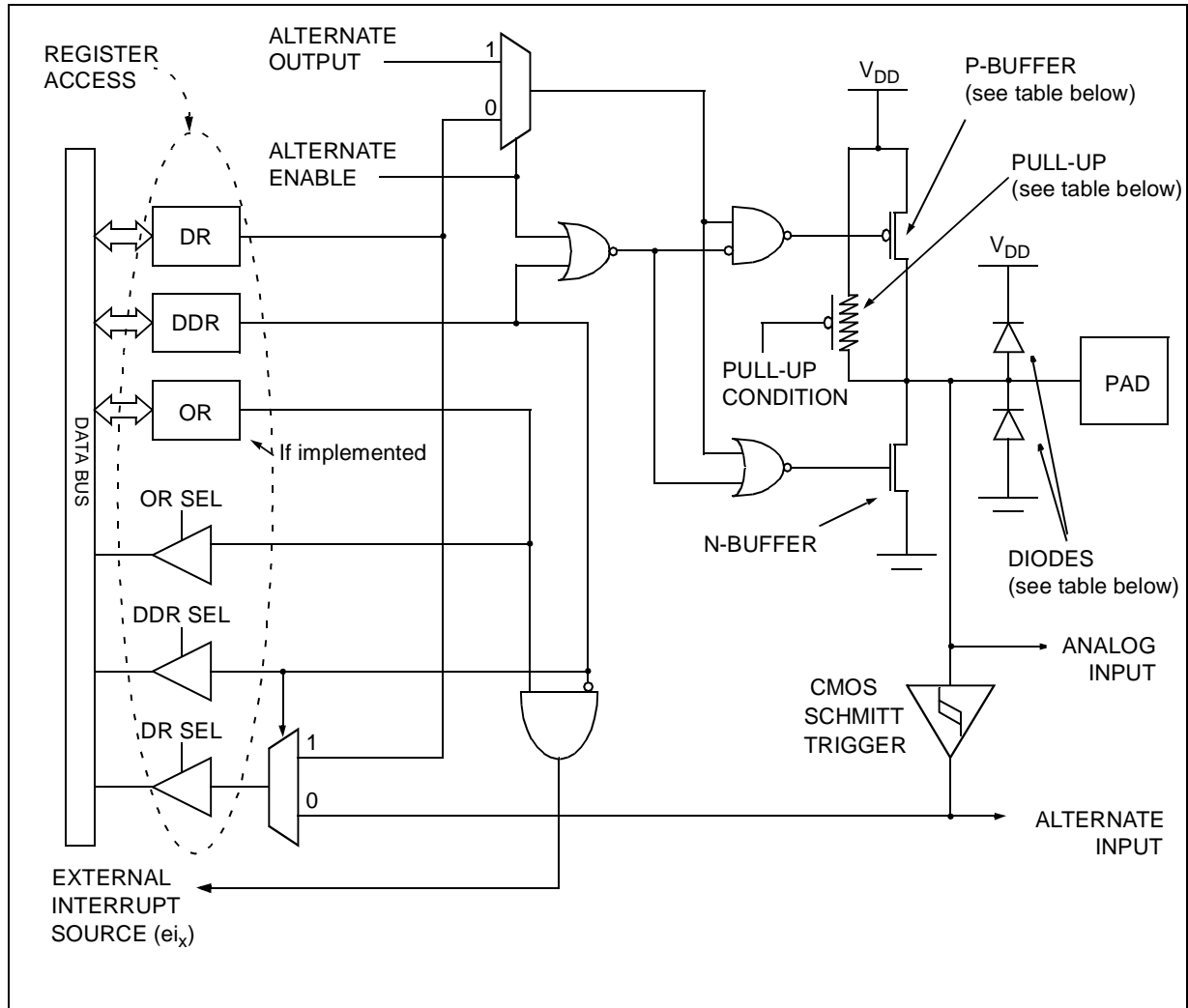


Table 10. I/O Port Mode Options

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

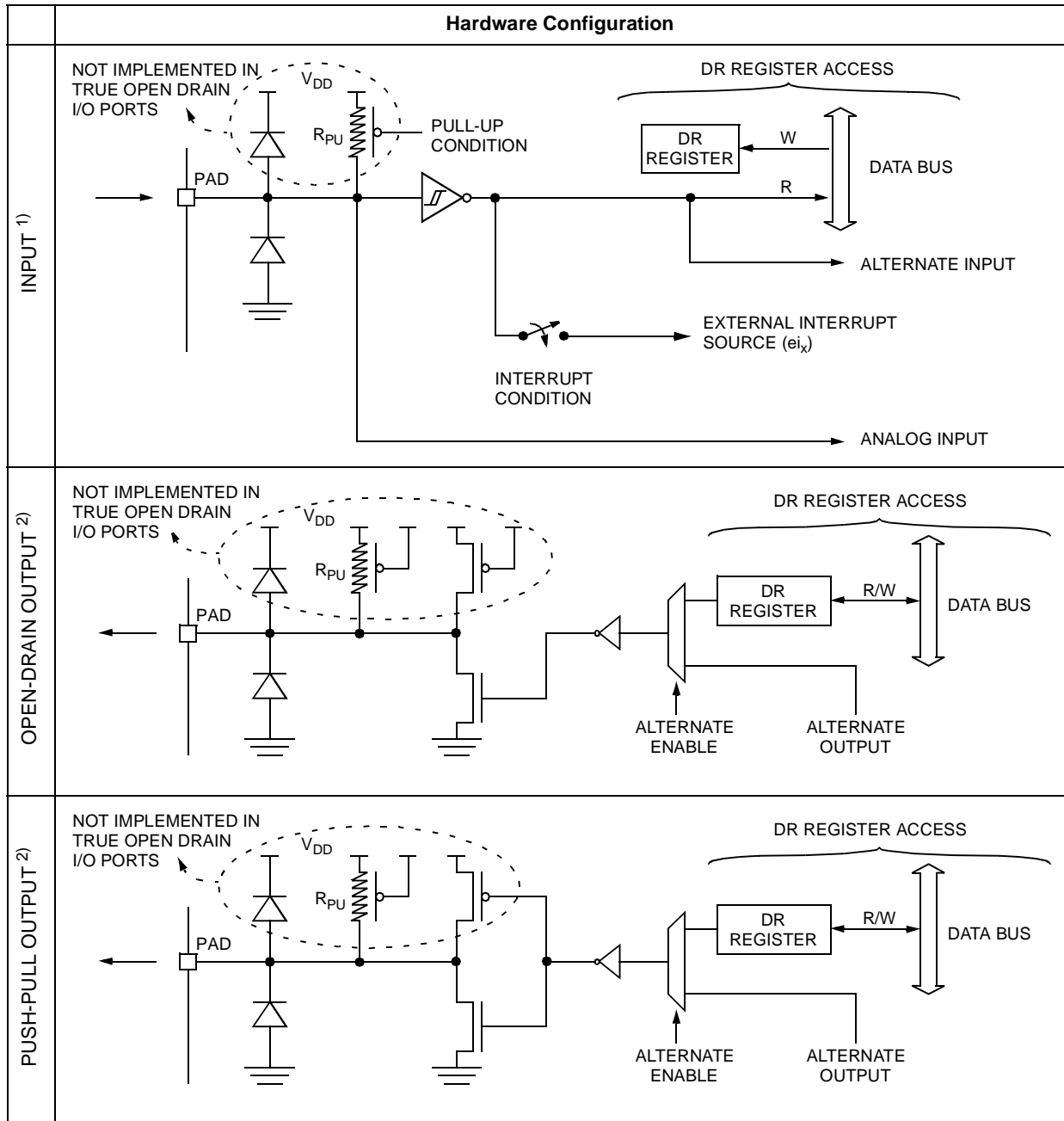
**Legend:** NI - not implemented  
 Off - implemented not activated  
 On - implemented and activated

**Note:** The diode to V<sub>DD</sub> is not implemented in the true open drain pads. A local protection between the pad and V<sub>SS</sub> is implemented to protect the device against positive stress.



I/O PORTS (Cont'd)

Table 11. I/O Port Configurations



Notes:

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

**I/O PORTS (Cont'd)**

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

**Analog alternate function**

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

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

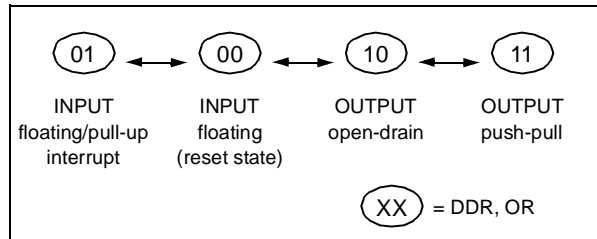
**WARNING:** The analog input voltage level must be within the limits stated in the absolute maximum ratings.

**9.3 I/O PORT IMPLEMENTATION**

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

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in Figure 31 Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

**Figure 31. Interrupt I/O Port State Transitions**



**9.4 LOW POWER MODES**

| Mode | Description  |
|------|--|
| WAIT | No effect on I/O ports. External interrupts cause the device to exit from WAIT mode. |
| HALT | No effect on I/O ports. External interrupts cause the device to exit from HALT mode. |

**9.5 INTERRUPTS**

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers 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 |
|---|------------|--------------------|----------------|----------------|
| External interrupt on selected external event | -          | DDRx ORx           | Yes            | Yes            |

## I/O PORTS (Cont'd)

## 9.5.1 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 (CANTX requirement)

PE2

| MODE          |
|---------------|
| pull-up input |

Table 12. 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,<br>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 |

\* Note: when the CANTX alternate function is selected the I/O port operates in output push-pull mode.

I/O PORTS (Cont'd)

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

| Address (Hex.)                        | Register Label | 7   | 6 | 5 | 4 | 3 | 2 | 1 | 0   |
|---------------------------------------|----------------|-----|---|---|---|---|---|---|-----|
| Reset Value of all 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           |     |   |   |   |   |   |   |     |

## 10 ON-CHIP PERIPHERALS

### 10.1 WATCHDOG TIMER (WDG)

#### 10.1.1 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.

#### 10.1.2 Main Features

- Programmable timer
- 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

#### 10.1.3 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 low the reset pin for typically 500ns.

The application program must write in the WDGCR register at regular intervals during normal operation to prevent an MCU reset. The value to be stored in the WDGCR register must be between FFh and C0h:

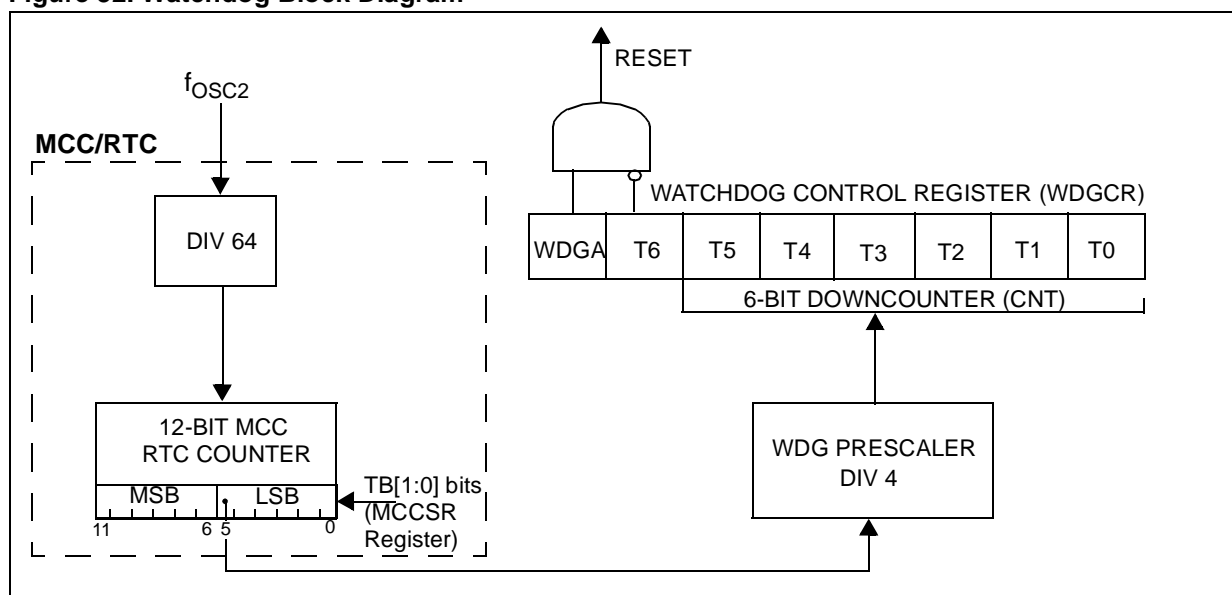
- The WDGA bit is set (watchdog enabled)
- The T6 bit is set to prevent generating an immediate reset
- The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset (see Figure 33. Approximate Timeout Duration). The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WDGCR register (see Figure 34).

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 32. Watchdog Block Diagram



**WATCHDOG TIMER (Cont'd)**

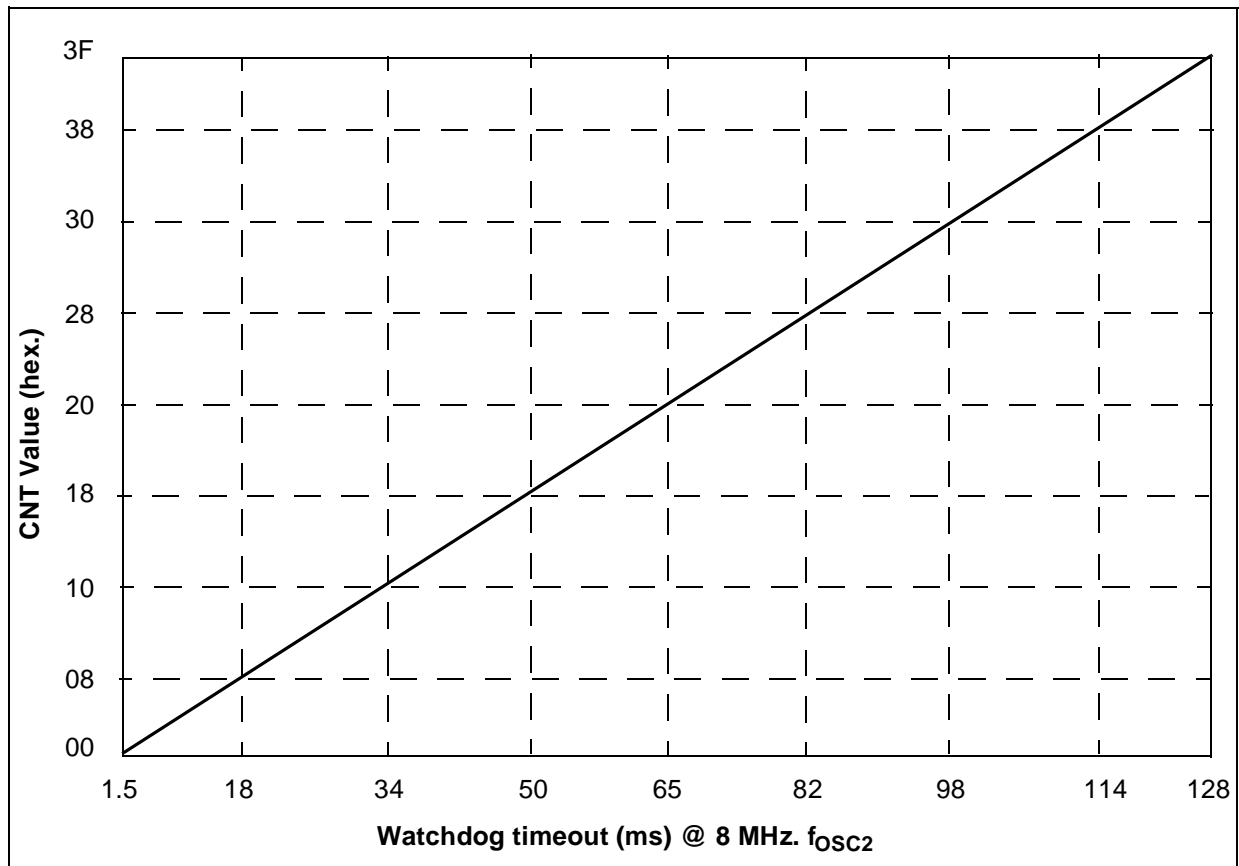
**10.1.4 How to Program the Watchdog Timeout**

Figure 33 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 34.

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

**Figure 33. Approximate Timeout Duration**



## WATCHDOG TIMER (Cont'd)

Figure 34. Exact Timeout Duration ( $t_{\min}$  and  $t_{\max}$ )**WHERE:**

$$t_{\min 0} = (\text{LSB} + 128) \times 64 \times t_{\text{OSC2}}$$

$$t_{\max 0} = 16384 \times t_{\text{OSC2}}$$

$$t_{\text{OSC2}} = 125\text{ns if } f_{\text{OSC2}} = 8 \text{ MHz}$$

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 MCCR register

| TB1 Bit<br>(MCCR Reg.) | TB0 Bit<br>(MCCR Reg.) | Selected MCCR<br>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 ( $t_{\min}$ ):**

$$\text{IF } \text{CNT} < \left\lfloor \frac{\text{MSB}}{4} \right\rfloor \quad \text{THEN} \quad t_{\min} = t_{\min 0} + 16384 \times \text{CNT} \times t_{\text{osc2}}$$

$$\text{ELSE} \quad t_{\min} = t_{\min 0} + \left[ 16384 \times \left( \text{CNT} - \left\lfloor \frac{4\text{CNT}}{\text{MSB}} \right\rfloor \right) + (192 + \text{LSB}) \times 64 \times \left\lfloor \frac{4\text{CNT}}{\text{MSB}} \right\rfloor \right] \times t_{\text{osc2}}$$

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

$$\text{IF } \text{CNT} \leq \left\lfloor \frac{\text{MSB}}{4} \right\rfloor \quad \text{THEN} \quad t_{\max} = t_{\max 0} + 16384 \times \text{CNT} \times t_{\text{osc2}}$$

$$\text{ELSE} \quad t_{\max} = t_{\max 0} + \left[ 16384 \times \left( \text{CNT} - \left\lfloor \frac{4\text{CNT}}{\text{MSB}} \right\rfloor \right) + (192 + \text{LSB}) \times 64 \times \left\lfloor \frac{4\text{CNT}}{\text{MSB}} \right\rfloor \right] \times t_{\text{osc2}}$$

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

**Example:**

With 2ms timeout selected in MCCR register

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

**WATCHDOG TIMER (Cont'd)**

**10.1.5 Low Power Modes**

| Mode | Description              |                            |   |
|------|--------------------------|----------------------------|---|
| SLOW | No effect on Watchdog.   |                            |   |
| WAIT | No effect on Watchdog.   |                            |   |
| HALT | OIE bit in MCCR 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.<br>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 10.1.7 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.  |

**10.1.6 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.

**10.1.7 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.

**10.1.8 Interrupts**

None.

**10.1.9 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<sub>OSC2</sub> cycles (approx.). A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).



**Table 14. Watchdog Timer Register Map and Reset Values**

| Address (Hex.) | Register Label              | 7         | 6       | 5       | 4       | 3       | 2       | 1       | 0       |
|----------------|-----------------------------|-----------|---------|---------|---------|---------|---------|---------|---------|
| 002Ah          | <b>WDGCR</b><br>Reset Value | WDGA<br>0 | T6<br>1 | T5<br>1 | T4<br>1 | T3<br>1 | T2<br>1 | T1<br>1 | T0<br>1 |

### 10.2 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.

#### 10.2.1 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.

#### 10.2.2 Clock-out Capability

The clock-out capability is an alternate function of an I/O port pin that outputs a  $f_{OSC2}$  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.

#### 10.2.3 Real Time Clock Timer (RTC)

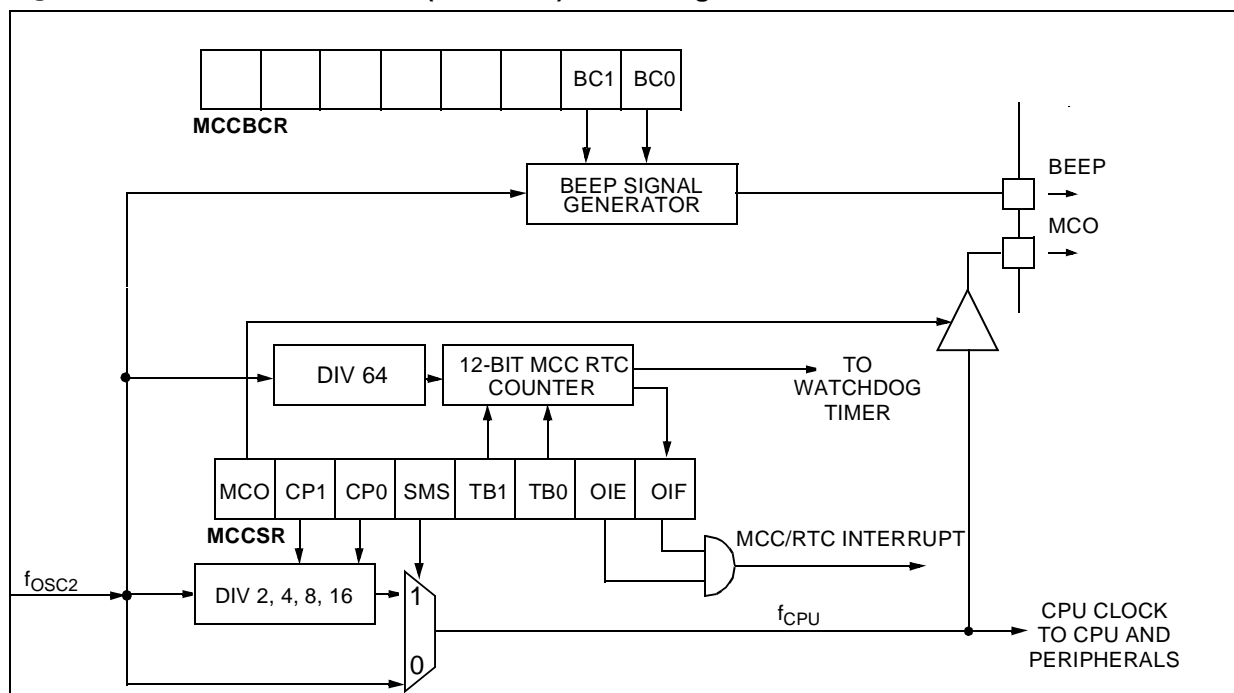
The counter of the real time clock timer allows an interrupt to be generated based on an accurate real time clock. Four different time bases depending directly on  $f_{OSC2}$  are available. The whole functionality is controlled by four bits of the MCCSR register: TB[1:0], OIE and OIF.

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.

#### 10.2.4 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 35. Main Clock Controller (MCC/RTC) Block Diagram



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

### 10.2.5 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. |

### 10.2.6 Interrupts

The MCC/RTC interrupt event generates an interrupt if the OIE bit of the MCCR 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 <sup>1)</sup> |

#### Note:

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

### 10.2.7 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 10.2 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 |
|-------------------|-------------------|-------------------|-----|-----|
|                   | $f_{OSC2} = 4MHz$ | $f_{OSC2} = 8MHz$ |     |     |
| 16000             | 4ms               | 2ms               | 0   | 0   |
| 32000             | 8ms               | 4ms               | 0   | 1   |
| 80000             | 20ms              | 10ms              | 1   | 0   |
| 200000            | 50ms              | 25ms              | 1   | 1   |

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

#### Bit 1 = OIE Oscillator interrupt enable

This bit set and cleared by software.

0: Oscillator interrupt disabled

1: Oscillator interrupt enabled

This interrupt can be used to exit from ACTIVE-HALT mode.

When this bit is set, calling the ST7 software HALT instruction enters the ACTIVE-HALT power saving mode.

**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 $f_{OSC2}=8\text{MHz}$ |  |
|-----|-----|---------------------------------------|--|
| 0   | 0   | Off                                   |  |
| 0   | 1   | ~2-KHz                                | Output<br>Beep signal<br>~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 15. Main Clock Controller Register Map and Reset Values**

| Address (Hex.) | Register Label               | 7        | 6         | 5        | 4          | 3        | 2         | 1         | 0          |
|----------------|------------------------------|----------|-----------|----------|------------|----------|-----------|-----------|------------|
| 002Bh          | <b>SICSR</b><br>Reset Value  | VDS<br>0 | VDIE<br>0 | VDF<br>0 | LVDRF<br>x | 0        | CFIE<br>0 | CSSD<br>0 | WDGRF<br>x |
| 002Ch          | <b>MCCSR</b><br>Reset Value  | MCO<br>0 | CP1<br>0  | CP0<br>0 | SMS<br>0   | TB1<br>0 | TB0<br>0  | OIE<br>0  | OIF<br>0   |
| 002Dh          | <b>MCCBCR</b><br>Reset Value | 0        | 0         | 0        | 0          | 0        | 0         | BC1<br>0  | BC0<br>0   |

**10.3 PWM AUTO-RELOAD TIMER (ART)**

**10.3.1 Introduction**

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

These resources allow five possible operating modes:

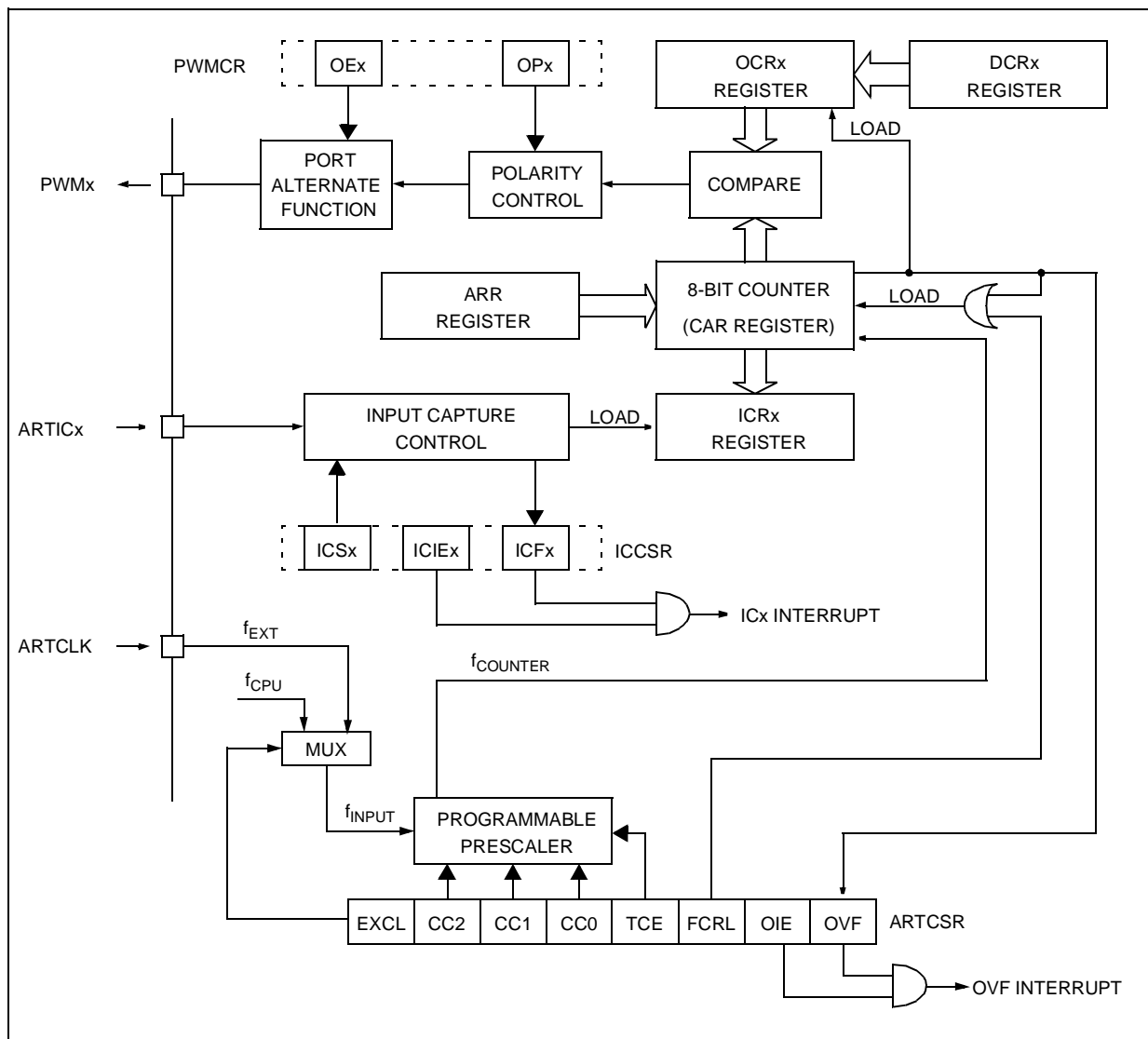
- Generation of up to 4 independent PWM signals
- Output compare and Time base interrupt

- 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 36. PWM Auto-Reload Timer Block Diagram**



**PWM AUTO-RELOAD TIMER (Cont'd)**

**10.3.2 Functional Description**

**Counter**

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

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

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

**Counter clock and prescaler**

The counter clock frequency is given by:

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

The timer counter's input clock ( $f_{\text{INPUT}}$ ) 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  $n = 0, 1, \dots, 7$ ).

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

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  $f_{\text{INPUT}} = f_{\text{CPU}}$ .

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.

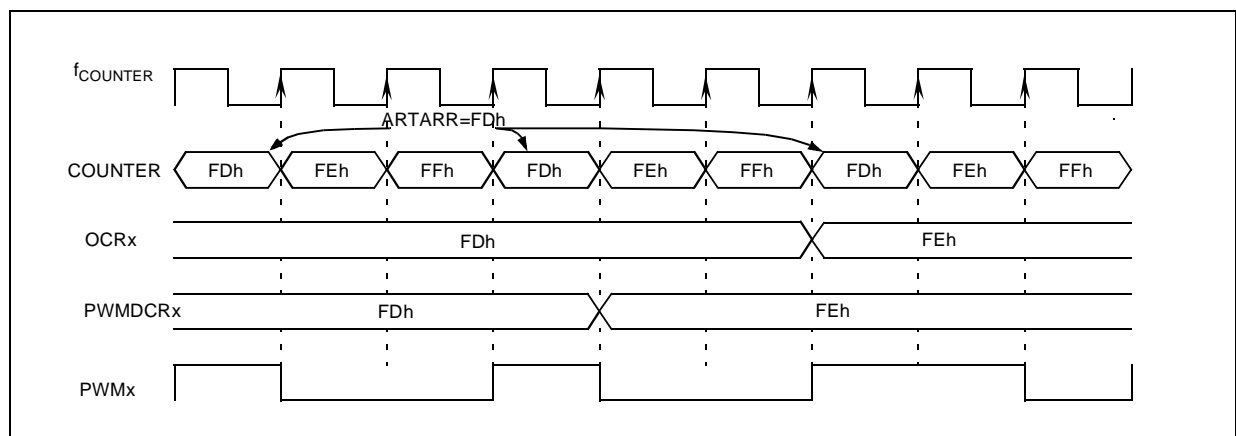
Direct access to the prescaler is not possible.

**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 37. Output compare control**



**PWM AUTO-RELOAD TIMER (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.

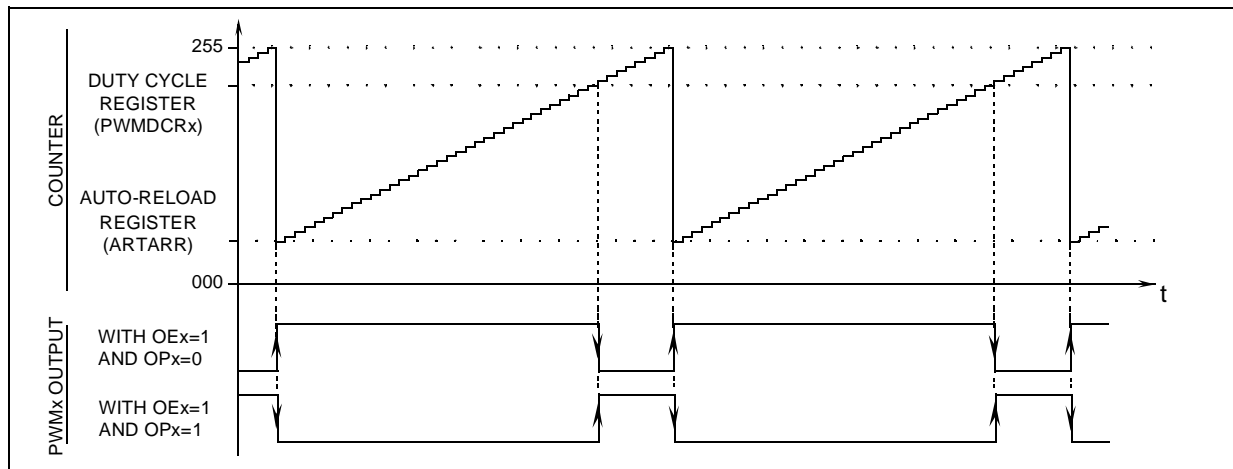
It should be noted that the reload values will also affect the value and the resolution of the duty cycle of the PWM output signal. To obtain a signal on a PWMx pin, the contents of the OCRx register must be greater than the contents of the ARTARR register.

The maximum available resolution for the PWMx duty cycle is:

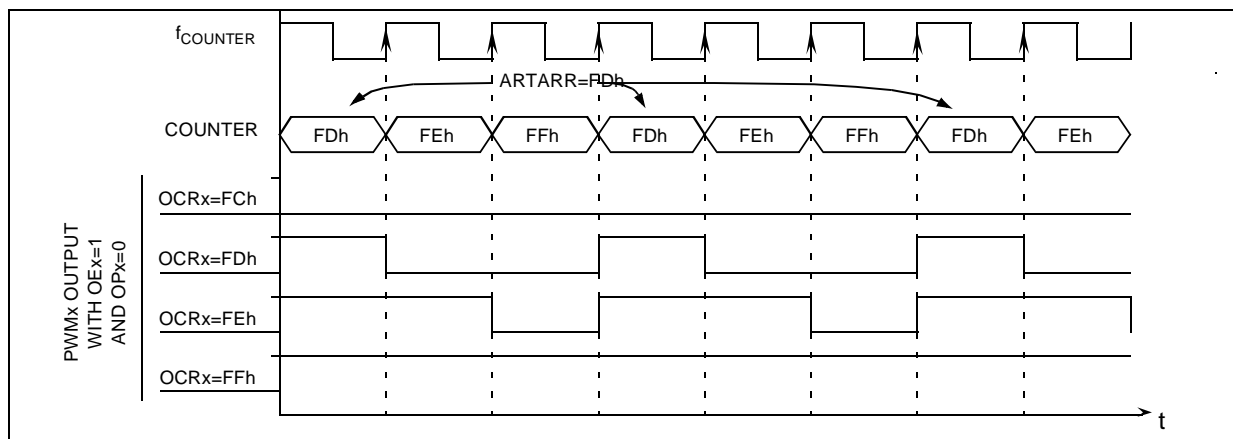
$$\text{Resolution} = 1 / (256 - ARTARR)$$

**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 38. PWM Auto-reload Timer Function**



**Figure 39. PWM Signal from 0% to 100% Duty Cycle**



**PWM AUTO-RELOAD TIMER (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.

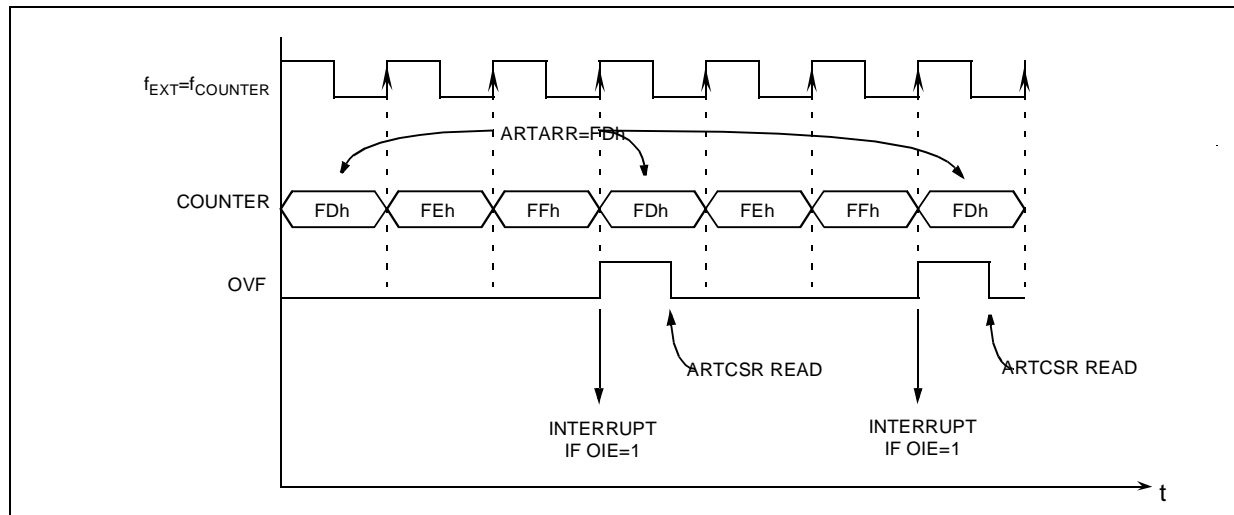
**External clock and event detector mode**

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.

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

When entering HALT mode while  $f_{EXT}$  is selected, all the timer control registers are frozen but the counter continues to increment. If the OIE bit is set, the next overflow of the counter will generate an interrupt which wakes up the MCU.

**Figure 40. External Event Detector Example (3 counts)**





## PWM AUTO-RELOAD TIMER (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 ( $1/f_{\text{COUNTER}}$ ).

**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.

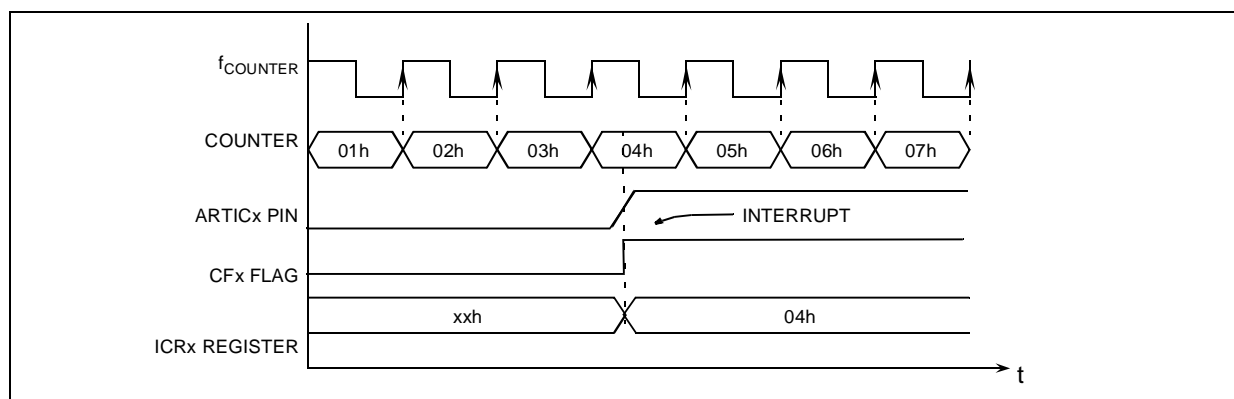
### External interrupt capability

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 41. Input Capture Timing Diagram**



PWM AUTO-RELOAD TIMER (Cont'd)

10.3.3 Register Description

**CONTROL / STATUS REGISTER (ARTCSR)**

Read/Write

Reset Value: 0000 0000 (00h)

|      |     |     |     |     |      |     |     |
|------|-----|-----|-----|-----|------|-----|-----|
| 7    | 0   |     |     |     |      |     |     |
| EXCL | CC2 | CC1 | CC0 | TCE | FCRL | OIE | OVF |

Bit 7 = **EXCL** External Clock

This bit is set and cleared by software. It selects the input clock for the 7-bit prescaler.

0: CPU clock.

1: External clock.

Bit 6:4 = **CC[2:0]** Counter Clock Control

These bits are set and cleared by software. They determine the prescaler division ratio from  $f_{INPUT}$ .

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

Bit 3 = **TCE** Timer Counter Enable

This bit is set and cleared by software. It puts the timer in the lowest power consumption mode.

0: Counter stopped (prescaler and counter frozen).

1: Counter running.

Bit 2 = **FCRL** Force Counter Re-Load

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

Bit 1 = **OIE** Overflow Interrupt Enable

This bit is set and cleared by software. It allows to enable/disable the interrupt which is generated when the OVF bit is set.

0: Overflow Interrupt disable.

1: Overflow Interrupt enable.

Bit 0 = **OVF** Overflow Flag

This bit is set by hardware and cleared by software reading the ARTCSR register. It indicates the transition of the counter from FFh to the ARTARR value.

0: New transition not yet reached

1: Transition reached

**COUNTER ACCESS REGISTER (ARTCAR)**

Read/Write

Reset Value: 0000 0000 (00h)

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7   | 0   |     |     |     |     |     |     |
| CA7 | CA6 | CA5 | CA4 | CA3 | CA2 | CA1 | CA0 |

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

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

**AUTO-RELOAD REGISTER (ARTARR)**

Read/Write

Reset Value: 0000 0000 (00h)

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7   | 0   |     |     |     |     |     |     |
| AR7 | AR6 | AR5 | AR4 | AR3 | AR2 | AR1 | AR0 |

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

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

This register has two PWM management functions:

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

PWM Frequency vs. Resolution:

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

**PWM AUTO-RELOAD TIMER (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.

**PWM AUTO-RELOAD TIMER (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 16. PWM Auto-Reload Timer Register Map and Reset Values

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

## 10.4 16-BIT TIMER

### 10.4.1 Introduction

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

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

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

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

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

### 10.4.2 Main Features

- Programmable prescaler:  $f_{CPU}$  divided by 2, 4 or 8.
- Overflow status flag and maskable interrupt
- External clock input (must be at least 4 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 42.

