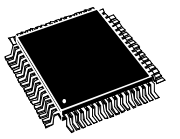
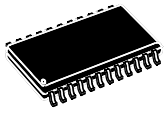
ST7SCR

8-BIT LOW-POWER, FULL-SPEED USB MCU WITH 16K FLASH, 768 RAM, SMARTCARD I/F, TIMER

- **Memories**
  - Up to 16K of ROM or High Density Flash (HDFlash) program memory with read/write protection, HDFlash In-Circuit and In-Application Programming. 100 write/erase cycles guaranteed, data retention: 20 years at 55°C
  - Up to 768 bytes of RAM including up to 128 bytes stack and 256 bytes USB buffer
- **Clock, Reset and Supply Management**
  - Low Voltage Reset
  - 2 power saving modes: Halt and Wait modes
  - PLL for generating 48 MHz USB clock using a 4 MHz crystal
- **Interrupt Management**
  - Nested Interrupt Controller
- **USB (Universal Serial Bus) Interface**
  - 256-byte buffer for full speed bulk, control and interrupt transfer types compliant with USB specification (version 2.0)
  - On-Chip 3.3V USB voltage regulator and transceivers with software power-down
  - 7 USB Endpoints:
    One 8-byte Bidirectional Control Endpoint
    One 64-byte In Endpoint,
    One 64-byte Out Endpoint
    Four 8-byte In Endpoints
- **35 or 4 I/O ports**:
  - Up to 4 LED outputs with software programmable constant current (3 or 7 mA).
  - 2 General purpose I/Os programmable as interrupts
  - Up to 8 line inputs programmable as interrupts
  - Up to 20 Outputs
  - 1 line assigned by default as static input after reset



TQFP64 14x14



SO24

- **ISO7816-3 UART Interface:**
  - 4 Mhz Clock generation
  - Synchronous/Asynchronous protocols (T=0, T=1)
  - Automatic retry on parity error
  - Programmable Baud rate from 372 clock pulses up to 11.625 clock pulses (D=32/F=372)
  - Card Insertion/Removal Detection
- **Smartcard Power Supply:**
  - Selectable card V_{CC} 1.8V, 3V, and 5V
  - Internal Step-up converter for 5V supplied Smartcards (with a current of up to 55mA) using only two external components.
  - Programmable Smartcard Internal Voltage Regulator (1.8V to 3.0V) with current overload protection and 4 KV ESD protection (Human Body Model) for all Smartcard Interface I/Os
- **One 8-bit Timer**
  - Time Base Unit (TBU) for generating periodic interrupts.
- **Development Tools**
  - Full hardware/software development package

Table 1. Device Summary

Features	ST7FSCR1R4	ST7SCR1R4	ST7FSCR1E4	ST7SCR1E4
Program memory	16K FLASH	16K ROM	16K FLASH	16K ROM
User RAM (stack) - bytes	768 (128)			
Peripherals	USB Full-Speed (7 Ep), TBU, Watchdog timer, ISO7816-3 Interface			
Operating Supply	4.0 to 5.5V			
Package	TQFP64		SO24	
CPU Frequency	4 or 8 Mhz			
Operating temperature	0°C to +70°C			

Rev. 2.0

Table of Contents

Table of Contents

To obtain the most recent version of this datasheet,
please check at www.st.com>products>technical literature>datasheet
Please also pay special attention to the Section “IMPORTANT NOTES” on page 99

1 INTRODUCTION

The ST7SCR and ST7FSCR devices are members of the ST7 microcontroller family designed for USB applications. All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

The ST7SCR ROM devices are factory-programmed and are not reprogrammable.

The ST7FSCR versions feature dual-voltage Flash memory with Flash Programming capability.

They operate at a 4MHz external oscillator frequency.

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

The devices include an ST7 Core, up to 16 Kbytes of program memory, up to 512 bytes of user RAM, up to 35 I/O lines and the following on-chip peripherals:

– USB full speed interface with 7 endpoints, programmable in/out configuration and embedded 3.3V voltage regulator and transceivers (no external components are needed).

– ISO7816-3 UART interface with Programmable Baud rate from 372 clock pulses up to 11.625 clock pulses

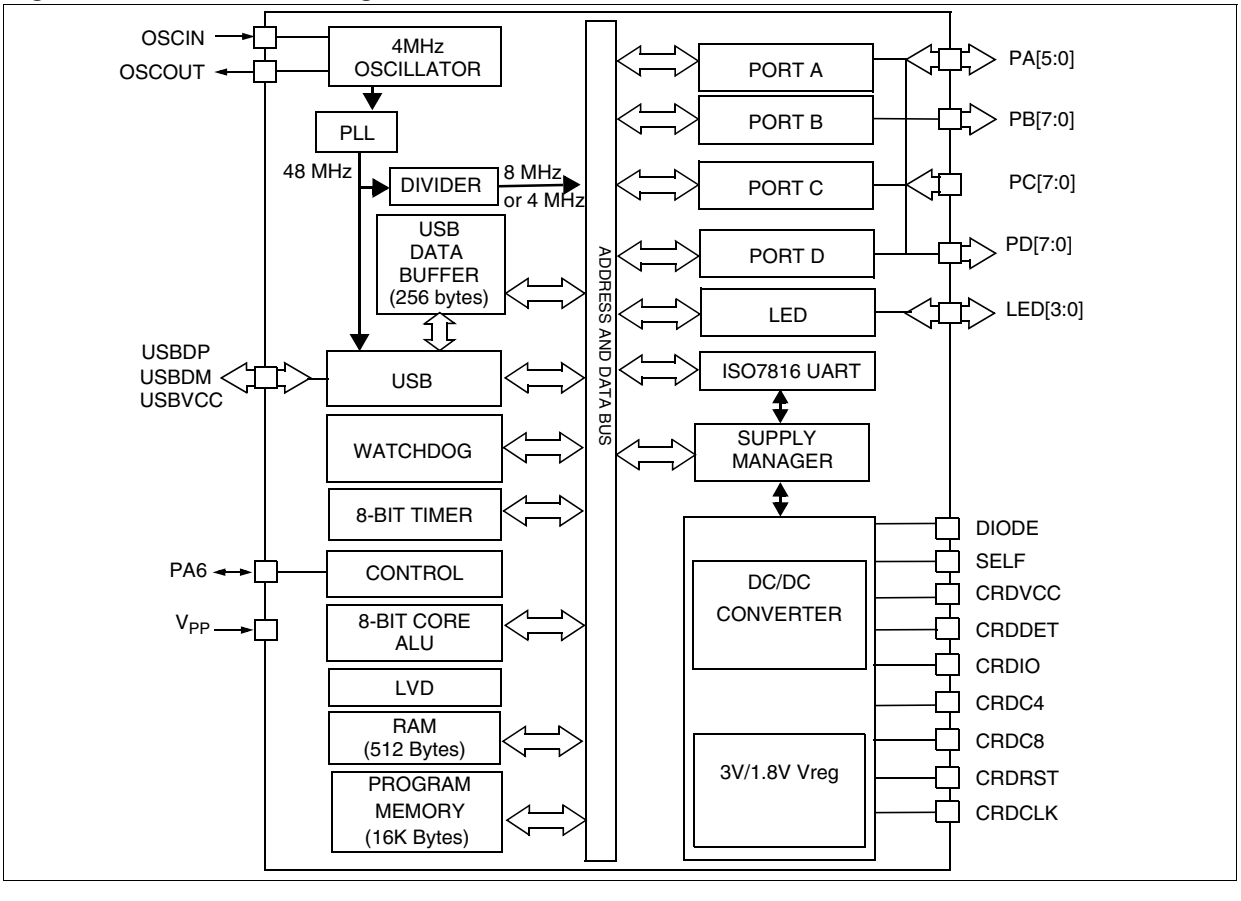
– Smartcard Supply Block able to provide programmable supply voltage and I/O voltage levels to the smartcards

– Low voltage reset ensuring proper power-on or power-off of the device (selectable by option)

– Watchdog Timer

– 8-bit Timer (TBU)

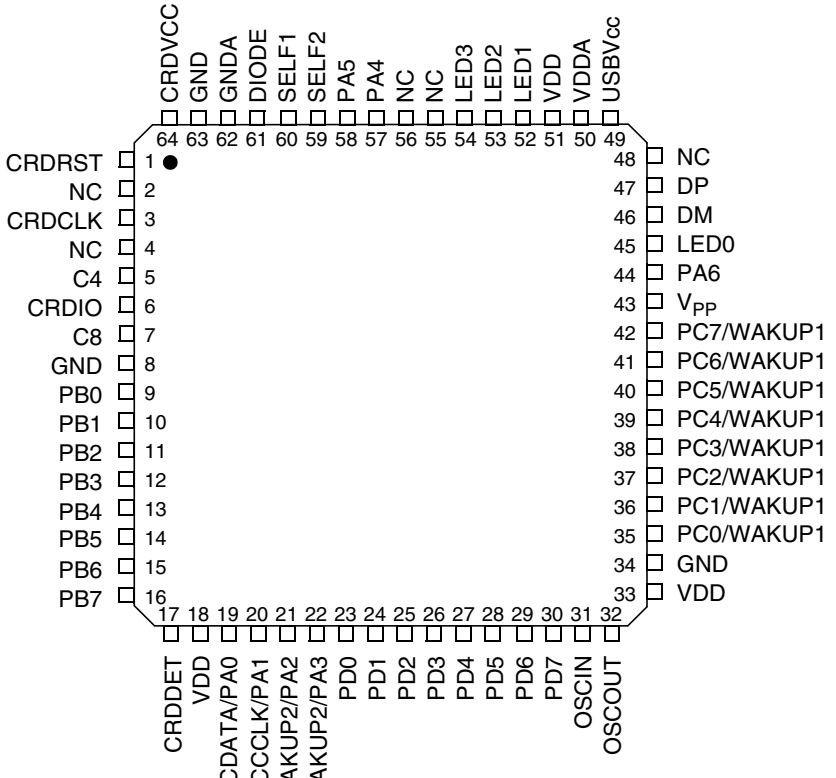
Figure 1. ST7SCR Block Diagram



2 PIN DESCRIPTION

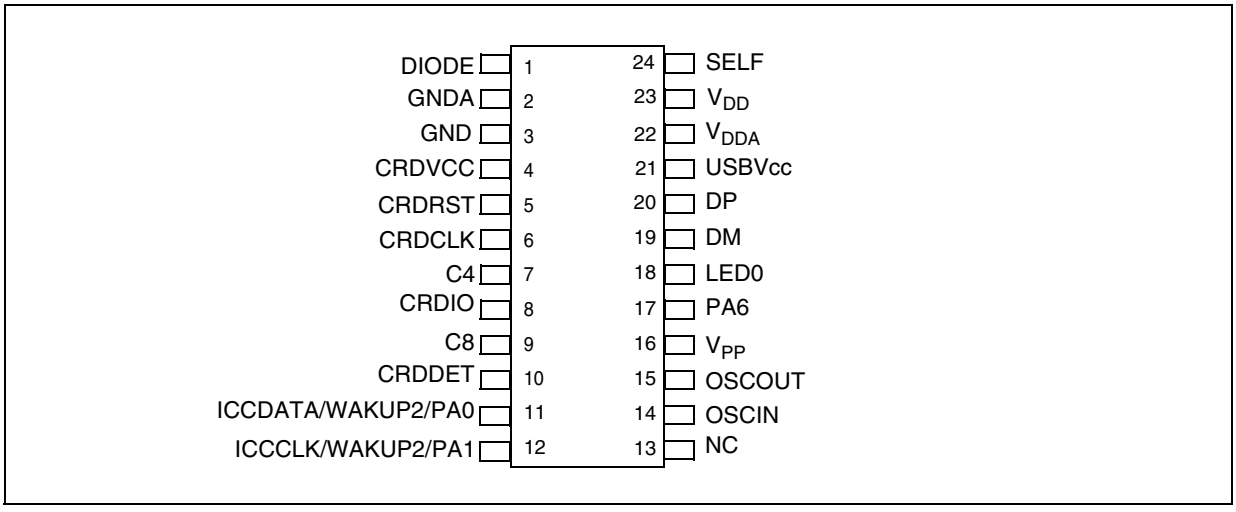
Figure 2. 64-Pin TQFP Package Pinout

NC = Not Connected



PIN DESCRIPTION (Cont'd)

Figure 3. 24-Pin SO Package Pinout



PIN DESCRIPTION (Cont'd)

Legend / Abbreviations:

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

In/Output level: C_T = CMOS $0.3V_{DD}/0.7V_{DD}$ with input trigger

Output level: HS = 10mA high sink (on N-buffer only)

Port and control configuration:

– Input:float = floating, wpu = weak pull-up, int = interrupt, ana = analog

– Output: OD = open drain, PP = push-pull

Refer to "I/O PORTS" on page 32 for more details on the software configuration of the I/O ports.

Table 2. Pin Description

Pin n°		Pin Name	Type	Level		V_{CARD} supplied	Port / Control				Main Function (after reset)	Alternate Function
							Input		Output			
TQFP64	SO24			Input	Output		wpu	int	OD	PP		
1	5	CRDRST	O		C_T	X				X	Smartcard Reset	
2		NC									Not Connected	
3	6	CRDCLK	O		C_T	X				X	Smartcard Clock	
4		NC									Not Connected	
5	7	C4	O		C_T	X				X	Smartcard C4	
6	8	CRDIO	I/O	C_T		X	X		X		Smartcard I/O	
7	9	C8	O		C_T	X				X	Smartcard C8	
8	3	GND	S								Ground	
9		PB0	O		C_T				X	X	Port B0 [1]	
10		PB1	O		C_T				X	X	Port B1 [1]	
11		PB2	O		C_T				X	X	Port B2 [1]	
12		PB3	O		C_T				X	X	Port B3 [1]	
13		PB4	O		C_T				X	X	Port B4 [1]	
14		PB5	O		C_T				X	X	Port B5 [1]	
15		PB6	O		C_T				X	X	Port B6 [1]	
16		PB7	O		C_T				X	X	Port B7 [1]	
17	10	CRDDET	I	C_T			X				Smartcard Detection	
18		VDD	S								Power Supply voltage 4V-5.5V	
19	11	PA0/WAKUP2/ ICCDATA	I/O	C_T			X	X	X	X	Port A0	Interrupt, In-Circuit Communication Data Input
20	12	PA1/WAKUP2/ ICCCLK	I/O	C_T			X	X	X	X	Port A1	Interrupt, In-Circuit Communication Clock Input
21		PA2/WAKUP2	I/O	C_T			X	X	X	X	Port A2 [1]	Interrupt
22		PA3/WAKUP2	I/O	C_T			X	X	X	X	Port A3 [1]	Interrupt
23		PD0	O		C_T				X	X	Port D0 [1]	
24		PD1	O		C_T				X	X	Port D1 [1]	
25		PD2	O		C_T				X	X	Port D2 [1]	

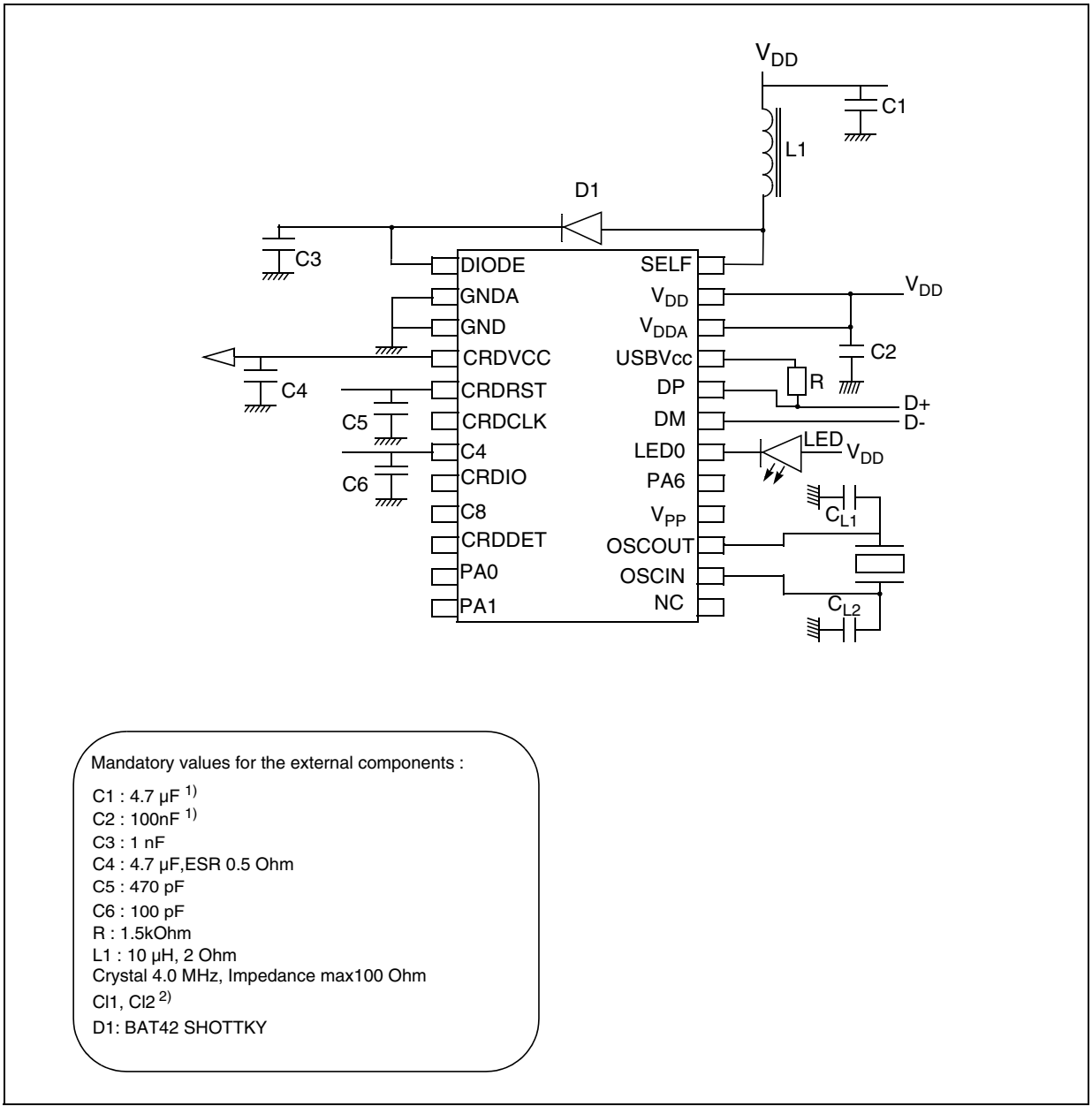
Pin n°		Pin Name	Type	Level		V_{CARD} supplied	Port / Control				Main Function (after reset)	Alternate Function
							Input		Output			
TQFP64	SO24			Input	Output		wpu	int	OD	PP		
26		PD3	O		C_T				X	X	Port D3 1)	
27		PD4	O		C_T				X	X	Port D4 1)	
28		PD5	O		C_T				X	X	Port D5 1)	
29		PD6	O		C_T				X	X	Port D6 1)	
30		PD7	O		C_T				X	X	Port D7 1)	
31	14	OSCIN		C_T							Input/Output Oscillator pins. These pins connect a 4MHz parallel-resonant crystal, or an external source to the on-chip oscillator.	
32	15	OSCOUT			C_T							
33		VDD	S								Power Supply voltage 4V-5.5V	
34		GND	S								Ground	
35		PC0/WAKUP1	I	C_T			X	X			PC0 1)	External interrupt
36		PC1/WAKUP1	I	C_T			X	X			PC1 1)	External interrupt
37		PC2/WAKUP1	I	C_T			X	X			PC2 1)	External interrupt
38		PC3/WAKUP1	I	C_T			X	X			PC3 1)	External interrupt
39		PC4/WAKUP1	I	C_T			X	X			PC4 1)	External interrupt
40		PC5/WAKUP1	I	C_T			X	X			PC5 1)	External interrupt
41		PC6/WAKUP1	I	C_T			X	X			PC6 1)	External interrupt
42		PC7/WAKUP1	I	C_T			X	X			PC7 1)	External interrupt
43	16	V_{PP}	S								Flash programming voltage. Must be held low in normal operating mode.	
44	17	PA6	I	C_T							PA6	
45	18	LED0	O		HS				X		Constant Current Output	
46	19	DM	I/O	C_T							USB Data Minus line	
47	20	DP	I/O	C_T							USB Data Plus line	
48		NC									Not Connected	
49	21	USBVCC	O		C_T						3.3 V Output for USB	
50	22	V_{DDA}	S								power Supply voltage 4V-5.5V	
51	23	V_{DD}	S								power Supply voltage 4V-5.5V	
52		LED1	O		HS				X		Constant Current Output	
53		LED2	O		HS				X		Constant Current Output	
54		LED3	O		HS				X		Constant Current Output	
55		NC									Not Connected	
56		NC									Not Connected	
57		PA4	I/O	C_T			X	X	X	X	Port A4	

Pin n°		Pin Name	Type	Level		V_{CARD} supplied	Port / Control				Main Function (after reset)	Alternate Function
							Input		Output			
TQFP64	SO24			Input	Output		wpu	int	OD	PP		
58		PA5	I/O	C_T			X	X	X	X	Port A5	
59	24	SELF2	O		C_T						An External inductance must be connected to these pins for the step up converter (refer to Figure 4 to choose the right capacitance)	
60	24	SELF1	O		C_T							
61	1	DIODE	S		C_T						An External diode must be connected to this pin for the step up converter (refer to Figure 4 to choose the right component)	
62	2	GNDA	S								Ground	
63	3	GND	S									
64	4	CDRVCC	O		C_T	X					Smartcard Supply pin	

Note 1 : Keyboard interface

PIN DESCRIPTION

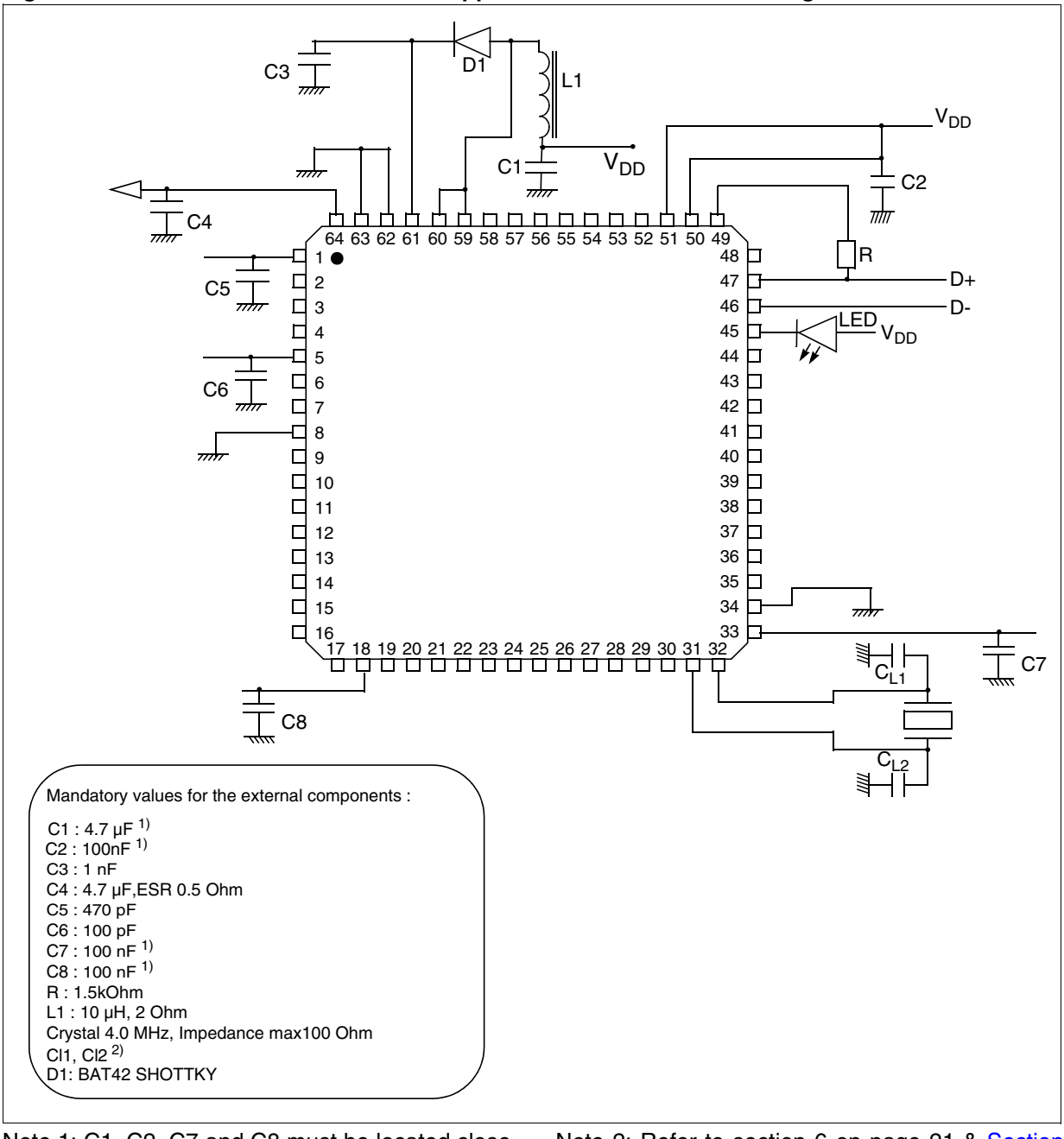
Figure 4. Smartcard Interface Reference Application - 24-Pin SO Package



Note 1: C1 and C2 must be located close to the chip.

Note 2: Refer to section 6 on page 21 & Section 14.4.3 Crystal Resonator Oscillators.

Figure 5. Smartcard Interface Reference Application - 64-Pin TQFP Package



Note 1: C1, C2, C7 and C8 must be located close to the chip.

Note 2: Refer to section 6 on page 21 & Section 14.4.3 Crystal Resonator Oscillators.

3 REGISTER & MEMORY MAP

As shown in Figure 6, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 40 bytes of register locations, up to 512 bytes of RAM and up to 16K bytes 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 noted "Reserved" must never be accessed. Accessing a reserved area can have unpredictable effects on the device.

