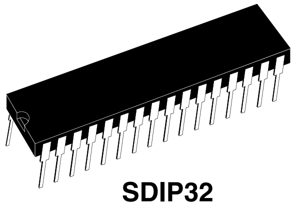
Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

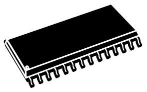
ST72104Gx, ST72215Gx, ST72216Gx, ST72254Gx

8-BIT MCU WITH SINGLE VOLTAGE FLASH MEMORY, ADC, 16-BIT TIMERS, SPI, I^{2}C INTERFACES

- **Memories**
  - 4K or 8K bytes Program memory (ROM and single voltage FLASH) with read-out protection and in-situ programming (remote ISP)
  - 256 bytes RAM
- **Clock, Reset and Supply Management**
  - Enhanced reset system
  - Enhanced low voltage supply supervisor with 3 programmable levels
  - Clock sources: crystal/ceramic resonator oscillators or RC oscillators, external clock, backup Clock Security System
  - Clock-out capability
  - 3 Power Saving Modes: Halt, Wait and Slow
- **Interrupt Management**
  - 7 interrupt vectors plus TRAP and RESET
  - 22 external interrupt lines (on 2 vectors)
- **22 I/O Ports**
  - 22 multifunctional bidirectional I/O lines
  - 14 alternate function lines
  - 8 high sink outputs
- **3 Timers**
  - 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 (one only on ST72104Gx and ST72216G1)
- **2 Communications Interfaces**
  - SPI synchronous serial interface
  - I2C multimaster interface (only on ST72254Gx)
- **1 Analog peripheral**
  - 8-bit ADC with 6 input channels (except on ST72104Gx)



SDIP32



SO28

- **Instruction Set**
  - 8-bit data manipulation
  - 63 basic instructions
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instruction
  - True bit manipulation
- **Development Tools**
  - Full hardware/software development package

Device Summary

Features	ST72104G1	ST72104G2	ST72216G1	ST72215G2	ST72254G1	ST72254G2
Program memory - bytes	4K	8K	4K	8K	4K	8K
RAM (stack) - bytes	256 (128)					
Peripherals	Watchdog timer, One 16-bit timer, SPI		Watchdog timer, One 16-bit timer, SPI, ADC	Watchdog timer, Two 16-bit timers, SPI, ADC	Watchdog timer, Two 16-bit timers, SPI, I^{2}C, ADC	
Operating Supply	3.2V to 5.5V					
CPU Frequency	Up to 8 MHz (with oscillator up to 16 MHz)					
Operating Temperature	0°C to 70°C / -10°C to +85°C (-40°C to +85°C / -40°C to105°C / -40°C to 125°C optional)					
Packages	SO28 / SDIP32					

Table of Contents

Table of Contents

1 INTRODUCTION

The ST72104G, ST72215G, ST72216G and ST72254G devices are members of the ST7 microcontroller family. They can be grouped as follows:

– ST72254G devices are designed for mid-range applications with ADC and I²C interface capabilities.

– ST72215/6G devices target the same range of applications but without I²C interface.

– ST72104G devices are for applications that do not need ADC and I²C peripherals.

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

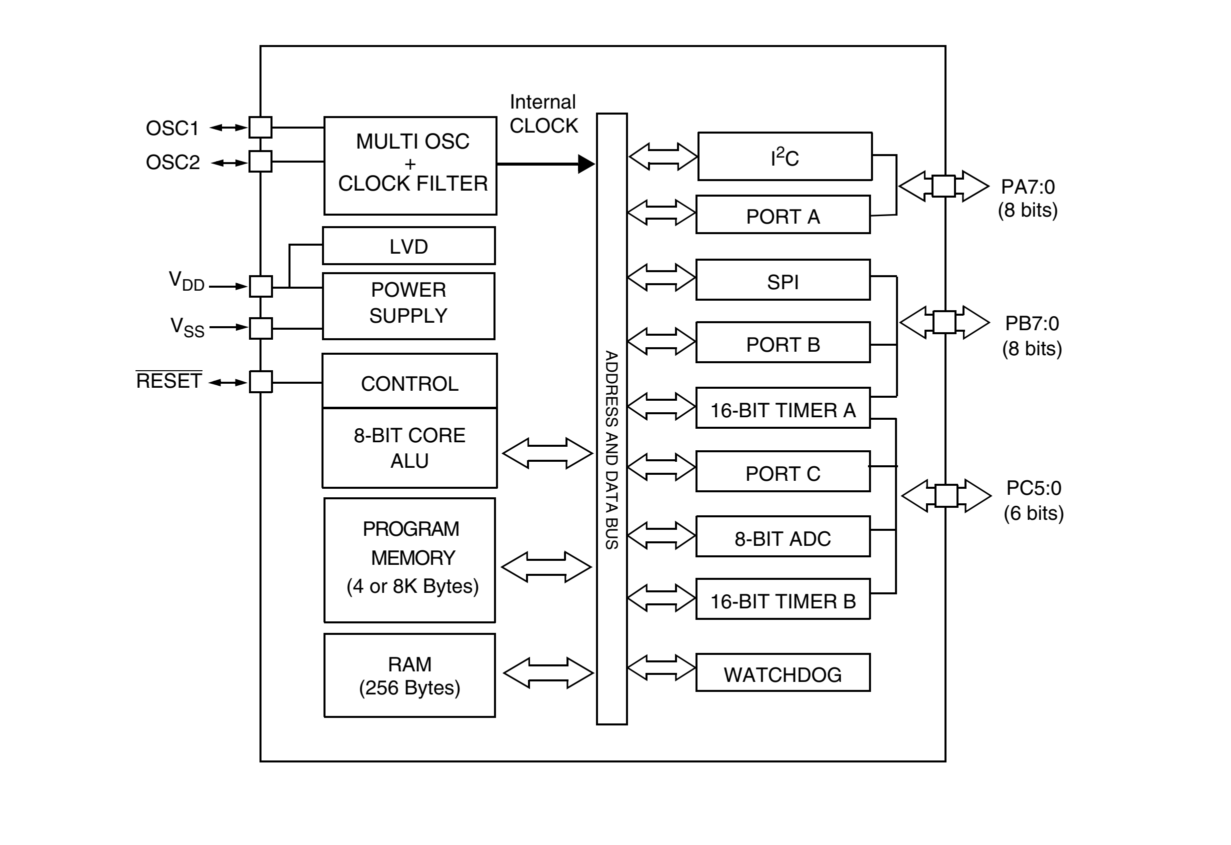
The ST72C104G, ST72C215G, ST72C216G and ST72C254G versions feature single-voltage FLASH memory with byte-by-byte In-Situ Programming (ISP) capability.

Under software control, all devices can be placed in WAIT, SLOW, 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.

For easy reference, all parametric data are located in Section 13 on page 96.