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

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

### 10.4.3 Functional Description

#### 10.4.3.1 Counter

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

Counter Register (CR):

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

Alternate Counter Register (ACR)

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

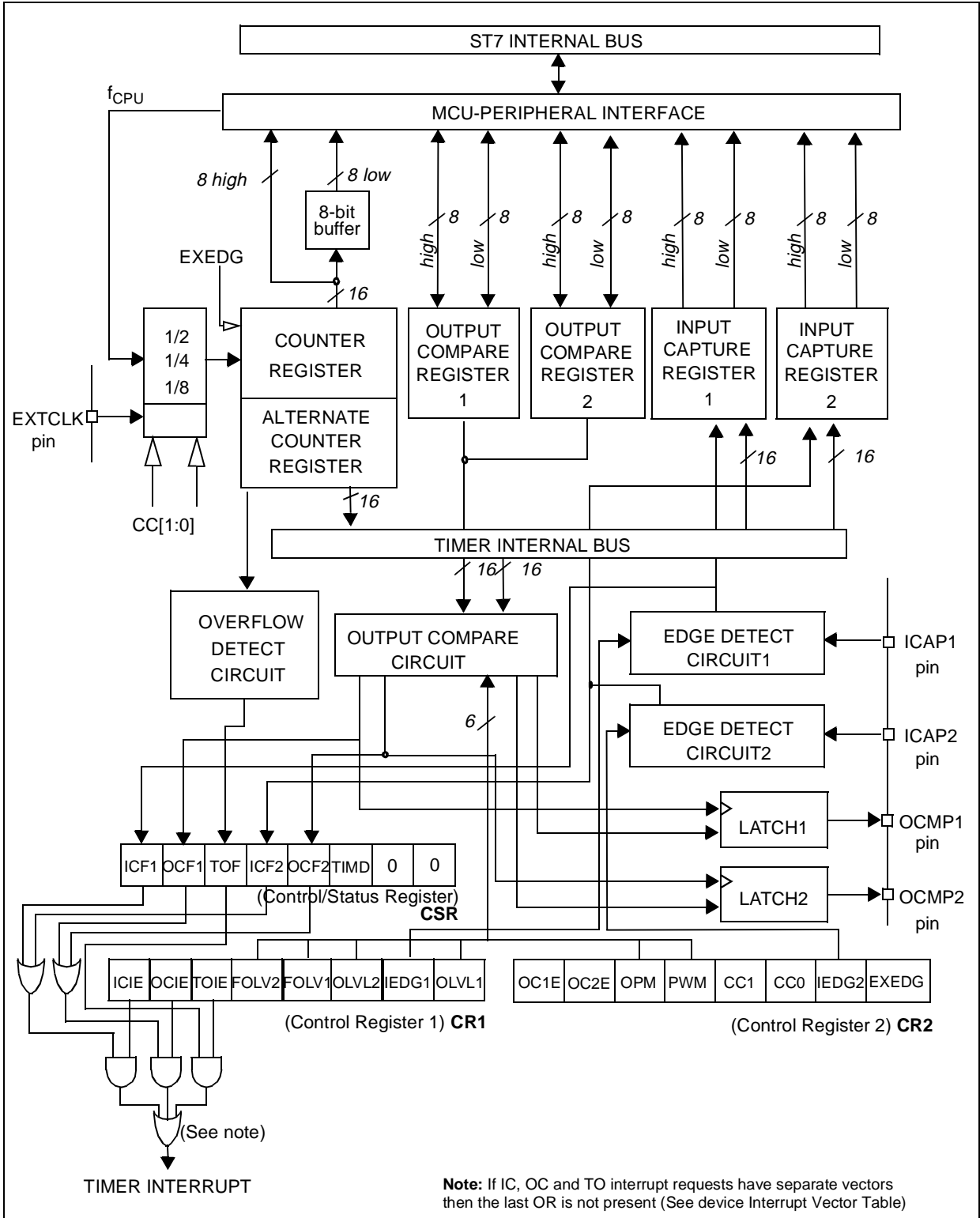
These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR), (see note at the end of paragraph titled 16-bit read sequence).

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

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

16-BIT TIMER (Cont'd)

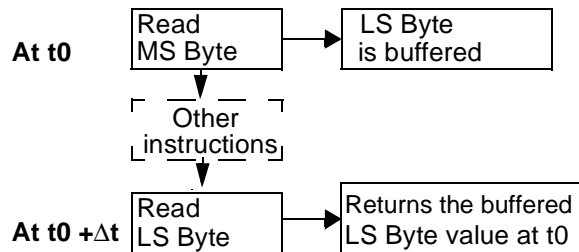
Figure 42. Timer Block Diagram



**16-BIT TIMER** (Cont'd)

**16-bit read sequence:** (from either the Counter Register or the Alternate Counter Register).

*Beginning of the sequence*



*Sequence completed*

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).

**10.4.3.2 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 43. Counter Timing Diagram, internal clock divided by 2

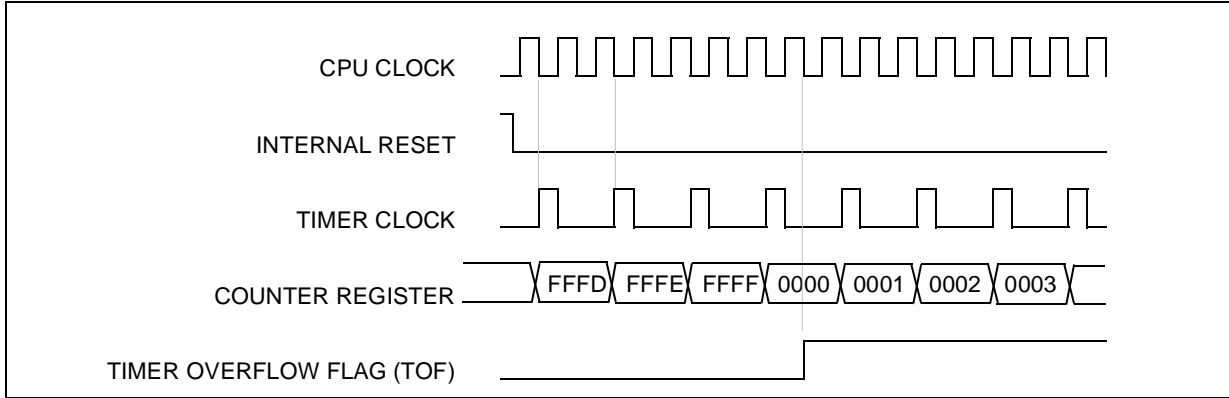


Figure 44. Counter Timing Diagram, internal clock divided by 4

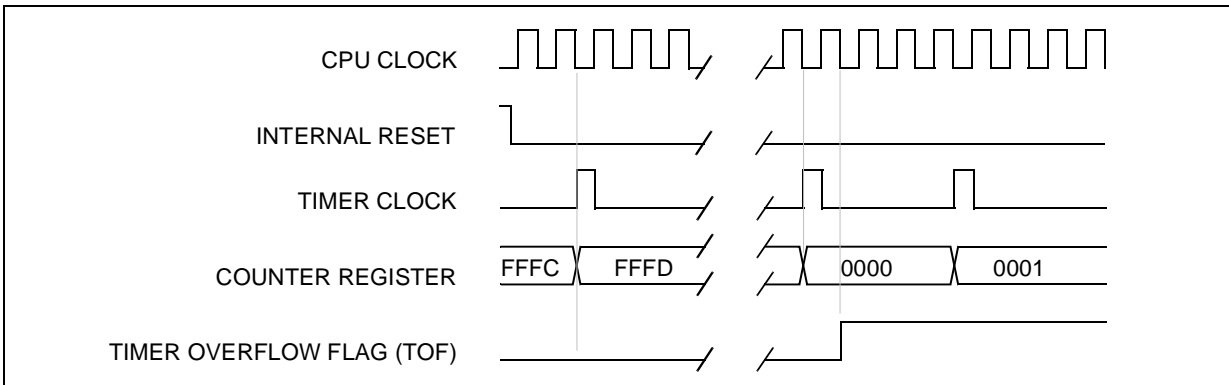
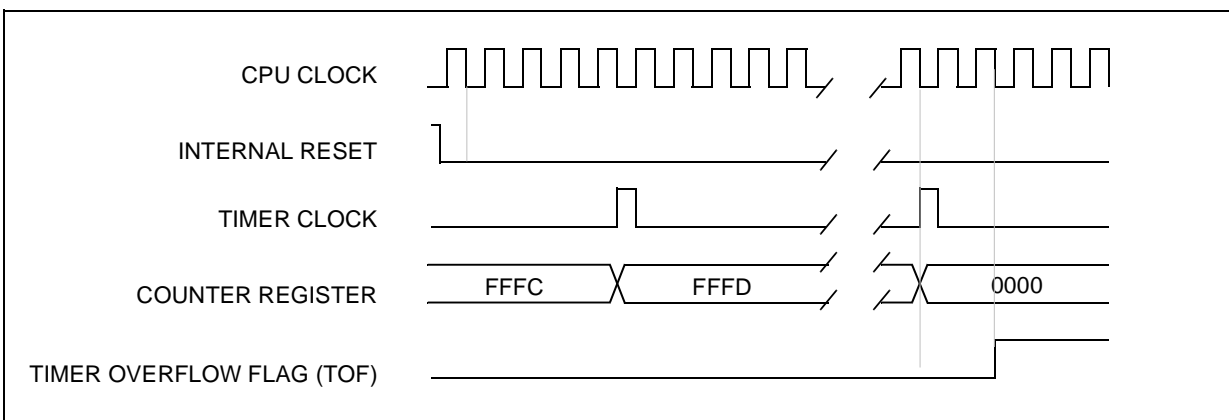


Figure 45. Counter Timing Diagram, internal clock divided by 8



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

**16-BIT TIMER (Cont'd)**

**10.4.3.3 Input Capture**

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

The two 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 5).

|               |                |                |
|---------------|----------------|----------------|
|               | MS Byte        | LS Byte        |
| IC <i>i</i> R | IC <i>i</i> HR | IC <i>i</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<sub>CPU</sub>/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 17 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 ICAP1 pin 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 47).
- A timer interrupt is generated if the ICIE bit is set and the I bit is cleared in the CC register. Otherwise, the interrupt remains pending until both conditions become true.

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

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

**Notes:**

1. After reading the IC*i*HR register, transfer of input capture data is inhibited and ICF*i* will never be set until the IC*i*LR register is also read.
2. The IC*i*R register contains the free running counter value which corresponds to the most recent input capture.
3. The 2 input capture functions can be used together even if the timer also uses the 2 output compare functions.
4. In One pulse Mode and PWM mode only Input Capture 2 can be used.
5. The alternate inputs (ICAP1 & 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 46. Input Capture Block Diagram

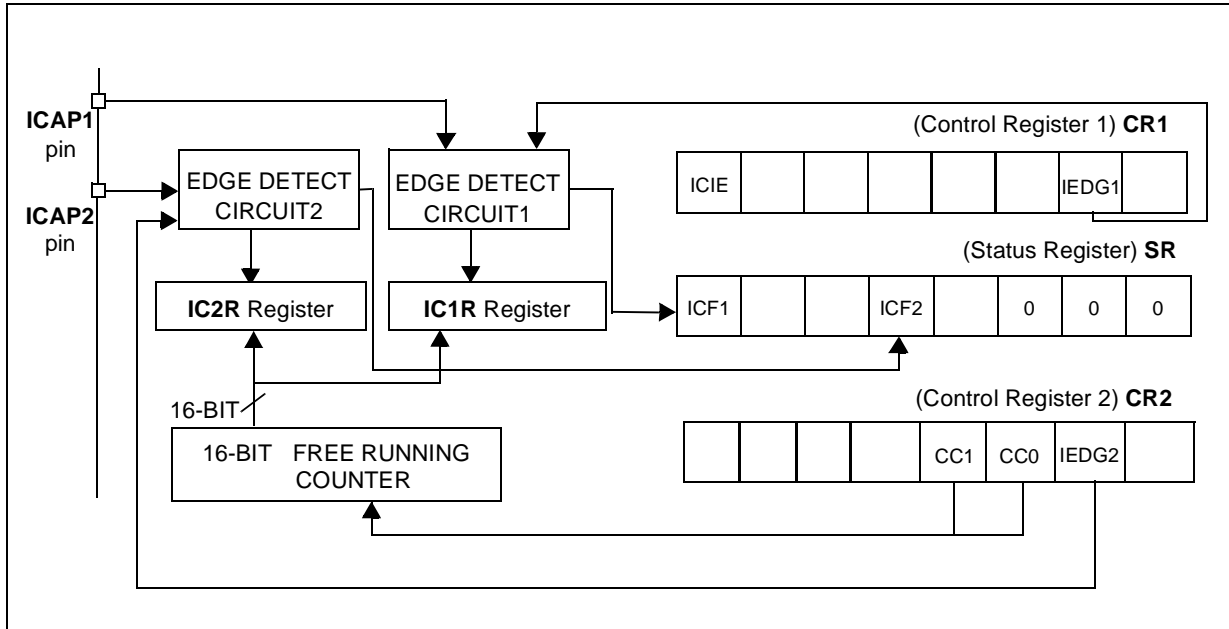
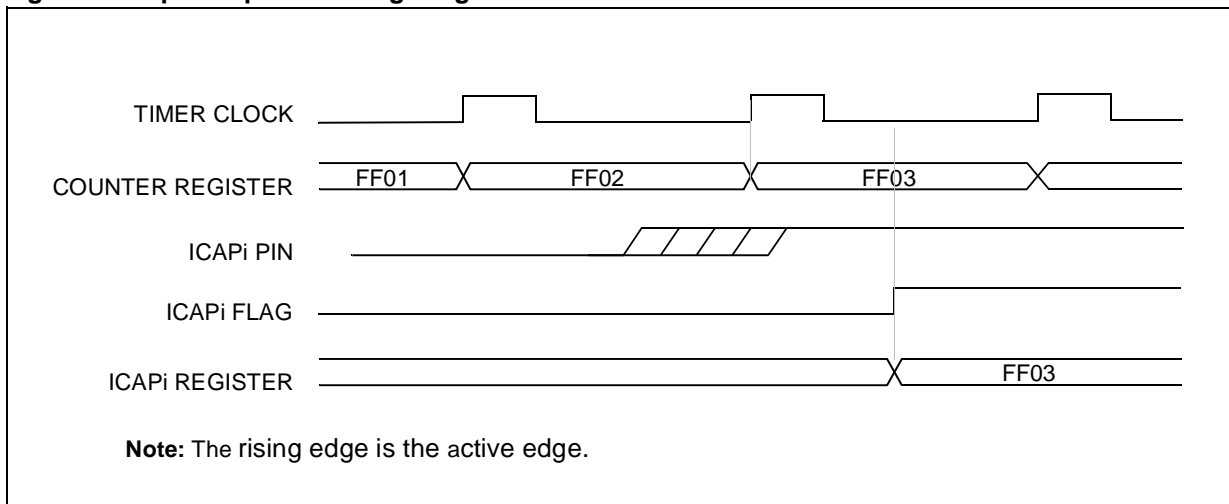


Figure 47. Input Capture Timing Diagram



**16-BIT TIMER (Cont'd)**

**10.4.3.4 Output Compare**

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

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

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

- Assigns pins with a programmable value if the OCiE 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 |
| OCiR | OCiHR   | OCiLR   |

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OCiR 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 OCiE bit if an output is needed then the OCMPi pin is dedicated to the output compare *i* signal.
- Select the timer clock (CC[1:0]) (see Table 17 Clock Control Bits).

And select the following in the CR1 register:

- Select the OLVLi/bit to applied to the OCMPi pins after the match occurs.
- Set the OCIE bit to generate an interrupt if it is needed.

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

- OCFi bit is set.

- The OCMPi pin takes OLVLi bit value (OCMPi 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 OCiR register value required for a specific timing application can be calculated using the following formula:

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

Where:

- $\Delta t$  = 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 17 Clock Control Bits)

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

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

Where:

- $\Delta t$  = Output compare period (in seconds)
- $f_{EXT}$  = External timer clock frequency (in hertz)

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

1. Reading the SR register while the OCFi bit is set.
  2. An access (read or write) to the OCiLR register.
- The following procedure is recommended to prevent the OCFi bit from being set between the time it is read and the write to the OCiR register:
- Write to the OCiHR register (further compares are inhibited).
  - Read the SR register (first step of the clearance of the OCFi bit, which may be already set).
  - Write to the OCiLR register (enables the output compare function and clears the OCFi bit).

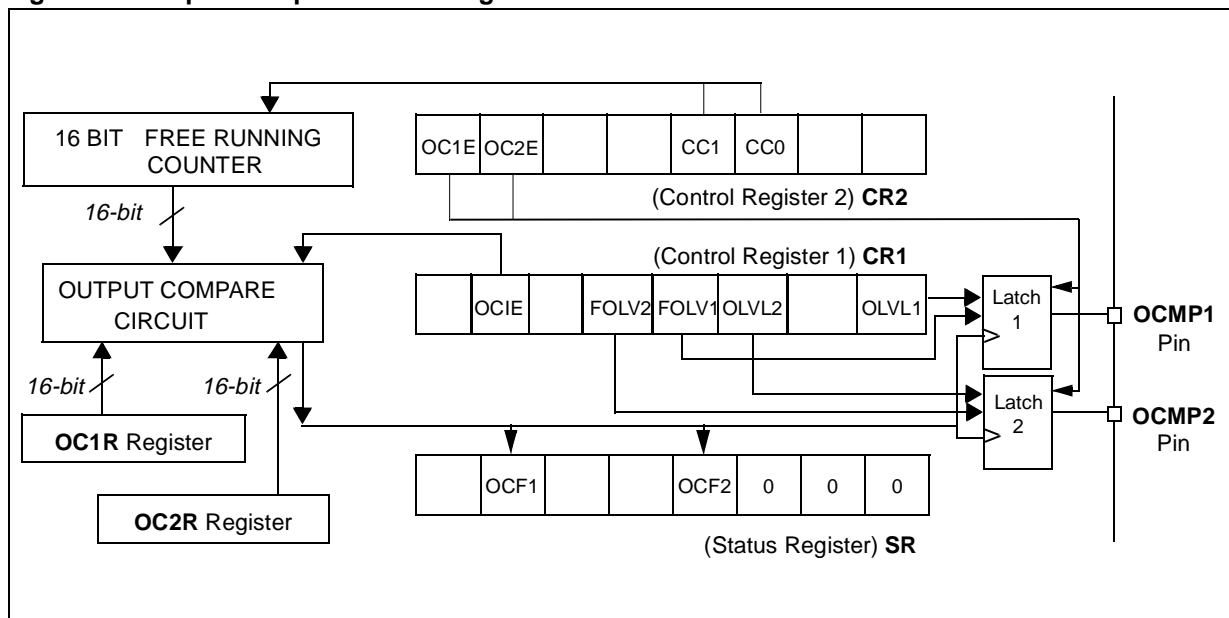
**16-BIT TIMER (Cont'd)****Notes:**

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 OLV $i$  bit will not appear when a match is found but an interrupt could be generated if the OC $i$ E bit is set.
3. When the timer clock is  $f_{\text{CPU}}/2$ , OCF $i$  and OCMP $i$  are set while the counter value equals the OC $i$ R register value (see Figure 49 on page 78). This behaviour is the same in OPM or PWM mode.  
When the timer clock is  $f_{\text{CPU}}/4$ ,  $f_{\text{CPU}}/8$  or in external clock mode, OCF $i$  and OCMP $i$  are set while the counter value equals the OC $i$ R register value plus 1 (see Figure 50 on page 78).
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 OLV $i$  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 FOLV $i$  bits have no effect in both one pulse mode and PWM mode.

**Figure 48. Output Compare Block Diagram**

16-BIT TIMER (Cont'd)

Figure 49. Output Compare Timing Diagram,  $f_{TIMER} = f_{CPU}/2$

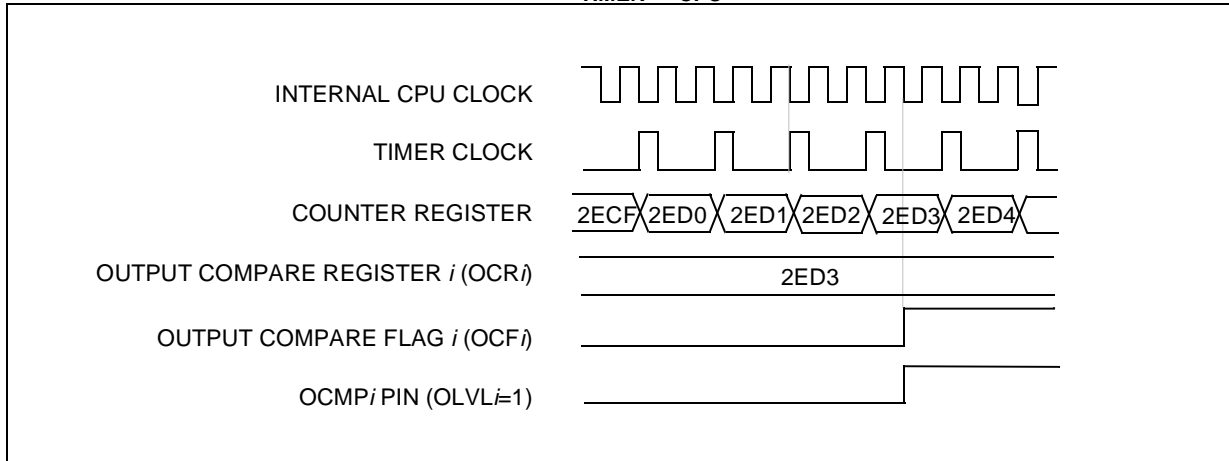
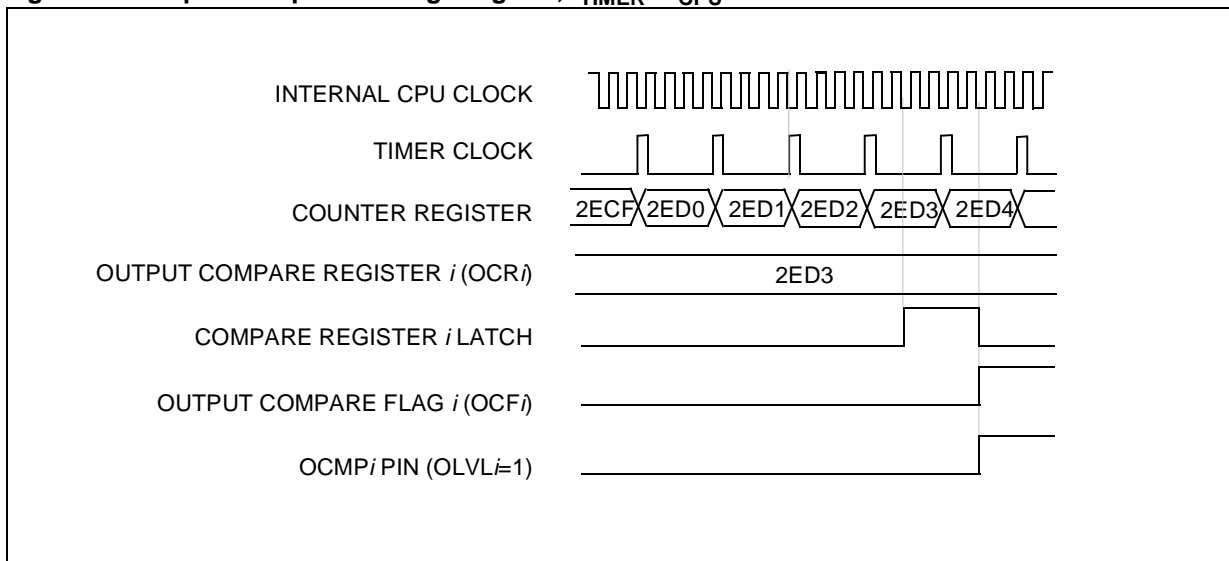


Figure 50. Output Compare Timing Diagram,  $f_{TIMER} = f_{CPU}/4$



## 16-BIT TIMER (Cont'd)

### 10.4.3.5 One Pulse Mode

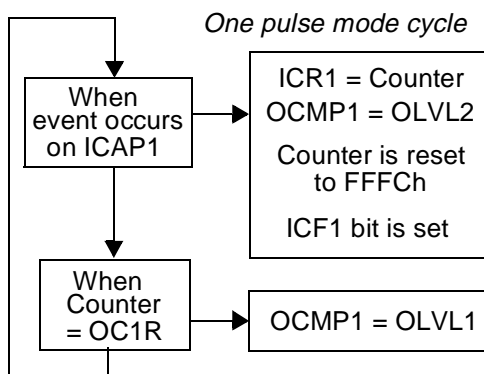
One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

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 17 Clock Control Bits).



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 ICR1 register.

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

Clearing the Input Capture interrupt request (i.e. clearing the ICF1 bit) is done in two steps:

1. Reading the SR register while the ICF1 bit is set.
2. An access (read or write) to the IC1R register.

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

$$\text{OC1R Value} = \frac{t * f_{\text{CPU}}}{\text{PRESC}} - 5$$

Where:

$t$  = Pulse period (in seconds)

$f_{\text{CPU}}$  = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on the CC[1:0] bits, see Table 17 Clock Control Bits)

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

$$\text{OC1R} = t * f_{\text{EXT}} - 5$$

Where:

$t$  = Pulse period (in seconds)

$f_{\text{EXT}}$  = 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 51).

#### 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 generate 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 51. One Pulse Mode Timing Example

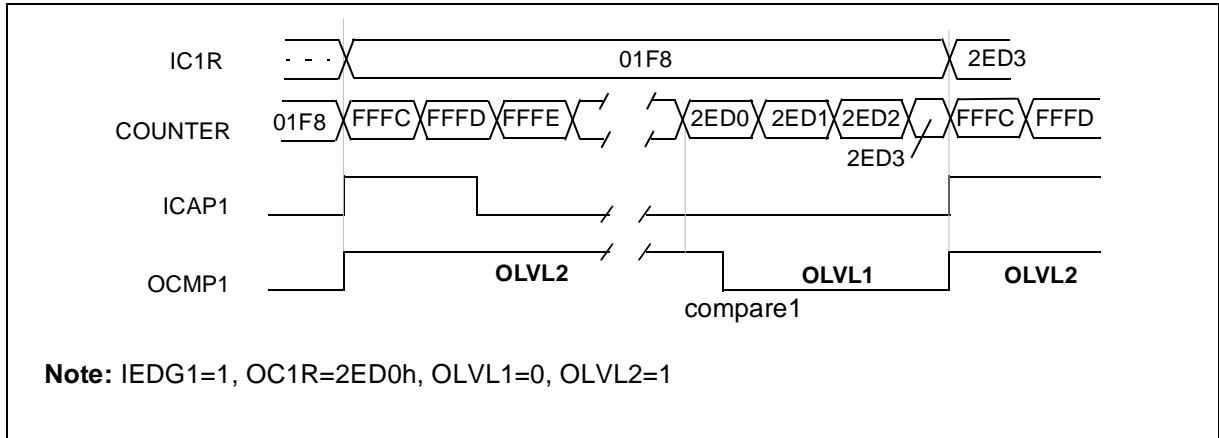
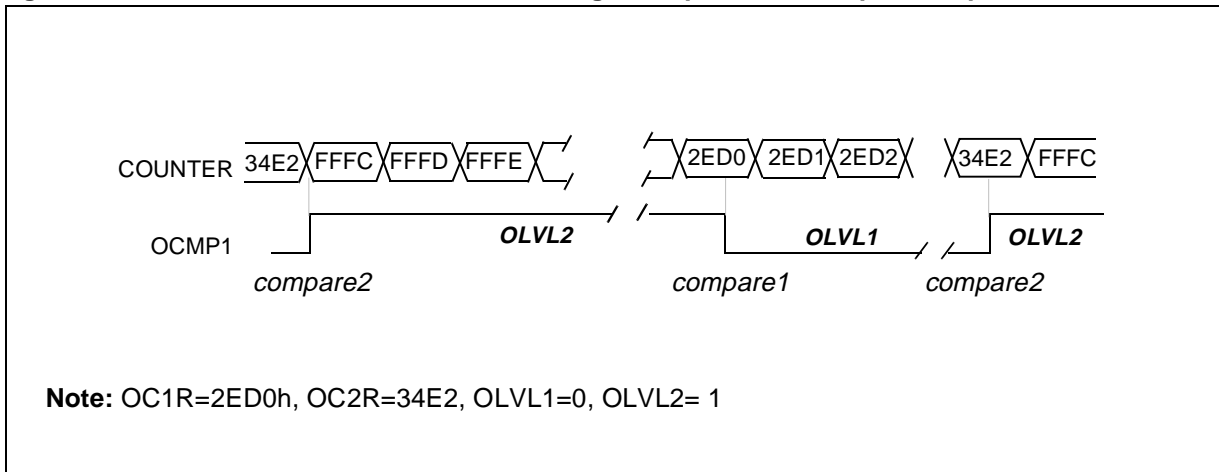


Figure 52. Pulse Width Modulation Mode Timing Example with 2 Output Compare Functions



**Note:** On timers with only 1 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)****10.4.3.6 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.

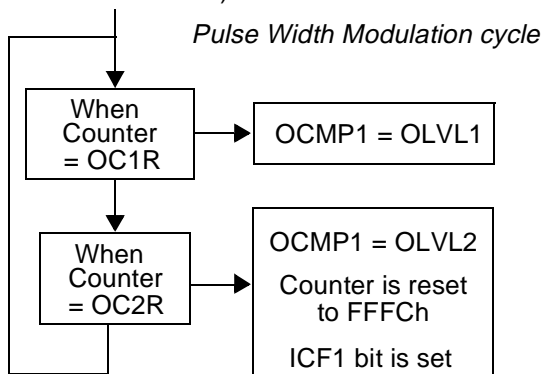
Pulse Width Modulation mode uses the complete Output Compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

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 17 Clock Control Bits).



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

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

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

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

Where:

$t$  = Signal or pulse period (in seconds)

$f_{\text{CPU}}$  = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 17 Clock Control Bits)

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

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

Where:

$t$  = Signal or pulse period (in seconds)

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

The Output Compare 2 event causes the counter to be initialized to FFFCh (See Figure 52)

**Notes:**

1. After a write instruction to the OC*R*HR register, the output compare function is inhibited until the OC*R*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)**

**10.4.4 Low Power Modes**

| Mode | Description  |
|------|--|
| WAIT | No effect on 16-bit Timer.<br>Timer interrupts cause the device to exit from WAIT mode.  |
| HALT | 16-bit Timer registers are frozen.<br>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.<br>If an input capture event occurs on the ICAP <i>i</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>i</i> bit is set, and the counter value present when exiting from HALT mode is captured into the IC <i>R</i> register. |

**10.4.5 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       |                    | Yes            | No             |
| Output Compare 1 event (not available in PWM mode) | OCF1       | OCIE               | Yes            | No             |
| Output Compare 2 event (not available in PWM mode) | OCF2       |                    | Yes            | No             |
| Timer Overflow event                               | TOF        | TOIE               | Yes            | No             |

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

**10.4.6 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) | Yes             | Yes                           | Yes              | Yes                     |
| One Pulse Mode              | No              | Not Recommended <sup>1)</sup> | No               | Partially <sup>2)</sup> |
| PWM Mode                    | No              | Not Recommended <sup>3)</sup> | No               | No                      |

- 1) See note 4 in Section 10.4.3.5 One Pulse Mode
- 2) See note 5 in Section 10.4.3.5 One Pulse Mode
- 3) See note 4 in Section 10.4.3.6 Pulse Width Modulation Mode

**16-BIT TIMER** (Cont'd)**10.4.7 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 17. Clock Control Bits

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

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

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

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

**16-BIT TIMER** (Cont'd)**CONTROL/STATUS REGISTER (CSR)**

Read Only (except bit 2 R/W)

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).



**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.



**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).



**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.



**16-BIT TIMER (Cont'd)**

**OUTPUT COMPARE 2 HIGH REGISTER (OC2HR)**

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

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



**OUTPUT COMPARE 2 LOW REGISTER (OC2LR)**

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

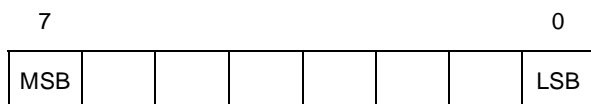
This is an 8-bit register that contains the low part of the value to be compared to the CLR register.



**COUNTER HIGH REGISTER (CHR)**

Read Only  
Reset Value: 1111 1111 (FFh)

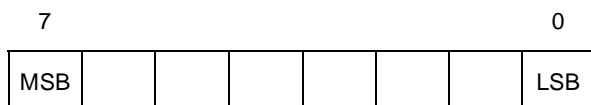
This is an 8-bit register that contains the high part of the counter value.



**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.



**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.



**ALTERNATE COUNTER LOW REGISTER (ACLR)**

Read Only  
Reset Value: 1111 1100 (FCh)

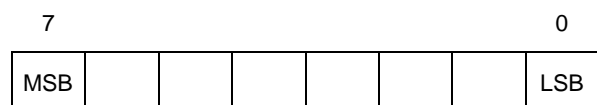
This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after an access to CSR register does not clear the TOF bit in the CSR register.



**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).



**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).



16-BIT TIMER (Cont'd)

Table 18. 16-Bit Timer Register Map and Reset Values

| Address (Hex.)             | Register Label              | 7         | 6         | 5         | 4          | 3          | 2          | 1          | 0          |
|----------------------------|-----------------------------|-----------|-----------|-----------|------------|------------|------------|------------|------------|
| Timer A: 32<br>Timer B: 42 | <b>CR1</b><br>Reset Value   | ICIE<br>0 | OCIE<br>0 | TOIE<br>0 | FOLV2<br>0 | FOLV1<br>0 | OLVL2<br>0 | IEDG1<br>0 | OLVL1<br>0 |
| Timer A: 31<br>Timer B: 41 | <b>CR2</b><br>Reset Value   | OC1E<br>0 | OC2E<br>0 | OPM<br>0  | PWM<br>0   | CC1<br>0   | CC0<br>0   | IEDG2<br>0 | EXEDG<br>0 |
| Timer A: 33<br>Timer B: 43 | <b>CSR</b><br>Reset Value   | ICF1<br>x | OCF1<br>x | TOF<br>x  | ICF2<br>x  | OCF2<br>x  | TIMD<br>0  | -<br>x     | -<br>x     |
| Timer A: 34<br>Timer B: 44 | <b>IC1HR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 35<br>Timer B: 45 | <b>IC1LR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 36<br>Timer B: 46 | <b>OC1HR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 37<br>Timer B: 47 | <b>OC1LR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 3E<br>Timer B: 4E | <b>OC2HR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 3F<br>Timer B: 4F | <b>OC2LR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 38<br>Timer B: 48 | <b>CHR</b><br>Reset Value   | MSB<br>1  | 1         | 1         | 1          | 1          | 1          | 1          | LSB<br>1   |
| Timer A: 39<br>Timer B: 49 | <b>CLR</b><br>Reset Value   | MSB<br>1  | 1         | 1         | 1          | 1          | 1          | 0          | LSB<br>0   |
| Timer A: 3A<br>Timer B: 4A | <b>ACHR</b><br>Reset Value  | MSB<br>1  | 1         | 1         | 1          | 1          | 1          | 1          | LSB<br>1   |
| Timer A: 3B<br>Timer B: 4B | <b>ACLHR</b><br>Reset Value | MSB<br>1  | 1         | 1         | 1          | 1          | 1          | 0          | LSB<br>0   |
| Timer A: 3C<br>Timer B: 4C | <b>IC2HR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |
| Timer A: 3D<br>Timer B: 4D | <b>IC2LR</b><br>Reset Value | MSB<br>-  | -         | -         | -          | -          | -          | -          | LSB<br>-   |

<sup>1</sup>These bits are not used in Timer A and must be kept cleared.



## 10.5 SERIAL PERIPHERAL INTERFACE (SPI)

### 10.5.1 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.

### 10.5.2 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
- SS Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

### 10.5.3 General Description

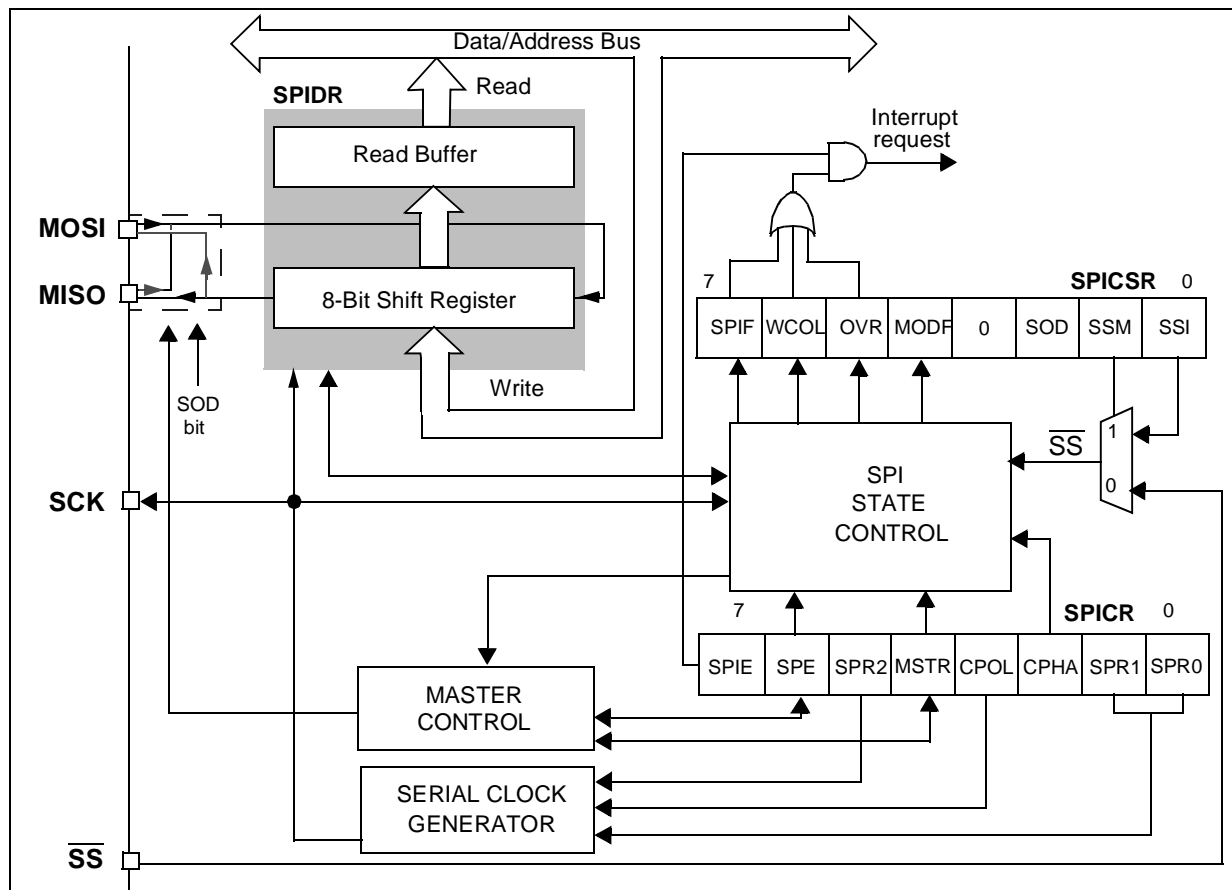
Figure 53 shows the serial peripheral interface (SPI) block diagram. There are 3 registers:

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

The SPI is connected to external devices through 3 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
- $\overline{SS}$ : 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 SS inputs can be driven by standard I/O ports on the master MCU.

Figure 53. Serial Peripheral Interface Block Diagram



**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**10.5.3.1 Functional Description**

A basic example of interconnections between a single master and a single slave is illustrated in Figure 54.

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).

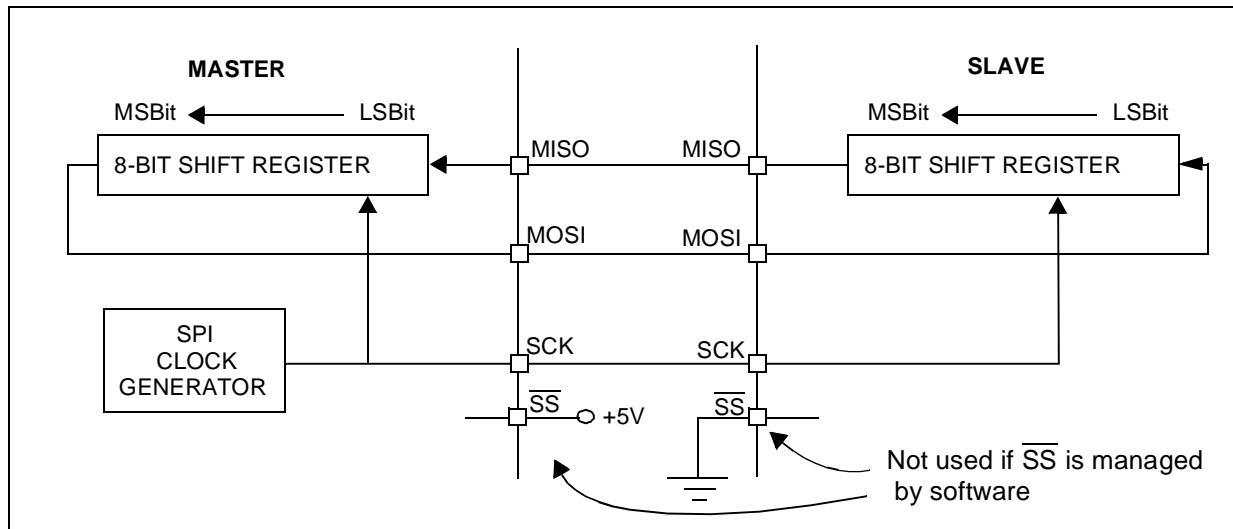
The communication is always initiated by the master. When the master device transmits data to a slave device via MOSI pin, the slave device re-

sponds 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 57) but master and slave must be programmed with the same timing mode.

