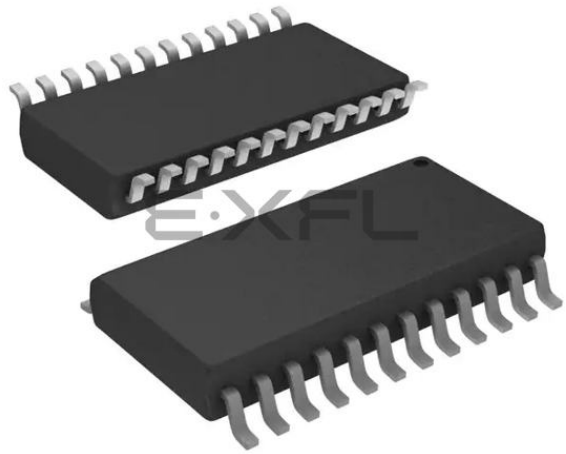


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	ST7
Core Size	8-Bit
Speed	8MHz
Connectivity	SmartCard, UART/USART, USB
Peripherals	LED, LVD, POR, WDT
Number of I/O	4
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	4V ~ 5.5V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	24-SOIC (0.295", 7.50mm Width)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/stmicroelectronics/st7fscr1e4m1

2 Pin description

Figure 2. 64-pin LQFP package pinout

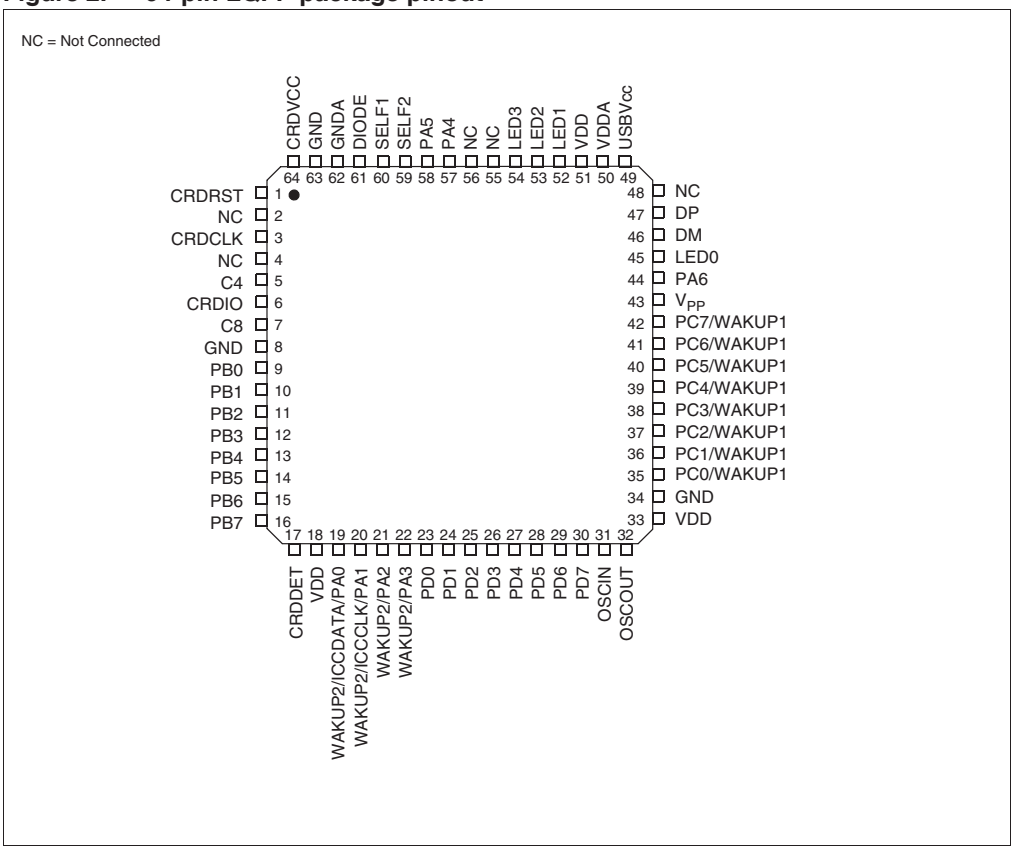


Figure 3. 24-Pin SO package pinout