Figure 1. General Block Diagram



2 PIN DESCRIPTION

Figure 2. 28-Pin SO Package Pinout

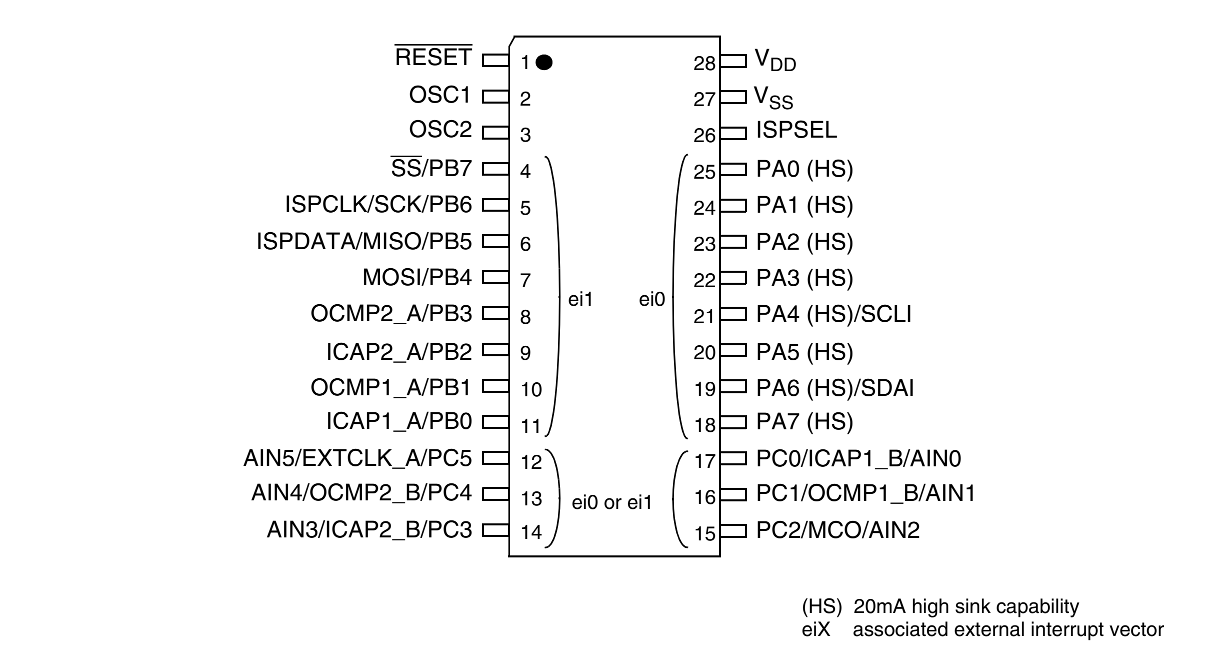
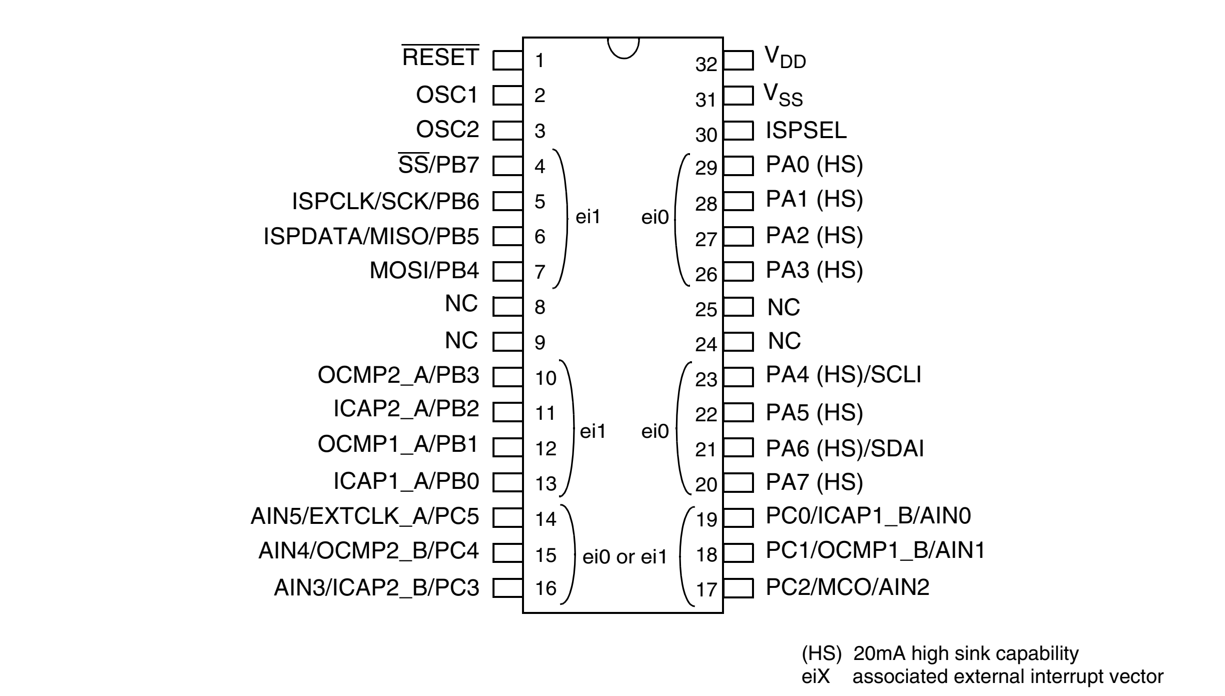

RESET 1 ● 28 VDD
OSC1 2 27 VSS
OSC2 3 26 ISPSEL
SS/PB7 4 25 PA0 (HS)
ISPCLK/SCK/PB6 5 24 PA1 (HS)
ISPDATA/MISO/PB5 6 23 PA2 (HS)
MOSI/PB4 7 22 PA3 (HS)
OCMP2_A/PB3 8 21 PA4 (HS)/SCLI
ICAP2_A/PB2 9 20 PA5 (HS)
OCMP1_A/PB1 10 19 PA6 (HS)/SDAI
ICAP1_A/PB0 11 18 PA7 (HS)
AIN5/EXTCLK_A/PC5 12 17 PC0/ICAP1_B/AIN0
AIN4/OCMP2_B/PC4 13 16 PC1/OCMP1_B/AIN1
AIN3/ICAP2_B/PC3 14 15 PC2/MCO/AIN2
ei1
ei0
ei0 or ei1
(HS) 20mA high sink capability
eiX associated external interrupt vector

Figure 3. 32-Pin SDIP Package Pinout


RESET 1 32 VDD
OSC1 2 31 VSS
OSC2 3 30 ISPSEL
SS/PB7 4 29 PA0 (HS)
ISPCLK/SCK/PB6 5 28 PA1 (HS)
ISPDATA/MISO/PB5 6 27 PA2 (HS)
MOSI/PB4 7 26 PA3 (HS)
NC 8 25 NC
NC 9 24 NC
OCMP2_A/PB3 10 23 PA4 (HS)/SCLI
ICAP2_A/PB2 11 22 PA5 (HS)
OCMP1_A/PB1 12 21 PA6 (HS)/SDAI
ICAP1_A/PB0 13 20 PA7 (HS)
AIN5/EXTCLK_A/PC5 14 19 PC0/ICAP1_B/AIN0
AIN4/OCMP2_B/PC4 15 18 PC1/OCMP1_B/AIN1
AIN3/ICAP2_B/PC3 16 17 PC2/MCO/AIN2
ei1
ei0
ei1
ei0
ei0 or ei1
(HS) 20mA high sink capability
eiX associated external interrupt vector

PIN DESCRIPTION (Cont'd)

For external pin connection guidelines, refer to Section 13 "ELECTRICAL CHARACTERISTICS" on page 96.