**Figure 54. Single Master/ Single Slave Application**



**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**10.5.3.2 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 56)

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 55):

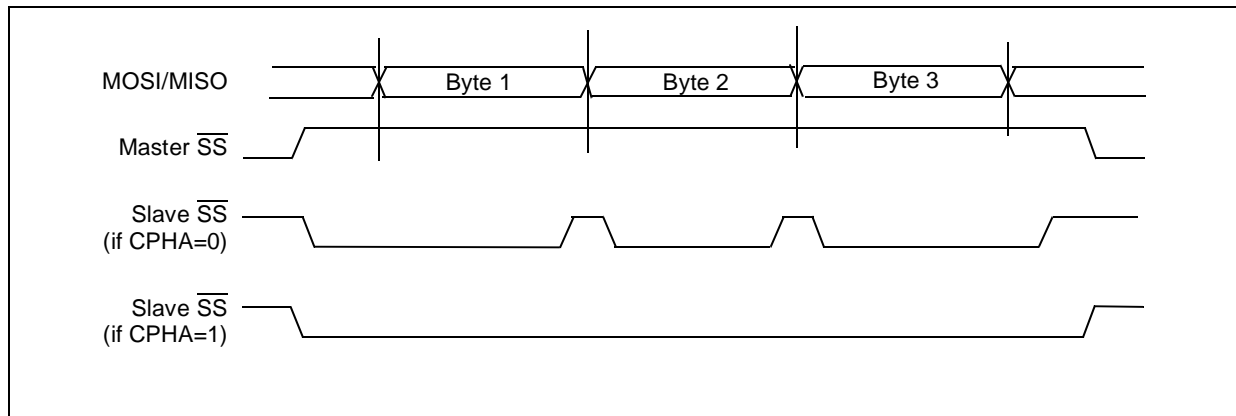
If CPHA=1 (data latched on 2nd clock edge):

- $\overline{SS}$  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  $\overline{SS}$  function by software (SSM= 1 and SSI=0 in the in the SPICSR register)

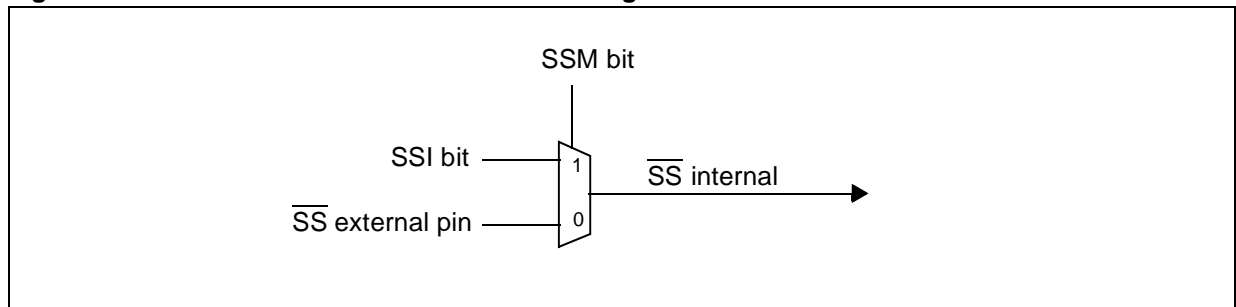
If CPHA=0 (data latched on 1st clock edge):

- $\overline{SS}$  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 10.5.5.3).

**Figure 55. Generic  $\overline{SS}$  Timing Diagram**



**Figure 56. Hardware/Software Slave Select Management**



## SERIAL PERIPHERAL INTERFACE (Cont'd)

### 10.5.3.3 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 two steps in order (if the SPICSR register is not written first, the SPICR register setting may be not taken into account):

- Write to the SPICSR 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 57 shows the four possible configurations.
 

**Note:** The slave must have the same CPOL and CPHA settings as the master.
  - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the SS pin high for the complete byte transmit sequence.
- Write to the SPICR register:
  - Set the MSTR and SPE bits
 

**Note:** MSTR and SPE bits remain set only if SS is high).

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

### 10.5.3.4 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:

- An access to the SPICSR register while the SPIF bit is set
- 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.

### 10.5.3.5 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:

- 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 57).
 

**Note:** The slave must have the same CPOL and CPHA settings as the master.
  - Manage the SS pin as described in Section 10.5.3.2 and Figure 55. If CPHA=1 SS must be held low continuously. If CPHA=0 SS must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
- Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

### 10.5.3.6 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:

- An access to the SPICSR register while the SPIF bit is set.
- 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 10.5.5.2).

## SERIAL PERIPHERAL INTERFACE (Cont'd)

### 10.5.4 Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 57).

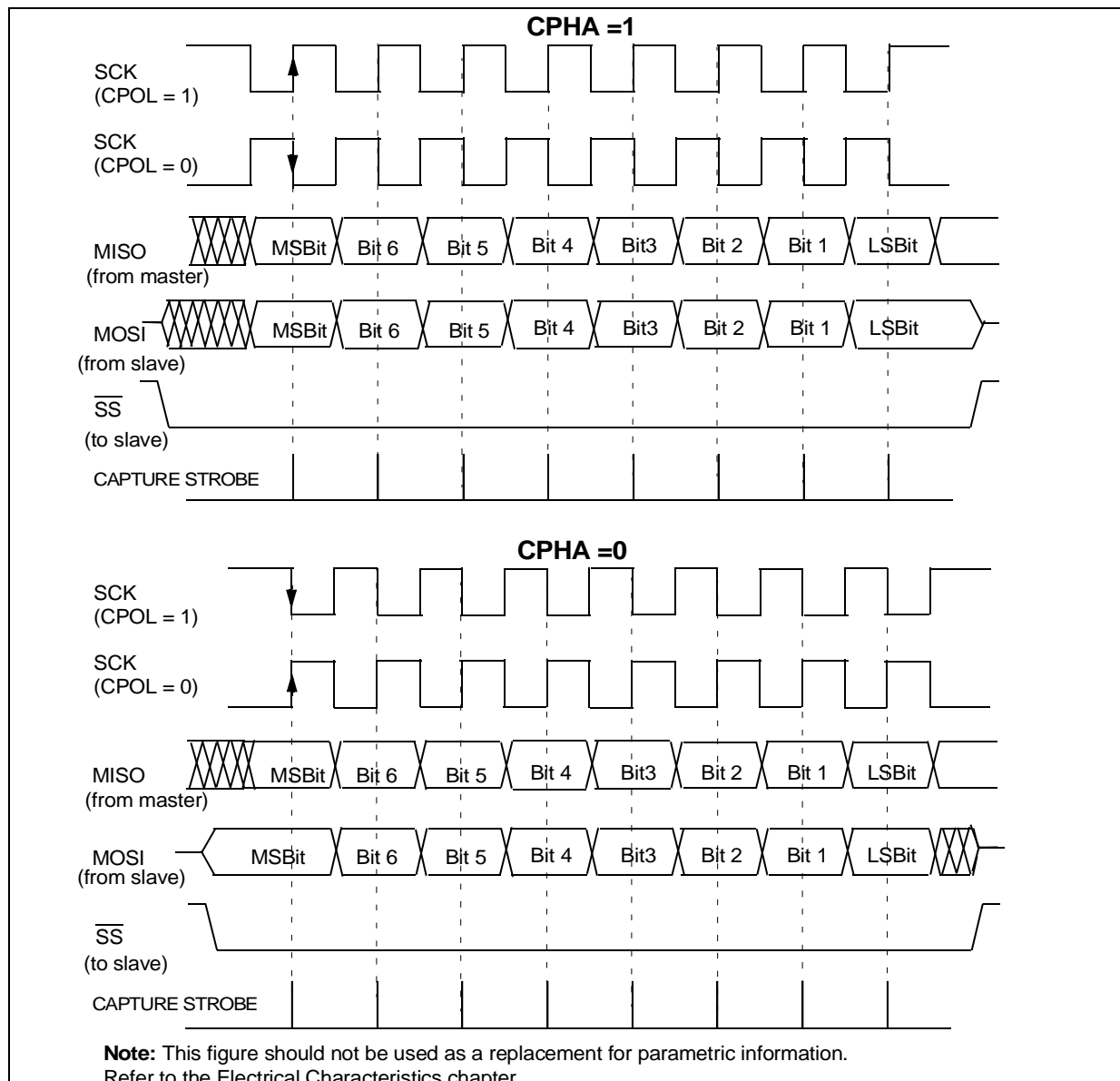
**Note:** The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL=1 or pulling down SCK if CPOL=0).

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

Figure 57, 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 57. Data Clock Timing Diagram**



**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**10.5.5 Error Flags**

**10.5.5.1 Master Mode Fault (MODF)**

Master mode fault occurs when the master device has its 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 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.

**10.5.5.2 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.

**10.5.5.3 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 10.5.3.2 Slave Select Management.

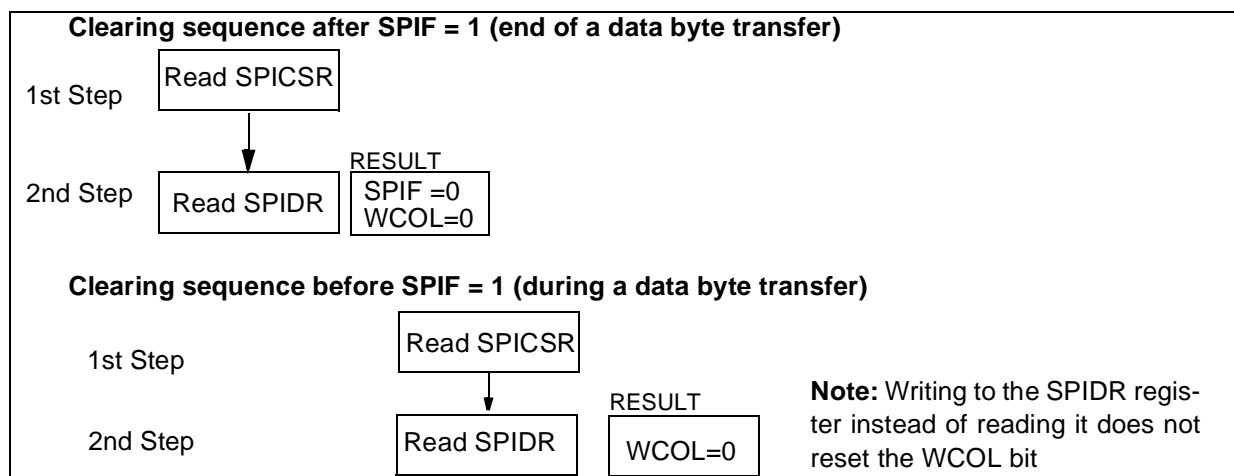
**Note:** a "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the MCU operation.

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 58).

**Figure 58. Clearing the WCOL bit (Write Collision Flag) Software Sequence**



## SERIAL PERIPHERAL INTERFACE (Cont'd)

## 10.5.5.4 Single Master Systems

A typical single master system may be configured, using an MCU as the master and four MCUs as slaves (see Figure 59).

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.

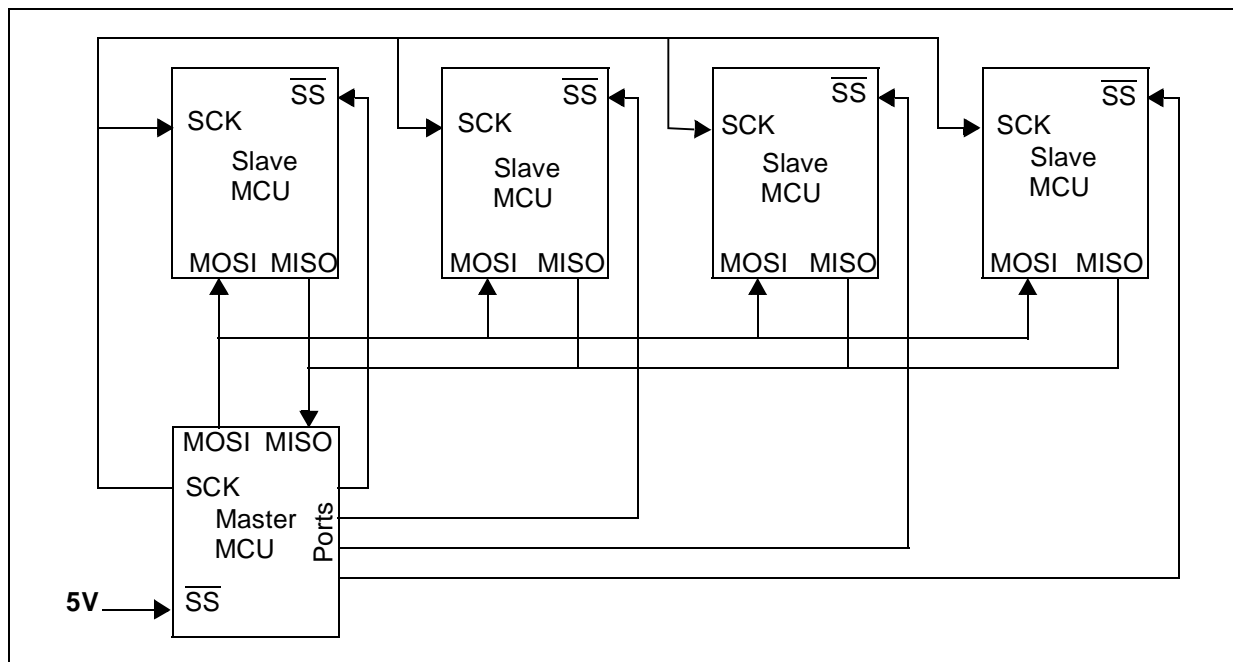
The  $\overline{SS}$  pins are pulled high during reset since the master device ports will be forced to be inputs at that time, thus disabling the slave devices.

**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 59. Single Master / Multiple Slave Configuration



**SERIAL PERIPHERAL INTERFACE (Cont'd)**

**10.5.6 Low Power Modes**

| Mode | Description  |
|------|--|
| WAIT | No effect on SPI.<br>SPI interrupt events cause the device to exit from WAIT mode.   |
| HALT | SPI registers are frozen.<br>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. |

**10.5.6.1 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 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 10.5.3.2), make sure the master drives a low level on the SS pin when the slave enters Halt mode.

**10.5.7 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)****10.5.8 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 10.5.5.1 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 19 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 10.5.5.1 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 re-setting 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 19. 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 58).

0: No write collision occurred

1: A write collision has been detected

Bit 5 = **OVR** *SPI 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 10.5.5.2). An interrupt is generated if SPIE = 1 in SPICSR 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 10.5.5.1 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 SPICSR 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 10.5.3.2 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 53).

## SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 20. SPI Register Map and Reset Values

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

**10.6 SERIAL COMMUNICATIONS INTERFACE (SCI)****10.6.1 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.

**10.6.2 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 500K baud.
- 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

**10.6.3 General Description**

The interface is externally connected to another device by two pins (see Figure 61):

- 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:

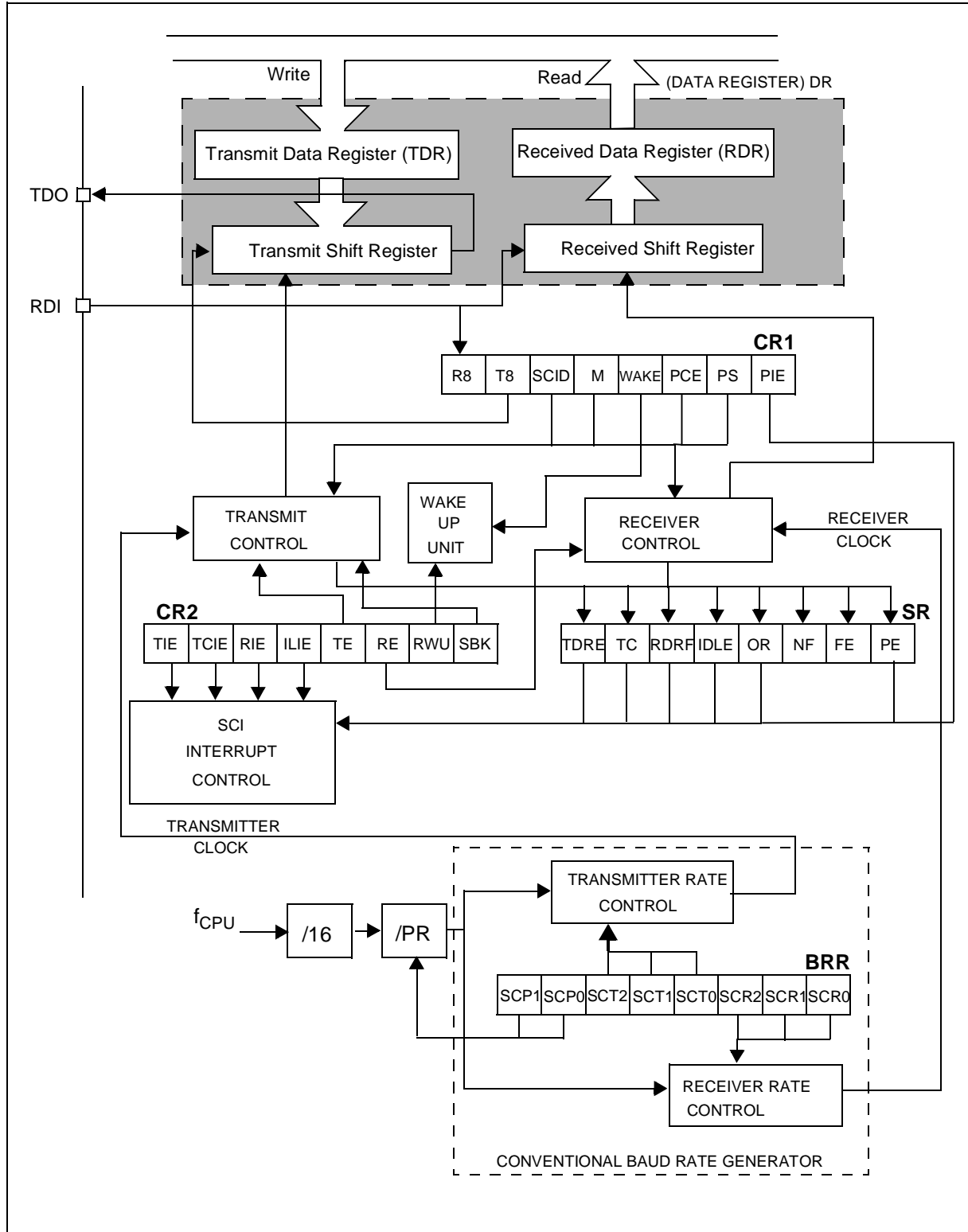
- An Idle Line prior to transmission or reception
- A start bit
- A data word (8 or 9 bits) least significant bit first
- A Stop bit indicating that the frame is complete.

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 60. SCI Block Diagram



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**10.6.4 Functional Description**

The block diagram of the Serial Control Interface, is shown in Figure 60. It contains 6 dedicated registers:

- Two control registers (SCICR1 & SCICR2)
- A status register (SCISR)
- A baud rate register (SCIBRR)
- An extended prescaler receiver register (SCIERR)
- An extended prescaler transmitter register (SCIETPR)

Refer to the register descriptions in Section 10.6.7 for the definitions of each bit.

**10.6.4.1 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 60).

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

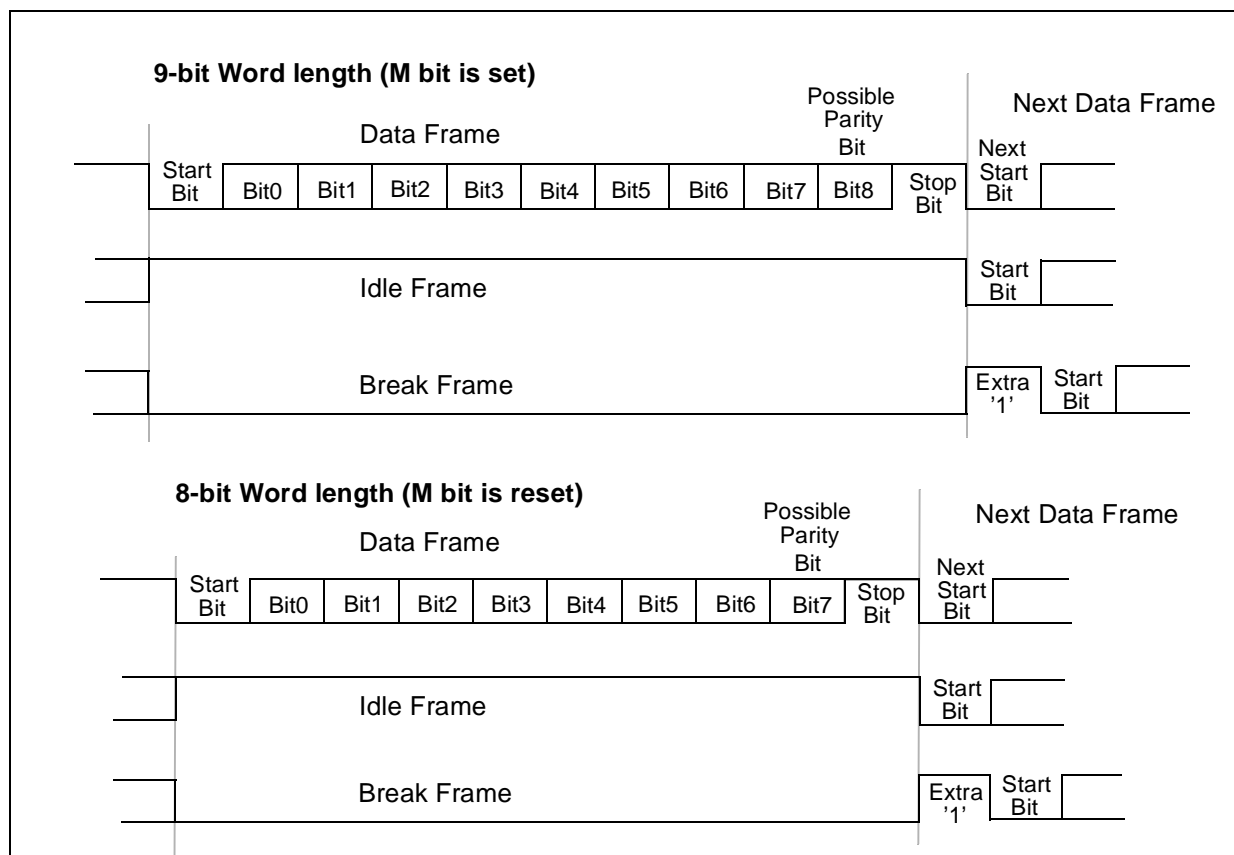
The TDO pin is in high state during the stop bit.

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 61. Word Length Programming**



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****10.6.4.2 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 60).

**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 or after the break frame) the TC bit is set and an interrupt is generated if the TCIE is set and the I bit is cleared in the CCR register.

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

1. An access to the 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 61).

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

**Idle Characters**

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

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

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

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****10.6.4.3 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 of a buffer (RDR) between the internal bus and the received shift register (see Figure 60).

**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 SPI handles it as a framing error.

**Idle Character**

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

**Overrun Error**

An overrun error occurs when a character is received when RDRF has not been reset. Data 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 will not be lost.
- The shift register will be overwritten.
- An interrupt is generated if the RIE bit is set and the I bit is cleared in the CCR register.

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

**Noise Error**

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

When noise is detected in a frame:

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

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

**Framing Error**

A framing error is detected when:

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

When the framing error is detected:

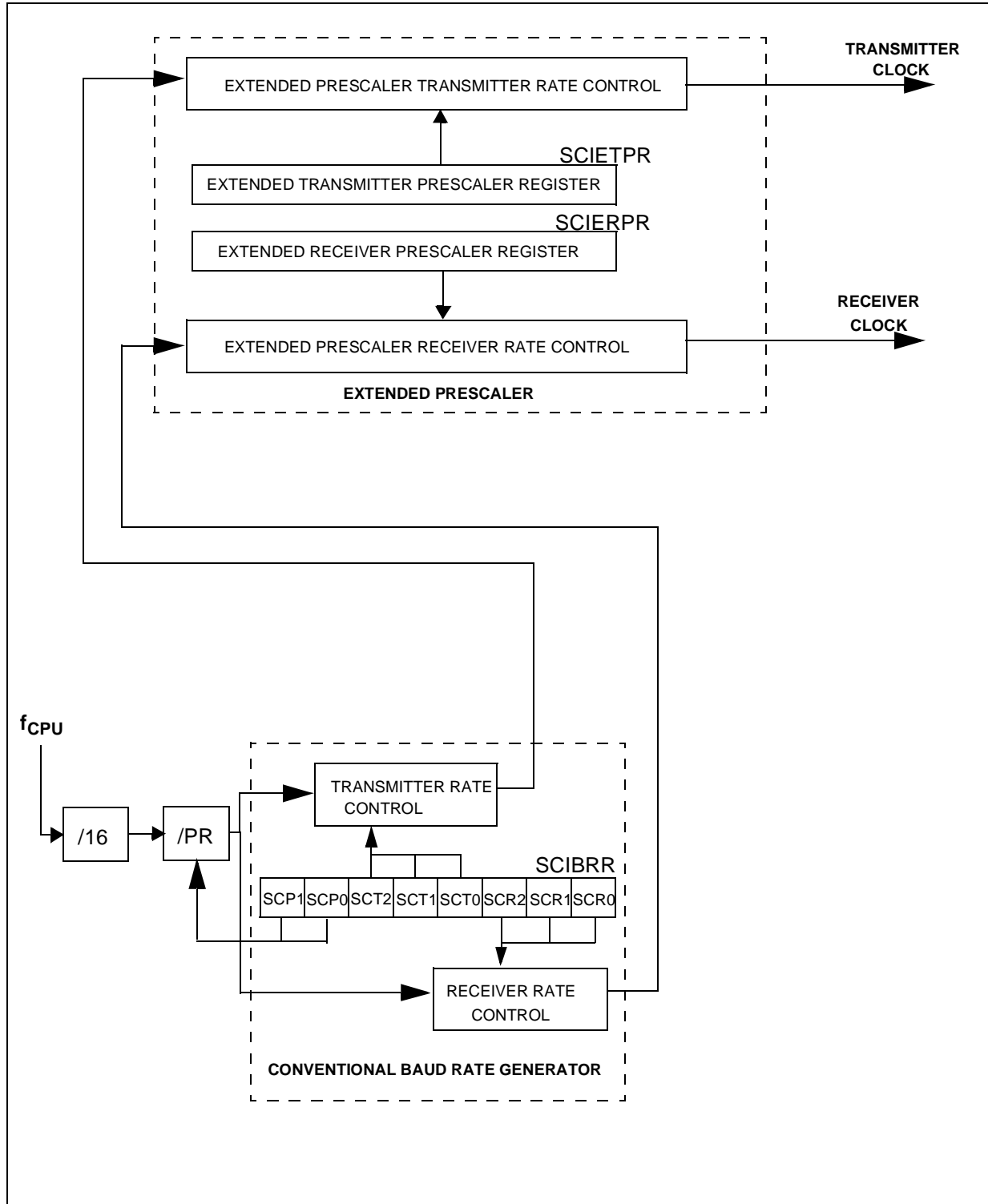
- the FE bit is set by hardware
- Data is transferred from the Shift register to the SCIDR register.
- No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

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



SERIAL COMMUNICATIONS INTERFACE (Cont'd)

Figure 62. SCI Baud Rate and Extended Prescaler Block Diagram



**SERIAL COMMUNICATIONS INTERFACE (Cont'd)****10.6.4.4 Conventional Baud Rate Generation**

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

$$T_x = \frac{f_{CPU}}{(16 \cdot PR) \cdot TR} \quad R_x = \frac{f_{CPU}}{(16 \cdot PR) \cdot RR}$$

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.

**10.6.4.5 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 62.

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

$$T_x = \frac{f_{CPU}}{16 \cdot ETPR \cdot (PR \cdot TR)} \quad R_x = \frac{f_{CPU}}{16 \cdot ERPR \cdot (PR \cdot RR)}$$

with:

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

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

**10.6.4.6 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 recognised an Idle Frame. Then the RWU bit is reset by hardware but the IDLE bit is not set.

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

**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 will be 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)****10.6.4.7 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 21.

**Table 21. Frame Formats**

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

**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.

Ex: data=00110101; 4 bits set => parity bit will be 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.

Ex: data=00110101; 4 bits set => parity bit will be 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.

**10.6.5 Low Power Modes**

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

**10.6.6 Interrupts**

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

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).

**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**10.6.7 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 will not be 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, a Preamble or a Break is complete. An interrupt is generated if TCIE=1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).  
 0: Transmission is not complete  
 1: Transmission is complete

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

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

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

**Note:** The IDLE bit will not be set again until the RDRF bit has been set itself (i.e. 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 will not be lost but the shift register will be overwritten.

**Bit 2 = NF** *Noise flag.*  
 This bit is set by hardware when noise is detected on a received frame. It is cleared by a software sequence (an access to the 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 will be 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 wakeup 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 will send 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 60).

The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 60).

**BAUD RATE REGISTER (SCIBRR)**

Read/Write

Reset Value: 00xx xxxx (xxh)

|      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|
| 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    |

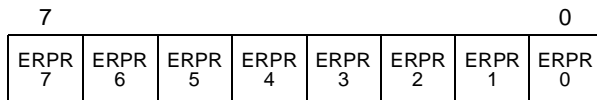
**SERIAL COMMUNICATIONS INTERFACE (Cont'd)**

**EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERP)**

Read/Write

Reset Value: 0000 0000 (00h)

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



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 62) is divided by the binary factor set in the SCIERP register (in the range 1 to 255).

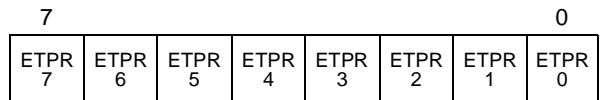
The extended baud rate generator is not used after a reset.

**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.



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 62) 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 22. Baudrate Selection**

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



## SERIAL COMMUNICATION INTERFACE (Cont'd)

Table 23. SCI Register Map and Reset Values

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

## 10.7 I<sup>2</sup>C BUS INTERFACE (I2C)

### 10.7.1 Introduction

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

### 10.7.2 Main Features

- Parallel-bus/I<sup>2</sup>C protocol converter
- Multi-master capability
- 7-bit/10-bit Addressing
- Transmitter/Receiver flag
- End-of-byte transmission flag
- Transfer problem detection

#### I<sup>2</sup>C Master Features:

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

#### I<sup>2</sup>C Slave Features:

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

### 10.7.3 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<sup>2</sup>C bus by a data pin (SDAI) and by a clock pin (SCLI). It can be connected both with a standard I<sup>2</sup>C bus and a Fast I<sup>2</sup>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

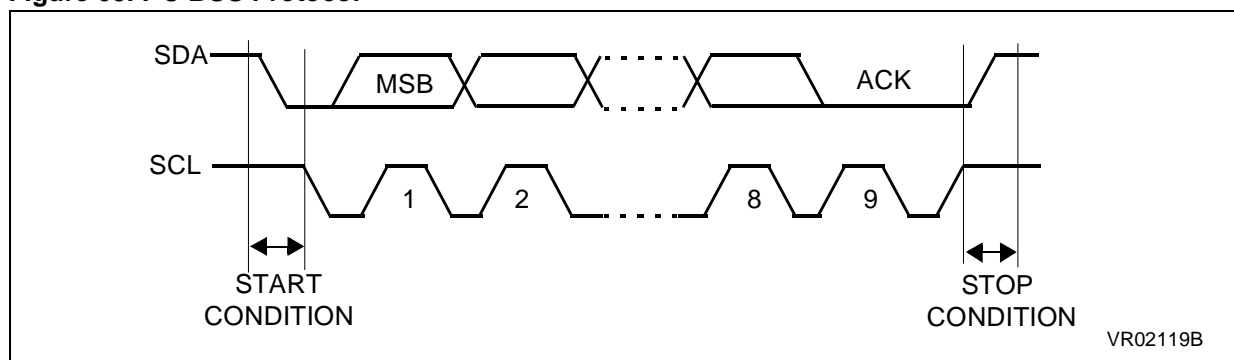
In Master mode, it initiates a data transfer and generates the clock signal. A serial data transfer always begins with a start condition and ends with a stop condition. Both start and stop conditions are generated in master mode by software.

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 63.

Figure 63. I<sup>2</sup>C BUS Protocol



## I<sup>2</sup>C BUS INTERFACE (Cont'd)

Acknowledge may be enabled and disabled by software.

The I<sup>2</sup>C interface address and/or general call address can be selected by software.

The speed of the I<sup>2</sup>C interface may be selected between Standard (0-100KHz) and Fast I<sup>2</sup>C (100-400KHz).

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

When the I<sup>2</sup>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.

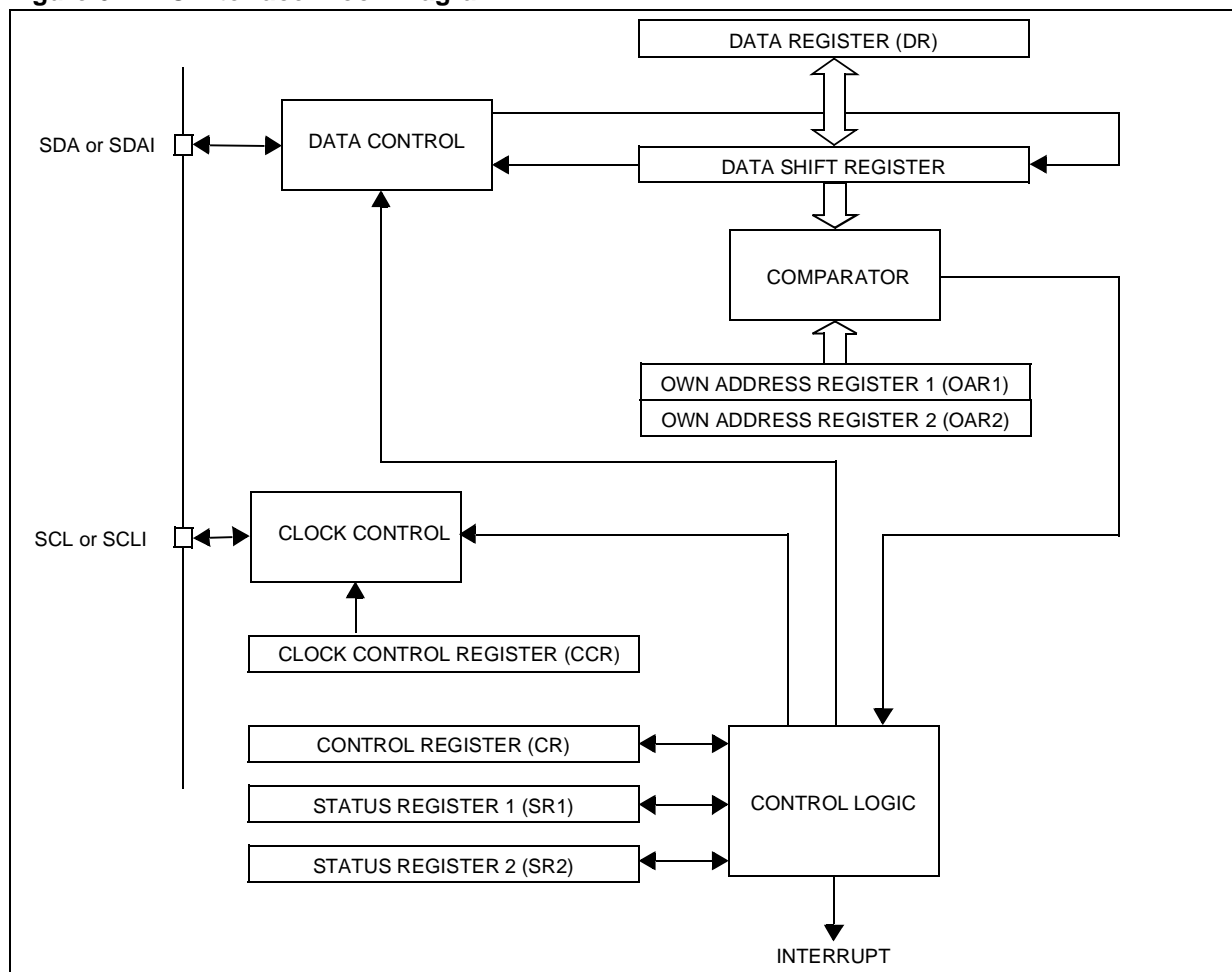
When the I<sup>2</sup>C cell is disabled, the SDA and SCL ports revert to being standard I/O port pins.

### 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.

**Figure 64. I<sup>2</sup>C Interface Block Diagram**



## I<sup>2</sup>C BUS INTERFACE (Cont'd)

### 10.7.4 Functional Description

Refer to the CR, SR1 and SR2 registers in Section 10.7.7. for the bit definitions.

By default the I<sup>2</sup>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.

#### 10.7.4.1 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 65 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 65 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 65 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 65 Transfer sequencing EV4).

#### Error Cases

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

If it is a Stop then the interface discards the data, released the lines and waits for another Start condition.

If it is a Start then the interface discards the data and waits for the next slave address on the bus.

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

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

## I<sup>2</sup>C BUS INTERFACE (Cont'd)

### How to release the SDA / SCL lines

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

### 10.7.4.2 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 65 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 65 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 65 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 65 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<sup>2</sup>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 65 Transfer sequencing EV8).

When the acknowledge bit is received, the interface sets:

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

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

#### Error Cases

- **BERR**: Detection of a Stop or a Start condition during a byte transfer. In this case, the EVF and

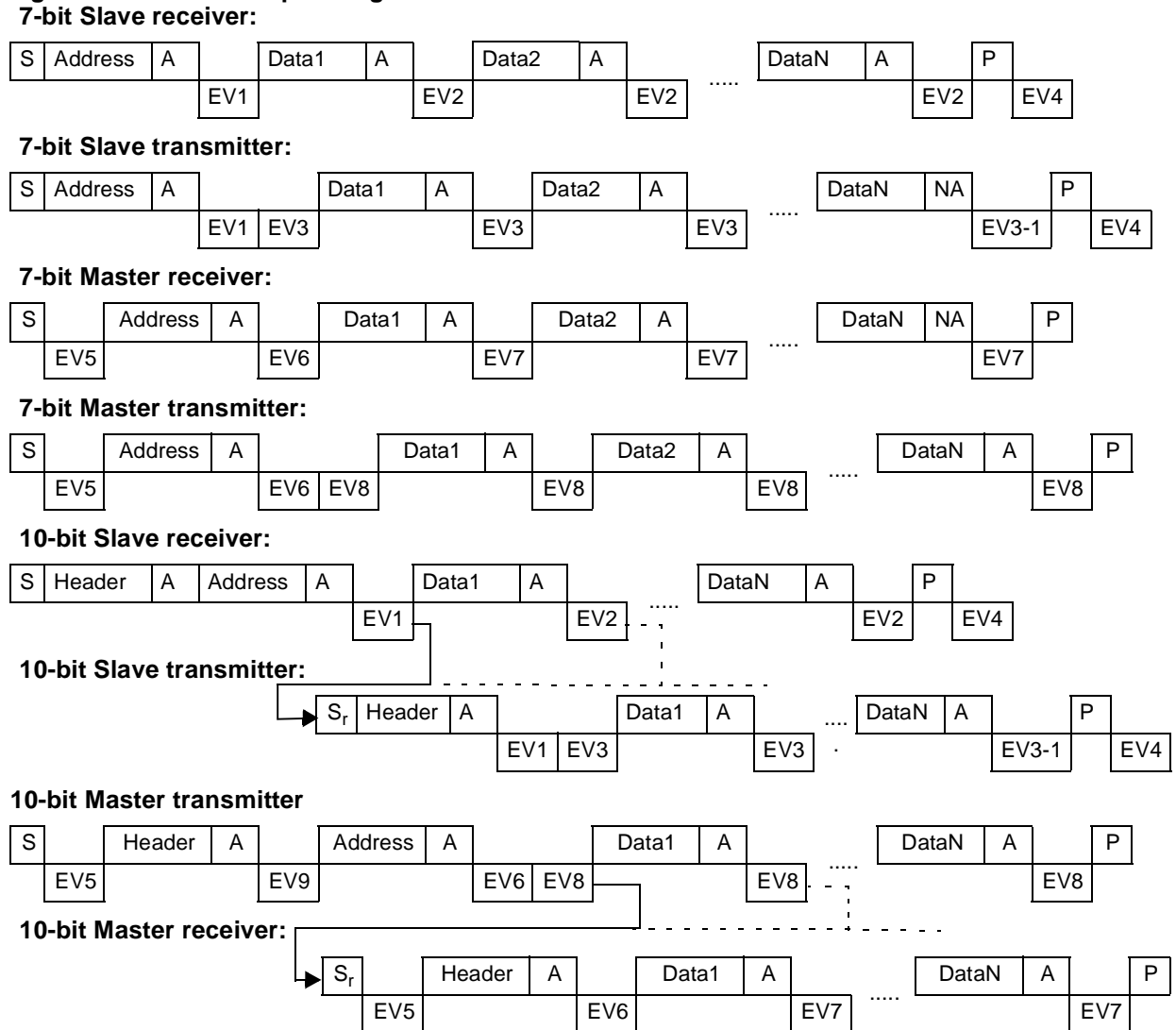
BERR bits are set by hardware with an interrupt if ITE is set.