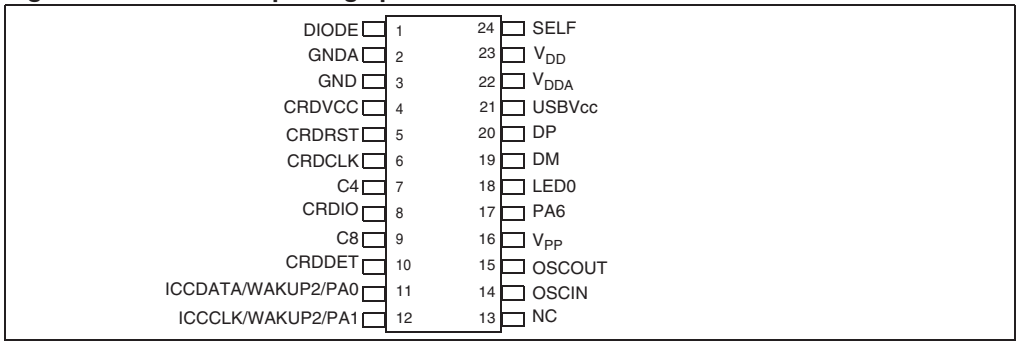


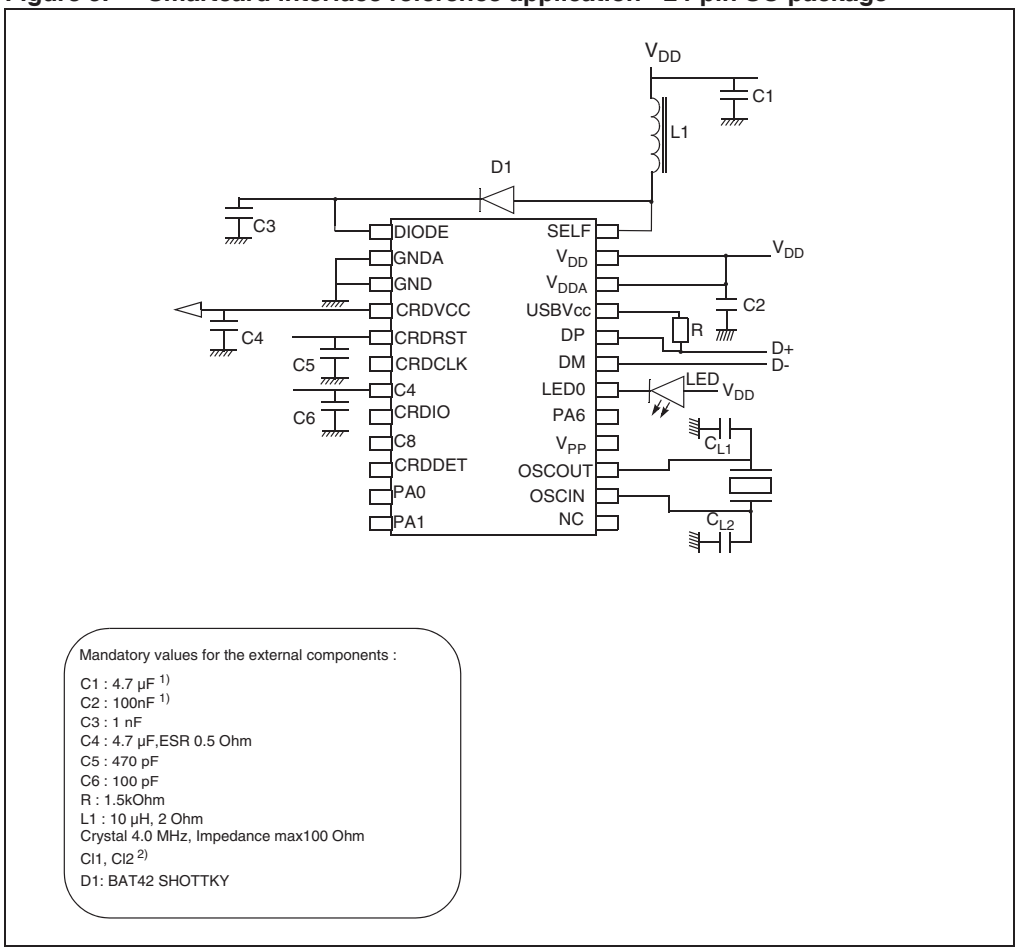
Table 3. Pin description (continued)

Pin n°			Pin name	Type	Level		V_{CARD} supplied	Port / Control				Main function (after reset)	Alternate function
								Input		Output			
LQFP64	QFN24	SO24			Input	Output		wpu	int	OD	PP		
62	23	2	GNDA	S								Ground	
63	24	3	GND	S									
64	1	4	CRDVCC	O		C_T	X					Smartcard Supply pin	

1. Keyboard interface

Note: It is mandatory to connect all available VDD and VDDA pins to the supply voltage and all VSS and VSSA pins to ground.