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

Port and control configuration:

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

Refer to Section 9 "I/O PORTS" on page 30 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 / Control						Main Function (after reset)	Alternate Function
SDIP32	SO28			Input	Output	Input				Output			
						float	wpu	int	ana	OD	PP		
1	1	$\overline{\text{RESET}}$	I/O	C_T			X			X		Top priority non maskable interrupt (active low)	
2	2	OSC1 [3]	I									External clock input or Resonator oscillator inverter input or resistor input for RC oscillator	
3	3	OSC2 [3]	O									Resonator oscillator inverter output or capacitor input for RC oscillator	
4	4	PB7/$\overline{\text{SS}}$	I/O	C_T		**X**	ei1			X	X	**Port B7**	SPI Slave Select (active low)
5	5	PB6/SCK/ISPCLK	I/O	C_T		**X**	ei1			X	X	**Port B6**	SPI Serial Clock or ISP Clock
6	6	PB5/MISO/ISPDATA	I/O	C_T		**X**	ei1			X	X	**Port B5**	SPI Master In/ Slave Out Data or ISP Data
7	7	PB4/MOSI	I/O	C_T		**X**	ei1			X	X	**Port B4**	SPI Master Out / Slave In Data
8		NC	Not Connected										
9		NC											
10	8	PB3/OCMP2_A	I/O	C_T		**X**	ei1			X	X	**Port B3**	Timer A Output Compare 2
11	9	PB2/ICAP2_A	I/O	C_T		**X**	ei1			X	X	**Port B2**	Timer A Input Capture 2
12	10	PB1 /OCMP1_A	I/O	C_T		**X**	ei1			X	X	**Port B1**	Timer A Output Compare 1
13	11	PB0 /ICAP1_A	I/O	C_T		**X**	ei1			X	X	**Port B0**	Timer A Input Capture 1
14	12	PC5/EXTCLK_A/AIN5	I/O	C_T		**X**	ei0/ei1			X	X	**Port C5**	Timer A Input Clock or ADC Analog Input 5
15	13	PC4/OCMP2_B/AIN4	I/O	C_T		**X**	ei0/ei1			X	X	**Port C4**	Timer B Output Compare 2 or ADC Analog Input 4
16	14	PC3/ ICAP2_B/AIN3	I/O	C_T		**X**	ei0/ei1		X	X	X	**Port C3**	Timer B Input Capture 2 or ADC Analog Input 3
17	15	PC2/MCO/AIN2	I/O	C_T		**X**	ei0/ei1		X	X	X	**Port C2**	Main clock output (f_{CPU}) or ADC Analog Input 2

Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
SDIP32	SO28			Input	Output	Input				Output			
						float	wpu	int	ana	OD	PP		
18	16	PC1/OCMP1_B/AIN1	I/O	C_T		**X**	ei0/ei1		X	X	X	**Port C1**	Timer B Output Compare 1 or ADC Analog Input 1
19	17	PC0/ICAP1_B/AIN0	I/O	C_T		**X**	ei0/ei1		X	X	X	**Port C0**	Timer B Input Capture 1 or ADC Analog Input 0
20	18	PA7	I/O	C_T	HS	**X**	ei0			X	X	**Port A7**	
21	19	PA6 /SDAI	I/O	C_T	HS	**X**		ei0		T		**Port A6**	I^2C Data
22	20	PA5	I/O	C_T	HS	**X**	ei0			X	X	**Port A5**	
23	21	PA4 /SCLI	I/O	C_T	HS	**X**		ei0		T		**Port A4**	I^2C Clock
24		NC	Not Connected										
25		NC											
26	22	PA3	I/O	C_T	HS	**X**	ei0			X	X	**Port A3**	
27	23	PA2	I/O	C_T	HS	**X**	ei0			X	X	**Port A2**	
28	24	PA1	I/O	C_T	HS	**X**	ei0			X	X	**Port A1**	
29	25	PA0	I/O	C_T	HS	**X**	ei0			X	X	**Port A0**	
30	26	ISPSEL	I	C		**X**						In situ programming selection (Should be tied low in standard user mode).	
31	27	V_{SS}	S									Ground	
32	28	V_{DD}	S									Main power supply	

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 Section 9 "I/O PORTS" on page 30 and Section 13.8 "I/O PORT PIN CHARACTERISTICS" on page 118 for more details.

3. OSC1 and OSC2 pins connect a crystal or ceramic resonator, an external RC, or an external source to the on-chip oscillator see Section 2 "PIN DESCRIPTION" on page 7 and Section 13.5 "CLOCK AND TIMING CHARACTERISTICS" on page 105 for more details.

3 REGISTER & MEMORY MAP

As shown in the 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 location, 256 bytes of RAM and up to 8Kbytes of user program memory. The RAM space includes up to 128 bytes for the stack from 0100h to 017Fh.

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

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

Figure 4. Memory Map

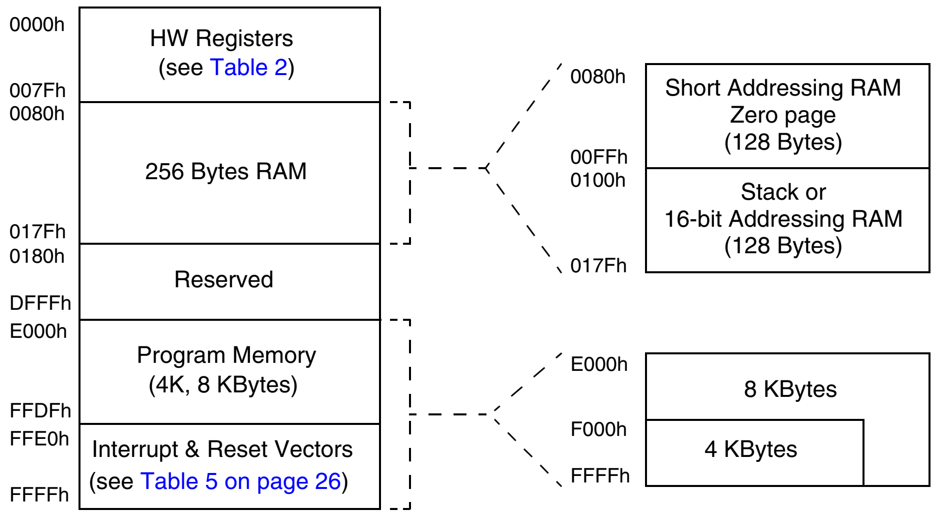


Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h	Port C	PCDR	Port C Data Register	00h [1)]	R/W [2)]
0001h		PCDDR	Port C Data Direction Register	00h	R/W [2)]
0002h		PCOR	Port C Option Register	00h	R/W [2)]
0003h	Reserved (1 Byte)				
0004h	Port B	PBDR	Port B Data Register	00h [1)]	R/W
0005h		PBDDR	Port B Data Direction Register	00h	R/W
0006h		PBOR	Port B Option Register	00h	R/W.
0007h	Reserved (1 Byte)				
0008h	Port A	PADR	Port A Data Register	00h [1)]	R/W
0009h		PADDR	Port A Data Direction Register	00h	R/W
000Ah		PAOR	Port A Option Register	00h	R/W
000Bh to 001Fh	Reserved (21 Bytes)				
0020h		MISCR1	Miscellaneous Register 1	00h	R/W
0021h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
0022h		SPICR	SPI Control Register	0xh	R/W
0023h		SPISR	SPI Status Register	00h	Read Only
0024h	WATCHDOG	WDGCR	Watchdog Control Register	7Fh	R/W
0025h		CRSR	Clock, Reset, Supply Control / Status Register	000x 000x	R/W
0026h 0027h	Reserved (2 bytes)				
0028h	I^{2}C	I2CCR	Control Register	00h	R/W
0029h		I2CSR1	Status Register 1	00h	Read Only
002Ah		I2CSR2	Status Register 2	00h	Read Only
002Bh		I2CCCR	Clock Control Register	00h	R/W
002Ch		I2COAR1	Own Address Register 1	00h	R/W
002Dh		I2COAR2	Own Address Register 2	00h	R/W
002Eh		I2CDR	Data Register	00h	R/W
002Fh to 0030h	Reserved (2 Bytes)				