- **AF**: Detection of a non-acknowledge bit. In this case, the EVF and AF bits are set by hardware with an interrupt if the ITE bit is set. To resume, set the START or STOP bit.
- **ARLO**: Detection of an arbitration lost condition. In this case the ARLO bit is set by hardware (with an interrupt if the ITE bit is set and the interface goes automatically back to slave mode (the M/SL bit is cleared).

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

I<sup>2</sup>C BUS INTERFACE (Cont'd)

Figure 65. Transfer Sequencing



**Legend:** S=Start, S<sub>r</sub> = Repeated Start, P=Stop, A=Acknowledge, NA=Non-acknowledge, EV<sub>x</sub>=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.

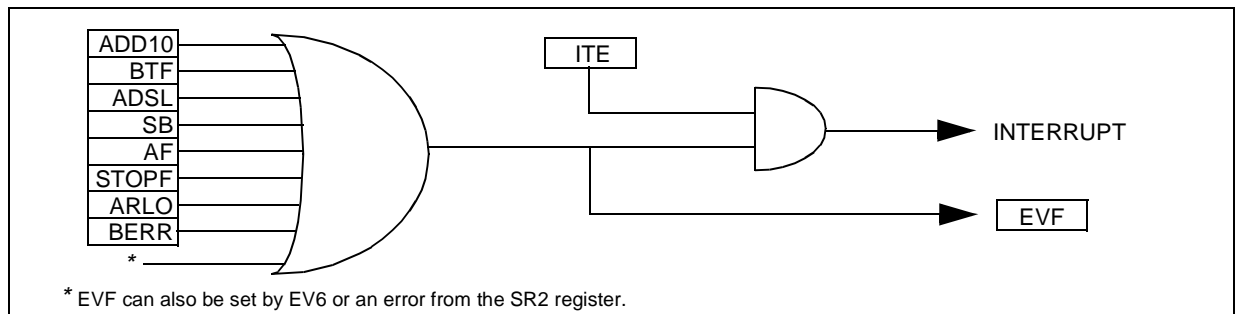
I<sup>2</sup>C BUS INTERFACE (Cont'd)

10.7.5 Low Power Modes

| Mode | Description   |
|------|---|
| WAIT | No effect on I <sup>2</sup> C interface.<br>I <sup>2</sup> C interrupts cause the device to exit from WAIT mode.  |
| HALT | I <sup>2</sup> C registers are frozen.<br>In HALT mode, the I <sup>2</sup> C interface is inactive and does not acknowledge data on the bus. The I <sup>2</sup> C interface resumes operation when the MCU is woken up by an interrupt with "exit from HALT mode" capability. |

10.7.6 Interrupts

Figure 66. Event Flags and Interrupt Generation



| 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 I<sup>2</sup>C 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<sup>2</sup>C BUS INTERFACE** (Cont'd)**10.7.7 Register Description****I<sup>2</sup>C 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<sup>2</sup>C interface, write the CR register **TWICE** with PE=1 as the first write only activates the interface (only PE is set).

Bit 4 = **ENGC** *Enable General Call*.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0). The 00h General Call address is acknowledged (01h ignored).

0: General Call disabled

1: General Call enabled

Bit 3 = **START** *Generation of a Start condition*.

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

– In master mode:

0: No start generation

1: Repeated start generation

– In slave mode:

0: No start generation

1: Start generation when the bus is free

Bit 2 = **ACK** *Acknowledge enable*.

This bit is set and cleared by software. It is also cleared by hardware when the interface is disabled (PE=0).

0: No acknowledge returned

1: Acknowledge returned after an address byte or a data byte is received

Bit 1 = **STOP** *Generation of a Stop condition*.

This bit is set and cleared by software. It is also cleared by hardware in master mode. Note: This bit is not cleared when the interface is disabled (PE=0).

– In master mode:

0: No stop generation

1: Stop generation after the current byte transfer or after the current Start condition is sent. The STOP bit is cleared by hardware when the Stop condition is sent.

– In slave mode:

0: No stop generation

1: Release the SCL and SDA lines after the current byte transfer (BTF=1). In this mode the STOP bit has to be cleared by software.

Bit 0 = **ITE** *Interrupt enable*.

This bit is set and cleared by software and cleared by hardware when the interface is disabled (PE=0).

0: Interrupts disabled

1: Interrupts enabled

Refer to Figure 66 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 65) is detected.

**I<sup>2</sup>C BUS INTERFACE** (Cont'd)

**I<sup>2</sup>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 65. It is also cleared by hardware when the interface is disabled (PE=0).

0: No event

1: One of the following events has occurred:

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

**Bit 6 = ADD10 10-bit addressing in Master mode.**

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

0: No ADD10 event occurred.

1: Master has sent first address byte (header)

**Bit 5 = TRA Transmitter/Receiver.**

When BTF is set, TRA=1 if a data byte has been transmitted. It is cleared automatically when BTF is cleared. It is also cleared by hardware after detection of Stop condition (STOPF=1), loss of bus

arbitration (ARLO=1) or when the interface is disabled (PE=0).

0: Data byte received (if BTF=1)

1: Data byte transmitted

**Bit 4 = BUSY Bus busy.**

This bit is set by hardware on detection of a Start condition and cleared by hardware on detection of a Stop condition. It indicates a communication in progress on the bus. This information is still updated when the interface is disabled (PE=0).

0: No communication on the bus

1: Communication ongoing on the bus

**Bit 3 = BTF Byte transfer finished.**

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

- Following a byte transmission, this bit is set after reception of the acknowledge clock pulse. In case an address byte is sent, this bit is set only after the EV6 event (See Figure 65). 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<sup>2</sup>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<sup>2</sup>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.

0: No acknowledge failure

1: Acknowledge failure

Bit 3 = **STOPF** *Stop detection (Slave mode)*.

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

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

0: No Stop condition detected

1: Stop condition detected

Bit 2 = **ARLO** *Arbitration lost*.

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

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

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

0: No arbitration lost detected

1: Arbitration lost detected

Bit 1 = **BERR** *Bus error*.

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

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

0: No misplaced Start or Stop condition

1: Misplaced Start or Stop condition

Bit 0 = **GCAL** *General Call (Slave mode)*.

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

0: No general call address detected on bus

1: general call address detected on bus

**I<sup>2</sup>C BUS INTERFACE** (Cont'd)

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

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

– Standard mode (FM/SM=0): F<sub>SCL</sub> ≤ 100kHz

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

– Fast mode (FM/SM=1): F<sub>SCL</sub> > 100kHz

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

Note: The programmed F<sub>SCL</sub> assumes no load on SCL and SDA lines.

**I<sup>2</sup>C 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<sup>2</sup>C BUS INTERFACE** (Cont'd)**I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C bus address of the interface. They are not cleared when the interface is disabled (PE=0).

**I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C bus address of the interface (10-bit mode only). They are not cleared when the interface is disabled (PE=0).

Bit 0 = Reserved.

I<sup>2</sup>C BUS INTERFACE (Cont'd)Table 24. I<sup>2</sup>C Register Map and Reset Values

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

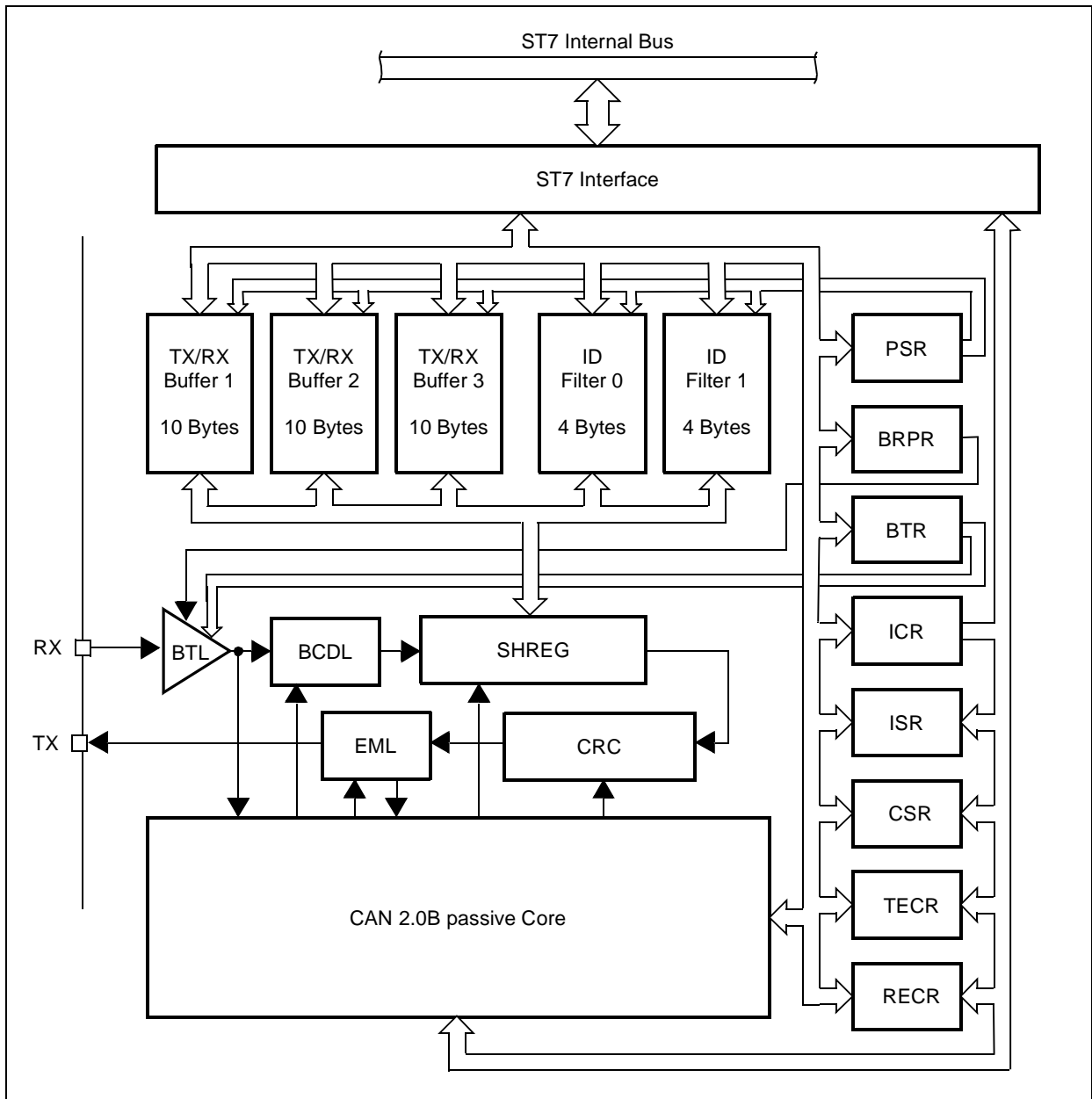
**10.8 CONTROLLER AREA NETWORK (CAN)**

**10.8.1 Introduction**

This peripheral is designed to support serial data exchanges using a multi-master contention based priority scheme as described in CAN specification Rev. 2.0 part A. It can also be connected to a 2.0 B network without problems, since extended frames

are checked for correctness and acknowledged accordingly although such frames cannot be transmitted nor received. The same applies to overload frames which are recognized but never initiated.

**Figure 67. CAN Block Diagram**



**CONTROLLER AREA NETWORK (Cont'd)****10.8.2 Main Features**

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

**10.8.3 Functional Description****10.8.3.1 Frame Formats**

A summary of all the CAN frame formats is given in Figure 68 for reference. It covers only the standard frame format since the extended one is only acknowledged.

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

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

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

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

**10.8.3.2 Hardware Blocks**

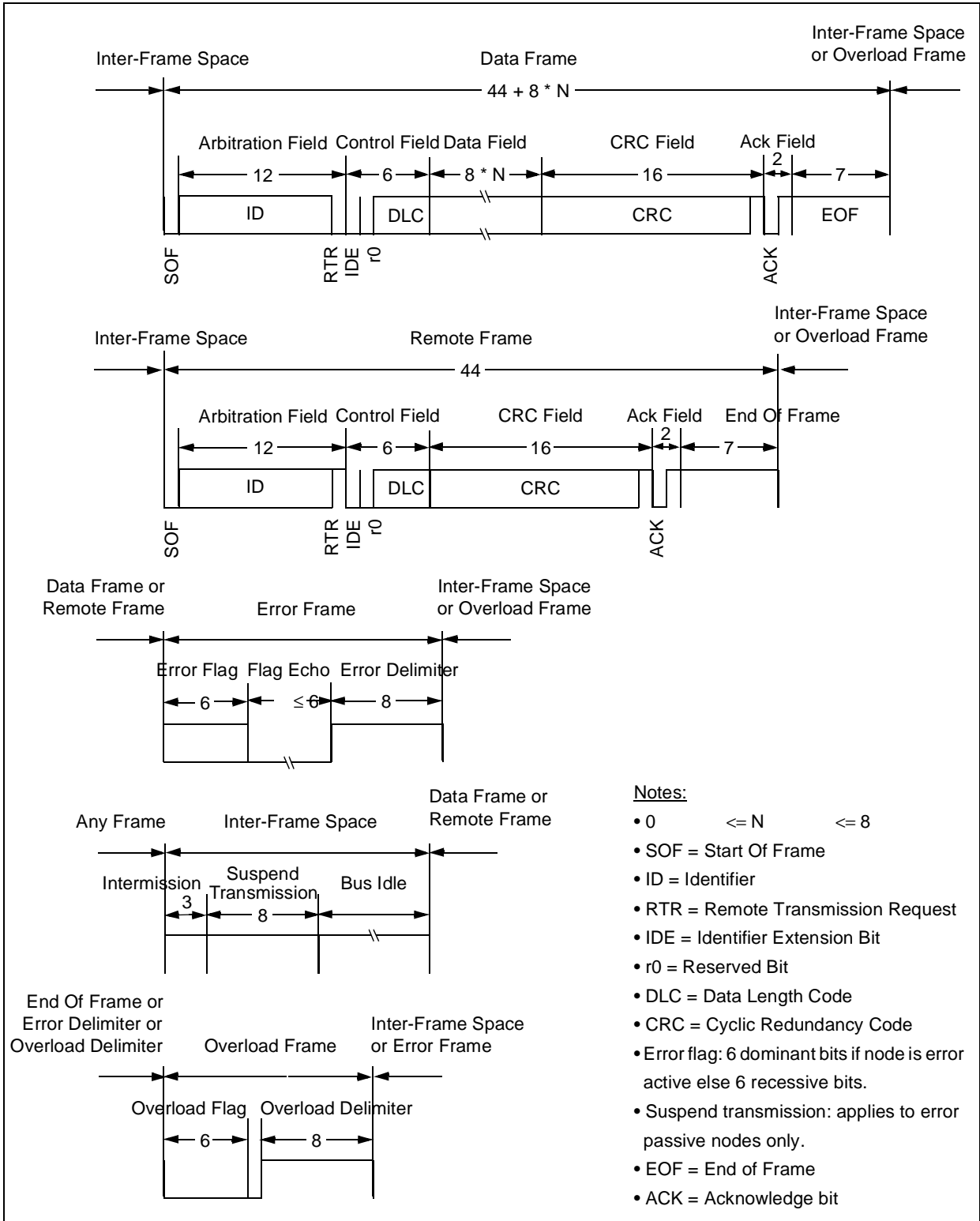
The CAN controller contains the following functional blocks (refer to Figure 67):

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



CONTROLLER AREA NETWORK (Cont'd)

Figure 68. CAN Frames



**CONTROLLER AREA NETWORK (Cont'd)**

**10.8.3.3 Modes of Operation**

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

- **STANDBY.** Standby mode is entered either on a chip reset or on resetting the RUN bit in the Control/Status Register (CSR). Any on-going transmission or reception operation is not interrupted and completes normally before the Bit Time Logic and the clock prescaler are turned off for minimum power consumption. This state is signalled by the RUN bit being read-back as 0. Once in standby, the only event monitored is the reception of a dominant bit which causes a wake-up interrupt if the SCIE bit of the Interrupt Control

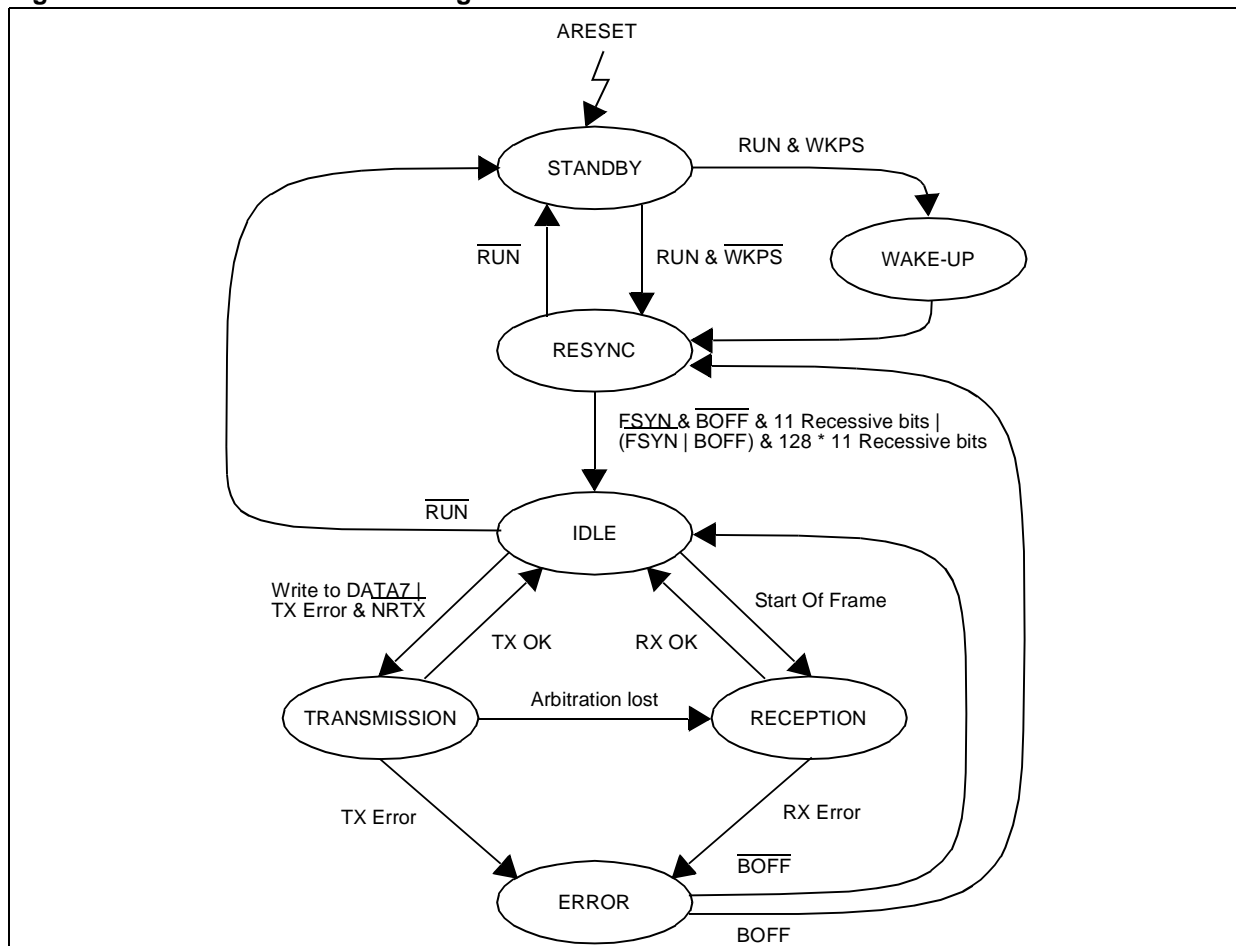
Register (ICR) is set.

The STANDBY mode is left by setting the RUN bit. If the WKPS bit is set in the CSR register, then the controller passes through WAKE-UP otherwise it enters RESYNC directly.

It is important to note that the wake-up mechanism is software-driven and therefore carries a significant time overhead. All messages received after the wake-up bit and before the controller is set to run and has completed synchronization are ignored.

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

**Figure 69. CAN Controller State Diagram**



**CONTROLLER AREA NETWORK (Cont'd)**

- **RESYNC.** The resynchronization mode is used to find the correct entry point for starting transmission or reception after the node has gone asynchronous either by going into the STANDBY or bus-off states.  
Resynchronization is achieved when 128 sequences of 11 recessive bits have been monitored unless the node is not bus-off and the FSYN bit in the CSR register is set in which case a single sequence of 11 recessive bits needs to be monitored.
- **IDLE.** The CAN controller looks for one of the following events: the RUN bit is reset, a Start Of Frame appears on the CAN bus or the DATA7 register of the currently active page is written to.
- **TRANSMISSION.** Once the LOCK bit of a Buffer Control/Status Register (BCSRx) has been set and read back as such, a transmit job can be submitted by writing to the DATA7 register. The message with the highest priority will be transmitted as soon as the CAN bus becomes idle. Among those messages with a pending transmission request, the highest priority is given to Buffer 3 then 2 and 1. If the transmission fails due to a lost arbitration or to an error while the NRTX bit of the CSR register is reset, then a new transmission attempt is performed. This goes on until the transmission ends successfully or until the job is cancelled by unlocking the buffer, by setting the NRTX bit or if the node ever enters bus-off or if a higher priority message becomes pending. The RDY bit in the BCSRx register, which was set since the job was submitted, gets reset. When a transmission is in progress, the BUSY bit in the BCSRx register is set. If it ends successfully then the TXIF bit in the Interrupt Status Register (ISR) is set, else the TEIF bit is set. An interrupt is generated in either case provided the TXIE and TEIE bits of the ICR register are set. The ETX bit in the same register is used to get an early transmit interrupt and to automatically unlock the transmitting buffer upon successful completion of its job. This enables the CPU to get a new transmit job pending by the end of the current transmission while always leaving two buffers available for reception. An uninterrupted stream of messages may be transmitted in this way at no overrun risk.

**Note 1:** Setting the SRTE bit of the CSR register allows transmitted messages to be simultaneously received when they pass the acceptance filtering. This is particularly useful for checking the integrity of the communication path.

**Note 2:** When the ETX bit is reset, the buffer with the highest priority and with a pending transmission request is always transmitted. When the ETX bit is set, once a buffer participates in the arbitration phase, it is sent until it wins the arbitration even if another transmission is requested from a buffer with a higher priority.

- **RECEPTION.** Once the CAN controller has synchronized itself onto the bus activity, it is ready for reception of new messages. Every incoming message gets its identifier compared to the acceptance filters. If the bitwise comparison of the selected bits ends up with a match for at least one of the filters then that message is elected for reception and a target buffer is searched for. This buffer will be the first one - order is 1 to 3 - that has the LOCK and RDY bits of its BCSRx register reset.
  - When no such buffer exists then an overrun interrupt is generated if the ORIE bit of the ICR register has been set. In this case the identifier of the last message is made available in the Last Identifier Register (LIDHR and LIDLRL) at least until it gets overwritten by a new identifier picked-up from the bus.
  - When a buffer does exist, the accepted message gets written into it, the ACC bit in the BCSRx register gets the number of the matching filter, the RDY and RXIF bits get set and an interrupt is generated if the RXIE bit in the ISR register is set.

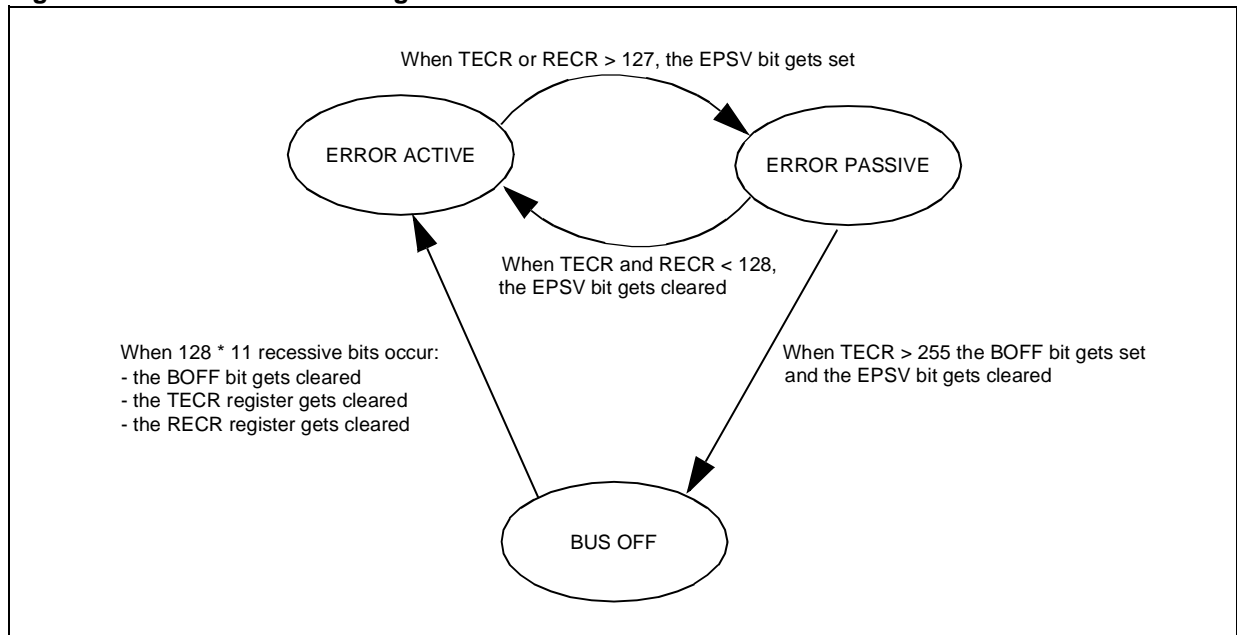
Up to three messages can be automatically received without intervention from the CPU because each buffer has its own set of status bits, greatly reducing the reactivity requirements in the processing of the receive interrupts.

**CONTROLLER AREA NETWORK (Cont'd)**

– **ERROR.** The error management as described in the CAN protocol is completely handled by hardware using 2 error counters which get incremented or decremented according to the error condition. Both of them may be read by the appli-

cation to determine the stability of the network. Moreover, as one of the node status bits (EPSV or BOFF of the CSR register) changes, an interrupt is generated if the SCIE bit is set in the ICR Register. Refer to Figure 70.

**Figure 70. CAN Error State Diagram**



## CONTROLLER AREA NETWORK (Cont'd)

### 10.8.3.4 Bit Timing Logic

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

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

- **Synchronisation segment (SYNC\_SEG)**: a bit change is expected to lie within this time segment. It has a fixed length of one time quanta ( $1 \times t_{CAN}$ ).
- **Bit segment 1 (BS1)**: defines the location of the sample point. It includes the PROP\_SEG and PHASE\_SEG1 of the CAN standard. Its duration is programmable between 1 and 16 time quanta but may be automatically lengthened to compensate for positive phase drifts due to differences in the frequency of the various nodes of the network.
- **Bit segment 2 (BS2)**: defines the location of the transmit point. It represents the PHASE\_SEG2 of the CAN standard. Its duration is programmable between 1 and 8 time quanta but may also be

automatically shortened to compensate for negative phase drifts.

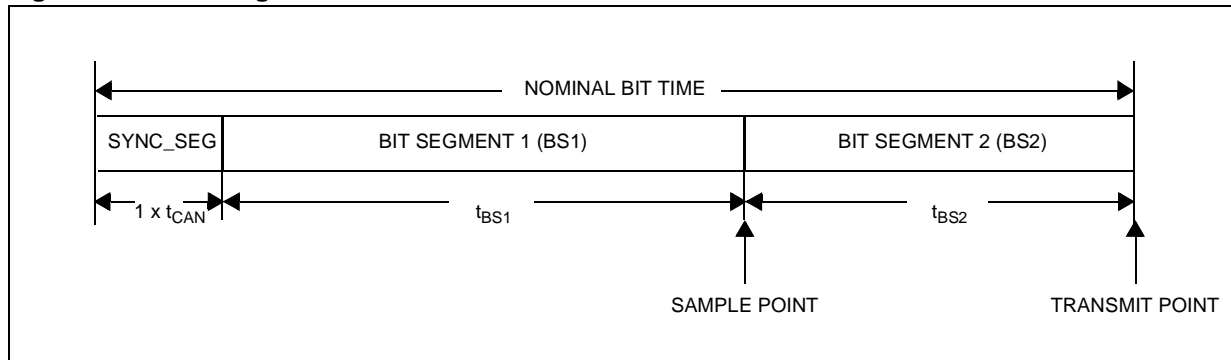
- **Resynchronization Jump Width (RJW)**: defines an upper bound to the amount of lengthening or shortening of the bit segments. It is programmable between 1 and 4 time quanta.

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

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

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

Figure 71. Bit Timing



**CONTROLLER AREA NETWORK (Cont'd)**

**10.8.4 Register Description**

The CAN registers are organized as 6 general purpose registers plus 5 pages of 16 registers spanning the same address space and primarily used for message and filter storage. The page actually selected is defined by the content of the Page Selection Register. Refer to Figure 72.

**10.8.4.1 General Purpose Registers**

**INTERRUPT STATUS REGISTER (ISR)**

Read/Write

Reset Value: 00h

|       |       |       |      |      |      |      |      |
|-------|-------|-------|------|------|------|------|------|
| 7     |       |       |      |      |      |      | 0    |
| RXIF3 | RXIF2 | RXIF1 | TXIF | SCIF | ORIF | TEIF | EPND |

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

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

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

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

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

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

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

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

**Caution:**

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

**CONTROLLER AREA NETWORK (Cont'd)****INTERRUPT CONTROL REGISTER (ICR)**

Read/Write

Reset Value: 00h

|   |      |      |      |      |      |      |     |
|---|------|------|------|------|------|------|-----|
| 7 |      |      |      |      |      |      | 0   |
| 0 | ESCI | RXIE | TXIE | SCIE | ORIE | TEIE | ETX |

Bit 6 = **ESCI** *Extended Status Change Interrupt*  
— Read/Set/Clear

Set by software to specify that SCIF is to be set on receive errors also.

Cleared by software to set SCIF only on status changes and wake-up but not on all receive errors.

Bit 5 = **RXIE** *Receive Interrupt Enable*  
— Read/Set/Clear

Set by software to enable an interrupt request whenever a message has been received free of errors.

Cleared by software to disable receive interrupt requests.

Bit 4 = **TXIE** *Transmit Interrupt Enable*  
— Read/Set/Clear

Set by software to enable an interrupt request whenever a message has been successfully transmitted.

Cleared by software to disable transmit interrupt requests.

Bit 3 = **SCIE** *Status Change Interrupt Enable*  
— Read/Set/Clear

Set by software to enable an interrupt request whenever the node's status changes in run mode or whenever a dominant pulse is received in standby mode.

Cleared by software to disable status change interrupt requests.

Bit 2 = **ORIE** *Overrun Interrupt Enable*  
— Read/Set/Clear

Set by software to enable an interrupt request whenever a message should be stored and no receive buffer is available.

Cleared by software to disable overrun interrupt requests.

Bit 1 = **TEIE** *Transmit Error Interrupt Enable*  
— Read/Set/Clear

Set by software to enable an interrupt whenever an error has been detected during transmission of a message.

Cleared by software to disable transmit error interrupts.

Bit 0 = **ETX** *Early Transmit Interrupt*  
— Read/Set/Clear

Set by software to request the transmit interrupt to occur as soon as the arbitration phase has been passed successfully.

Cleared by software to request the transmit interrupt to occur at the completion of the transfer.

**CONTROLLER AREA NETWORK (Cont'd)**

**CONTROL/STATUS REGISTER (CSR)**

Read/Write

Reset Value: 00h

|   |      |      |      |      |      |      |     |
|---|------|------|------|------|------|------|-----|
| 7 |      |      |      |      |      |      | 0   |
| 0 | BOFF | EPSV | SRTE | NRTX | FSYN | WKPS | RUN |

Bit 6 = **BOFF** *Bus-Off State*

— Read Only

Set by hardware to indicate that the node is in bus-off state, i.e. the Transmit Error Counter exceeds 255.

Reset by hardware to indicate that the node is involved in bus activities.

Bit 5 = **EPSV** *Error Passive State*

— Read Only

Set by hardware to indicate that the node is error passive.

Reset by hardware to indicate that the node is either error active (BOFF = 0) or bus-off.

Bit 4 = **SRTE** *Simultaneous Receive/Transmit Enable* — Read/Set/Clear

Set by software to enable simultaneous transmission and reception of a message passing the acceptance filtering. Allows to check the integrity of the communication path.

Reset by software to discard all messages transmitted by the node. Allows remote and data frames to share the same identifier.

Bit 3 = **NRTX** *No Retransmission*

— Read/Set/Clear

Set by software to disable the retransmission of unsuccessful messages.

Cleared by software to enable retransmission of messages until success is met.

Bit 2 = **FSYN** *Fast Synchronization*

— Read/Set/Clear

Set by software to enable a fast resynchronization when leaving standby mode, i.e. wait for only 11 recessive bits in a row.

Cleared by software to enable the standard resynchronization when leaving standby mode, i.e. wait for 128 sequences of 11 recessive bits.

Bit 1 = **WKPS** *Wake-up Pulse*

— Read/Set/Clear

Set by software to generate a dominant pulse when leaving standby mode.

Cleared by software for no dominant wake-up pulse.

Bit 0 = **RUN** *CAN Enable*

— Read/Set/Clear

Set by software to leave standby mode after 128 sequences of 11 recessive bits or just 11 recessive bits if FSYN is set.

Cleared by software to request a switch to the standby or low-power mode as soon as any on-going transfer is complete. Read-back as 1 in the meantime to enable proper signalling of the standby state. The CPU clock may therefore be safely switched OFF whenever RUN is read as 0.



**CONTROLLER AREA NETWORK (Cont'd)**

**BAUD RATE PRESCALER REGISTER (BRPR)**

Read/Write in Standby mode  
Reset Value: 00h

|      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|
| 7    |      |      |      |      |      |      | 0    |
| RJW1 | RJW0 | BRP5 | BRP4 | BRP3 | BRP2 | BRP1 | BRP0 |

**RJW[1:0]** determine the maximum number of time quanta by which a bit period may be shortened or lengthened to achieve resynchronization.

$$t_{RJW} = t_{CAN} * (RJW + 1)$$

**BRP[5:0]** determine the CAN system clock cycle time or time quanta which is used to build up the individual bit timing.

$$t_{CAN} = t_{CPU} * (BRP + 1)$$

Where  $t_{CPU}$  = time period of the CPU clock.  
The resulting baud rate can be computed by the formula:

$$BR = \frac{1}{t_{CPU} * (BRP + 1) * (BS1 + BS2 + 3)}$$

Note: Writing to this register is allowed only in Standby mode to prevent any accidental CAN protocol violation through programming errors.

**BIT TIMING REGISTER (BTR)**

Read/Write in Standby mode  
Reset Value: 23h

|   |      |      |      |      |      |      |      |
|---|------|------|------|------|------|------|------|
| 7 |      |      |      |      |      |      | 0    |
| 0 | BS22 | BS21 | BS20 | BS13 | BS12 | BS11 | BS10 |

**BS2[2:0]** determine the length of Bit Segment 2.  
 $t_{BS2} = t_{CAN} * (BS2 + 1)$

**BS1[3:0]** determine the length of Bit Segment 1.  
 $t_{BS1} = t_{CAN} * (BS1 + 1)$

Note: Writing to this register is allowed only in Standby mode to prevent any accidental CAN protocol violation through programming errors.

**PAGE SELECTION REGISTER (PSR)**

Read/Write  
Reset Value: 00h

|   |   |   |   |   |        |        |        |
|---|---|---|---|---|--------|--------|--------|
| 7 |   |   |   |   |        |        | 0      |
| 0 | 0 | 0 | 0 | 0 | PAGE 2 | PAGE 1 | PAGE 0 |

**PAGE[2:0]** determine which buffer or filter page is mapped at addresses 0010h to 001Fh.

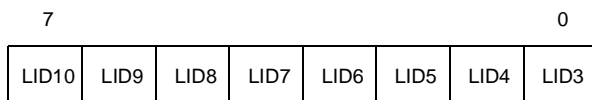
| PAGE2 | PAGE1 | PAGE0 | Page Title |
|-------|-------|-------|------------|
| 0     | 0     | 0     | Diagnosis  |
| 0     | 0     | 1     | Buffer 1   |
| 0     | 1     | 0     | Buffer 2   |
| 0     | 1     | 1     | Buffer 3   |
| 1     | 0     | 0     | Filters    |
| 1     | 0     | 1     | Reserved   |
| 1     | 1     | 0     | Reserved   |
| 1     | 1     | 1     | Reserved   |

**CONTROLLER AREA NETWORK (Cont'd)**

**10.8.4.2 Paged Registers**

**LAST IDENTIFIER HIGH REGISTER (LIDHR)**

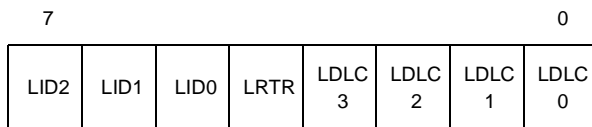
Read/Write  
Reset Value: Undefined



**LID[10:3]** are the most significant 8 bits of the last Identifier read on the CAN bus.

**LAST IDENTIFIER LOW REGISTER (LIDLR)**

Read/Write  
Reset Value: Undefined



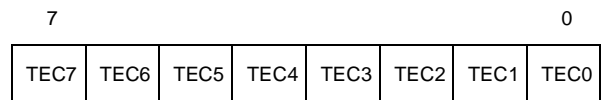
**LID[2:0]** are the least significant 3 bits of the last Identifier read on the CAN bus.

**LRTR** is the last Remote Transmission Request bit read on the CAN bus.

**LDLC[3:0]** is the last Data Length Code read on the CAN bus.

**TRANSMIT ERROR COUNTER REG. (TECR)**

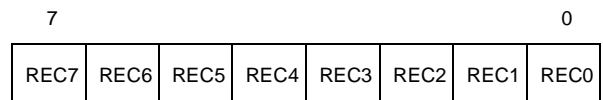
Read Only  
Reset Value: 00h



**TEC[7:0]** is the least significant byte of the 9-bit Transmit Error Counter implementing part of the fault confinement mechanism of the CAN protocol. In case of an error during transmission, this counter is incremented by 8. It is decremented by 1 after every successful transmission. When the counter value exceeds 127, the CAN controller enters the error passive state. When a value of 256 is reached, the CAN controller is disconnected from the bus.

**RECEIVE ERROR COUNTER REG. (RECR)**

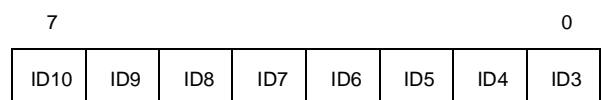
Page: 00h — Read Only  
Reset Value: 00h



**REC[7:0]** is the Receive Error Counter implementing part of the fault confinement mechanism of the CAN protocol. In case of an error during reception, this counter is incremented by 1 or by 8 depending on the error condition as defined by the CAN standard. After every successful reception the counter is decremented by 1 or reset to 120 if its value was higher than 128. When the counter value exceeds 127, the CAN controller enters the error passive state.

**IDENTIFIER HIGH REGISTERS (IDHRx)**

Read/Write  
Reset Value: Undefined



**ID[10:3]** are the most significant 8 bits of the 11-bit message identifier. The identifier acts as the message's name, used for bus access arbitration and acceptance filtering.

**CONTROLLER AREA NETWORK (Cont'd)****IDENTIFIER LOW REGISTERS (IDLRx)**

Read/Write

Reset Value: Undefined

|     |     |     |     |      |      |      |      |
|-----|-----|-----|-----|------|------|------|------|
| 7   |     |     |     |      |      |      | 0    |
| ID2 | ID1 | ID0 | RTR | DLC3 | DLC2 | DLC1 | DLC0 |

**ID[2:0]** are the least significant 3 bits of the 11-bit message identifier.

**RTR** is the Remote Transmission Request bit. It is set to indicate a remote frame and reset to indicate a data frame.

**DLC[3:0]** is the Data Length Code. It gives the number of bytes in the data field of the message. The valid range is 0 to 8.

**DATA REGISTERS (DATA0-7x)**

Read/Write

Reset Value: Undefined

|           |           |           |           |           |           |           |           |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| 7         |           |           |           |           |           |           | 0         |
| DATA<br>7 | DATA<br>6 | DATA<br>5 | DATA<br>4 | DATA<br>3 | DATA<br>2 | DATA<br>1 | DATA<br>0 |

**DATA[7:0]** is a message data byte. Up to eight such bytes may be part of a message. Writing to byte DATA7 initiates a transmit request and should always be done even when DATA7 is not part of the message.

**BUFFER CONTROL/STATUS REGs. (BCSRx)**

Read/Write

Reset Value: 00h

|   |   |   |   |     |     |      |      |
|---|---|---|---|-----|-----|------|------|
| 7 |   |   |   |     | 0   |      |      |
| 0 | 0 | 0 | 0 | ACC | RDY | BUSY | LOCK |

Bit 3 = **ACC** *Acceptance Code*

— Read Only

Set by hardware with the id of the highest priority filter which accepted the message stored in the buffer.

ACC = 0: Match for Filter/Mask0. Possible match for Filter/Mask1.

ACC = 1: No match for Filter/Mask0 and match for Filter/Mask1.

Reset by hardware when either RDY or RXIF gets reset.

Bit 2 = **RDY** *Message Ready*

— Read/Clear

Set by hardware to signal that a new error-free message is available (LOCK = 0) or that a transmission request is pending (LOCK = 1).

Cleared by software when LOCK = 0 to release the buffer and to clear the corresponding RXIF bit in the Interrupt Status Register.

Cleared by hardware when LOCK = 1 to indicate that the transmission request has been serviced or cancelled.

Bit 1 = **BUSY** *Busy Buffer*

— Read Only

Set by hardware when the buffer is being filled (LOCK = 0) or emptied (LOCK = 1).

