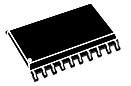
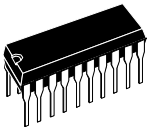
ST7261

LOW SPEED USB 8-BIT MCU WITH 3 ENDPOINTS, ROM MEMORY, LVD, WDG, TIMER

- **Memories**
  - 4K Program memory (ROM) with read-write protection.
    In-Circuit programming for Flash versions
  - 256 bytes RAM memory (128-byte stack)
- **Clock, Reset and Supply Management**
  - Enhanced Reset System (Power On Reset)
  - Low Voltage Detector (LVD)
  - Clock-out capability
  - 6 or 12 MHz Oscillator (8, 4, 2, 1 MHz internal freq.)
  - 3 Power saving modes: Halt, Wait and Slow
- **USB (Universal Serial Bus) Interface**
  - DMA for low speed applications compliant with USB specification (v 2.0):
  - Integrated 3.3V voltage regulator and transceivers
  - Suspend and Resume operations
  - 3 Endpoints
- **11 I/O Ports**
  - 11 multifunctional bidirectional I/O lines
  - Up to 7 External interrupts (2 vectors)
  - 8 high sink outputs (8mA@0.4 V/20mA@1.3)
- **2 Timers**
  - Configurable watchdog timer (8 to 500ms timeout)
  - 8-bit Time Base Unit (TBU) for generating periodic interrupts
- **Instruction Set**
  - 8-bit data manipulation
  - 63 basic instructions
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instruction
  - True bit manipulation
- **Nested interrupts**
- **Development Tools**
  - Full hardware/software development package



SO20



PDIP20

Device Summary

Features	ST72611F1
Program memory - bytes	4K ROM
RAM (stack) - bytes	256 (128)
Peripherals	USB, Watchdog, Low Voltage Detector, Time Base Unit
I/Os	11
Operating Supply	4.0V to 5.5V
CPU Frequency	Up to 8 MHz (with 6 or 12 MHz oscillator)
Operating Temperature	0°C to +70°C
Packages	PDIP20/SO20

Rev. 2.2

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 pay special attention to the Section “IMPORTANT NOTE” on page 71.

1 INTRODUCTION

The ST7261 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 ST7261 devices are ROM versions. The FLASH version is supported by the ST72F623F2.

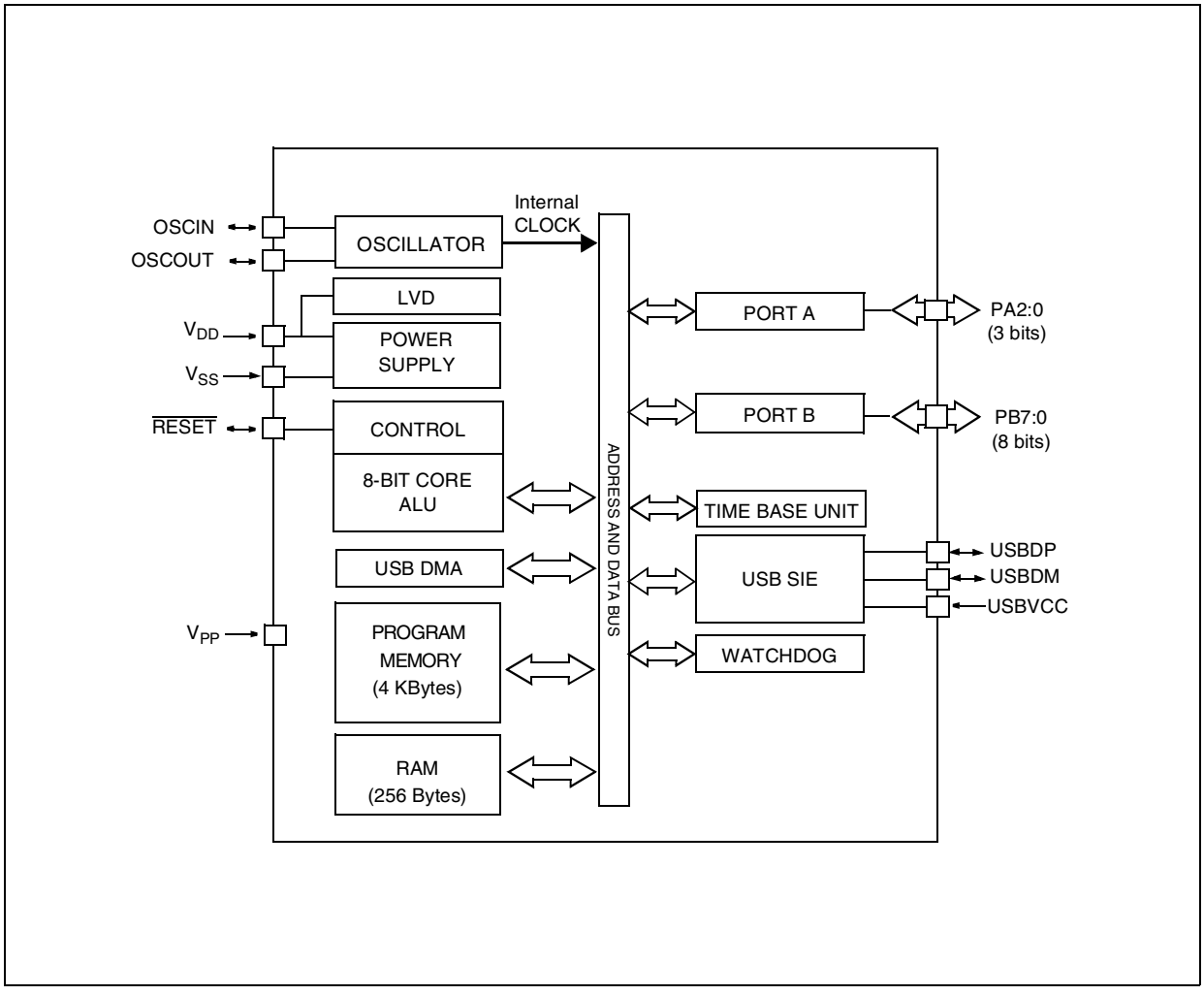
Under software control, all devices can be placed in WAIT, SLOW, or HALT mode, reducing power consumption when the application is in idle or standby state.

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.

Related Documentation

AN1071: Half Duplex USB-to-Serial bridge using the ST72611 USB microcontroller

Figure 1. General Block Diagram



2 PIN DESCRIPTION

Figure 2. 20-pin SO20 Package Pinout

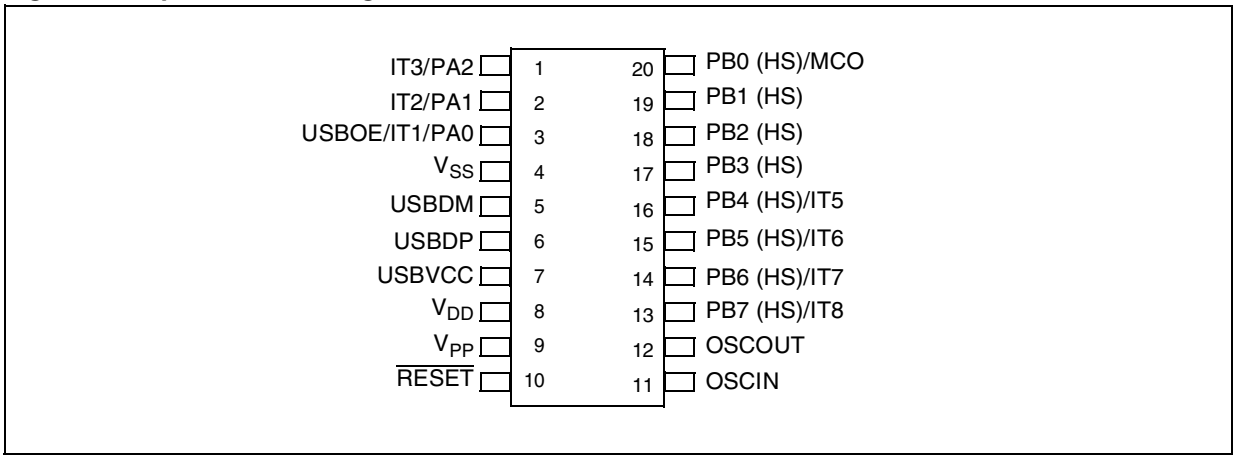
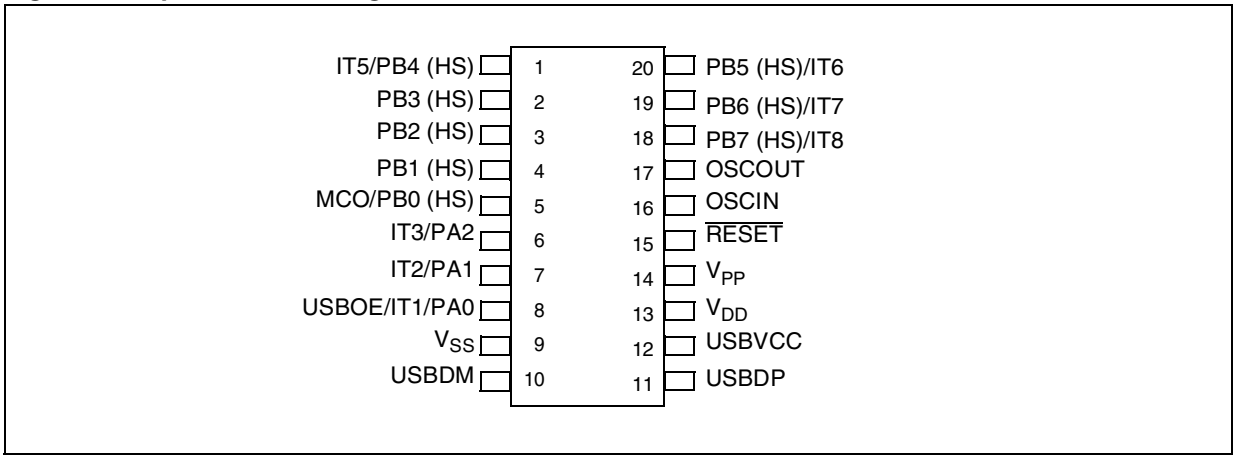


Figure 3. 20-pin DIP20 Package Pinout



PIN DESCRIPTION (Cont'd)

Legend / Abbreviations:

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

Input level: A = Dedicated analog input

Input level: C = CMOS $0.3V_{DD}/0.7V_{DD}$,
C_T= CMOS $0.3V_{DD}/0.7V_{DD}$ with input trigger

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

Port configuration capabilities:

– Input: float = floating, wpu = weak pull-up, int = interrupt (\ =falling edge, / =rising edge), ana = analog

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

Table 1. Device Pin Description

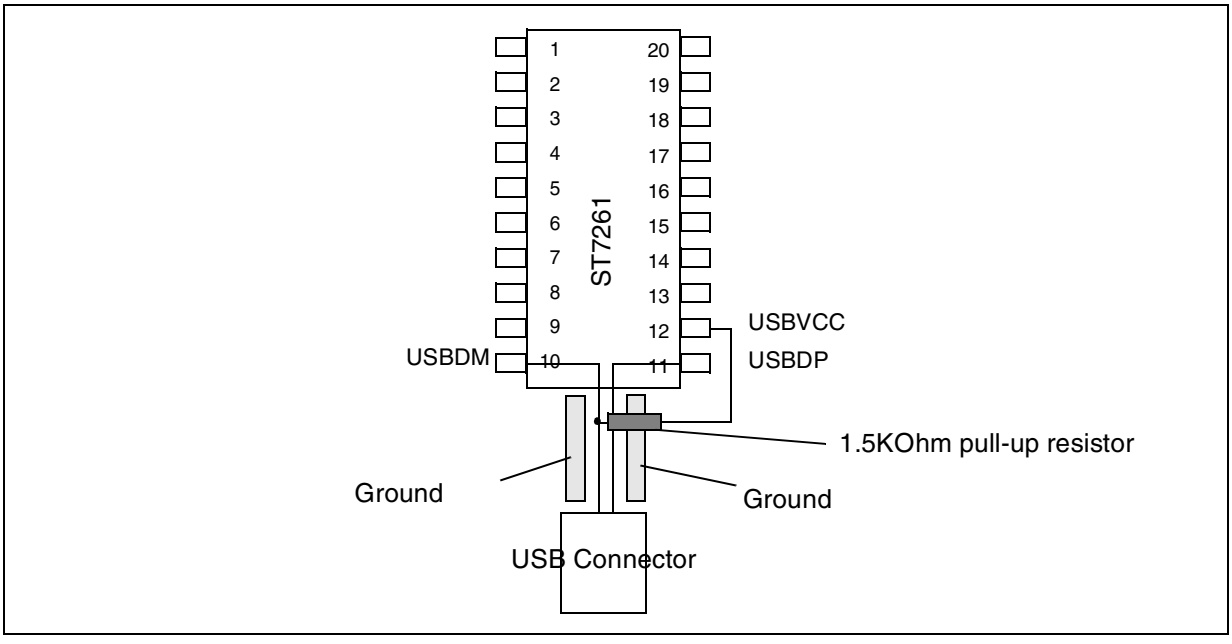
Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Function
						Input				Output			
SO20	DIP20			Input	Output	float	wpu	int	ana	OD	PP		
9	14	V_{PP}	S				x					FLASH programming voltage (12V), must be tied low in user mode.	
11	16	OSCIN										These pins are used connect an external clock source to the on-chip main oscillator.	
12	17	OSCOUT											
4	9	V_{SS}	S									Digital Ground Voltage	
8	13	V_{DD}	S									Digital Main Power Supply Voltage	
13	18	PB7/IT8	I/O	C_T	HS	x		\			x	**Port B7**	Interrupt 8 input
14	19	PB6/IT7	I/O	C_T	HS	x		\			x	**Port B6**	Interrupt 7 input
15	20	PB5/IT6	I/O	C_T	HS	x		/			x	**Port B5**	Interrupt 6 input
16	1	PB4/IT5	I/O	C_T	HS	x		/			x	**Port B4**	Interrupt 5 input
17	2	PB3	I/O	C_T	HS	x					x	**Port B3**	
18	3	PB2	I/O	C_T	HS	x					x	**Port B2**	
19	4	PB1	I/O	C_T	HS	x					x	**Port B1**	
20	5	PB0/MCO	I/O	C_T	HS	x					x	**Port B0**	CPU clock output
1	6	PA2/IT3	I/O	C_T		x		\			x	**Port A2**	Interrupt 3 input
2	7	PA1/IT2	I/O	C_T		X		\			x	**Port A1**	Interrupt 2 input
3	8	PA0/IT1/USBOE	I/O	C_T		X		\			x	**Port A0**	Interrupt 1 input/USB Output Enable
10	15	$\overline{\text{RESET}}$	I/O	C								Top priority non maskable interrupt (active low)	
5	10	USBDM	I/O									USB bidirectional data (data -)	
6	11	USBDP	I/O									USB bidirectional data (data +)	
7	12	USBVCC	S									USB power supply 3.3V output	

PIN DESCRIPTION (Cont'd)

2.1 PCB LAYOUT RECOMMENDATION

In the case of DIP20 devices the user should layout the PCB so that the DIP20 ST7261 device and the USB connector are centered on the same axis ensuring that the D- and D+ lines are of equal length. Refer to Figure 4

Figure 4. Recommended PCB Layout for USB Interface with DIP20 package



3 REGISTER & MEMORY MAP

As shown in the Figure 5, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 64 bytes of register locations, 256 bytes of RAM and 4 Kbytes of user program memory. The RAM space includes up to 128 bytes for the stack from 0100h to 017Fh.

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

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

Related Documentation

AN 985: Executing Code in ST7 RAM

Figure 5. Memory Map

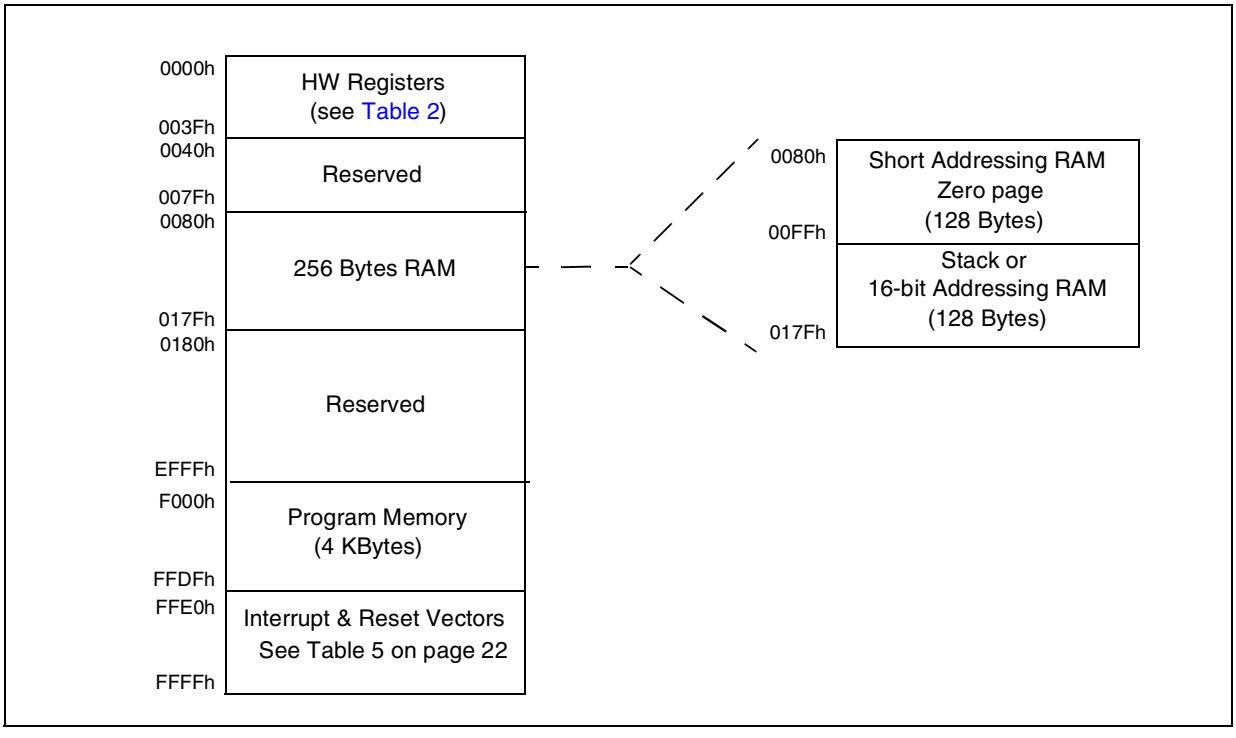


Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0000h 0001h	Port A	PADR PADDR	Port A Data Register Port A Data Direction Register	00h 00h	R/W R/W
0002h 0003h	Port B	PBDR PBDDR	Port B Data Register Port B Data Direction Register	00h 00h	R/W R/W.
0004h to 0007h	Reserved Area (4 Bytes)				
0008h		ITRFRE1	Interrupt Register 1	00h	R/W
0009h		MISC	Miscellaneous Register	00h	R/W
000Ah to 000Ch	Reserved Area (2 Bytes)				
000Dh	WDG	WDGCR	Watchdog Control Register	7Fh	R/W
000Eh to 0024h	Reserved Area (23 Bytes)				
0025h 0026h 0027h 0028h 0029h 002Ah 002Bh 002Ch 002Dh 002Eh 002Fh 0030h 0031h	USB	USBPIDR USBDMAR USBIDR USBISTR USBIMR USBCTLR USBDADDR USBEP0RA USBEP0RB USBEP1RA USBEP1RB USBEP2RA USBEP2RB	USB PID Register USB DMA Address register USB Interrupt/DMA Register USB Interrupt Status Register USB Interrupt Mask Register USB Control Register USB Device Address Register USB Endpoint 0 Register A USB Endpoint 0 Register B USB Endpoint 1 Register A USB Endpoint 1 Register B USB Endpoint 2 Register A USB Endpoint 2 Register B	x0h xxh x0h 00h 00h 06h 00h 0000 xxxxb 80h 0000 xxxxb 0000 xxxxb 0000 xxxxb 0000 xxxxb	Read Only R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W
0032h to 0035h	Reserved Area (4 Bytes)				
0036h 0037h	TBU	TBUCV TBUCSR	TBU Counter Value Register TBU Control/Status Register	00h 00h	R/W R/W
0038h to 003Fh	Reserved Area (8 Bytes)				