Figure 6. Memory Map

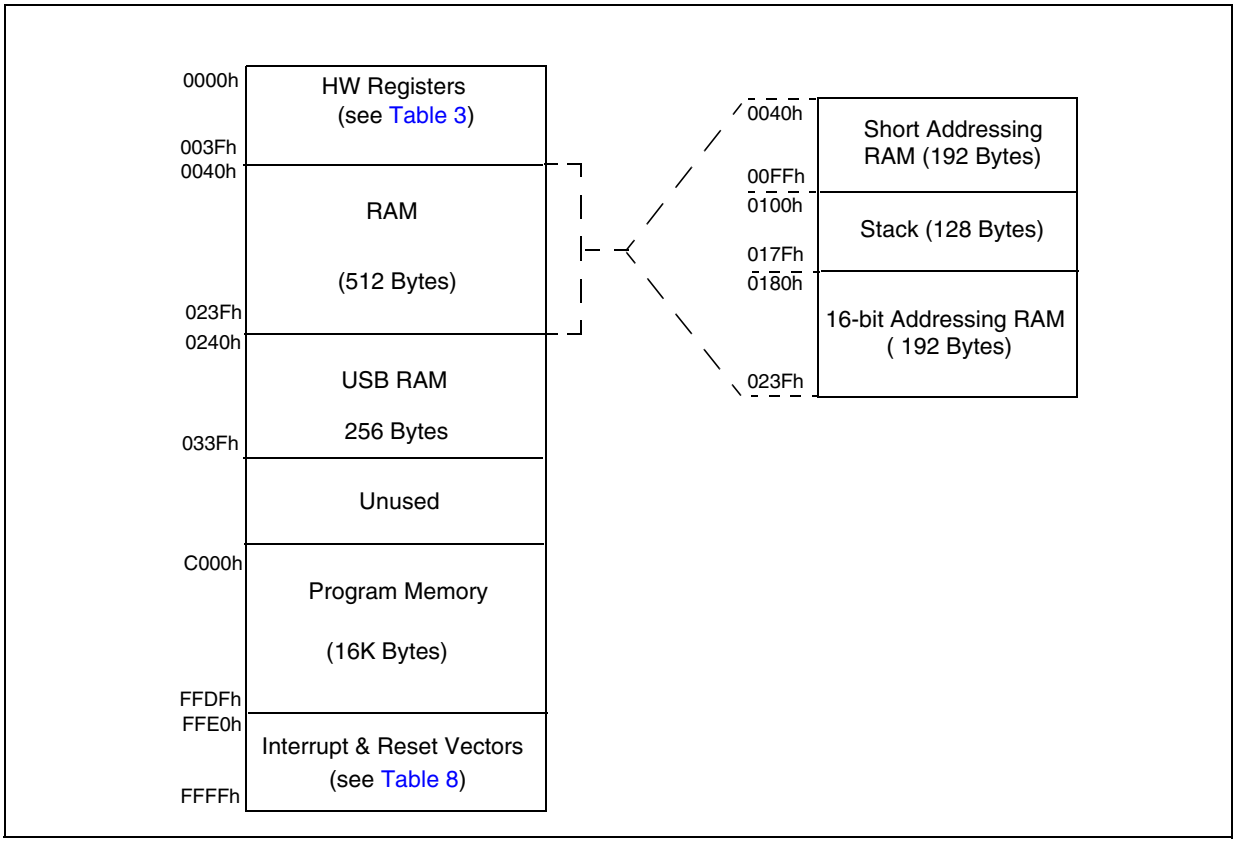


Table 3. Hardware Register Memory Map

Address	Block	Register Label	Register name	Reset Status	Remarks
0000h	CRD	CRDCR	Smartcard Interface Control Register	00h	R/W
0001h		CRDSR	Smartcard Interface Status Register	80h	R/W
0002h		CRDCCR	Smartcard Contact Control Register	xxh	R/W
0003h		CRDETU1	Smartcard Elementary Time Unit 1	01h	R/W
0004h		CRDETU0	Smartcard Elementary Time Unit 0	74h	R/W
0005h		CRDGT1	Smartcard Guard time 1	00h	R/W
0006h		CRDGT0	Smartcard Guard time 0	0Ch	R/W
0007h		CRDWT2	Smartcard Character Waiting Time 2	00h	R/W
0008h		CRDWT1	Smartcard Character Waiting Time 1	25h	R/W
0009h		CRDWT0	Smartcard Character Waiting Time 0	80h	R/W
000Ah		CRDIER	Smartcard Interrupt Enable Register	00h	R/W
000Bh		CRDIPR	Smartcard Interrupt Pending Register	00h	R
000Ch		CRDTXB	Smartcard Transmit Buffer Register	00h	R/W
000Dh		CRDRXB	Smartcard Receive Buffer Register	00h	R
000Eh	Watchdog	WDGCR	Watchdog Control Register	00h	R/W
0011h	Port A	PADR	Port A Data Register	00h	R/W
0012h		PADDR	Port A Data Direction Register	00h	R/W
0013h		PAOR	Option Register	00h	R/W
0014h		PAPUCR	Pull up Control Register	00h	R/W
0015h	Port B	PBDR	Port B Data Register	00h	R/W
0016h		PBOR	Option Register	00h	R/W
0017h		PBPUCR	Pull up Control Register	00h	R/W
0018h	Port C	PCDR	Port C Data Register	00h	R/W
0019h	Port D	PDDR	Port D Data Register	00h	R/W
001Ah		PDOR	Option Register	00h	R/W
001Bh		PDPUCR	Pull up Control Register	00h	R/W
001Ch	MISC	MISCR1	Miscellaneous Register 1	00h	R/W
001Dh		MISCR2	Miscellaneous Register 2	00h	R/W
001Eh		MISCR3	Miscellaneous Register 3	00h	R/W
001Fh		MISCR4	Miscellaneous Register 4	00h	R/W

Address	Block	Register Label	Register name	Reset Status	Remarks
0020h	USB	USBISTR	USB Interrupt Status Register	00h	R/W
0021h		USBIMR	USB Interrupt Mask Register	00h	R/W
0022h		USBCTLR	USB Control Register	06h	R/W
0023h		DADDR	Device Address Register	00h	R/W
0024h		USBSR	USB Status Register	00h	R/W
0025h		EPOR	Endpoint 0 Register	0xh	R/W
0026h		CNT0RXR	EP 0 ReceptionCounter Register	00h	R/W
0027h		CNT0TXR	EP 0 Transmission Counter Register	00h	R/W
0028h		EP1TXR	EP 1 Transmission Register	00h	R/W
0029h		CNT1TXR	EP 1 Transmission Counter Register	00h	R/W
002Ah		EP2RXR	EP 2 Reception Register	00h	R/W
002Bh		CNT2RXR	EP 2 Reception Counter Register	0xh	R/W
002Ch		EP2TXR	EP 2 Transmission Register	00h	R/W
002Dh		CNT2TXR	EP 2 Transmission Counter Register	00h	R/W
002Eh		EP3TXR	EP 3 Transmission Register	00h	R/W
002Fh		CNT3TXR	EP 3 Transmission Counter Register	00h	R/W
0030h		EP4TXR	EP 4 Transmission Register	00h	R/W
0031h		CNT4TXR	EP 4 Transmission Counter Register	00h	R/W
0032h		EP5TXR	EP 5 Transmission Register	00h	R/W
0033h		CNT5TXR	EP 5 Transmission Counter Register	00h	R/W
0034h		ERRSR	Error Status Register	00h	R/W
0035h	TBU	TBUCV	Timer counter value	00h	R/W
0036h		TBUCSR	Timer control status	00h	R/W
0037h	ITC	ITSPR0	Interrupt Software Priority Register 0	FFh	R/W
0038h		ITSPR1	Interrupt Software Priority Register 1	FFh	R/W
0039h		ITSPR2	Interrupt Software Priority Register 2	FFh	R/W
003Ah		ITSPR3	Interrupt Software Priority Register 3	FFh	R/W
003Bh	Flash	FCSR	Flash Control Status Register	00h	R/W
003Eh		LED_CTRL	LED Control Register	00h	R/W

4 FLASH PROGRAM MEMORY

4.1 Introduction

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

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

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

4.2 Main Features

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

4.3 Structure

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

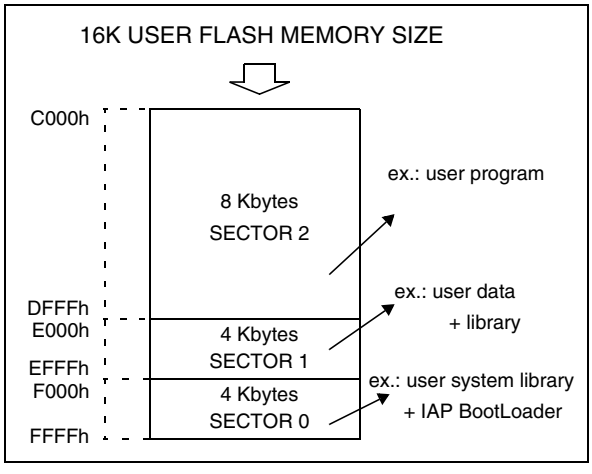
Depending on the overall FLASH memory size in the microcontroller device, there are up to three user sectors (see Table 4). Each of these sectors can be erased independently to avoid unnecessary erasing of the whole Flash memory when only a partial erasing is required.

The first two sectors have a fixed size of 4 Kbytes (see Figure 7). They are mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

Table 4. Sectors available in FLASH devices

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

Figure 7. Memory map and sector address



FLASH PROGRAM MEMORY (Cont'd)

4.4 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 8). For more details on the pin locations, refer to the device pinout description.

ICP needs six signals to be connected to the programming tool. These signals are:

- V_{SS}: device power supply ground
- V_{DD}: for reset by LVD
- OSCIN: to force the clock during power-up
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input serial data pin
- V_{PP}: ICC mode selection and programming voltage.

If ICCCLK or ICCDATA are used for other purposes in the application, a serial resistor has to be implemented to avoid a conflict in case one of the other devices forces the signal level.

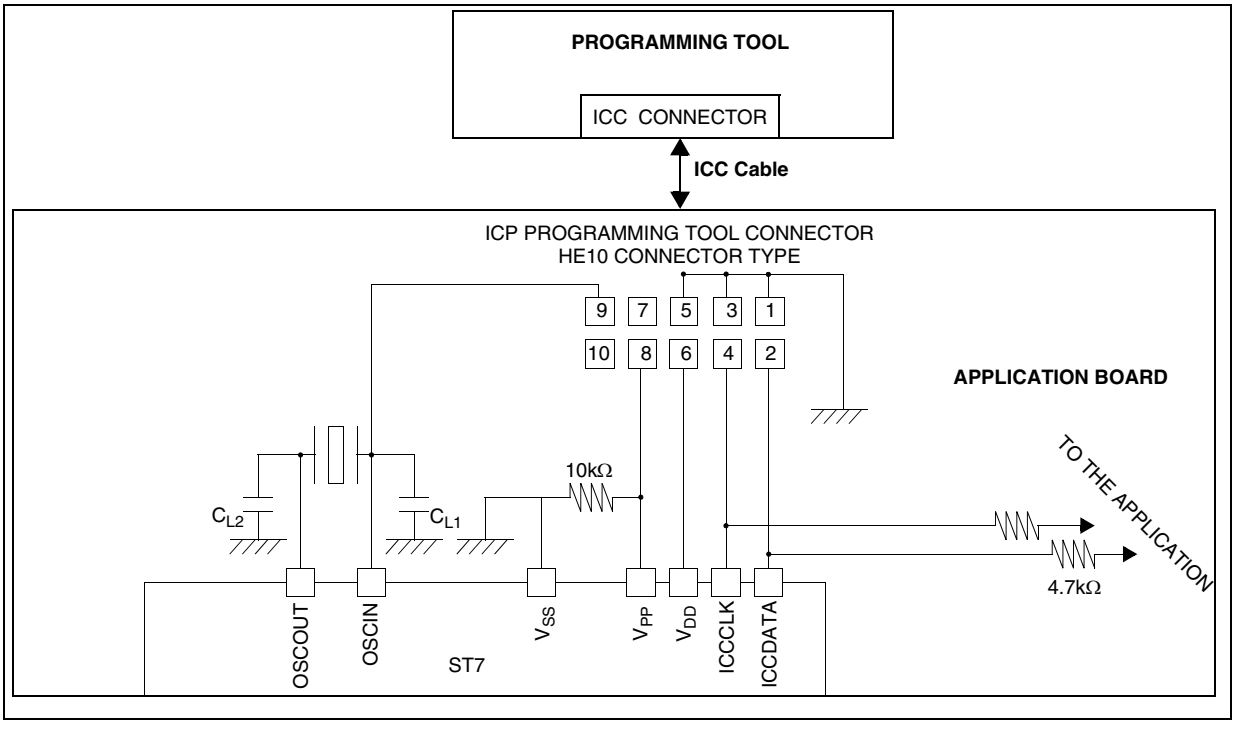
Note: To develop a custom programming tool, refer to the ST7 FLASH Programming and ICC Reference Manual which gives full details on the ICC protocol hardware and software.

4.5 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 USB 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.

Figure 8. Typical ICP Interface



FLASH PROGRAM MEMORY (Cont'd)

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

4.6 Program 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 (including a re-write protection). When this protection is removed by reprogramming the Option Byte, the entire Flash program memory is first automatically erased and the device can be reprogrammed.

Refer to the Option Byte description for more details.

4.7 Related Documentation

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

4.8 Register Description

FLASH CONTROL/STATUS REGISTER (FCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	0

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

5 CENTRAL PROCESSING UNIT

5.1 INTRODUCTION

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

5.2 MAIN FEATURES

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

5.3 CPU REGISTERS

The 6 CPU registers shown in Figure 9 are not present in the memory mapping and are accessed by specific instructions.

Accumulator (A)

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

Index Registers (X and Y)

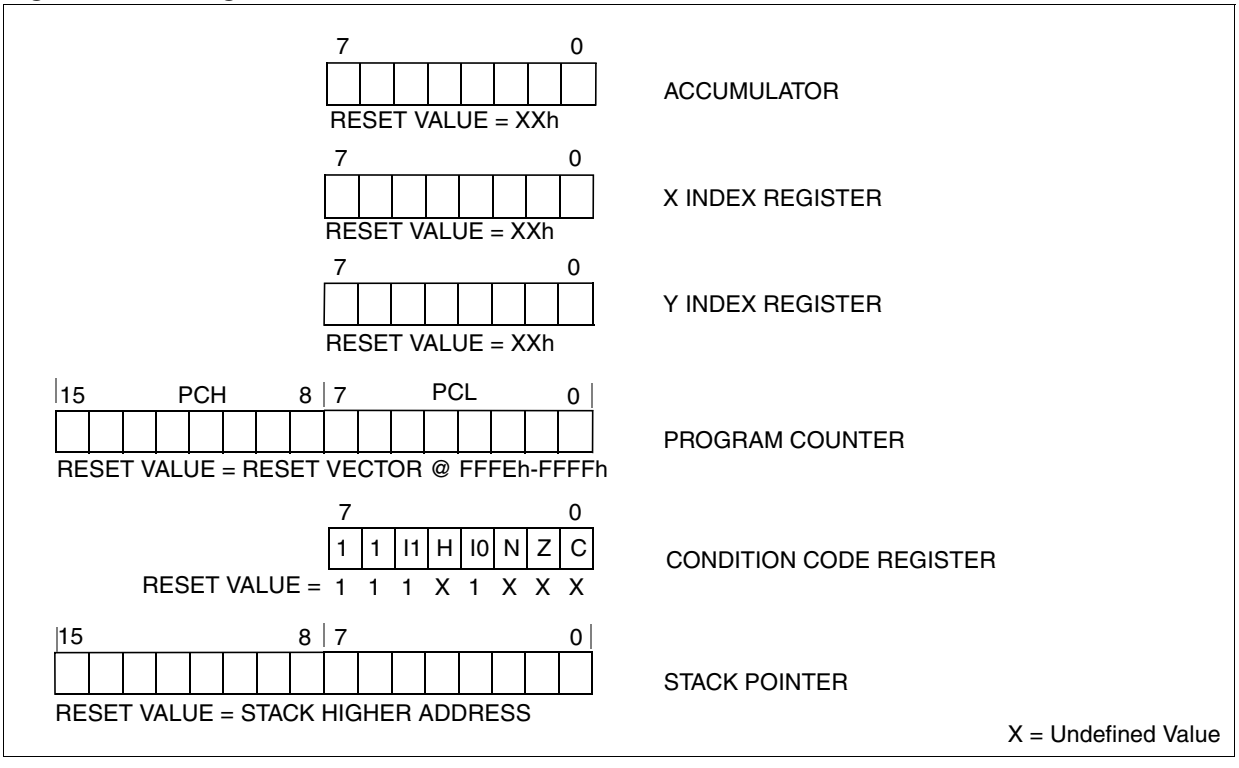
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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

Program Counter (PC)

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

Figure 9. CPU Registers



CENTRAL PROCESSING UNIT (Cont'd)

Condition Code Register (CC)

Read/Write

Reset Value: 111x1xxx

7							0
1	1	I1	H	I0	N	Z	C

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

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

Arithmetic Management Bits

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

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

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

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

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

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

This bit is accessed by the JRMI and JRPL instructions.

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

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

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

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

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

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

Interrupt Management Bits

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

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

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

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

See the interrupt management chapter for more details.

CENTRAL PROCESSING UNIT (Cont'd)

Stack Pointer (SP)

Read/Write

Reset Value: 017Fh

15							8
0	0	0	0	0	0	0	1

7							0
SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see Figure 10).

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.

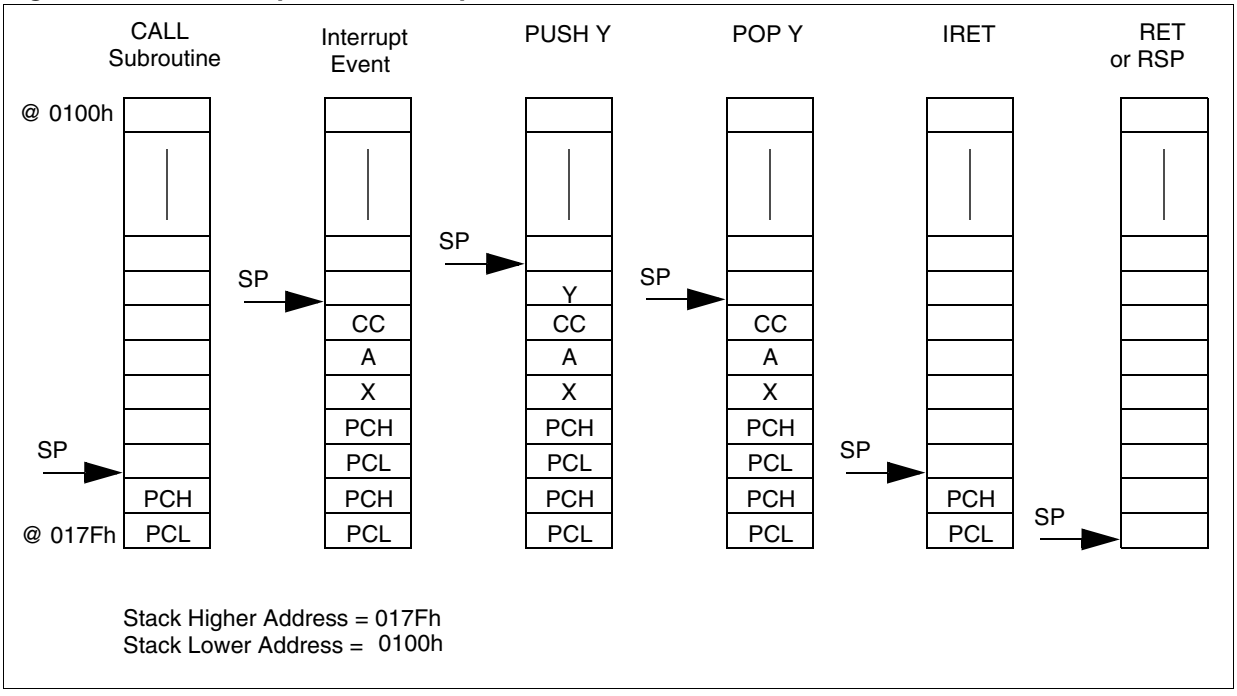
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 10

– 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 10. Stack Manipulation Example



6 SUPPLY, RESET AND CLOCK MANAGEMENT

6.1 CLOCK SYSTEM

6.1.1 General Description

The MCU accepts either a 4MHz crystal or an external clock signal to drive the internal oscillator. The internal clock (f_{CPU}) is derived from the internal oscillator frequency (f_{OSC}), which is 4Mhz.

After reset, the internal clock (f_{CPU}) is provided by the internal oscillator (4Mhz frequency).

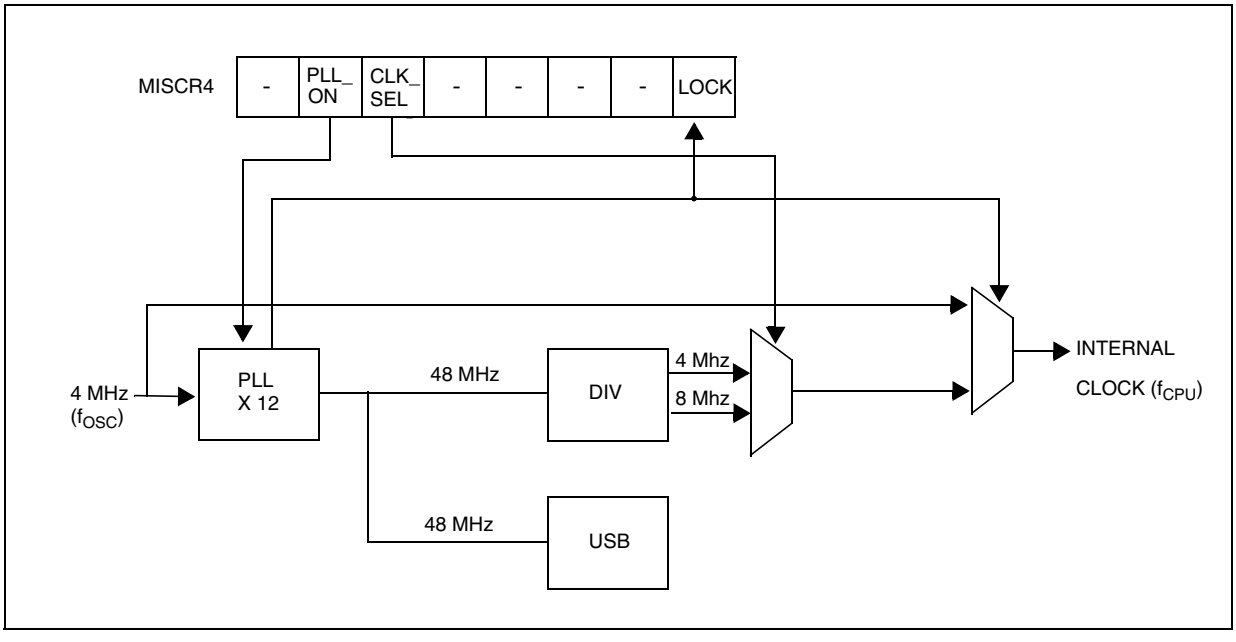
To activate the 48-MHz clock for the USB interface, the user must turn on the PLL by setting the PLL_ON bit in the MISCR4 register. When the PLL is locked, the LOCK bit is set by hardware.

The user can then select an internal frequency (f_{CPU}) of either 4 MHz or 8MHz by programming the CLK_SEL bit in the MISCR4 register (refer to MISCELLANEOUS REGISTERS section on page 39).

The PLL provides a signal with a duty cycle of 50 %.

The internal clock signal (f_{CPU}) is also routed to the on-chip peripherals. The CPU clock signal consists of a square wave with a duty cycle of 50%.

Figure 11. Clock, Reset and Supply Block Diagram



The internal oscillator is designed to operate with an AT-cut parallel resonant quartz in the frequency range specified for f_{osc}. The circuit shown in Figure 13 is recommended when using a crystal, and Table 5 lists the recommended capacitance. The crystal and associated components should be mounted as close as possible to the input pins in order to minimize output distortion and start-up stabilisation time. The LOCK bit in the MISCR4 register can also be used to generate the f_{CPU} directly from f_{OSC} if the PLL and the USB interface are not active.

Table 5. Recommended Values for 4 MHz Crystal Resonator

R_{SMAX}	20 Ω	25 Ω	70 Ω
C_{OSCIN}	56pF	47pF	22pF
C_{OSCOUT}	56pF	47pF	22pF

Note: R_{SMAX} is the equivalent serial resistor of the crystal (see crystal specification).

CLOCK SYSTEM (Cont'd)

6.1.2 External Clock

An external clock may be applied to the OSCIN input with the OSCOUT pin not connected, as shown on Figure 12.

Figure 12. .External Clock Source Connections

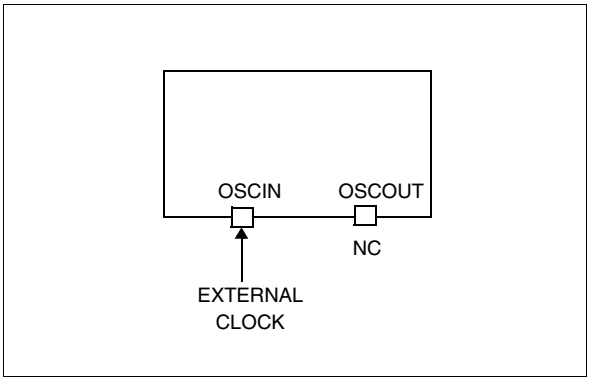
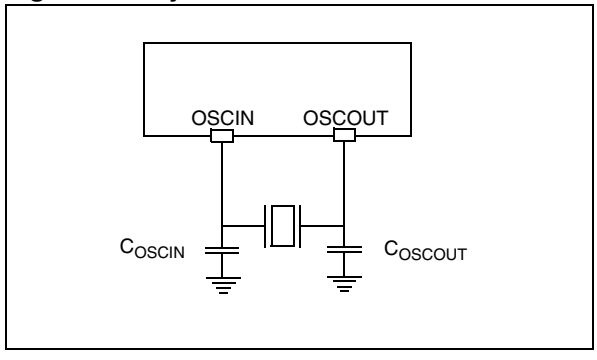


Figure 13. Crystal Resonator



6.2 RESET SEQUENCE MANAGER (RSM)

6.2.1 Introduction

The reset sequence manager has two reset sources:

- Internal LVD reset (Low Voltage Detection) which includes both a power-on and a voltage drop reset
- Internal watchdog reset generated by an internal watchdog counter underflow as shown in Figure 15.

6.2.2 Functional Description

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 14:

- A first delay of 30µs + 127 t_{CPU} cycles during which the internal reset is maintained.
- A second delay of 512 t_{CPU} cycles after the internal reset is generated. It allows the oscillator to stabilize and ensures that recovery has taken place from the Reset state.
- Reset vector fetch (duration: 2 clock cycles)

Low Voltage Detector

The low voltage detector generates a reset when $V_{DD}<V_{IT+}$ (rising edge) or $V_{DD}<V_{IT-}$ (falling edge), as shown in Figure 14.

The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets. See "SUPPLY AND RESET CHARACTERISTICS" on page 81.

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

Figure 14. LVD RESET Sequence

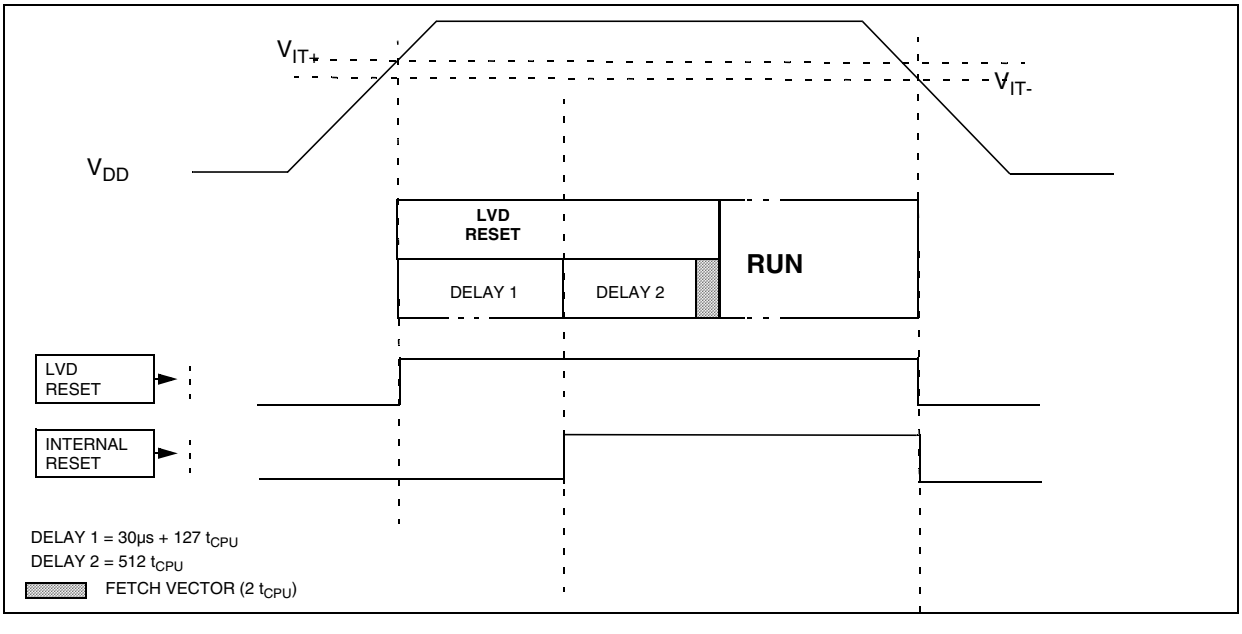
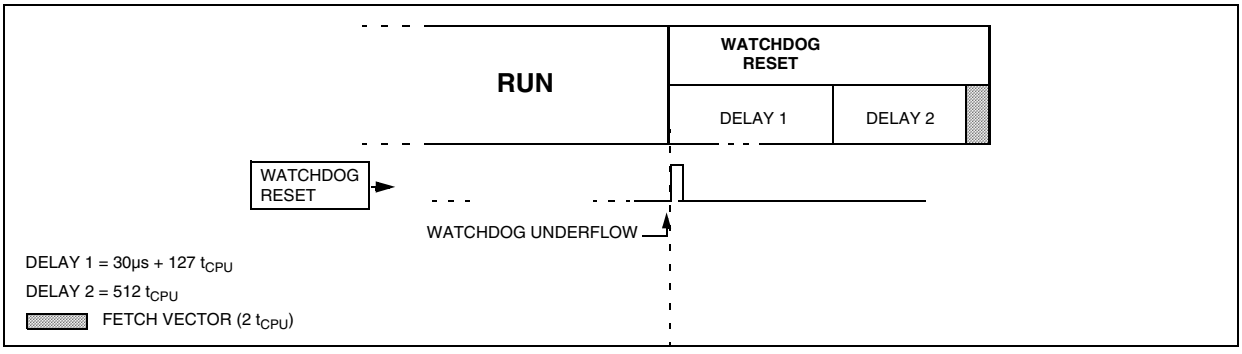


Figure 15. Watchdog RESET Sequence



7 INTERRUPTS

7.1 INTRODUCTION

The CPU 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: RESET, TRAP, TLI

This interrupt management is based on:

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

– Interrupt software priority registers (ISPRx),

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

This enhanced interrupt controller guarantees full upward compatibility with the standard (not nested) CPU 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 16.

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).