Address	Block	Register Label	Register Name	Reset Status	Remarks
0031h	TIMER A	TACR2	Timer A Control Register 2	00h	R/W
0032h		TACR1	Timer A Control Register 1	00h	R/W
0033h		TASR	Timer A Status Register	xxh	Read Only
0034h		TAIC1HR	Timer A Input Capture 1 High Register	xxh	Read Only
0035h		TAIC1LR	Timer A Input Capture 1 Low Register	xxh	Read Only
0036h		TAOC1HR	Timer A Output Compare 1 High Register	80h	R/W
0037h		TAOC1LR	Timer A Output Compare 1 Low Register	00h	R/W
0038h		TACHR	Timer A Counter High Register	FFh	Read Only
0039h		TACLR	Timer A Counter Low Register	FCh	Read Only
003Ah		TAACHR	Timer A Alternate Counter High Register	FFh	Read Only
003Bh		TAACLR	Timer A Alternate Counter Low Register	FCh	Read Only
003Ch		TAIC2HR	Timer A Input Capture 2 High Register	xxh	Read Only
003Dh		TAIC2LR	Timer A Input Capture 2 Low Register	xxh	Read Only
003Eh		TAOC2HR	Timer A Output Compare 2 High Register	80h	R/W
003Fh		TAOC2LR	Timer A Output Compare 2 Low Register	00h	R/W
0040h		MISCR2	Miscellaneous Register 2	00h	R/W
0041h	TIMER B	TBCR2	Timer B Control Register 2	00h	R/W
0042h		TBCR1	Timer B Control Register 1	00h	R/W
0043h		TBSR	Timer B Status Register	xxh	Read Only
0044h		TBIC1HR	Timer B Input Capture 1 High Register	xxh	Read Only
0045h		TBIC1LR	Timer B Input Capture 1 Low Register	xxh	Read Only
0046h		TBOC1HR	Timer B Output Compare 1 High Register	80h	R/W
0047h		TBOC1LR	Timer B Output Compare 1 Low Register	00h	R/W
0048h		TBCHR	Timer B Counter High Register	FFh	Read Only
0049h		TBCLR	Timer B Counter Low Register	FCh	Read Only
004Ah		TBACHR	Timer B Alternate Counter High Register	FFh	Read Only
004Bh		TBACLR	Timer B Alternate Counter Low Register	FCh	Read Only
004Ch		TBIC2HR	Timer B Input Capture 2 High Register	xxh	Read Only
004Dh		TBIC2LR	Timer B Input Capture 2 Low Register	xxh	Read Only
004Eh		TBOC2HR	Timer B Output Compare 2 High Register	80h	R/W
004Fh		TBOC2LR	Timer B Output Compare 2 Low Register	00h	R/W
0050h to 006Fh	Reserved (32 Bytes)				
0070h	ADC	ADCDR	Data Register	00h	Read Only
0071h		ADCCSR	Control/Status Register	00h	R/W
0072h to 007Fh	Reserved (14 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

FLASH devices have a single voltage non-volatile FLASH memory that may be programmed in-situ (or plugged in a programming tool) on a byte-by-byte basis.

4.2 MAIN FEATURES

- Remote In-Situ Programming (ISP) mode
- Up to 16 bytes programmed in the same cycle
- MTP memory (Multiple Time Programmable)
- Read-out memory protection against piracy

4.3 STRUCTURAL ORGANISATION

The FLASH program memory is organised in a single 8-bit wide memory block which can be used for storing both code and data constants.

The FLASH program memory is mapped in the upper part of the ST7 addressing space and includes the reset and interrupt user vector area .

4.4 IN-SITU PROGRAMMING (ISP) MODE

The FLASH program memory can be programmed using Remote ISP mode. This ISP mode allows the contents of the ST7 program memory to be updated using a standard ST7 programming tools after the device is mounted on the application board. This feature can be implemented with a minimum number of added components and board area impact.

An example Remote ISP hardware interface to the standard ST7 programming tool is described below. For more details on ISP programming, refer to the ST7 Programming Specification.

Remote ISP Overview

The Remote ISP mode is initiated by a specific sequence on the dedicated ISPSEL pin.

The Remote ISP is performed in three steps:

- Selection of the RAM execution mode
- Download of Remote ISP code in RAM
- Execution of Remote ISP code in RAM to program the user program into the FLASH

Remote ISP hardware configuration

In Remote ISP mode, the ST7 has to be supplied with power (V_{DD} and V_{SS}) and a clock signal (oscillator and application crystal circuit for example).

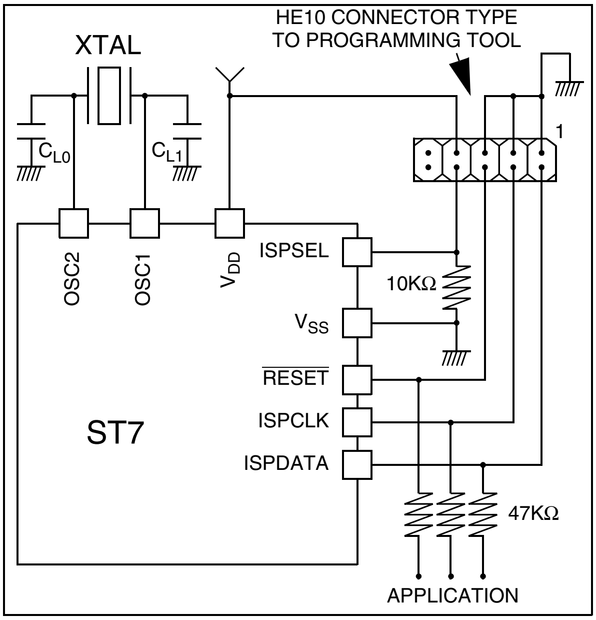
This mode needs five signals (plus the V_{DD} signal if necessary) to be connected to the programming tool. This signals are:

- $\overline{\text{RESET}}$: device reset
- V_{SS}: device ground power supply
- ISPCLK: ISP output serial clock pin
- ISPDATA: ISP input serial data pin
- ISPSEL: Remote ISP mode selection. This pin must be connected to V_{SS} on the application board through a pull-down resistor.

If any of these pins are used for other purposes on the application, a serial resistor has to be implemented to avoid a conflict if the other device forces the signal level.

Figure 5 shows a typical hardware interface to a standard ST7 programming tool. For more details on the pin locations, refer to the device pinout description.

Figure 5. Typical Remote ISP Interface



4.5 MEMORY READ-OUT PROTECTION

The read-out protection is enabled through an option bit.

For FLASH devices, when this option is selected, the program and data stored in the FLASH memory are protected against read-out piracy (including a re-write protection). When this protection option is removed the entire FLASH program memory is first automatically erased. However, the E^2PROM data memory (when available) can be protected only with ROM devices.

5 CENTRAL PROCESSING UNIT

5.1 INTRODUCTION

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

5.2 MAIN FEATURES

- 63 basic instructions
- Fast 8-bit by 8-bit multiply
- 17 main addressing modes
- Two 8-bit index registers
- 16-bit stack pointer
- Low power modes
- Maskable hardware interrupts
- Non-maskable software interrupt

5.3 CPU REGISTERS

The six CPU registers shown in Figure 1 are not present in the memory mapping and are accessed by specific instructions.

Accumulator (A)

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

Index Registers (X and Y)