4 CENTRAL PROCESSING UNIT

4.1 INTRODUCTION

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

4.2 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

4.3 CPU REGISTERS

The 6 CPU registers shown in Figure 6 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)

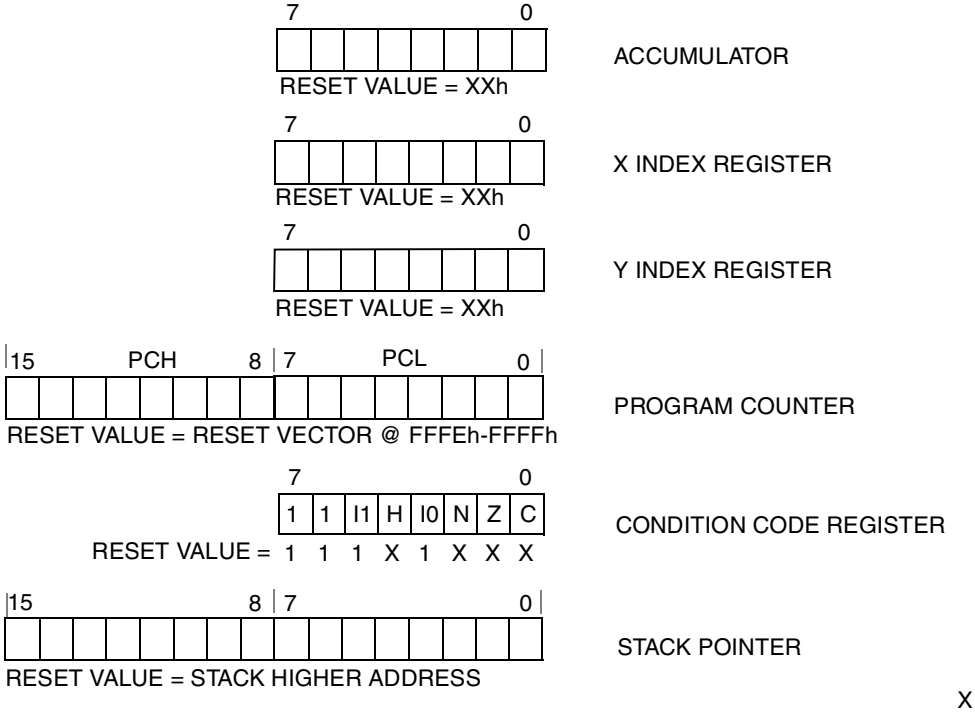
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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 6. 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.

CPU REGISTERS (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 017Fh

15							8
0	0	0	0	0	0	0	1

7							0
1	SP6	SP5	SP4	SP3	SP2	SP1	SP0

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

Since the stack is 128 bytes deep, the 9 most significant bits are forced by hardware. Following an MCU Reset, or after a Reset Stack Pointer instruction (RSP), the Stack Pointer contains its reset value (the SP6 to SP0 bits are set) which is the stack higher address.

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

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

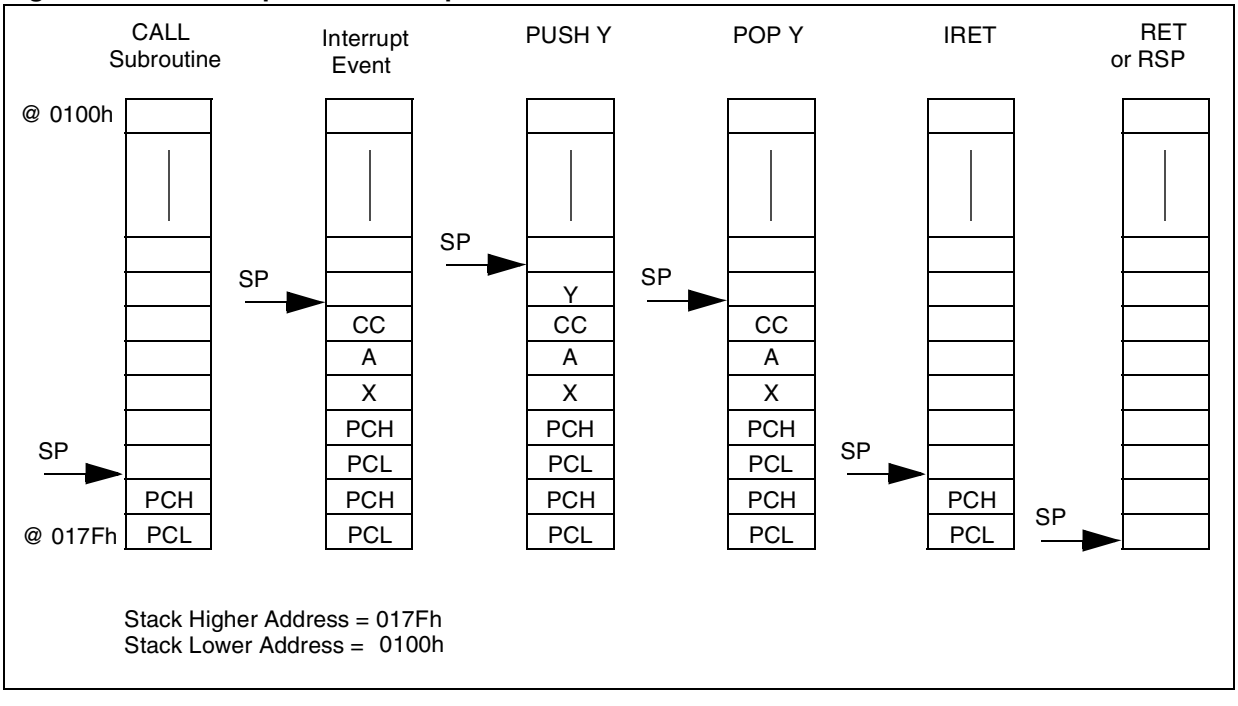
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 7.

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

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

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

Figure 7. Stack Manipulation Example



5 CLOCKS AND RESET

5.1 CLOCK SYSTEM

5.1.1 General Description

The MCU accepts either a Crystal or Ceramic resonator, or an external clock signal to drive the internal oscillator. The internal clock (f_{CPU}) is derived from the external oscillator frequency (f_{OSC}), by dividing by 3 and multiplying by 2. By setting the OSC12/6 bit in the option byte, a 12 MHz external clock can be used giving an internal frequency of 8 MHz while maintaining a 6 MHz clock for USB (refer to Figure 10).

The internal clock signal (f_{CPU}) consists of a square wave with a duty cycle of 50%.

It is further divided by 1, 2, 4 or 8 depending on the Slow Mode Selection bits in the Miscellaneous register (SMS[1:0])

The internal oscillator is designed to operate with an AT-cut parallel resonant quartz or ceramic resonator in the frequency range specified for f_{osc}. The circuit shown in Figure 9 is recommended when using a crystal, and Table 3 lists the recommended capacitors. 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 stabilization time.

Table 3. Recommended Values for 12 MHz Crystal Resonator

R_{SMAX}	20 Ω	25 Ω	70 Ω
C_{OSCIN}	56pF	47pF	22pF
C_{OSCOUT}	56pF	47pF	22pF
R_P	1-10 MΩ	1-10 MΩ	1-10 MΩ

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

5.1.2 External Clock input

An external clock may be applied to the OSCIN input with the OSCOUT pin not connected, as shown on Figure 8. The t_{OXOV} specifications does not apply when using an external clock input. The equivalent specification of the external clock source should be used instead of t_{OXOV} (see Electrical Characteristics).

5.1.3 Clock Output Pin (MCO)

The internal clock (f_{CPU}) can be output on Port B0 by setting the MCO bit in the Miscellaneous register.

Figure 8. External Clock Source Connections

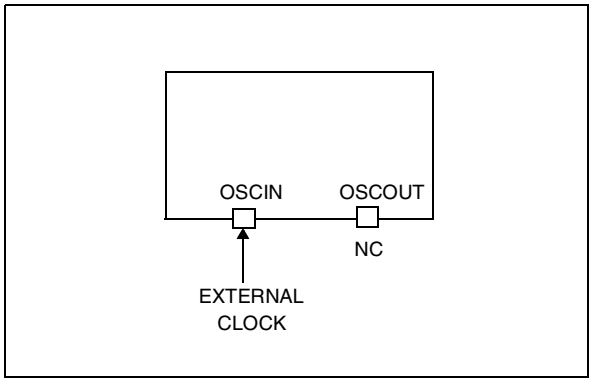


Figure 9. Crystal/Ceramic Resonator

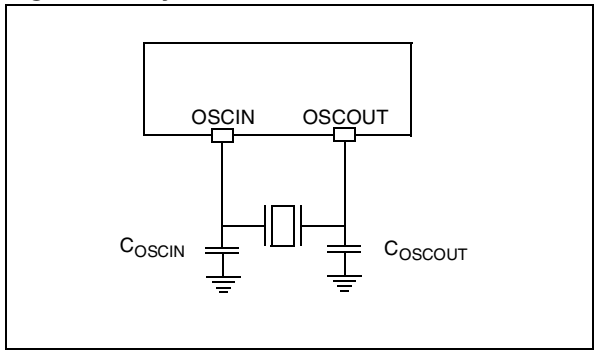
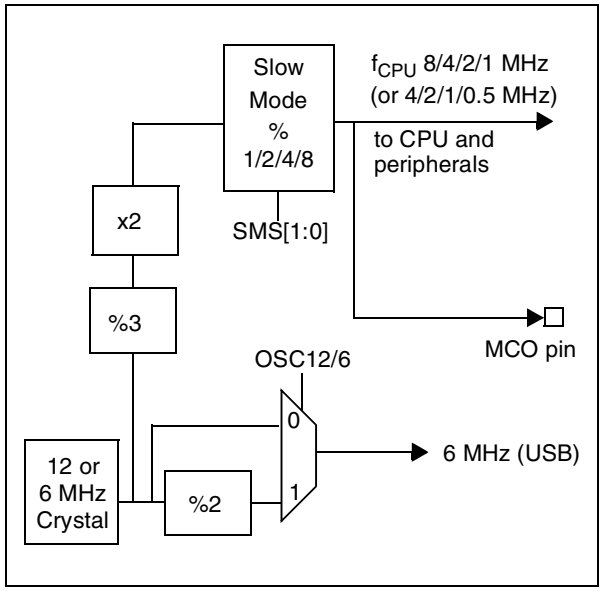


Figure 10. Clock block diagram



5.2 RESET

The Reset procedure is used to provide an orderly software start-up or to exit low power modes.

Three reset modes are provided: a low voltage reset, a watchdog reset and an external reset at the $\overline{\text{RESET}}$ pin.

A reset causes the reset vector to be fetched from addresses FFFEh and FFFFh in order to be loaded into the PC and with program execution starting from this point.

An internal circuitry provides a 514 CPU clock cycle delay from the time that the oscillator becomes active.

5.2.1 Low Voltage Reset

Low voltage reset circuitry generates a reset when V_{DD} is:

- below V_{IT+} when V_{DD} is rising,
- below V_{IT-} when V_{DD} is falling.

During low voltage reset, the $\overline{\text{RESET}}$ pin is held low, thus permitting the MCU to reset other devices.

The Low Voltage Detector must be kept enabled by resetting the bit 3 of the option byte.

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.

5.2.2 Watchdog Reset

When a watchdog reset occurs, the $\overline{\text{RESET}}$ pin is pulled low permitting the MCU to reset other devices as when low voltage reset (Figure 11).

5.2.3 External Reset

The external reset is an active low input signal applied to the $\overline{\text{RESET}}$ pin of the MCU.
As shown in Figure 14, the $\overline{\text{RESET}}$ signal must stay low for a minimum of one and a half CPU clock cycles.

An internal Schmitt trigger at the $\overline{\text{RESET}}$ pin is provided to improve noise immunity.

Figure 11. Low Voltage Reset functional Diagram

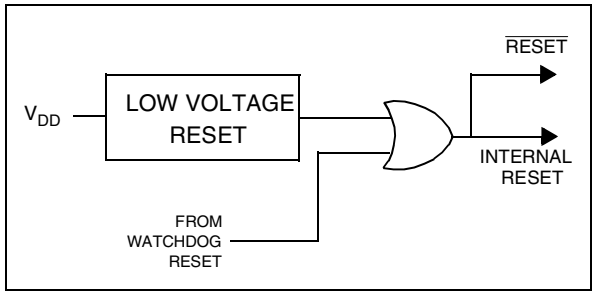
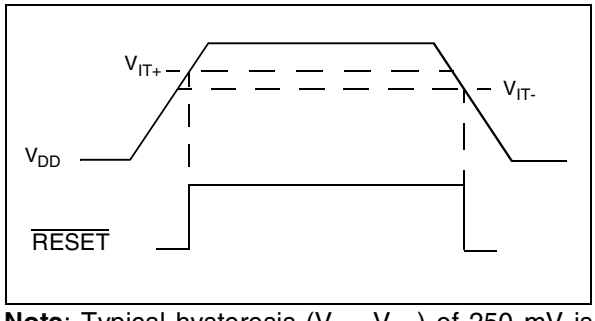


Figure 12. Low Voltage Reset Signal Output



Note: Typical hysteresis (V_{IT+}-V_{IT-}) of 250 mV is expected

Figure 13. Temporization Timing Diagram after an internal Reset

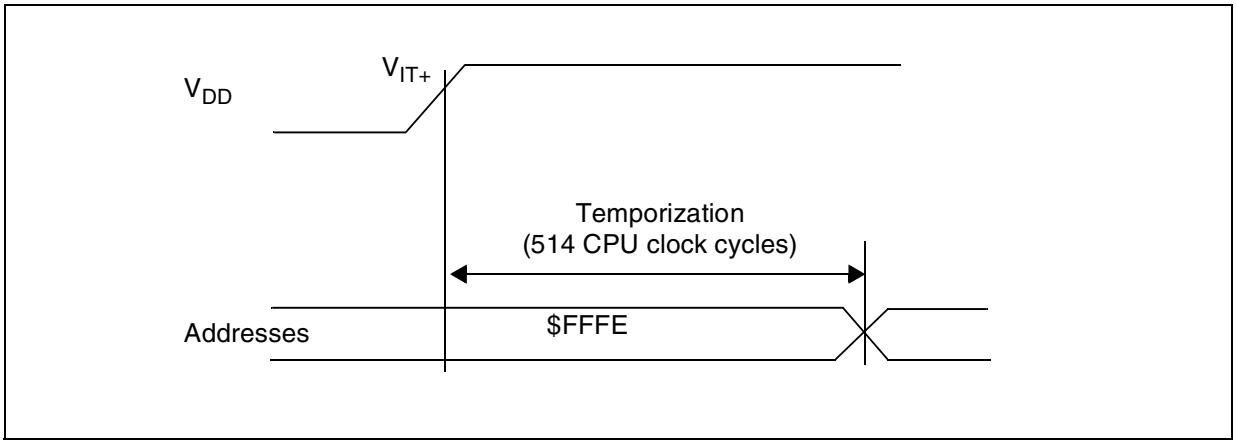
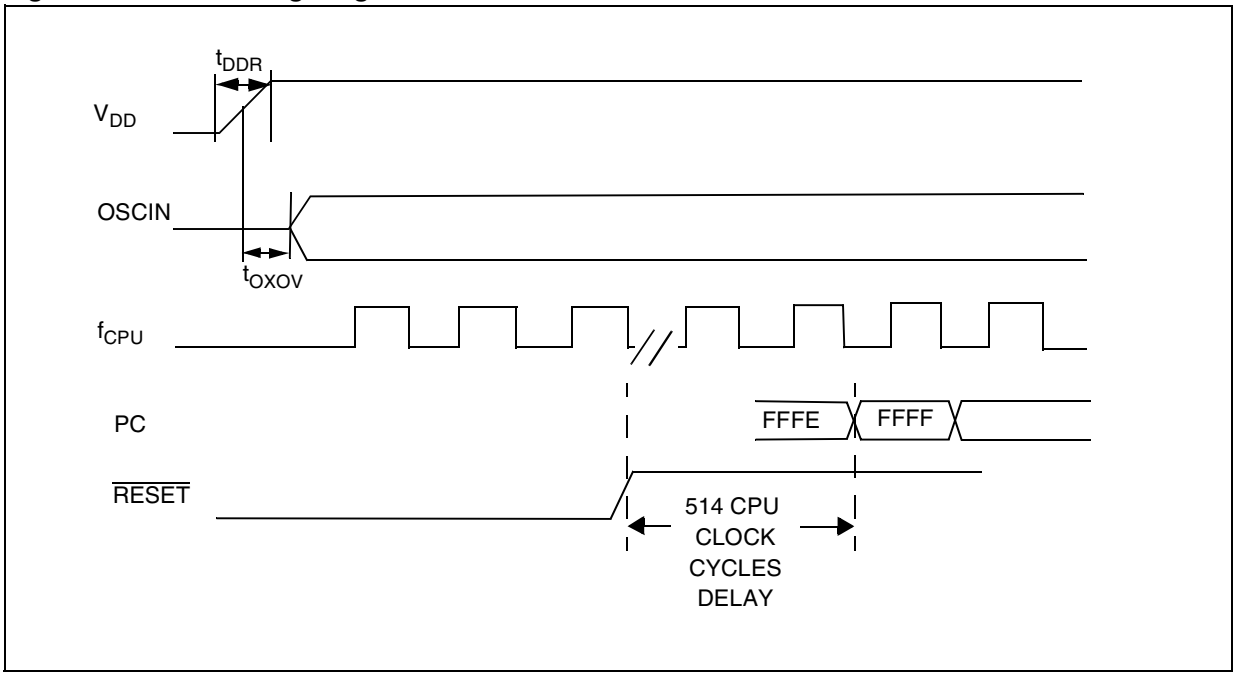
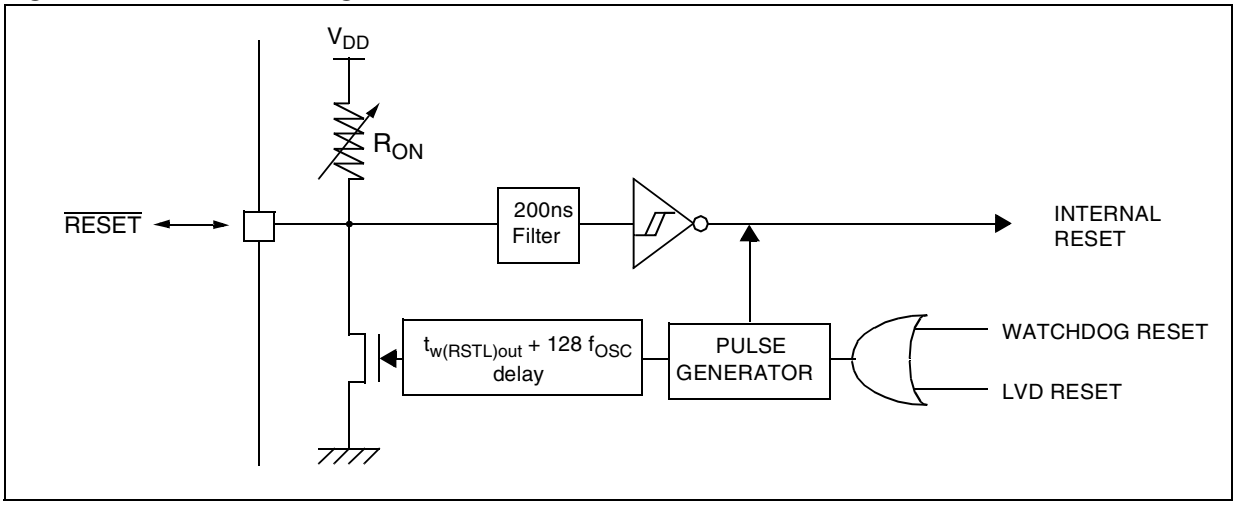