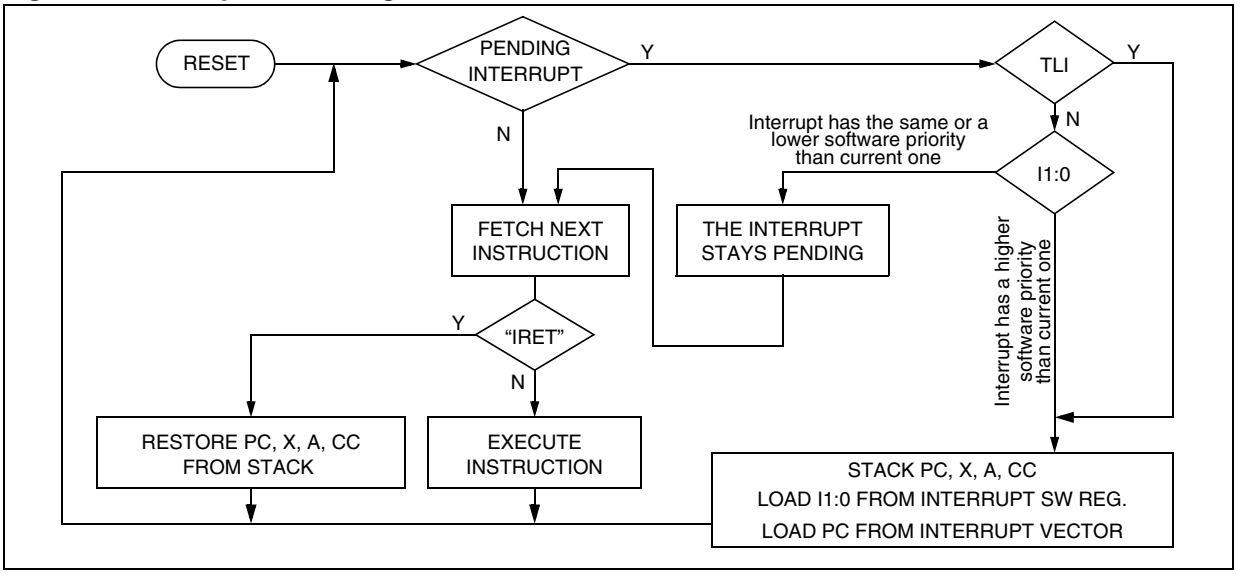
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

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 16. Interrupt Processing Flowchart



INTERRUPTS (Cont'd)

Servicing Pending Interrupts

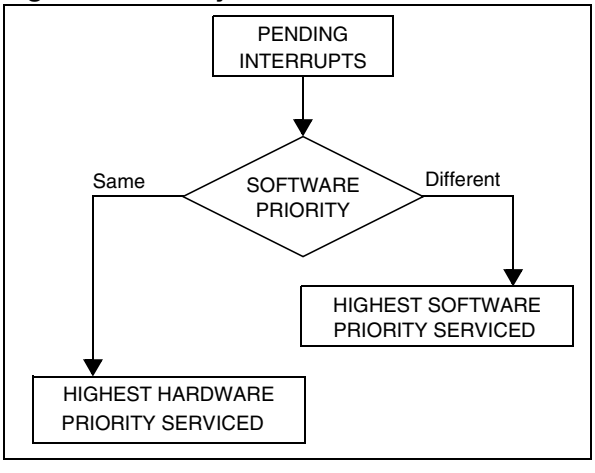
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– 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 17 describes this decision process.

Figure 17. Priority Decision Process



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

Note 1: The hardware priority is exclusive while the software one is not. This allows the previous process to succeed with only one interrupt.
Note 2: RESET, TRAP and TLI 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 CPU 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 16). 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.
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 16 as a TLI.
Caution: TRAP can be interrupted by a TLI.

- RESET

The RESET source has the highest priority in the CPU. 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 MISCR3 register.
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 NANDed.

- Peripheral Interrupts

Usually the peripheral interrupts cause the Device 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 17.

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 18 and Figure 19 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 19. 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 18. Concurrent Interrupt Management

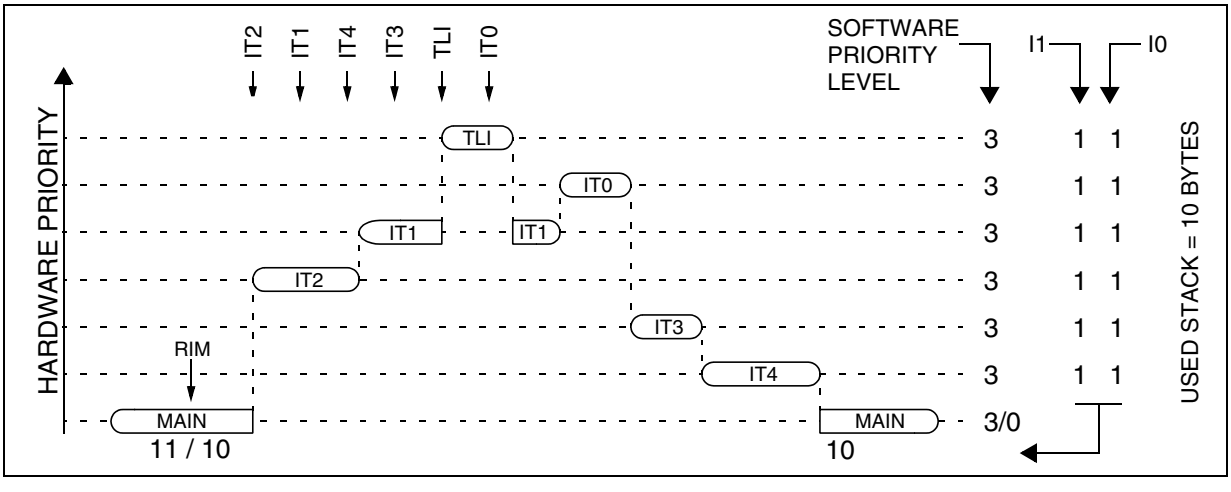
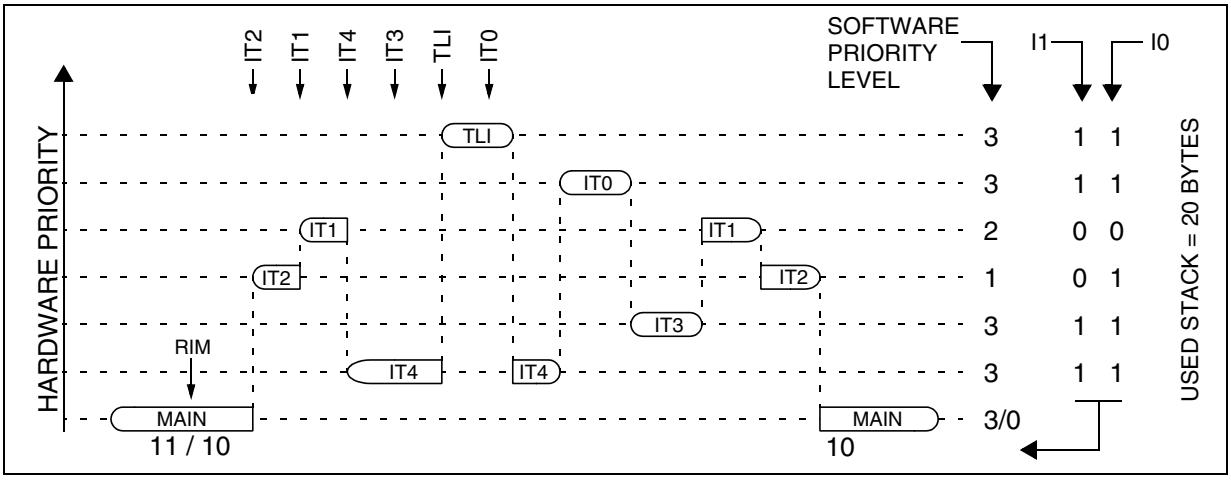


Figure 19. Nested Interrupt Management



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 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 RESET, TRAP and TLI 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	I1:0=11 ?						
JRNM	Jump if I1:0<>11	I1:0<>11 ?						
POP CC	Pop CC from the Stack	Mem => CC	I1	H	I0	N	Z	C
RIM	Enable interrupt (level 0 set)	Load 10 in I1:0 of CC	1		0			
SIM	Disable interrupt (level 3 set)	Load 11 in I1:0 of CC	1		1			
TRAP	Software trap	Software NMI	1		1			
WFI	Wait for interrupt		1		0			

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

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

Table 8. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Address Vector
	RESET	Reset	N/A	Highest Priority	yes	FFFEh-FFFFh
	TRAP	Software Interrupt			no	FFFCh-FFFDh
0	ICP	FLASH Start programming NMI interrupt (TLI)				FFFAh-FFFBh
1	UART	ISO7816-3 UART Interrupt	UIC			FFF8h-FFF9h
2	USB	USB Communication Interrupt	USBISTR			FFF6h-FFF7h
3	WAKUP1	External Interrupt Port C			yes	FFF4h-FFF5h
4	WAKUP2	External Interrupt Port A			yes	FFF2h-FFF3h
5	TIM	TBU Timer Interrupt	TBUSR		no	FFF0h-FFF1h
6	CARDDET [1]	Smartcard Insertion/Removal Interrupt [1]	USCUR		yes	FFEEh-FFEFh
7	ESUSP	End suspend Interrupt	USBISTR			FFECh-FFEDh
8	Not used			Lowest Priority	no	FFEAh-FFEBh

Note 1: This interrupt can be used to exit from USB suspend mode.

8 POWER SAVING MODES

8.1 INTRODUCTION

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

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

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.

8.2 WAIT MODE

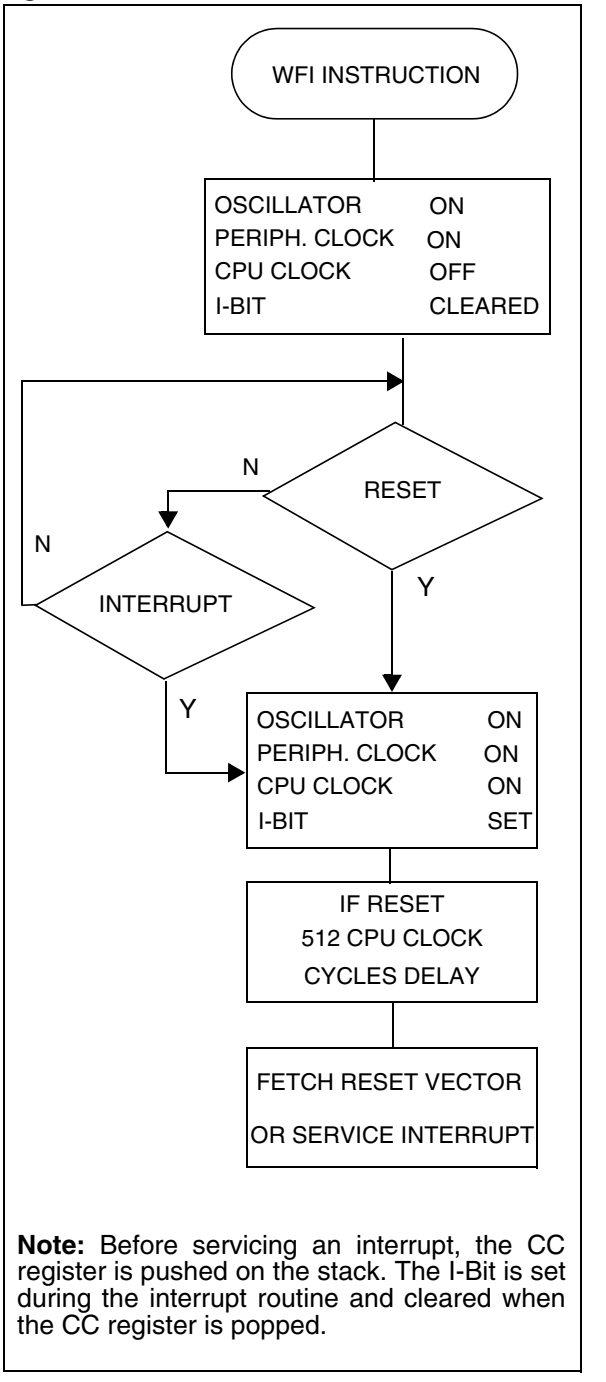
WAIT mode places the MCU in a low power consumption mode by stopping the CPU.

This power saving mode is selected by calling the "WFI" ST7 software instruction.

All peripherals remain active. During WAIT mode, the I bit of the CC register is forced to 0, to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in WAIT mode until an interrupt or Reset occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine. The MCU will remain in WAIT mode until a Reset or an Interrupt occurs, causing it to wake up.

Refer to Figure 20.

Figure 20. WAIT Mode Flow Chart



Note: Before servicing an interrupt, the CC register is pushed on the stack. The I-Bit is set during the interrupt routine and cleared when the CC register is popped.

POWER SAVING MODES (Cont'd)

8.3 HALT MODE

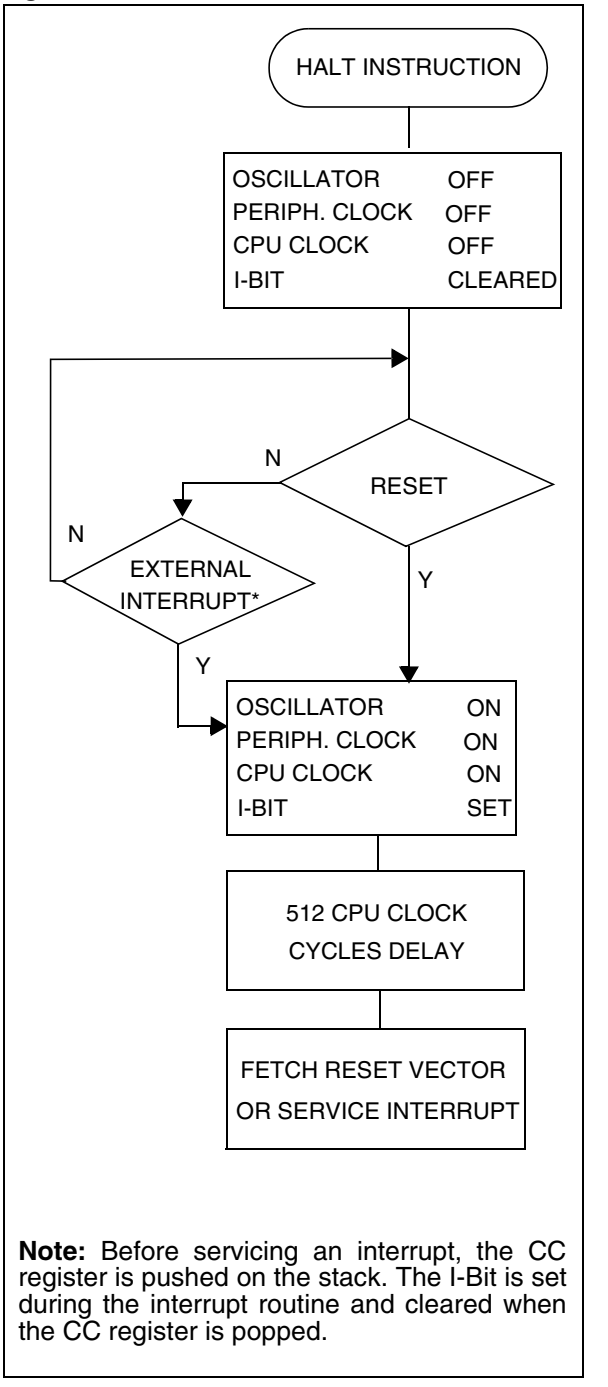
The HALT mode is the MCU lowest power consumption mode. The HALT mode is entered by executing the HALT instruction. The internal oscillator is then turned off, causing all internal processing to be stopped, including the operation of the on-chip peripherals.

Note: The PLL must be disabled before a HALT instruction.

When entering HALT mode, the I bit in the Condition Code Register is cleared. Thus, any of the external interrupts (ITi or USB end suspend mode), are allowed and if an interrupt occurs, the CPU clock becomes active.

The MCU can exit HALT mode on reception of either an external interrupt on ITi, an end suspend mode interrupt coming from USB peripheral, or a reset. The oscillator is then turned on and a stabilization time is provided before releasing CPU operation. The stabilization time is 512 CPU clock cycles.
After the start up delay, the CPU continues operation by servicing the interrupt which wakes it up or by fetching the reset vector if a reset wakes it up.

Figure 21. HALT Mode Flow Chart



Note: Before servicing an interrupt, the CC register is pushed on the stack. The I-Bit is set during the interrupt routine and cleared when the CC register is popped.

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:

– alternate signal input/output for the on-chip peripherals.

– external interrupt detection

An I/O port is composed of 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 is associated to 4 main registers:

– Data Register (DR)

– Data Direction Register (DDR)

– Option Register (OR)

– Pull Up Register (PU)

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

Table 9. I/O Pin Functions

DDR	MODE
0	Input
1	Output

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.

Note 1: All the inputs are triggered by a Schmitt trigger.
Note 2: When switching from input mode to output mode, the DR register should be written first to output the correct value as soon as the port is configured as an output.

Interrupt function

When an I/O is configured in Input with Interrupt, an event on this I/O can generate an external Interrupt request to the CPU. The interrupt sensitivity is given independently according to the description mentioned in the ITRFRE interrupt register.

Each pin can independently generate an Interrupt request.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see Interrupts section). If more than one input pin is selected simultaneously as interrupt source, this is logically ORed. For this reason if one of the interrupt pins is tied low, it masks the other ones.

Output Mode

The pin is configured in output mode by setting the corresponding DDR register bit (see Table 7).

In this mode, writing “0” or “1” to 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.

Note: In this mode, the interrupt function is disabled.

Digital Alternate Function

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over 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 has to be configured in input mode. In this case, the pin’s state is also digitally readable by addressing the DR register.

Notes:
1. Input pull-up configuration can cause an unexpected value at the input of the alternate peripheral input.
2. When the on-chip peripheral uses a pin as input and output, this pin must be configured as an input (DDR = 0).

***Warning*:** 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.

I/O PORTS (Cont'd)

9.3 I/O Port Implementation

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

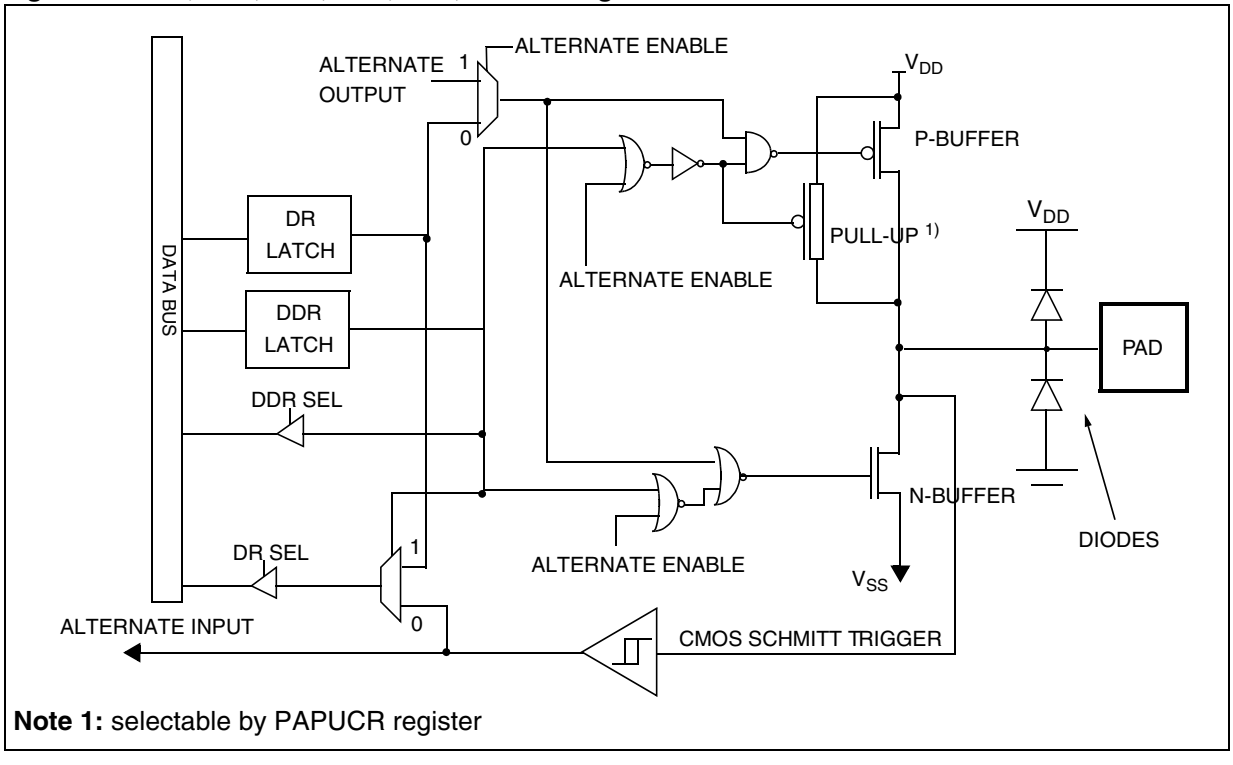
9.3.1 Port A

Table 10. Port A Description

PORT A	I / O	
	Input	Output
PA[5:0]	without pull-up *	push-pull or open drain with software selectable pull-up
PA6	without pull-up	-

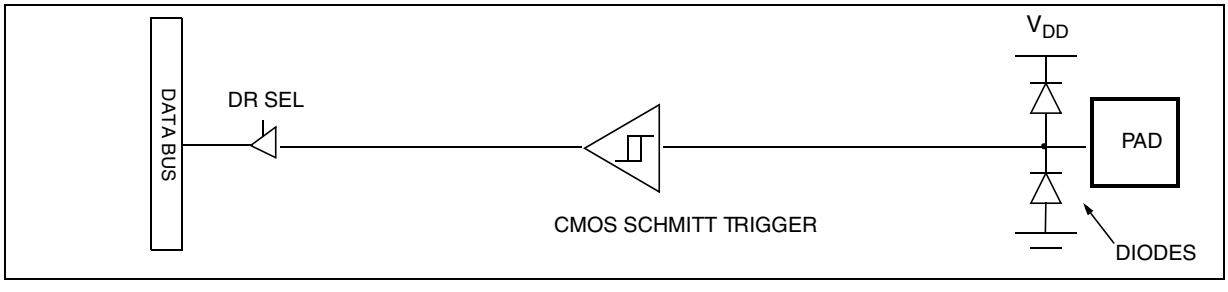
*Reset State

Figure 22. PA0, PA1, PA2, PA3, PA4, PA5 Configuration



Note 1: selectable by PAPUCR register

Figure 23. PA6 Configuration



I/O PORTS (Cont'd)

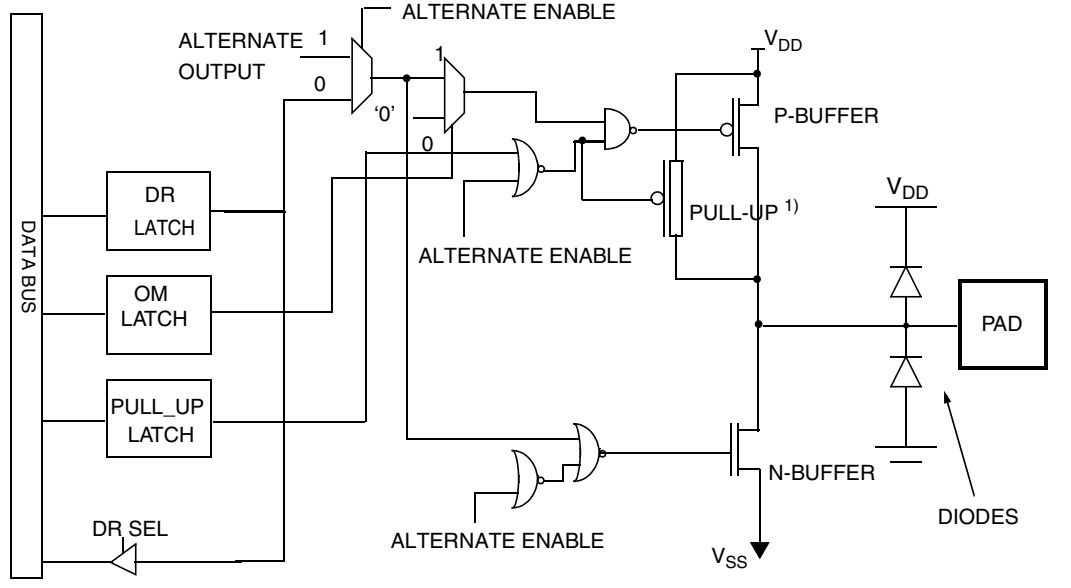
9.3.2 Ports B and D

Table 11. Port B and D Description

PORTS B AND D	Output *
PB[7:0]	push-pull or open drain with software selectable pull-up
PD[7:0]	

*Reset State = open drain

Figure 24. Port B and D Configuration



Note 1: selectable by PAPUCR register

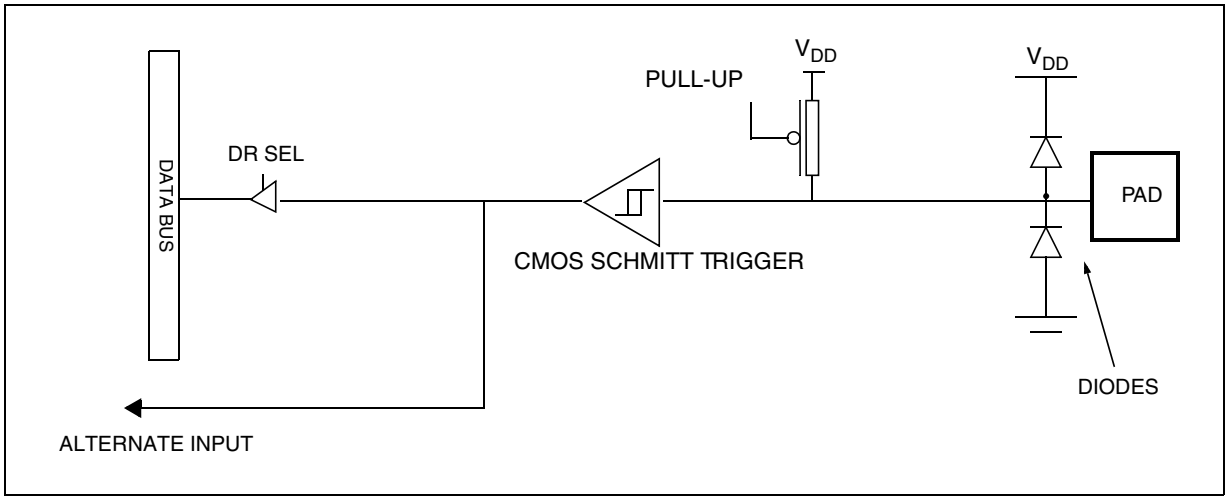
I/O PORTS (Cont'd)

9.3.3 Port C

Table 12. Port C Description

PORT C	Input
PC[7:0]	with pull-up

Figure 25. Port C Configuration



I/O PORTS (Cont'd)

9.4 Register Description

DATA REGISTERS (PxDR)

Port A Data Register (PADR): 0011h

Port B Data Register (PBDR): 0015h

Port C Data Register (PCDR): 0018h

Port D Data Register (PCDR): 0019h

Read/Write

Reset Value Port A: 0000 0000 (00h)

Reset Value Port B: 0000 0000 (00h)

Reset Value Port C: 0000 0000 (00h)

Reset Value Port D: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

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

The DR register has a specific behaviour according to the selected input/output configuration. Writing the DR register is always taken in account even if the pin is configured as an input. Reading the DR register returns either the DR register latch content (pin configured as output) or the digital value applied to the I/O pin (pin configured as input).

DATA DIRECTION REGISTER (PADDR)

Port A Data Direction Register (PADDR): 0012h

Read/Write

Reset Value Port A: 0000 0000 (00h)

7							0
DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0

Bits 7:0 = **DD7-DD0** *Data Direction Register 8 bits.*

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

0: Input mode
1: Output mode

OPTION REGISTER (PxOR)

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

Port A Option Register (PAOR): 0013h

Port B Option Register (PBOR): 0016h

Port D Option Register (PDOR): 001Ah

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OM7	OM6	OM5	OM4	OM3	OM2	OM1	OM0

Bits 7:0 = **OM[7:0]** *Option register 8 bits.*

The OR register allows to distinguish in output mode if the push-pull or open drain configuration is selected.

Each bit is set and cleared by software.
0: Output open drain
1: Output push-pull

PULL UP CONTROL REGISTER (PxPUCR)

Port x Pull Up Register
PxPUCR with x = A, B, or D

Port A Pull up Register (PAPUCR): 0014h

Port B Pull up Register (PBPUCR): 0017h

Port D Pull up Register (PDPUCR): 001Bh

Read/Write

Reset Value: 0000 0000 (00h)

7							0
PU7	PU6	PU5	PU4	PU3	PU2	PU1	PU0

Bits 7:0 = **PU[7:0]** *Pull up register 8 bits.*

The PU register is used to control the pull up.

Each bit is set and cleared by software.
0: Pull up inactive
1: Pull up active

I/O PORTS (Cont'd)

Table 13. I/O Ports Register Map

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
11	**PADR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
12	**PADDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
13	**PAOR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
14	**PAPUCR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
15	**PBDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
16	**PBOR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
17	**PBPUCR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
18	**PCDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
19	**PDDR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
1A	**PDOR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
1B	**PDPUCR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0

10 MISCELLANEOUS REGISTERS

MISCELLANEOUS REGISTER 1 (MISCR1)

Reset Value : 0000 0000 (00h)

Read/Write

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

Writing the ITIFREC register enables or disables external interrupt on Port C. Each bit can be masked independantly. The ITMx bit masks the external interrupt on PC.x.