Reset by hardware when the buffer is not accessed by the CAN core for transmission nor reception purposes.

Bit 0 = **LOCK** *Lock Buffer*

— Read/Set/Clear

Set by software to lock a buffer. No more message can be received into the buffer thus preserving its content and making it available for transmission.

Cleared by software to make the buffer available for reception. Cancels any pending transmission request.

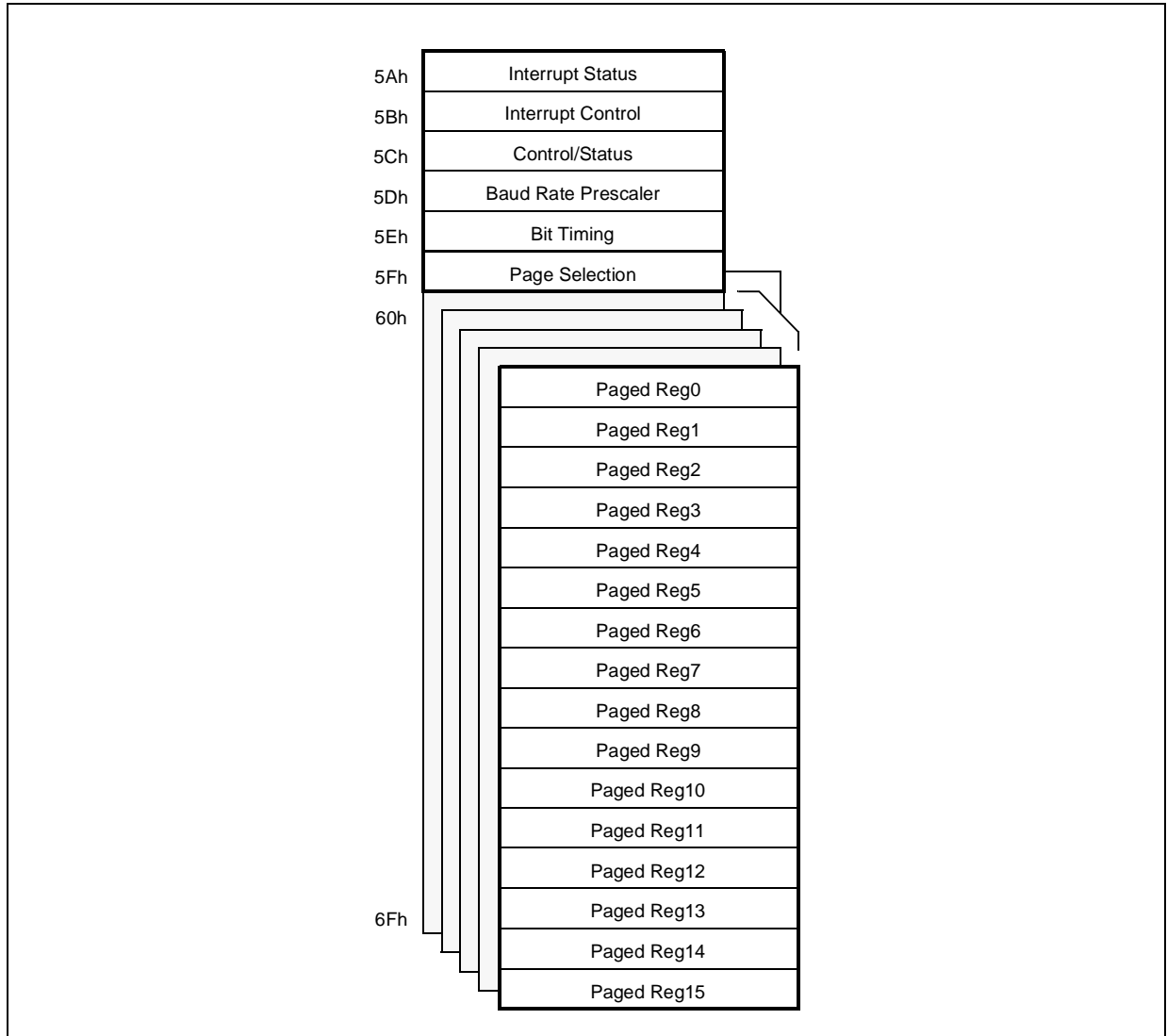
Cleared by hardware once a message has been successfully transmitted provided the early transmit interrupt mode is on. Left untouched otherwise.

Note that in order to prevent any message corruption or loss of context, LOCK cannot be set nor reset while BUSY is set. Trying to do so will result in LOCK not changing state.



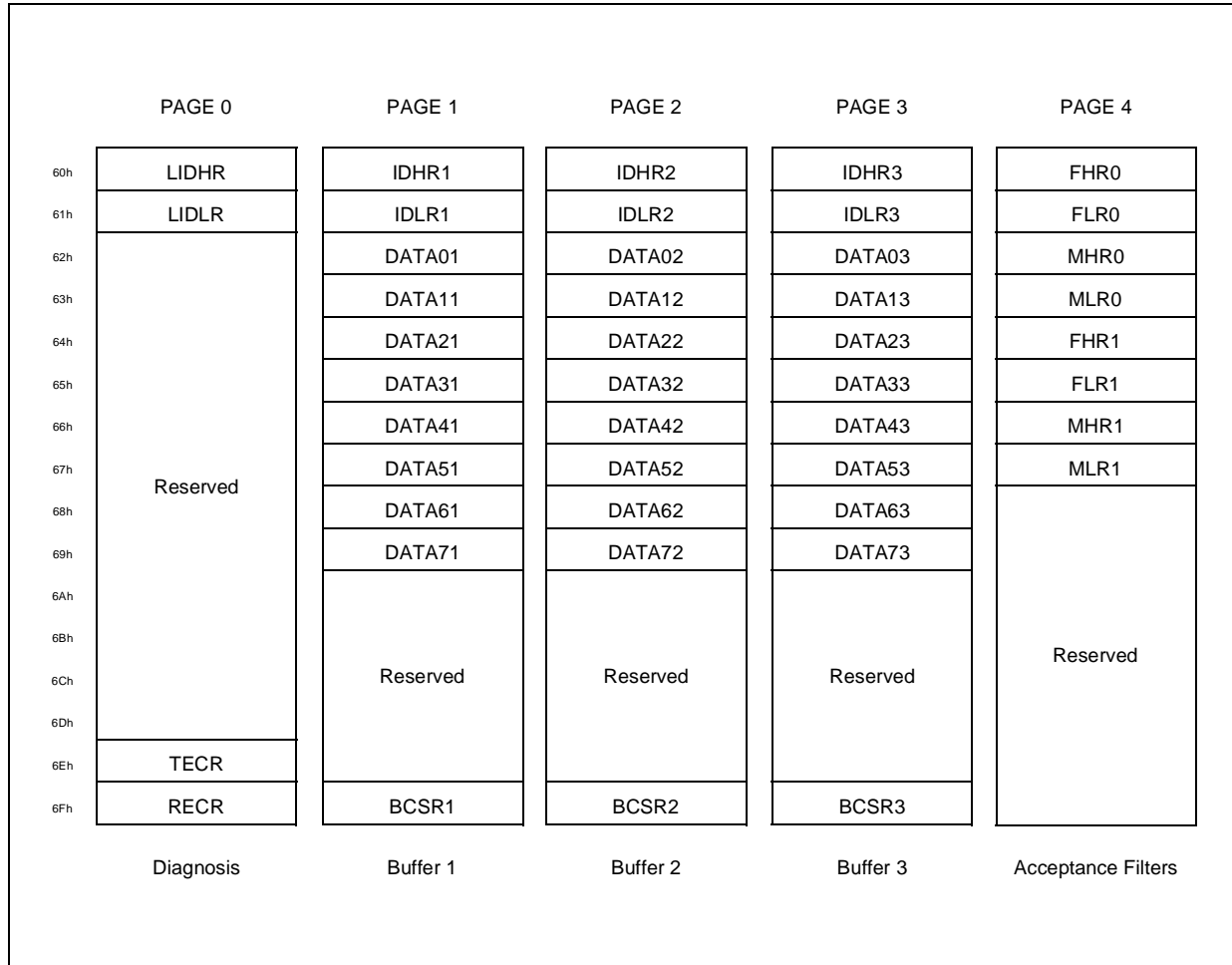
## CONTROLLER AREA NETWORK (Cont'd)

Figure 72. CAN Register Map



CONTROLLER AREA NETWORK (Cont'd)

Figure 73. Page Maps



## CONTROLLER AREA NETWORK (Cont'd)

Table 25. CAN Register Map and Reset Values

| Address (Hex.) | Page   | Register Label                 | 7          | 6          | 5          | 4         | 3          | 2          | 1          | 0          |
|----------------|--------|--------------------------------|------------|------------|------------|-----------|------------|------------|------------|------------|
| 5A             | X      | <b>CANISR</b><br>Reset Value   | RXIF3<br>0 | RXIF2<br>0 | RXIF1<br>0 | TXIF<br>0 | SCIF<br>0  | ORIF<br>0  | TEIF<br>0  | EPND<br>0  |
| 5B             |        | <b>CANICR</b><br>Reset Value   | 0          | ESCI<br>0  | RXIE<br>0  | TXIE<br>0 | SCIE<br>0  | ORIE<br>0  | TEIE<br>0  | ETX<br>0   |
| 5C             |        | <b>CANCSR</b><br>Reset Value   | 0          | BOFF<br>0  | EPSV<br>0  | SRTE<br>0 | NRTX<br>0  | FSYN<br>0  | WKPS<br>0  | RUN<br>0   |
| 5D             |        | <b>CANBRPR</b><br>Reset Value  | RJW1<br>0  | RJW0<br>0  | BRP5<br>0  | BRP4<br>0 | BRP3<br>0  | BRP2<br>0  | BRP1<br>0  | BRP0<br>0  |
| 5E             |        | <b>CANBTR</b><br>Reset Value   | 0          | BS22<br>0  | BS21<br>1  | BS20<br>0 | BS13<br>0  | BS12<br>0  | BS11<br>1  | BS10<br>1  |
| 5F             |        | <b>CANPSR</b><br>Reset Value   | 0          | 0          | 0          | 0         | 0          | PAGE2<br>0 | PAGE1<br>0 | PAGE0<br>0 |
| 60             | 0      | <b>CANLIDHR</b><br>Reset Value | LID10<br>x | LID9<br>x  | LID8<br>x  | LID7<br>x | LID6<br>x  | LID5<br>x  | LID4<br>x  | LID3<br>x  |
|                | 1 to 3 | <b>CANIDHRx</b><br>Reset Value | ID10<br>x  | ID9<br>x   | ID8<br>x   | ID7<br>x  | ID6<br>x   | ID5<br>x   | ID4<br>x   | ID3<br>x   |
| 60, 64         | 4      | <b>CANFHRx</b><br>Reset Value  | FIL11<br>x | FIL10<br>x | FIL9<br>x  | FIL8<br>x | FIL7<br>x  | FIL6<br>x  | FIL5<br>x  | FIL4<br>x  |
| 61             | 0      | <b>CANLIDLR</b><br>Reset Value | LID2<br>x  | LID1<br>x  | LID0<br>x  | LRTR<br>x | LDLC3<br>x | LDLC2<br>x | LDLC1<br>x | LDLC0<br>x |
|                | 1 to 3 | <b>CANIDLRx</b><br>Reset Value | ID2<br>x   | ID1<br>x   | ID0<br>x   | RTR<br>x  | DLC3<br>x  | DLC2<br>x  | DLC1<br>x  | DLC0<br>x  |
| 61, 65         | 4      | <b>CANFLRx</b><br>Reset Value  | FIL3<br>x  | FIL2<br>x  | FIL1<br>x  | FIL0<br>x | 0          | 0          | 0          | 0          |
| 62 to 69       | 1 to 3 | <b>CANDRx</b><br>Reset Value   | MSB<br>x   | x          | x          | x         | x          | x          | x          | LSB<br>x   |
| 62, 66         | 4      | <b>CANMHRx</b><br>Reset Value  | MSK11<br>x | MSK10<br>x | MSK9<br>x  | MSK8<br>x | MSK7<br>x  | MSK6<br>x  | MSK5<br>x  | MSK4<br>x  |
| 63, 67         | 4      | <b>CANMLRx</b><br>Reset Value  | MSK3<br>x  | MSK2<br>x  | MSK1<br>x  | MSK0<br>x | 0          | 0          | 0          | 0          |
| 6E             | 0      | <b>CANTECR</b><br>Reset Value  | MSB<br>0   | 0          | 0          | 0         | 0          | 0          | 0          | LSB<br>0   |
| 6F             |        | <b>CANRECR</b><br>Reset Value  | MSB<br>0   | 0          | 0          | 0         | 0          | 0          | 0          | LSB<br>0   |
|                | 1 to 3 | <b>CANBCSRx</b><br>Reset Value | 0          | 0          | 0          | 0         | ACC<br>0   | RDY<br>0   | BUSY<br>0  | LOCK<br>0  |

### 10.9 10-BIT A/D CONVERTER (ADC)

#### 10.9.1 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.

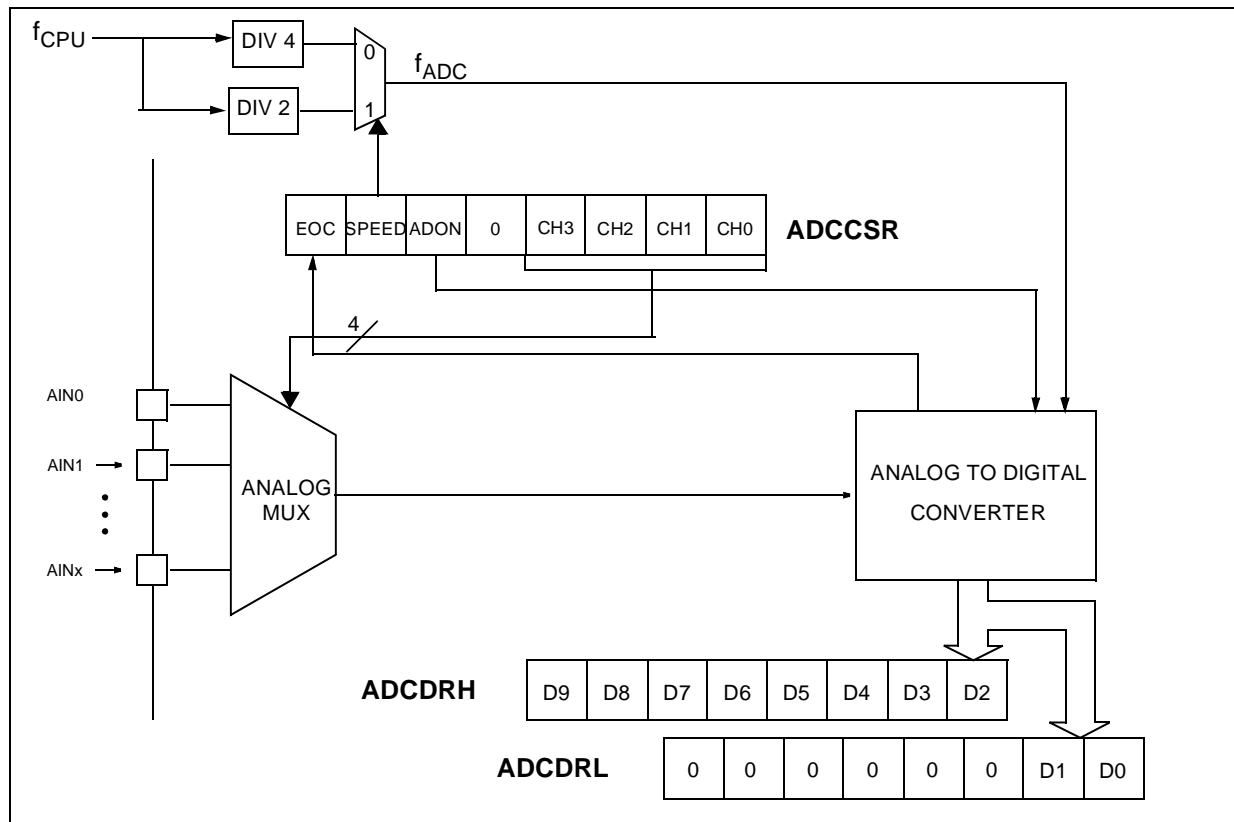
The result of the conversion is stored in a 10-bit Data Register. The A/D converter is controlled through a Control/Status Register.

#### 10.9.2 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 74.

Figure 74. ADC Block Diagram





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

### 10.9.3 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 allotted time.

#### 10.9.3.1 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.

#### 10.9.3.2 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 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.

#### 10.9.3.3 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.

#### 10.9.4 Low Power Modes

**Note:** The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed and between single shot conversions.

| Mode | Description   |
|------|---|
| WAIT | No effect on A/D Converter  |
| HALT | A/D Converter disabled.<br>After wakeup from Halt mode, the A/D Converter requires a stabilization time $t_{STAB}$ (see Electrical Characteristics) before accurate conversions can be performed. |

#### 10.9.5 Interrupts

None.

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

10.9.6 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 | 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 26. ADC Register Map and Reset Values

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

## 11 INSTRUCTION SET

### 11.1 ST7 ADDRESSING MODES

The ST7 Core features 17 different addressing modes which can be classified in 7 main groups:

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

The ST7 Instruction set is designed to minimize the number of bytes required per instruction: To do

so, most of the addressing modes may be subdivided in two sub-modes called long and short:

- Long addressing mode is more powerful because it can use the full 64 Kbyte address space, however it uses more bytes and more CPU cycles.
- Short addressing mode is less powerful because it can generally only access page zero (0000h - 00FFh range), but the instruction size is more compact, and faster. All memory to memory instructions use short addressing modes only (CLR, CPL, NEG, BSET, BRES, BTJT, BTJF, INC, DEC, RLC, RRC, SLL, SRL, SRA, SWAP)

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

**Table 27. ST7 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)

### 11.1.1 Inherent

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

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

### 11.1.2 Immediate

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

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

### 11.1.3 Direct

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

The direct addressing mode consists of two sub-modes:

#### Direct (short)

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

#### Direct (long)

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

### 11.1.4 Indexed (No Offset, Short, Long)

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

The indirect addressing mode consists of three sub-modes:

#### Indexed (No Offset)

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

#### Indexed (Short)

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

#### Indexed (long)

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

### 11.1.5 Indirect (Short, Long)

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

The pointer address follows the opcode. The indirect addressing mode consists of two sub-modes:

#### Indirect (short)

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

#### Indirect (long)

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

**INSTRUCTION SET OVERVIEW (Cont'd)**

**11.1.6 Indirect Indexed (Short, Long)**

This is a combination of indirect and short indexed addressing modes. The operand is referenced by its memory address, which is defined by the unsigned addition of an index register value (X or Y) with a pointer value located in memory. The pointer address follows the opcode.

The indirect indexed addressing mode consists of two sub-modes:

**Indirect Indexed (Short)**

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

**Indirect Indexed (Long)**

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

**Table 28. 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/Subtractions 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      |

**11.1.7 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 sub-modes:

**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)

### 11.2 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 |      |      |      |      |       |     |     |
| Interrupt management             | TRAP | WFI  | HALT | IRET |      |       |     |     |
| Condition Code Flag modification | SIM  | RIM  | SCF  | RCF  |      |       |     |     |

#### Using a pre-byte

The instructions are described with one to four opcodes.

In order to extend the number of available opcodes for an 8-bit CPU (256 opcodes), three different prebyte opcodes are defined. These prebytes modify the meaning of the instruction they precede.

The whole instruction becomes:

PC-2            End of previous instruction  
 PC-1            Prebyte  
 PC              opcode  
 PC+1            Additional word (0 to 2) according to the number of bytes required to compute the effective address

These prebytes enable instruction in Y as well as indirect addressing modes to be implemented. They precede the opcode of the instruction in X or the instruction using direct addressing mode. The prebytes are:

PDY 90        Replace an X based instruction using immediate, direct, indexed, or inherent addressing mode by a Y one.

PIX 92        Replace an instruction using direct, direct bit, or direct relative addressing mode to an instruction using the corresponding indirect addressing mode.

It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.

PIY 91        Replace an instruction using X indirect indexed addressing mode by a Y one.

INSTRUCTION SET OVERVIEW (Cont'd)

| Mnemo | Description                | Function/Example       | Dst    | Src | I1 | H | I0 | N | Z | C |
|-------|----------------------------|------------------------|--------|-----|----|---|----|---|---|---|
| ADC   | Add with Carry             | A = A + M + C          | A      | M   |    | H |    | N | Z | C |
| ADD   | Addition                   | A = A + M              | A      | M   |    | H |    | N | Z | C |
| AND   | Logical And                | A = A . M              | A      | M   |    |   |    | N | Z |   |
| BCP   | Bit compare A, Memory      | tst (A . M)            | A      | M   |    |   |    | N | Z |   |
| BRES  | Bit Reset                  | bres Byte, #3          | M      |     |    |   |    |   |   |   |
| BSET  | Bit Set                    | bset Byte, #3          | M      |     |    |   |    |   |   |   |
| BTJF  | Jump if bit is false (0)   | btjf Byte, #3, Jmp1    | M      |     |    |   |    |   |   | C |
| BTJT  | Jump if bit is true (1)    | btjt Byte, #3, Jmp1    | M      |     |    |   |    |   |   | C |
| CALL  | Call subroutine            |                        |        |     |    |   |    |   |   |   |
| CALLR | Call subroutine relative   |                        |        |     |    |   |    |   |   |   |
| CLR   | Clear                      |                        | reg, M |     |    |   |    | 0 | 1 |   |
| CP    | Arithmetic Compare         | tst(Reg - M)           | reg    | M   |    |   |    | N | Z | C |
| CPL   | One Complement             | A = FFH-A              | reg, M |     |    |   |    | N | Z | 1 |
| DEC   | Decrement                  | dec Y                  | reg, M |     |    |   |    | N | Z |   |
| HALT  | Halt                       |                        |        |     | 1  |   | 0  |   |   |   |
| IRET  | Interrupt routine return   | Pop CC, A, X, PC       |        |     | I1 | H | I0 | N | Z | C |
| INC   | Increment                  | inc X                  | reg, M |     |    |   |    | N | Z |   |
| JP    | Absolute Jump              | jp [TBL.w]             |        |     |    |   |    |   |   |   |
| JRA   | Jump relative always       |                        |        |     |    |   |    |   |   |   |
| JRT   | Jump relative              |                        |        |     |    |   |    |   |   |   |
| JRF   | Never jump                 | jrf *                  |        |     |    |   |    |   |   |   |
| JRIH  | Jump if Port B INT pin = 1 | (no Port B Interrupts) |        |     |    |   |    |   |   |   |
| JRIL  | Jump if Port B INT pin = 0 | (Port B interrupt)     |        |     |    |   |    |   |   |   |
| JRH   | Jump if H = 1              | H = 1 ?                |        |     |    |   |    |   |   |   |
| JRNH  | Jump if H = 0              | H = 0 ?                |        |     |    |   |    |   |   |   |
| JRM   | Jump if I1:0 = 11          | I1:0 = 11 ?            |        |     |    |   |    |   |   |   |
| JRNM  | Jump if I1:0 <> 11         | I1:0 <> 11 ?           |        |     |    |   |    |   |   |   |
| JRMI  | Jump if N = 1 (minus)      | N = 1 ?                |        |     |    |   |    |   |   |   |
| JRPL  | Jump if N = 0 (plus)       | N = 0 ?                |        |     |    |   |    |   |   |   |
| JREQ  | Jump if Z = 1 (equal)      | Z = 1 ?                |        |     |    |   |    |   |   |   |
| JRNE  | Jump if Z = 0 (not equal)  | Z = 0 ?                |        |     |    |   |    |   |   |   |
| JRC   | Jump if C = 1              | C = 1 ?                |        |     |    |   |    |   |   |   |
| JRNC  | Jump if C = 0              | C = 0 ?                |        |     |    |   |    |   |   |   |
| JRULT | Jump if C = 1              | Unsigned <             |        |     |    |   |    |   |   |   |
| JRUGE | Jump if C = 0              | Jmp if unsigned >=     |        |     |    |   |    |   |   |   |
| JRUGT | Jump if (C + Z = 0)        | Unsigned >             |        |     |    |   |    |   |   |   |



## INSTRUCTION SET OVERVIEW (Cont'd)

| Mnemo | Description            | Function/Example    | Dst     | Src     | I1 | H | I0 | N | Z | C |
|-------|------------------------|---------------------|---------|---------|----|---|----|---|---|---|
| JRULE | Jump if (C + Z = 1)    | Unsigned <=         |         |         |    |   |    |   |   |   |
| LD    | Load                   | dst <= src          | reg, M  | M, reg  |    |   |    | N | Z |   |
| MUL   | Multiply               | X,A = X * A         | A, X, Y | X, Y, A |    | 0 |    |   |   | 0 |
| NEG   | Negate (2's compl)     | neg \$10            | reg, M  |         |    |   |    | N | Z | C |
| NOP   | No Operation           |                     |         |         |    |   |    |   |   |   |
| OR    | OR operation           | A = A + M           | A       | M       |    |   |    | N | Z |   |
| POP   | Pop from the Stack     | pop reg             | reg     | M       |    |   |    |   |   |   |
|       |                        | pop CC              | CC      | M       | I1 | H | I0 | N | Z | C |
| PUSH  | Push onto the Stack    | push Y              | M       | reg, CC |    |   |    |   |   |   |
| RCF   | Reset carry flag       | C = 0               |         |         |    |   |    |   |   | 0 |
| RET   | Subroutine Return      |                     |         |         |    |   |    |   |   |   |
| RIM   | Enable Interrupts      | I1:0 = 10 (level 0) |         |         | 1  |   | 0  |   |   |   |
| RLC   | Rotate left true C     | C <= A <= C         | reg, M  |         |    |   |    | N | Z | C |
| RRC   | Rotate right true C    | C => A => C         | reg, M  |         |    |   |    | N | Z | C |
| RSP   | Reset Stack Pointer    | S = Max allowed     |         |         |    |   |    |   |   |   |
| SBC   | Substract with Carry   | A = A - M - C       | A       | M       |    |   |    | N | Z | C |
| SCF   | Set carry flag         | C = 1               |         |         |    |   |    |   |   | 1 |
| SIM   | Disable Interrupts     | I1:0 = 11 (level 3) |         |         | 1  |   | 1  |   |   |   |
| SLA   | Shift left Arithmetic  | C <= A <= 0         | reg, M  |         |    |   |    | N | Z | C |
| SLL   | Shift left Logic       | C <= A <= 0         | reg, M  |         |    |   |    | N | Z | C |
| SRL   | Shift right Logic      | 0 => A => C         | reg, M  |         |    |   | 0  | Z | C |   |
| SRA   | Shift right Arithmetic | A7 => A => C        | reg, M  |         |    |   |    | N | Z | C |
| SUB   | Substraction           | A = A - M           | A       | M       |    |   |    | N | Z | C |
| SWAP  | SWAP nibbles           | A7-A4 <=> A3-A0     | reg, M  |         |    |   |    | N | Z |   |
| TNZ   | Test for Neg & Zero    | tnz lbl1            |         |         |    |   |    | N | Z |   |
| TRAP  | S/W trap               | S/W interrupt       |         |         | 1  |   | 1  |   |   |   |
| WFI   | Wait for Interrupt     |                     |         |         | 1  |   | 0  |   |   |   |
| XOR   | Exclusive OR           | A = A XOR M         | A       | M       |    |   |    | N | Z |   |

## 12 ELECTRICAL CHARACTERISTICS

### 12.1 PARAMETER CONDITIONS

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

#### 12.1.1 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  $T_A=25^\circ\text{C}$  and  $T_A=T_{A\text{max}}$  (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 ( $\text{mean} \pm 3\Sigma$ ).

#### 12.1.2 Typical values

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

Typical ADC accuracy values are determined by characterization of a batch of samples from a standard diffusion lot over the full temperature range, where 95% of the devices have an error less than or equal to the value indicated ( $\text{mean} \pm 2\Sigma$ ).

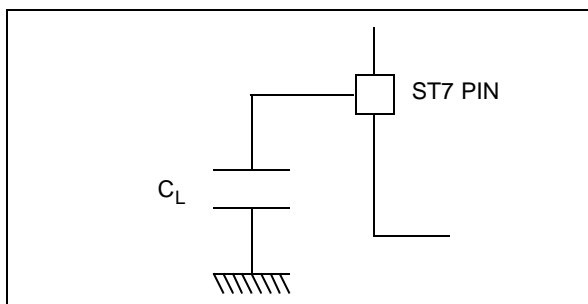
#### 12.1.3 Typical curves

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

#### 12.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in Figure 75.

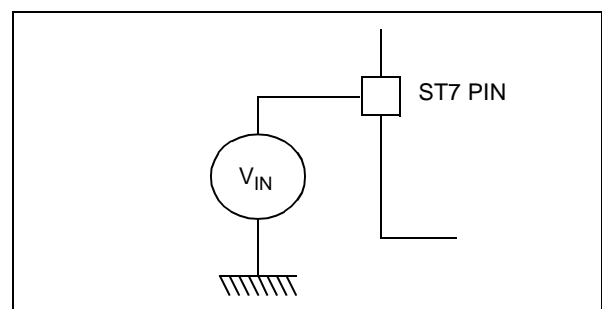
**Figure 75. Pin loading conditions**



#### 12.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 76.

**Figure 76. Pin input voltage**



## 12.2 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 condi-

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

### 12.2.1 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 12.7.3 on page 171 |      |
| $V_{ESD(MM)}$                             | Electro-static discharge voltage (Machine Model)    |                                |      |

### 12.2.2 Current Characteristics

| Symbol                     | Ratings  | Maximum value | Unit |
|----------------------------|--|---------------|------|
| $I_{VDD}$                  | Total current into $V_{DD}$ power lines (source) <sup>3)</sup>         | 150           | mA   |
| $I_{VSS}$                  | Total current out of $V_{SS}$ ground lines (sink) <sup>3)</sup>        | 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                                       | $\pm 5$       |      |
|                            | Injected current on $\overline{RESET}$ pin                             | $\pm 5$       |      |
|                            | Injected current on OSC1 and OSC2 pins                                 | $\pm 5$       |      |
|                            | Injected current on any other pin <sup>5) &amp; 6)</sup>               | $\pm 5$       |      |
| $\Sigma I_{INJ(PIN)}^{2)}$ | Total injected current (sum of all I/O and control pins) <sup>5)</sup> | $\pm 25$      |      |

#### Notes:

- Directly connecting the  $\overline{RESET}$  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: 4.7k $\Omega$  for  $\overline{RESET}$ , 10k $\Omega$  for I/Os). For the same reason, unused I/O pins must not be directly tied to  $V_{DD}$  or  $V_{SS}$ .
- When the current limitation is not possible, the  $V_{IN}$  absolute maximum rating 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  $V_{IN} < V_{SS}$ .
- All power ( $V_{DD}$ ) and ground ( $V_{SS}$ ) lines must always be connected to the external supply.
- Negative injection disturbs the analog performance of the device. See note in “ADC Accuracy” on page 186. For best reliability, it is recommended to avoid negative injection of more than 1.6mA.
- 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.
- True open drain I/O port pins do not accept positive injection.

12.2.3 Thermal Characteristics

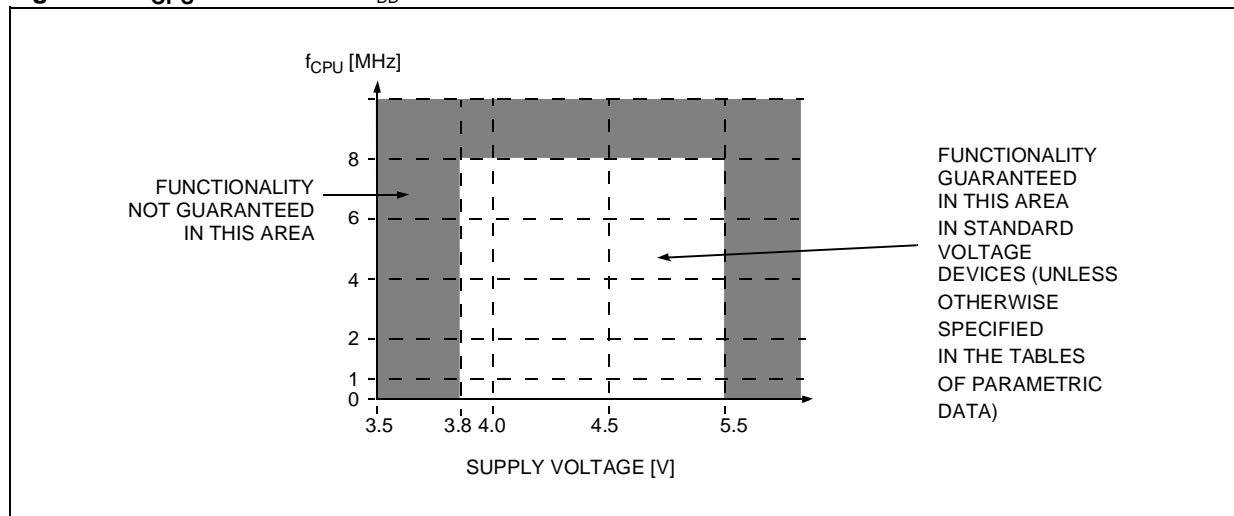
| Symbol           | Ratings   | Value       | Unit |
|------------------|---|-------------|------|
| T <sub>STG</sub> | Storage temperature range   | -65 to +150 | °C   |
| T <sub>J</sub>   | Maximum junction temperature (see Section 13.2 THERMAL CHARACTERISTICS) |             |      |

12.3 OPERATING CONDITIONS

12.3.1 General Operating Conditions

| Symbol           | Parameter   | Conditions                      | Min | Max | Unit |
|------------------|---|---------------------------------|-----|-----|------|
| f <sub>CPU</sub> | Internal clock frequency                            |                                 | 0   | 8   | MHz  |
| V <sub>DD</sub>  | Standard voltage devices (except Flash Write/Erase) |                                 | 3.8 | 5.5 | V    |
|                  | Operating Voltage for Flash Write/Erase             | V <sub>PP</sub> = 11.4 to 12.6V | 4.5 | 5.5 |      |
| T <sub>A</sub>   | 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 |      |
|                  |   | C Suffix Version                | -40 | 125 |      |

Figure 77. f<sub>CPU</sub> Max Versus V<sub>DD</sub>



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

## OPERATING CONDITIONS (Cont'd)

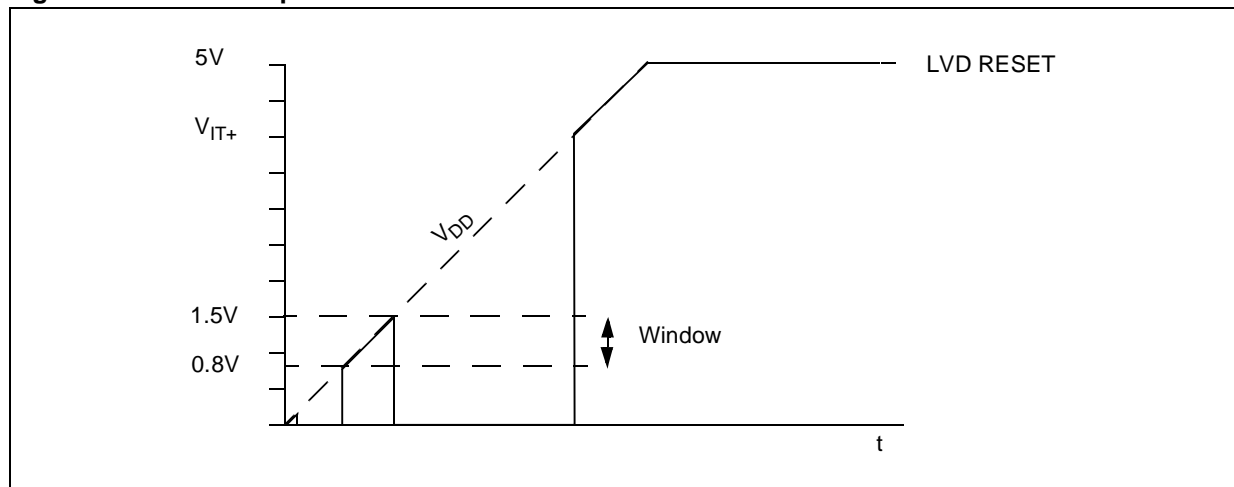
## 12.3.2 Operating Conditions with Low Voltage Detector (LVD)

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 <sup>1)</sup> | 4.2 | 4.5                | V         |
|                |   | VD level = Med. in option byte<br>VD level = Low in option byte | Not Applicable    |     |                    |           |
| $V_{IT-(LVD)}$ | Reset generation threshold ( $V_{DD}$ fall)     | VD level = High in option byte                                  | 3.8               | 4.0 | 4.25 <sup>1)</sup> | V         |
|                |   | VD level = Med. in option byte<br>VD level = Low in option byte | Not Applicable    |     |                    |           |
| $V_{hys(LVD)}$ | LVD voltage threshold hysteresis <sup>1)</sup>  | $V_{IT+(LVD)} - V_{IT-(LVD)}$                                   | 150               | 200 | 250                | mV        |
| $V_{tPOR}$     | $V_{DD}$ rise time <sup>1)2)</sup>              |   | 6                 |     | $\infty$           | $\mu s/V$ |
| $t_g(V_{DD})$  | Filtered glitch delay on $V_{DD}$ <sup>1)</sup> | Not detected by the LVD   |                   |     | 40                 | ns        |

**Notes:**

1. Data based on characterization results, not tested in production.
2. When  $V_{tPOR}$  is faster than 100  $\mu s/V$ , the Reset signal is released after a delay of max. 42 $\mu s$  after  $V_{DD}$  crosses the  $V_{IT+(LVD)}$  threshold.

**Figure 78. LVD Startup Behaviour**

**Note:** When the LVD is enabled, the MCU reaches its authorized operating voltage from a reset state. However, in some devices, the reset state is released when  $V_{DD}$  is approximately between 0.8V and 1.5V. As a consequence, the I/Os may toggle when  $V_{DD}$  is within this window.

This may be an issue especially for applications where the MCU drives power components.

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

**12.3.3 Auxiliary Voltage Detector (AVD) Thresholds**

Subject to general operating condition 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 <sup>1)</sup> | 4.6 | 4.9                | V    |
|                  |   | VD level = Med. in option byte<br>VD level = Low in option byte | Not Applicable    |     |                    |      |
| $V_{IT-(AVD)}$   | 0⇒1 AVDF flag toggle threshold ( $V_{DD}$ fall)           | VD level = High in option byte                                  | 4.2               | 4.4 | 4.65 <sup>1)</sup> |      |
|                  |   | VD level = Med. in option byte<br>VD level = Low in option byte | Not Applicable    |     |                    |      |
| $V_{hys(AVD)}$   | AVD voltage threshold hysteresis                          | $V_{IT+(AVD)} - V_{IT-(AVD)}$                                   |                   | 200 |                    | mV   |
| $\Delta V_{IT-}$ | 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.

**12.3.4 External Voltage Detector (EVD) Thresholds**

Subject to general operating condition 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) <sup>1)</sup> |                               | 1.15 | 1.26 | 1.35 | V    |
| $V_{IT-(EVD)}$ | 0⇒1 AVDF flag toggle threshold ( $V_{DD}$ fall) <sup>1)</sup> |                               | 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.

## 12.4 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).

### 12.4.1 RUN and SLOW Modes (Flash devices)

| Symbol          | Parameter  | Conditions   | Typ                      | Max <sup>1)</sup>         | Unit |
|-----------------|--|--|--------------------------|---------------------------|------|
| I <sub>DD</sub> | Supply current in RUN mode <sup>2)</sup><br>(see Figure 79)  | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =1MHz<br>f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =2MHz<br>f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =4MHz<br>f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =8MHz          | 1.3<br>2.0<br>3.6<br>7.1 | 3.0<br>5.0<br>8.0<br>15.0 | mA   |
|                 | Supply current in SLOW mode <sup>2)</sup><br>(see Figure 80) | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =62.5kHz<br>f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =125kHz<br>f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =250kHz<br>f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =500kHz | 0.6<br>0.7<br>0.8<br>1.1 | 2.7<br>3.0<br>3.6<br>4.0  |      |
|                 |  | 3.8V ≤ V <sub>DD</sub> ≤ 5.5V  |                          |                           |      |

Figure 79. Typical I<sub>DD</sub> in RUN vs. f<sub>CPU</sub>

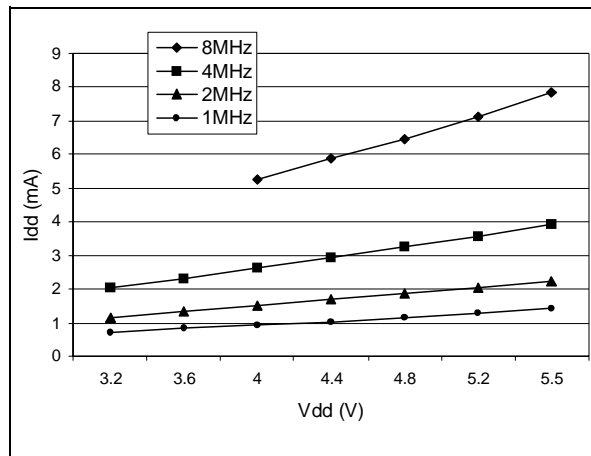
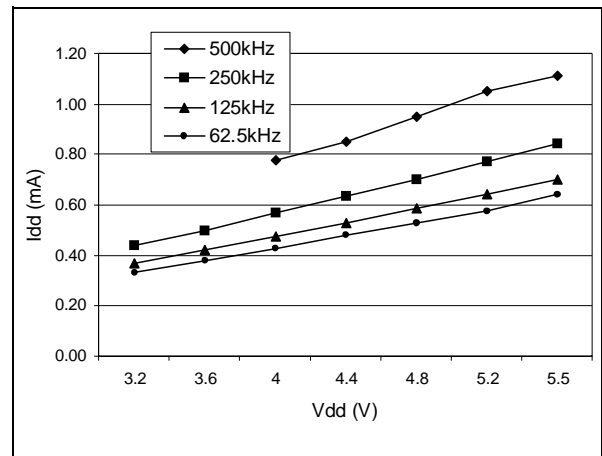


Figure 80. Typical I<sub>DD</sub> in SLOW vs. f<sub>CPU</sub>



#### Notes:

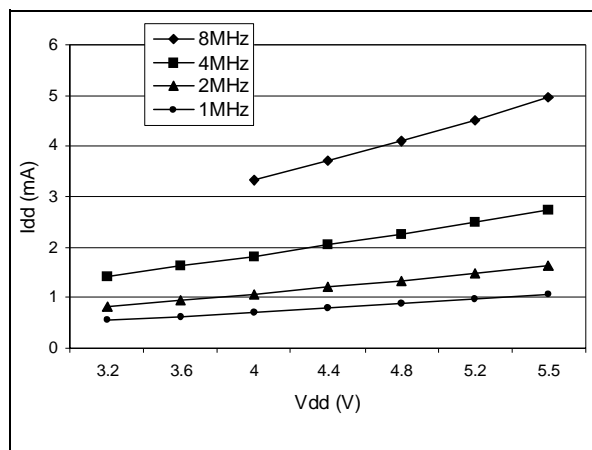
- Data based on characterization results, tested in production at V<sub>DD</sub> max. and f<sub>CPU</sub> max.
  - Measurements are done in the following conditions:
    - Program executed from RAM, CPU running with RAM access. The increase in consumption when executing from Flash is 50%.
    - All I/O pins in input mode with a static value at V<sub>DD</sub> or V<sub>SS</sub> (no load)
    - All peripherals in reset state.
    - CSS and LVD disabled.
    - Clock input (OSC1) driven by external square wave.
    - In SLOW and SLOW WAIT mode, f<sub>CPU</sub> is based on f<sub>OSC</sub> divided by 32.
- To obtain the total current consumption of the device, add the clock source (Section 12.5.3 and Section 12.5.4) and the peripheral power consumption (Section 12.4.7).