Figure 14. Reset Timing Diagram



Note: Refer to Electrical Characteristics for values of t_{DDR}, t_{OXOV}, V_{IT+} and V_{IT-}.

Figure 15. Reset Block Diagram



Note: The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD or watchdog).

6 INTERRUPTS

6.1 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.

6.2 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 4). 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).

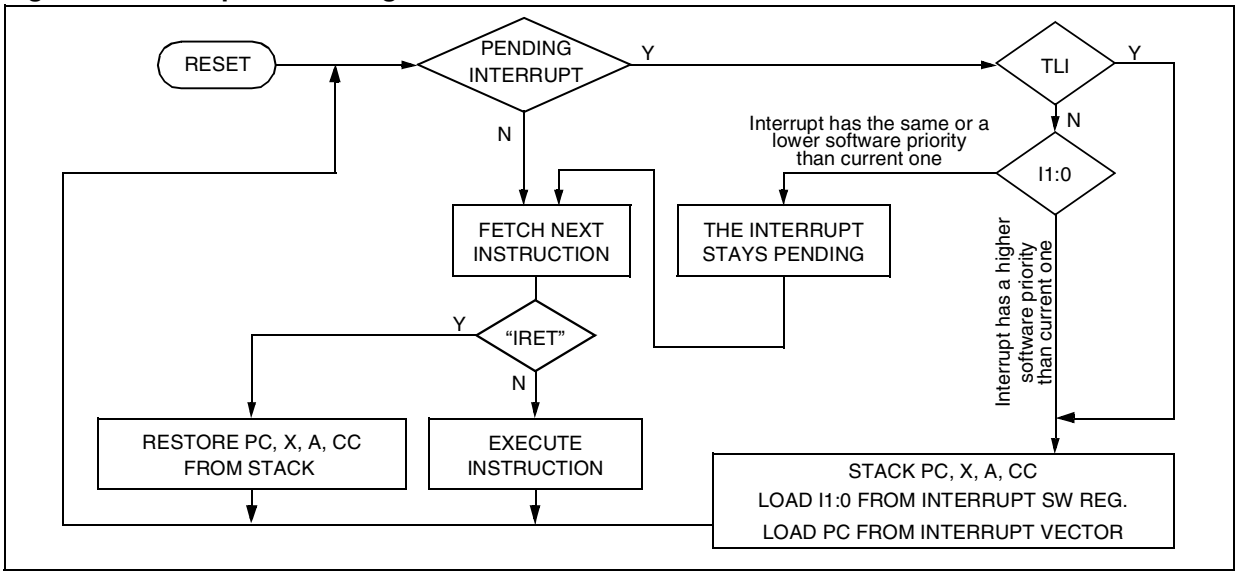
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

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 4. 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

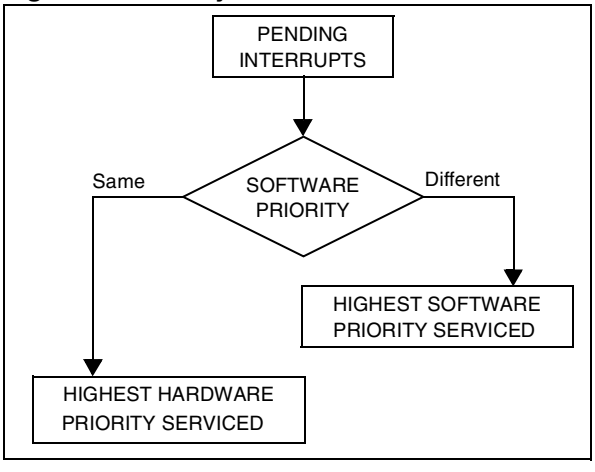
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– 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 ITRFRE2 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)

6.3 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.

6.4 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

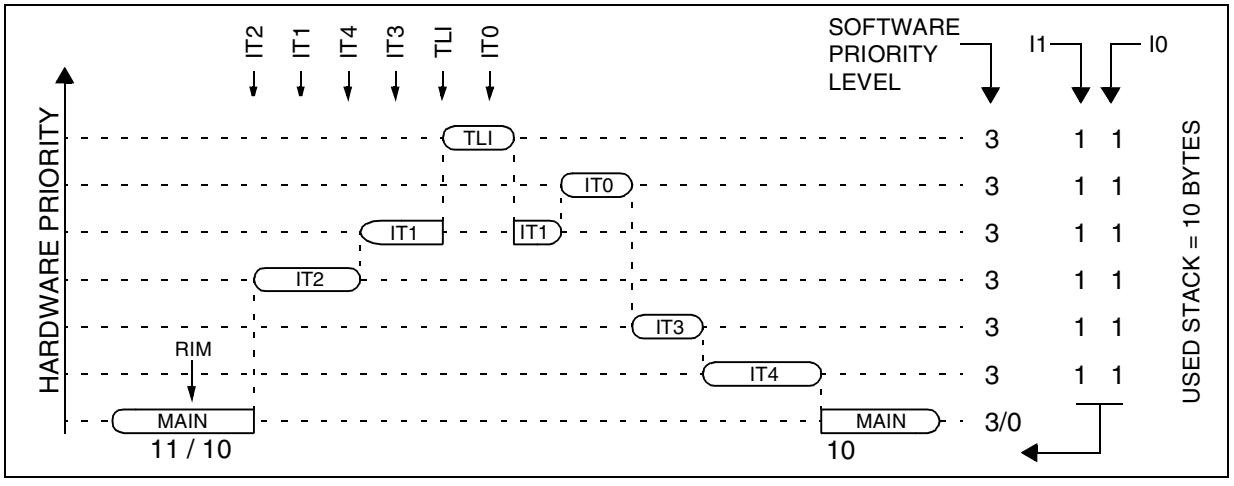
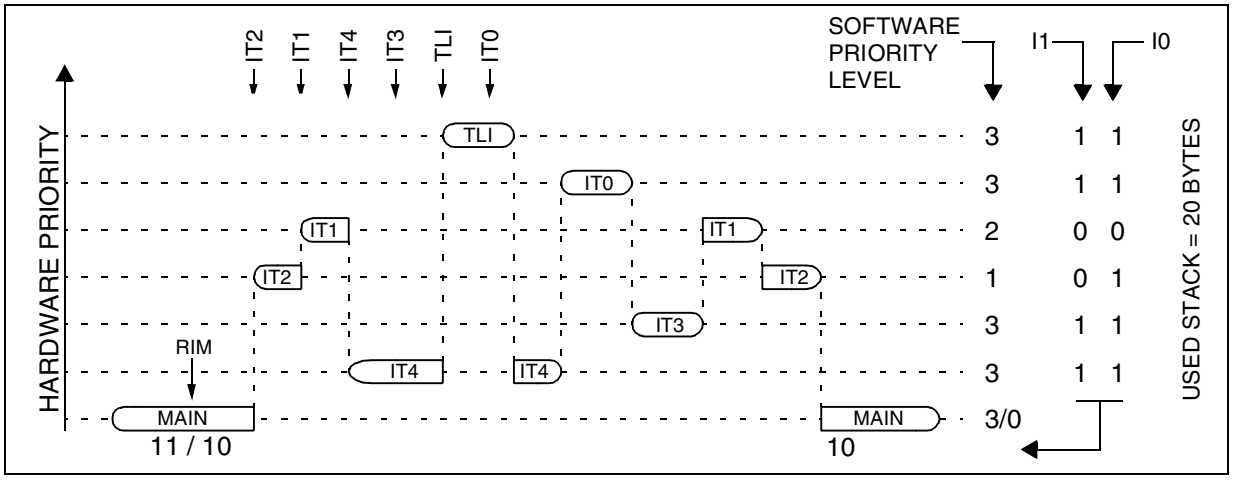


Figure 19. Nested Interrupt Management



INTERRUPTS (Cont'd)

6.5 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).

6.6 Interrupt Register

INTERRUPT REGISTER 1 (ITRFRE1)

Address: 0008h - Read/Write

Reset Value: 0000 0000 (00h)

7							0
IT8E	IT7E	IT6E	IT5E	IT4E	IT3E	IT2E	IT1E

Bit 7:0 = **ITiE** *Interrupt Enable*
0: I/O pin free for general purpose I/O
1: ITi external interrupt enabled.

Note: The corresponding interrupt is generated when:

– a rising edge occurs on the IT5/IT6 pins

– a falling edge occurs on the IT1, 2, 3, 4, 7 and 8 pins

INTERRUPT REGISTER 2 (ITRFRE2)

Address: 0039h - Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTL3	CTL2	CTL1	CTL0	IT12E	IT11E	IT10E	IT9E

Bit 7:6 = **CTL[3:2]** *IT[12:11] Interrupt Sensitivity*

These bits are set and cleared by software. They are used to configure the edge and level sensitivity of the IT12 and IT11 external interrupt pins (this means that both must have the same sensitivity).

CTL3	CTL2	IT[12:11] Sensitivity
0	0	Falling edge and low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Bit 5:4 = **CTL[1:0]** *IT[10:9]1nterrupt Sensitivity*

These bits are set and cleared by software. They are used to configure the edge and level sensitivity of the IT10 and IT9 external interrupt pins (this means that both must have the same sensitivity).

CTL1	CTL0	IT[10:9] Sensitivity
0	0	Falling edge and low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Bit 3:0 = **ITiE** *Interrupt Enable*
0: I/O pin free for general purpose I/O
1: ITi external interrupt enabled.

INTERRUPTS (Cont'd)

Table 5. Interrupt Mapping

N°	Source Block	Description	Register Label	Exit from HALT	Address Vector	Priority Order
		Reset Vector		Yes	FFFEh-FFFFh	Highest Priority
		TRAP software interrupt vector		No	FFFCh-FFFDh	
0	NOT USED				FFFAh-FFFBh	
1	USB	USB End Suspend interrupt vector	USBISTR	Yes	FFF8h-FFF9h	
2	I/O Ports	Port A external interrupts IT[3:1]	ITRFRE1	Yes	FFF6h-FFF7h	
3		Port B external interrupts IT[8:5]		Yes	FFF4h-FFF5h	
4	NOT USED				FFF2h-FFF3h	
5	TBU	Timebase Unit interrupt vector	TBUCSR	No	FFF0h-FFF1h	
6	NOT USED				FFEEh-FFEFh	
7	NOT USED				FFECh-FFEDh	
8	NOT USED				FFEAh-FFEBh	
9	USB	USB interrupt vector	USBISTR	No	FFE8h-FFE9h	Lowest Priority
10	NOT USED				FFE6h-FFE7h	

Table 6. Nested Interrupts Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0032h		Ext. Interrupt Port B		Ext. Interrupt Port A		USB END SUSP		Not Used	
	ISPR0 Reset Value	I1_3 1	I0_3 1	I1_2 1	I0_2 1	I1_1 1	I0_1 1	1	1
0033h		SPI		ART		TBU		Ext. Interrupt Port C	
	ISPR1 Reset Value	I1_7 1	I0_7 1	I1_6 1	I0_6 1	I1_5 1	I0_5 1	I1_4 1	I0_4 1
0034h		Not Used		ADC		USB		SCI	
	ISPR2 Reset Value	I1_11 1	I0_11 1	I1_10 1	I0_10 1	I1_9 1	I0_9 1	I1_8 1	I0_8 1
0035h						Not Used		Not Used	
	ISPR3 Reset Value	1	1	1	1	I1_13 1	I0_13 1	I1_12 1	I0_12 1

7 POWER SAVING MODES

7.1 INTRODUCTION

There are three Power Saving modes. Slow Mode is selected by setting the SMS bits in the Miscellaneous register. Wait and Halt modes may be entered using the WFI and HALT instructions.

After a RESET the normal operating mode is selected by default (RUN mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency divided by 3 and multiplied by 2 (f_{CPU}).

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

7.1.1 Slow Mode

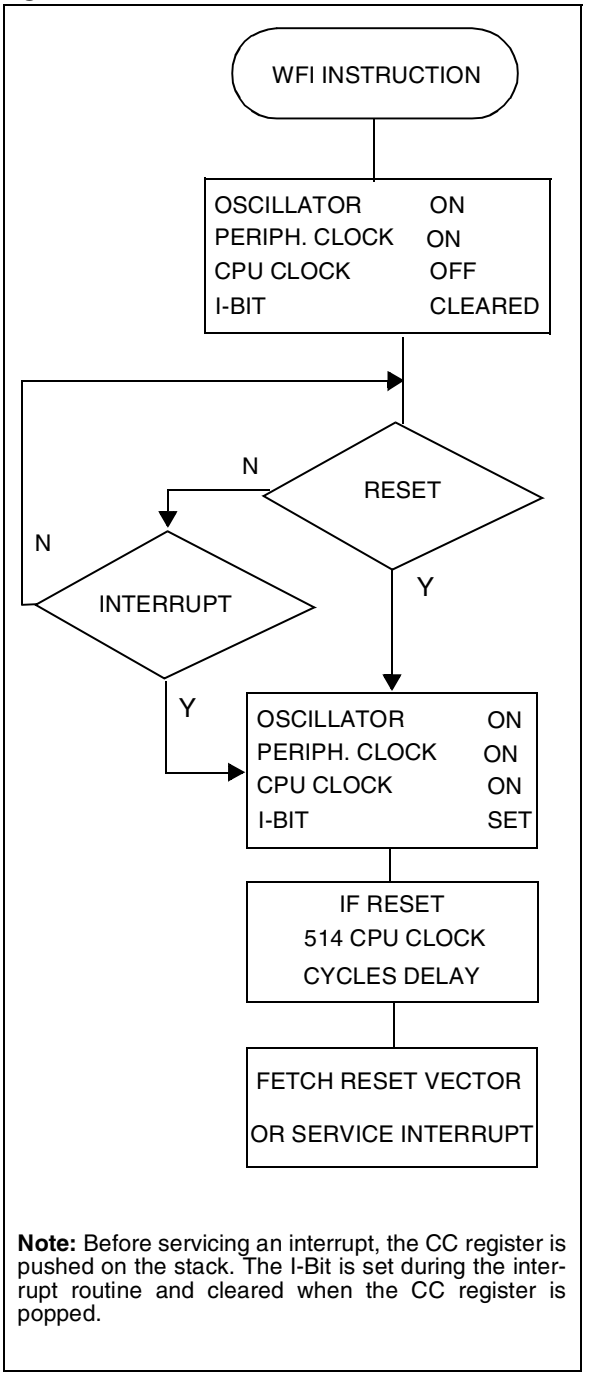
In Slow mode, the oscillator frequency can be divided by a value defined in the Miscellaneous Register. The CPU and peripherals are clocked at this lower frequency. Slow mode is used to reduce power consumption, and enables the user to adapt clock frequency to available supply voltage.

7.2 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)

7.3 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.

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 514 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.