Figure 5. Smartcard interface reference application - 24-pin SO package



5 Central processing unit

5.1 Introduction

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

5.2 Main features

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

5.3 CPU registers

The 6 CPU registers shown in *Figure 10* 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)

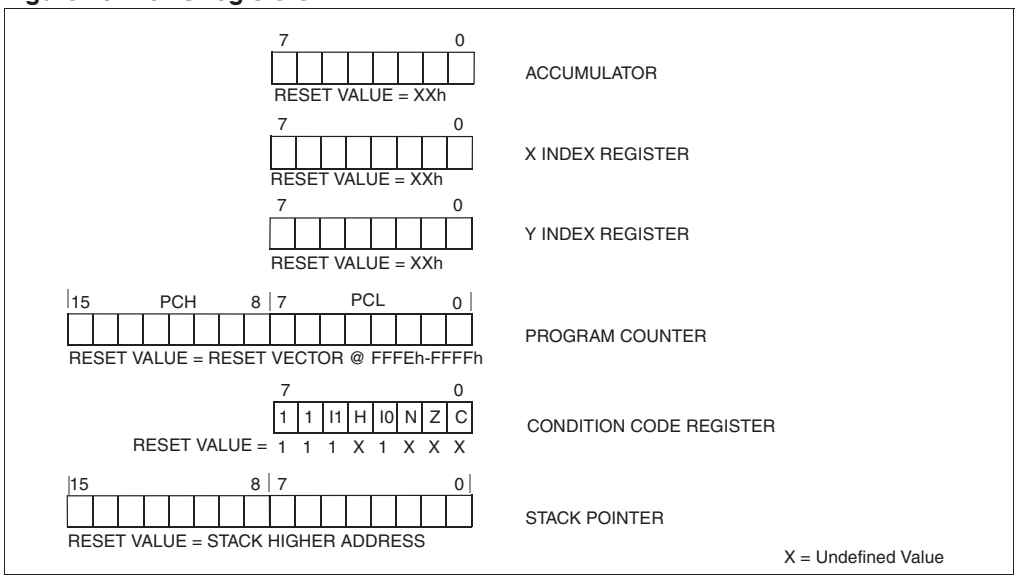
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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 10. CPU registers



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.

Stack Pointer (SP)

Read/Write

Reset Value: 017Fh

15							8
0	0	0	0	0	0	0	1
7							0
SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0

The Stack Pointer is a 16-bit register which is always pointing to the next free location in the stack. It is then decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see *Figure 11*).

6 Supply, reset and clock management

6.1 Clock system

6.1.1 General description

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

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

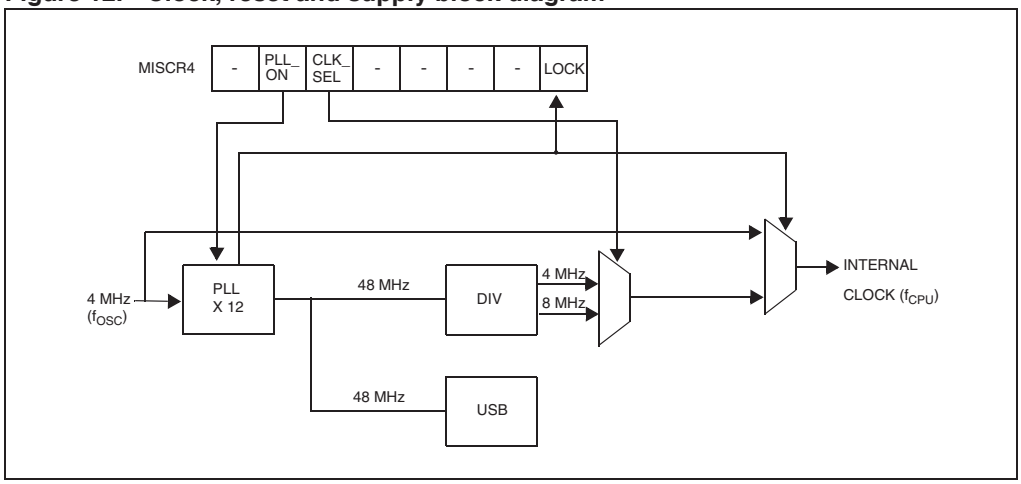
To activate the 48-MHz clock for the USB interface, the user must turn on the PLL by setting the PLL_ON bit in the MISCR4 register. When the PLL is locked, the LOCK bit is set by hardware.