**SUPPLY CURRENT CHARACTERISTICS (Cont'd)**

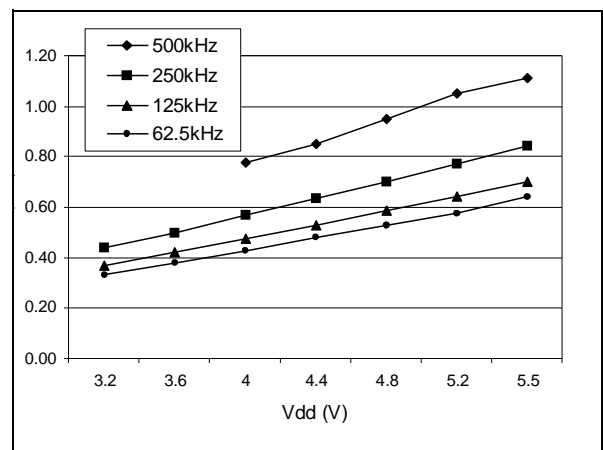
**12.4.2 WAIT and SLOW WAIT Modes (Flash devices)**

| Symbol          | Parameter   | Conditions                                      | Typ   | Max <sup>1)</sup> | Unit |    |
|-----------------|---|---|---|-------------------|------|----|
| I <sub>DD</sub> | Supply current in WAIT mode <sup>2)</sup><br>(see Figure 81)      | 3.8V ≤ V <sub>DD</sub> ≤ 5.5V                   | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =1MHz    | 1.0               | 3.0  | mA |
|                 |   |   | f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =2MHz    | 1.5               | 4.0  |    |
|                 |   | f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =4MHz  | 2.5   | 5.0               |      |    |
|                 |   | f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =8MHz | 4.5   | 7.0               |      |    |
|                 | Supply current in SLOW WAIT mode <sup>2)</sup><br>(see Figure 82) |   | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =62.5kHz | 0.58              | 1.2  | mA |
|                 |   |   | f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =125kHz  | 0.65              | 1.3  |    |
|                 |   |   | f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =250kHz  | 0.77              | 1.8  |    |
|                 |   |   | f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =500kHz | 1.05              | 2.0  |    |

**Figure 81. Typical I<sub>DD</sub> in WAIT vs. f<sub>CPU</sub>**



**Figure 82. Typical I<sub>DD</sub> in SLOW-WAIT vs. f<sub>CPU</sub>**



**Notes:**

- Data based on characterization results, tested in production at V<sub>DD</sub> max. and f<sub>CPU</sub> max.
  - 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<sub>DD</sub> or V<sub>SS</sub> (no load)
    - All peripherals in reset state.
    - CSS and LVD disabled.
    - Clock input (OSC1) driven by external square wave.
    - In SLOW and SLOW WAIT mode, f<sub>CPU</sub> is based on f<sub>OSC</sub> divided by 32.
- To obtain the total current consumption of the device, add the clock source (Section 12.5.3 and Section 12.5.4) and the peripheral power consumption (Section 12.4.7).



**SUPPLY CURRENT CHARACTERISTICS (Cont'd)****12.4.3 RUN and SLOW Modes (ROM devices)**

| Symbol          | Parameter                                 | Conditions  | Typ  | Max <sup>1)</sup> | Unit |    |
|-----------------|---|---|--|-------------------|------|----|
| I <sub>DD</sub> | Supply current in RUN mode <sup>2)</sup>  | 3.8V ≤ V <sub>DD</sub> ≤ 5.5V                     | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =1MHz | 1.3               | 3.0  | mA |
|                 | Supply current in SLOW mode <sup>2)</sup> |   | f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =2MHz | 2.0               | 5.0  |    |
|                 |   | f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =4MHz    | 3.6  | 8.0               |      |    |
|                 |   | f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =8MHz   | 7.1  | 15.0              |      |    |
|                 |   | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =62.5kHz | 0.6  | 2.7               |      |    |
|                 |   | f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =125kHz  | 0.7  | 3.0               |      |    |
|                 |   | f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =250kHz  | 0.8  | 3.6               |      |    |
|                 |   | f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =500kHz | 1.1  | 4.0               |      |    |

**12.4.4 WAIT and SLOW WAIT Modes (ROM devices)**

| Symbol          | Parameter                                      | Conditions  | Typ  | Max <sup>1)</sup> | Unit |    |
|-----------------|--|---|--|-------------------|------|----|
| I <sub>DD</sub> | Supply current in WAIT mode <sup>2)</sup>      | 3.8V ≤ V <sub>DD</sub> ≤ 5.5V                     | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =1MHz | 1.0               | 3.0  | mA |
|                 | Supply current in SLOW WAIT mode <sup>2)</sup> |   | f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =2MHz | 1.5               | 4.0  |    |
|                 |  | f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =4MHz    | 2.5  | 5.0               |      |    |
|                 |  | f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =8MHz   | 4.5  | 7.0               |      |    |
|                 |  | f <sub>OSC</sub> =2MHz, f <sub>CPU</sub> =62.5kHz | 0.07   | 1.2               |      |    |
|                 |  | f <sub>OSC</sub> =4MHz, f <sub>CPU</sub> =125kHz  | 0.1  | 1.3               |      |    |
|                 |  | f <sub>OSC</sub> =8MHz, f <sub>CPU</sub> =250kHz  | 0.2  | 1.8               |      |    |
|                 |  | f <sub>OSC</sub> =16MHz, f <sub>CPU</sub> =500kHz | 0.35   | 2.0               |      |    |

**Notes:**

- Data based on characterization results, tested in production at V<sub>DD</sub> max. and f<sub>CPU</sub> max.
  - Measurements are done in the following conditions:
    - Program executed from RAM, CPU running with RAM access. There is no increase in consumption for if programs are executed in ROM
    - All I/O pins in input mode with a static value at V<sub>DD</sub> or V<sub>SS</sub> (no load)
    - All peripherals in reset state.
    - CSS and LVD disabled.
    - Clock input (OSC1) driven by external square wave.
    - In SLOW and SLOW WAIT mode, f<sub>CPU</sub> is based on f<sub>OSC</sub> divided by 32.
- To obtain the total current consumption of the device, add the clock source (Section 12.5.3 and Section 12.5.4) and the peripheral power consumption (Section 12.4.7).

**SUPPLY CURRENT CHARACTERISTICS (Cont'd)**

**12.4.5 HALT and ACTIVE-HALT Modes**

| Symbol          | Parameter  | Conditions                                      | Typ | Max                             | Unit |    |
|-----------------|--|---|-----|---------------------------------|------|----|
| I <sub>DD</sub> | Supply current in HALT mode <sup>1)</sup>        | V <sub>DD</sub> =5.5V                           |     | -40°C ≤ T <sub>A</sub> ≤ +85°C  | 10   | μA |
|                 |  |   |     | -40°C ≤ T <sub>A</sub> ≤ +125°C | 50   |    |
| I <sub>DD</sub> | Supply current in ACTIVE-HALT mode <sup>2)</sup> | f <sub>OSC</sub> = 16 MHz, V <sub>DD</sub> = 5V | 650 | No max. guaranteed              | μA   |    |

**Notes:**

1. All I/O pins in push-pull 0 mode (when applicable) with a static value at V<sub>DD</sub> or V<sub>SS</sub> (no load), CSS and LVD disabled. Data based on characterization results, tested in production at V<sub>DD</sub> max. and f<sub>CPU</sub> max.
2. 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<sub>DD</sub> or V<sub>SS</sub> (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 12.5.3 and Section 12.5.4).

**12.4.6 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 <sup>1)</sup> | Unit |
|------------------------|--|---------------------------------|--------------------------------|-------------------|------|
| I <sub>DD(RCINT)</sub> | Supply current of internal RC oscillator                             |                                 | 625                            |                   | μA   |
| I <sub>DD(RCEXT)</sub> | Supply current of external RC oscillator <sup>2)</sup>               |                                 | see Section 12.5.4 on page 167 |                   |      |
| I <sub>DD(RES)</sub>   | Supply current of resonator oscillator <sup>2)</sup> & <sup>3)</sup> |                                 | see Section 12.5.3 on page 165 |                   |      |
| I <sub>DD(PLL)</sub>   | PLL supply current   | V <sub>DD</sub> = 5V            | 360                            |                   |      |
| I <sub>DD(CSS)</sub>   | Clock security system supply current                                 | V <sub>DD</sub> = 5V            | 250                            |                   |      |
| I <sub>DD(LVD)</sub>   | LVD supply current   | HALT mode, V <sub>DD</sub> = 5V | 150                            | 300               |      |

**Notes:**

1. Data based on characterisation results, not tested in production.
2. Data based on characterization results done with the external components specified in Section 12.5.3 and Section 12.5.4, not tested in production.
3. As the oscillator is based on a current source, the consumption does not depend on the voltage.

**SUPPLY CURRENT CHARACTERISTICS (Cont'd)****12.4.7 On-Chip Peripherals**

Measured on S72F521R9T3 on TQFP64 generic board  $T_A = 25^\circ\text{C}$   $f_{\text{CPU}}=4\text{MHz}$ .

| Symbol               | Parameter  | Conditions                  | Typ | Unit          |
|----------------------|--|-----------------------------|-----|---------------|
| $I_{\text{DD(TIM)}}$ | 16-bit Timer supply current <sup>1)</sup>        | $V_{\text{DD}}=5.0\text{V}$ | 50  | $\mu\text{A}$ |
| $I_{\text{DD(ART)}}$ | ART PWM supply current <sup>2)</sup>             | $V_{\text{DD}}=5.0\text{V}$ | 75  |               |
| $I_{\text{DD(SPI)}}$ | SPI supply current <sup>3)</sup>                 | $V_{\text{DD}}=5.0\text{V}$ | 400 | $\mu\text{A}$ |
| $I_{\text{DD(I2C)}}$ | I2C supply current <sup>4)</sup>                 | $V_{\text{DD}}=5.0\text{V}$ | 175 |               |
| $I_{\text{DD(CAN)}}$ | CAN supply current <sup>5)</sup>                 | $V_{\text{DD}}=5.0\text{V}$ | 400 | $\mu\text{A}$ |
| $I_{\text{DD(ADC)}}$ | ADC supply current when converting <sup>6)</sup> | $V_{\text{DD}}=5.0\text{V}$ | 400 | $\mu\text{A}$ |

**Notes:**

1. Data based on a differential  $I_{\text{DD}}$  measurement between reset configuration (timer counter running at  $f_{\text{CPU}}/4$ ) and timer counter stopped (only TIMD bit set). Data valid for one timer.
2. Data based on a differential  $I_{\text{DD}}$  measurement between reset configuration (timer stopped) and timer counter enabled (only TCE bit set).
3. Data based on a differential  $I_{\text{DD}}$  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  $I_{\text{DD}}$  measurement between reset configuration (I2C disabled) and a permanent I2C master communication at 100kHz (data sent equal to 55h). This measurement include the pad toggling consumption (27kOhm external pull-up on clock and data lines).
5. Data based on a differential  $I_{\text{DD}}$  measurement between reset configuration (CAN disabled) and a permanent CAN data transmit sequence with RX and TX connected together. This measurement include the pad toggling consumption.
6. Data based on a differential  $I_{\text{DD}}$  measurement between reset configuration and continuous A/D conversions.

12.5 CLOCK AND TIMING CHARACTERISTICS

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

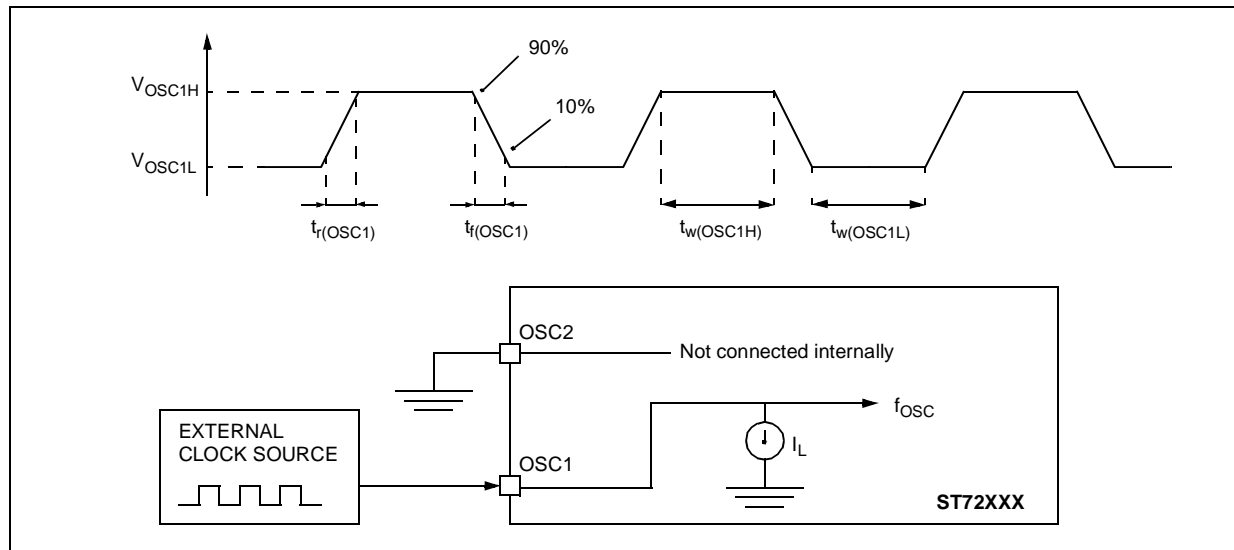
12.5.1 General Timings

| Symbol        | Parameter  | Conditions     | Min  | Typ <sup>1)</sup> | Max  | Unit      |
|---------------|--|----------------|------|-------------------|------|-----------|
| $t_{c(INST)}$ | Instruction cycle time   |                | 2    | 3                 | 12   | $t_{CPU}$ |
|               |  | $f_{CPU}=8MHz$ | 250  | 375               | 1500 | ns        |
| $t_{v(IT)}$   | Interrupt reaction time <sup>2)</sup><br>$t_{v(IT)} = \Delta t_{c(INST)} + 10$ |                | 10   |                   | 22   | $t_{CPU}$ |
|               |  | $f_{CPU}=8MHz$ | 1.25 |                   | 2.75 | $\mu s$   |

12.5.2 External Clock Source

| Symbol                           | Parameter                            | Conditions                       | Min        | Typ | Max        | Unit    |
|----------------------------------|--------------------------------------|----------------------------------|------------|-----|------------|---------|
| $V_{OSC1H}$                      | OSC1 input pin high level voltage    | see Figure 83                    | $V_{DD}-1$ |     | $V_{DD}$   | V       |
| $V_{OSC1L}$                      | OSC1 input pin low level voltage     |                                  | $V_{SS}$   |     | $V_{SS}+1$ |         |
| $t_{w(OSC1H)}$<br>$t_{w(OSC1L)}$ | OSC1 high or low time <sup>3)</sup>  |                                  | 5          |     |            | ns      |
| $t_r(OSC1)$<br>$t_f(OSC1)$       | OSC1 rise or fall time <sup>3)</sup> |                                  |            |     | 15         |         |
| $I_L$                            | OSCx Input leakage current           | $V_{SS} \leq V_{IN} \leq V_{DD}$ |            |     | $\pm 1$    | $\mu A$ |

Figure 83. Typical Application with an External Clock Source



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)**

**12.5.3 Crystal and Ceramic Resonator Oscillators**

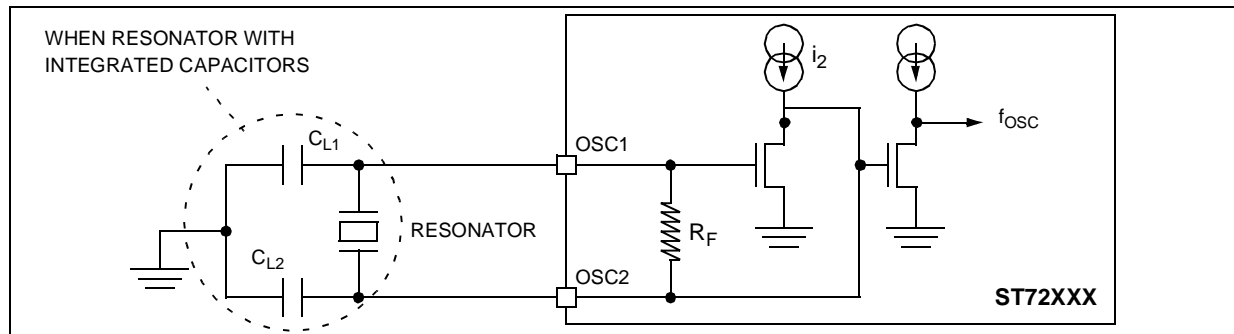
The ST7 internal clock can be supplied with four different Crystal/Ceramic resonator oscillators. All the information given in this paragraph are based on characterization results with specified typical external components. In the application, the resonator and the load capacitors have to be placed as

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 <sup>1)</sup>   | LP: Low power oscillator<br>MP: Medium power oscillator<br>MS: Medium speed oscillator<br>HS: High speed oscillator              | 1<br>>2<br>>4<br>>8  | 2<br>4<br>8<br>16    | MHz        |
| $R_F$                | Feedback resistor  |  | 20                   | 40                   | k $\Omega$ |
| $C_{L1}$<br>$C_{L2}$ | Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator ( $R_S$ ) | $R_S=200\Omega$ LP oscillator<br>$R_S=200\Omega$ MP oscillator<br>$R_S=200\Omega$ MS oscillator<br>$R_S=100\Omega$ HS oscillator | 22<br>22<br>18<br>15 | 56<br>46<br>33<br>33 | pF         |

| Symbol | Parameter            | Conditions   | Typ                     | Max                      | Unit    |
|--------|----------------------|--|-------------------------|--------------------------|---------|
| $i_2$  | OSC2 driving current | $V_{DD}=5V$<br>$V_{IN}=V_{SS}$<br>LP oscillator<br>MP oscillator<br>MS oscillator<br>HS oscillator | 80<br>160<br>310<br>610 | 150<br>250<br>460<br>910 | $\mu A$ |

**Figure 84. Typical Application with a Crystal or Ceramic Resonator**



**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.

**CLOCK AND TIMING CHARACTERISTICS (Cont'd)**

| Oscil.  |    | Typical Crystal or Ceramic Resonators |                |                              |  | C <sub>L1</sub><br>[pF] | C <sub>L2</sub><br>[pF] | t <sub>SU(osc)</sub><br>[ms] <sup>2)</sup> |
|---------|----|---------------------------------------|----------------|------------------------------|--|-------------------------|-------------------------|--|
|         |    | Reference <sup>3)</sup>               | Freq.          | Characteristic <sup>1)</sup> |  |                         |                         |  |
| Ceramic | LP | MURATA                                | CSA2.00MG      | 2MHz                         | $\Delta f_{OSC} = [\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, \pm X.X\%_{\text{correl}}]$ | 22                      | 22                      | 4  |
|         | MP |                                       | CSA4.00MG      | 4MHz                         | $\Delta f_{OSC} = [\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, \pm X.X\%_{\text{correl}}]$ | 22                      | 22                      | 2  |
|         | MS |                                       | CSA8.00MTZ     | 8MHz                         | $\Delta f_{OSC} = [\pm 0.5\%_{\text{tolerance}}, \pm 0.5\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, \pm X.X\%_{\text{correl}}]$ | 33                      | 33                      | 1  |
|         | HS |                                       | CSA16.00MXZ040 | 16MHz                        | $\Delta f_{OSC} = [\pm 0.5\%_{\text{tolerance}}, \pm 0.3\%_{\Delta T_a}, \pm 0.3\%_{\text{aging}}, \pm X.X\%_{\text{correl}}]$ | 33                      | 33                      | 0.7  |

**Notes:**

1. Resonator characteristics given by the crystal/ceramic resonator manufacturer.
2. t<sub>SU(OSC)</sub> is the typical oscillator start-up time measured between V<sub>DD</sub>=2.8V and the fetch of the first instruction (with a quick V<sub>DD</sub> ramp-up from 0 to 5V (<50μs)).
3. Contact the supplier for updated product information.

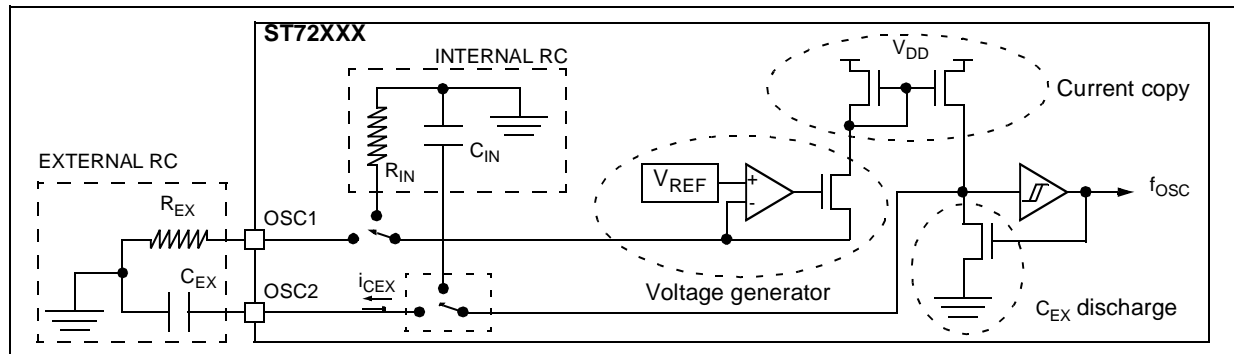
**CLOCK CHARACTERISTICS (Cont'd)**

**12.5.4 RC Oscillators**

The ST7 internal clock can be supplied with an RC oscillator. This oscillator can be used with internal or external components (selectable by option byte).

| Symbol           | Parameter   | Conditions                                  | Min                         | Typ | Max | Unit          |
|------------------|---|---|-----------------------------|-----|-----|---------------|
| $f_{OSC(RCINT)}$ | Internal RC oscillator frequency<br>See Figure 86 and Figure 87 | $T_A=25^\circ\text{C}$ , $V_{DD}=5\text{V}$ | 2                           | 3.5 | 5.6 | MHz           |
| $f_{OSC(RCEXT)}$ | External RC oscillator frequency <sup>1)</sup>                  |   | $5 / (R_{EX} \cdot C_{EX})$ |     |     |               |
| $R_{EX}$         | Oscillator external resistor <sup>2)</sup>                      |   | 56                          |     | 100 | K $\Omega$    |
| $C_{EX}$         | Oscillator external capacitor                                   |   | 22                          |     | 470 | pF            |
| $ i_{CEX} $      | Capacitor load current <sup>3)</sup>                            | $OSC1 = V_{SS}$ or 1.5V                     |                             | 290 | 350 | $\mu\text{A}$ |

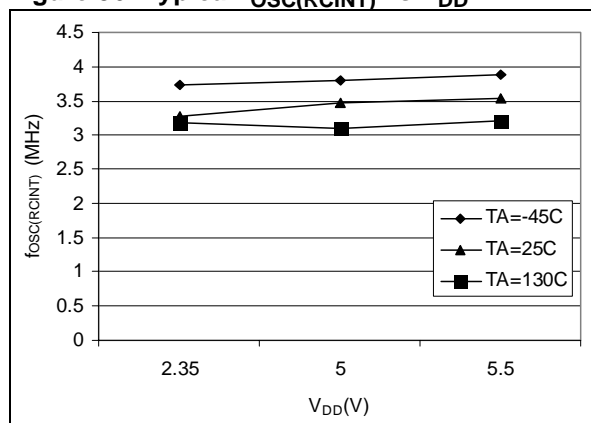
**Figure 85. Typical Application with RC oscillator**



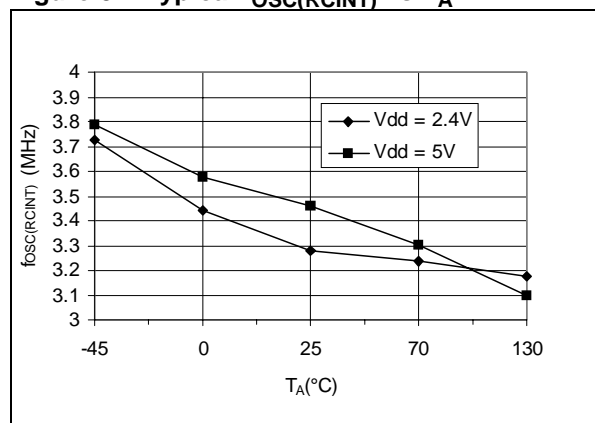
**Notes:**

1. Data based on design simulation.
2.  $R_{EX}$  must have a positive temperature coefficient (ppm/ $^\circ\text{C}$ ), carbon resistors should therefore not be used.
3.  $i_{CEX}$  is the current needed to load the  $C_{EX}$  capacitor while OSC1 is forced to  $V_{SS}$  or 1.5V (RC oscillation voltage range).

**Figure 86. Typical  $f_{OSC(RCINT)}$  vs  $V_{DD}$**



**Figure 87. Typical  $f_{OSC(RCINT)}$  vs  $T_A$**



**CLOCK CHARACTERISTICS (Cont'd)**

**12.5.5 Clock Security System (CSS)**

| Symbol             | Parameter                               | Conditions | Min | Typ | Max | Unit |
|--------------------|---|------------|-----|-----|-----|------|
| f <sub>SFOSC</sub> | Safe Oscillator Frequency <sup>1)</sup> |            |     | 3   |     | MHz  |

**Note:**

1. Data based on characterization results.

**12.5.6 PLL Characteristics**

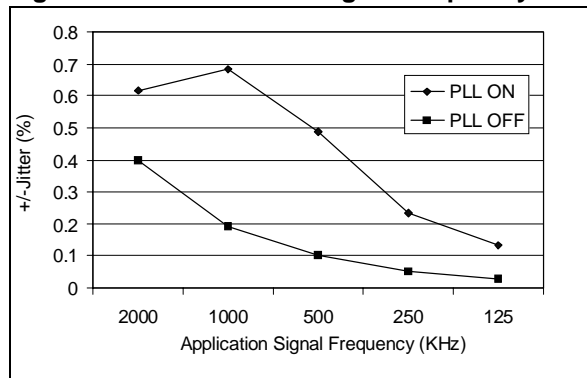
Operating conditions: V<sub>DD</sub> 3.8 to 5.5V @ T<sub>A</sub> 0 to 70°C<sup>1)</sup> or V<sub>DD</sub> 4.5 to 5.5V @ T<sub>A</sub> -40 to 125°C

| Symbol                               | Parameter                              | Conditions                   | Min | Typ | Max | Unit |
|--------------------------------------|--|------------------------------|-----|-----|-----|------|
| V <sub>DD(PLL)</sub>                 | PLL Operating Range                    | T <sub>A</sub> 0 to 70°C     | 3.8 |     | 5.5 | V    |
|                                      |  | T <sub>A</sub> -40 to +125°C | 4.5 |     | 5.5 |      |
| f <sub>OSC</sub>                     | PLL input frequency range              |                              | 2   |     | 4   | MHz  |
| f <sub>CPU</sub> /Δ f <sub>CPU</sub> | Instantaneous PLL jitter <sup>1)</sup> | f <sub>OSC</sub> = 4 MHz.    |     | 1.0 | 2.5 | %    |
|                                      |  | f <sub>OSC</sub> = 2 MHz.    |     | 2.5 | 4.0 | %    |

**Note:**

1. Data characterized but not tested.

**Figure 88. PLL Jitter vs. Signal frequency<sup>1</sup>**



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 88 shows the PLL jitter integrated on application signals in the range 125kHz to 2MHz. At frequencies of less than 125KHz, the jitter is negligible.

**Note 1:** Measurement conditions: f<sub>CPU</sub> = 4MHz, T<sub>A</sub> = 25°C



## 12.6 MEMORY CHARACTERISTICS

### 12.6.1 RAM and Hardware Registers

| Symbol   | Parameter                         | Conditions           | Min | Typ | Max | Unit |
|----------|-----------------------------------|----------------------|-----|-----|-----|------|
| $V_{RM}$ | Data retention mode <sup>1)</sup> | HALT mode (or RESET) | 1.6 |     |     | V    |

### 12.6.2 FLASH Memory

| DUAL VOLTAGE HDFLASH MEMORY |  |                               |                   |     |                   |            |
|-----------------------------|--|-------------------------------|-------------------|-----|-------------------|------------|
| Symbol                      | Parameter                                | Conditions                    | Min <sup>2)</sup> | Typ | Max <sup>2)</sup> | Unit       |
| $f_{CPU}$                   | Operating frequency                      | Read mode                     | 0                 |     | 8                 | MHz        |
|                             |  | Write / Erase mode            | 1                 |     | 8                 |            |
| $V_{PP}$                    | Programming voltage <sup>3)</sup>        | $4.5V \leq V_{DD} \leq 5.5V$  | 11.4              |     | 12.6              | V          |
| $I_{DD}$                    | Supply current <sup>4)</sup>             | RUN mode ( $f_{CPU} = 4MHz$ ) |                   |     | 3                 | mA         |
|                             |  | Write / Erase                 |                   | 0   |                   |            |
|                             |  | Power down mode / HALT        |                   | 1   | 10                | $\mu A$    |
| $I_{PP}$                    | $V_{PP}$ current <sup>4)</sup>           | Read ( $V_{PP}=12V$ )         |                   |     | 200               | mA         |
|                             |  | Write / Erase                 |                   |     | 30                |            |
| $t_{VPP}$                   | Internal $V_{PP}$ stabilization time     |                               |                   | 10  |                   | $\mu s$    |
| $t_{RET}$                   | Data retention                           | $T_A=55^\circ C$              | 20                |     |                   | years      |
| $N_{RW}$                    | Write erase cycles                       | $T_A=25^\circ C$              | 100               |     |                   | cycles     |
| $T_{PROG}$<br>$T_{ERASE}$   | Programming or erasing temperature range |                               | -40               | 25  | 85                | $^\circ C$ |

#### 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.

**12.7 EMC CHARACTERISTICS**

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

**12.7.1 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.

| Symbol     | Parameter  | Conditions   | Neg <sup>1)</sup> | Pos <sup>1)</sup> | Unit |
|------------|--|--|-------------------|-------------------|------|
| $V_{FESD}$ | Voltage limits to be applied on any I/O pin to induce a functional disturbance   | $V_{DD}=5V, T_A=+25^{\circ}C, f_{OSC}=8MHz$ conforms to IEC 1000-4-2 | -1                | 1.5               | kV   |
| $V_{FFTB}$ | Fast transient voltage burst limits to be applied through 100pF on $V_{DD}$ and $V_{DD}$ pins to induce a functional disturbance | $V_{DD}=5V, T_A=+25^{\circ}C, f_{OSC}=8MHz$ conforms to IEC 1000-4-4 | -1.5              | 1.5               |      |

**12.7.2 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. [ $f_{OSC}/f_{CPU}$ ] |         | Unit       |
|-----------|------------|--|--------------------------|-------------------------------|---------|------------|
|           |            |  |                          | 8/4MHz                        | 16/8MHz |            |
| $S_{EMI}$ | Peak level | $V_{DD}=5V, T_A=+25^{\circ}C,$<br>TQFP64 14x14 package<br>conforming to SAE J 1752/3 | 0.1MHz to 30MHz          | 13                            | 13      | dB $\mu$ V |
|           |            |  | 30MHz to 130MHz          | 19                            | 24      |            |
|           |            |  | 130MHz to 1GHz           | 7                             | 13      |            |
|           |            |  | SAE EMI Level            | 2.0                           | 2.5     | -          |

**Notes:**

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

**EMC CHARACTERISTICS (Cont'd)**

**12.7.3 Absolute Electrical Sensitivity**

Based on three different tests (ESD, LU and DLU) 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 AN1181 ST7 application note.

**12.7.3.1 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 of the number of supply pins of the device (3 parts\*(n+1) supply pin). Two models are usually simulated: Human Body Model and Machine Model. This test conforms to the JESD22-A114A/A115A standard. See Figure 89 and the following test sequences.

**Human Body Model Test Sequence**

- $C_L$  is loaded through S1 by the HV pulse generator.
- S1 switches position from generator to R.
- A discharge from  $C_L$  through R (body resistance) to the ST7 occurs.
- S2 must be closed 10 to 100ms after the pulse delivery period to ensure the ST7 is not left in charge state. S2 must be opened at least 10ms prior to the delivery of the next pulse.

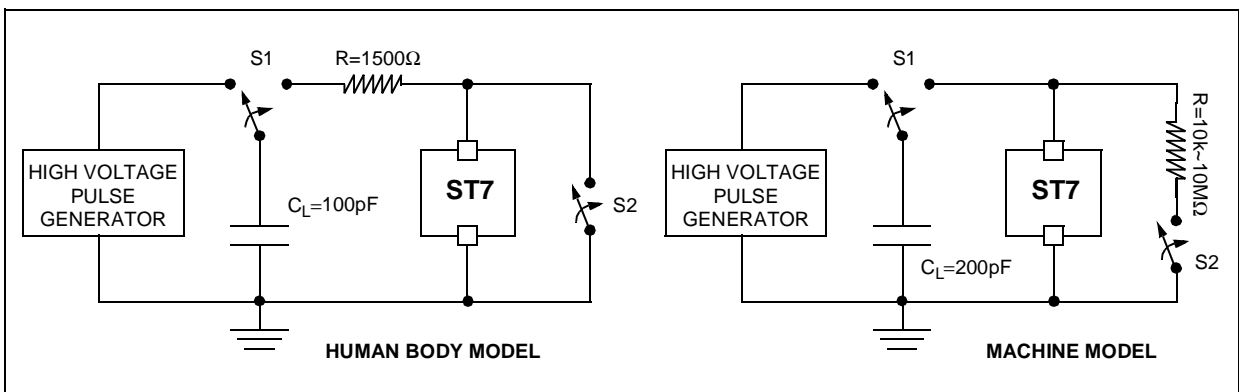
**Machine Model Test Sequence**

- $C_L$  is loaded through S1 by the HV pulse generator.
- S1 switches position from generator to ST7.
- A discharge from  $C_L$  to the ST7 occurs.
- S2 must be closed 10 to 100ms after the pulse delivery period to ensure the ST7 is not left in charge state. S2 must be opened at least 10ms prior to the delivery of the next pulse.
- R (machine resistance), in series with S2, ensures a slow discharge of the ST7.

**Absolute Maximum Ratings**

| Symbol         | Ratings   | Conditions         | Maximum value <sup>1)</sup> | Unit |
|----------------|---|--------------------|-----------------------------|------|
| $V_{ESD(HBM)}$ | Electro-static discharge voltage (Human Body Model) | $T_A=+25^{\circ}C$ | 2000                        | V    |
| $V_{ESD(MM)}$  | Electro-static discharge voltage (Machine Model)    | $T_A=+25^{\circ}C$ | 200                         |      |

**Figure 89. Typical Equivalent ESD Circuits**



**Notes:**

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

**EMC CHARACTERISTICS (Cont'd)**

**12.7.3.2 Static and Dynamic Latch-Up**

- **LU:** 3 complementary static tests are required on 10 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. For more details, refer to the AN1181 ST7 application note.
- **DLU:** Electro-Static Discharges (one positive then one negative test) are applied to each pin of 3 samples when the micro is running to assess the latch-up performance in dynamic mode. Power supplies are set to the typical values, the oscillator is connected as near as possible to the pins of the micro and the component is put in reset mode. This test conforms to the IEC1000-4-2 and SAEJ1752/3 standards and is described in Figure 90. For more details, refer to the AN1181 ST7 application note.

**12.7.3.3 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:**

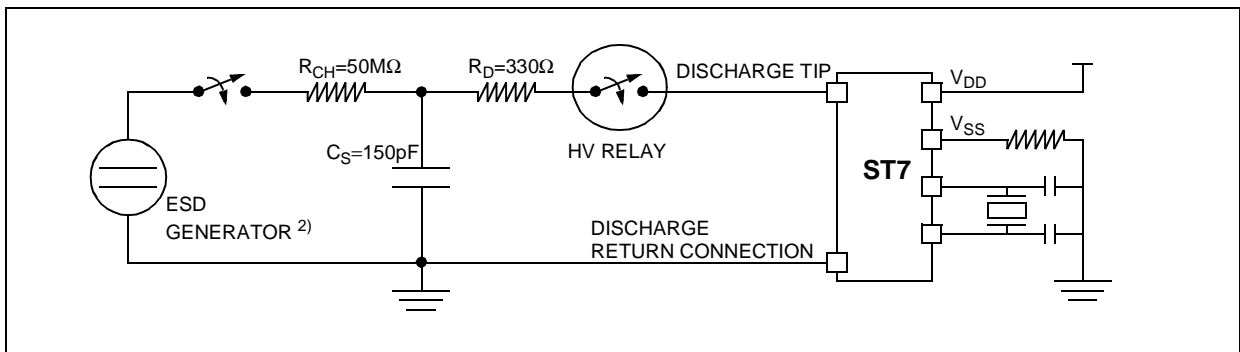
Most of the common failures (unexpected reset and program counter corruption) can be reproduced by manually forcing a low state on the RESET pin or the Oscillator pins for 1 second.

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).

**Electrical Sensitivities**

| Symbol | Parameter              | Conditions   | Class <sup>1)</sup> |
|--------|------------------------|--|---------------------|
| LU     | Static latch-up class  | T <sub>A</sub> =+25°C<br>T <sub>A</sub> =+85°C<br>T <sub>A</sub> =+125°C | A<br>A<br>A         |
| DLU    | Dynamic latch-up class | V <sub>DD</sub> =5.5V, f <sub>OSC</sub> =4MHz, T <sub>A</sub> =+25°C     | A                   |

**Figure 90. Simplified Diagram of the ESD Generator for DLU**



**Notes:**

1. Class description: A Class is an STMicroelectronics internal specification. All its limits are higher than the JEDEC specifications, that means when a device belongs to Class A it exceeds the JEDEC standard. B Class strictly covers all the JEDEC criteria (international standard).

2. Schaffner NSG435 with a pointed test finger.

## EMC CHARACTERISTICS (Cont'd)

## 12.7.4 ESD Pin Protection Strategy

To protect an integrated circuit against Electro-Static Discharge the stress must be controlled to prevent degradation or destruction of the circuit elements. The stress generally affects the circuit elements which are connected to the pads but can also affect the internal devices when the supply pads receive the stress. The elements to be protected must not receive excessive current, voltage or heating within their structure.

An ESD network combines the different input and output ESD protections. This network works, by allowing safe discharge paths for the pins subjected to ESD stress. Two critical ESD stress cases are presented in Figure 91 and Figure 92 for standard pins and in Figure 93 and Figure 94 for true open drain pins.