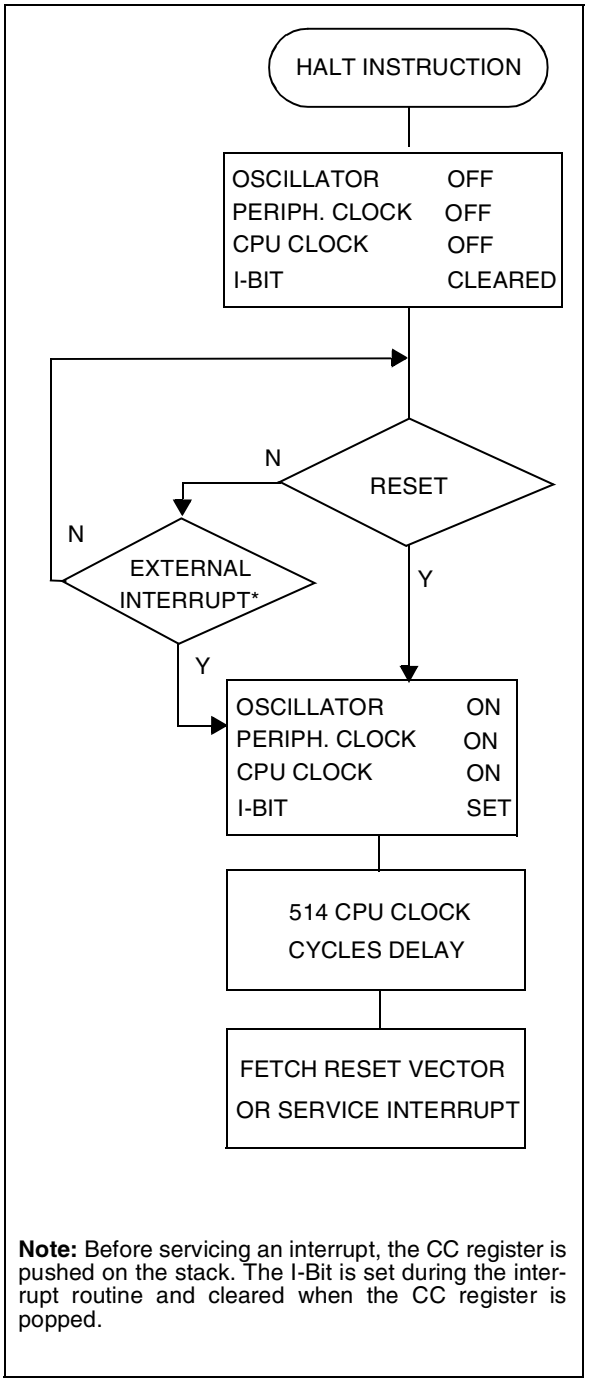
Related Documentation

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

AN1014: How to Minimize the ST7 Power Consumption

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

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.

8 I/O PORTS

8.1 INTRODUCTION

The I/O ports offer different functional modes:

transfer of data through digital inputs and outputs and for specific pins:

– Analog signal input (ADC)

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

– External interrupt generation

An I/O port is composed of up to 8 pins. Each pin can be programmed independently as digital input or digital output.

8.2 FUNCTIONAL DESCRIPTION

Each port is associated with 2 main registers:

– Data Register (DR)

– Data Direction Register (DDR)

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 7. I/O Pin Functions

DDR	MODE
0	Input
1	Output

8.2.1 Input Modes

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

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

Notes:

1. All the inputs are triggered by a Schmitt trigger.
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 external interrupt function of an I/O pin, is enabled using the ITFRE registers, an event on this I/O can generate an external Interrupt request to the CPU. The interrupt sensitivity is programmable, the options are given in the description of the ITRFRE interrupt registers.

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 ANDed and inverted. For this reason, if an event occurs on one of the interrupt pins, it masks the other ones.

8.2.2 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.

8.2.3 Alternate Functions

Digital Alternate Functions

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over 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 state is also digitally readable by addressing the DR register.

Notes:

1. Input pull-up configuration can cause an unexpected value at 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*:** Alternate functions of peripherals must must not be activated when the external interrupts are enabled on the same pin, in order to avoid generating spurious interrupts.

I/O PORTS (Cont'd)

Analog Alternate Functions

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

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

Warning: The analog input voltage level must be within the limits stated in the Absolute Maximum Ratings.

8.2.4 I/O Port Implementation

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

I/O PORTS (Cont'd)

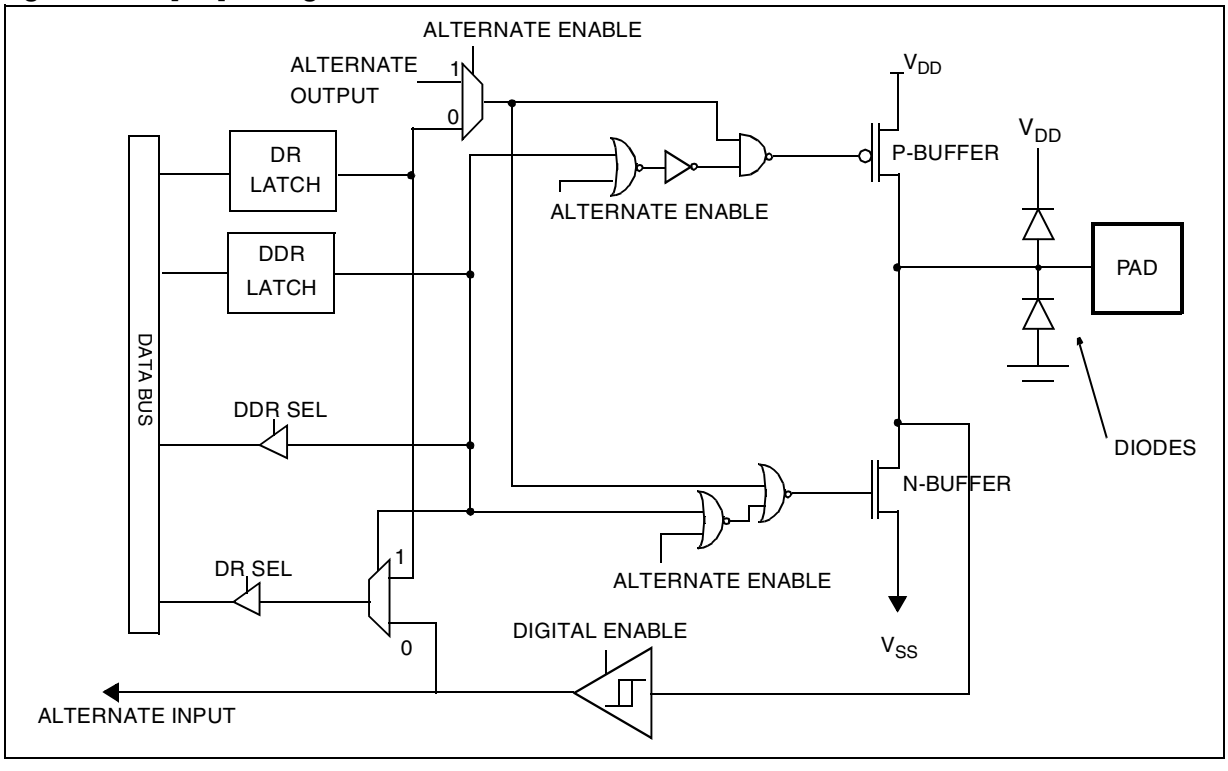
8.2.5 Port A

Table 8. Port A Description

PORT A	I/O		Alternate Function	
	Input*	Output	Signal	Condition
PA0	floating	push-pull	USBOE	USBOE = 1 (MISC)
			IT1 Schmitt triggered input	IT1E = 1 (ITRFRE1)
PA1	floating	push-pull	IT2 Schmitt triggered input	IT2E = 1 (ITRFRE1)
PA2	floating	push-pull	IT3 Schmitt triggered input	IT3E = 1 (ITRFRE1)

*Reset State

Figure 22. PA[2:0] Configuration



I/O PORTS (Cont'd)

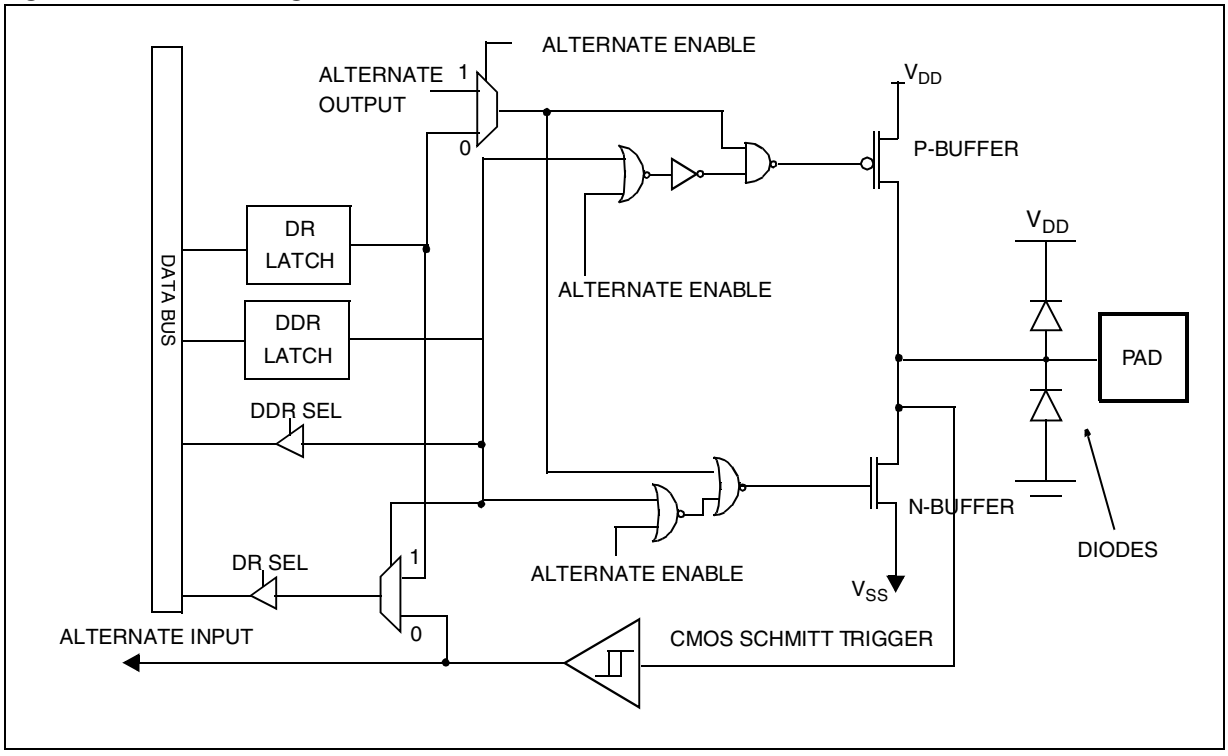
8.2.6 Port B

Table 9. Port B Description

PORT B	I/O		Alternate Function	
	Input*	Output	Signal	Condition
PB0	floating	push-pull (high sink)	MCO (Main Clock Output)	MCO = 1 (MISCR)
PB1	floating	push-pull (high sink)		
PB2	floating	push-pull (high sink)		
PB3	floating	push-pull (high sink)		
PB4	floating	push-pull (high sink)	IT5 Schmitt triggered input	IT5E = 1 (ITRFRE1)
PB5	floating	push-pull (high sink)	IT6 Schmitt triggered input	IT6E = 1 (ITRFRE1)
PB6	floating	push-pull (high sink)	IT7 Schmitt triggered input	IT7E = 1 (ITRFRE1)
PB7	floating	push-pull (high sink)	IT8 Schmitt triggered input	IT8E = 1 (ITRFRE1)

*Reset State

Figure 23. Port B Configuration



I/O PORTS (Cont'd)

8.2.7 Register Description

DATA REGISTER (DR)

Port x Data Register
PxDR with x = A or B.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

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

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

DATA DIRECTION REGISTER (DDR)

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

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0

Bit 7:0 = **DD[7:0]** *Data direction register 8 bits.*

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

0: Input mode
1: Output mode

I/O PORTS (Cont'd)

Table 10. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all I/O port registers		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PBDR**	MSB							LSB
0003h	**PBDDR**								

8.3 MISCELLANEOUS REGISTER

MISCELLANEOUS REGISTER
Read Write
Reset Value - 0000 0000 (00h)

7							0
-	-	-	-	SMS1	SMS0	US-BOE	MCO

Bits 7:4 = Reserved

Bits 3:2 = **SMS[1:0]** *Slow Mode Selection*
These bits select the Slow Mode frequency (depending on the oscillator frequency configured by option byte).

OSC12/6	SMS1	SMS0	Slow Mode Frequency (MHz.)
f_{OSC}= 6 MHz.	0	0	4
	0	1	2
	1	0	1
	1	1	0.5
f_{OSC}= 12 MHz.	0	0	8
	0	1	4
	1	0	2
	1	1	1

Bit 1 = **USBOE** *USB Output Enable*
0: PA0 port free for general purpose I/O
1: USBOE alternate function enabled. The USB output enable signal is output on the PA0 port (at “1” when the ST7 USB is transmitting data).

Bit 0 = **MCO** *Main Clock Out*
0: PB0 port free for general purpose I/O
1: MCO alternate function enabled (f_{CPU} output on PB0 I/O port)

Related Documentation

AN 970: SPI Communication between ST7 and EEPROM

AN1045: S/W implementation of I2C bus master

AN1048: Software LCD driver

9 ON-CHIP PERIPHERALS

9.1 WATCHDOG TIMER (WDG)

9.1.1 Introduction

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

9.1.2 Main Features

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

9.1.3 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 11):

– The WDGA bit is set (watchdog enabled)

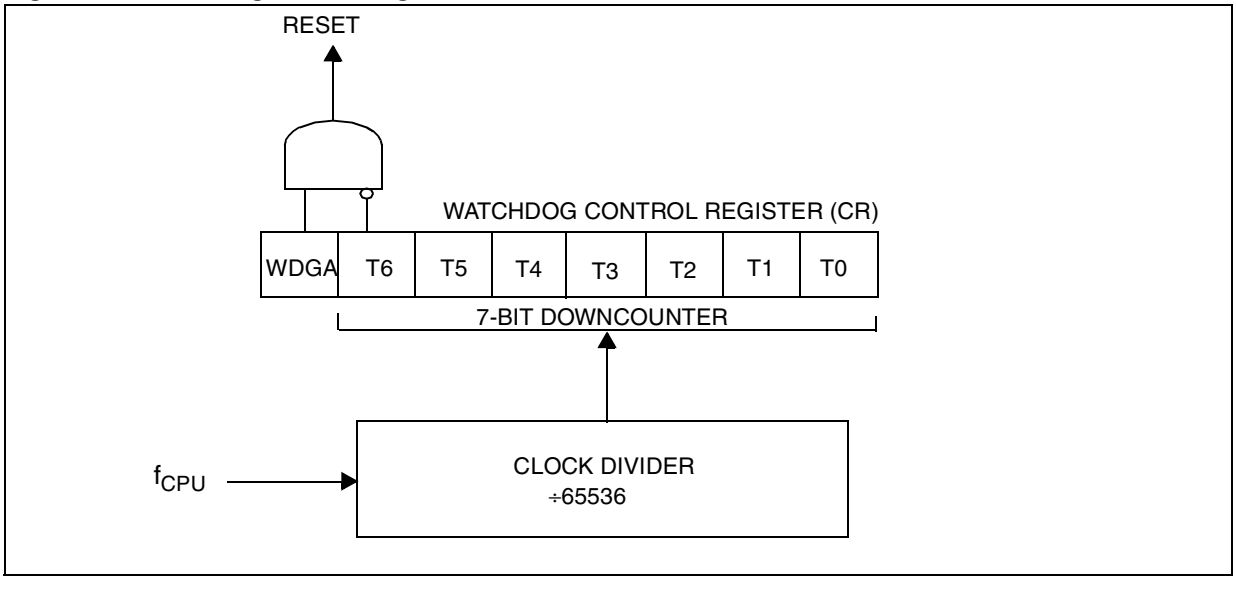
– The T6 bit is set to prevent generating an immediate reset

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

Table 11.Watchdog Timing (f_{CPU} = 8 MHz)

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

Figure 24. Watchdog Block Diagram



WATCHDOG TIMER (Cont'd)

9.1.4 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).

9.1.5 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.

9.1.6 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).

9.1.7 Interrupts

None.

9.1.8 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.

Bits 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).

Table 12. Watchdog Timer Register Map and Reset Values

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

9.2 TIMEBASE UNIT (TBU)

9.2.1 Introduction

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

9.2.2 Main Features

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

9.2.3 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.

9.2.4 Programming Example

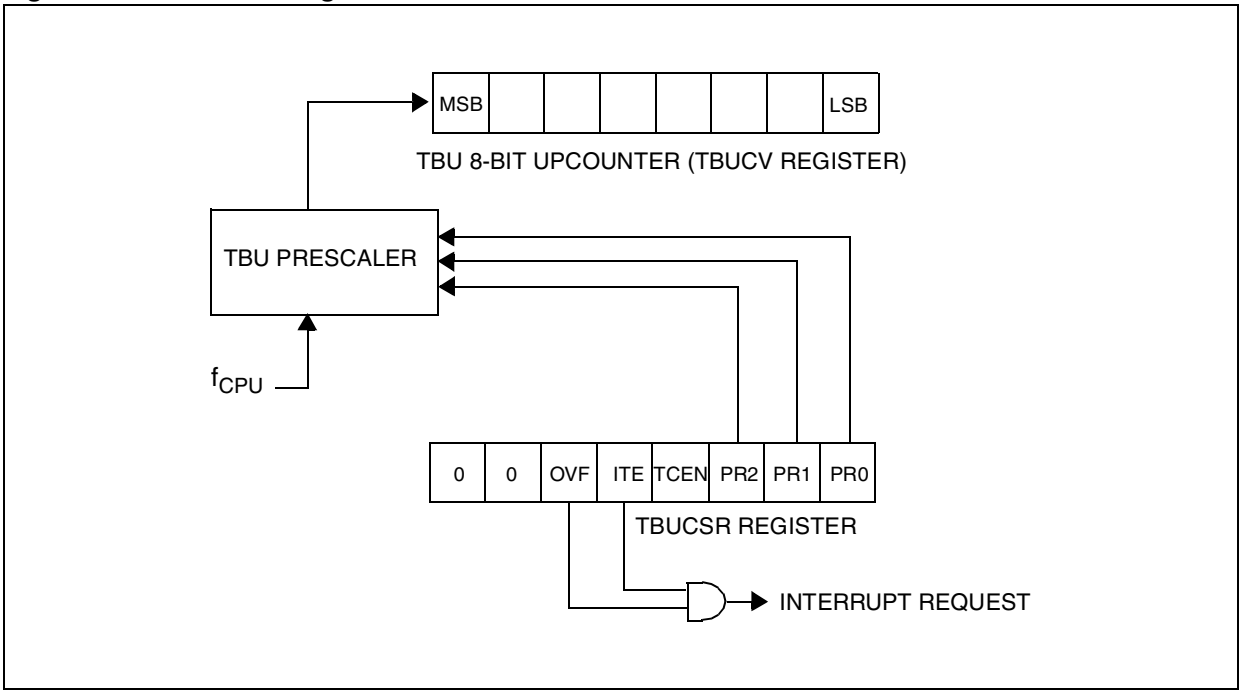
In this example, timer is required to generate an interrupt after a delay of 1 ms.

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 25. TBU Block Diagram



TIMEBASE UNIT (Cont'd)

9.2.5 Low Power Modes

Mode	Description
WAIT	No effect on TBU
HALT	TBU halted.

9.2.6 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).