Bits[7:0] = **ITM [7:0]** *Interrupt Mask*
0: external interrupt disabled
1: external interrupt enabled

MISCELLANEOUS REGISTER 2 (MISCR2)

Reset Value : 0000 0000 (00h)

Read/Write

7							0
-	CRD IRM	ITM 14	ITM 13	ITM 12	ITM 11	ITM 10	ITM 9

Writing the ITIFREA register enables or disables external interrupt on port A.

Bit 7 = Reserved.

Bit 6 = **CRDIRM** *CRD Insertion/Removal Interrupt Mask*
0: CRDIR interrupt disabled
1: CRDIR interrupt enabled

Bits [5:0] = **ITM [14:9]** *Interrupt Mask*

Bit x of MISCR2 masks the external interrupt on port A.x.

Bit x = **ITM *n*** *Interrupt Mask n*
0: external interrupt disabled on PA.x.
1: external interrupt enabled on PA.x.

MISCELLANEOUS REGISTER 3 (MISCR3)

Reset Value: 0000 0000 (00h)

Read/Write

7							0
CTR L1_A	CTR L0_A	CTR L1_C	CTR L0_C	-	-	-	-

This register is used to configure the edge and the level sensitivity of the Port A and Port C external interrupt. This means that all bits of a port must have the same sensitivity.

If a write access modifies bits 7:4, it clears the pending interrupts.

CTRL0_C, CTRL1_C : Sensitivity on port C

CTRL0_A, CTRL1_A : Sensitivity on port A

CTR L1_X	CTR L0_X	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

MISCELLANEOUS REGISTER 4 (MISCR4)

Reser Value : 0000 0000 (00h).

Read/Write

7							0
-	PLL _ON	CLK_ SEL	-	-	-	-	LOCK

Bit 7 = Reserved.

Bit 6 = **PLL_ON** *PLL Activation*
0: PLL disabled
1: PLL enabled

Note: The PLL must be disabled before a HALT instruction.

Bit 5 = **CLK_SEL** *Clock Selection*

This bit is set and cleared by software.
0: CPU frequency = 4MHz
1: CPU frequency = 8MHz

Bits 4:1 = Reserved.

Bit 0 = **LOCK** *PLL status bit*
0: PLL not locked. $f_{CPU} = f_{OSC}$ external clock frequency.
1: PLL locked. f_{CPU} = 4 or 8 MHz depending on CLKSEL bit.

MISCELLANEOUS REGISTERS (Cont'd)

Table 14. Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
001C	MISCR1 Reset Value	ITM7 0	ITM6 0	ITM5 0	ITM4 0	ITM3 0	ITM2 0	ITM1 0	ITM0 0
001D	MISCR2 Reset Value	0	0	ITM14 0	ITM13 0	ITM12 0	ITM11 0	ITM10 0	ITM9 0
001E	MISCR3 Reset Value	CTRL1_A 0	CTRL0_A 0	CTRL1_C 0	CTRL0_C 0	0	0	0	0
001Fh	MISCR4 Reset Value	0	PLL_ON 0	RST_IN 0	CLK_SE 0L	0	0	0	LOCK 0

11 LEDs

Each of the four available LEDs can be selected using the LED_CTRL register. Two types of LEDs are supported: 3mA and 7mA.

LED_CTRL REGISTER

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

7							0
LD3	LD2	LD1	LD0	LD3_I	LD2_I	LD1_I	LD0_I

Bits 7:4 = **LDx** *LED Enable*
0: LED disabled
1: LED enabled

Bits 3:0 = **LDx_I** *Current selection on LDx*

0: 3mA current on LDx pad
1: 7mA current on LDx pad

12 ON-CHIP PERIPHERALS

12.1 WATCHDOG TIMER (WDG)

12.1.1 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.

12.1.2 Main Features

- Programmable free-running downcounter (64 increments of 65536 CPU cycles)
- Programmable reset
- Reset (if watchdog activated) when the T6 bit reaches zero
- Hardware Watchdog selectable by option byte
- Watchdog Reset indicated by status flag

12.1.3 Functional Description

The counter value stored in the CR register (bits T[6:0]), is decremented every 65,536 machine cycles, and the length of the timeout period can be programmed by the user in 64 increments.

If the watchdog is activated (the WDGA bit is set) and when the 7-bit timer (bits 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 CR register at regular intervals during normal operation to prevent an MCU reset. This downcounter is free-running: it counts down even if the watchdog is disabled. The value to be stored in the CR register must be between FFh and C0h (see Table 15):

– The WDGA bit is set (watchdog enabled)

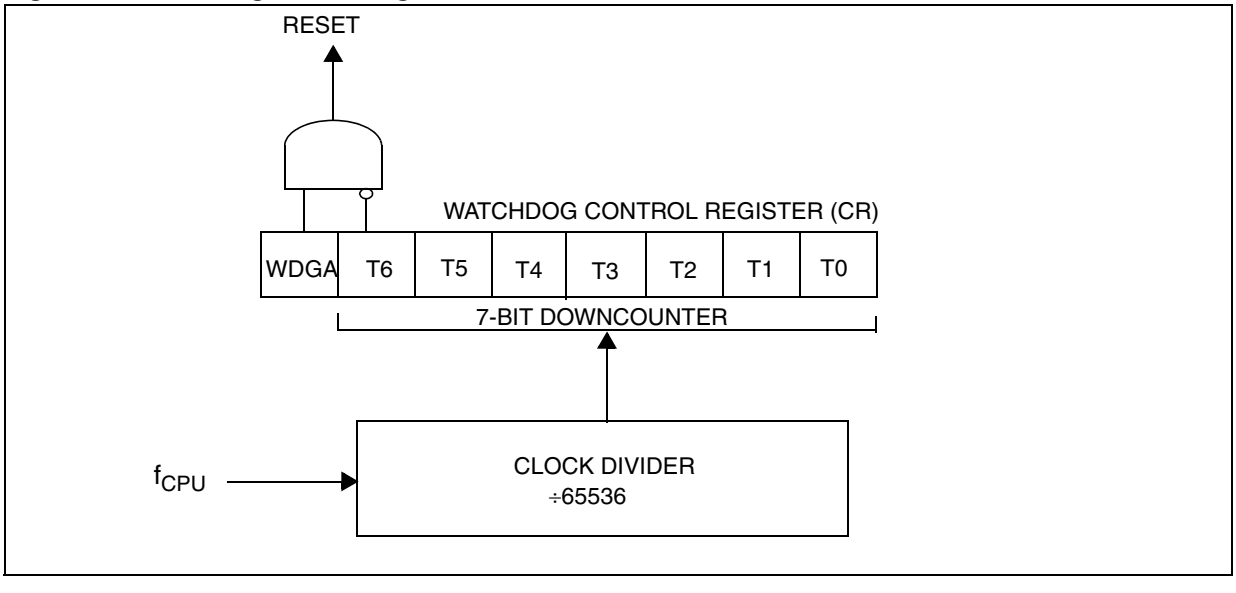
– The T6 bit is set to prevent generating an immediate reset

– The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset.

Table 15.Watchdog Timing (f_{CPU} = 8 MHz)

	CR Register initial value	WDG timeout period (ms)
Max	FFh	524.288
Min	C0h	8.192

Figure 26. Watchdog Block Diagram



WATCHDOG TIMER (Cont'd)

12.1.4 Software Watchdog Option

If Software Watchdog is selected by option byte, the watchdog is disabled following a reset. 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).

12.1.5 Hardware Watchdog Option

If Hardware Watchdog is selected by option byte, the watchdog is always active and the WDGA bit in the CR is not used.

12.1.6 Low Power Modes

WAIT Instruction

No effect on Watchdog.

HALT Instruction

Halt mode can be used when the watchdog is enabled. When the oscillator is stopped, the WDG stops counting and is no longer able to generate a reset until the microcontroller receives an external interrupt or a reset.

If an external interrupt is received, the WDG restarts counting after 514 CPU clocks. In the case of the Software Watchdog option, if a reset is generated, the WDG is disabled (reset state).

Recommendations

– Make sure that an external event is available to wake up the microcontroller from Halt mode.

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

– When using an external interrupt to wake up the microcontroller, reinitialize the corresponding I/O as Input 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.

– 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 I bit 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).

12.1.7 Interrupts

None.

12.1.8 Register Description

CONTROL REGISTER (CR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

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

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

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

12.2 TIME BASE UNIT (TBU)

12.2.1 Introduction

The Timebase unit (TBU) can be used to generate periodic interrupts.

12.2.2 Main Features

- 8-bit upcounter
- Programmable prescaler
- Period between interrupts: max. 8.1ms (at 8 MHz f_{CPU})
- Maskable interrupt

12.2.3 Functional Description

The TBU operates as a free-running upcounter.

When the TCEN bit in the TBUCSR register is set by software, counting starts at the current value of the TBUCV register. The TBUCV register is incremented at the clock rate output from the prescaler selected by programming the PR[2:0] bits in the TBUCSR register.

When the counter rolls over from FFh to 00h, the OVF bit is set and an interrupt request is generated if ITE is set.

The user can write a value at any time in the TBUCV register.

12.2.4 Programming Example

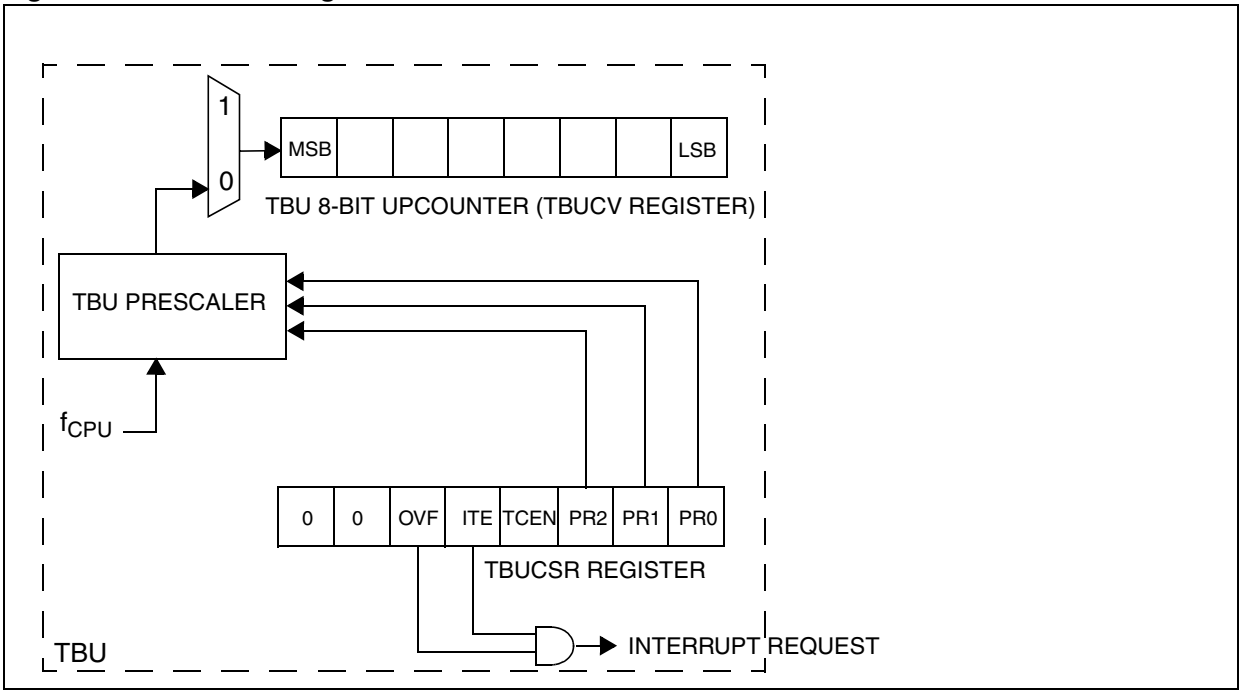
In this example, timer is required to generate an interrupt after a delay of 1 ms.

Assuming that f_{CPU} is 8 MHz and a prescaler division factor of 256 will be programmed using the PR[2:0] bits in the TBUCSR register, 1 ms = 32 TBU timer ticks.

In this case, the initial value to be loaded in the TBUCV must be (256-32) = 224 (E0h).

```
ld A, E0h
ld TBUCV, A   ; Initialize counter value
ld A 1Fh      ;
ld TBUCSR, A  ; Prescaler factor = 256,
              ; interrupt enable,
              ; TBU enable
```

Figure 27. TBU Block Diagram



TIMEBASE UNIT (Cont'd)

12.2.5 Low Power Modes

Mode	Description
WAIT	No effect on TBU
HALT	TBU halted.

12.2.6 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Counter Overflow Event	OVF	ITE	Yes	No

Note: The OVF interrupt event is connected to an interrupt vector (see Interrupts chapter).
It generates an interrupt if the ITE bit is set in the TBUCSR register and the I-bit in the CC register is reset (RIM instruction).

12.2.7 Register Description

TBU COUNTER VALUE REGISTER (TBUCV)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
CV7	CV6	CV5	CV4	CV3	CV2	CV1	CV0

Bits 7:0 = **CV[7:0]** *Counter Value*

This register contains the 8-bit counter value which can be read and written anytime by software. It is continuously incremented by hardware if TCEN=1.

TBU CONTROL/STATUS REGISTER (TBUCSR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	OVF	ITE	TCEN	PR2	PR1	PR0

Bits [7:6] = Reserved. Forced by hardware to 0.

Bit 5 = **OVF** *Overflow Flag*

This bit is set only by hardware, when the counter value rolls over from FFh to 00h. It is cleared by software reading the TBUCSR register. Writing to this bit does not change the bit value.
0: No overflow
1: Counter overflow

Bit 4 = **ITE** *Interrupt enabled.*

This bit is set and cleared by software.
0: Overflow interrupt disabled
1: Overflow interrupt enabled. An interrupt request is generated when OVF=1.

Bit 3 = **TCEN** *TBU Enable.*

This bit is set and cleared by software.
0: TBU counter is frozen and the prescaler is reset.
1: TBU counter and prescaler running.

Bits 2:0 = **PR[2:0]** *Prescaler Selection*

These bits are set and cleared by software to select the prescaling factor.

PR2	PR1	PR0	Prescaler Division Factor
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	256

12.3 USB INTERFACE (USB)

12.3.1 Introduction

The USB Interface implements a full-speed function interface between the USB and the ST7 microcontroller. It is a highly integrated circuit which includes the transceiver, 3.3 voltage regulator, SIE and USB Data Buffer interface. No external components are needed apart from the external pull-up on USBDP for full speed recognition by the USB host.

12.3.2 Main Features

- USB Specification Version 1.1 Compliant
- Supports Full-Speed USB Protocol
- Seven Endpoints (including default endpoint)
- CRC generation/checking, NRZI encoding/decoding and bit-stuffing
- USB Suspend/Resume operations
- On-Chip 3.3V Regulator
- On-Chip USB Transceiver

12.3.3 Functional Description

The block diagram in Figure 28, gives an overview of the USB interface hardware.

For general information on the USB, refer to the "Universal Serial Bus Specifications" document available at http//:www.usb.org.

Serial Interface Engine

The SIE (Serial Interface Engine) interfaces with the USB, via the transceiver.

The SIE processes tokens, handles data transmission/reception, and handshaking as required by the USB standard. It also performs frame formatting, including CRC generation and checking.

Endpoints

The Endpoint registers indicate if the microcontroller is ready to transmit/receive, and how many bytes need to be transmitted.

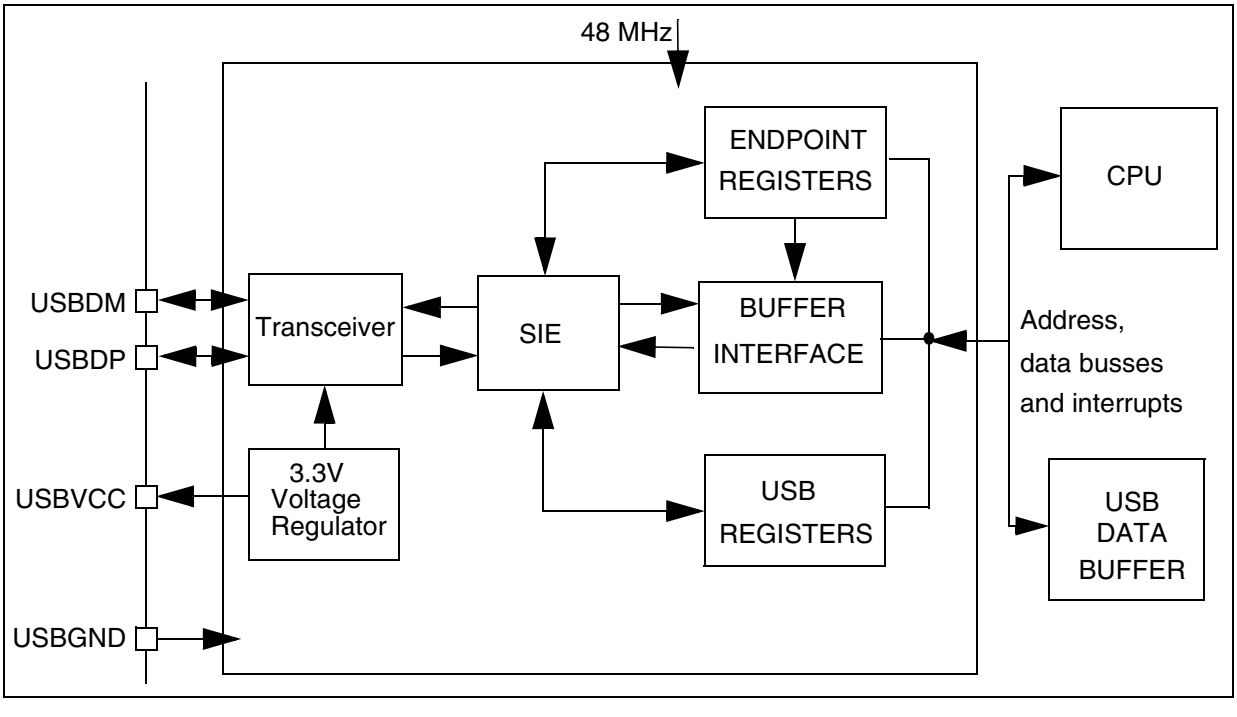
Data Transfer to/from USB Data Buffer Memory

When a token for a valid Endpoint is recognized by the USB interface, the related data transfer takes place to/from the USB data buffer. At the end of the transaction, an interrupt is generated.

Interrupts

By reading the Interrupt Status register, application software can know which USB event has occurred.

Figure 28. USB Block Diagram



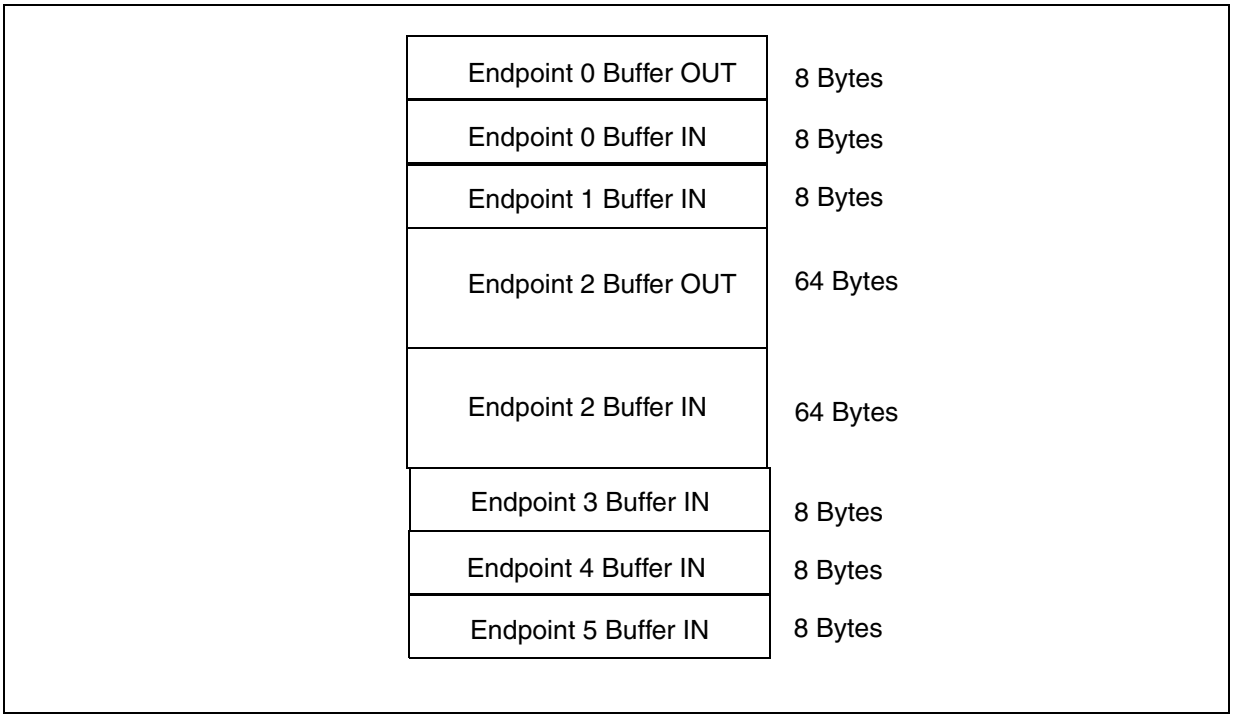
USB INTERFACE (Cont'd)

USB Endpoint RAM Buffers

There are seven Endpoints including one bidirectional control Endpoint (Endpoint 0), five IN Endpoints (Endpoint 1, 2, 3, 4, 5) and one OUT endpoint (Endpoint 2).

Endpoint 0 is 2 x 8 bytes in size, Endpoint 1, 3, 4, and Endpoint 5 are 8 bytes in size and Endpoint 2 is 2 x 64 bytes in size .

Figure 29. Endpoint Buffer Size



USB INTERFACE (Cont'd)

12.3.4 Register Description

INTERRUPT STATUS REGISTER (USBISTR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTR	0	SOVR	ERROR	SUSP	ESUSP	RESET	SOF

These bits cannot be set by software. When an interrupt occurs these bits are set by hardware. Software must read them to determine the interrupt type and clear them after servicing.
Note: The CTR bit (which is an OR of all the endpoint CTR flags) cannot be cleared directly, only by clearing the CTR flags in the Endpoint registers.

Bit 7 = **CTR** *Correct Transfer*.
This bit is set by hardware when a correct transfer operation is performed. This bit is an OR of all CTR flags (CTR0 in the EP0R register and CTR_RX and CTR_TX in the EPnRXR and EPnTXR registers). By looking in the USBSR register, the type of transfer can be determined from the PID[1:0] bits for Endpoint 0. For the other Endpoints, the Endpoint number on which the transfer was made is identified by the EP[1:0] bits and the type of transfer by the IN/OUT bit.
0: No Correct Transfer detected
1: Correct Transfer detected

Note: A transfer where the device sent a NAK or STALL handshake is considered not correct (the host only sends ACK handshakes). A transfer is considered correct if there are no errors in the PID and CRC fields, if the DATA0/DATA1 PID is sent as expected, if there were no data overruns, bit stuffing or framing errors.

Bit 6 = Reserved, forced by hardware to 0.

Bit 5 = **SOVR** Setup Overrun.
This bit is set by hardware when a correct Setup transfer operation is performed while the software is servicing an interrupt which occured on the same Endpoint (CTR0 bit in the EP0R register is still set when SETUP correct transfer occurs).
0: No SETUP overrun detected
1: SETUP overrun detected

When this event occurs, the USBSR register is not updated because the only source of the SOVR event is the SETUP token reception on the Control Endpoint (EP0).

Bit 4 = **ERR** *Error*.
This bit is set by hardware whenever one of the errors listed below has occurred:
0: No error detected
1: Timeout, CRC, bit stuffing, nonstandard framing or buffer overrun error detected

Note: Refer to the ERR[2:0] bits in the USBSR register to determine the error type.

Bit 3 = **SUSP** *Suspend mode request*.
This bit is set by hardware when a constant idle state is present on the bus line for more than 3 ms, indicating a suspend mode request from the USB.

The suspend request check is active immediately after each USB reset event and is disabled by hardware when suspend mode is forced (FSUSP bit in the USBCTLR register) until the end of resume sequence.

Bit 2 = **ESUSP** *End Suspend mode*.
This bit is set by hardware when, during suspend mode, activity is detected that wakes the USB interface up from suspend mode.

This interrupt is serviced by a specific vector, in order to wake up the ST7 from HALT mode.
0: No End Suspend detected
1: End Suspend detected

Bit 1 = **RESET** *USB reset.*
This bit is set by hardware when the USB reset sequence is detected on the bus.
0: No USB reset signal detected
1: USB reset signal detected

Note: The DADDR, EP0R, EP1RXR, EP1TXR, EP2RXR and EP2TXR registers are reset by a USB reset.

Bit 0 = **SOF** *Start of frame.*
This bit is set by hardware when a SOF token is received on the USB.
0: No SOF received
1: SOF received

Note: To avoid spurious clearing of some bits, it is recommended to clear them using a load instruction where all bits which must not be altered are set, and all bits to be cleared are reset. Avoid read-modify-write instructions like AND, XOR...

USB INTERFACE (Cont'd)

INTERRUPT MASK REGISTER (USBIMR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTRM	0	SOVRM	ERRM	SUSPM	ESUSPM	RESETM	SOFM

These bits are mask bits for all the interrupt condition bits included in the USBISTR register. Whenever one of the USBIMR bits is set, if the corresponding USBISTR bit is set, and the I- bit in the CC register is cleared, an interrupt request is generated. For an explanation of each bit, please refer to the description of the USBISTR register.

CONTROL REGISTER (USBCTLR)

Read/Write

Reset value: 0000 0110 (06h)

7							0
RSM	USB_RST	0	0	RESUME	PDWN	FSUSP	FRES

Bit 7 = **RSM** *Resume Detected*
This bit shows when a resume sequence has started on the USB port, requesting the USB interface to wake-up from suspend state. It can be used to determine the cause of an ESUSP event.
0: No resume sequence detected on USB
1: Resume sequence detected on USB

Bit 6 = **USB_RST** *USB Reset detected*.
This bit shows that a reset sequence has started on the USB. It can be used to determine the cause of an ESUSP event (Reset sequence).
0: No reset sequence detected on USB
1: Reset sequence detected on USB

Bits [5:4] = Reserved, forced by hardware to 0.

Bit 3 = **RESUME** *Resume*.
This bit is set by software to wake-up the Host when the ST7 is in suspend mode.
0: Resume signal not forced
1: Resume signal forced on the USB bus.

Software should clear this bit after the appropriate delay.

Bit 2 = **PDWN** *Power down*.
This bit is set by software to turn off the 3.3V on-chip voltage regulator that supplies the external pull-up resistor and the transceiver.
0: Voltage regulator on
1: Voltage regulator off

Note: After turning on the voltage regulator, software should allow at least 3 µs for stabilisation of the power supply before using the USB interface.

Bit 1 = **FSUSP** *Force suspend mode*.
This bit is set by software to enter Suspend mode. The ST7 should also be put in Halt mode to reduce power consumption.
0: Suspend mode inactive
1: Suspend mode active

When the hardware detects USB activity, it resets this bit (it can also be reset by software).

Bit 0 = **FRES** *Force reset.*
This bit is set by software to force a reset of the USB interface, just as if a RESET sequence came from the USB.
0: Reset not forced
1: USB interface reset forced.

The USB is held in RESET state until software clears this bit, at which point a "USB-RESET" interrupt will be generated if enabled.

USB INTERFACE (Cont'd)

DEVICE ADDRESS REGISTER (DADDR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0

Bit 7 = Reserved, forced by hardware to 0.

Bits 6:0 = **ADD[6:0]** *Device address, 7 bits.*

Software must write into this register the address sent by the host during enumeration.

Note: This register is also reset when a USB reset is received or forced through bit FRES in the USBCTLR register.

USB STATUS REGISTER (USBSR)

Read only

Reset Value: 0000 0000 (00h)

7							0
PID1	PID0	IN/ OUT	0	0	EP2	EP1	EP0

Bits 7:6 = **PID[1:0]** *Token PID bits 1 & 0 for Endpoint 0 Control.*
USB token PIDs are encoded in four bits. PID[1:0] correspond to the most significant bits of the PID field of the last token PID received by Endpoint 0.
Note: The least significant PID bits have a fixed value of 01.
When a CTR interrupt occurs on Endpoint 0 (see register USBISTR) the software should read the PID[1:0] bits to retrieve the PID name of the token received.
The USB specification defines PID bits as:

PID1	PID0	PID Name
0	0	OUT
1	0	IN
1	1	SETUP

Bit 5 = **IN/OUT** *Last transaction direction for Endpoint 1, 2 , 3, 4 or 5.*

This bit is set by hardware when a CTR interrupt occurs on Endpoint 1, 2, 3, 4 or 5.
0: OUT transaction
1: IN transaction

Bits 4:3 = Reserved, forced by hardware to 0.