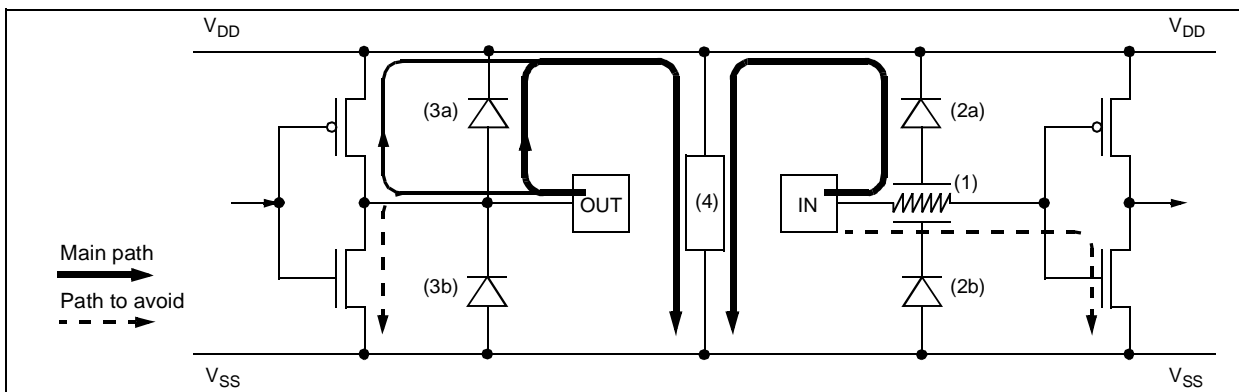
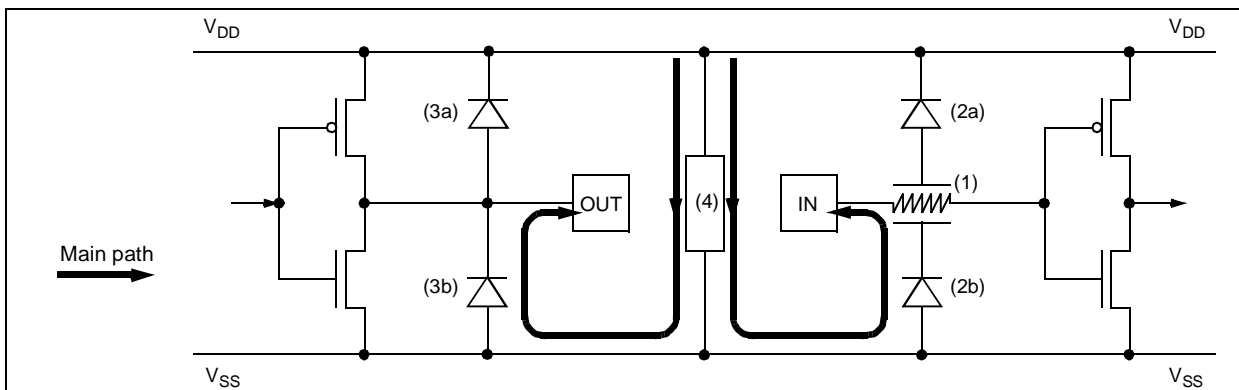
## Standard Pin Protection

To protect the output structure the following elements are added:

- A diode to  $V_{DD}$  (3a) and a diode from  $V_{SS}$  (3b)
- A protection device between  $V_{DD}$  and  $V_{SS}$  (4)

To protect the input structure the following elements are added:

- A resistor in series with the pad (1)
- A diode to  $V_{DD}$  (2a) and a diode from  $V_{SS}$  (2b)
- A protection device between  $V_{DD}$  and  $V_{SS}$  (4)

Figure 91. Positive Stress on a Standard Pad vs.  $V_{SS}$ Figure 92. Negative Stress on a Standard Pad vs.  $V_{DD}$ 



12.8 I/O PORT PIN CHARACTERISTICS

12.8.1 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 <sup>1)</sup>                    | CMOS ports                        |                     |     | $0.3 \times V_{DD}$ | V          |
| $V_{IH}$                   | Input high level voltage <sup>1)</sup>                   |                                   | $0.7 \times V_{DD}$ |     |                     |            |
| $V_{hys}$                  | Schmitt trigger voltage hysteresis <sup>2)</sup>         |                                   |                     | 0.7 |                     |            |
| $V_{IL}$                   | Input low level voltage <sup>1)</sup>                    | TTL ports                         |                     |     | 0.8                 |            |
| $V_{IH}$                   | Input high level voltage <sup>1)</sup>                   |                                   | 2                   |     |                     |            |
| $V_{hys}$                  | Schmitt trigger voltage hysteresis <sup>2)</sup>         |                                   |                     | 1   |                     |            |
| $I_{INJ(PIN)}^{3)}$        | Injected Current on an I/O pin                           | $V_{DD}=5V$                       |                     |     | $\pm 4$             | mA         |
| $\Sigma I_{INJ(PIN)}^{3)}$ | Total injected current (sum of all I/O and control pins) |                                   |                     |     | $\pm 25$            |            |
| $I_L$                      | Input leakage current                                    | $V_{SS} \leq V_{IN} \leq V_{DD}$  |                     |     | $\pm 1$             | $\mu A$    |
| $I_S$                      | Static current consumption                               | Floating input mode <sup>4)</sup> |                     |     | 200                 |            |
| $R_{PU}$                   | Weak pull-up equivalent resistor <sup>5)</sup>           | $V_{IN}=V_{SS}$   $V_{DD}=5V$     | 50                  | 120 | 250                 | k $\Omega$ |
| $C_{IO}$                   | I/O pin capacitance                                      |                                   |                     | 5   |                     | pF         |
| $t_{f(I/O)out}$            | Output high to low level fall time <sup>1)</sup>         | $C_L=50pF$<br>Between 10% and 90% |                     | 25  |                     | ns         |
| $t_{r(I/O)out}$            | Output low to high level rise time <sup>1)</sup>         |                                   |                     | 25  |                     |            |
| $t_w(IT)in$                | External interrupt pulse time <sup>6)</sup>              |                                   | 1                   |     |                     | $t_{CPU}$  |

Figure 96. Connecting Unused I/O Pins

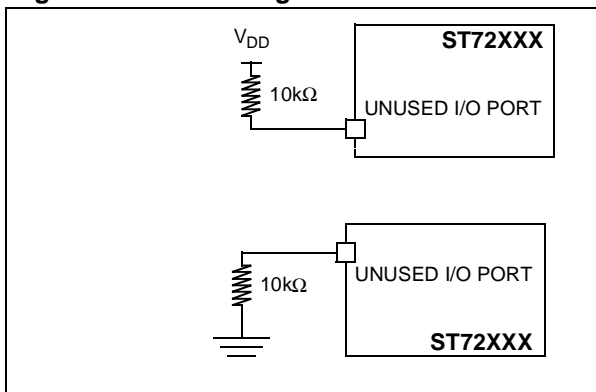
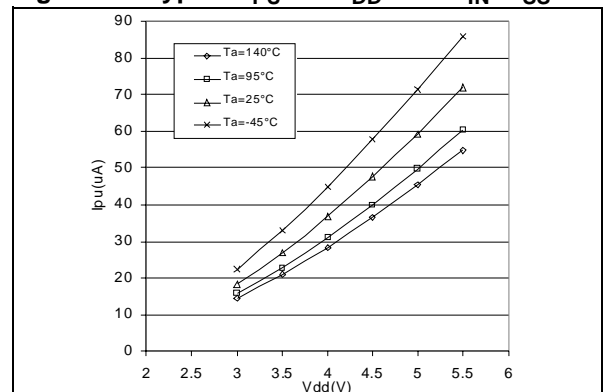


Figure 97. Typical  $I_{PU}$  vs.  $V_{DD}$  with  $V_{IN}=V_{SS}$



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 12.2.2 on page 155 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 or an external pull-up or pull-down resistor (see Figure 96). 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 97).
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)

12.8.2 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 98)             | $I_{IO}=+5mA$   |                              | 1.2 | V    |
|               |   | $I_{IO}=+2mA$   |                              | 0.5 |      |
|               | $I_{IO}=+20mA, T_A \leq 85^\circ C$<br>$T_A \geq 85^\circ C$  |   | 1.3                          |     |      |
|               |   |   | 1.5                          |     |      |
| $V_{OH}^{2)}$ | Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 100 and Figure 103) | $I_{IO}=-5mA, T_A \leq 85^\circ C$<br>$T_A \geq 85^\circ C$ | $V_{DD}-1.4$<br>$V_{DD}-1.6$ |     |      |
|               |   | $I_{IO}=-2mA$   | $V_{DD}-0.7$                 |     |      |

Figure 98. Typical  $V_{OL}$  at  $V_{DD}=5V$  (standard)

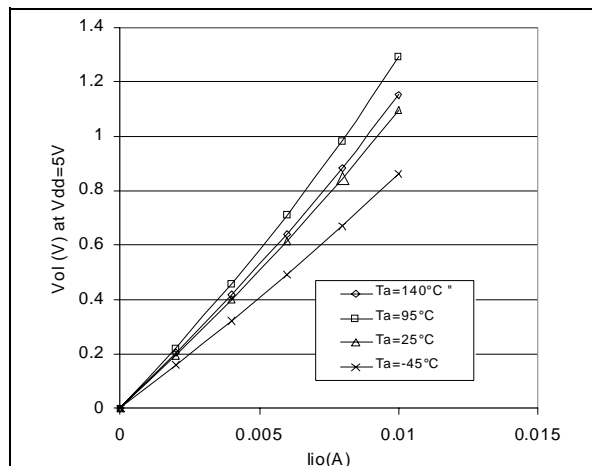


Figure 100. Typical  $V_{OH}$  at  $V_{DD}=5V$

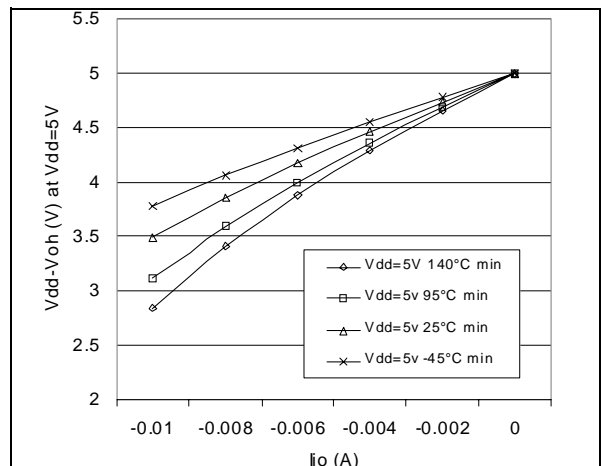
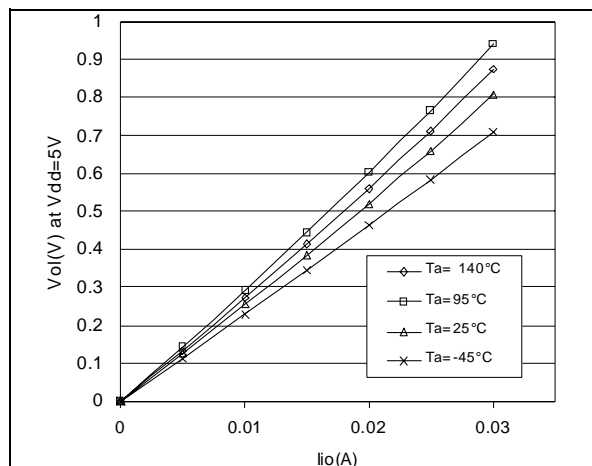


Figure 99. Typical  $V_{OL}$  at  $V_{DD}=5V$  (high-sink)



Notes:

1. The  $I_{IO}$  current sunk must always respect the absolute maximum rating specified in Section 12.2.2 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 12.2.2 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 101. Typical  $V_{OL}$  vs.  $V_{DD}$  (standard)

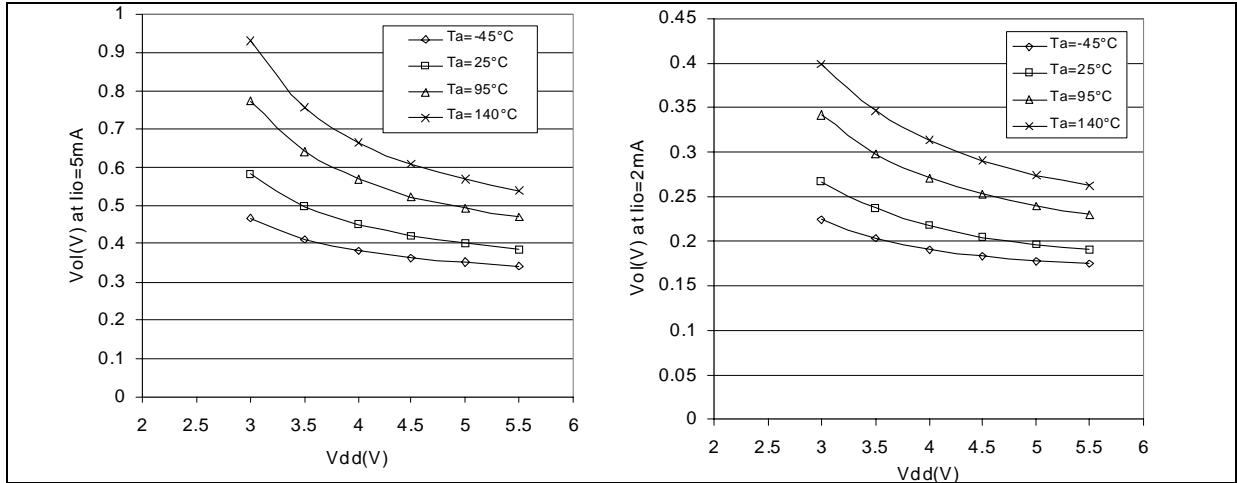


Figure 102. Typical  $V_{OL}$  vs.  $V_{DD}$  (high-sink)

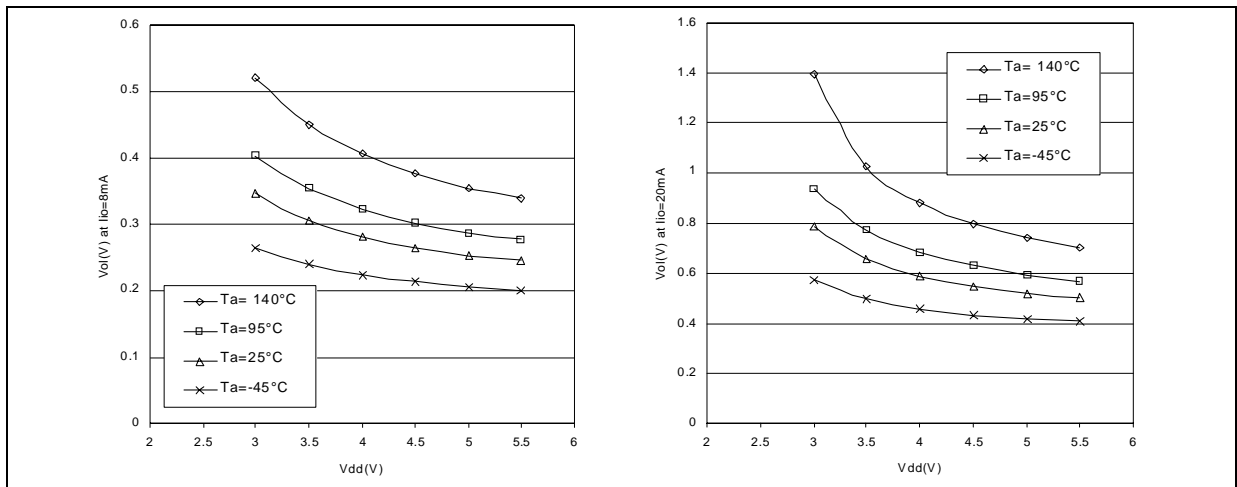
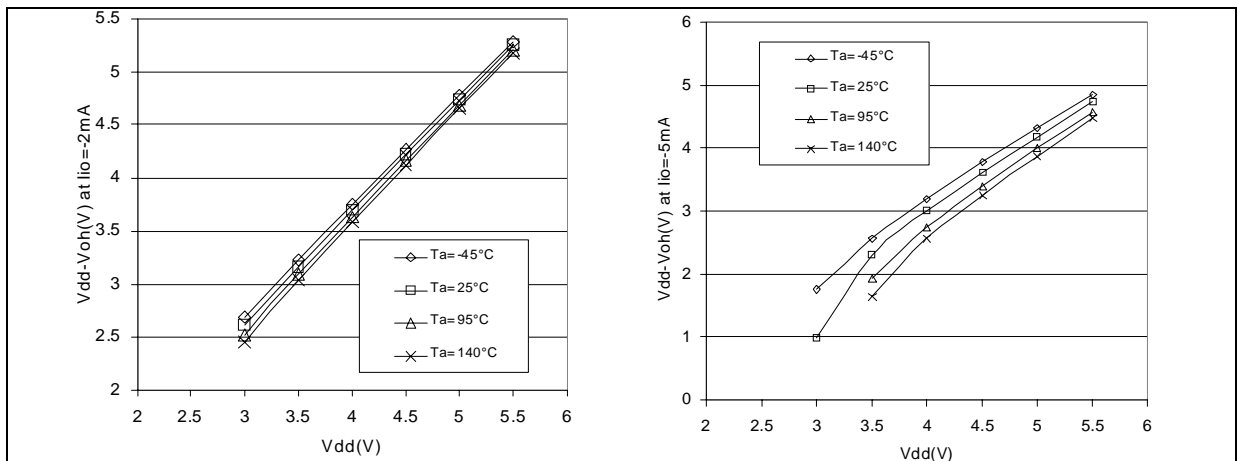


Figure 103. Typical  $V_{DD} - V_{OH}$  vs.  $V_{DD}$



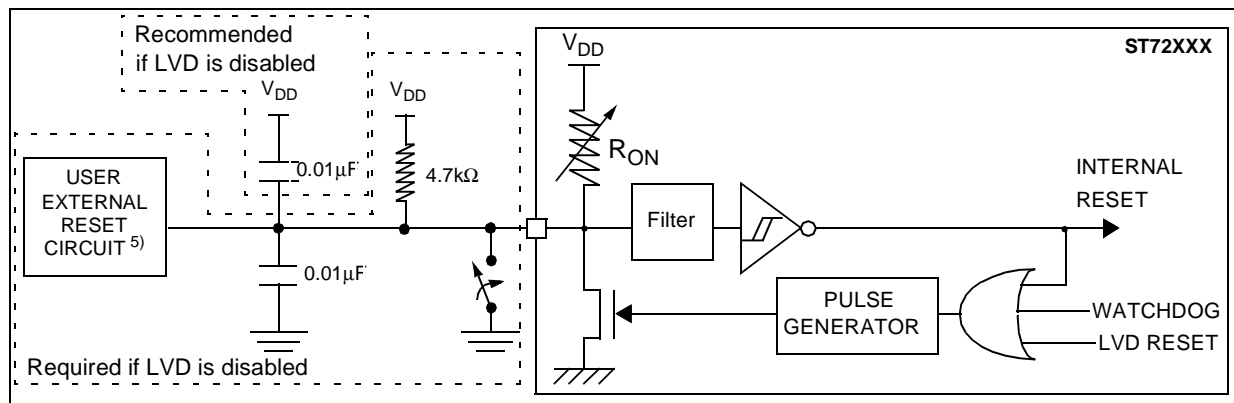
12.9 CONTROL PIN CHARACTERISTICS

12.9.1 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 <sup>1)</sup>            |                        |                      |     | $0.16 \times V_{DD}$ | V          |
| $V_{IH}$         | Input high level voltage <sup>1)</sup>           |                        | $0.85 \times V_{DD}$ |     |                      |            |
| $V_{hys}$        | Schmitt trigger voltage hysteresis <sup>2)</sup> |                        |                      | 2.5 |                      |            |
| $V_{OL}$         | Output low level voltage <sup>3)</sup>           | $V_{DD}=5V$            | $I_{IO}=+5mA$        | 0.5 | 1.2                  | mA         |
|                  |  |                        | $I_{IO}=+2mA$        | 0.2 | 0.5                  |            |
| $I_{IO}$         | Input current on $\overline{\text{RESET}}$ pin   |                        |                      | 2   | TBD                  |            |
| $R_{ON}$         | Weak pull-up equivalent resistor                 |                        | 20                   | 30  | 120                  | k $\Omega$ |
| $t_{w(RSTL)out}$ | Generated reset pulse duration                   | External pin           | 0                    |     | $42^9)$              | $\mu s$    |
|                  |  | Internal reset sources | 20                   | 30  | $42^9)$              | $\mu s$    |
| $t_{h(RSTL)in}$  | External reset pulse hold time <sup>4)</sup>     |                        | 2.5                  |     |                      | $\mu s$    |
| $t_{g(RSTL)in}$  | Filtered glitch duration <sup>5)</sup>           |                        |                      | 200 |                      | ns         |

Figure 104. Typical Application with  $\overline{\text{RESET}}$  pin <sup>6)7)8)</sup>



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 12.2.2 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  $\overline{\text{RESET}}$  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. 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).
7. 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 12.9.1. Otherwise the reset will not be taken into account internally.
8. 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 (by an external pull-up for example) is less than the absolute maximum value specified for  $I_{INJ(RESET)}$  in Section 12.2.2 on page 155.
9. Data guaranteed by design, not tested in production.

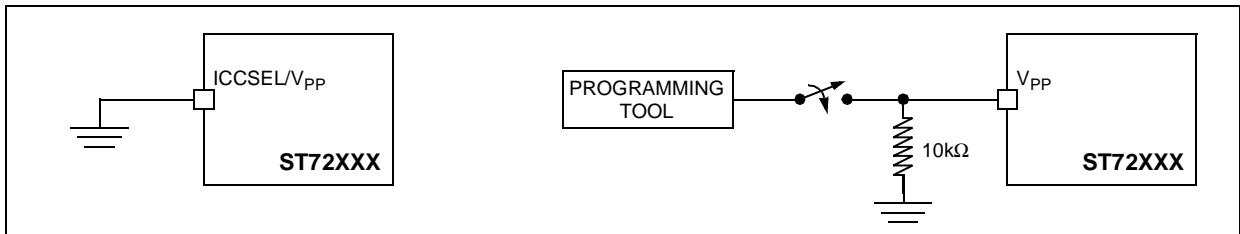
**CONTROL PIN CHARACTERISTICS (Cont'd)**

**12.9.2 ICCSEL/V<sub>PP</sub> Pin**

Subject to general operating conditions for V<sub>DD</sub>, f<sub>CPU</sub>, and T<sub>A</sub> unless otherwise specified.

| Symbol          | Parameter                              | Conditions                       | Min                  | Max                 | Unit |
|-----------------|--|----------------------------------|----------------------|---------------------|------|
| V <sub>IL</sub> | Input low level voltage <sup>1)</sup>  | FLASH versions                   | V <sub>SS</sub>      | 0.2                 | V    |
|                 |  | ROM versions                     | V <sub>SS</sub>      | 0.3xV <sub>DD</sub> |      |
| V <sub>IH</sub> | Input high level voltage <sup>1)</sup> | FLASH versions                   | V <sub>DD</sub> -0.1 | 12.6                | V    |
|                 |  | ROM versions                     | 0.7xV <sub>DD</sub>  | V <sub>DD</sub>     |      |
| I <sub>L</sub>  | Input leakage current                  | V <sub>IN</sub> =V <sub>SS</sub> |                      | ±1                  | µA   |

**Figure 105. Two typical Applications with ICCSEL/V<sub>PP</sub> Pin <sup>2)</sup>**



**Notes:**

1. Data based on design simulation and/or technology characteristics, not tested in production.
2. When ICC mode is not required by the application ICCSEL/V<sub>PP</sub> pin must be tied to V<sub>SS</sub>.

**12.10 TIMER PERIPHERAL CHARACTERISTICS**

Subject to general operating conditions for V<sub>DD</sub>, f<sub>OSC</sub>, and T<sub>A</sub> 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...).

**12.10.1 8-Bit PWM-ART Auto-Reload Timer**

| Symbol                | Parameter                    | Conditions                      | Min | Typ | Max                 | Unit             |
|-----------------------|------------------------------|---------------------------------|-----|-----|---------------------|------------------|
| t <sub>res(PWM)</sub> | PWM resolution time          |                                 | 1   |     |                     | t <sub>CPU</sub> |
|                       |                              | f <sub>CPU</sub> =8MHz          | 125 |     |                     | ns               |
| f <sub>EXT</sub>      | ART external clock frequency |                                 | 0   |     | f <sub>CPU</sub> /2 | MHz              |
| f <sub>PWM</sub>      | PWM repetition rate          |                                 | 0   |     | f <sub>CPU</sub> /2 |                  |
| Res <sub>PWM</sub>    | PWM resolution               |                                 |     |     | 8                   | bit              |
| V <sub>OS</sub>       | PWM/DAC output step voltage  | V <sub>DD</sub> =5V, Res=8-bits |     | 20  |                     | mV               |

**12.10.2 16-Bit Timer**

| Symbol                 | Parameter                      | Conditions             | Min | Typ | Max                 | Unit             |
|------------------------|--------------------------------|------------------------|-----|-----|---------------------|------------------|
| t <sub>w(ICAP)in</sub> | Input capture pulse time       |                        | 1   |     |                     | t <sub>CPU</sub> |
| t <sub>res(PWM)</sub>  | PWM resolution time            |                        | 2   |     |                     | t <sub>CPU</sub> |
|                        |                                | f <sub>CPU</sub> =8MHz | 250 |     |                     | ns               |
| f <sub>EXT</sub>       | Timer external clock frequency |                        | 0   |     | f <sub>CPU</sub> /4 | MHz              |
| f <sub>PWM</sub>       | PWM repetition rate            |                        | 0   |     | f <sub>CPU</sub> /4 | MHz              |
| Res <sub>PWM</sub>     | PWM resolution                 |                        |     |     | 16                  | bit              |

12.11 COMMUNICATION INTERFACE CHARACTERISTICS

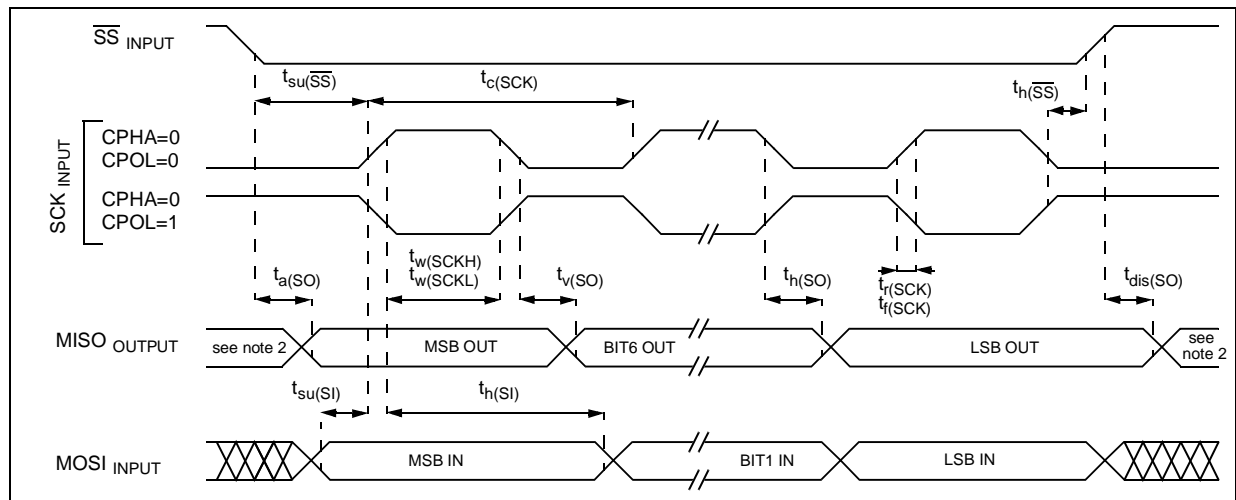
12.11.1 SPI - Serial Peripheral Interface

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 (SS, SCK, MOSI, MISO).

| Symbol                       | Parameter                    | Conditions                   | Min                          | Max              | Unit      |
|------------------------------|------------------------------|------------------------------|------------------------------|------------------|-----------|
| $f_{SCK}$<br>1/ $t_c(SCK)$   | SPI clock frequency          | Master<br>$f_{CPU}=8MHz$     | $f_{CPU}/128$<br>0.0625      | $f_{CPU}/4$<br>2 | MHz       |
|                              |                              | Slave<br>$f_{CPU}=8MHz$      | 0                            | $f_{CPU}/2$<br>4 |           |
| $t_r(SCK)$<br>$t_f(SCK)$     | 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                          |                  |           |
| $t_w(SCKH)$<br>$t_w(SCKL)$   | SCK high and low time        | Master                       | 100                          |                  |           |
|                              |                              | Slave                        | 90                           |                  |           |
| $t_{su}(MI)$<br>$t_{su}(SI)$ | Data input setup time        | Master                       | 100                          |                  |           |
|                              |                              | Slave                        | 100                          |                  |           |
| $t_h(MI)$<br>$t_h(SI)$       | Data input hold time         | Master                       | 100                          |                  |           |
|                              |                              | Slave                        | 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 106. SPI Slave Timing Diagram with CPHA=0<sup>3)</sup>



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:  $0.3 \times V_{DD}$  and  $0.7 \times V_{DD}$ .

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

Figure 107. SPI Slave Timing Diagram with CPHA=1<sup>1)</sup>

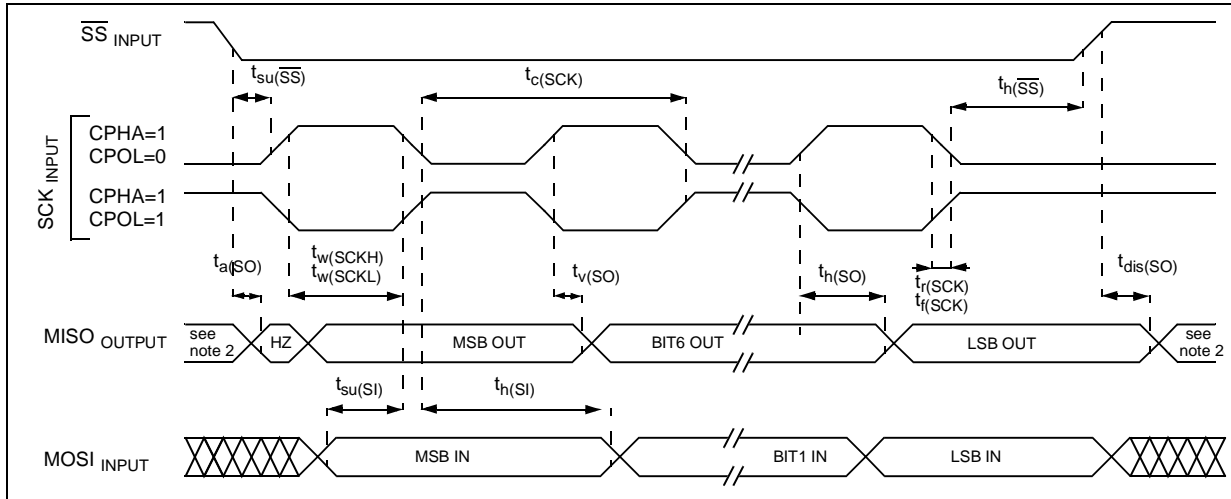
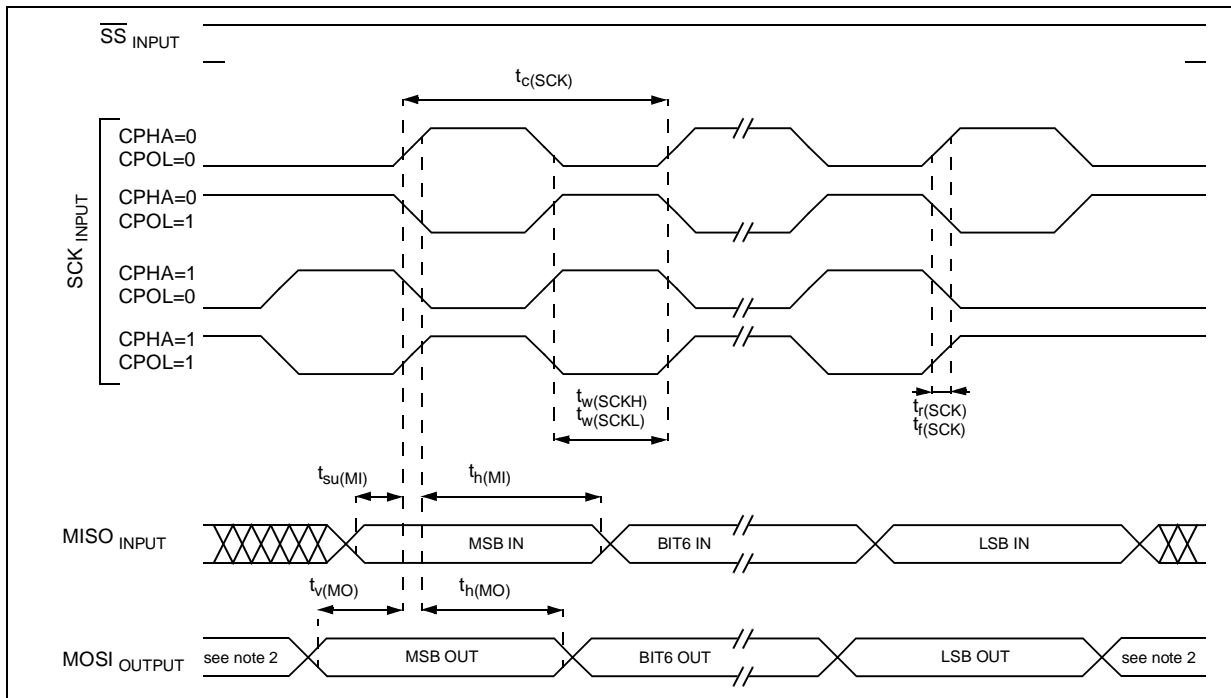


Figure 108. SPI Master Timing Diagram<sup>1)</sup>



Notes:

1. Measurement points are done at CMOS levels:  $0.3 \times V_{DD}$  and  $0.7 \times V_{DD}$ .
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.

**COMMUNICATIONS INTERFACE CHARACTERISTICS (Cont'd)**

**12.11.2 CAN - Controller Area Network Interface**

Subject to general operating condition for  $V_{DD}$ ,  $f_O$ ,  $S_C$ , and  $T_A$  unless otherwise specified. the input/output alternate function characteristics (CANTX and CANRX).  
 Refer to I/O port characteristics for more details on

| Symbol         | Parameter                                     | Conditions | Min | Typ | Max | Unit |
|----------------|---|------------|-----|-----|-----|------|
| $t_{p(RX:TX)}$ | CAN controller propagation time <sup>1)</sup> |            |     |     | 60  | ns   |

**Notes:**

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

**COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)**

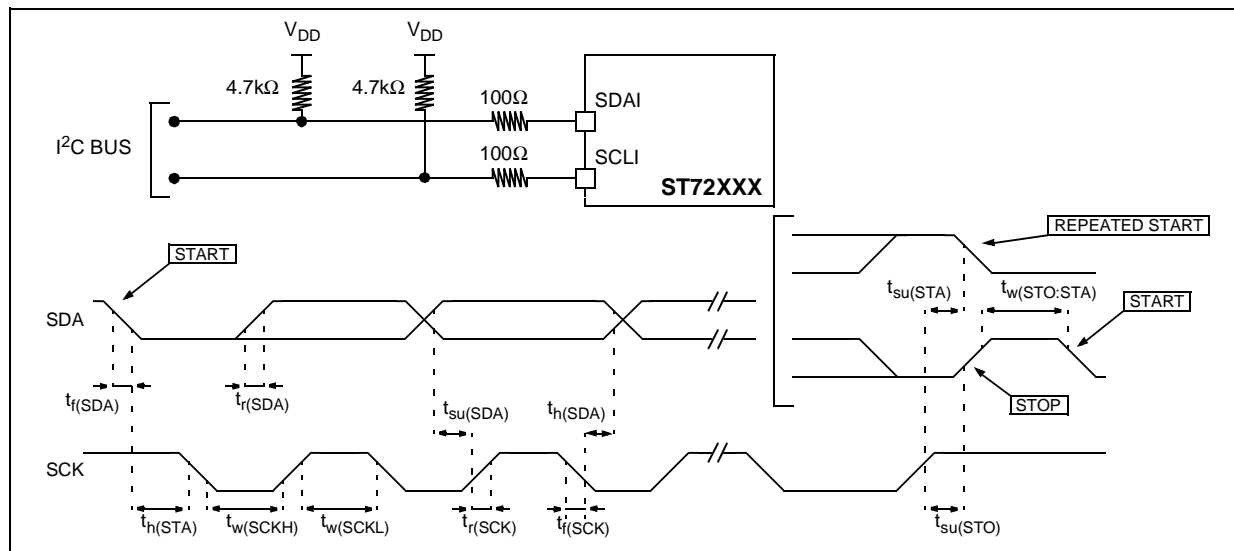
**12.11.3 I<sup>2</sup>C - Inter IC Control Interface**

Subject to general operating conditions for V<sub>DD</sub>, f<sub>CPU</sub>, and T<sub>A</sub> unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (SDAI and SCLI). The ST7 I<sup>2</sup>C interface meets the requirements of the Standard I<sup>2</sup>C communication protocol described in the following table.

| Symbol                                       | Parameter                               | Standard mode I <sup>2</sup> C |                   | Fast mode I <sup>2</sup> C |                   | Unit |
|--|---|--------------------------------|-------------------|----------------------------|-------------------|------|
|  |   | Min <sup>1)</sup>              | Max <sup>1)</sup> | Min <sup>1)</sup>          | Max <sup>1)</sup> |      |
| t <sub>w</sub> (SCLL)                        | SCL clock low time                      | 4.7                            |                   | 1.3                        |                   | μs   |
| t <sub>w</sub> (SCLH)                        | SCL clock high time                     | 4.0                            |                   | 0.6                        |                   |      |
| t <sub>su</sub> (SDA)                        | SDA setup time                          | 250                            |                   | 100                        |                   | ns   |
| t <sub>h</sub> (SDA)                         | SDA data hold time                      | 0 <sup>3)</sup>                |                   | 0 <sup>2)</sup>            | 900 <sup>3)</sup> |      |
| t <sub>r</sub> (SDA)<br>t <sub>r</sub> (SCL) | SDA and SCL rise time                   |                                | 1000              | 20+0.1C <sub>b</sub>       | 300               |      |
| t <sub>f</sub> (SDA)<br>t <sub>f</sub> (SCL) | SDA and SCL fall time                   |                                | 300               | 20+0.1C <sub>b</sub>       | 300               |      |
| t <sub>h</sub> (STA)                         | START condition hold time               | 4.0                            |                   | 0.6                        |                   | μs   |
| t <sub>su</sub> (STA)                        | Repeated START condition setup time     | 4.7                            |                   | 0.6                        |                   |      |
| t <sub>su</sub> (STO)                        | STOP condition setup time               | 4.0                            |                   | 0.6                        |                   | ns   |
| t <sub>w</sub> (STO:STA)                     | STOP to START condition time (bus free) | 4.7                            |                   | 1.3                        |                   | ms   |
| C <sub>b</sub>                               | Capacitive load for each bus line       |                                | 400               |                            | 400               | pF   |

**Figure 109. Typical Application with I<sup>2</sup>C Bus and Timing Diagram <sup>4)</sup>**



**Notes:**

1. Data based on standard I<sup>2</sup>C protocol requirement, not tested in production.
2. The device must internally provide a hold time of at least 300ns for the SDA signal in order to bridge the undefined region of the falling edge of SCL.
3. The maximum hold time of the START condition has only to be met if the interface does not stretch the low period of SCL signal.
4. Measurement points are done at CMOS levels: 0.3xV<sub>DD</sub> and 0.7xV<sub>DD</sub>.

12.12 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 <sup>1)</sup> | Max   | Unit        |
|------------|--|--|--------------------|-------------------|---|-------------|
| $f_{ADC}$  | ADC clock frequency  |  | 0.4                |                   | 2   | MHz         |
| $V_{AREF}$ | Analog reference voltage <sup>2)</sup>                                     |  | $0.7 \cdot V_{DD}$ |                   | 5.5   | V           |
| $V_{AIN}$  | Conversion voltage range <sup>3)</sup>                                     |  | $V_{SSA}$          |                   | $V_{AREF}$                                      |             |
| $I_L$      | Input leakage current for analog input                                     | $-40^\circ C \leq T_A \leq 85^\circ C$ range |                    |                   | $\pm 250$                                       | nA          |
|            |  | Other $T_A$ ranges                           |                    |                   | $\pm 1$   | $\mu A$     |
| $R_{AIN}$  | External input impedance   |  |                    |                   | see Figure 110 and Figure 111 <sup>3)4)5)</sup> | k $\Omega$  |
| $C_{AIN}$  | External capacitor on analog input   |  |                    |                   |   | pF          |
| $f_{AIN}$  | Variation freq. of analog input signal                                     |  |                    |                   |   | Hz          |
| $C_{ADC}$  | Internal sample and hold capacitor   |  |                    | 12                |   | pF          |
| $t_{STAB}$ | Stabilization time after ADC enable  | $f_{CPU}=8MHz, SPEED=0$<br>$f_{ADC}=2MHz$    |                    | 0 <sup>5)</sup>   |   | $\mu s$     |
| $t_{ADC}$  | Conversion time (Sample+Hold)  |  |                    | 7.5               |   |             |
|            | - No of sample capacitor loading cycles<br>- No. of Hold conversion cycles |  |                    | 4<br>11           |   | $1/f_{ADC}$ |

Figure 110.  $R_{AIN}$  max. vs  $f_{ADC}$  with  $C_{AIN}=0pF^4)$

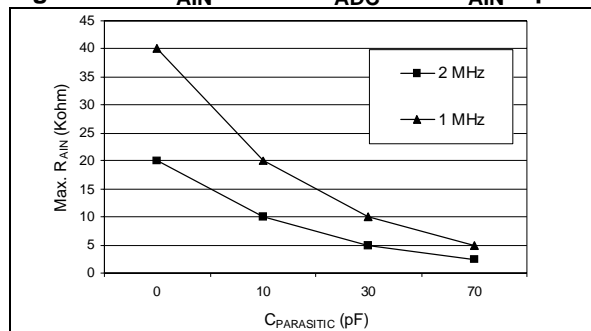


Figure 111. Recommended  $C_{AIN}$  &  $R_{AIN}$  values.<sup>5)</sup>

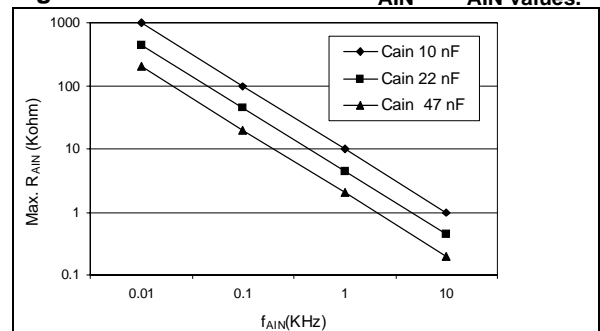
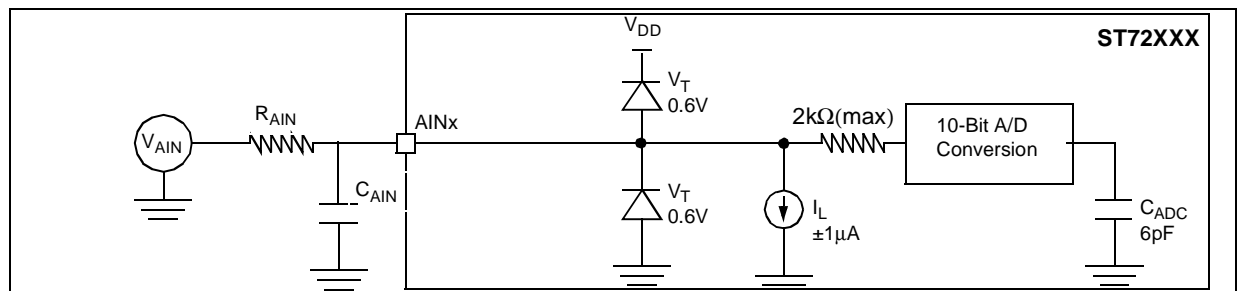


Figure 112. Typical A/D Converter Application



Notes:

1. Unless otherwise specified, typical data are based on  $T_A=25^\circ C$  and  $V_{DD}-V_{SS}=5V$ . They are given only as design guidelines and are not tested.
2. When  $V_{DDA}$  and  $V_{SSA}$  pins are not available on the pinout, the ADC refers to  $V_{DD}$  and  $V_{SS}$ .
3. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10k $\Omega$ ). Data based on characterization results, not tested in production.
4.  $C_{PARASITIC}$  represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high  $C_{PARASITIC}$  value will downgrade conversion accuracy. To remedy this,  $f_{ADC}$  should be reduced.
5. This graph shows that depending on the input signal variation ( $f_{AIN}$ ),  $C_{AIN}$  can be increased for stabilization and to allow the use of a larger serial resistor ( $R_{AIN}$ ).