9.2.7 Register Description

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

7							0
CV7	CV6	CV5	CV4	CV3	CV2	CV1	CV0

Bit 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

Bit 7:6 = Reserved must be kept cleared.

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.

Bit 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	1	8
1	0	0	16
1	0	1	32
1	0	1	64
1	1	0	128
1	1	1	256

TIMEBASE UNIT (Cont'd)

Table 13. TBU Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0036h	**TBUCV** Reset Value	CV7 0	CV6 0	CV5 0	CV4 0	CV3 0	CV2 0	CV1 0	CV0 0
0037h	**TBUSR** Reset Value	- 0	- 0	OVF 0	ITE 0	TCEN 0	PR2 0	PR1 0	PR0 0

9.3 USB INTERFACE (USB)

9.3.1 Introduction

The USB Interface implements a low-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 DMA. No external components are needed apart from the external pull-up on USBDM for low speed recognition by the USB host. The use of DMA architecture allows the endpoint definition to be completely flexible. Endpoints can be configured by software as in or out.

9.3.2 Main Features

- USB Specification Version 1.1 Compliant
- Supports Low-Speed USB Protocol
- Two or Three Endpoints (including default one) depending on the device (see device feature list and register map)
- CRC generation/checking, NRZI encoding/decoding and bit-stuffing
- USB Suspend/Resume operations
- DMA Data transfers
- On-Chip 3.3V Regulator
- On-Chip USB Transceiver

9.3.3 Functional Description

The block diagram in Figure 26, 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.

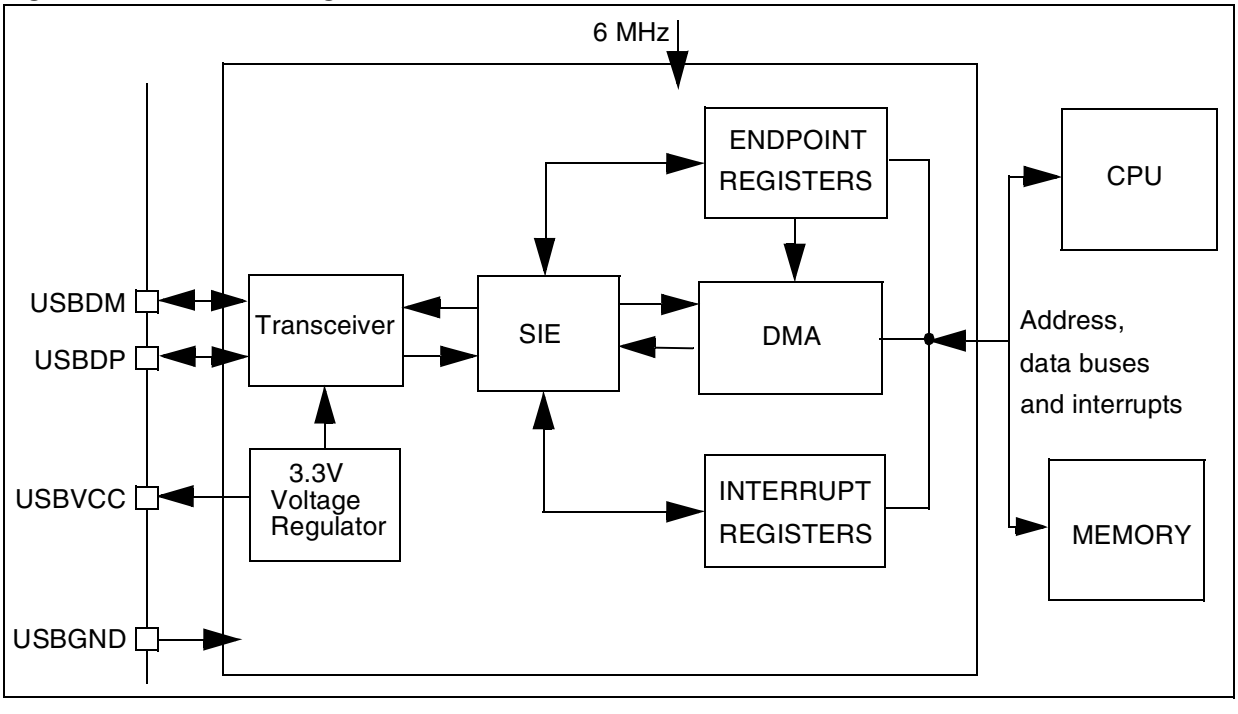
DMA

When a token for a valid Endpoint is recognized by the USB interface, the related data transfer takes place, using DMA. 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 26. USB Block Diagram



USB INTERFACE (Cont'd)

9.3.4 Register Description

DMA ADDRESS REGISTER (DMAR)

Read / Write

Reset Value: Undefined

7							0
DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8

Bits 7:0=**DA[15:8]** *DMA address bits 15-8.*
Software must write the start address of the DMA memory area whose most significant bits are given by DA15-DA6. The remaining 6 address bits are set by hardware. See the description of the IDR register and Figure 27.

INTERRUPT/DMA REGISTER (IDR)

Read / Write

Reset Value: xxxx 0000 (x0h)

7							0
DA7	DA6	EP1	EP0	CNT3	CNT2	CNT1	CNT0

Bits 7:6 = **DA[7:6]** *DMA address bits 7-6.*
Software must reset these bits. See the description of the DMAR register and Figure 27.

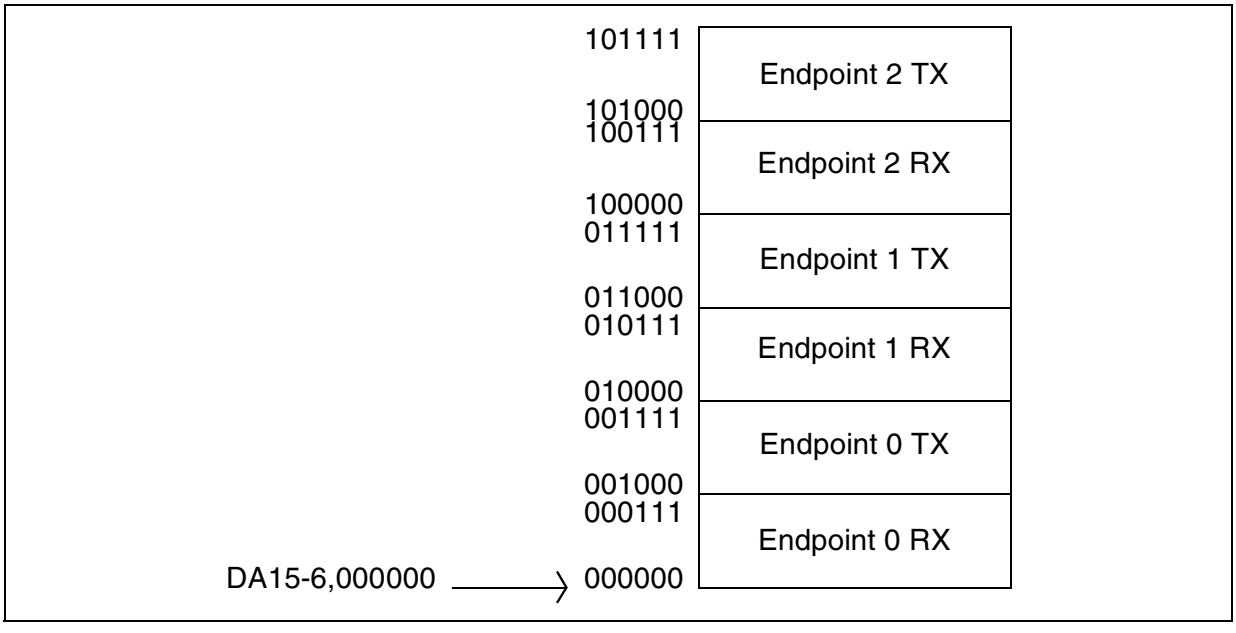
Bits 5:4 = **EP[1:0]** *Endpoint number* (read-only).
These bits identify the endpoint which required attention.
00: Endpoint 0
01: Endpoint 1
10: Endpoint 2

When a CTR interrupt occurs (see register ISTR) the software should read the EP bits to identify the endpoint which has sent or received a packet.

Bits 3:0 = **CNT[3:0]** *Byte count* (read only).
This field shows how many data bytes have been received during the last data reception.

Note: Not valid for data transmission.

Figure 27. DMA Buffers



USB INTERFACE (Cont'd)

PID REGISTER (PIDR)

Read only

Reset Value: xx00 0000 (x0h)

7							0
TP3	TP2	0	0	0	RX_ SEZ	RXD	0

Bits 7:6 = **TP[3:2]** *Token PID bits 3 & 2*.
USB token PIDs are encoded in four bits. **TP[3:2]** correspond to the variable token PID bits 3 & 2.
Note: PID bits 1 & 0 have a fixed value of 01.
When a CTR interrupt occurs (see register ISTR) the software should read the TP3 and TP2 bits to retrieve the PID name of the token received.
The USB standard defines TP bits as:

TP3	TP2	PID Name
0	0	OUT
1	0	IN
1	1	SETUP

Bits 5:3 Reserved. Forced by hardware to 0.

Bit 2 = **RX_SEZ** *Received single-ended zero*
This bit indicates the status of the RX_SEZ transceiver output.
0: No SE0 (single-ended zero) state
1: USB lines are in SE0 (single-ended zero) state

Bit 1 = **RXD** *Received data*
0: No K-state
1: USB lines are in K-state

This bit indicates the status of the RXD transceiver output (differential receiver output).

Note: If the environment is noisy, the RX_SEZ and RXD bits can be used to secure the application. By interpreting the status, software can distinguish a valid End Suspend event from a spurious wake-up due to noise on the external USB line. A valid End Suspend is followed by a Resume or Reset sequence. A Resume is indicated by RXD=1, a Reset is indicated by RX_SEZ=1.

Bit 0 = Reserved. Forced by hardware to 0.

INTERRUPT STATUS REGISTER (ISTR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
SUSP	DOVR	CTR	ERR	IOVR	ESUSP	RESET	SOF

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: These bits cannot be set by software.

Bit 7 = **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 bus. The suspend request check is active immediately after each USB reset event and its disabled by hardware when suspend mode is forced (FSUSP bit of CTLR register) until the end of resume sequence.

Bit 6 = **DOVR** *DMA over/underrun*.
This bit is set by hardware if the ST7 processor can't answer a DMA request in time.
0: No over/underrun detected
1: Over/underrun detected

Bit 5 = **CTR** *Correct Transfer.* This bit is set by hardware when a correct transfer operation is performed. The type of transfer can be determined by looking at bits TP3-TP2 in register PIDR. The Endpoint on which the transfer was made is identified by bits EP1-EP0 in register IDR.
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 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 or nonstandard framing error detected

USB INTERFACE (Cont'd)

Bit 3 = **IOVR** *Interrupt overrun.*
This bit is set when hardware tries to set ERR, or SOF before they have been cleared by software.
0: No overrun detected
1: Overrun detected

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, EP0RA, EP0RB, EP1RA, EP1RB, EP2RA and EP2RB registers are reset by a USB reset.

Bit 0 = **SOF** *Start of frame.*
This bit is set by hardware when a low-speed SOF indication (keep-alive strobe) is seen on the USB bus. It is also issued at the end of a resume sequence.
0: No SOF signal detected
1: SOF signal detected

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..

INTERRUPT MASK REGISTER (IMR)

Read / Write

Reset Value: 0000 0000 (00h)

7							0
SUSPM	DOVRM	CTRM	ERRM	IOVRM	ESUSPM	RESETM	SOFM

Bits 7:0 = These bits are mask bits for all interrupt condition bits included in the ISTR. Whenever one of the IMR bits is set, if the corresponding ISTR 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 corresponding bit description in ISTR.

CONTROL REGISTER (CTLR)

Read / Write

Reset Value: 0000 0110 (06h)

7							0
0	0	0	0	RESUME	PDWN	FSUSP	FRES

Bits 7:4 = 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 halted allowing at least 600 ns before issuing the HALT instruction.
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 from the USB bus or forced through bit FRES in the CTLR register.

ENDPOINT n REGISTER A (EPnRA)

Read / Write

Reset Value: 0000 xxxx (0xh)

7							0
ST_ OUT	DTOG _TX	STAT _TX1	STAT _TX0	TBC 3	TBC 2	TBC 1	TBC 0

These registers (**EP0RA**, **EP1RA** and **EP2RA**) are used for controlling data transmission. They are also reset by the USB bus reset.

Note: Endpoint 2 and the EP2RA register are not available on some devices (see device feature list and register map).

Bit 7 = **ST_OUT** *Status out.*
This bit is set by software to indicate that a status out packet is expected: in this case, all nonzero OUT data transfers on the endpoint are STALLed instead of being ACKed. When ST_OUT is reset, OUT transactions can have any number of bytes, as needed.

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 at the 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 (see EPnRB) are normally updated by hardware, at the receipt of a relevant PID. They can be also written by software.

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:

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. Hardware sets the STAT_TX bits to NAK when a correct transfer has occurred (CTR=1) related to a IN or SETUP transaction addressed to this endpoint; this allows the software to prepare the next set of data to be transmitted.

Bits 3:0 = **TBC[3:0]** *Transmit byte count for Endpoint n.*
Before transmission, after filling the transmit buffer, software must write in the TBC field the transmit packet size expressed in bytes (in the range 0-8).

Warning: Any value outside the range 0-8 will-induce undesired effects (such as continuous data transmission).

USB INTERFACE (Cont'd)

ENDPOINT n REGISTER B (EPnRB)

Read / Write

Reset Value: 0000 xxxx (0xh)

7							0
CTRL	DTOG _RX	STAT _RX1	STAT _RX0	EA3	EA2	EA1	EA0

These registers (**EP1RB** and **EP2RB**) are used for controlling data reception on Endpoints 1 and 2. They are also reset by the USB bus reset.

Note: Endpoint 2 and the EP2RB register are not available on some devices (see device feature list and register map).

Bit 7 = **CTRL** *Control.*
This bit should be 0.

Note: If this bit is 1, the Endpoint is a control endpoint. (Endpoint 0 is always a control Endpoint, but it is possible to have more than one control Endpoint).

Bit 6 = **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.

Bits 5:4 = **STAT_RX [1:0]** *Status bits, for reception transfers.*
These bits contain the information about the endpoint status, which are listed below:

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. Hardware sets the STAT_RX bits to NAK when a correct transfer has occurred (CTR=1) related to an OUT or SETUP transaction addressed to this endpoint, so the software has the time to elaborate the received data before acknowledging a new transaction.

Bits 3:0 = **EA[3:0]** *Endpoint address*.
Software must write in this field the 4-bit address used to identify the transactions directed to this endpoint. Usually EP1RB contains "0001" and EP2RB contains "0010".

ENDPOINT 0 REGISTER B (EP0RB)

Read / Write

Reset Value: 1000 0000 (80h)

7							0
1	DTOG RX	STAT RX1	STAT RX0	0	0	0	0

This register is used for controlling data reception on Endpoint 0. It is also reset by the USB bus reset.

Bit 7 = Forced by hardware to 1.

Bits 6:4 = Refer to the EPnRB register for a description of these bits.

Bits 3:0 = Forced by hardware to 0.

USB INTERFACE (Cont'd)

9.3.5 Programming Considerations

The interaction between the USB interface and the application program is described below. Apart from system reset, action is always initiated by the USB interface, driven by one of the USB events associated with the Interrupt Status Register (ISTR) bits.

9.3.5.1 Initializing the Registers

At system reset, the software must initialize all registers to enable the USB interface to properly generate interrupts and DMA requests.

1. Initialize the DMAR, IDR, and IMR registers (choice of enabled interrupts, address of DMA buffers). Refer the paragraph titled initializing the DMA Buffers.
2. Initialize the EP0RA and EP0RB registers to enable accesses to address 0 and endpoint 0 to support USB enumeration. Refer to the paragraph titled Endpoint Initialization.
3. When addresses are received through this channel, update the content of the DADDR.
4. If needed, write the endpoint numbers in the EA fields in the EP1RB and EP2RB register.

9.3.5.2 Initializing DMA buffers

The DMA buffers are a contiguous zone of memory whose maximum size is 48 bytes. They can be placed anywhere in the memory space to enable the reception of messages. The 10 most significant bits of the start of this memory area are specified by bits DA15-DA6 in registers DMAR and IDR, the remaining bits are 0. The memory map is shown in Figure 27.

Each buffer is filled starting from the bottom (last 3 address bits=000) up.

9.3.5.3 Endpoint Initialization

To be ready to receive:

Set STAT_RX to VALID (11b) in EP0RB to enable reception.

To be ready to transmit:

1. Write the data in the DMA transmit buffer.
2. In register EPnRA, specify the number of bytes to be transmitted in the TBC field
3. Enable the endpoint by setting the STAT_TX bits to VALID (11b) in EPnRA.

Note: Once transmission and/or reception are enabled, registers EPnRA and/or EPnRB (respectively) must not be modified by software, as the hardware can change their value on the fly.

When the operation is completed, they can be accessed again to enable a new operation.

9.3.5.4 Interrupt Handling

Start of Frame (SOF)

The interrupt service routine may monitor the SOF events for a 1 ms synchronization event to the USB bus. This interrupt is generated at the end of a resume sequence and can also be used to detect this event.

USB Reset (RESET)

When this event occurs, the DADDR register is reset, and communication is disabled in all endpoint registers (the USB interface will not respond to any packet). Software is responsible for reenabling endpoint 0 within 10 ms of the end of reset. To do this, set the STAT_RX bits in the EP0RB register to VALID.

Suspend (SUSP)

The CPU is warned about the lack of bus activity for more than 3 ms, which is a suspend request. The software should set the USB interface to suspend mode and execute an ST7 HALT instruction to meet the USB-specified power constraints.

End Suspend (ESUSP)

The CPU is alerted by activity on the USB, which causes an ESUSP interrupt. The ST7 automatically terminates HALT mode.

Correct Transfer (CTR)

1. When this event occurs, the hardware automatically sets the STAT_TX or STAT_RX to NAK.
   Note: Every valid endpoint is NAKed until software clears the CTR bit in the ISTR register, independently of the endpoint number addressed by the transfer which generated the CTR interrupt.
   Note: If the event triggering the CTR interrupt is a SETUP transaction, both STAT_TX and STAT_RX are set to NAK.
2. Read the PIDR to obtain the token and the IDR to get the endpoint number related to the last transfer.
   Note: When a CTR interrupt occurs, the TP3-TP2 bits in the PIDR register and EP1-EP0 bits in the IDR register stay unchanged until the CTR bit in the ISTR register is cleared.
3. Clear the CTR bit in the ISTR register.