Bits 2:0 = **EP[2:0]** *Endpoint number.*
These bits identify the endpoint which required attention.
000 = Endpoint 0
001 = Endpoint 1
010 = Endpoint 2
011 = Endpoint 3
100 = Endpoint 4
101 = Endpoint 5

ERROR STATUS REGISTER (ERRSR)

Read only

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	ERR2	ERR1	ERR0

Bits 7:3 = Reserved, forced by hardware to 0.

Bits 2:0 = **ERR[2:0]** *Error type.*

These bits identify the type of error which occurred.

ERR2	ERR1	ERR0	Meaning
0	0	0	No error
0	0	1	Bitstuffing error
0	1	0	CRC error
0	1	1	EOP error (unexpected end of packet or SE0 not followed by J-state)
1	0	0	PID error (PID encoding error, unexpected or unknown PID)
1	0	1	Memory over / underrun (memory controller has not answered in time to a memory data request)
1	1	1	Other error (wrong packet, timeout error)

Note: these bits are set by hardware when an error interrupt occurs and are reset automatically when the error bit (USBISTR bit 4) is cleared by software.

USB INTERFACE (Cont'd)

ENDPOINT 0 REGISTER (EP0R)

Read/Write

Reset value: 0000 0000(00h)

7							0
CTR0	DTOG_TX	STAT_TX1	STAT_TX0	0	DTOG_RX	STAT_RX1	STAT_RX0

This register is used for controlling Endpoint 0. Bits 6:4 and bits 2:0 are also reset by a USB reset, either received from the USB or forced through the FRES bit in USBCTLR.

Bit 7 = **CTR0** *Correct Transfer*.
This bit is set by hardware when a correct transfer operation is performed on Endpoint 0. This bit must be cleared after the corresponding interrupt has been serviced.
0: No CTR on Endpoint 0
1: Correct transfer on Endpoint 0

Bit 6 = **DTOG_TX** *Data Toggle, for transmission transfers*.
It contains the required value of the toggle bit (0=DATA0, 1=DATA1) for the next transmitted data packet. This bit is set by hardware on reception of a SETUP PID. DTOG_TX toggles only when the transmitter has received the ACK signal from the USB host. DTOG_TX and also DTOG_RX are normally updated by hardware, on receipt of a relevant PID. They can be also written by the user, both for testing purposes and to force a specific (DATA0 or DATA1) token.

Bits 5:4 = **STAT_TX [1:0]** *Status bits, for transmission transfers*.

These bits contain the information about the endpoint status, which are listed below

Table 16. Transmission Status Encoding

STAT_TX1	STAT_TX0	Meaning
0	0	**DISABLED:** no function can be executed on this endpoint and messages related to this endpoint are ignored.
0	1	**STALL**: the endpoint is stalled and all transmission requests result in a STALL handshake.
1	0	**NAK**: the endpoint is NAKed and all transmission requests result in a NAK handshake.
1	1	**VALID**: this endpoint is enabled (if an address match occurs, the USB interface handles the transaction).

These bits are written by software. Hardware sets the STAT_TX and STAT_RX bits to NAK when a correct transfer has occurred (CTR=1) addressed to this endpoint; this allows software to prepare the next set of data to be transmitted.

Bit 3 = Reserved, forced by hardware to 0.

Bit 2 = **DTOG_RX** *Data Toggle, for reception transfers*.
It contains the expected value of the toggle bit (0=DATA0, 1=DATA1) for the next data packet. This bit is cleared by hardware in the first stage (Setup Stage) of a control transfer (SETUP transactions start always with DATA0 PID). The receiver toggles DTOG_RX only if it receives a correct data packet and the packet's data PID matches the receiver sequence bit.

USB INTERFACE (Cont'd)

Bits 1:0 = **STAT_RX [1:0]** *Status bits, for reception transfers*.
These bits contain the information about the endpoint status, which are listed below:

Table 17. Reception Status Encoding

STAT_RX1	STAT_RX0	Meaning
0	0	**DISABLED:** no function can be executed on this endpoint and messages related to this endpoint are ignored.
0	1	**STALL:** the endpoint is stalled and all reception requests result in a STALL handshake.
1	0	**NAK**: the endpoint is NAKed and all reception requests result in a NAK handshake.
1	1	**VALID**: this endpoint is enabled (if an address match occurs, the USB interface handles the transaction).

These bits are written by software. Hardware sets the STAT_RX and STAT_TX bits to NAK when a correct transfer has occurred (CTR=1) addressed to this endpoint, so the software has the time to examine the received data before acknowledging a new transaction.

Note 1:

If a SETUP transaction is received while the status is different from DISABLED, it is acknowleded and the two directional status bits are set to NAK by hardware.

Note 2:

When a STALL is answered by the USB device, the two directional status bits are set to STALL by hardware.

ENDPOINT TRANSMISSION REGISTER (EP1TXR, EP2TXR, EP3TXR, EP4TXR, EP5TXR)

Read/Write

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	CTR_TX	DTOG_TX	STAT_TX1	STAT_TX0

This register is used for controlling Endpoint 1, 2, 3, 4 or 5 transmission. Bits 2:0 are also reset by a USB reset, either received from the USB or forced through the FRES bit in the USBCTLR register.

Bits [7:4] = Reserved, forced by hardware to 0.

Bit 3 = **CTR_TX** *Correct Transmission Transfer*.
This bit is set by hardware when a correct transfer operation is performed in transmission. This bit must be cleared after the corresponding interrupt has been serviced.
0: No CTR in transmission on Endpoint 1, 2, 3, 4 or 5
1: Correct transfer in transmission on Endpoint 1, 2, 3, 4 or 5

Bit 2 = **DTOG_TX** *Data Toggle, for transmission transfers*.
This bit contains the required value of the toggle bit (0=DATA0, 1=DATA1) for the next data packet. DTOG_TX toggles only when the transmitter has received the ACK signal from the USB host. DTOG_TX and DTOG_RX are normally updated by hardware, at the receipt of a relevant PID. They can be also written by the user, both for testing purposes and to force a specific (DATA0 or DATA1) token.

Bits [1:0] = **STAT_TX [1:0]** *Status bits, for transmission transfers*.
These bits contain the information about the endpoint status, which is listed below

Table 18. Transmission Status Encoding

STAT_TX1	STAT_TX0	Meaning
0	0	**DISABLED:** transmission transfers cannot be executed.
0	1	**STALL**: the endpoint is stalled and all transmission requests result in a STALL handshake.
1	0	**NAK**: the endpoint is naked and all transmission requests result in a NAK handshake.
1	1	**VALID**: this endpoint is enabled for transmission.

These bits are written by software, but hardware sets the STAT_TX bits to NAK when a correct transfer has occurred (CTR=1) addressed to this endpoint. This allows software to prepare the next set of data to be transmitted.

USB INTERFACE (Cont'd)

ENDPOINT 2 RECEPTION REGISTER (EP2RXR)

Read/Write

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	CTR_RX	DTOG_RX	STAT_RX1	STAT_RX0

This register is used for controlling Endpoint 2 reception. Bits 2:0 are also reset by a USB reset, either received from the USB or forced through the FRES bit in the USBCTLR register.

Bits [7:4] = Reserved, forced by hardware to 0.

Bit 3 = **CTR_RX** *Reception Correct Transfer*.
This bit is set by hardware when a correct transfer operation is performed in reception. This bit must be cleared after that the corresponding interrupt has been serviced.

Bit 2 = **DTOG_RX** *Data Toggle, for reception transfers*.
It contains the expected value of the toggle bit (0=DATA0, 1=DATA1) for the next data packet.

The receiver toggles DTOG_RX only if it receives acorrect data packet and the packet's data PID matches the receiver sequence bit.

Bits [1:0] = **STAT_RX [1:0]** *Status bits, for reception transfers*.
These bits contain the information about the endpoint status, which is listed below:

Table 19. Reception Status Encoding

STAT_RX1	STAT_RX0	Meaning
0	0	**DISABLED:** reception transfers cannot be executed.
0	1	**STALL:** the endpoint is stalled and all reception requests result in a STALL handshake.
1	0	**NAK**: the endpoint is naked and all reception requests result in a NAK handshake.
1	1	**VALID**: this endpoint is enabled for reception.

These bits are written by software, but hardware sets the STAT_RX bits to NAK when a correct transfer has occurred (CTR=1) addressed to this endpoint, so the software has the time to examine the received data before acknowledging a new transaction.

USB INTERFACE (Cont'd)

RECEPTION COUNTER REGISTER (CNT0RXR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	CNT3	CNT2	CNT1	CNT0

This register contains the allocated buffer size for endpoint 0 reception, setting the maximum number of bytes the related endpoint can receive with the next OUT or SETUP transaction. At the end of a reception, the value of this register is the max size decremented by the number of bytes received (to determine the number of bytes received, the software must subtract the content of this register from the allocated buffer size).

TRANSMISSION COUNTER REGISTER (CNT0TXR, CNT1TXR, CNT3TXR, CNT4TXR, CNT5TXR)

Read/Write

Reset Value 0000 0000 (00h)

7							0
0	0	0	0	CNT3	CNT2	CNT1	CNT0

This register contains the number of bytes to be transmitted by Endpoint 0, 1, 3, 4 or 5 at the next IN token addressed to it.

RECEPTION COUNTER REGISTER (CNT2RXR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	CNT6	CNT5	CNT4	CNT3	CNT2	CNT	CNT0

This register contains the allocated buffer size for endpoint 2 reception, setting the maximum number of bytes the related endpoint can receive with the next OUT transaction. At the end of a reception, the value of this register is the max size decremented by the number of bytes received (to determine the number of bytes received, the software must subtract the content of this register from the allocated buffer size).

TRANSMISSION COUNTER REGISTER (CNT2TXR)

Read/Write

Reset Value 0000 0000 (00h)

7							0
0	CNT6	CNT5	CNT4	CNT3	CNT2	CNT1	CNT0

This register contains the number of bytes to be transmitted by Endpoint 2 at the next IN token addressed to it.

USB INTERFACE (Cont'd)

Table 20. USB Register Map and Reset values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
20	USBISTR Reset Value	CTR 0	0 0	SOVR 0	ERR 0	SUSP 0	ESUSP 0	RESET 0	SOF 0
21	USBIMR Reset Value	CTRM 0	0 0	SOVRM 0	ERRM 0	SUSPM 0	ESUSPM 0	RESETM 0	SOFM 0
22	USBCTLR Reset Value	RSM 0	USB_RST 0	0	0	RESUME 0	PDWN 1	FSUSP 1	FRES 0
23	DADDR Reset Value	0	ADD6 0	ADD5 0	ADD4 0	ADD3 0	ADD2 0	ADD1 0	ADD0 0
24	USBSR Reset Value	PID1 0	PID0 0	IN /OUT 0	0	0	EP2 0	EP1 0	EP0 0
25	EP0R Reset Value	CTR0 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0	0 0	DTOG_RX 0	STAT_RX1 0	STAT_RX0 0
26	CNT0RXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
27	CNT0TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
28	EP1TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
29	CNT1TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
2A	EP2RXR Reset Value	0	0	0	0	CTR_RX 0	DTOG_RX 0	STAT_RX1 0	STAT_RX0 0
2B	CNT2RXR Reset Value	0	CNT6 0	CNT5 0	CNT4 0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
2C	EP2TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
2D	CNT2TXR Reset Value	0	CNT6 0	CNT5 0	CNT4 0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
2E	EP3TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
2F	CNT3TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
30	EP4TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
31	CNT4TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
32	EP5TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
33	CNT5TXR	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
34	ERRSR	0	0	0	0	0	ERR2 0	ERR1 0	ERR0 0

12.4 SMARTCARD INTERFACE (CRD)

12.4.1 Introduction

The Smartcard Interface (CRD) provides all the required signals for acting as a smartcard interface device.

The interface is electrically compatible with (and certifiable to) the ISO7816, EMV, GSM and WHQL standards.

Both synchronous (e.g. memory cards) and asynchronous smartcards (e.g. microprocessor cards) are supported.

The CRD generates the required voltages to be applied to the smartcard lines.

The power-off sequence is managed by the CRD.

Card insertion or card removal is detected by the CRD using a card presence switch connected to the external CRDDET pin. If a card is removed, the CRD automatically deactivates the smartcard using the ISO7816 deactivation sequence.

An maskable interrupt is generated when a card is inserted or removed.

Any malfunction is reported to the microcontroller via the Smartcard Interrupt Pending Register (CRDIPR) and Smartcard Status (CRDSR) Registers.

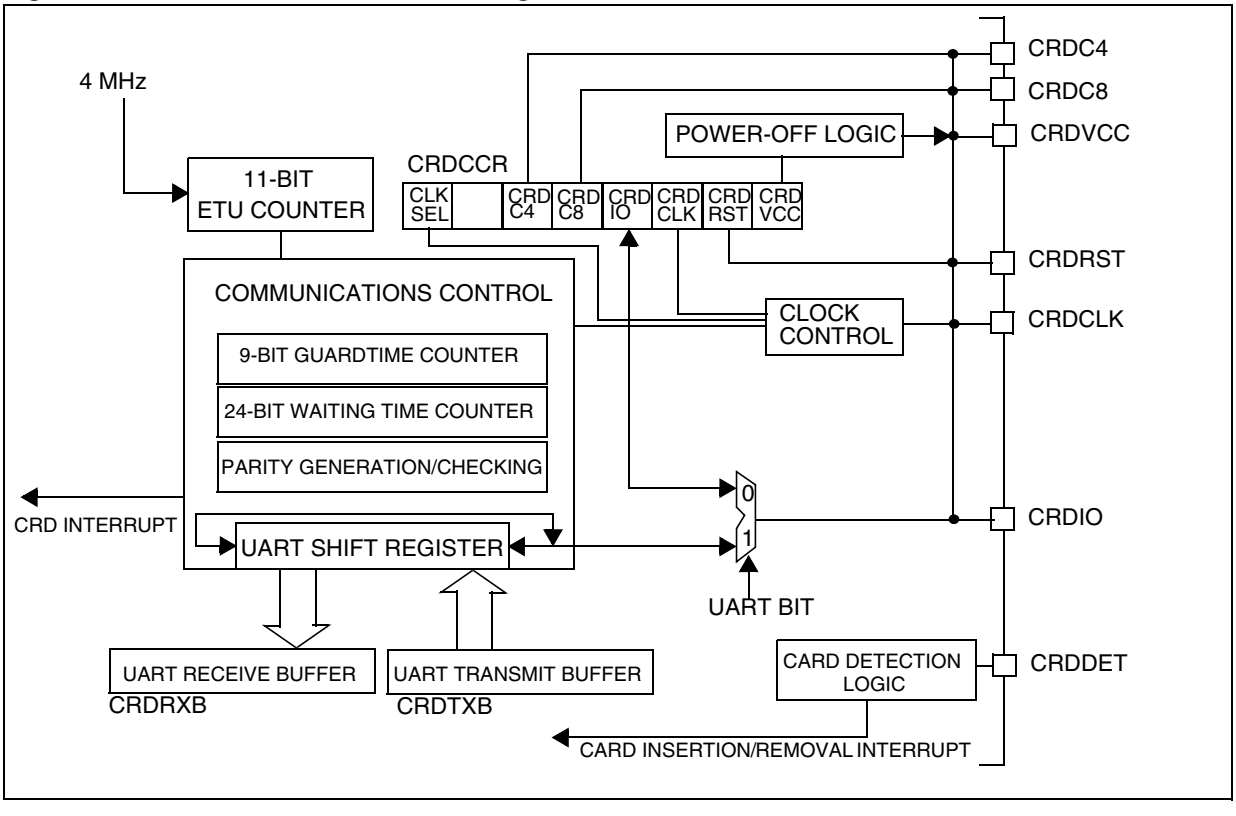
12.4.2 Main features

- Support for ISO 7816-3 standard
- Character mode
- 1 transmit buffer and 1 receive buffer
- 4-Mhz fixed card clock
- 11-bit etu (elementary time unit) counter
- 9-bit guardtime counter
- 24-bit general purpose waiting time counter
- Parity generation and checking
- Automatic character repetition on parity error detection in transmission mode
- Automatic retry on parity error detection in reception mode
- Card power-off deactivation sequence generation
- Manual mode for driving the card I/O directly for synchronous protocols

12.4.3 Functional Description

Figure 30 gives an overview of the smartcard interface.

Figure 30. Smartcard Interface Block Diagram



SMARTCARD INTERFACE (Cont'd)

12.4.3.1 Power Supply Management

Smartcard Power Supply Selection

The Smartcard interface consists of a power supply output on the CRDVCC pin and a set of card interface I/Os which are powered by the same rail.

The card voltage (CRDVCC) is user programmable via the VCARD [1:0] bits in the CRDCR register (refer to the Smartcard Interface section).

Four voltage values can be selected:
5V, 3 V, 1.8 V or 0V.

Current Overload Detection and Card Removal

For each voltage, when an overload current is detected (refer to section 12.4 on page 57), or when a card is removed, the CRDVCC power supply output is directly connected to ground.

12.4.3.2 I/O Driving Modes

Smartcard I/Os are driven in two principal modes:

– UART mode (i.e. when the UART bit of the CRDCR register is set)

– Manual mode, driven directly by software using the Smartcard Contact register (i.e. when the UART bit of the CRDCR register is reset).

Card power-on activation must driven by software.

Card deactivation is handled automatically by the Power-off functional state machine hardware.

12.4.3.3 UART Mode

Two registers are connected to the UART shift register: CRDTXB for transmission and CRDRXB for reception. They act as buffers to off-load the CPU.

A parity checker and generator is coupled to the shifter.

Character repetition and retry are supported.

The UART is in reception mode by default and switches automatically to transmission mode when a byte is written in the buffer.

Priority is given to transmission.

Elementary Time Unit Counter

This 11-bit counter controls the working frequency of the UART. The operating frequency of the clock is the same as the card clock frequency (i.e. 4 MHz).

A compensation mode can be activated via the COMP bit of the CRDETU1 register to allow a frequency granularity down to a half-etu.

Note: The decimal value is limited to a half clock cycle. The bit duration is not fixed. It alternates between n clock cycles and n-1 clock cycles, where n is the value to be written in the CRDETU register. The character duration (10 bits) is also equal to 10*(n - ½) clock cycles This is precise enough to obtain the character duration specified by the ISO7816-3 standard.

For example, if F=372 and D=32 (F being the clock rate conversion factor and D the baud rate adjustment), then etu =11.625 clock cycles.
To achieve this clock rate, compensation mode must be activated and the etu duration must be programmed to 12 clock cycles.
The result will be an average character duration of 11.5 clock cycles (for 10 bits).

See Figure 31.

Guardtime counter

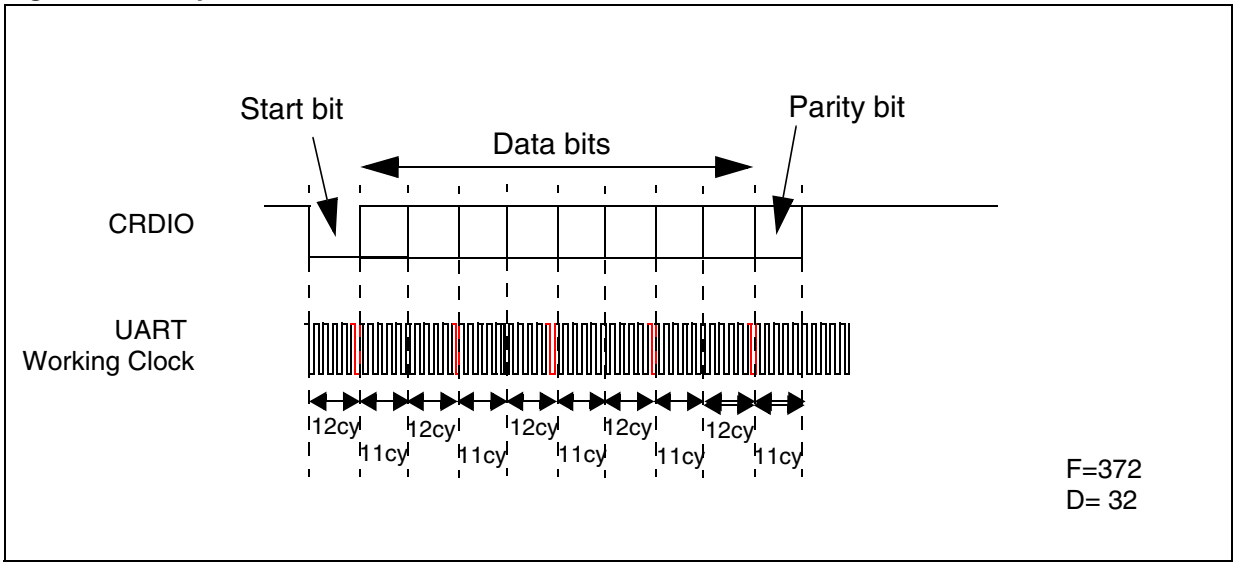
The guardtime counter is a 9-bit counter which manages the character frame. It controls the duration between two consecutive characters in transmission.

It is incremented at the etu rate.

No guardtime is inserted for the first character transmitted.

The guardtime between the last byte received from the card and the next byte transmitted by the reader must be handled by software.

Figure 31. Compensation Mode



SMARTCARD INTERFACE (Cont'd)

Waiting Time Counter

The Waiting Time counter is a 24-bit counter used to generate a timeout signal.

The elementary time unit counter acts as a prescaler to the Waiting Time counter which is incremented at the etu rate.

The Waiting Time Counter can be used in both UART mode and Manual mode and acts in different ways depending on the selected mode.

The CRDWT2, CRDWT1 and CRDWT0 are load registers only, the counter itself is not directly accessible.

UART Mode

The load conditions are either:

– A Start bit is detected while UART bit =1 and the WTEN bit =1.

or

– A write access to the CRDWT2 register is performed while the UART bit = 1 and the WTEN bit = 0. In this case, the Waiting Time counter can be used as a general purpose timer.

In UART mode, if the WTEN bit of the CRDCR register is set, the counter is loaded automatically on start bit detection. Software can change the time out value on-the-fly by writing to the CRDWT registers. For example, in T=1 mode, software must load the Block Waiting Time (BWT) time-out in the CRDWT registers before the start bit of the last transmitted character.
Then, after transmission of this last character, signalled by the TXC interrupt, software must write the CWT value (Character Waiting Time) in the CRDWT registers. See example in Figure 32.

Manual mode

The load conditions are:

– A write access to the CRDWT2 register is performed while the UART bit = 0 and the WTEN bit = 0

In Manual mode, if the WTEN bit of the CRDCR register is reset, the timer acts as a general purpose timer. The timer is loaded when a write access to the CRDWT2 register occurs. The timer starts when the WTEN bit = 1.

12.4.3.4 Interrupt generator

The Smartcard Interface has 2 interrupt vectors:

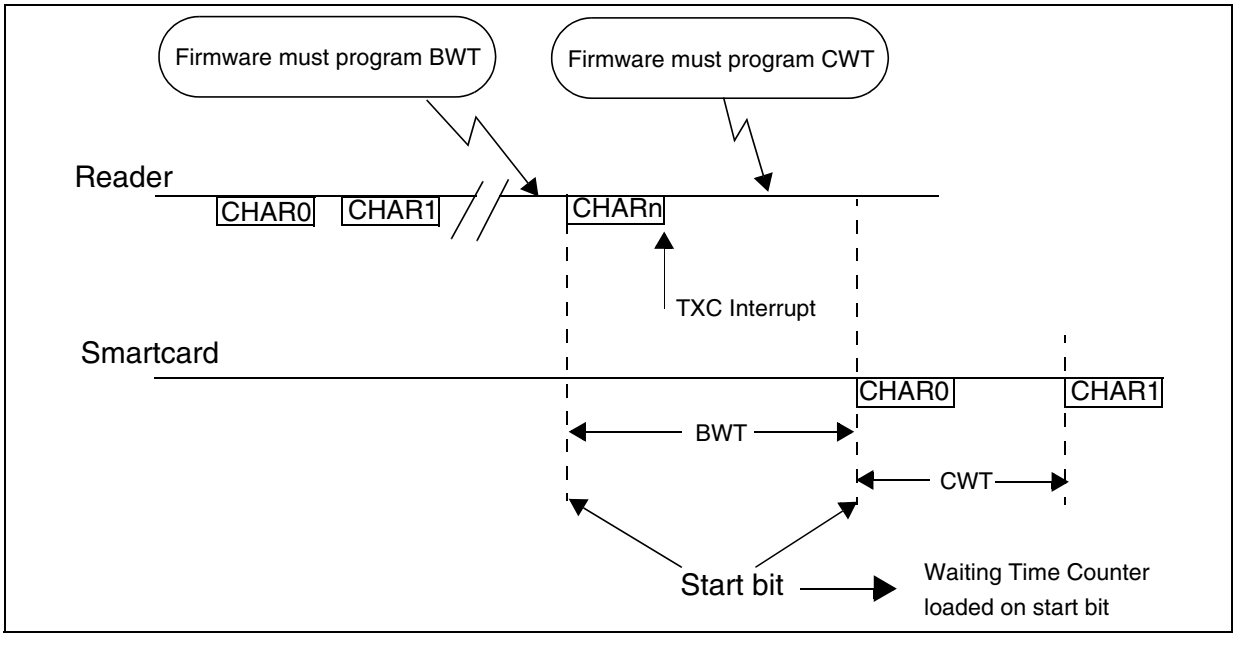
– Card Insertion/Removal Interrupt

– CRD Interrupt

The CRD interrupt is cleared when software reads the CRDIPR register. The Card Insertion/Removal is an external interrupt and is cleared automatically by hardware at the end of the interrupt service routine (IRET instruction).

If an interrupt occurs while the CRDIPR register is being read, the corresponding bit will be set by hardware after the read access is done.

Figure 32. Waiting Time Counter Example



SMARTCARD INTERFACE (Cont'd)

12.4.3.5 Card detection mechanism

The CRDDET bit in the CRDCR Register indicates if the card presence detector (card switch) is open or closed when a card is inserted. When the CRDIRF bit of the CRDSR is set, it indicates that a card is present.

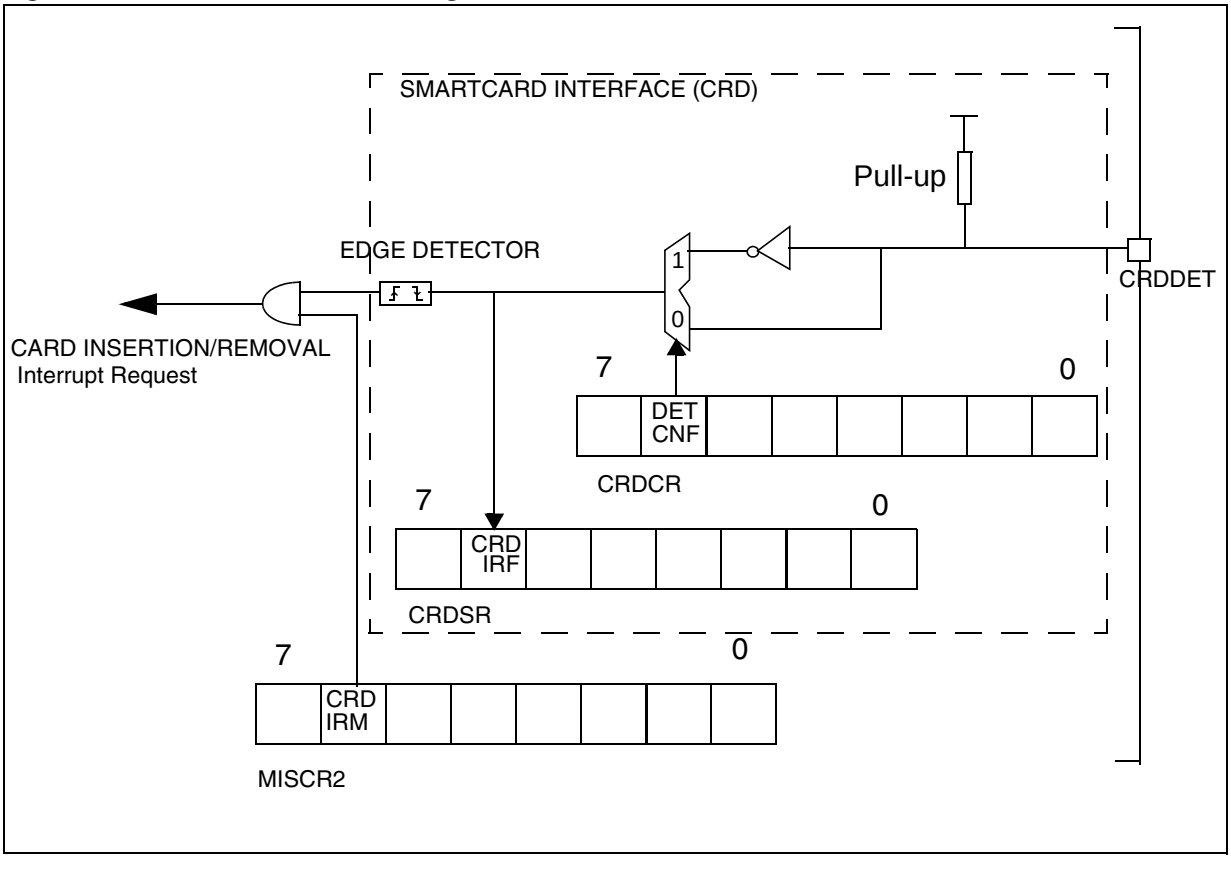
To be able to power-on the smartcard, card presence is mandatory. Removing the smartcard will automatically start the ISO7816-3 card deactivation sequence (see Section 12.4.3.6).