The user can then select an internal frequency (f_{CPU}) of either 4 MHz or 8 MHz by programming the CLK_SEL bit in the MISCR4 register (refer to *Section 10: Miscellaneous registers*).

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

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

Figure 12. Clock, reset and supply block diagram



The internal oscillator is designed to operate with an AT-cut parallel resonant quartz in the frequency range specified for f_{osc}. The circuit shown in *Figure 14* is recommended when using a crystal, and *Table 6* lists the recommended capacitance. The crystal and associated components should be mounted as close as possible to the input pins in order to minimize output distortion and start-up stabilization time. The LOCK bit in the MISCR4 register can also be used to generate the f_{CPU} directly from f_{OSC} if the PLL and the USB interface are not active.

7.3 Interrupts and low power modes

All interrupts allow the processor to exit the WAIT low power mode. On the contrary, only external and other specified interrupts allow the processor to exit from the HALT modes (see column “Exit from HALT” in “Interrupt Mapping” table). When several pending interrupts are present while exiting HALT mode, the first one serviced can only be an interrupt with exit from HALT mode capability and it is selected through the same decision process shown in *Figure 18*.

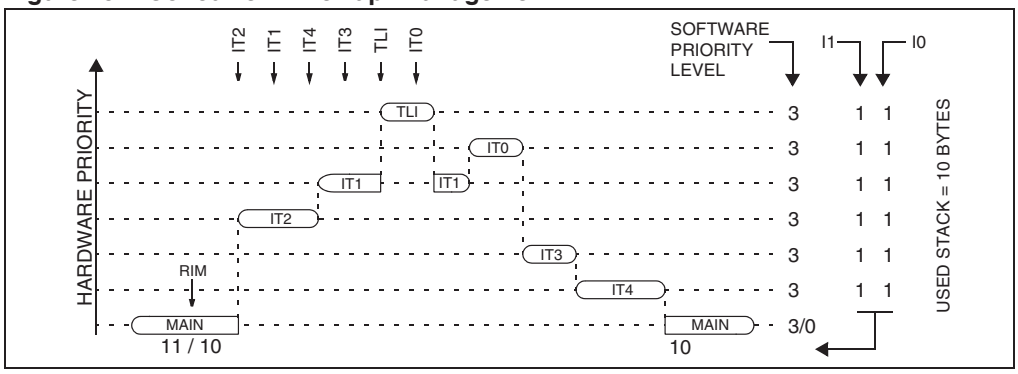
Note: If an interrupt, that is not able to Exit from HALT mode, is pending with the highest priority when exiting HALT mode, this interrupt is serviced after the first one serviced.

7.4 Concurrent and nested management

The following *Figure 19* and *Figure 20* show two different interrupt management modes. The first is called concurrent mode and does not allow an interrupt to be interrupted, unlike the nested mode in *Figure 20*. The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0, TLI. The software priority is given for each interrupt.

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

Figure 19. Concurrent interrupt management



8 Power saving modes

8.1 Introduction

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

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

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

8.2 Wait mode

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

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

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

Refer to *Figure 21*.

Read/Write

7							0
CTRL1_A	CTRL0_A	CTRL1_C	CTRL0_C	-	-	-	-

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

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

CTRL0_C, CTRL1_C: Sensitivity on port C

CTRL0_A, CTRL1_A: Sensitivity on port A

CTRL1_X	CTRL0_X	External interrupt sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

Miscellaneous register 4 (MISCR4)

Reset Value: 0000 0000 (00h).