## ADC CHARACTERISTICS (Cont'd)

### 12.12.0.1 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. In smaller packages  $V_{AREF}$  and  $V_{SSA}$  pins are not available and the analog supply and reference pads are internally bonded to the  $V_{DD}$  and  $V_{SS}$  pins.

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 12.12.0.2 General PCB Design Guidelines).

### 12.12.0.2 General PCB Design Guidelines

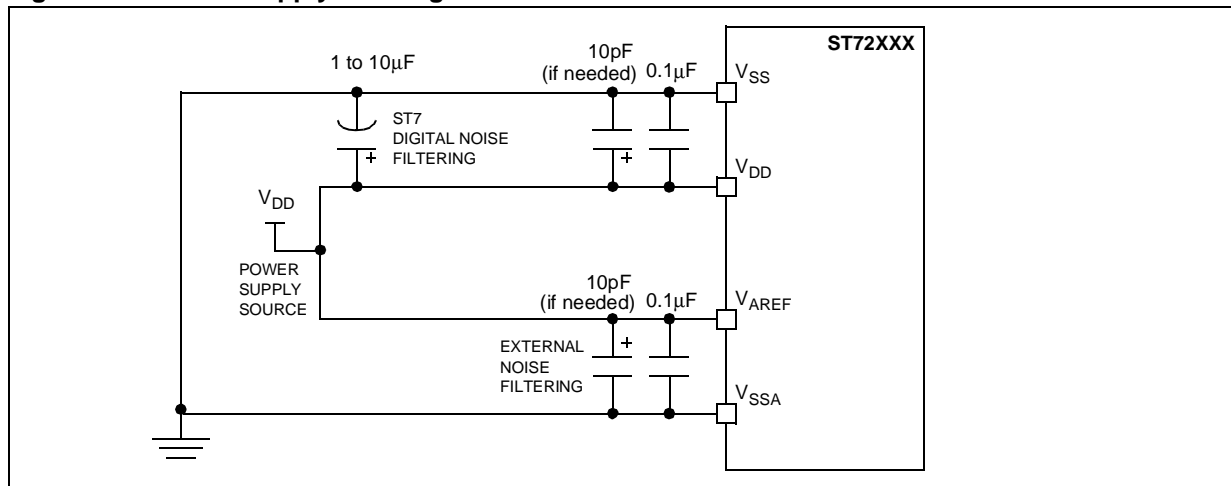
To obtain best results, some general design and layout rules should be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

- 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 0.1 $\mu$ F and optionally, if needed 10pF capacitors as close as possible to the ST7 power supply pins and a 1 to 10 $\mu$ F capacitor close to the power source (see Figure 113).
- 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 113. Power Supply Filtering



10-BIT ADC CHARACTERISTICS (Cont'd)

12.12.1 ADC Accuracy

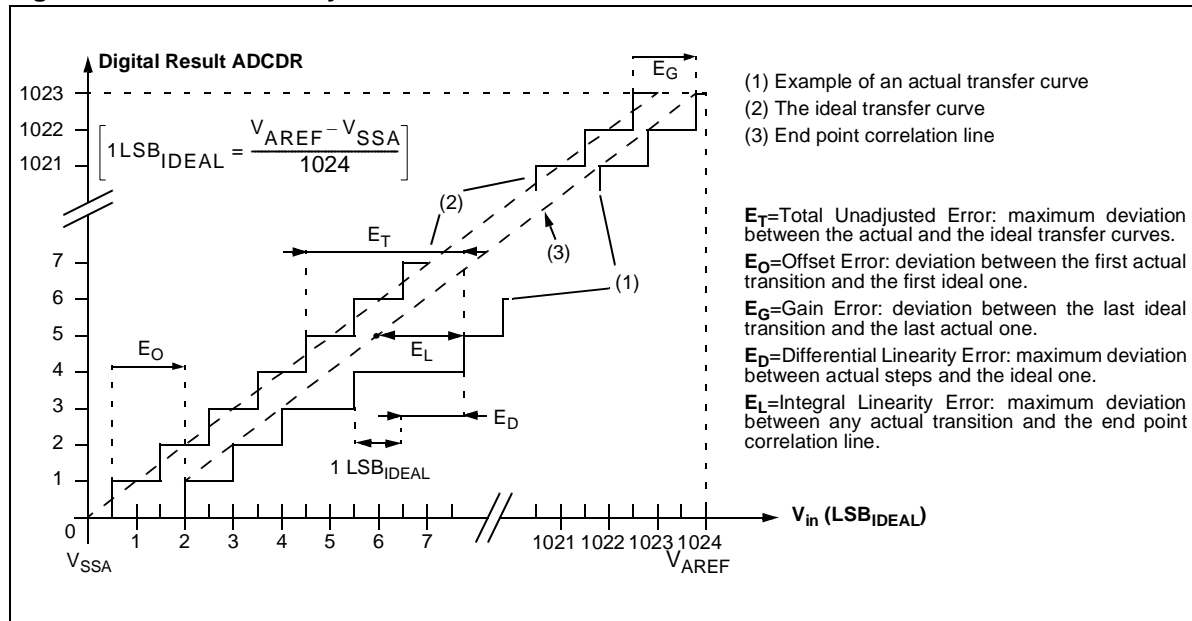
Conditions:  $V_{DD}=5V$

| Symbol  | Parameter                                  | Conditions                         | Typ  | Max               | Unit |
|---------|--|------------------------------------|------|-------------------|------|
| $ E_T $ | Total unadjusted error <sup>1)</sup>       |                                    | 4    |                   | LSB  |
| $E_O$   | Offset error <sup>1)</sup>                 |                                    | 3    | 3.5 <sup>2)</sup> |      |
| $E_G$   | Gain Error <sup>1)</sup>                   |                                    | -0.5 | -2 <sup>2)</sup>  |      |
| $ E_D $ | Differential linearity error <sup>1)</sup> | CPU in run mode @ $f_{ADC}$ 2 MHz. | 1.5  | 4.5 <sup>2)</sup> |      |
| $ E_L $ | Integral linearity error <sup>1)</sup>     | CPU in run mode @ $f_{ADC}$ 2 MHz. | 1.5  | 4.5 <sup>2)</sup> |      |

Notes:

- 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 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 12.8 does not affect the ADC accuracy.
- Data based on characterization results, monitored in production.

Figure 114. ADC Accuracy Characteristics



### 13 PACKAGE CHARACTERISTICS

#### 13.1 PACKAGE MECHANICAL DATA

Figure 115. 80-Pin Thin Quad Flat Package

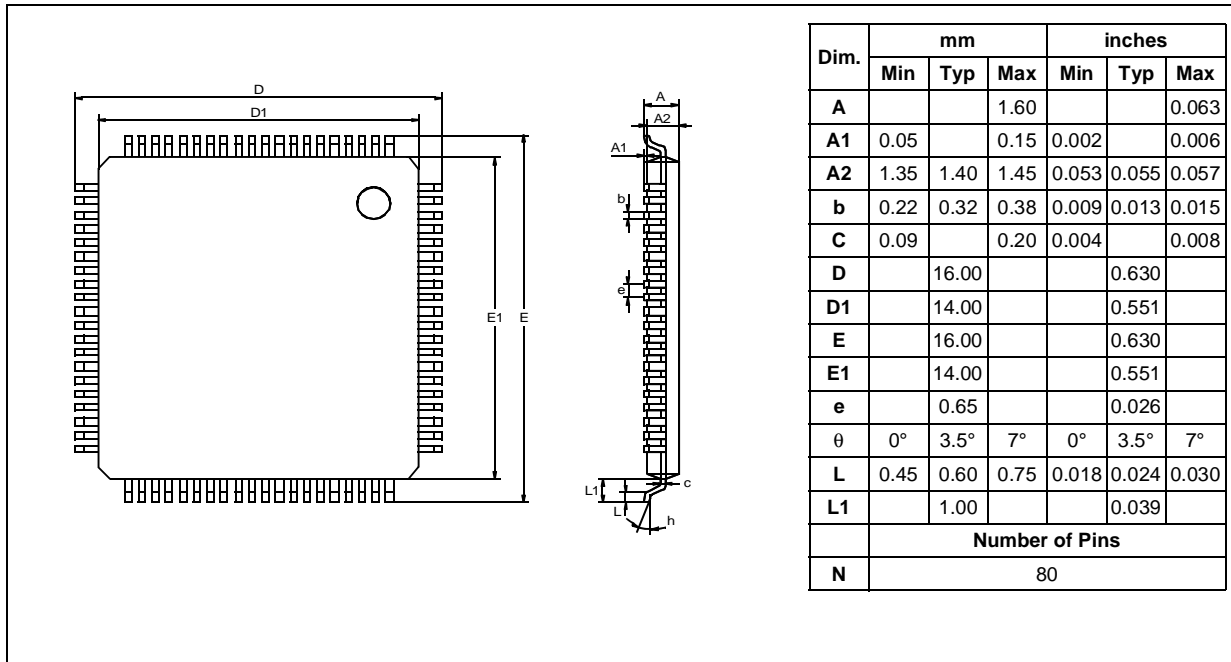
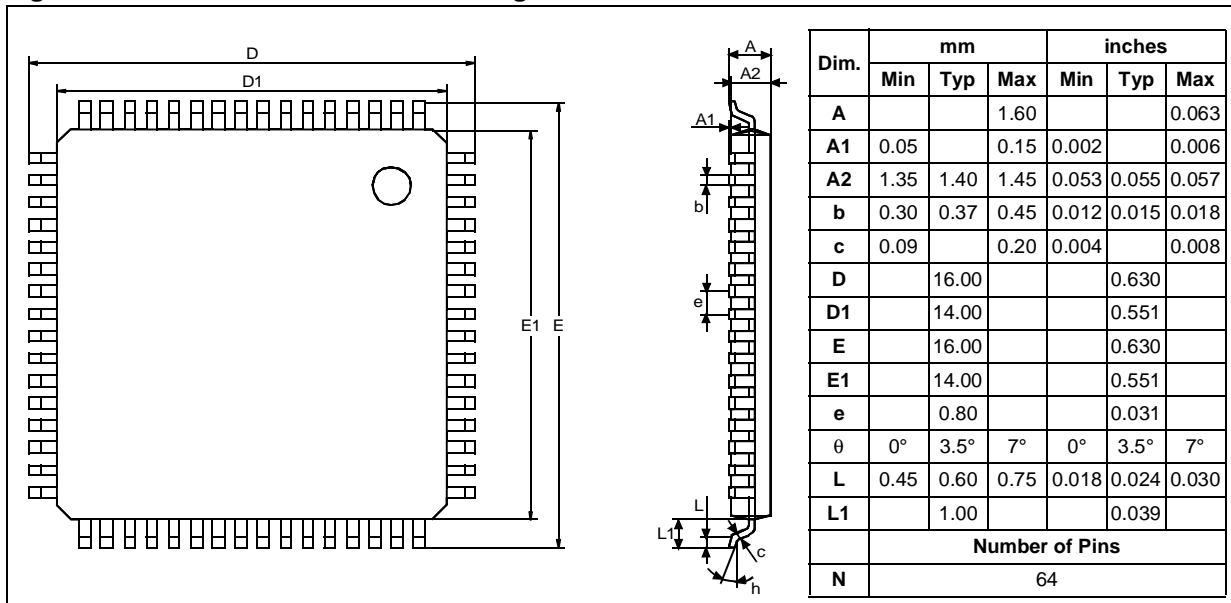
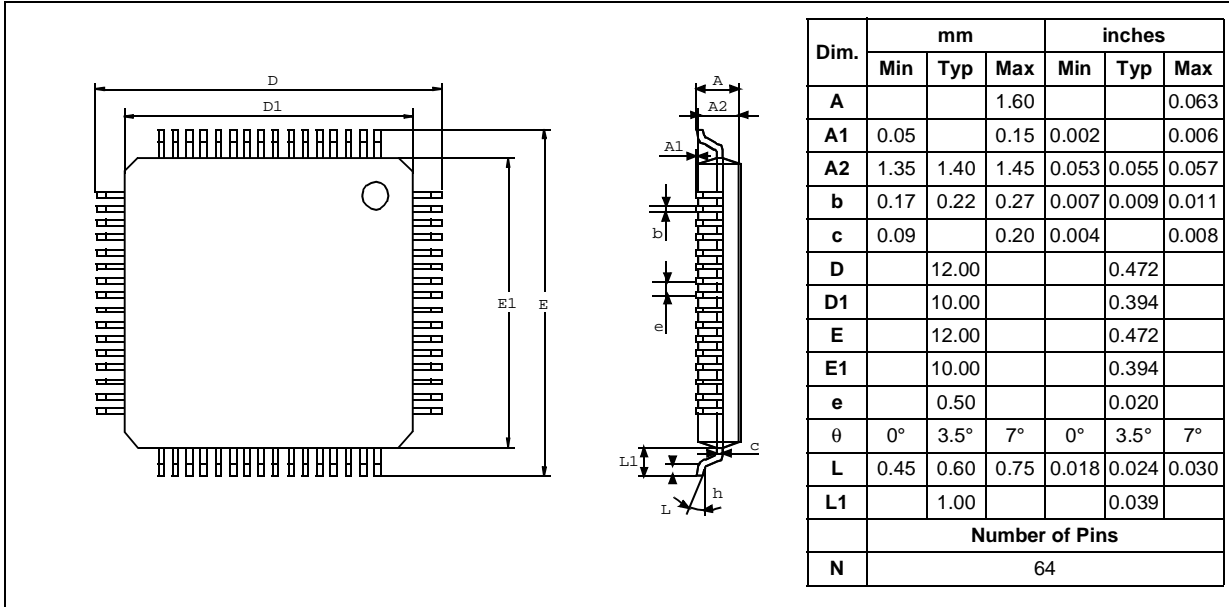


Figure 116. 64-Pin Thin Quad Flat Package



PACKAGE MECHANICAL DATA (Cont'd)

Figure 117. 64-Pin Thin Quad Flat Package



## 13.2 THERMAL CHARACTERISTICS

| Symbol     | Ratings  | Value | Unit |
|------------|--|-------|------|
| $R_{thJA}$ | Package thermal resistance (junction to ambient) |       |      |
|            | TQFP80 14x14                                     | 55    | °C/W |
|            | TQFP64 14x14                                     | 47    |      |
|            | TQFP64 10x10                                     | 50    |      |
|            |  |       |      |
| $P_D$      | Power dissipation <sup>1)</sup>                  | 500   | mW   |
| $T_{Jmax}$ | Maximum junction temperature <sup>2)</sup>       | 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 ( $I_{DD} \times V_{DD}$ ) and  $P_{PORT}$  is the port power dissipation determined by the user.
2. The average chip-junction temperature can be obtained from the formula  $T_J = T_A + P_D \times R_{thJA}$ .

13.3 SOLDERING AND GLUEABILITY INFORMATION

Recommended soldering information given only as design guidelines.

Figure 118. Recommended Wave Soldering Profile (with 37% Sn and 63% Pb)

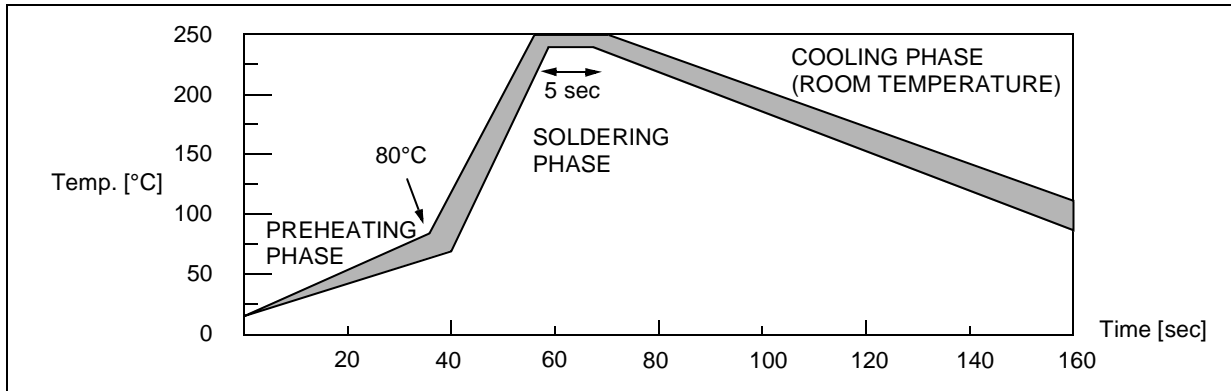
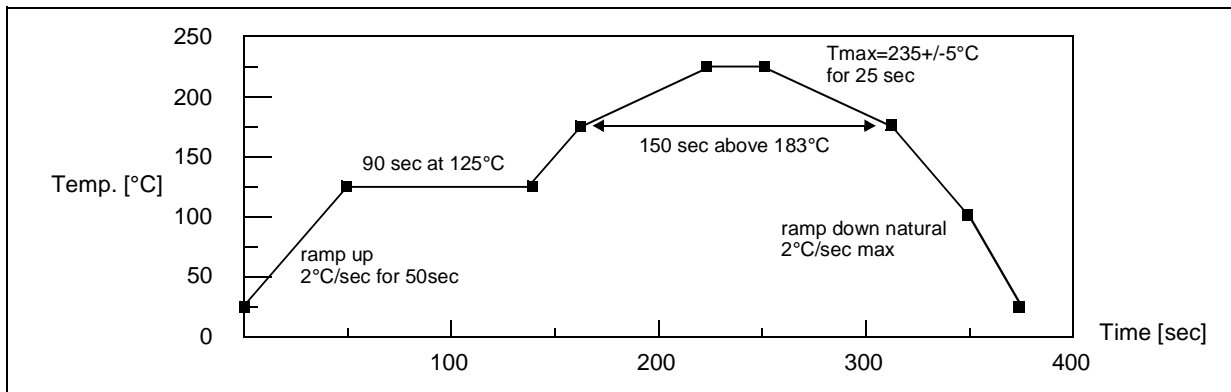


Figure 119. Recommended Reflow Soldering Oven Profile (MID JEDEC)



Recommended glue for SMD plastic packages dedicated to molding compound with silicone:

- Heraeus: PD945, PD955
- Loctite: 3615, 3298

## 14 ST72521M/(A)R DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in user programmable versions (FLASH) as well as in factory coded versions (ROM). FLASH devices are shipped to customers with a default content, while ROM factory coded parts contain the code sup-

plied by the customer. This implies that FLASH devices have to be configured by the customer using the Option Bytes while the ROM devices are factory-configured.

### 14.1 FLASH OPTION BYTES

|         | STATIC OPTION BYTE 0 |    |     |    |   |      |       | STATIC OPTION BYTE 1 |      |         |   |          |   |   |        |
|---------|----------------------|----|-----|----|---|------|-------|----------------------|------|---------|---|----------|---|---|--------|
|         | WDG                  |    | CSS | VD |   | 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       | 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. In masked ROM devices, the option bytes are fixed in hardware by the ROM code (see option list).

#### 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 = **CSS** *Clock security system on/off*

This option bit enables or disables the clock security system function (CSS) which includes the clock filter and the backup safe oscillator.

0: CSS enabled  
1: CSS disabled

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   |
| Not Used (Lowest Threshold: $V_{DD} \sim 3V$ ) | 1   | 0   |
| Not Used (Med. Threshold $V_{DD} \sim 3.5V$ )  | 0   | 1   |
| Highest Voltage Threshold ( $V_{DD} \sim 4V$ ) | 0   | 0   |

**Note:** For details on the AVD and LVD threshold levels refer to Section 12.3.2 on page 157

**ST72521M/(A)R DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)**

**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*  
 This option indicates if the user flash memory is protected against read-out piracy. This protection is based on a read and write protection of the memory in test modes and ICP mode. Erasing the option bytes when the FMP\_R option is selected causes the whole user memory to be erased first.  
 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       | TQFP80           | 1     | 1     |
| (A)R    | TQFP64           | 1     | 0     |

**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  
**Note:** When the CSS is enabled, the device starts to count immediately using the backup oscillator.

**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 |
| External RC Oscillator | 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               | 1~2MHz  | 0        | 0 | 0 |
| MP               | 2~4MHz  | 0        | 0 | 1 |
| MS               | 4~8MHz  | 0        | 1 | 0 |
| HS               | 8~16MHz | 0        | 1 | 1 |

**OPT0 = PLL OFF** *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. The PLL is guaranteed only with an input frequency between 2 and 4MHz.

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 - 2~4MHz". Otherwise, the device functionality is not guaranteed.



ST72521M/(A)R DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

14.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

Customer code is made up of the ROM contents and the list of the selected options (if any). The ROM contents are to be sent on diskette, or by electronic means, with the S19 hexadecimal file generated by the development tool. All unused bytes must be set to FFh.

The selected options are communicated to STMicroelectronics using the correctly completed OPTION LIST appended.

The STMicroelectronics Sales Organization will be pleased to provide detailed information on contractual points.

Figure 120. ROM Factory Coded Device Types

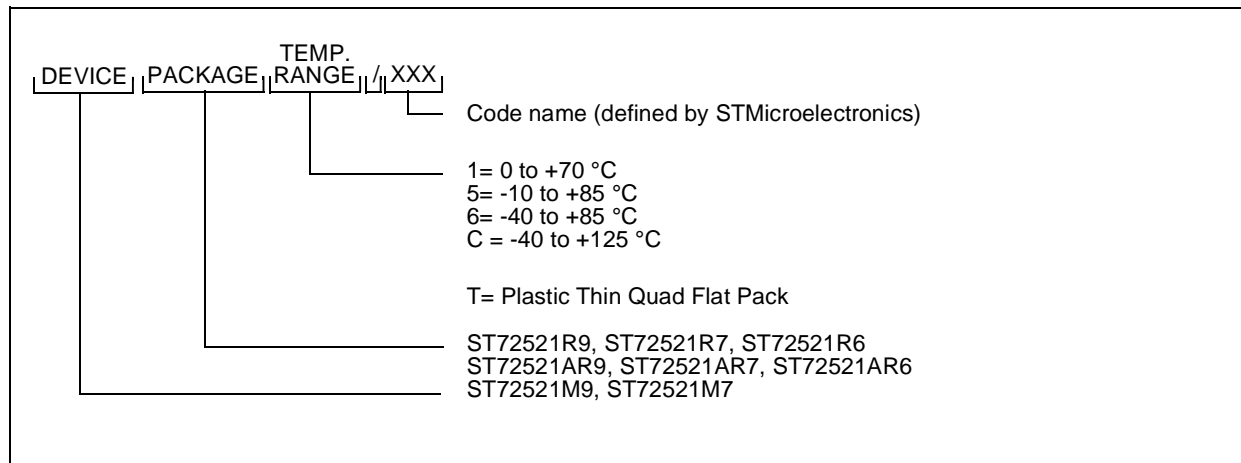


Table 29. Orderable Flash Device Types

| Part Number   | Program Memory (Bytes) | Temp. Range  | Package        |
|---------------|------------------------|--------------|----------------|
| ST72F521M9TC  | 60KB FLASH             | -40°C +125°C | TQFP80         |
| ST72F521M9T6  |                        | -40°C +85°C  | TQFP80         |
| ST72F521R9TC  |                        | -40°C +125°C | TQFP64 14 x 14 |
| ST72F521R9T6  |                        | -40°C +85°C  | TQFP64 14 x 14 |
| ST72F521AR9TC |                        | -40°C +125°C | TQFP64 10 x 10 |
| ST72F521AR9T6 |                        | -40°C +85°C  | TQFP64 10 x 10 |
| ST72F521R6TC  | 32KB FLASH             | -40°C +125°C | TQFP64 14 x 14 |
| ST72F521R6T6  |                        | -40°C +85°C  | TQFP64 14 x 14 |
| ST72F521AR6TC |                        | -40°C +125°C | TQFP64 10 x 10 |
| ST72F521AR6T6 |                        | -40°C +85°C  | TQFP64 10 x 10 |

# ST72521M/R/AR

## ST72521M/(A)R DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

| ST72521 MICROCONTROLLER OPTION LIST                                       |                |  |                         |
|---|----------------|--|-------------------------|
| Customer:   | .....          |  |                         |
| Address:  | .....          |  |                         |
| Contact:  | .....          |  |                         |
| Phone No:   | .....          |  |                         |
| Reference/ROM Code* :   | .....          |  |                         |
| *The ROM code name is assigned by STMicroelectronics.                     |                |  |                         |
| ROM code must be sent in .S19 format. .Hex extension cannot be processed. |                |  |                         |
| Device Type/Memory Size/Package (check only one option):                  |                |  |                         |
| ROM DEVICE:   | 60K            | 48K                                    | 32K                     |
| TQFP80:   | [ ] ST72521M9  | [ ] ST72521M7                          |                         |
| TQFP64 14x14:   | [ ] ST72521R9  | [ ] ST72521R7                          | [ ] ST72521R6           |
| TQFP64 10x10:   | [ ] ST72521AR9 | [ ] ST72521AR7                         | [ ] ST72521AR6          |
| DIE FORM:   | 60K            | 48K                                    | 32K                     |
| 80-pin:   | [ ]            | [ ]                                    |                         |
| 64-pin:   | [ ]            | [ ]                                    | [ ]                     |
| Conditioning (check only one option):                                     |                |  |                         |
| Packaged Product  |                | Die Product (dice tested at 25°C only) |                         |
| [ ] Tape & Reel   | [ ] Tray       | [ ] Tape & Reel                        | [ ] Inked wafer         |
|   |                | [ ] Sawn wafer on sticky foil          |                         |
| Temp. Range (do not check for die product):                               |                |  |                         |
| [ ] 0°C to +70°C  |                |  |                         |
| [ ] -10°C to +85°C  |                | [ ] -40°C to +105°C                    |                         |
| [ ] -40°C to +85°C  |                | [ ] -40°C to +125°C                    |                         |
| Special Marking: [ ] No [ ] Yes " _____ " (10 char. max)                  |                |  |                         |
| Authorized characters are letters, digits, '.', '-', '/' and spaces only. |                |  |                         |
| Clock Source Selection:   |                |  |                         |
| [ ] Resonator: [ ] LP: Low power resonator (1 to 2 MHz)                   |                |  |                         |
| [ ] MP: Medium power resonator (2 to 4 MHz)                               |                |  |                         |
| [ ] MS: Medium speed resonator (4 to 8 MHz)                               |                |  |                         |
| [ ] HS: High speed resonator (8 to 16 MHz)                                |                |  |                         |
| [ ] RC Network: <sup>1</sup> [ ] Internal                                 |                |  |                         |
| [ ] External  |                |  |                         |
| [ ] External Clock  |                |  |                         |
| PLL <sup>1</sup>  |                | [ ] Disabled                           | [ ] Enabled             |
| Clock Security System:  |                | [ ] Disabled                           | [ ] Enabled             |
| LVD Reset   |                | [ ] Disabled [ ] Enabled:              | [ ] Highest threshold   |
| Reset Delay   |                | [ ] 256 Cycles [ ] 4096 Cycles         |                         |
| Watchdog Selection:   |                | [ ] Software Activation                | [ ] Hardware Activation |
| Halt when Watchdog on:  |                | [ ] Reset                              | [ ] No reset            |
| Readout Protection:   |                | [ ] Disabled                           | [ ] Enabled             |
| Date .....  |                |  |                         |
| Signature .....   |                |  |                         |
| <sup>1</sup> PLL must not be enabled if internal RC Network is selected   |                |  |                         |

## DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

### 14.3 DEVELOPMENT TOOLS

STMicroelectronics offers a range of hardware and software development tools for the ST7 micro-controller family. Full details of tools available for the ST7 from third party manufacturers can be obtained from the STMicroelectronics Internet site:

→ <http://mcu.st.com>.

Tools from these manufacturers include C compilers, emulators and gang programmers.

#### ST Emulators

The emulator is delivered with everything (probes, TEB, adapters etc.) needed to start emulating the devices. To configure the emulator to emulate different ST7 subfamily devices, the active probe for the ST7 EMU3 can be changed and the ST7EMU3 probe is designed for easy interchange of TEBs

(Target Emulation Board). See Table 30 for more details.

#### 14.3.1 Socket and Emulator Adapter Information

For information on the type of socket that is supplied with the emulator, refer to the suggested list of sockets in Table 31.

**Note:** Before designing the board layout, it is recommended to check the overall dimensions of the socket as they may be greater than the dimensions of the device.

For footprint and other mechanical information about these sockets and adapters, refer to the manufacturer's datasheet ([www.yamaichi.de](http://www.yamaichi.de) for TQFP64 10 x 10 and TQFP80 14 x 14 and [www.cabgmbh.com](http://www.cabgmbh.com) for TQFP64 14 x 14)..

**Table 30. STMicroelectronics Development Tools**

| Supported Products                               | ST7 Evaluation Board  | ST7 Emulator       | Active Probe & T.E.B. | ST7 Programming Board                                    |
|--|---|--------------------|-----------------------|--|
| ST72521(A)R, ST72F521(A)R<br>ST72521M, ST72F521M | ST7MDT2-TRAIN/<br>EU ST7MDT2-<br>TRAIN/US<br>ST7MDT2-TRAIN/<br>UK ST7CAN-<br>DEMO | ST7MDT20M-<br>EMU3 | ST7MDT20M-<br>TEB     | ST7MDT20M-EPB/EU<br>ST7MDT20M-EPB/US<br>ST7MDT20M-EPB/UK |

**Note:**

1. Flash Programming interface for FLASH devices.

**Table 31. Suggested List of Socket Types**

| Device         | Socket (supplied with ST7MDT20M-EMU3) | Emulator Adapter (supplied with ST7MDT20M-EMU3) |
|----------------|---------------------------------------|---|
| TQFP64 14 x14  | CAB 3303262                           | CAB 3303351                                     |
| TQFP64 10 x10  | YAMAICHI IC149-064-*75-*5             | YAMAICHI ICP-064-6                              |
| TQFP80 14 X 14 | YAMAICHI IC149-080-*51-*5             | YAMAICHI ICP-080-7                              |

14.4 ST7 APPLICATION NOTES

| IDENTIFICATION              | DESCRIPTION  |
|-----------------------------|--|
| <b>EXAMPLE DRIVERS</b>      |  |
| AN 969                      | SCI COMMUNICATION BETWEEN ST7 AND PC   |
| AN 970                      | SPI COMMUNICATION BETWEEN ST7 AND EEPROM   |
| AN 971                      | I <sup>2</sup> C COMMUNICATING BETWEEN ST7 AND M24CXX EEPROM                         |
| AN 972                      | ST7 SOFTWARE SPI MASTER COMMUNICATION  |
| AN 973                      | SCI SOFTWARE COMMUNICATION WITH A PC USING ST72251 16-BIT TIMER                      |
| AN 974                      | REAL TIME CLOCK WITH ST7 TIMER OUTPUT COMPARE  |
| AN 976                      | DRIVING A BUZZER THROUGH ST7 TIMER PWM FUNCTION                                      |
| AN 979                      | DRIVING AN ANALOG KEYBOARD WITH THE ST7 ADC  |
| AN 980                      | ST7 KEYPAD DECODING TECHNIQUES, IMPLEMENTING WAKE-UP ON KEYSTROKE                    |
| AN1017                      | USING THE ST7 UNIVERSAL SERIAL BUS MICROCONTROLLER                                   |
| AN1041                      | USING ST7 PWM SIGNAL TO GENERATE ANALOG OUTPUT (SINUSOID)                            |
| AN1042                      | ST7 ROUTINE FOR I <sup>2</sup> C SLAVE MODE MANAGEMENT                               |
| AN1044                      | MULTIPLE INTERRUPT SOURCES MANAGEMENT FOR ST7 MCUS                                   |
| AN1045                      | ST7 S/W IMPLEMENTATION OF I <sup>2</sup> C BUS MASTER                                |
| AN1046                      | UART EMULATION SOFTWARE  |
| AN1047                      | MANAGING RECEPTION ERRORS WITH THE ST7 SCI PERIPHERALS                               |
| AN1048                      | ST7 SOFTWARE LCD DRIVER  |
| AN1078                      | PWM DUTY CYCLE SWITCH IMPLEMENTING TRUE 0% & 100% DUTY CYCLE                         |
| AN1082                      | DESCRIPTION OF THE ST72141 MOTOR CONTROL PERIPHERAL REGISTERS                        |
| AN1083                      | ST72141 BLDC MOTOR CONTROL SOFTWARE AND FLOWCHART EXAMPLE                            |
| AN1105                      | ST7 PCAN PERIPHERAL DRIVER   |
| AN1129                      | PERMANENT MAGNET DC MOTOR DRIVE.   |
| AN1130                      | AN INTRODUCTION TO SENSORLESS BRUSHLESS DC MOTOR DRIVE APPLICATIONS WITH THE ST72141 |
| AN1148                      | USING THE ST7263 FOR DESIGNING A USB MOUSE   |
| AN1149                      | HANDLING SUSPEND MODE ON A USB MOUSE   |
| AN1180                      | USING THE ST7263 KIT TO IMPLEMENT A USB GAME PAD                                     |
| AN1276                      | BLDC MOTOR START ROUTINE FOR THE ST72141 MICROCONTROLLER                             |
| AN1321                      | USING THE ST72141 MOTOR CONTROL MCU IN SENSOR MODE                                   |
| AN1325                      | USING THE ST7 USB LOW-SPEED FIRMWARE V4.X  |
| AN1445                      | USING THE ST7 SPI TO EMULATE A 16-BIT SLAVE  |
| AN1475                      | DEVELOPING AN ST7265X MASS STORAGE APPLICATION                                       |
| AN1504                      | STARTING A PWM SIGNAL DIRECTLY AT HIGH LEVEL USING THE ST7 16-BIT TIMER              |
| <b>PRODUCT EVALUATION</b>   |  |
| AN 910                      | PERFORMANCE BENCHMARKING   |
| AN 990                      | ST7 BENEFITS VERSUS INDUSTRY STANDARD  |
| AN1077                      | OVERVIEW OF ENHANCED CAN CONTROLLERS FOR ST7 AND ST9 MCUS                            |
| AN1086                      | U435 CAN-DO SOLUTIONS FOR CAR MULTIPLEXING   |
| AN1150                      | BENCHMARK ST72 VS PC16   |
| AN1151                      | PERFORMANCE COMPARISON BETWEEN ST72254 & PC16F876                                    |
| AN1278                      | LIN (LOCAL INTERCONNECT NETWORK) SOLUTIONS   |
| <b>PRODUCT MIGRATION</b>    |  |
| AN1131                      | MIGRATING APPLICATIONS FROM ST72511/311/214/124 TO ST72521/321/324                   |
| AN1322                      | MIGRATING AN APPLICATION FROM ST7263 REV.B TO ST7263B                                |
| AN1365                      | GUIDELINES FOR MIGRATING ST72C254 APPLICATION TO ST72F264                            |
| <b>PRODUCT OPTIMIZATION</b> |  |

| IDENTIFICATION               | DESCRIPTION   |
|------------------------------|---|
| AN 982                       | USING ST7 WITH CERAMIC RENATOR  |
| AN1014                       | HOW TO MINIMIZE THE ST7 POWER CONSUMPTION                                       |
| AN1015                       | SOFTWARE TECHNIQUES FOR IMPROVING MICROCONTROLLER EMC PERFORMANCE               |
| AN1040                       | MONITORING THE VBUS SIGNAL FOR USB SELF-POWERED DEVICES                         |
| AN1070                       | ST7 CHECKSUM SELF-CHECKING CAPABILITY   |
| AN1324                       | CALIBRATING THE RC OSCILLATOR OF THE ST7FLITE0 MCU USING THE MAINS              |
| AN1477                       | EMULATED DATA EEPROM WITH XFLASH MEMORY   |
| AN1502                       | EMULATED DATA EEPROM WITH ST7 HDFLASH MEMORY                                    |
| AN1529                       | EXTENDING THE CURRENT & VOLTAGE CAPABILITY ON THE ST7265 VDDF SUPPLY            |
| AN1530                       | ACCURATE TIMEBASE FOR LOW-COST ST7 APPLICATIONS WITH INTERNAL RC OSCILLATOR     |
| <b>PROGRAMMING AND TOOLS</b> |   |
| AN 978                       | KEY FEATURES OF THE STVD7 ST7 VISUAL DEBUG PACKAGE                              |
| AN 983                       | KEY FEATURES OF THE COSMIC ST7 C-COMPILER PACKAGE                               |
| AN 985                       | EXECUTING CODE IN ST7 RAM   |
| AN 986                       | USING THE INDIRECT ADDRESSING MODE WITH ST7                                     |
| AN 987                       | ST7 SERIAL TEST CONTROLLER PROGRAMMING  |
| AN 988                       | STARTING WITH ST7 ASSEMBLY TOOL CHAIN   |
| AN 989                       | GETTING STARTED WITH THE ST7 HIWARE C TOOLCHAIN                                 |
| AN1039                       | ST7 MATH UTILITY ROUTINES   |
| AN1064                       | WRITING OPTIMIZED HIWARE C LANGUAGE FOR ST7                                     |
| AN1071                       | HALF DUPLEX USB-TO-SERIAL BRIDGE USING THE ST72611 USB MICROCONTROLLER          |
| AN1106                       | TRANSLATING ASSEMBLY CODE FROM HC05 TO ST7                                      |
| AN1179                       | PROGRAMMING ST7 FLASH MICROCONTROLLERS IN REMOTE ISP MODE (IN-SITU PROGRAMMING) |
| AN1446                       | USING THE ST72521 EMULATOR TO DEBUG A ST72324 TARGET APPLICATION                |
| AN1478                       | PORTING AN ST7 PANTA PROJECT TO CODEWARRIOR IDE                                 |
| AN1527                       | DEVELOPING A USB SMARTCARD READER WITH ST7SCR                                   |
| AN1575                       | ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUS                    |

**15 SUMMARY OF CHANGES**

| Revision | Main Changes   | Date   |
|----------|--|--------|
| 1.6      | <p>Changed ITSPR register names to ISPR in Table 2 on page 15</p> <p>Modified CSS functional description (Glitch filtering with PLL on) in Section 6.4.3 on page 31</p> <p>Removed AVD interrupt Exit from Halt capability in Section 6.4.4.1 on page 31</p> <p>V<sub>PP</sub> absolute max changed from 14 to 13V in Section 12.2 on page 155.</p> <p>Modified I<sub>DD</sub> max values in Section 12.4 on page 159</p> <p>Removed description of low and medium thresholds. Updated LVD min rise time rate. Added note and figure on LVD startup behaviour in Section 12.3 on page 156</p> <p>Updated ADC accuracy data and modified note on negative current injection in Section 10.9 on page 144</p> <p>Updated PLL characteristics Section 12.5.6 on page 168</p> <p>External Reset stretch min value changed to 0. in Section 12.9 on page 178</p> <p>Changed presentation of Option bytes: Byte 0 is displayed left of Byte 1. Option byte default value changed (AVD/LVD on) in section Section 14.1 on page 191</p> | Oct 02 |

**Notes:**

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without the express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics

©2002 STMicroelectronics - All Rights Reserved.

Purchase of I<sup>2</sup>C Components by STMicroelectronics conveys a license under the Philips I<sup>2</sup>C Patent. Rights to use these components in an I<sup>2</sup>C system is granted provided that the system conforms to the I<sup>2</sup>C Standard Specification as defined by Philips.

STMicroelectronics Group of Companies

Australia - Brazil - Canada - China - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan  
Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - U.S.A.

<http://www.st.com>