USB INTERFACE (Cont'd)

Table 14. USB Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
25	PIDR	TP3	TP2	0	0	0	RX_SEZ	RXD	0
	Reset Value	x	x	0	0	0	0	0	0
26	DMAR	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
	Reset Value	x	x	x	x	x	x	x	x
27	IDR	DA7	DA6	EP1	EP0	CNT3	CNT2	CNT1	CNT0
	Reset Value	x	x	x	x	0	0	0	0
28	ISTR	SUSP	DOVR	CTR	ERR	IOVR	ESUSP	RESET	SOF
	Reset Value	0	0	0	0	0	0	0	0
29	IMR	SUSPM	DOVRM	CTRM	ERRM	IOVRM	ESUSPM	RESETM	SOFM
	Reset Value	0	0	0	0	0	0	0	0
2A	CTLR	0	0	0	0	RESUME	PDWN	FSUSP	FRES
	Reset Value	0	0	0	0	0	1	1	0
2B	DADDR	0	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
	Reset Value	0	0	0	0	0	0	0	0
2C	EP0RA	ST_OUT	DTOG_TX	STAT_TX1	STAT_TX0	TBC3	TBC2	TBC1	TBC0
	Reset Value	0	0	0	0	x	x	x	x
2D	EP0RB	1	DTOG_RX	STAT_RX1	STAT_RX0	0	0	0	0
	Reset Value	1	0	0	0	0	0	0	0
2E	EP1RA	ST_OUT	DTOG_TX	STAT_TX1	STAT_TX0	TBC3	TBC2	TBC1	TBC0
	Reset Value	0	0	0	0	x	x	x	x
2F	EP1RB	CTRL	DTOG_RX	STAT_RX1	STAT_RX0	EA3	EA2	EA1	EA0
	Reset Value	0	0	0	0	x	x	x	x
30	EP2RA	ST_OUT	DTOG_TX	STAT_TX1	STAT_TX0	TBC3	TBC2	TBC1	TBC0
	Reset Value	0	0	0	0	x	x	x	x
31	EP2RB	CTRL	DTOG_RX	STAT_RX1	STAT_RX0	EA3	EA2	EA1	EA0
	Reset Value	0	0	0	0	x	x	x	x

10 INSTRUCTION SET

10.1 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 15. CPU Addressing Mode Overview

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

INSTRUCTION SET OVERVIEW (Cont'd)

10.1.1 Inherent

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

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

10.1.2 Immediate

Immediate instructions have 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

10.1.3 Direct

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

The direct addressing mode consists of two submodes:

Direct (short)

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

Direct (long)

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

10.1.4 Indexed (No Offset, Short, Long)

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

The indirect addressing mode consists of three 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.

10.1.5 Indirect (Short, Long)

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

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

10.1.6 Indirect Indexed (Short, Long)

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

The indirect indexed addressing mode consists of two 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 16. Instructions Supporting Direct, Indexed, Indirect and Indirect Indexed Addressing Modes

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

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

10.1.7 Relative mode (Direct, Indirect)

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

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

The relative addressing mode consists of two submodes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

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

INSTRUCTION SET OVERVIEW (Cont'd)

10.2 INSTRUCTION GROUPS

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

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

Using a 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	

11 ELECTRICAL CHARACTERISTICS

11.1 PARAMETER CONDITIONS

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

11.1.1 Minimum and Maximum Values

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

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

11.1.2 Typical Values

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

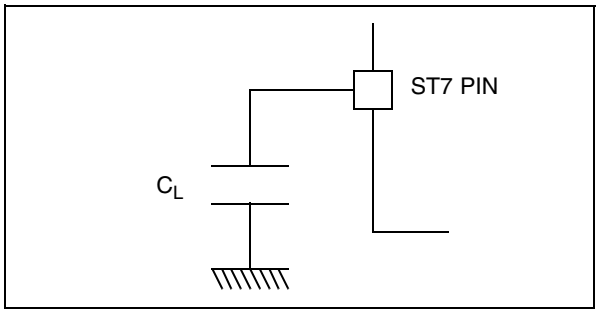
11.1.3 Typical Curves

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

11.1.4 Loading Capacitor

The loading conditions used for pin parameter measurement are shown in Figure 28.

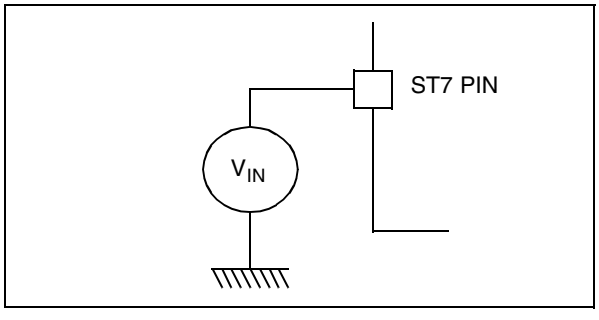
Figure 28. Pin Loading Conditions



11.1.5 Pin Input Voltage

The input voltage measurement on a pin of the device is described in Figure 29.

Figure 29. Pin Input Voltage



11.2 ABSOLUTE MAXIMUM RATINGS

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

11.2.1 Voltage Characteristics

Symbol	Ratings	Maximum value	Unit
V_{DD} - V_{SS}	Supply voltage	6.0	V
V_{IN} 1) & 2)	Input voltage on true open drain pin	V_{SS}-0.3 to 6.0	
	Input voltage on any other pin	V_{SS}-0.3 to V_{DD}+0.3	
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)	See “Absolute Maximum Ratings (Electrical Sensitivity)” on page 59.	

11.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) 3)	80	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink) 3)	80	
I_{IO}	Output current sunk by any standard I/O and control pin	25	
	Output current sunk by any high sink I/O pin	50	
	Output current source by any I/Os and control pin	- 25	
$I_{INJ(PIN)}$ 2) & 4)	Injected current on V_{PP} pin	75	
	Injected current on $\overline{\text{RESET}}$ pin	± 5	
	Injected current on OSCIN and OSCOUT pins	± 5	
	Injected current on any other pin 5) & 6)	± 5	
$\Sigma I_{INJ(PIN)}$ 2)	Total injected current (sum of all I/O and control pins) 5)	± 20	

Notes:

1. Directly connecting the $\overline{\text{RESET}}$ and I/O pins to V_{DD} or V_{SS} could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection has to be done through a pull-up or pull-down resistor (typical: 4.7kΩ for $\overline{\text{RESET}}$, 10kΩ for I/Os). Unused I/O pins must be tied in the same way to V_{DD} or V_{SS} according to their reset configuration.

2. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$.

3. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.

4. Negative injection disturbs the analog performance of the device. In particular, it induces leakage currents throughout the device including the analog inputs. To avoid undesirable effects on the analog functions, care must be taken:
- Analog input pins must have a negative injection less than 0.8 mA (assuming that the impedance of the analog voltage is lower than the specified limits)
- Pure digital pins must have a negative injection less than 1.6mA. In addition, it is recommended to inject the current as far as possible from the analog input pins.

5. When several inputs are submitted to a current injection, the maximum $\Sigma I_{INJ(PIN)}$ is the absolute sum of the positive and negative injected currents (instantaneous values). These results are based on characterisation with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

6. True open drain I/O port pins do not accept positive injection.

11.2.3 Thermal Characteristics

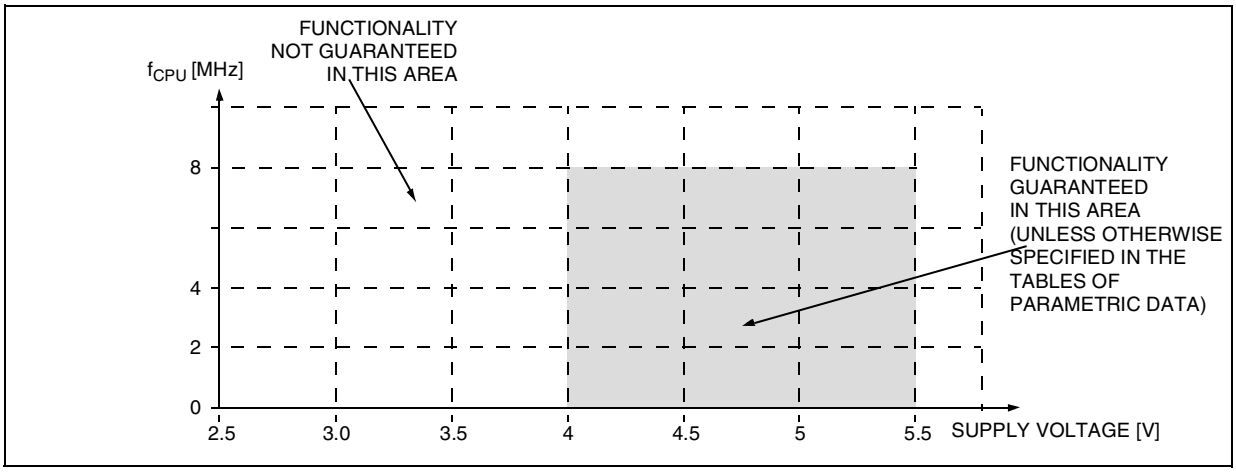
Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature	175	°C

11.3 OPERATING CONDITIONS

11.3.1 General Operating Conditions

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DD}	Operating Supply Voltage	f_{CPU} = 8 MHz	4	5	5.5	V
f_{CPU}	Operating frequency	f_{OSC} = 12MHz			8	MHz
		f_{OSC} = 6MHz			4	
T_A	Ambient temperature range		0		70	°C

Figure 30. f_{CPU} Maximum Operating Frequency Versus V_{DD} Supply Voltage



11.3.2 Operating Conditions with Low Voltage Detector (LVD)

Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A. Refer to Figure 11 on page 14.

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
V_{IT+}	Low Voltage Reset Threshold (V_{DD} rising)	V_{DD} Max. Variation 50V/ms	3.6	3.8	3.95	V
V_{IT-}	Low Voltage Reset Threshold (V_{DD} falling)	V_{DD} Max. Variation 50V/ms	3.45	3.65	3.8	V
V_{hyst}	Hysteresis (V_{IT+} - V_{IT-})		120 [2]	150 [2]	180 [2]	mV
Vt_{POR}	V_{DD} rise time rate [3]		0.5		50	V/ms

Notes:

1. Not tested, guaranteed by design.
2. Not tested in production, guaranteed by characterization.
3. The V_{DD} rise time rate condition is needed to insure a correct device power-on and LVD reset. Not tested in production.

11.4 SUPPLY CURRENT CHARACTERISTICS

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

Symbol	Parameter	Conditions		Typ [1]	Max	Unit
$\Delta I_{DD(\Delta Ta)}$	Supply current variation vs. temperature	Constant V_{DD} and f_{CPU}			10	%
I_{DD}	CPU RUN mode	I/Os in input mode. USB transceiver and LVD disabled	f_{CPU} = 4 MHz	4	6	mA
			f_{CPU} = 8 MHz	6	12	
		LVD enabled. USB in Transmission[2]	f_{CPU} = 4 MHz	10	14	mA
			f_{CPU} = 8 MHz	12	20	mA
	CPU WAIT mode[2]	I/Os in input mode. USB transceiver and LVD disabled	f_{CPU} = 8 MHz	4.5	8	mA
		LVD enabled. USB in Transmission	f_{CPU} = 8 MHz	11	18	mA
	CPU HALT mode[3]	with LVD[4]		130	200	µA
		without LVD		30	50	
	USB Suspend mode[4]			130	200	µA

Note 1: Typical data are based on T_A=25°C and not tested in production

Note 2: Data based on design simulation, not tested in production.

Note 3: USB Transceiver is powered down.

Note 4: Low voltage reset function enabled.
CPU in HALT mode.
Current consumption of external pull-up (1.5Kohms to USBVCC) and pull-down (15Kohms to V_{SSA}) not included.

Figure 31. Typ. I_{DD} in RUN at 4 and 8 MHz f_{CPU}

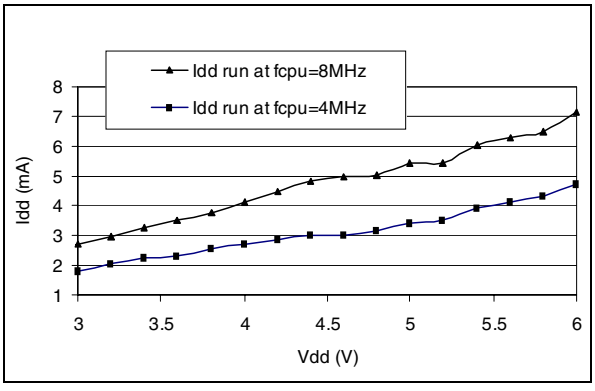
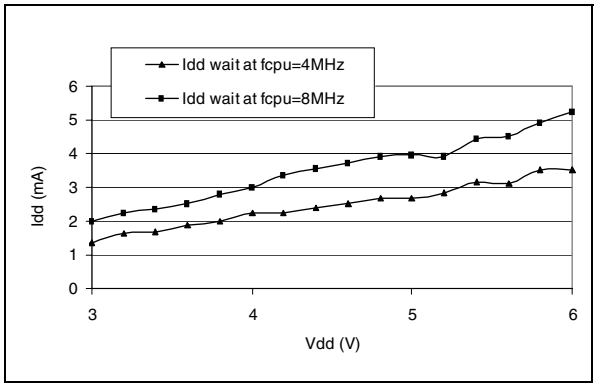


Figure 32. Typ. I_{DD} in WAIT at 4 and 8 MHz f_{CPU}



11.5 CLOCK AND TIMING CHARACTERISTICS

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

11.5.1 General Timings

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
$t_{c(INST)}$	Instruction cycle time	f_{CPU}=8MHz	2	3	12	t_{CPU}
			250	375	1500	ns
$t_{v(IT)}$	Interrupt reaction time [2] $t_{v(IT)} = \Delta t_{c(INST)} + 10\ t_{CPU}$	f_{CPU}=8MHz	10		22	t_{CPU}
			1.25		2.75	µs

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.

11.5.2 CONTROL TIMING CHARACTERISTICS

CONTROL TIMINGS						
Symbol	Parameter	Conditions	Value			Unit
			Min [1]	Typ. [1]	Max [1]	
f_{OSC}	Oscillator Frequency				12	MHz
f_{CPU}	Operating Frequency				8	MHz
t_{RL}	External RESET Input pulse Width		1.5			t_{CPU}
t_{PORL}	Internal Power Reset Duration		514			t_{CPU}
T_{RSTL}	Reset Pin Output Pulse Width		10			µs
t_{WDG}	Watchdog Time-out		65536		4194304	t_{CPU}
		f_{cpu} = 8MHz	8.192		524.288	ms
t_{OXOV}	Crystal Oscillator Start-up Time		20	30	40	ms
t_{DDR}	Power up rise time	from V_{DD} = 0 to 4V			100	ms

Note 1:
Not tested in production, guaranteed by design.

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

11.5.3 External Clock Source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSCINH}	OSCIN input pin high level voltage	see Figure 33	$0.7xV_{DD}$		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 1)		15			ns
$t_{r(OSCIN)}$ $t_{f(OSCIN)}$	OSCIN rise or fall time1)				15	
I_L	OSCx Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA

Note 1: Data based on design simulation and/or technology characteristics, not tested in production

Figure 33. Typical Application with an External Clock Source

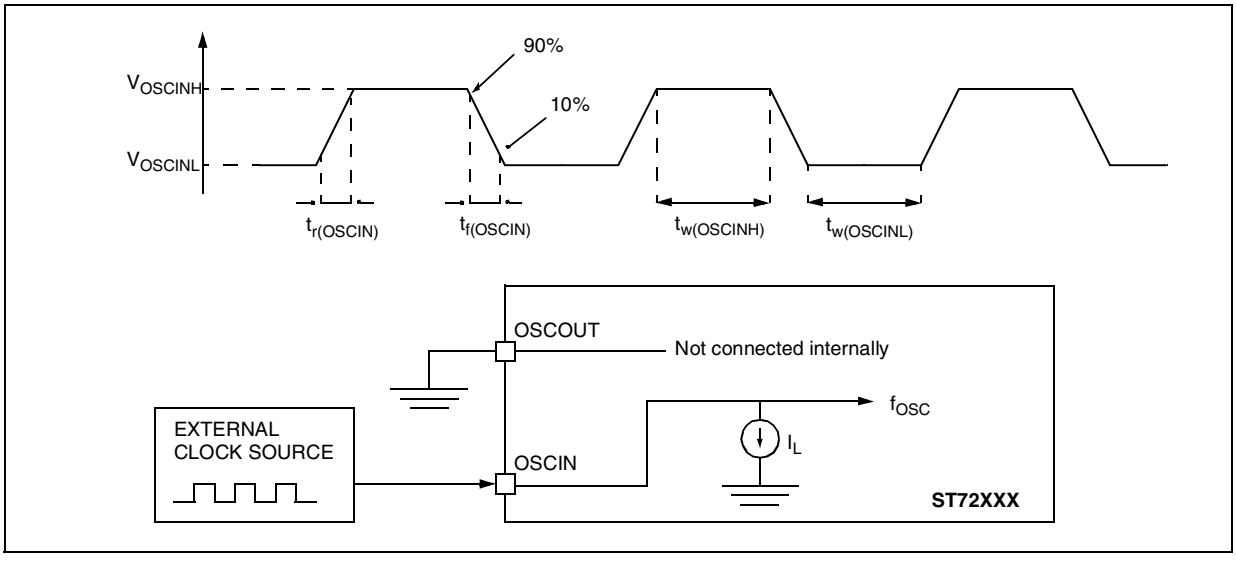
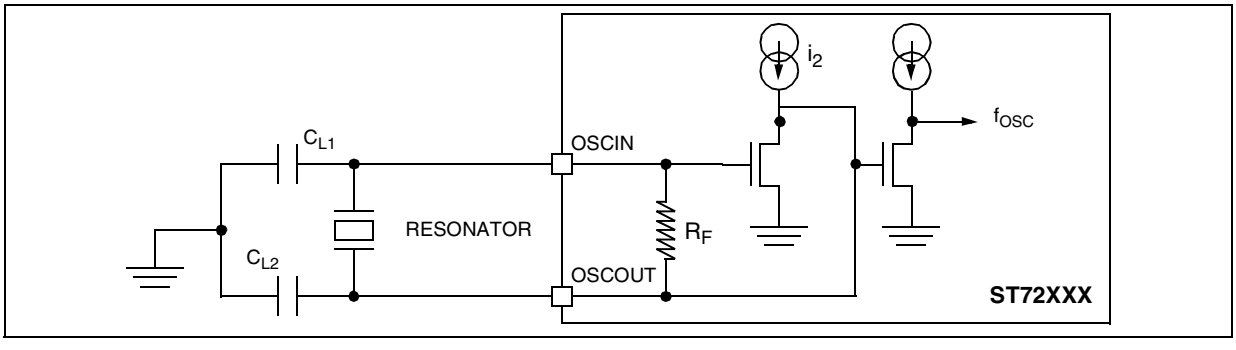


Figure 34. Typical Application with a Crystal Resonator



11.6 MEMORY CHARACTERISTICS

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

11.6.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode [1)]	HALT mode (or RESET)	2.0			V

Note 1: Guaranteed by design. Not tested in production.