Read/Write

7							0
-	PLL_ON	CLK_SEL	-	-	-	-	LOCK

Bit 7 = Reserved.

Bit 6 = **PLL_ON** *PLL Activation*

0: PLL disabled

1: PLL enabled

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

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

This bit is set and cleared by software.

0: CPU frequency = 4MHz

1: CPU frequency = 8MHz

Bits 4:1 = Reserved.

Bit 0 = **LOCK** *PLL status bit*

0: PLL not locked. $f_{CPU} = f_{OSC}$ external clock frequency.

1: PLL locked. f_{CPU} = 4 or 8 MHz depending on CLKSEL bit.

11 LEDs

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

LED_CTRL register

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

7							0
LD3	LD2	LD1	LD0	LD3_I	LD2_I	LD1_I	LD0_I

Bits 7:4 = **LDx** *LED Enable*

0: LED disabled

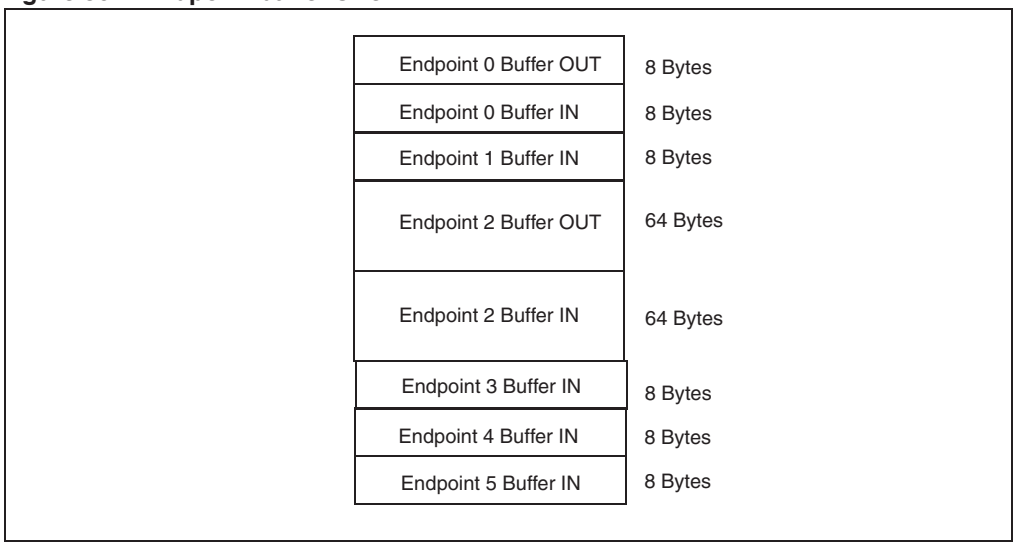
1: LED enabled

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

0: 3mA current on LDx pad

1: 7mA current on LDx pad

Figure 30. Endpoint buffer size



12.3.4 Register description

Interrupt status register (USBISTR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTR	0	SOVR	ERROR	SUSP	ESUSP	RESET	SOF

These bits cannot be set by software. When an interrupt occurs these bits are set by hardware. Software must read them to determine the interrupt type and clear them after servicing.

Note: The CTR bit (which is an OR of all the endpoint CTR flags) cannot be cleared directly, only by clearing the CTR flags in the Endpoint registers.

Bit 7 = **CTR** *Correct Transfer.*
This bit is set by hardware when a correct transfer operation is performed. This bit is an OR of all CTR flags (CTR0 in the EP0R register and CTR_RX and CTR_TX in the EPnRXR and EPnTXR registers). By looking in the USBSR register, the type of transfer can be determined from the PID[1:0] bits for Endpoint 0. For the other Endpoints, the Endpoint number on which the transfer was made is identified by the EP[1:0] bits and the type of transfer by the IN/OUT bit.

0: No Correct Transfer detected

1: Correct Transfer detected

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

Bit 0 = **SOF** *Start of frame.*
This bit is set by hardware when a SOF token is received on the USB.

0: No SOF received

1: SOF received

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

Interrupt mask register (USBIMR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTRM	0	SOVRM	ERRM	SUSPM	ESUSPM	RESETM	SOFM

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

Control register (USBCTLR)

Read/Write

Reset value: 0000 0110 (06h)

7							0
RSM	USB_RST	0	0	RESUME	PDWN	FSUSP	FRES

Bit 7 = **RSM** *Resume Detected*
This bit shows when a resume sequence has started on the USB port, requesting the USB interface to wake-up from suspend state. It can be used to determine the cause of an ESUSP event.