There is no hardware debouncing: The CRDIRF bit changes whenever the level on the CRDDET pin changes. The card switch can generate an interrupt which can be used to wake up the device from suspend mode and for software debouncing.

Three different cases can occur:

– The microcontroller is in run mode, waiting for card insertion:
  Card insertion generates an interrupt and the CRDIRF bit in the CRDSR register is set. Debouncing is managed by software. After the time required for debouncing, if the CRDIRF bit is set, the CRDVCC bit in the CRDCR register is set by software to apply the selected voltage to the CRDVCC pin
– The microcontroller is in suspend mode and a card is inserted:
  The ST7 is woken up by the interrupt. The card insertion is then handled in the same way as in the previous case.
– The card is removed:
  – The CRDIRF bit is reset without hardware debouncing
  – A Card Insertion/Removal interrupt is generated, (if enabled by the CRDIRM bit in the MISCR2 register)
  – The CRDVCC bit is immediately reset by hardware, starting the card deactivation sequence.

Figure 33. Card detection block diagram



SMARTCARD INTERFACE (Cont'd)

12.4.3.6 Card Deactivation Sequence

This sequence can be activated in two different ways:

– Automatically as soon as the card presence detector detects a card removal (via the CRDIRF bit in the CRDSR register, refer to Section 12.4.3.5).

– By software, writing the CRDVCC bit in the CRDCR register, for example:
  – If there is a smartcard current overflow (i.e. when the IOVFF bit in the CRDSR register is set)
  – If the voltage is not within the specified range (i.e. when the VCARDOK bit in the CRDSR register is cleared), but software must clear the CRDVCC bit in the CRDCCR register to start the deactivation sequence.

When the CRDVCC bit is cleared, this starts the deactivation sequence. CRDCLK, CRDIO, CRDC4 and CRDC8 pins are then deactivated as shown in Figure 34:

Figure 34. Card deactivation sequence

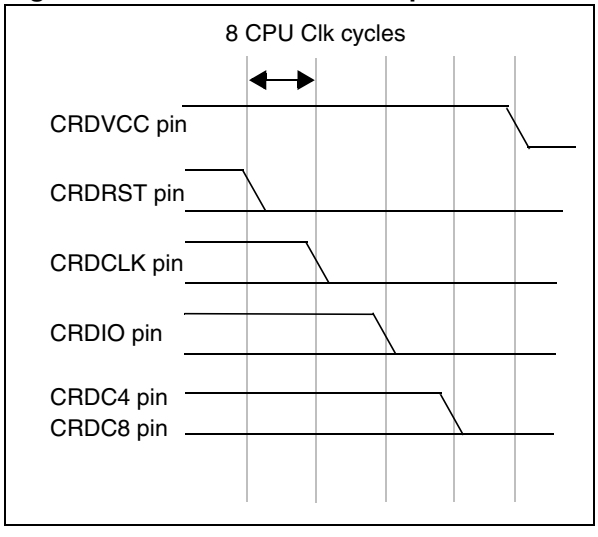
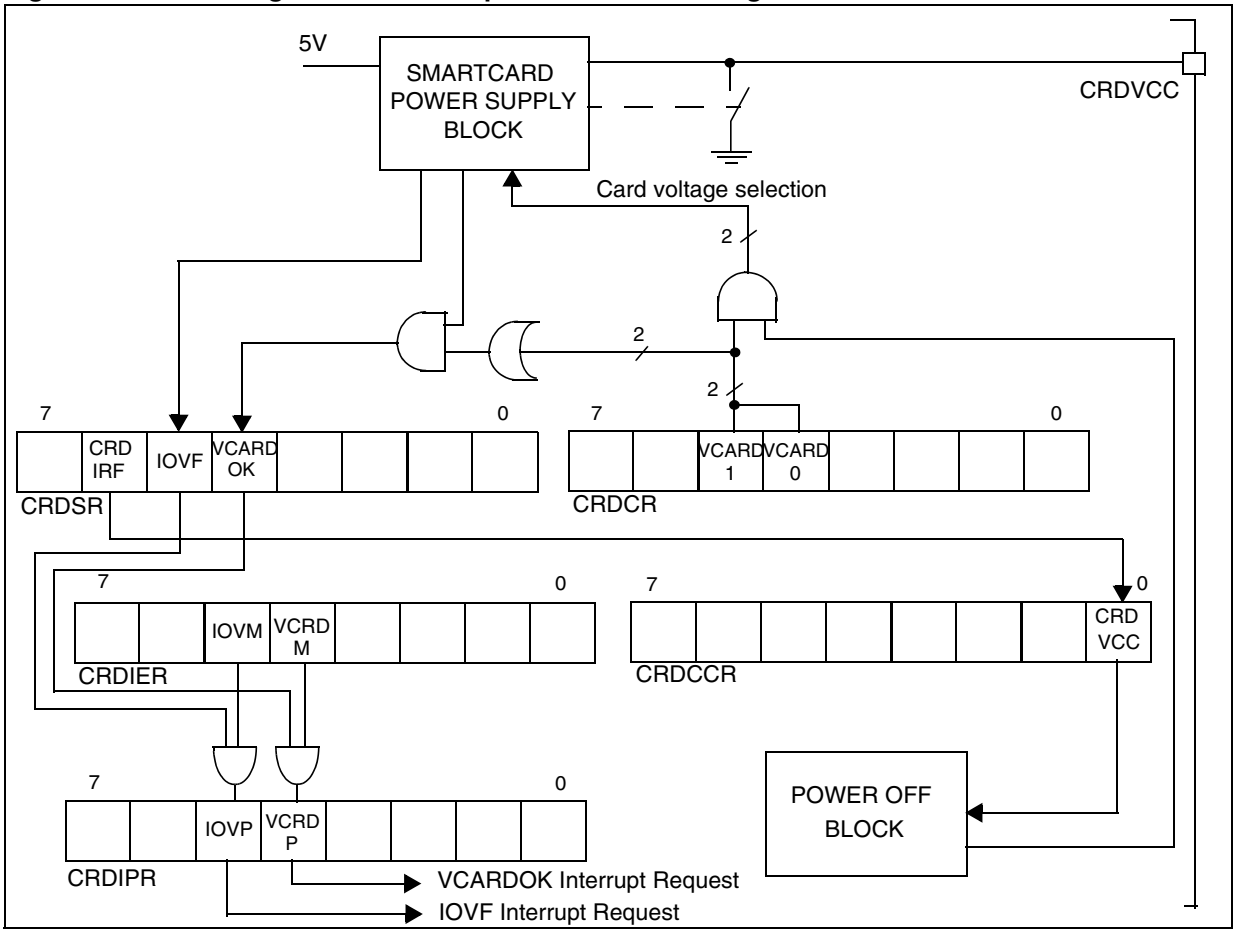
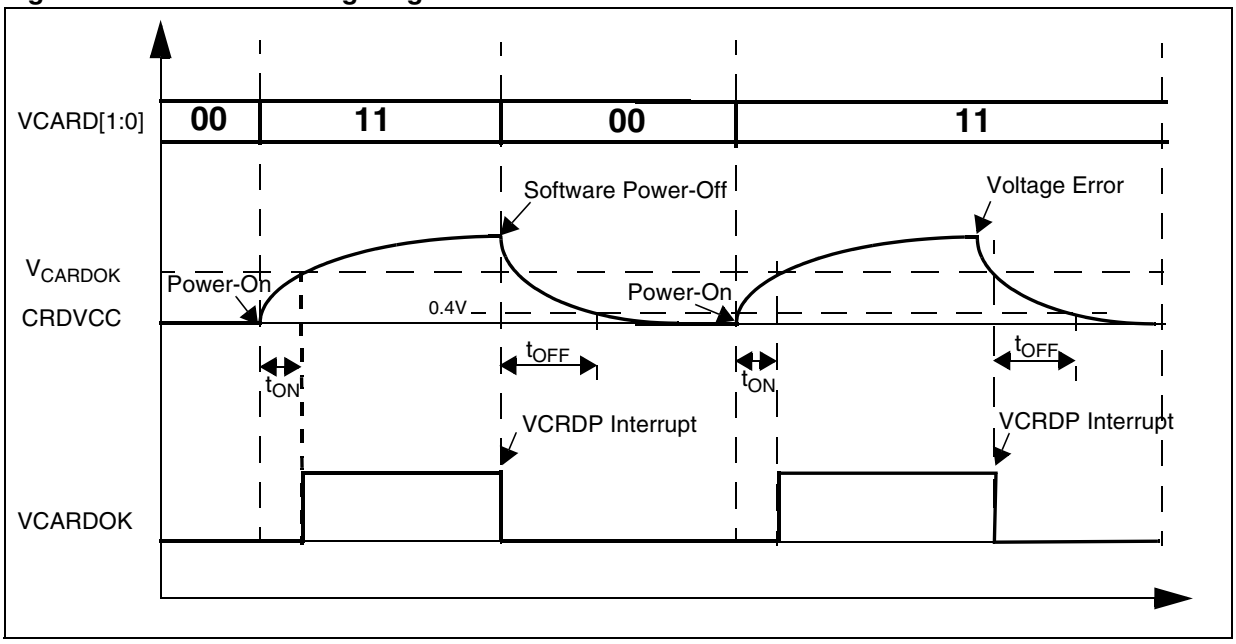


Figure 35. Card voltage selection and power OFF block diagram



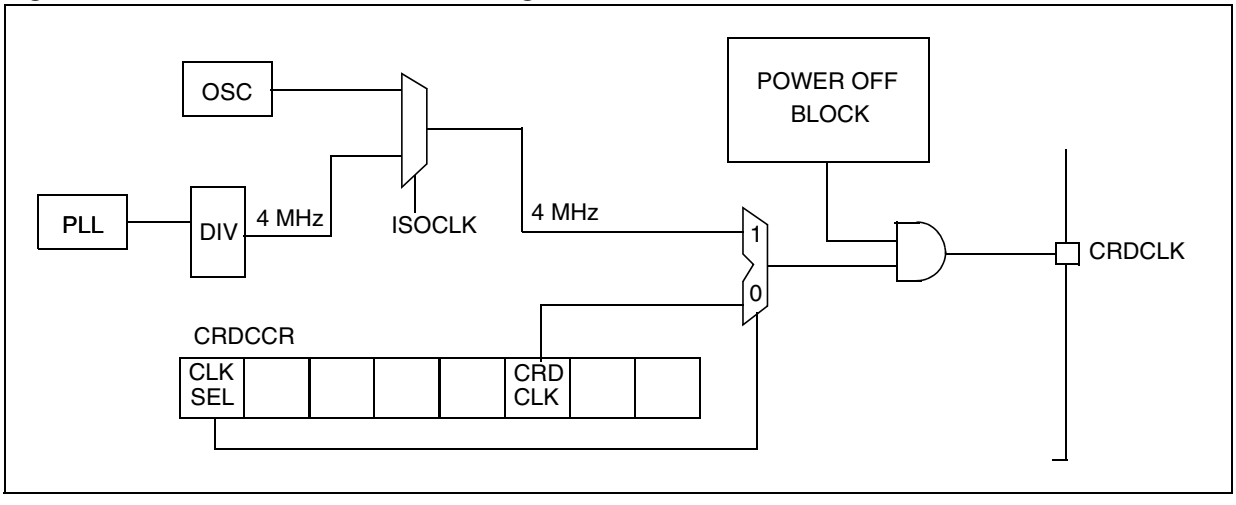
SMARTCARD INTERFACE (Cont'd)

Figure 36. Power Off Timing Diagram



Note: Refer to the Electrical Characteristics section for the values of t_{ON} and t_{OFF}.

Figure 37. Card clock selection block diagram



SMARTCARD INTERFACE (Cont'd)

12.4.4 Register Description

SMARTCARD INTERFACE CONTROL REGISTER (CRDCR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
CRD RST	CRD DET	VCAR D 1	VCAR D 0	U ART	WT EN	C REP	CO NV

Bit 7 = **CRDRST** *Smartcard Interface Reset.*

This bit is set by software to reset the UART of the Smartcard interface.
0: No Smartcard UART Reset
1: Smartcard UART Reset

Bit 6 = **CRDDET** *Card Presence Detector.*
This bit is set and cleared by software to configure the card presence detector switch.
0: Switch open if no card is present
1: Switch closed if no card is present

Bits [5:4] = **VCARD[1:0]** *Card voltage selection.*

These bits select the card voltage.

Bit 1	Bit 0	Vcard
0	0	0V
0	1	1.8V
1	0	3V
1	1	5V

Bit 3 = **UART** *UART Mode Selection.*
This bit is set and cleared by software to select UART or manual mode.
0: CRDIO pin is a copy of the CRDIO bit in the CRDCCR register (Manual mode).
1: CRDIO pin is the output of the smartcard UART (UART mode).

Caution: Before switching from Manual mode to UART mode, software must set the CRDIO bit in the CRDCCR register.

Bit 2 = **WTEN** *Waiting Time Counter enable.*
0: Waiting Time counter stopped. While WTEN = 0, a write access to the CRDWT2 register loads the Waiting time counter with the load value held in the CRDWT0, CRDWT1 and CRDWT2 registers.
1: Start counter. In UART mode, the counter is automatically reloaded on start bit detection.

Bit 1 = **CREP** *Automatic character repetition in case of parity error.*
0: In reception mode: no parity error signal indication (no retry on parity error).
In transmission mode: no error signal processing. No retransmission of a refused character on parity error.
1: Automatic parity management:
In transmission mode: up to 4 character repetitions on parity error.
In reception mode: up to 4 retries are made on parity error.

The PARF parity error flag is set by hardware if a parity error is detected.

If the transmitted character is refused, the PARF bit is set (but the TXCF bit is reset) and an interrupt is generated if the PARM bit is set.

Note: If CREP=1, the PARF flag is set at the 5th error (after 4 character repetitions or 4 retries).

If CREP=0, the PARF bit is set after the first parity error.

Bit 0 = **CONV** *ISO convention selection.*
0: Direct convention, the B0 bit (LSB) is sent first, a '1' is a level 1 on the Card I/O pin, the parity bit is added after the B7 bit.
1: Inverse convention, the B7 bit (MSB) is sent first, a '1' is a level 0 on Card I/O pin, the parity bit is added after the B0 bit.

Note: To detect the convention used by any card, apply the following rule. If a card uses the convention selected by the reader, an RXC event occurs at answer to reset. Otherwise a parity error also occurs.

SMARTCARD INTERFACE (Cont'd)

SMARTCARD INTERFACE STATUS REGISTER (CRDSR)

Read only (Read/Write on some bits)
Reset Value: 1000 0000 (80h)

7							0
TXBEF	CRDIRF	IOVF	VCARDOK	WTF	TXCF	RXCF	PARF

Bit 7 =**TxBEF** *Transmit Buffer Empty Flag.*
- Read only
0: Transmit buffer is not empty
1: Transmit buffer is empty

Bit 6 = **CRDIRF** *Card Insertion/Removal Flag.*
- Read only
0: No card is present
1: A card is present

Bit 5 = **IOVF** *Card Overload Current Flag.*
- Read only
0: No card overload current
1: Card overload current

Bit 4 = **VCARDOK** *Card voltage status Flag.*
- Read only
0: The card voltage is not in the specified range
1: The card voltage is within the specified range

Bit 3 = **WTF** *Waiting Time Counter overflow Flag.*
- Read only
0: The WT Counter has not reached its maximum value
1: The WT Counter has reached its maximum value

Bit 2 = **TXCF** *Transmitted character Flag.*
- Read/Write
This bit is set by hardware and cleared by software.
0: No character transmitted
1: A character has been transmitted

Bit 1 = **RXCF** *Received character Flag.*
- Read only
This bit is set by hardware and cleared by hardware when the CRDRXB buffer is read.
0: No character received
1: A character has been received

Bit 0 = **PARF** *Parity Error Flag.*
- Read/Write
This bit is set by hardware and cleared by software.
0: No parity error
1: Parity error

Note: When a character is received, the RXCF bit is always set.When a character is received with a parity error, the PARF bit is also set.

SMARTCARD INTERFACE (Cont'd)

SMARTCARD CONTACT CONTROL REGISTER (CRDCCR)
Read/Write
Reset Value: 00xx xx00 (xxh)

7							0
CLK SEL	-	CRD C8	CRD C4	CRD IO	CRD CLK	CRD RST	CRD VCC

Note: To modify the content of this register, the LD instruction must be used (do not use the BSET and BRES instructions).

Bit 7 = **CLKSEL** *Card clock selection.*

This bit is set and cleared by software.
0: The signal on the CRDCLK pin is a copy of the CRDCLK bit.
1: The signal on the CRDCLK pin is a 4MHz frequency clock.

Note: To start the clock at a known level, the CRDCLK bit should be changed before the CLKSEL bit.

Bit 6 = Reserved, must be kept cleared.

Bit 5 = **CRDC8** *CRDC8 pin control.*
Reading this bit returns the value present on the CRDC8 pin. Writing this bit outputs the bit value on the pin.

Bit 4 = **CRDC4** *CRDC4 pin control*
Reading this bit returns the value present on the CRDC4 pin. Writing this bit outputs the bit value on the pin.

Bit 3 = **CRDIO** *CRDIO pin control*.

This bit is active only if the UART bit in the CRDCR Register is reset. Reading this bit returns the value present on the CRDIO pin.

If the UART bit is reset:

– Writing "0" forces a low level on the CRDIO pin
– Writing "1" forces the CRDIO pin to open drain Hi-Z.

Bit 2 = **CRDCLK** *CRDCLK pin control*

This bit is active only if the CLKSEL bit of the CRDCCR register is reset. Reading this bit returns the value present in the register (not the CRDCLK pin value).

When the CLKSEL bit is reset:
0: Level 0 to be applied on CRDCLK pin.
1: Level 1 to be applied on CRDCLK pin.

Note: To ensure that the clock stops at a given value, write the desired value in the CRDCLK bit prior to changing the CLKSEL bit from 1 to 0.

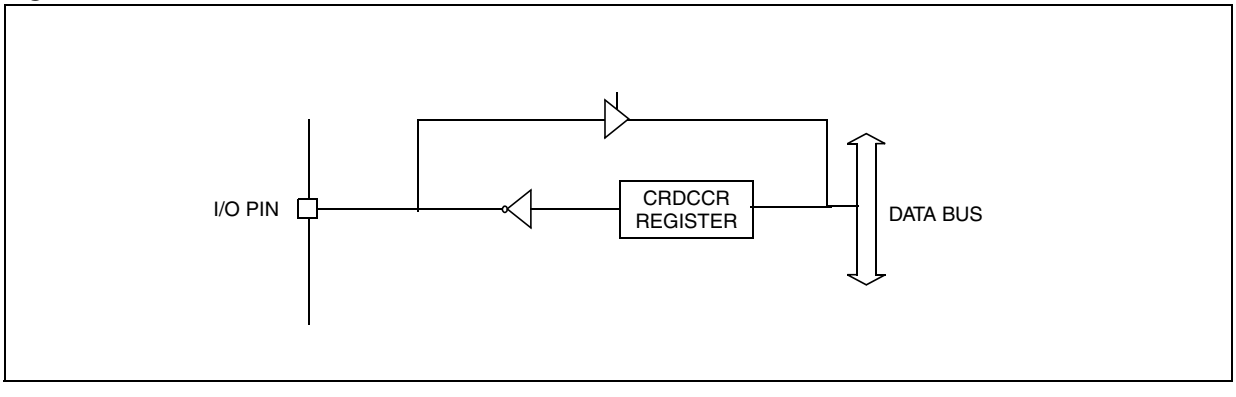
Bit 1 = **CRDRST** *CRDRST pin control.*

Reading this bit returns the value present on the CRDRST pin. Writing this bit outputs the bit value on the pin.

Bit 0 = **CRDVCC** *CRDVCC Pin Control.*

This bit is set and cleared by software and forced to 0 by hardware when no card is present (CRDIRF bit=0).
0: No voltage to be applied on the CRDVCC pin.
1: The selected voltage must be applied on the CRDVCC pin.

Figure 38. Smartcard I/O Pin Structure



SMARTCARD INTERFACE (Cont'd)

SMARTCARD ELEMENTARY TIME UNIT REGISTER (CRDETUx)

CRDETU1
Read/Write
Reset Value: 0000 0001 (01h)

7							0
COMP	0	0	0	0	ETU10	ETU9	ETU8

Bit 7 = **COMP** *Elementary Time Unit Compensation.*
0: Compensation mode disabled.
1: Compensation mode enabled. To allow non integer value, one clock cycle is subtracted from the ETU value on odd bits. See Figure 31.

Bit [6:3] = Reserved

Bits 2:0 = **ETU [10:8]** *ETU value in card clock cycles.*

Writing CRDETU1 register reloads the ETU counter.

CRDETU0
Read/Write
Reset Value: 0111 0100 (74h)

7							0
ETU7	ETU6	ETU5	ETU4	ETU3	ETU2	ETU1	ETU0

Bits 7:0 = **ETU [7:0]** *ETU value in card clock cycles.*

Note: The value of ETU [10:0] must in the range 12 to 2047. To write 2048, clear all the bits.

GUARDTIME REGISTER (CRDGTx)

CRDGT1
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	GT8

CRDGT0
Read/Write
Reset Value: 0000 1100 (0Ch)

7							0
GT7	GT6	GT5	GT4	GT3	GT2	GT1	GT0

Software writes the Guardtime value in this register. The value is loaded at the end of the current Guard period.

GT: Guard Time: Minimum time between two consecutive start bits in transmission mode. Value expressed in Elementary Time Units (from 11 to 511).

The Guardtime between the last byte received from the card and the next byte transmitted by the reader must be handled by software.

SMARTCARD INTERFACE (Cont'd)

CHARACTER WAITING TIME REGISTER (CRDWTx)

CRDWT2

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

7							0
WT 23	WT 22	WT 21	WT 20	WT 19	WT 18	WT 17	WT 16

CRDWT1

Read/Write
Reset Value: 0010 0101 (25h)

7							0
WT 15	WT 14	WT 13	WT 12	WT 11	WT 10	WT9	WT8

CRDWT0

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

7							0
WT 7	WT6	WT5	WT4	WT3	WT2	WT1	WT0

WT: Character waiting time value expressed in ETU (0 / 16777215).

The CRDWT0, CRDWT1 and CRDWT2 registers hold the load value of the Waiting Time counter.

Note: A read operation does not return the counter value.

This counter can be used as a general purpose timer.

If the WTEN bit of the CRDCR register is reset, the counter is reloaded when a write access in the CRDWT2 register occurs. It starts when the WTEN bit is set.

If the WTEN bit in the CRDCR register is set and if UART mode is activated, the counter acts as an autoreload timer. The timer is reloaded when a start bit is sent or detected. An interrupt is generated if the timer overflows between two consecutive start bits.

Note: When loaded with a 0 value, the Waiting Time counter stays at 0 and the WTF bit = 1.

SMARTCARD INTERFACE (Cont'd)

SMARTCARD INTERRUPT ENABLE REGISTER (CRDIER)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
TXBEM	-	IOVFM	VCRDM	WTM	TXCM	RXCM	PARM

Bit 7 = **TXBEM** *Transmit buffer empty interrupt mask.*

This bit is set and cleared by software to enable or disable the TXBE interrupt.
0: TXBE interrupt disabled
1: TXBE interrupt enabled

Bit 6 = Reserved.

Bit 5 = **IOVFM** *Card Overload Current Interrupt Mask.*
This bit is set and cleared by software to enable or disable the IOVF interrupt.
0: IOVF interrupt disabled
1: IOVF interrupt enabled

Bit 4= **VCRDM** *Card Voltage Error Interrupt Mask.*

This bit is set and cleared by software to enable or disable the VCRD interrupt.
0: VCRD interrupt disabled
1: VCRD interrupt enabled

Bit 3 = **WTM** *Waiting Timer Interrupt Mask.*
This bit is set and cleared by software to enable or disable the Waiting Timer overflow interrupt.
0: WT interrupt disabled
1: WT interrupt enabled

Bit 2 =**TXCM** *Transmitted Character Interrupt Mask*
This bit is set and cleared by software to enable or disable the TXC interrupt.
0: TXC interrupt disabled
1: TXC interrupt enabled

Bit 1 =**RXCM** *Received Character Interrupt Mask*
This bit is set and cleared by software to enable or disable the RXC interrupt.
0: RXC interrupt disabled
1: RXC interrupt enabled

Bit 0 = **PARM** *Parity Error Interrupt. Mask*
This bit is set and cleared by software to enable or disable the parity error interrupt for parity error.
0: PAR interrupt disabled
1: PAR error interrupt enabled

SMARTCARD INTERFACE (Cont'd)

SMARTCARD INTERRUPT PENDING REGISTER (CRDIPR)
Read Only
Reset Value: 0000 0000 (00h)

7							0
TXBEP	-	IOVFP	VCRDP	WTP	TXCP	RXCP	PARP

This register indicates the interrupt source. It is cleared after a read operation.

Bit 7 = **TXBEP** *Transmit buffer empty interrupt pending.*

This bit is set by hardware when a TXBE event occurs and the TXBEM bit is set.
0: No TXBE interrupt pending
1: TXBE interrupt pending

Bit 6 = Reserved.

Bit 5 = **IOVF** *Card Overload Current interrupt pending.*

This bit is set by hardware when a IOVF event occurs and the IOVFM bit is set.
0: No IOVF interrupt pending
1: IOVF interrupt pending

Bit 4 = **VCRDP** *Card Voltage Error interrupt pending.*

This bit is set by hardware when the VCARDOK bit goes from 1 to 0 while the VCRDM bit is set.
0: No VCRD interrupt pending.
1: VCRD interrupt pending.

Bit 3 = **WTP** *Waiting Timer Overflow interrupt pending.*

This bit is set by hardware when a WTP event occurs and the WTPM bit is set.
0: No WT interrupt pending
1: WT interrupt pending

Bit 2 = **TXCP** *Transmitted character interrupt pending.*

This bit is set by hardware when a character is transmitted and the TXCM bit is set. It indicates that the CRDTXB buffer can be loaded with the next character to be transmitted.
0: No TXC interrupt pending
1: TXC interrupt pending

Bit 1 = **RXCP** *Received character interrupt pending.*

This bit is set by hardware when a character is received and the RXCM bit is set. It indicates that the CRDRXB buffer can be read.
0: No RXC interrupt pending
1: RXC interrupt pending

Bit 0 = **PARP** *Parity Error interrupt pending.*

This bit is set by hardware when a PAR event occurs and the PARM bit is set.
0: No PAR interrupt pending
1: PAR interrupt pending

SMARTCARD TRANSMIT BUFFER (CRDTXB)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0

This register is used to send a byte to the smartcard.

SMARTCARD RECEIVE BUFFER (CRDRXB)
Read
Reset Value: 0000 0000 (00h)

7							0
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0

This register is used to receive a byte from the smartcard.