11.7 EMC CHARACTERISTICS

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

11.7.1 Functional EMS (Electro Magnetic Susceptibility)

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

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

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

11.7.1.1 Designing hardened software to avoid noise problems

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

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

Software recommendations:

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

– Corrupted program counter

– Unexpected reset

– Critical Data corruption (control registers...)

Prequalification trials:

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

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

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

11.7.2 Electro Magnetic Interference (EMI)

Based on a simple application running on the product (toggling 2 LEDs through the I/O ports), the product is monitored in terms of emission. This emission test is in line with the norm SAE J 1752/3 which specifies the board and the loading of each pin.

Symbol	Parameter	Conditions	Monitored Frequency Band	Max vs. [f_{OSC}/f_{CPU}]	Unit
				12/8MHz	
S_{EMI}	Peak level	V_{DD}=5V, T_A=+25°C, DIP20, conforming to SAE J 1752/3	0.1MHz to 30MHz	33	dBµV
			30MHz to 130MHz	41	
			130MHz to 1GHz	38	
			SAE EMI Level	4.5	-

Notes:

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

EMC CHARACTERISTICS (Cont'd)

11.7.3 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.

11.7.3.1 Electro-Static Discharge (ESD)

Electro-Static Discharges (a positive then a negative pulse separated by 1 second) are applied to the pins of each sample according to each pin combination. The sample size depends on the number of supply pins in the device (3 parts*(n+1) supply pin). 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.

11.7.3.2 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).

11.8 I/O PORT PIN CHARACTERISTICS

11.8.1 General Characteristics

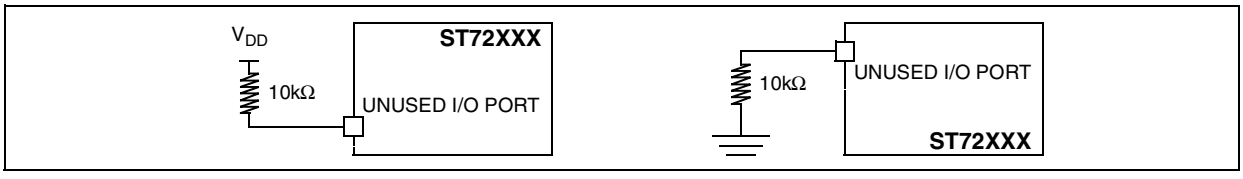
Subject to general operating conditions for V_{DD}, f_{CPU}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
V_{IL}	Input low level voltage				$0.3xV_{DD}$	V
V_{IH}	Input high level voltage		$0.7xV_{DD}$			
V_{IN}	Input voltage	True Open Drain I/O pins	V_{SS}		6.0	V
		Other I/O pins			V_{DD}	
V_{hys}	Schmitt trigger voltage hysteresis			400		mV
I_L	Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA
I_S	Static current consumption [2]	Floating input mode			200	
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time	C_L=50pF Between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time			25		
$t_{w(IT)in}$	External interrupt pulse time [3]		1			t_{CPU}

Notes:

1. Unless otherwise specified, typical data are based on T_A=25°C and V_{DD}=5V, not tested in production.
2. Configuration not recommended, all unused pins must be kept at a fixed voltage: using the output mode of the I/O for example or an external pull-up or pull-down resistor (see Figure 35). Data based on design simulation and/or technology characteristics, not tested in production.
3. To generate an external interrupt, a minimum pulse width has to be applied on an I/O port pin configured as an external interrupt source.

Figure 35. Two typical Applications with unused I/O Pin



I/O PORT PIN CHARACTERISTICS (Cont'd)

11.8.2 Output Driving Current

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

Symbol	Parameter	Conditions		Min	Max	Unit
V_{OL} [1]	Output low level voltage for a standard I/O pin when up to 8 pins are sunk at same time (see Figure 36)	V_{DD}=5V	I_{IO}=+5mA		1.3	V
			I_{IO}=+2mA		0.4	
	Output low level voltage for a high sink I/O pin when up to4 pins are sunk at same time (see Figure 37)		I_{IO}=+20mA		1.3	
			I_{IO}=+8mA		0.4	
V_{OH} [2]	Output high level voltage for an I/O pin when up to 8 pins are sourced at same time (see Figure 38)		I_{IO}=-5mA	V_{DD}-2.0		
			I_{IO}=-2mA	V_{DD}-0.8		

Figure 36. Typ. V_{OL} at V_{DD}=5V (std. port)

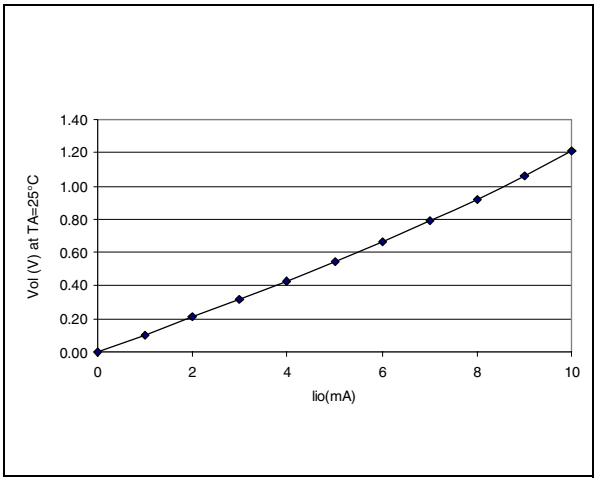


Figure 37. Typ. V_{OL} at V_{DD}=5V (high-sink)

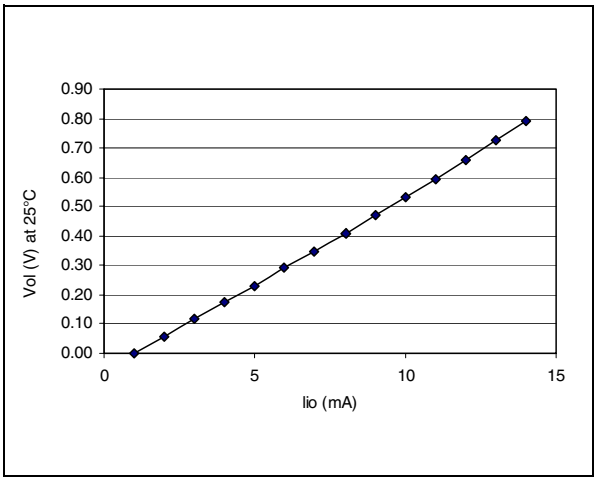


Figure 38. Typ. V_{DD}-V_{OH} at V_{DD}=5V (std. port)

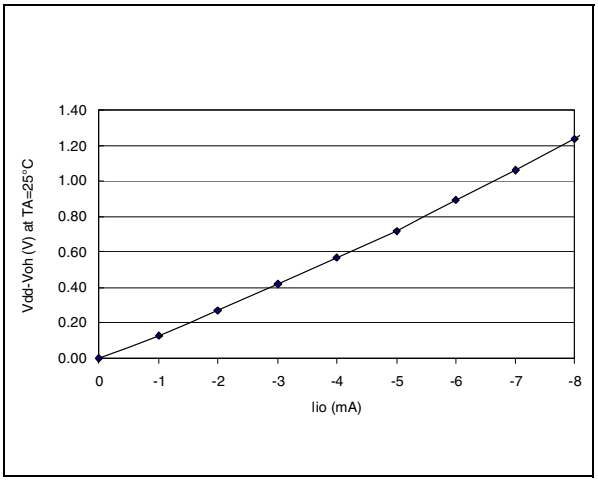
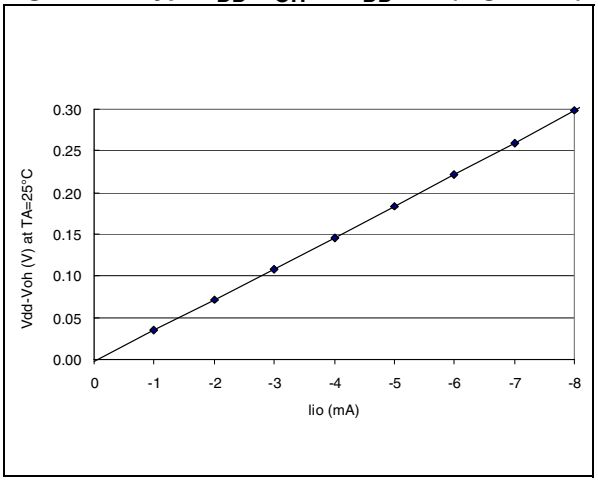


Figure 39. Typ. V_{DD}-V_{OH} at V_{DD}=5V (high-sink)



Notes:

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 11.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.

2. The I_{IO} current sourced must always respect the absolute maximum rating specified in Section 11.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}. True open drain I/O pins does not have V_{OH}.

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 40. Typical V_{OL} vs. V_{DD} (standard port)

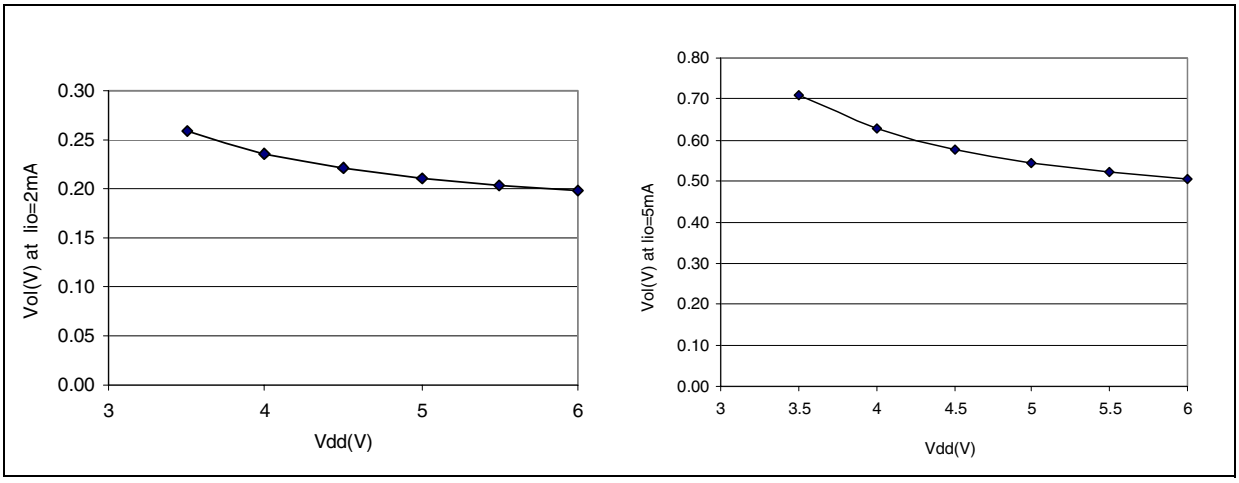


Figure 41. Typical V_{OL} vs. V_{DD} (high-sink port)

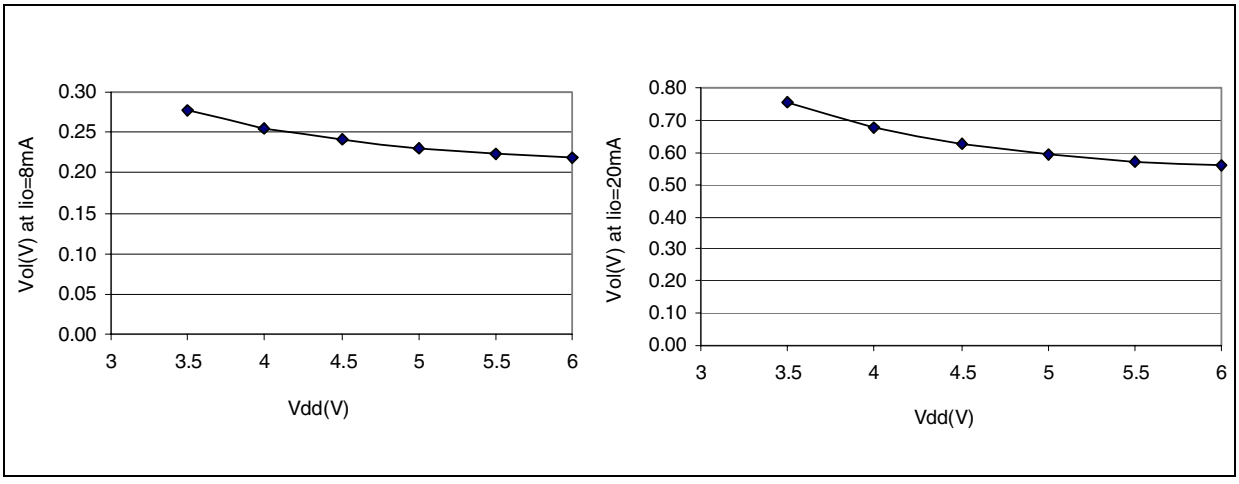
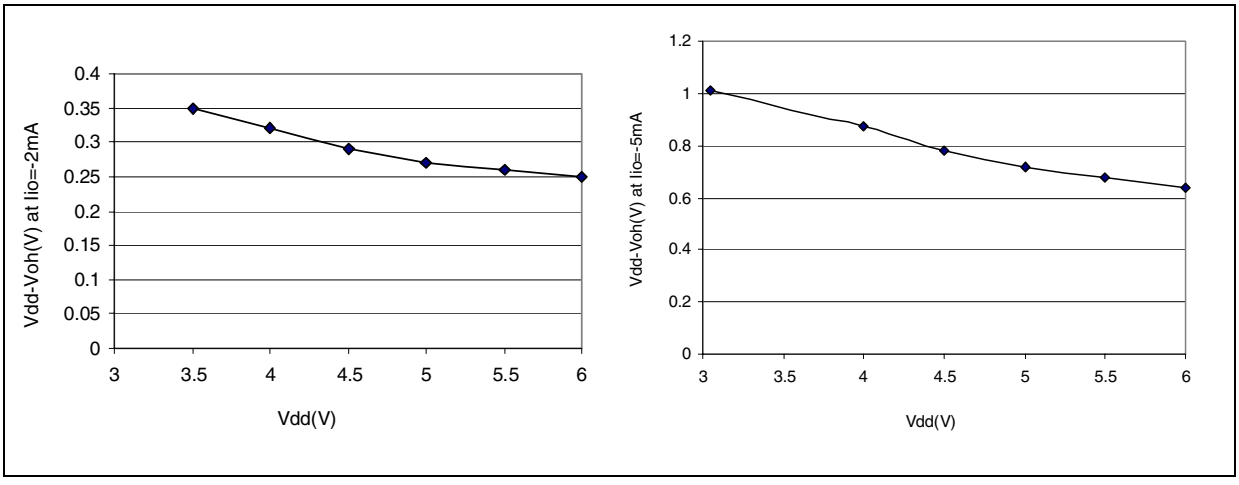
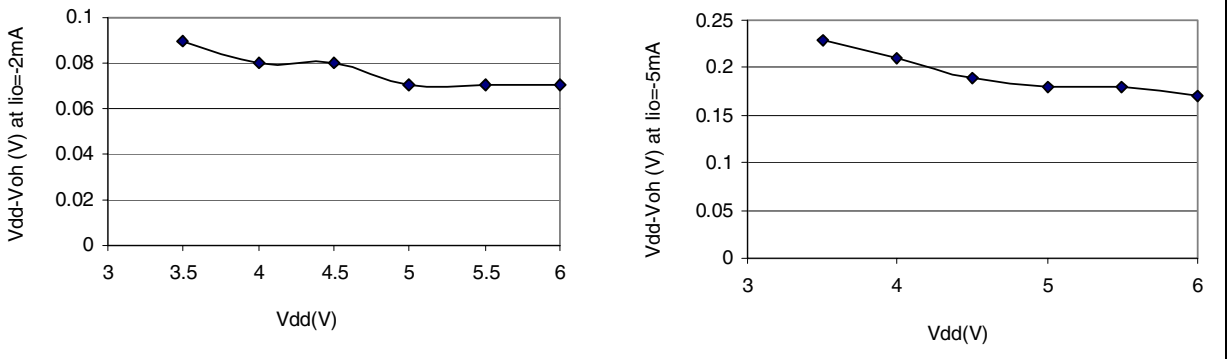


Figure 42. Typical V_{DD}-V_{OH} vs. V_{DD} (standard port)



I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 43. Typical V_{DD}-V_{OH} vs. V_{DD} (high sink port)



11.9 CONTROL PIN CHARACTERISTICS

11.9.1 Asynchronous $\overline{\text{RESET}}$ Pin

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

Symbol	Parameter	Conditions		Min	Typ [1]	Max	Unit
V_{IH}	Input High Level Voltage			0.7xV_{DD}		V_{DD}	V
V_{IL}	Input Low Voltage			V_{SS}		0.3xV_{DD}	V
V_{hys}	Schmitt trigger voltage hysteresis [3]				400		mV
V_{OL}	Output low level voltage [4] (see Figure 45, Figure 46)	V_{DD}=5V	I_{IO}=5mA			1	V
			I_{IO}=2mA			0.4	
R_{ON}	Weak pull-up equivalent resistor [5]	V_{IN}=V_{SS}		80	160	280	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	External pin or internal reset sources			6 30		1/f_{SFOSC} µs
$t_{h(RSTL)in}$	External reset pulse hold time [6]			10			µs

Notes:

1. Unless otherwise specified, typical data are based on T_A=25°C and V_{DD}=5V, not tested in production.
2. Data based on characterization results, not tested in production.
3. Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.
4. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 11.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
5. The R_{ON} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{ON} current characteristics described in Figure 44). This data is based on characterization results, not tested in production.
6. To guarantee the reset of the device, a minimum pulse has to be applied to $\overline{\text{RESET}}$ pin. All short pulses applied on $\overline{\text{RESET}}$ pin with a duration below $t_{h(RSTL)in}$ can be ignored.

CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 44. Typical I_{ON} vs. V_{DD} with $V_{IN}=V_{SS}$

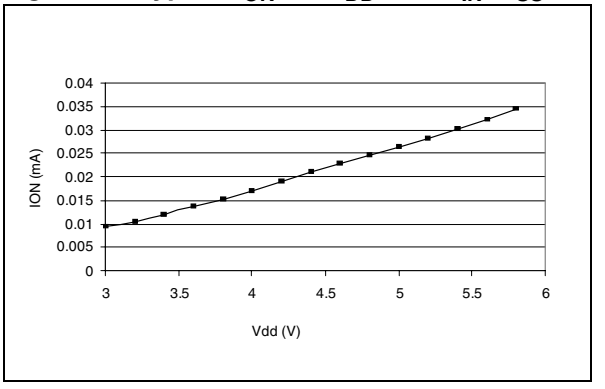


Figure 45. Typical V_{OL} at V_{DD}=5V ($\overline{RESET}$)

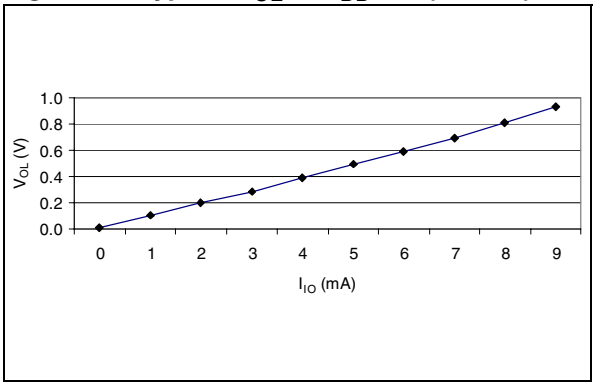
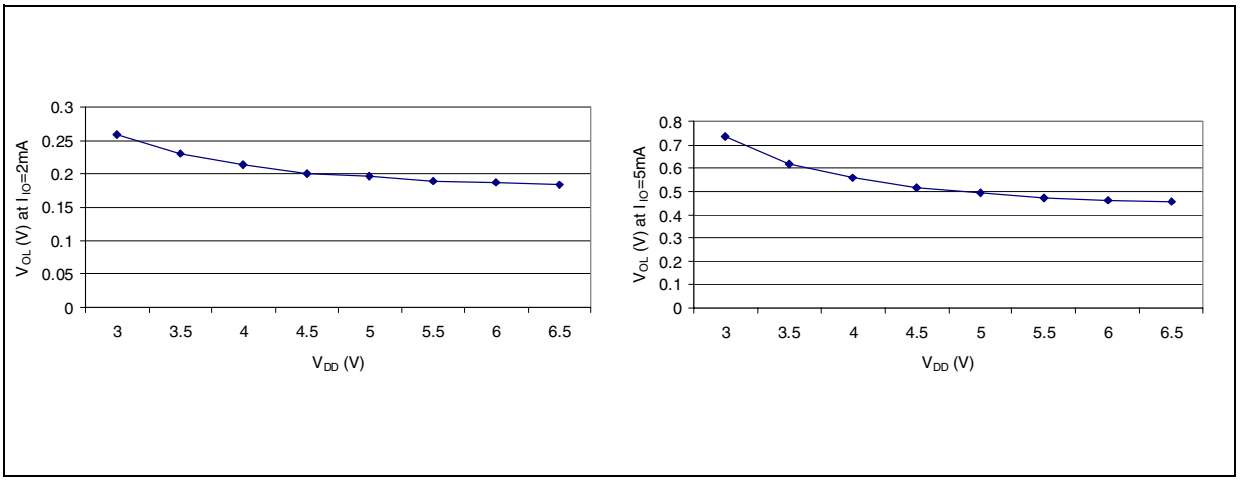


Figure 46. Typical V_{OL} vs. V_{DD} ($\overline{RESET}$)



11.10 COMMUNICATION INTERFACE CHARACTERISTICS

11.10.1 USB - Universal Bus Interface

(Operating conditions T_A = 0 to +70°C, V_{DD} = 4.0 to 5.25V unless otherwise specified)

USB DC Electrical Characteristics					
Parameter	**Symbol**	**Conditions**	**Min.**	**Max.**	**Unit**
Differential Input Sensitivity	VDI	I(D+, D-)	0.2 3)		V
Differential Common Mode Range	VCM	Includes VDI range	0.8 3)	2.5 3)	V
Single Ended Receiver Threshold	VSE		0.8 3)	2.0 3)	V
Static Output Low	VOL	RL of 1.5K ohms to 3.6v		0.3	V
Static Output High	VOH	RL of 15K ohms to V_{SS}	2.8	3.6	V
USBVCC: voltage level 4)	USBV	V_{DD}=5V	3.00	3.60	V

Note 1: RL is the load connected on the USB drivers.

Note 2: All the voltages are measured from the local ground potential.

Note 3: Not tested in production, guaranteed by design.

Note 4: To improve EMC performance (noise immunity), it is recommended to connect a 100nF capacitor to the USBVCC pin.

Figure 47. USB: Data Signal Rise and Fall Time

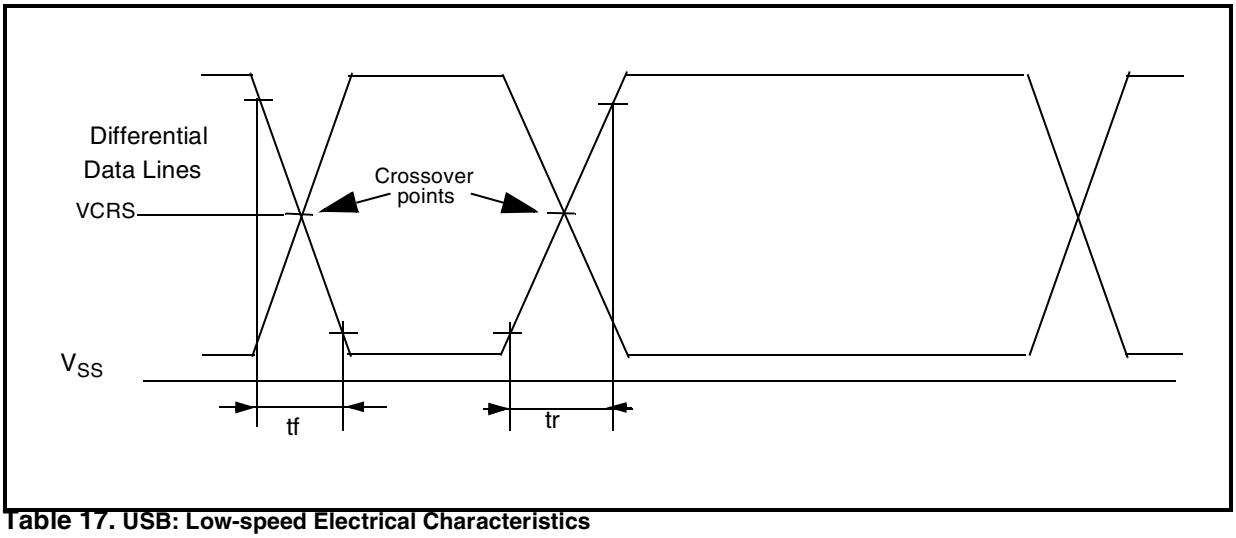


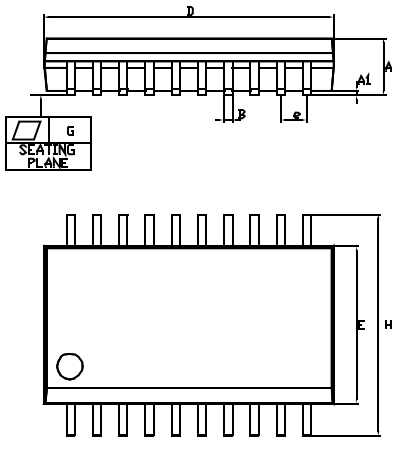
Table 17. USB: Low-speed Electrical Characteristics

Parameter	Symbol	Conditions	Min	Max	Unit
Driver characteristics:					
Rise time	tr	Note 1,CL=50 pF	75		ns
		Note 1, CL=600 pF		300	ns
Fall Time	tf	Note 1, CL=50 pF	75		ns
		Note 1, CL=600 pF		300	ns
Rise/ Fall Time matching	trfm	tr/tf	80	120	%
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).

12 PACKAGE MECHANICAL DATA

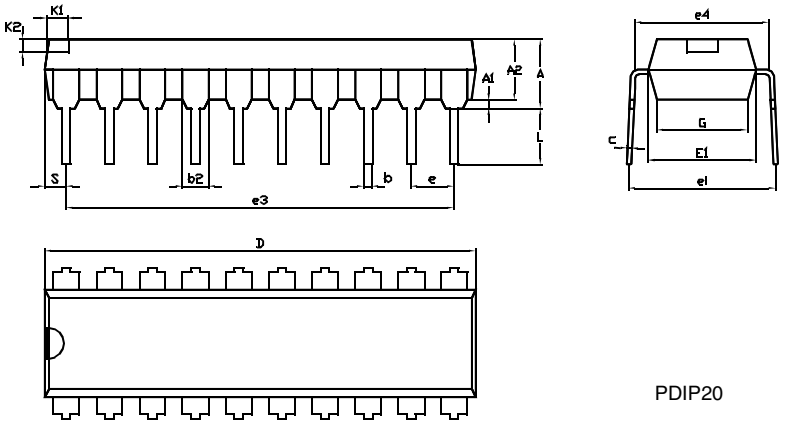
Figure 48. 20-Pin Plastic Small Outline Package, 300-mil Width



SO20

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A	2.35		2.65	0.0926		0.1043
A1	0.10			0.0040		
B	0.33		0.51	0.0130		0.0200
C			0.32			0.0125
D	4.98		13.00	0.1961		0.5118
E	7.40		7.60	0.2914		0.2992
e		1.27			0.050	
H	10.01		10.64	0.394		0.419
h	0.25		0.74	0.010		0.029
K	0°		8°	0°		8°
L	0.41		1.27	0.016		0.050
G			0.10			0.004
	Number of Pins					
N	20					

Figure 49. 20-Pin Plastic Dual In-Line Package, 300-mil Width



PDIP20

Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			5.33			0.210
A2	2.92	3.30	4.95	0.115	0.130	0.195
b	0.36	0.46	0.56	0.014	0.018	0.022
b2	1.14	1.52	1.78	0.045	0.060	0.070
c	0.20	0.25	0.36	0.008	0.010	0.014
D	24.89		26.92	0.980		1.060
e		2.54			0.100	
E1	6.10	6.35	7.11	0.240	0.250	0.280
L	2.92	3.30	3.81	0.115	0.130	0.150
	Number of Pins					
N	20					

13 DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in ROM versions. The user programmable version (FLASH) is supported by the ST72F623F2.

The ROM devices are factory-configured.

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.
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.

13.1 OPTION BYTE

The Option Byte allows the hardware configuration of the microcontroller to be selected.
The Option Byte has no address in the memory map and can be accessed only in programming mode using a standard ST7 programming tool.

OPTION BYTE

7							0
1	1	WDG SW	-	LVD	-	OSC 12/6	FMP_ R

Bit 7:6 = Reserved.

Bit 5 = **WDGSW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware enabled
1: Software enabled

Bit 4 = Reserved

Bit 3 = **LVD** *Low Voltage Detector Selection*
This option bit selects the LVD.
0: LVD enabled
1: LVD disabled

Important Note: The LVD must always be enabled (LVD bit programmed at 0).

Bit 2= Reserved.

Bit 1 = **OSC12/6** *Oscillator selection*
This option bit selects the clock divider used to drive the USB interface at 6MHz.
0: 6 MHz oscillator (no divider for USB)
1: 12 Mhz oscillator (2 divider for USB)

Bit 0 = **FMP_R** *Read out protection*
This option bit allows the protection of the software contents against piracy (program or data). When the protection is activated, read/write access is prevented by hardware. If the protection is deactivated, the memory is erased first and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual.
0: Read-out protection enabled
1: Read-out protection disabled

13.2 DEVICE ORDERING INFORMATION

Table 18. Supported part numbers

Part Number	Program Memory (Bytes)	RAM (Bytes)	Package
ST72611F1B1	4K ROM	256	PDIP20
ST72611F1M1			SO20

☐ Contact ST sales office for product availability

13.3 DEVELOPMENT TOOLS

STmicroelectronics offers a range of hardware and software development tools for the ST7 microcontroller family. Full details of tools available for the ST7 from third party manufacturers can be obtain from the STMicroelectronics Internet site:
➟ http//mcu.st.com.

Tools from these manufacturers include C compliers, emulators and gang programmers.

STMicroelectronics Tools

Three types of development tool are offered by ST see Table 19 and Table 20 for more details.

Table 19. STMicroelectronics Tools Features

	In-Circuit Emulation	Programming Capability[1)]	Software Included
ST7 Emulator	Yes, powerful emulation features including trace/ logic analyzer	No	ST7 CD ROM with: – ST7 Assembly toolchain – STVD7 powerful Source Level Debugger for Win 3.1, Win 9x and NT – C compiler demo versions – Windows Programming Tools for Win 3.1, Win 9x and NT
ST7 Programming Board	No	Yes (All packages)	

Note:
1. In-Circuit Programming (ICP) interface for FLASH devices.

Table 20. Dedicated STMicroelectronics Development Tools

Supported Products	Evaluation Board	ST7 Emulator	ST7 Programming Board	Active Probe & Target Emulation Board
ST7261	ST7MDTULS-EVAL	ST7MDTU2-EMU2B	ST7MDTU2-EPB [1)]	ST7MDTU2-DBE2B

Note:
1. Add Suffix /EU or /US for the power supply for your region.

Note: The FLASH version of the ST7261 is supported by the ST72F623F2.

ST7261 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:	4K
SDIP20:	[ ] ST72611F1B1
SO20:	[ ] ST72611F1M1
DIE FORM:	**4K**
20-pin:	[ ]

Conditioning (check only one option):

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

Special Marking:
[ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ "
Authorized characters are letters, digits, '.', '-', '/' and spaces only.
Maximum character count:
S020 (8 char. max) : _ _ _ _ _ _ _ _
DIP20 (10 char. max) : _ _ _ _ _ _ _ _ _ _

Watchdog Selection:	[ ] Software activation	[ ] Hardware activation
LVD Reset:	[ ] Disabled*	[x] Enabled *
Oscillator Selection:	[ ] 6 MHz.	[ ] 12 MHz.
Readout protection:	[ ] Enabled	[ ] Disabled

Date . . . . . .
Signature . . . . . .

* LVD must always be enabled

14 IMPORTANT NOTE

14.1 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.

14.2 USB BEHAVIOR WITH LVD DISABLED

If the LVD is disabled, the USB is disabled by hardware.

Workaround

The LVD option bit must always be programmed at 0.

14.2.1 SILICON IDENTIFICATION

This behavior is present on rev W ST72611 devices only.

They are identifiable:

On the device package, by the last letter of the Trace code marked on the device package

On the box, by the last 3 digits of the Internal Sales Type printed on the box label.

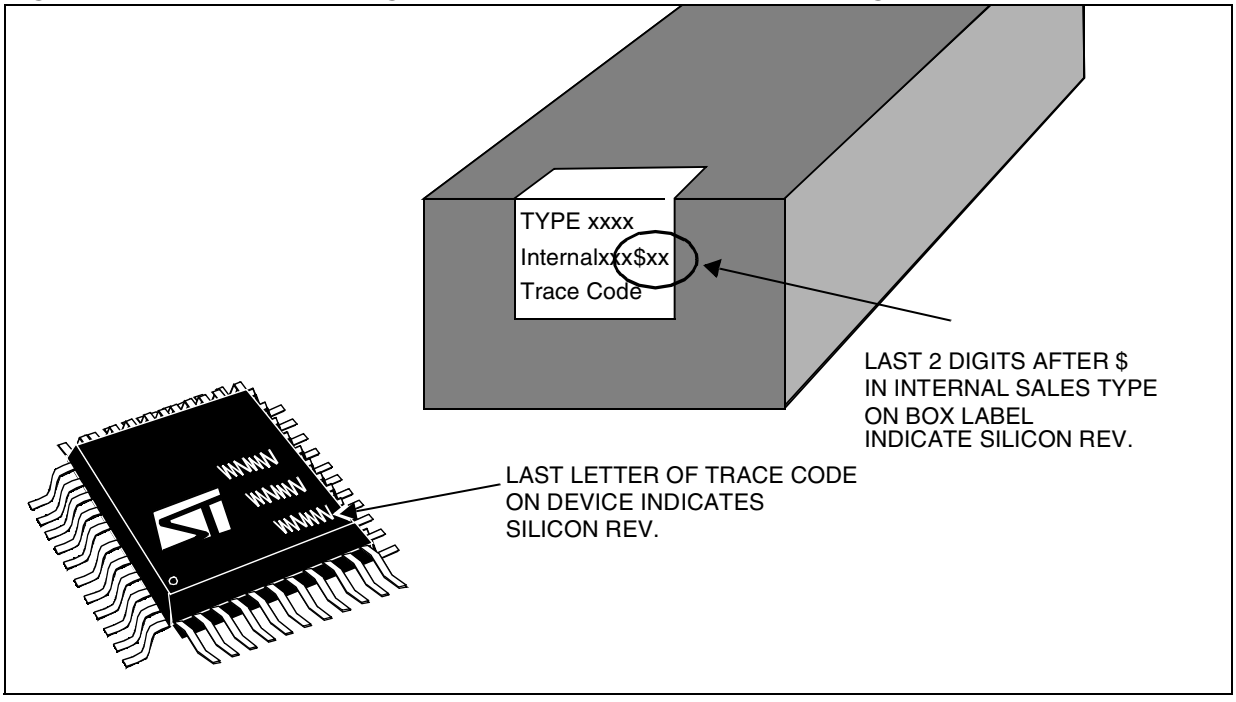
Table 21. Device Identification

Part Number	Trace Code marked on device	Internal Sales Type on box label
ST72611F1M1	“xxxxxxxxxW”	72611F1M1/xxx$U5
ST72611F1B1	“xxxxxxxxxW”	72611F1B1/xxx$U5

See also Figure 50.

IMPORTANT NOTES (Cont'd)

Figure 50. Revision Marking on Box Label and Device Marking



14.3 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 DEBUG 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

15 SUMMARY OF CHANGES

Description of the changes between the current release of the specification and the previous one.

Revision	Main changes	Date
2.2	Added “related documentation” section in specific chapters thoughout document Changed text on DMA on 1st page Added one note and changed section 5.2.1 on page 14 Changed note 2 in section 11.2 on page 52 Changed section 11.7 on page 58 Changed description of LVD bit in section 13.1 on page 67 Changed option list on page 70 Removed erratasheet (LVD Reset on V_{DD} Brownout) Added section 14.2 on page 71 **Please read carefully the Section “IMPORTANT NOTE” on page 71**	Februnary 04

Notes:

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics.

All other names are the property of their respective owners

© 2004 STMicroelectronics - All rights reserved

STMicroelectronics GROUP OF COMPANIES

Australia – Belgium - Brazil - Canada - China – Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States

www.st.com