0: No resume sequence detected on USB

1: Resume sequence detected on USB

Bit 6 = **USB_RST** *USB Reset detected*.
This bit shows that a reset sequence has started on the USB. It can be used to determine the cause of an ESUSP event (Reset sequence).

0: No reset sequence detected on USB

1: Reset sequence detected on USB

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

Table 23. USB register map and reset values (continued)

Address (Hex.)	Register name	7	6	5	4	3	2	1	0
2C	EP2TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
2D	CNT2TXR Reset Value	0	CNT6 0	CNT5 0	CNT4 0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
2E	EP3TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
2F	CNT3TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
30	EP4TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
31	CNT4TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
32	EP5TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
33	CNT5TXR	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
34	ERRSR	0	0	0	0	0	ERR2 0	ERR1 0	ERR0 0

12.4 Smartcard interface (CRD)

12.4.1 Introduction

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

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

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

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

The power-off sequence is managed by the CRD.

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

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

Then, after transmission of this last character, signalled by the TXC interrupt, software must write the CWT value (Character Waiting Time) in the CRDWT registers. See example in *Figure 33*.

Manual mode

The load conditions are:

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

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

Interrupt generator

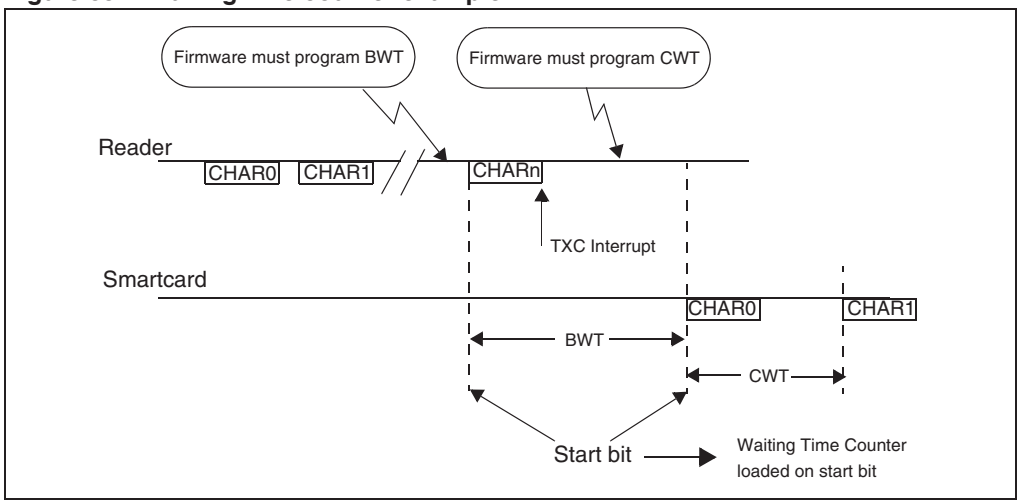
The Smartcard Interface has 2 interrupt vectors:

- Card Insertion/Removal Interrupt
- CRD Interrupt

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

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

Figure 33. Waiting time counter example



Card detection mechanism

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

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

1: Compensation mode enabled. To allow non integer value, one clock cycle is subtracted from the ETU value on odd bits. See *Figure 32*.

Bit [6:3] = Reserved

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

Writing CRDETU1 register reloads the ETU counter.

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

7							0
ETU7	ETU6	ETU5	ETU4	ETU3	ETU2	ETU1	ETU0

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

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

Guardtime register (CRDGTx)

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

7							0
0	0	0	0	0	0	0	GT8

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

7							0
GT7	GT6	GT5	GT4	GT3	GT2	GT1	GT0

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

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

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

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

This bit is set and cleared by software to enable or disable the TXBE interrupt.

0: TXBE interrupt disabled

1: TXBE interrupt enabled

Bit 6 = Reserved.

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

0: IOVF interrupt disabled

1: IOVF interrupt enabled

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

This bit is set and cleared by software to enable or disable the VCRD interrupt.