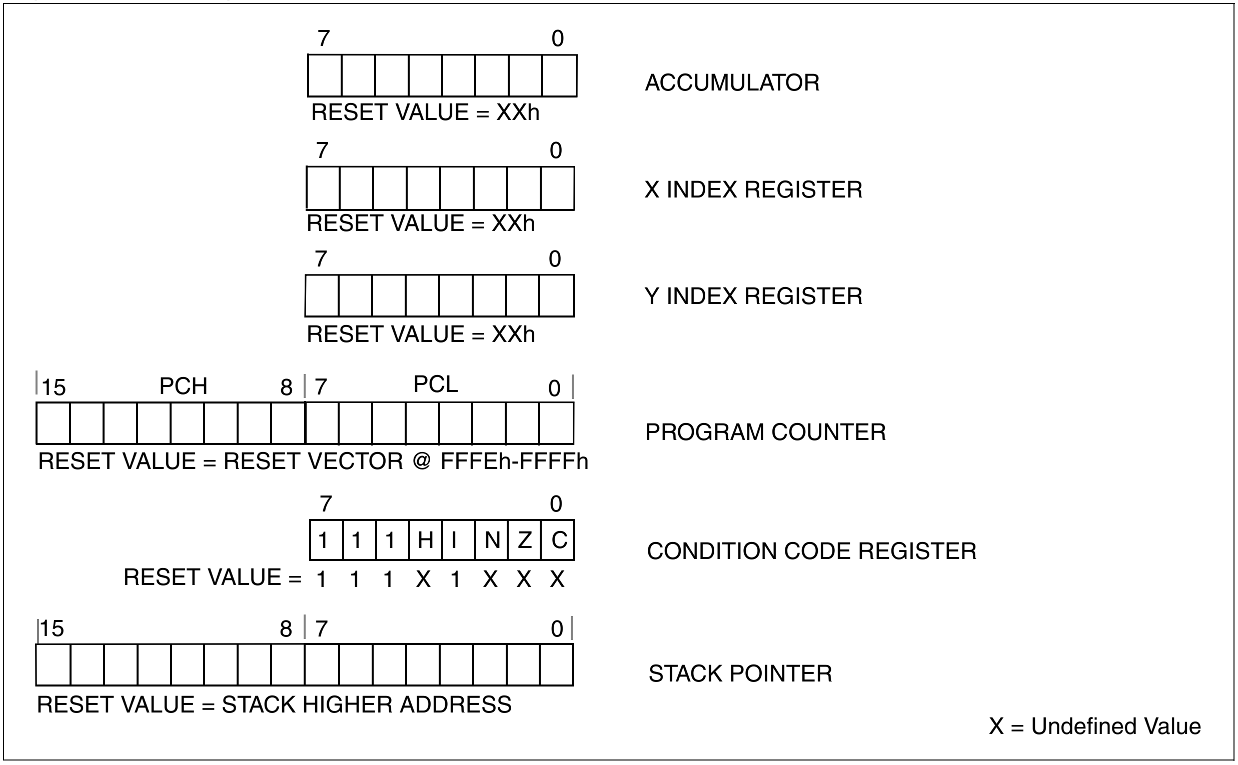
In indexed addressing modes, these 8-bit registers are used to create either effective addresses or temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

The Y register is not affected by the interrupt automatic procedures (not pushed to and popped from the stack).

Program Counter (PC)

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

Figure 6. CPU Registers



CPU REGISTERS (cont'd)

CONDITION CODE REGISTER (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	1	H	I	N	Z	C

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

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

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

This bit is set by hardware when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instruction. It is reset by hardware during the same instructions.
0: No half carry has occurred.
1: A half carry has occurred.

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

Bit 3 = **I** *Interrupt mask*

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

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

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

Bit 2 = **N** *Negative*

This bit is set and cleared by hardware. It is representative of the result sign of the last arithmetic, logical or data manipulation. It is a copy of the 7th bit of the result.
0: The result of the last operation is positive or null.
1: The result of the last operation is negative
(that is, the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

Bit 1 = **Z** *Zero*

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

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

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

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

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

CENTRAL PROCESSING UNIT (Cont'd)

Stack Pointer (SP)

Read/Write

Reset Value: 01 7Fh

15							8
0	0	0	0	0	0	0	1

7							0
0	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see Figure 7).

Since the stack is 128 bytes deep, the 9 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 SP6 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.

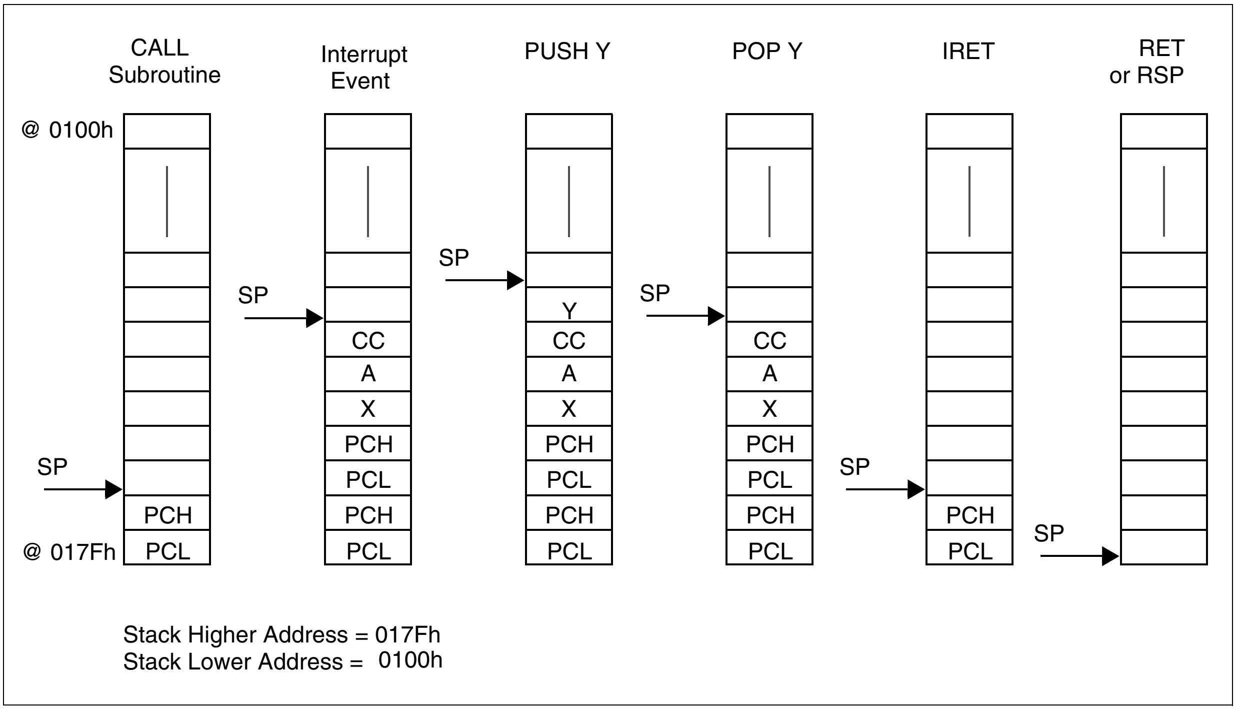
Note: When the lower limit is exceeded, the Stack Pointer wraps around to the stack upper limit, without indicating the stack overflow. The previously stored information is then overwritten and therefore lost. The stack also wraps in case of an underflow.

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 7.

- 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 7. Stack Manipulation Example



6 SUPPLY, RESET AND CLOCK MANAGEMENT

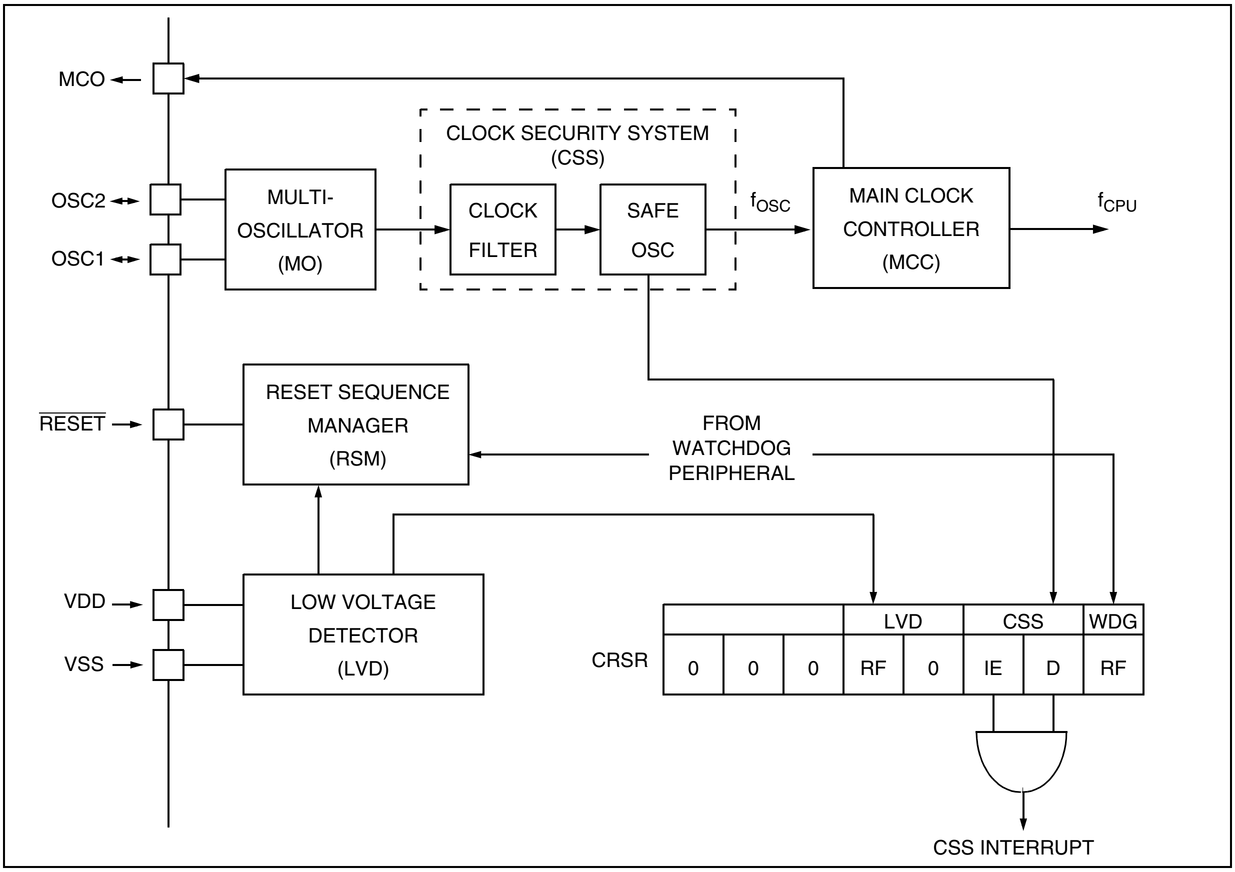
The ST72104G, ST72215G, ST72216G and ST72254G microcontrollers include a range of utility features for securing the application in critical situations (for example in case of a power brown-out), and reducing the number of external components. An overview is shown in Figure 8.

See Section 13 "ELECTRICAL CHARACTERISTICS" on page 96 for more details.

Main Features

- Supply Manager with main supply low voltage detection (LVD)
- Reset Sequence Manager (RSM)
- Multi-Oscillator (MO)
  - 4 Crystal/Ceramic resonator oscillators
  - 1 External RC oscillator
  - 1 Internal RC oscillator
- Clock Security System (CSS)
  - Clock Filter
  - Backup Safe Oscillator

Figure 8. Clock, Reset and Supply Block Diagram



			LVD		CSS		WDG
0	0	0	RF	0	IE	D	RF

6.1 LOW VOLTAGE DETECTOR (LVD)

To allow the integration of power management features in the application, 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 the Figure 9.

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

– under full software control

– in static safe reset

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

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

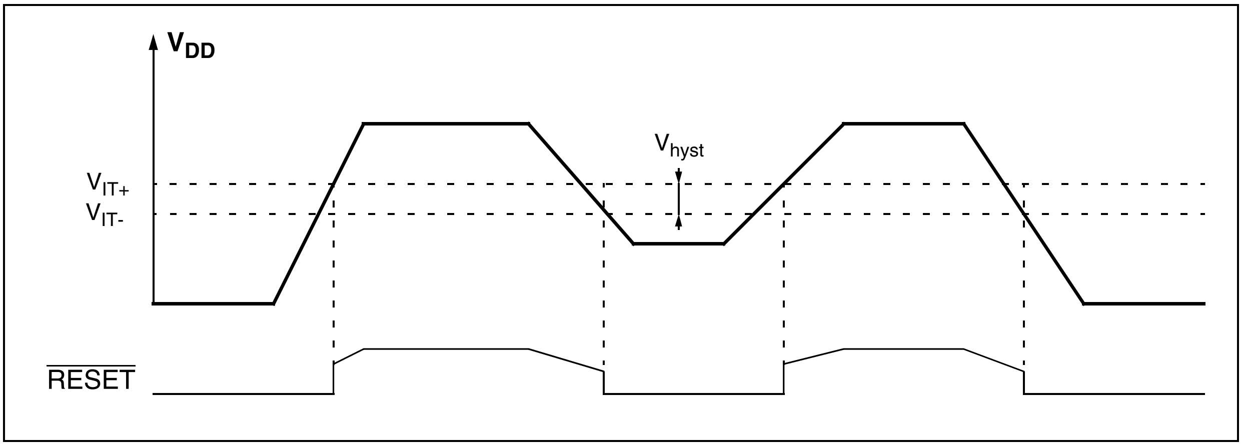
Notes:
1. The LVD allows the device to be used without any external RESET circuitry.
2. Three different reference levels are selectable through the option byte according to the application requirement.

LVD application note

Application software can detect a reset caused by the LVD by reading the LVDRF bit in the CRSR register.

This bit is set by hardware when a LVD reset is generated and cleared by software (writing zero).

Figure 9. Low Voltage Detector vs Reset



6.2 RESET SEQUENCE MANAGER (RSM)

6.2.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 11:

- 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 10:

- Delay depending on the RESET source
- 4096 CPU clock cycle delay
- RESET vector fetch

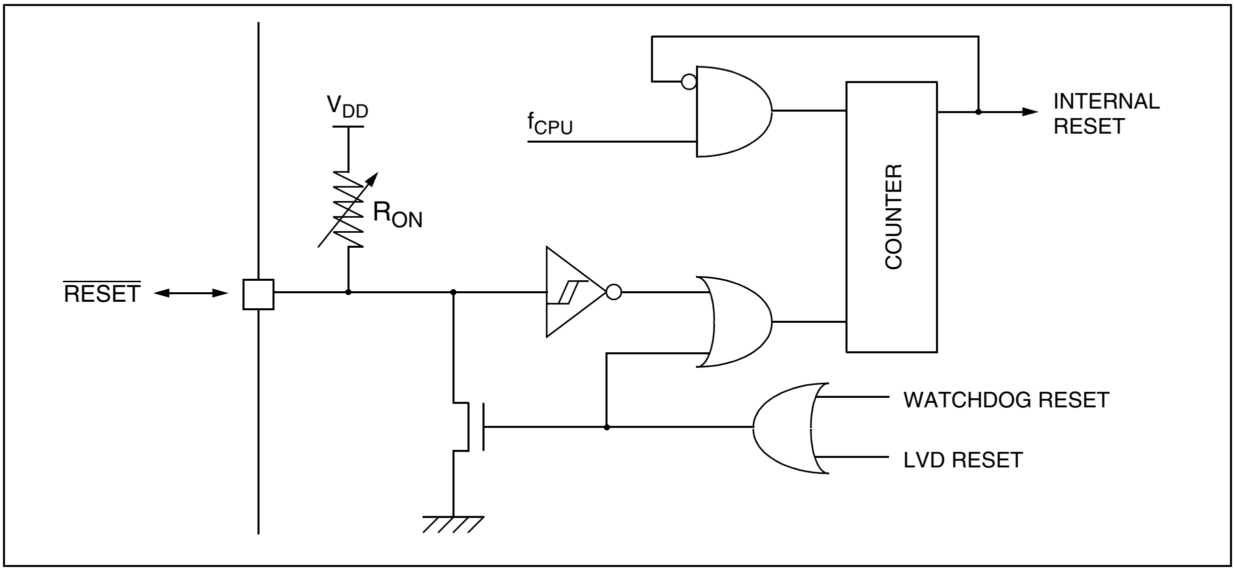
The 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state.

The RESET vector fetch phase duration is 2 clock cycles.

Figure 10. RESET Sequence Phases

RESET		
DELAY	INTERNAL RESET 4096 CLOCK CYCLES	FETCH VECTOR

Figure 11. Reset Block Diagram



RESET SEQUENCE MANAGER (Cont'd)

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

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

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

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.

Two RESET sequences can be associated with this RESET source: short or long external reset pulse (see Figure 12).

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

6.2.3 Internal Low Voltage Detection RESET

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

- Power-On RESET
- Voltage Drop RESET

The device $\overline{\text{RESET}}$ pin acts as an output that is pulled low when $V_{DD}<V_{IT+}$ (rising edge) or $V_{DD}<V_{IT-}$ (falling edge) as shown in Figure 12.

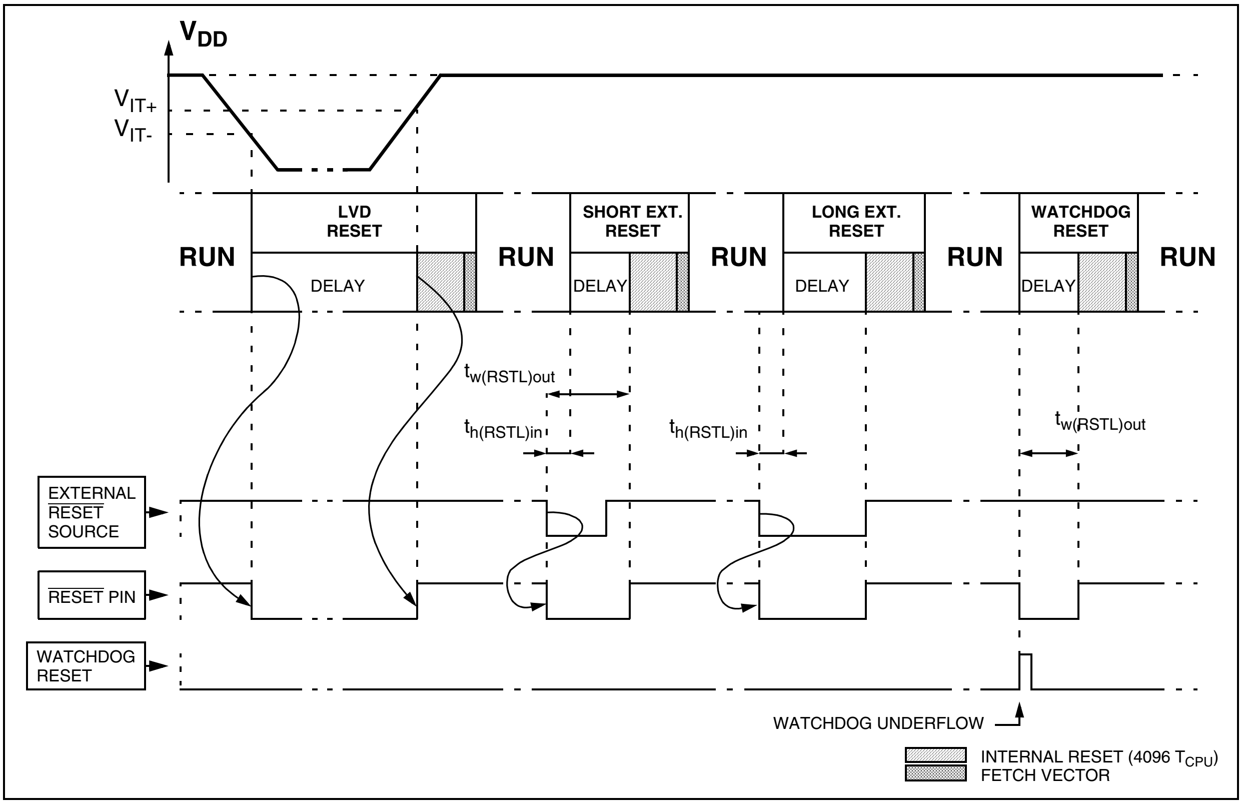
The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

6.2.4 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 12.

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

Figure 12. RESET Sequences



6.3 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 configuration are shown in Table 3. 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. 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

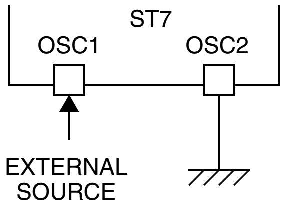
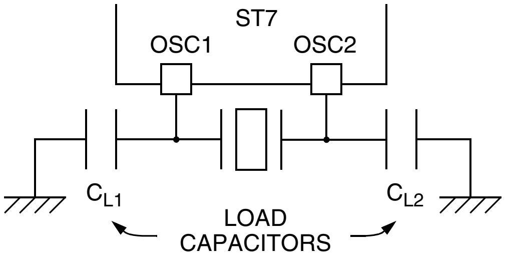
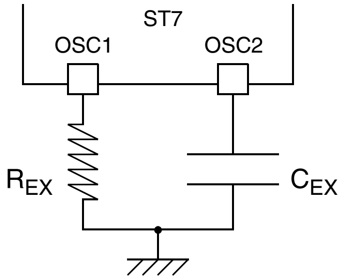
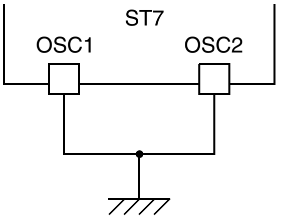
This oscillator allows a low cost solution for the main clock of the ST7 using only an external resistor and an external capacitor. The frequency of the external RC oscillator (in the range of some MHz.) is fixed by the resistor and the capacitor values. Consequently in this MO mode, the accuracy of the clock is dependent on V_{DD}, T_A, process variations and the accuracy of the discrete components used. This option should not be used in applications that require accurate timing.

Internal RC Oscillator

The internal RC oscillator mode is based on the same principle as the external RC oscillator including 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. This option should not be used in applications that require accurate timing.

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

Table 3. ST7 Clock Sources

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

6.4 CLOCK SECURITY SYSTEM (CSS)

The Clock Security System (CSS) protects the ST7 against main clock problems. To allow the integration of the security features in the applications, it is based on a clock filter control and an Internal safe oscillator. The CSS can be enabled or disabled by option byte.

6.4.1 Clock Filter Control

The clock filter is based on a clock frequency limitation function.

This filter function is able to detect and filter high frequency spikes on the ST7 main clock.

If the oscillator is not working properly (e.g. working at a harmonic frequency of the resonator), the current active oscillator clock can be totally filtered, and then no clock signal is available for the ST7 from this oscillator anymore. If the original clock source recovers, the filtering is stopped automatically and the oscillator supplies the ST7 clock.

6.4.2 Safe Oscillator Control

The safe oscillator of the CSS block is a low frequency back-up clock source (see Figure 13).

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

Automatically, the ST7 clock source switches back from the safe oscillator if the original clock source recovers.

Limitation detection

The automatic safe oscillator selection is notified by hardware setting the CSSD bit of the CRSR register. An interrupt can be generated if the CSSIE bit has been previously set.
These two bits are described in the CRSR register description.

6.4.3 Low Power Modes

Mode	Description
WAIT	No effect on CSS. CSS interrupt 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.

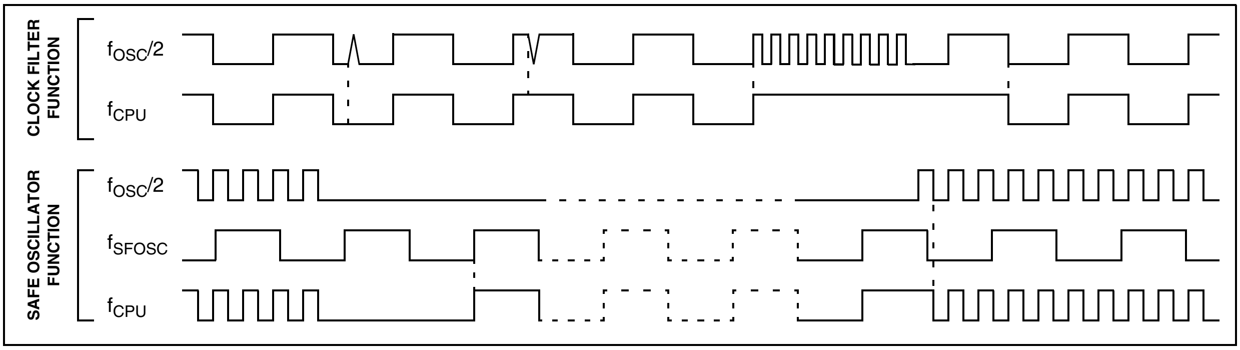
6.4.4 Interrupts

The CSS interrupt event generates an interrupt if the corresponding Enable Control Bit (CSSIE) 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[1)]
CSS event detection (safe oscillator activated as main clock)	CSSD	CSSIE	Yes	No

Note 1: This interrupt allows to exit from active-halt mode if this mode is available in the MCU.

Figure 13. Clock Filter Function and Safe Oscillator Function



6.5 CLOCK RESET AND SUPPLY REGISTER DESCRIPTION (CRSR)

Read/Write

Reset Value: 000x 000x (XXh)

7							0
0	0	0	LVD RF	0	CSS IE	CSS D	WDG RF

Bit 7:5 = **Reserved**, always read as 0.

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**, always read as 0.

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
Refer to Table 5, “Interrupt Mapping,” on page 26 for more details on the CSS interrupt vector. 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_{OSC}). It is set by hardware and cleared by reading the CRSR 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 the CPU starts).
Combined with the LVDRF flag information, the flag description is given by the following table.

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

Application notes

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

Table 4. Clock, Reset and Supply Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0025h	**CRSR** Reset Value	0	0	0	LVDRF x	0	CSSIE 0	CSSD 0	WDGRF x

6.6 MAIN CLOCK CONTROLLER (MCC)

The Main Clock Controller (MCC) supplies the clock for the ST7 CPU and its internal peripherals. It allows SLOW power saving mode to be managed by the application.

All functions are managed by the Miscellaneous register 1 (MISCR1).

The MCC block consists of:

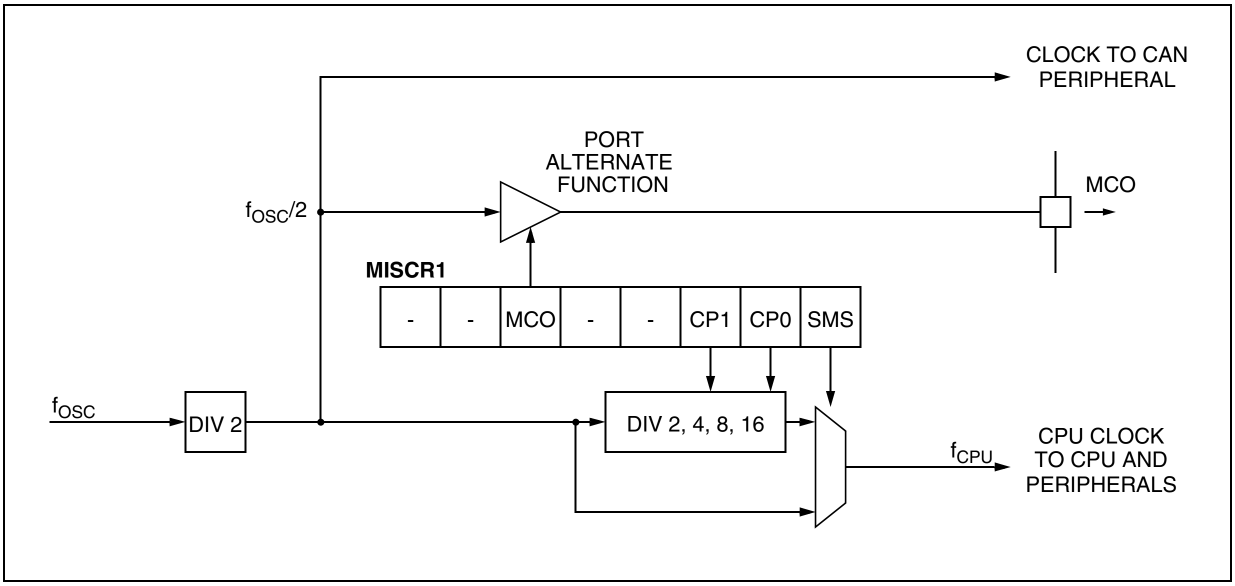
- A programmable CPU clock prescaler
- A clock-out signal to supply external devices

The prescaler allows the selection of the main clock frequency and is controlled by three bits of the MISCR1: CP1, CP0 and SMS.

The clock-out capability consists of a dedicated I/O port pin configurable as an f_{CPU} clock output to drive external devices. It is controlled by the MCO bit in the MISCR1 register.

See Section 10 "MISCELLANEOUS REGISTERS" on page 36 for more details.

Figure 14. Main Clock Controller (MCC) Block Diagram



7 INTERRUPTS

The ST7 core may be interrupted by one of two different methods: maskable hardware interrupts as listed in the Interrupt Mapping Table and a non-maskable software interrupt (TRAP). The Interrupt processing flowchart is shown in Figure 1.

The maskable interrupts must be enabled by clearing the I bit in order to be serviced. However, disabled interrupts may be latched and processed when they are enabled (see external interrupts subsection).

Note: After reset, all interrupts are disabled.

When an interrupt 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.

– The I bit of the CC register is set to prevent additional interrupts.

– 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 the Interrupt Mapping Table for vector addresses).

The interrupt service routine should finish 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 I bit will be cleared and the main program will resume.

Priority Management

By default, a servicing interrupt cannot be interrupted because the I bit is set by hardware entering in interrupt routine.

In the case when several interrupts are simultaneously pending, an hardware priority defines which one will be serviced first (see the Interrupt Mapping Table).

Interrupts and Low Power Mode

All interrupts allow the processor to leave the WAIT low power mode. Only external and specifically mentioned interrupts allow the processor to leave the HALT low power mode (refer to the “Exit from HALT“ column in the Interrupt Mapping Table).

7.1 NON-MASKABLE SOFTWARE INTERRUPT

This interrupt is entered when the TRAP instruction is executed regardless of the state of the I bit. It will be serviced according to the flowchart on Figure 1.

7.2 EXTERNAL INTERRUPTS

External interrupt vectors can be loaded into the PC register if the corresponding external interrupt occurred and if the I bit is cleared. These interrupts allow the processor to leave the Halt low power mode.

The external interrupt polarity is selected through the miscellaneous register or interrupt register (if available).

An external interrupt triggered on edge will be latched and the interrupt request automatically cleared upon entering the interrupt service routine.

If several input pins, connected to the same interrupt vector, are configured as interrupts, their signals are logically NANDed before entering the edge/level detection block.

Caution: The type of sensitivity defined in the Miscellaneous or Interrupt register (if available) applies to the ei source. In case of a NANDed source (as described on the I/O ports section), a low level on an I/O pin configured as input with interrupt, masks the interrupt request even in case of rising-edge sensitivity.

7.3 PERIPHERAL INTERRUPTS

Different peripheral interrupt flags in the status register are able to cause an interrupt when they are active if both:

– The I bit of the CC register is cleared.

– The corresponding enable bit is set in the control register.

If any of these two conditions is false, the interrupt is latched and thus remains pending.

Clearing an interrupt request is done by:

– Writing “0” to the corresponding bit in the status register or

– Access to the status register while the flag is set followed by a read or write of an associated register.

Note: The clearing sequence resets the internal latch. A pending interrupt (that is, waiting to be enabled) will therefore be lost if the clear sequence is executed.