SMARTCARD INTERFACE (Cont'd)

Table 21. Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
00	**CRDCR** Reset Value	CRDRST 0	DETCNF 0	VCARD1 0	VCARD0 0	UART 0	WTEN 0	CREP 0	CONV 0
01	**CRDSR** Reset Value	TXBEF 1	CRDIRF 0	IOVF 0	VCARDOK 0	WTF 0	TXCF 0	RXCF 0	PARF 0
02	**CRDCCR** Reset Value	CLKSEL 0	- 0	CRDC8 x	CRDC4 x	CRDIO x	CRDCLK 0	CRDRST x	CRDVCC 0
03	**CRDETU1** Reset Value	COMP 0	- 0	- 0	- 0	- 0	ETU10 1	ETU9 0	ETU8 0
04	**CRDETU0** Reset Value	ETU7 0	ETU6 1	ETU5 1	ETU4 1	ETU3 0	ETU2 1	ETU1 0	ETU0 0
05	**CRDGT1** Reset Value	- 0	- 0	- 0	- 0	- 0	- 0	- 0	GT8 0
06	**CRDGT0** Reset Value	GT7 0	GT6 0	GT5 0	GT4 0	GT3 1	GT2 1	GT1 0	GT0 0
07	**CRDWT2** Reset Value	WT23 0	WT22 0	WT21 0	WT20 0	WT19 0	WT18 0	WT17 0	WT16 0
08	**CRDWT1** Reset Value	WT15 0	WT14 0	WT13 1	WT12 0	WT11 0	WT10 1	WT9 0	WT8 1
09	**CRDWT0** Reset Value	WT7 1	WT6 0	WT5 0	WT4 0	WT3 0	WT2 0	WT1 0	WT0 0
0A	**CRDIER** Reset Value	TXBEM 0	- 0	IOVM 0	VCRDM 0	WTM 0	TXCM 0	RXCM 0	PARM 0
0B	**CRDIPR** Reset Value	TXBEP 0	- 0	IOVP 0	VCRDP	WTP 0	TXCP 0	RXCP 0	PARP 0
0C	**CRDTXB** Reset Value	TB7 0	TB6 0	TB5 0	TB4 0	TB3 0	TB2 0	TB1 0	TB0 0
0D	**CRDRXB** Reset Value	RB7 0	RB6 0	RB5 0	RB4 0	RB3 0	RB2 0	RB1 0	RB0 0

13 INSTRUCTION SET

13.1 CPU ADDRESSING MODES

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

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

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

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

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

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

Table 22. CPU Addressing Mode Overview

Mode			Syntax	Destination	Pointer Address (Hex.)	Pointer Size (Hex.)	Length (Bytes)
Inherent			nop				+ 0
Immediate			ld A,#$55				+ 1
Short	Direct		ld A,$10	00..FF			+ 1
Long	Direct		ld A,$1000	0000..FFFF			+ 2
No Offset	Direct	Indexed	ld A,(X)	00..FF			+ 0
Short	Direct	Indexed	ld A,($10,X)	00..1FE			+ 1
Long	Direct	Indexed	ld A,($1000,X)	0000..FFFF			+ 2
Short	Indirect		ld A,[$10]	00..FF	00..FF	byte	+ 2
Long	Indirect		ld A,[$10.w]	0000..FFFF	00..FF	word	+ 2
Short	Indirect	Indexed	ld A,([$10],X)	00..1FE	00..FF	byte	+ 2
Long	Indirect	Indexed	ld A,([$10.w],X)	0000..FFFF	00..FF	word	+ 2
Relative	Direct		jrne loop	PC+/-127			+ 1
Relative	Indirect		jrne [$10]	PC+/-127	00..FF	byte	+ 2
Bit	Direct		bset $10,#7	00..FF			+ 1
Bit	Indirect		bset [$10],#7	00..FF	00..FF	byte	+ 2
Bit	Direct	Relative	btjt $10,#7,skip	00..FF			+ 2
Bit	Indirect	Relative	btjt [$10],#7,skip	00..FF	00..FF	byte	+ 3

INSTRUCTION SET OVERVIEW (Cont'd)

13.1.1 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

13.1.2 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

13.1.3 Direct

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

The direct addressing mode consists of two submodes:

Direct (short)

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

Direct (long)

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

13.1.4 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.

13.1.5 Indirect (Short, Long)

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

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

Indirect (short)

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

Indirect (long)

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

INSTRUCTION SET OVERVIEW (Cont'd)

13.1.6 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 23. Instructions Supporting Direct, Indexed, Indirect and Indirect Indexed Addressing Modes

Long and Short Instructions	Function
LD	Load
CP	Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Additions/Substractions operations
BCP	Bit Compare

Short Instructions Only	Function
CLR	Clear
INC, DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
BSET, BRES	Bit Operations
BTJT, BTJF	Bit Test and Jump Operations
SLL, SRL, SRA, RLC, RRC	Shift and Rotate Operations
SWAP	Swap Nibbles
CALL, JP	Call or Jump subroutine

13.1.7 Relative mode (Direct, Indirect)

This addressing mode is used to modify the PC register value, by adding an 8-bit signed offset to it.

Available Relative Direct/Indirect Instructions	Function
JRxx	Conditional Jump
CALLR	Call Relative

The relative addressing mode consists of two submodes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

The offset is defined in memory, which address follows the opcode.

INSTRUCTION SET OVERVIEW (Cont'd)

13.2 INSTRUCTION GROUPS

The ST7 family devices use an Instruction Set consisting of 63 instructions. The instructions may be subdivided into 13 main groups as illustrated in the following table:

Load and Transfer	LD	CLR						
Stack operation	PUSH	POP	RSP					
Increment/Decrement	INC	DEC						
Compare and Tests	CP	TNZ	BCP					
Logical operations	AND	OR	XOR	CPL	NEG			
Bit Operation	BSET	BRES						
Conditional Bit Test and Branch	BTJT	BTJF						
Arithmetic operations	ADC	ADD	SUB	SBC	MUL			
Shift and Rotates	SLL	SRL	SRA	RLC	RRC	SWAP	SLA	
Unconditional Jump or Call	JRA	JRT	JRF	JP	CALL	CALLR	NOP	RET
Conditional Branch	JRxx							
Interruption management	TRAP	WFI	HALT	IRET				
Condition Code Flag modification	SIM	RIM	SCF	RCF				

Using a pre-byte

The instructions are described with one to four opcodes.

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

The whole instruction becomes:

PC-2 End of previous instruction

PC-1 Prebyte

PC opcode

PC+1 Additional word (0 to 2) according to the number of bytes required to compute the effective address

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

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

PIX 92 Replace an instruction using direct, direct bit, or direct relative addressing mode to an instruction using the corresponding indirect addressing mode.
It also changes an instruction using X indexed addressing mode to an instruction using indirect X indexed addressing mode.

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

INSTRUCTION SET OVERVIEW (Cont'd)

Mnemo	Description	Function/Example	Dst	Src	I1	H	I0	N	Z	C
ADC	Add with Carry	A = A + M + C	A	M		H		N	Z	C
ADD	Addition	A = A + M	A	M		H		N	Z	C
AND	Logical And	A = A . M	A	M				N	Z	
BCP	Bit compare A, Memory	tst (A . M)	A	M				N	Z	
BRES	Bit Reset	bres Byte, #3	M							
BSET	Bit Set	bset Byte, #3	M							
BTJF	Jump if bit is false (0)	btjf Byte, #3, Jmp1	M							C
BTJT	Jump if bit is true (1)	btjt Byte, #3, Jmp1	M							C
CALL	Call subroutine									
CALLR	Call subroutine relative									
CLR	Clear		reg, M					0	1	
CP	Arithmetic Compare	tst(Reg - M)	reg	M				N	Z	C
CPL	One Complement	A = FFH-A	reg, M					N	Z	1
DEC	Decrement	dec Y	reg, M					N	Z	
HALT	Halt				1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC			I1	H	I0	N	Z	C
INC	Increment	inc X	reg, M					N	Z	
JP	Absolute Jump	jp [TBL.w]								
JRA	Jump relative always									
JRT	Jump relative									
JRF	Never jump	jrf *								
JRIH	Jump if ext. INT pin = 1	(ext. INT pin high)								
JRIL	Jump if ext. INT pin = 0	(ext. INT pin low)								
JRH	Jump if H = 1	H = 1 ?								
JRNH	Jump if H = 0	H = 0 ?								
JRM	Jump if I1:0 = 11	I1:0 = 11 ?								
JRNM	Jump if I1:0 <> 11	I1:0 <> 11 ?								
JRMI	Jump if N = 1 (minus)	N = 1 ?								
JRPL	Jump if N = 0 (plus)	N = 0 ?								
JREQ	Jump if Z = 1 (equal)	Z = 1 ?								
JRNE	Jump if Z = 0 (not equal)	Z = 0 ?								
JRC	Jump if C = 1	C = 1 ?								
JRNC	Jump if C = 0	C = 0 ?								
JRULT	Jump if C = 1	Unsigned <								
JRUGE	Jump if C = 0	Jmp if unsigned >=								
JRUGT	Jump if (C + Z = 0)	Unsigned >								

INSTRUCTION SET OVERVIEW (Cont'd)

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

14 ELECTRICAL CHARACTERISTICS

14.1 ABSOLUTE MAXIMUM RATINGS

This product contains devices for protecting the inputs against damage due to high static voltages, however it is advisable to take normal precautions to avoid appying any voltage higher than the specified maximum rated voltages.

For proper operation it is recommended that V_I and V_O be higher than V_{SS} and lower than V_{DD}. Reliability is enhanced if unused inputs are connected to an appropriate logic voltage level (V_{DD} or V_{SS}).

Power Considerations. The average chip-junction temperature, T_J, in Celsius can be obtained from:

T_J = TA + PD x RthJA

Where: T_A = Ambient Temperature.

RthJA = Package thermal resistance (junction-to ambient).

P_D = P_{INT} + P_{PORT}.

P_{INT} = I_{DD} x V_{DD} (chip internal power).

P_{PORT} = Port power dissipation determined by the user)

Stresses above those listed as “absolute maximum ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these conditions is not implied. Exposure to maximum rating for extended periods may affect device reliability.

Symbol	Ratings	Value	Unit
V_{DD} - V_{SS}	Supply voltage	6.0	V
V_{IN}	Input voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
V_{OUT}	Output voltage	V_{SS} - 0.3 to V_{DD} + 0.3	V
ESD	ESD susceptibility	2000	V
ESDCard	ESD susceptibility for card pads	4000	V
I_{VDD_i}	Total current into V_{DD_i} (source)	250	mA
I_{VSS_i}	Total current out of V_{SS_i} (sink)	250	

General Warning: Direct connection to V_{DD} or V_{SS} of the I/O pins could damage the device in case of program counter corruption (due to unwanted change of the I/O configuration). To guarantee safe conditions, this connection has to be done through a typical 10KΩ pull-up or pull-down resistor.

Thermal Characteristics

Symbol	Ratings		Value	Unit
R_{thJA}	Package thermal resistance	TQFP64 SO24	60 80	°C/W
T_{Jmax}	Max. junction temperature		150	°C
T_{STG}	Storage temperature range		-65 to +150	°C
PD	Power dissipation (maximum value)		500	mW

14.2 RECOMMENDED OPERATING CONDITIONS

GENERAL						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{DD}	Supply voltage		4.0		5.5	V
f_{OSC}	External clock source			4		MHz
T_A	Ambient temperature range		0		70	°C

(Operating conditions T_A = 0 to +70°C unless otherwise specified)

CURRENT INJECTION ON I/O PORT AND CONTROL PINS						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
I_{INJ+}	Total positive injected current (1,2)	$V_{EXTERNAL} > V_{DD}$ (Standard I/Os) $V_{EXTERNAL} > V_{CRDVCC}$ (Smartcard I/Os)			20	mA
I_{INJ-}	Total negative injected current (3)	$V_{EXTERNAL} < V_{SS}$ Digital pins Analog pins			20	mA

Note 1: Positive injection

The I_{INJ+} is done through protection diodes insulated from the substrate of the die.

Note 2: For SmartCard I/Os, V_{CRDVCC} has to be considered.

Note 3: Negative injection

– The I_{INJ-} is done through protection diodes NOT INSULATED from the substrate of the die. The drawback is a small leakage (few µA) induced inside the die when a negative injection is performed. This leakage is tolerated by the digital structure, but it acts on the analog line according to the impedance versus a leakage current of few µA (if the MCU has an AD converter). The effect depends on the pin which is submitted to the injection. Of course, external digital signals applied to the component must have a maximum impedance close to 50KΩ.

Location of the negative current injection:

– Pure digital pins can tolerate 1.6mA. In addition, the best choice is to inject the current as far as possible from the analog input pins.

General Note: When several inputs are submitted to a current injection, the maximum I_{INJ} is the sum of the positive (resp. negative) currrents (instantaneous values).

RECOMMENDED OPERATING CONDITIONS (Cont'd)

(T_A=0 to +70°C, V_{DD}-V_{SS}=5.5V unless otherwise specified)

Symbol	Parameter	Conditions	Min	Typ.	Max	Unit
I_{DD}	Supply current in RUN mode [1]	f_{OSC} = 4MHz		10	15	mA
	Supply current in WAIT mode [2]			3		mA
	Supply current in suspend mode	External I_{LOAD} = 0mA (USB transceiver enabled)			500	µA
	Supply current in HALT mode	External I_{LOAD} = 0mA (USB transceiver disabled)		50	100	

Notes:

1. CPU running with memory access, all I/O pins in input mode with a static value at V_{DD} or V_{SS}; clock input (OSCIN) driven by external square wave.
2. All I/O pins in input mode with a static value at V_{DD} or V_{SS}; clock input (OSCIN) driven by external square wave.

T = 0... +70°C, voltages are referred to V_{SS} unless otherwise specified:

I/O PORT PINS						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{IL}	Input low level voltage	V_{DD}=5V			0.3xV_{DD}	V
V_{IH}	Input high level voltage	V_{DD}=5V	0.7xV_{DD}			
V_{HYS}	Schmidt trigger voltage hysteresis [1]			400		mV
V_{OL}	Output low level voltage for Standard I/O port pins	I=-5mA			1.3	V
		I=-2mA			0.4	
V_{OH}	Output high level voltage	I=3mA	V_{DD}-0.8			
I_L	Input leakage current	V_{SS}<V_{PIN}<V_{DD}			1	µA
R_{PU}	Pull-up equivalent resistor		50	90	170	KΩ
t_{OHL}	Output high to low level fall time for high sink I/O port pins (Port D) [2]	C_l=50pF	6	8	13	ns
t_{OHL}	Output high to low level fall time for standard I/O port pins (Port A, B or C) [2]		18		23	
t_{OLH}	Output L-H rise time (Port D) [2]		7	9	14	
t_{OLH}	Output L-H rise time for standard I/O port pins (Port A, B or C) [2]		19		28	
t_{ITEXT}	External interrupt pulse time		1			t_{CPU}

Note 1: Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.
Note 2: Guaranteed by design, not tested in production.

LED PINS						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
I_{Lsink}	Low current	Vpad > V_{DD}-2.4	2		4	mA
I_{Lsink}	High current	Vpad > V_{DD}-2.4 for ROM device	5	6 [1]	8.4	
		Vpad > V_{DD}-2.4 for FLASH device	5	7 [1]	8.4	

Note 1: Data based on characterization, not tested in production

14.3 SUPPLY AND RESET CHARACTERISTICS

(T = 0 to +70°C, V_{DD} - V_{SS} = 5.5V unless otherwise specified)

LOW VOLTAGE DETECTOR AND SUPERVISOR (LVDS)						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{IT+}	Reset release threshold (V_{DD} rising)			3.7	3.9	V
V_{IT-}	Reset generation threshold (V_{DD} falling)		3.3	3.5		V
V_{hys}	Hysteresis V_{IT+} - V_{IT-} [1]			200		mV
V_{tPOR}	V_{DD} rise time rate [1]		20			ms/V

Note 1 : Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.

14.4 CLOCK AND TIMING CHARACTERISTICS

14.4.1 General Timings

(Operating conditions T_A = 0 to +70°C unless otherwise specified)

Symbol	**Parameter**	**Conditions**	**Min**	**Typ** [1]	**Max**	**Unit**
$t_{c(INST)}$	Instruction cycle time		2	3	12	t_{CPU}
		f_{CPU}=4MHz	500	750	3000	ns
$t_{v(IT)}$	Interrupt reaction time [2] $t_{v(IT)} = \Delta t_{c(INST)} + 10$		10		22	t_{CPU}
		f_{CPU}=4MHz	2.5		5.5	µs

* Δt_{INST} is the number of t_{CPU} to finish the current instruction execution.

14.4.2 External Clock Source

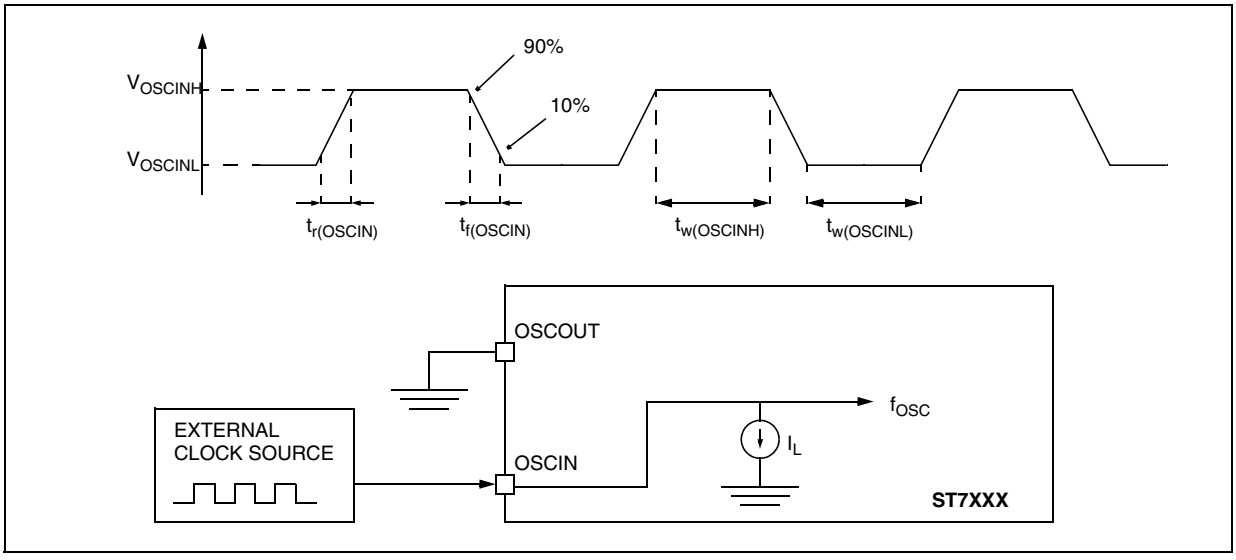
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
V_{OSCINH}	OSCIN input pin high level voltage	see Figure 39	0.7xV_{DD}		V_{DD}	V
V_{OSCINL}	OSCIN input pin low level voltage		V_{SS}		0.3xV_{DD}	
$t_{w(OSCINH)}$ $t_{w(OSCINL)}$	OSCIN high or low time [3]		15			ns
$t_{r(OSCIN)}$ $t_{f(OSCIN)}$	OSCIN rise or fall time [3]				15	
I_L	OSCx Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA

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)

Figure 39. Typical Application with an External Clock Source



CLOCK AND TIMING CHARACTERISTICS (Cont'd)

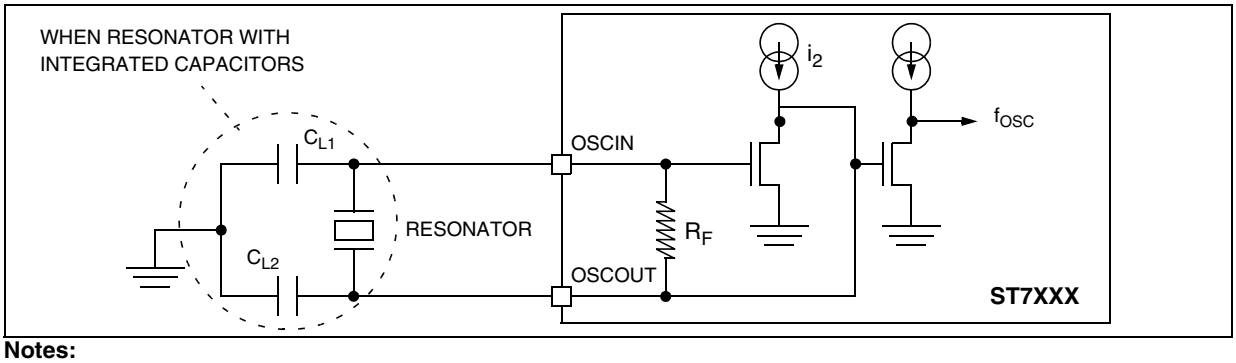
14.4.3 Crystal Resonator Oscillators

The ST7 internal clock is supplied with one Crystal resonator oscillator. All the information given in this paragraph are based on characterization results with specified typical external componants. 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 resonator manufacturer for more details (frequency, package, accuracy...).

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	Oscillator Frequency [3]	MP: Medium power oscillator		4		MHz
R_F	Feedback resistor		90		150	kΩ
C_{L1} C_{L2}	Recommanded load capacitances versus equivalent serial resistance of the crystal resonator (R_S)	See Ταβλε 5, ®Ρεχομμενδεδ ςαλυεσ φορ 4 ΜΗζ Χρψσταλ Ρεσονατορ,© ον παγε 21 (MP oscillator)	22		56	pF
i_2	OSCOUT driving current	V_{DD}=5V V_{IN}=V_{SS} (MP oscillator)	1.5		3.5	mA

Oscil.		Typical Crystal Resonator				C_{L1} [pF]	C_{L2} [pF]	$t_{SU(osc)}$ [ms] [2]
		Reference		Freq.	Characteristic [1]			
Crystal	MP	JAUCH	SS3-400-30-30/30	4MHz	Δf_{OSC}=[±30ppm$_{25°C}$,±30ppm$_{\Delta Ta}$], Typ. R_S=60Ω	33	33	7~10

Figure 40. Typical Application with a Crystal Resonator



Notes:

1. Resonator characteristics given by the crystal resonator manufacturer.

2. $t_{SU(OSC)}$ is the typical oscillator start-up time measured between V_{DD}=2.8V and the fetch of the first instruction (with a quick V_{DD} ramp-up from 0 to 5V (<50µs).

3. The oscillator selection can be optimized in terms of supply current using an high quality resonator with small R_S value. Refer to crystal resonator manufacturer for more details.

14.5 MEMORY CHARACTERISTICS

Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

14.5.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode [1]	HALT mode (or RESET)	2			V

Note 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.

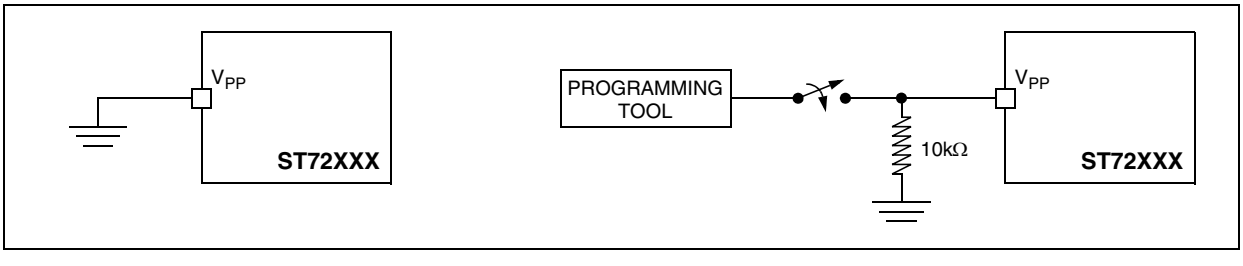
14.5.2 FLASH Memory

Operating Conditions: f_{CPU} = 8 MHz.

DUAL VOLTAGE FLASH MEMORY						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
f_{CPU}	Operating Frequency	Read mode			8	MHz
		Write / Erase mode, T_A=25°C			8	
V_{PP}	Programming Voltage	4.0V ≤V_{DD} ≤ 5.5V	11.4		12.6	V
I_{PP}	V_{PP} Current	Write / Erase			30	mA
t_{PROG}	Byte Programming Time	T_A=25°C		330		µs
	Block Programming Time (16KB)			0.8		s
t_{ERASE}	Sector Erasing Time (sector 0 ; 4KB)			2		
	Sector Erasing Time (sector 1 ; 4KB)			2		
	Sector Erasing Time (sector 2 ; 8KB)			2.5		
t_{VPP}	Internal V_{PP} Stabilization Time			10		µs
t_{RET}	Data Retention	T_A ≤ 55°C	20			years
N_{RW}	Write Erase Cycles	T_A=25°C	100			cycles

Warning: Do not connect 12V to V_{PP} before V_{DD} is powered on, as this may damage the device.

Figure 41. Two typical Applications with V_{PP} Pin [1]



14.6 SMARTCARD SUPPLY SUPERVISOR ELECTRICAL CHARACTERISTICS

(T_A = 0... +70°C, 4.0 < V_{DD} - V_{SS} < 5.5V unless otherwise specified)

SMARTCARD SUPPLY SUPERVISOR						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
5V regulator output (for IEC7816-3 Class A Cards)						
V_{CRDVCC}	SmartCard Power Supply Voltage		4.6	5.00	5.4	V

SMARTCARD SUPPLY SUPERVISOR						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
I_{SC}	SmartCard Supply Current				55	mA
I_{OVDET}	Current Overload Detection				120 [1]	mA
t_{IDET}	Detection time on Current Overload		170 [1]		1400 [1]	µs
t_{OFF}	V_{CRDVCC} Turn off Time (see Figure 36)	$C_{LOADmax} \leq 4.7uF$			750	µs
t_{ON}	V_{CRDVCC}Turn on Time (see Figure 36)	$C_{LOADmax} \leq 4.7uF$		150	500	µs
V_{CRDVCC}	V_{CARD} above minimum supply voltage		4.52 [1]		4.76 [1]	V
I_{VDD}	V_{DD} supply current	(See note 3)			100	mA
3V regulator output (for IEC7816-3 Class B Cards)						
V_{CRDVCC}	SmartCard Power Supply Voltage		2.7		3.3	V
I_{SC}	SmartCard Supply Current				50	mA
I_{OVDET}	Current Overload Detection				100 [1]	mA
t_{IDET}	Detection time on Current Overload		170 [1]		1400 [1]	us
t_{OFF}	V_{CRDVCC} Turn off Time (see Figure 36)	$C_{LOADmax} \leq 4.7uF$			750	us
t_{ON}	V_{CRDVCC} Turn on Time (see Figure 36)	$C_{LOADmax} \leq 4.7uF$		150	500	µs
1.8V regulator output (for IEC7816-3 Class C Cards)						
V_{CRDVCC}	SmartCard Power Supply Voltage		1.65		1.95	V
I_{SC}	SmartCard Supply Current				20	mA
I_{OVDET}	Current Overload Detection				100 [1]	mA
t_{IDET}	Detection time on Current Overload		170 [1]		1400 [1]	us
t_{OFF}	V_{CRDVCC} Turn off Time (see Figure 36)	$C_{LOADmax} \leq 4.7uF$			750	us
t_{ON}	V_{CRDVCC} Turn on Time (see Figure 36)	$C_{LOADmax} \leq 4.7uF$		150	500	µs
Smartcard CLKPin						
V_{OL}	Output Low Level Voltage	I=-50uA	-	-	0.4 [2]	V
V_{OH}	Output High Level Voltage	I=50uA	V_{CRDVCC}-0.5 [2]	-	-	V
T_{OHL}	Output H-L Fall Time [1]	C_l=30pF	-		20	ns
T_{OLH}	Output L-H Rise Time [1]	C_l=30pF	-		20	ns
F_{VAR}	Frequency variation [1]		-		1	%
F_{DUTY}	Duty cycle [1]		45		55	%
P_{OL}	Signal low perturbation [1]		-0.25		0.4	V
P_{OH}	Signal high perturbation [1]		V_{CRDVCC}-0.5		V_{CRDVCC}+0.25	V
I_{SGND}	Short-circuit to Ground [1]			15		mA
Smartcard I/O Pin						
V_{IL}	Input Low Level Voltage		-	-	0.5 [2]	V
V_{IH}	Input High Level Voltage		0.6V_{CRDVCC} [2]	-	-	V
V_{OL}	Output Low Level Voltage	I=-0.5mA	-	-	0.4 [2]	V
V_{OH}	Output High Level Voltage	I=20uA	0.8V_{CRDVCC} [2]	-	V_{CRDVCC} [2]	V
I_L	Input Leakage Current [1]	$V_{SS}<V_{IN}<V_{SC\_PWR}$	-10	-	10	µA
I_{RPU}	Pull-up Equivalent Resistance	$V_{IN}=V_{SS}$		24	30	KΩ
T_{OHL}	Output H-L Fall Time [1]	C_l=30pF	-		0.8	us
T_{OLH}	Output L-H Rise Time [1]	C_l=30pF	-		0.8	us
I_{SGND}	Short-circuit to Ground [1]			15		mA

SMARTCARD SUPPLY SUPERVISOR						
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Smartcard RST C4 and C8 Pin						
V_{OL}	Output Low Level Voltage	I=-0.5mA	-	-	0.4 [2]	V
V_{OH}	Output High Level Voltage	I=20uA	V_{CRDVCC}-0.5 [2]	-	V_{CRDVCC} [2]	V
T_{OHL}	Output H-L Fall Time [1]	C_l=30pF	-		0.8	us
T_{OLH}	Output L-H Rise Time [1]	C_l=30pF	-		0.8	us
I_{SGND}	Short-circuit to Ground [1]			15		mA

Note 1 : Guaranteed by design.
Note 2 : Data based on characterization results, not tested in production.
Note 3: V_{DD} = 4.75 V, Card consumption = 55mA, CRDCLK frequency = 4MHz, LED with a 3mA current, USB in reception mode and CPU in WFI mode.

14.7 EMC CHARACTERISTICS

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

14.7.1 Functional EMS (Electro Magnetic Susceptibility)

Based on a simple running application on the product (toggling 2 LEDs through I/O ports), the product is stressed by two electro magnetic events until a failure occurs (indicated by the LEDs).

- **ESD**: Electro-Static Discharge (positive and negative) is applied on all pins of the device until a functional disturbance occurs. This test conforms with the IEC 1000-4-2 standard.
- **FTB**: A Burst of Fast Transient voltage (positive and negative) is applied to V_{DD} and V_{SS} through a 100pF capacitor, until a functional disturbance occurs. This test conforms with the IEC 1000-4-4 standard.

A device reset allows normal operations to be resumed. The test results are given in the table below based on the EMS levels and classes defined in application note AN1709.

14.7.1.1 Designing hardened software to avoid noise problems

EMC characterization and optimization are performed at component level with a typical application environment and simplified MCU software. It should be noted that good EMC performance is highly dependent on the user application and the software in particular.

Therefore it is recommended that the user applies EMC software optimization and prequalification tests in relation with the EMC level requested for his application.