0: VCRD interrupt disabled

1: VCRD interrupt enabled

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

0: WT interrupt disabled

1: WT interrupt enabled

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

0: TXC interrupt disabled

1: TXC interrupt enabled

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

0: RXC interrupt disabled

1: RXC interrupt enabled

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

0: PAR interrupt disabled

1: PAR error interrupt enabled

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.

13.1.6 Indirect indexed (Short, Long)

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

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

Indirect indexed (Short)

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

Indirect indexed (Long)

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

Table 26. Instructions supporting direct, indexed, indirect and indirect indexed addressing modes

Long and short instructions	Function
LD	Load
CP	Compare
AND, OR, XOR	Logical Operations
ADC, ADD, SUB, SBC	Arithmetic Additions/Subtractions operations
BCP	Bit Compare

Short instructions only	Function
CLR	Clear
INC, DEC	Increment/Decrement
TNZ	Test Negative or Zero
CPL, NEG	1 or 2 Complement
BSET, BRES	Bit Operations

Table 27. Instruction set overview (continued)

Mnemo	Description	Function/ Example	Dst	Src	I1	H	I0	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 >								

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.

Table 37. 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

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

14.8 Communication interface characteristics

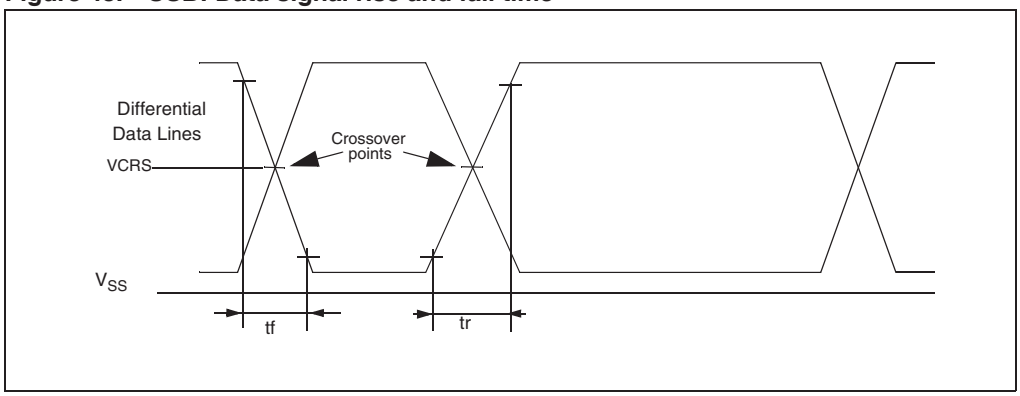
14.8.1 USB - Universal bus interface

Table 38. USB DC electrical characteristics

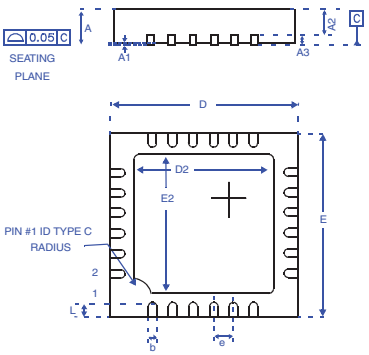
Parameter	Symbol	Conditions	Min.	Max.	Unit
Input Levels:					
Differential Input Sensitivity	VDI	I(D+, D-)	0.2		V
Differential Common Mode Range	VCM	Includes VDI range	0.8	2.5	V
Single Ended Receiver Threshold	VSE		1.3	2.0	V
Output Levels					
Static Output Low	VOL	RL of 1.5K ohms to 3.6v		0.3	V
Static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

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

Figure 43. USB: Data signal rise and fall time



15.2 Recommended reflow oven profile



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A	0.80	0.90	1.00	0.031	0.035	0.039
A1		0.02	0.05		0.001	0.002
A3		0.20			0.008	
b	0.25	0.30	0.35	0.010	0.012	0.014
D		5.00			0.197	
D2	3.50	3.60	3.70	0.138	0.142	0.146
E		5.00			0.197	
E2	3.50	3.60	3.70	0.138	0.142	0.146
e		0.65			0.026	
L	0.35	0.45	0.55	0.014	0.018	0.022
ddd		0.08			0.003	
	Number of Pins					
N	24					

1. Values in inches are converted from mm and rounded to 3 decimal digits.

Refer to JEDEC specification JSTD020D for a description of the recommended reflow oven profile for these packages.