Software recommendations:

The software flowchart must include the management of runaway conditions such as:

– Corrupted program counter

– Unexpected reset

– Critical Data corruption (control registers...)

Prequalification trials:

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

To complete these trials, ESD stress can be applied directly on the device, over the range of specification values. When unexpected behaviour is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Symbol	Parameter	Conditions	Level/ Class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz conforms to IEC 1000-4-2	2B
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°C, f_{OSC}=8MHz conforms to IEC 1000-4-4	4B

14.7.2 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
				4/8MHz	4/4MHz	
S_{EMI}	Peak level	V_{DD}=5V, T_A=+25°C, conforming to SAE J 1752/3	0.1MHz to 30MHz	19	18	dBµV
			30MHz to 130MHz	32	27	
			130MHz to 1GHz	31	26	
			SAE EMI Level	4	3.5	-

Notes:

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

EMC CHARACTERISTICS (Cont'd)

14.7.3 Absolute Maximum Ratings (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 application note AN1181.

14.7.3.1 Electro-Static Discharge (ESD)

Electro-Static Discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends on the number of supply pins in the device (3 parts*(n+1) supply pin). The Human Body Model is simulated. This test conforms to the JESD22-A114A standard.

Absolute Maximum Ratings

Symbol	Ratings	Conditions	Maximum value [1]	Unit
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)	T_A=+25°C	2000	V

Notes:

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

14.7.3.2 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 application note AN1181.
- **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. For more details, refer to the application note AN1181.

Electrical Sensitivities

Symbol	Parameter	Conditions	Class [1]
LU	Static latch-up class	T_A=+25°C	A
DLU	Dynamic latch-up class	V_{DD}=5.5V, f_{OSC}=4MHz, T_A=+25°C	A

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).

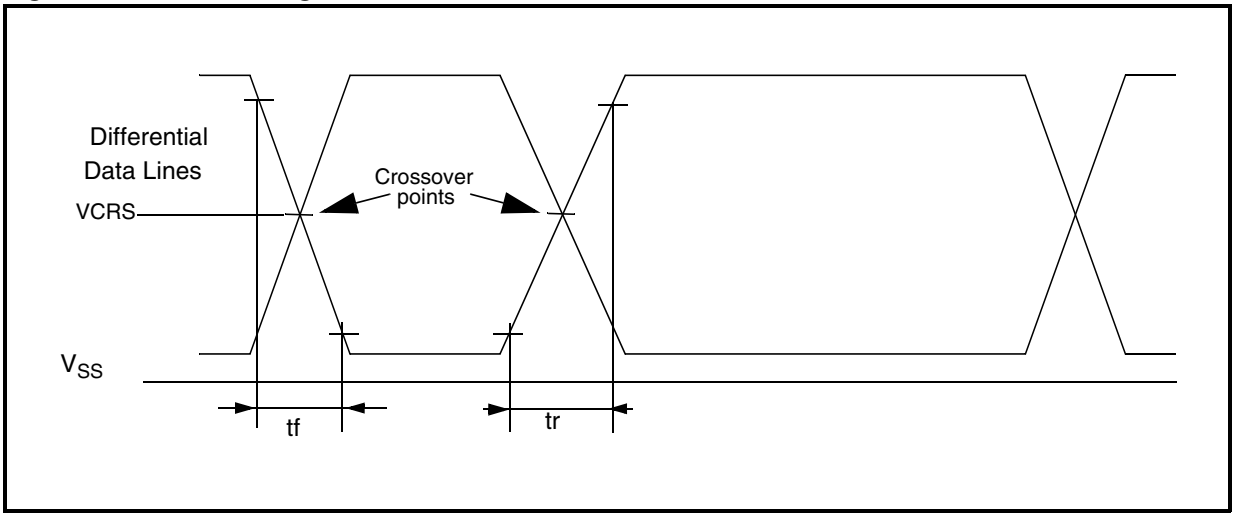
14.8 COMMUNICATION INTERFACE CHARACTERISTICS

14.8.1 USB - Universal Bus Interface

USB DC Electrical Characteristics					
Parameter	**Symbol**	**Conditions**	**Min.**	**Max.**	**Unit**
Input Levels:					
Differential Input Sensitivity	VDI	I(D+, D-)	0.2		V
Differential Common Mode Range	VCM	Includes VDI range	0.8	2.5	V
Single Ended Receiver Threshold	VSE		1.3	2.0	V
Output Levels					
Static Output Low	VOL	RL of 1.5K ohms to 3.6v		0.3	V
Static Output High	VOH	RL of 15K ohm to V_{SS}	2.8	3.6	V
USBVCC: voltage level	USBV	V_{DD}=5v	3.00	3.60	V

Notes: RL is the load connected on the USB drivers. All the voltages are measured from the local ground potential.

Figure 42. USB: Data Signal Rise and Fall Time



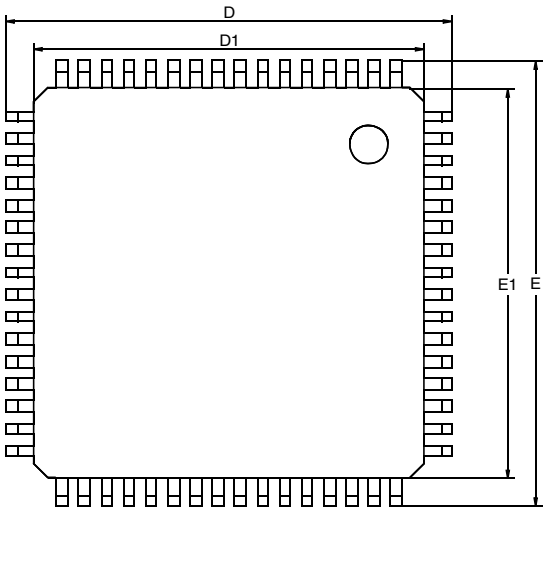
USB: Full speed electrical characteristics					
Parameter	**Symbol**	**Conditions**	**Min**	**Max**	**Unit**
Driver characteristics:					
Rise time	tr	Note 1,CL=50 pF	4	20	ns
Fall Time	tf	Note 1, CL=50 pF	4	20	ns
Rise/ Fall Time matching	trfm	tr/tf	90	110	%
Output signal Crossover Voltage	VCRS		1.3	2.0	V

Note 1: Measured from 10% to 90% of the data signal. For more detailed informations, please refer to Chapter 7 (Electrical) of the USB specification (version 1.1).

15 PACKAGE CHARACTERISTICS

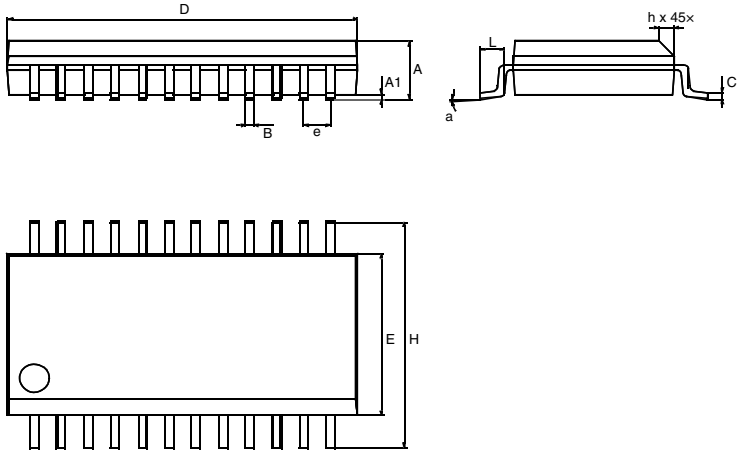
15.1 PACKAGE MECHANICAL DATA

Figure 43. 64-Pin Thin Quad Flat Package (14x14)



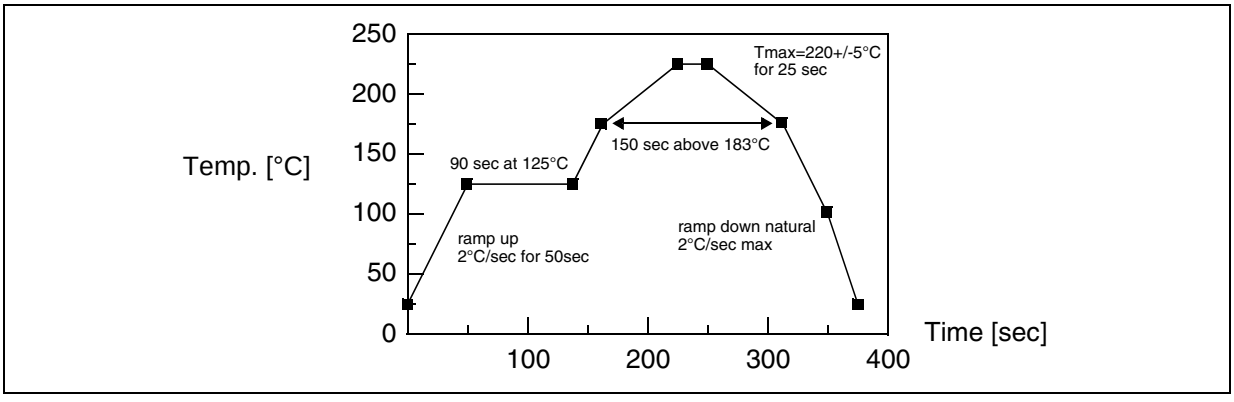
Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.063
A1	0.05		0.15	0.002		0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.30	0.37	0.45	0.012	0.015	0.018
c	0.09		0.20	0.004		0.008
D		16.00			0.630	
D1		14.00			0.551	
E		16.00			0.630	
E1		14.00			0.551	
e		0.80			0.031	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
	Number of Pins					
N	64					

Figure 44. 24-Pin Plastic Small Outline Package, 300-mil Width



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	2.35		2.65	0.093		0.104
A1	0.10		0.30	0.004		0.012
B	0.33		0.51	0.013		0.020
C	0.23		0.32	0.009		0.013
D	15.20		15.60	0.599		0.614
E	7.40		7.60	0.291		0.299
e		1.27			0.050	
H	10.00		10.65	0.394		0.419
h	0.25		0.75	0.010		0.030
α	0°		8°	0°		8°
L	0.40		1.27	0.016		0.050
	Number of Pins					
N	24					

Figure 45. PACKAGE MECHANICAL DATA (Cont'd)
Figure 46. Recommended Reflow Oven Profile (MID JEDEC)



16 DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in user programmable versions (High Density FLASH) as well as in factory coded versions (ROM/FASTROM).

ST7SCR devices are ROM versions. ST7PSCR devices are Factory Advanced Service Technique ROM (FASTROM) versions: they are factory programmed FLASH devices.

ST7FSCR FLASH devices are shipped to customers with a default content (FFh).

This implies that FLASH devices have to be configured by the customer using the Option Byte while the ROM devices are factory-configured.

16.0.1 Option Bytes

The 8 option bits from the flash are programmed through the static option byte SOB1. The description of each of these 8 bits is given below.

Static option Byte (SOB1)

OPT 7	6	5	4	3	2	1	OPT 0
--	--	WDG-SW	NEST	ISOCLK	RETRY	-	FMP_R

OPT7:6 = Reserved

OPT5= **WDGSW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware (watchdog always activated)
1: Software (watchdog to be activated by software)

OPT4 = **NEST** *Interrupt Controller*
This bit enables the nested Interrupt Controller.
0: Nested interrupt controller disabled
1: Nested interrupt controller enabled

OPT3 = **ISOCLK** *Clock source selection*
0: Card clock is generated by the divider (48MHz/12 = 4MHz).
1: Card clock is generated by the oscillator.

OPT2 = **RETRY** *Number of Retries for UART ISO*
0: In case of an erroneous transfer, character is transmitted 4 times.
1: In case of an erroneous transfer, character is transmitted 5 times.

OPT1 = Reserved, must be kept at 1.

OPT0 = **FMP_R** *Flash memory read-out protection*

Readout protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. This protection is based on read and a write protection of the memory in test modes and ICP mode. Erasing the option bytes when the FMP_R option is selected induce the whole user memory erasing first and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual and section 4.6 on page 17 for more details
.
0 : read-out protection enabled
1 : read-out protection disabled

16.1 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 hexadecimal file in .S19 format generated by the development tool. All unused bytes must be set to FFh.

The selected options are communicated to STMicroelectronics using the correctly completed OPTION LIST appended. See page 94.

Refer to application note AN1635 for information on the counter listing returned by ST after code has been transferred.

The STMicroelectronics Sales Organization will be pleased to provide detailed information on contractual points.

Figure 47. Sales Type Coding Rules

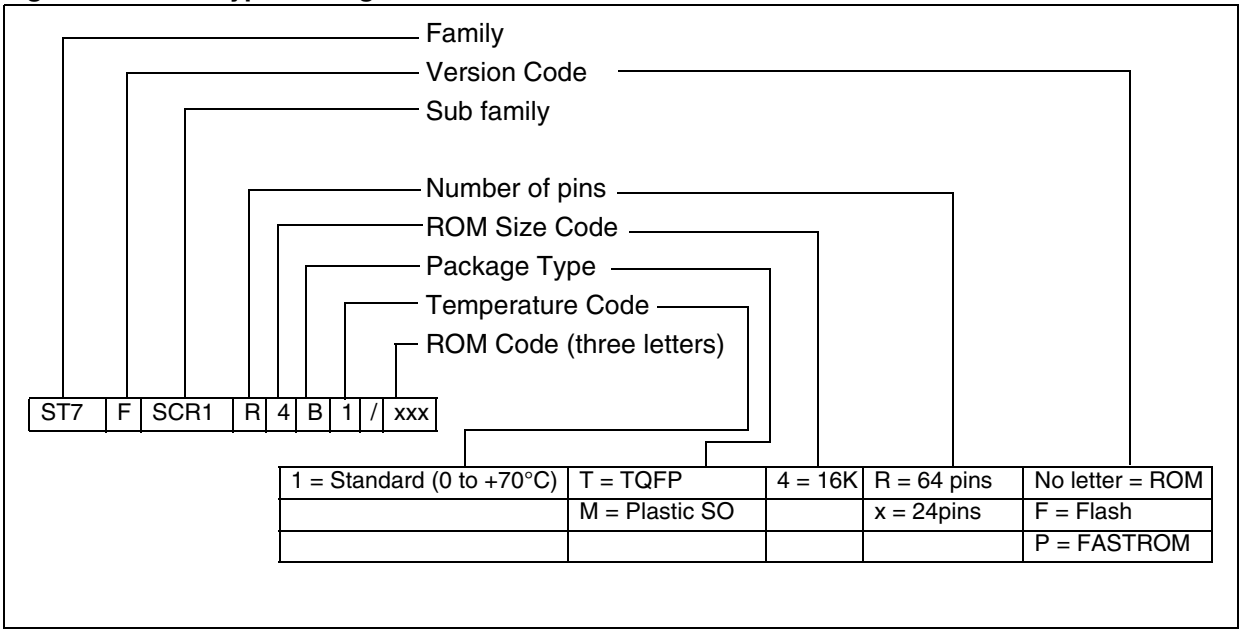


Table 24. Ordering Information

Sales Type [1]	Program Memory (bytes)	RAM (bytes)	Package
ST7SCR1R4T1/xxx	16K ROM	768	TQFP64
ST7PSCR1R4T1/xxx	16K FASTROM		
ST7FSCR1R4T1	16K Flash		
ST7SCR1E4M1/xxx	16K ROM		SO24
ST7PSCR1E4M1/xxx	16K FASTROM		
ST7FSCR1E4M1	16K Flash		

Note 1. /xxx stands for the ROM or FASTROM-code name assigned by STMicroelectronics.

ST7SCR 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:	16K
SO24:	[ ] ST7SCR1E4M1
TQFP64:	[ ] ST7SCR1R4T1
FASTROM Device:	16K
SO24:	[ ] ST7PSCR1E4M1
TQFP64:	[ ] ST7PSCR1R4T1

Conditioning (check only one option):

Packaged Product:		Die Product (dice tested at 25°C only)
[ ] Tape & Reel	[ ] Tray (TQFP package only)	[ ] Tape & Reel
	[ ] Tube (SO package only)	[ ] Inked wafer
		[ ] Sawn wafer on sticky foil

Note: Die product only for ROM device
Special Marking: [ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ "
Authorized characters are letters, digits, '.', '-', '/' and spaces only.
Maximum character count:
S024 (13 char. max) : _ _ _ _ _ _ _ _ TQFP64 (10 char. max) : _ _ _ _ _ _ _ _ _ _

Watchdog:	WDGSW	[ ] Software Activation [ ] Hardware Activation
Nested Interrupts	NEST	[ ] Nested Interrupts [ ] Non Nested Interrupts
ISO Clock Source	ISOCLK	[ ] Oscillator [ ] Divider
No. of Retries	RETRY	[ ] 5 [ ] 4
Readout Protection:	FMP_R	[ ] Disabled [ ] Enabled

Signature

Date

16.2 DEVELOPMENT TOOLS

Table 25. Development Tools

Development Tool	Sales Type	Remarks
Emulator	ST7MDTS1-EMU2B	
Programming Board	ST7MDTS1-EPB2	

16.2.1 ADAPTOR/SOCKET PROPOSAL

TBD

16.3 ST7 APPLICATION NOTES

IDENTIFICATION	DESCRIPTION
APPLICATION EXAMPLES	
AN1658	SERIAL NUMBERING IMPLEMENTATION
AN1720	MANAGING THE READ-OUT PROTECTION IN FLASH MICROCONTROLLERS
AN1755	A HIGH RESOLUTION/PRECISION THERMOMETER USING ST7 AND NE555
AN1756	CHOOSING A DALI IMPLEMENTATION STRATEGY WITH ST7DALI
EXAMPLE DRIVERS	
AN 969	SCI COMMUNICATION BETWEEN ST7 AND PC
AN 970	SPI COMMUNICATION BETWEEN ST7 AND EEPROM
AN 971	I²C COMMUNICATION 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 (SINUSOÏD)
AN1042	ST7 ROUTINE FOR I²C SLAVE MODE MANAGEMENT
AN1044	MULTIPLE INTERRUPT SOURCES MANAGEMENT FOR ST7 MCUS
AN1045	ST7 S/W IMPLEMENTATION OF I²C BUS MASTER
AN1046	UART EMULATION SOFTWARE
AN1047	MANAGING RECEPTION ERRORS WITH THE ST7 SCI PERIPHERALS
AN1048	ST7 SOFTWARE LCD DRIVER
AN1078	PWM DUTY CYCLE SWITCH IMPLEMENTING TRUE 0% & 100% DUTY CYCLE
AN1082	DESCRIPTION OF THE ST72141 MOTOR CONTROL PERIPHERALS REGISTERS
AN1083	ST72141 BLDC MOTOR CONTROL SOFTWARE AND FLOWCHART EXAMPLE
AN1105	ST7 PCAN PERIPHERAL DRIVER
AN1129	PWM MANAGEMENT FOR BLDC MOTOR DRIVES USING THE ST72141
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	EMULATED 16 BIT SLAVE SPI
AN1475	DEVELOPING AN ST7265X MASS STORAGE APPLICATION
AN1504	STARTING A PWM SIGNAL DIRECTLY AT HIGH LEVEL USING THE ST7 16-BIT TIMER
AN1602	16-BIT TIMING OPERATIONS USING ST7262 OR ST7263B ST7 USB MCUS
AN1633	DEVICE FIRMWARE UPGRADE (DFU) IMPLEMENTATION IN ST7 NON-USB APPLICATIONS
AN1712	GENERATING A HIGH RESOLUTION SINEWAVE USING ST7 PWMART
AN1713	SMBUS SLAVE DRIVER FOR ST7 I2C PERIPHERALS
AN1753	SOFTWARE UART USING 12-BIT ART
GENERAL PURPOSE	
AN1476	LOW COST POWER SUPPLY FOR HOME APPLIANCES
AN1526	ST7FLITE0 QUICK REFERENCE NOTE

IDENTIFICATION	DESCRIPTION
AN1709	EMC DESIGN FOR ST MICROCONTROLLERS
AN1752	ST72324 QUICK REFERENCE NOTE
PRODUCT EVALUATION	
AN 910	PERFORMANCE BENCHMARKING
AN 990	ST7 BENEFITS VERSUS INDUSTRY STANDARD
AN1077	OVERVIEW OF ENHANCED CAN CONTROLLERS FOR ST7 AND ST9 MCUS
AN1086	U435 CAN-DO SOLUTIONS FOR CAR MULTIPLEXING
AN1103	IMPROVED B-EMF DETECTION FOR LOW SPEED, LOW VOLTAGE WITH ST72141
AN1150	BENCHMARK ST72 VS PC16
AN1151	PERFORMANCE COMPARISON BETWEEN ST72254 & PC16F876
AN1278	LIN (LOCAL INTERCONNECT NETWORK) SOLUTIONS
PRODUCT MIGRATION	
AN1131	MIGRATING APPLICATIONS FROM ST72511/311/214/124 TO ST72521/321/324
AN1322	MIGRATING AN APPLICATION FROM ST7263 REV.B TO ST7263B
AN1365	GUIDELINES FOR MIGRATING ST72C254 APPLICATIONS TO ST72F264
AN1604	HOW TO USE ST7MDT1-TRAIN WITH ST72F264
PRODUCT OPTIMIZATION	
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
AN1181	ELECTROSTATIC DISCHARGE SENSITIVE MEASUREMENT
AN1324	CALIBRATING THE RC OSCILLATOR OF THE ST7FLITE0 MCU USING THE MAINS
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
AN1605	USING AN ACTIVE RC TO WAKEUP THE ST7LITE0 FROM POWER SAVING MODE
AN1636	UNDERSTANDING AND MINIMIZING ADC CONVERSION ERRORS
AN1828	PIR (PASSIVE INFRARED) DETECTOR USING THE ST7FLITE05/09/SUPERLITE
PROGRAMMING AND TOOLS	
AN 978	ST7 VISUAL DEVELOP SOFTWARE KEY DEBUGGING FEATURES
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
AN1477	EMULATED DATA EEPROM WITH XFLASH MEMORY
AN1478	PORTING AN ST7 PANTA PROJECT TO CODEWARRIOR IDE

IDENTIFICATION	DESCRIPTION
AN1527	DEVELOPING A USB SMARTCARD READER WITH ST7SCR
AN1575	ON-BOARD PROGRAMMING METHODS FOR XFLASH AND HDFLASH ST7 MCUS
AN1576	IN-APPLICATION PROGRAMMING (IAP) DRIVERS FOR ST7 HDFLASH OR XFLASH MCUS
AN1577	DEVICE FIRMWARE UPGRADE (DFU) IMPLEMENTATION FOR ST7 USB APPLICATIONS
AN1601	SOFTWARE IMPLEMENTATION FOR ST7DALI-EVAL
AN1603	USING THE ST7 USB DEVICE FIRMWARE UPGRADE DEVELOPMENT KIT (DFU-DK)
AN1635	ST7 CUSTOMER ROM CODE RELEASE INFORMATION
AN1754	DATA LOGGING PROGRAM FOR TESTING ST7 APPLICATIONS VIA ICC
AN1796	FIELD UPDATES FOR FLASH BASED ST7 APPLICATIONS USING A PC COMM PORT
SYSTEM OPTIMIZATION	
AN1827	IMPLEMENTATION OF SIGMA-DELTA ADC WITH ST7FLITE05/09

17 IMPORTANT NOTES

17.1 UNEXPECTED RESET FETCH

If an interrupt request occurs while a "POP CC" instruction is executed, the interrupt controller does not recognise the source of the interrupt and, by default, passes the RESET vector address to the CPU.

Workaround

To solve this issue, a "POP CC" instruction must always be preceded by a "SIM" instruction.

18 REVISION HISTORY

Table 26. Revision History.

Date	Revision	Description of Changes
11-Mar-04	1.5	Changed labelling of Capacitors on Figure 4 & removed 3. Inserted note that C_1 and C_2 must be close to the chip on Figure 4 Changed C_{L2} from 30pF to 33pF, section 14.4.3 on page 83 Added Figure 5.Smartcard Interface Reference Application - 64-Pin TQFP Package Changed ILsink Min from 5.6mA to 5, p80, LED Pins Table Changed values in 14.5.2FLASH Memory Table For table in 14.6SMARTCARD SUPPLY SUPERVISOR ELECTRICAL CHARACTERISTICS added many references to note 1 Section 14.7.3Absolute Maximum Ratings (Electrical Sensitivity) Changed VESD Max fro, 1500 to 2000 V Section 14.8.1USB - Universal Bus Interface table, merged notes 1 & 2 into one note. Replaced Errata sheet with Important Notes section (17IMPORTANT NOTES) Figure 14.4.3 Values changed: R_f : Min 90kΩ, Max 150kΩ; I_2 : Min 1.5mA, Max 3.5mA
15-Sep-04	2.0	Split High Current values in LED Pins table for ROM and FLASH devices Section 14.2 Clarification of read-out protection

ST7SCR LIMITATIONS AND CORRECTIONS

1 SILICON IDENTIFICATION

This document refers only to ST7FSCR devices shown in Table 1. They are identifiable both by the last letter of the **Trace code** marked on the device package and by the last 3 digits of the **Internal Sales Type** printed on the box label (see also Figure 1)

Table 1. Device Identification

	Trace Code marked on device	Internal Sales Type on box label
Flash Devices:	“xxxxxxxxxW”	7FSCR1R4T1$U6 7FSCR1E4M1$U6

2 REFERENCE SPECIFICATION

ST7SCR Datasheet 2.0 (September 2004).

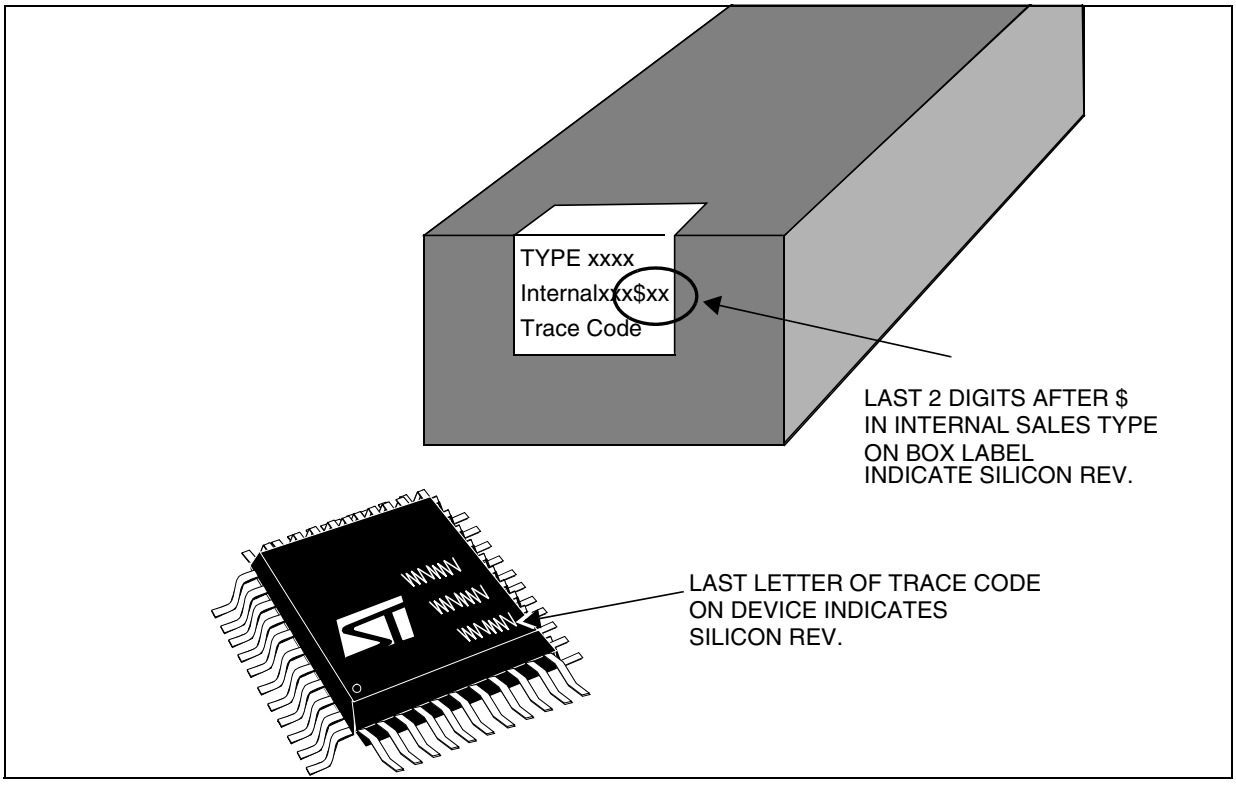
3 SILICON LIMITATIONS

3.1 SMART CARD UART AUTOMATIC REPETITION AND RETRY

A functional limitation affects the Smart Card UART automatic repetition and retry on parity error in reception and transmission mode. This failure occurrence is systematic: only 4 retries option is functional.

4 DEVICE MARKING

Figure 1. Revision Marking on Box Label and Device Marking



5 ERRATA SHEET REVISION HISTORY

Date	Revision	Description of Changes
1 July 04	1.3	New limitation for Revision W, Section 3.1 Smart Card UART Automatic Repetition and Retry Internal Sales Type codes changed, Table 1

Notes: