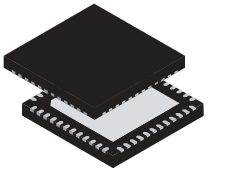
ST7WIND21

8-bit low-speed USB MCU with 3 EPs, ROM, timers, dual 27 MHz RF receiver, dual PS/2

Features

- **Memories**
  - 24K bytes ROM program memory with read/ write protection
  - 2.5K bytes RAM
- **Clock, Reset and Supply Management**
  - Dual Clock Source Management: 12 MHz monocrystal oscillator 6 MHz internal RC oscillator
  - Enhanced Reset System
  - 4 power saving modes: Slow, Wait, Halt and Auto Wake-Up From Halt
- **Interrupt Management**
  - Nested Interrupt Controller
  - 14 interrupt vectors plus TRAP and RESET
- **I/O ports**
  - 15 multifunctional bidirectional I/O lines
  - 7 with high sink capability
- **3 Timers**
  - Configurable Watchdog Timer
  - 8-bit Time Base Unit (TBU)
  - 16-bit Timer with: 2 input captures, 2 output compares, external clock input, PWM and pulse generator modes
- **Dual RF Receiver Interface**
  - 27 MHz radio frequency band
  - Demodulator
  - Delta Sigma A/D with digital filtering
  - Compatible with NRZ data



VFQFN48 (7x7)

- **USB (Universal Serial Bus) Interface**
  - DMA for low-speed applications compliant with USB 1.5 Mbs specification (v 2.0) and USB HID specification (v 1.1):
  - Integrated transceivers
  - Suspend and Resume operations
  - 3 Endpoints
  - PS/2-compatible I/Os
- **Other Communication Interfaces**
  - Dual PS/2 interface
  - SPI synchronous serial interface
- **Instruction Set**
  - 8-bit data manipulation
  - 63 basic instructions with illegal opcode detection
  - 17 main addressing modes
  - 8 x 8 unsigned multiply instruction
- **Development Tools**
  - Full hardware/software development package
  - DM (Debug module)

Table 1. Device summary

Features	
Program memory - bytes	24K ROM
RAM (stack) - bytes	2560 (256)
RF Interface	27 MHz frequency band, Dual receiver, NRZ data compatible
Peripherals	USB (3 Endpoints), Watchdog, TBU, 16-bit Timer, SPI, 2 x PS/2
Operating Supply	4.0V to 5.5V
CPU Frequency	6MHz or 12MHz
Operating temperature	0°C to +55°C
Package	QFN48 7x7/LQFP100 (EMUCHIP)

Table of Contents

Table of Contents

1 INTRODUCTION

The ST7WInD21 device, subsequently referred to as “the Device” throughout this datasheet, is a member of the ST7 microcontroller family. It is based on a common industry-standard 8-bit core, featuring an enhanced instruction set.

It operates with either 12MHz external oscillator or a 6MHz internal RC. The main clock can be switched dynamically to use either source.

Under software control, the Device can be placed in Slow, Wait, Auto Wake-up from Halt or Halt mode, reducing power consumption when the application is in idle or stand-by state.

The enhanced instruction set and addressing modes of the CPU 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, the ST7 CPU features 8x8 unsigned multiplication and indirect addressing modes.

The Device features a fully integrated RF communication block composed of a dual receiver (2RX). Each receiver can operate simultaneously in the 27MHz band covering worldwide usage. The RF communication block is designed to support NRZ coded data and to operate with a minimum of low-cost external components, including a crystal, capacitors and resistors.

Figure 56 on page 92 shows the RF Analog block diagram as well as the recommended external component circuitry. This power supply decoupling structure ensures opLtimum radio characteristics.

The Device includes an ST7 CPU, I/O lines and various peripherals as listed below:

– Embedded 3.3V voltage regulator for generating the 3.3V power supply (V_{33})

– Enhanced Reset System ensuring proper power-on or power-off of the Device

– Configurable Watchdog Timer

– Time Base Unit Timer

– 16-bit Timer

– Dual RF Receiver

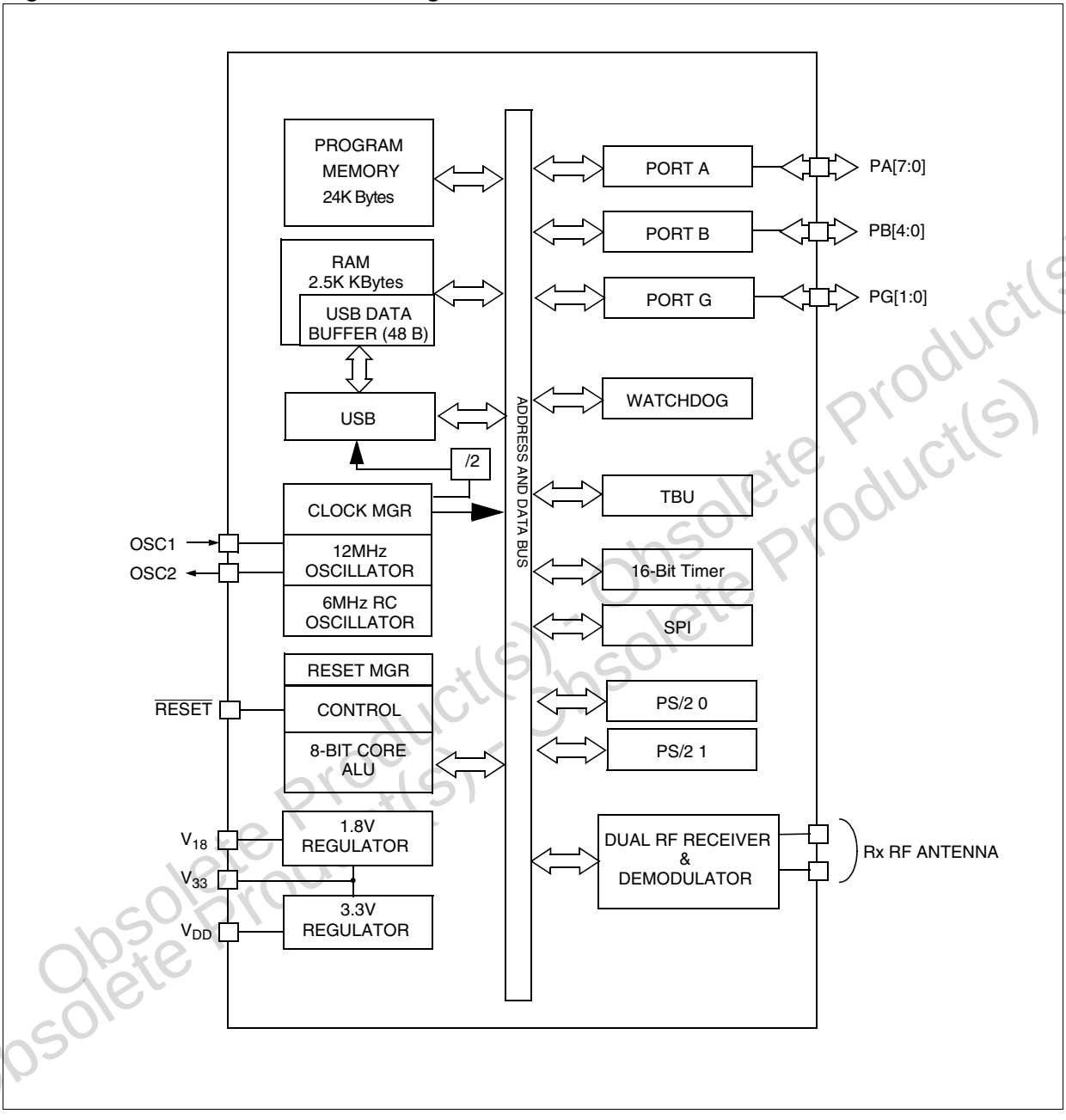
– USB low-speed interface with 3 endpoints, programmable in/out configuration and embedded transceivers (no external components are needed).

– Dual PS/2 interface

– Serial Peripheral Interface

INTRODUCTION (Cont'd)

Figure 1. ST7WIND21 Device Block Diagram



2 PIN DESCRIPTION

Figure 2. 48-pin QFN Package Pinout

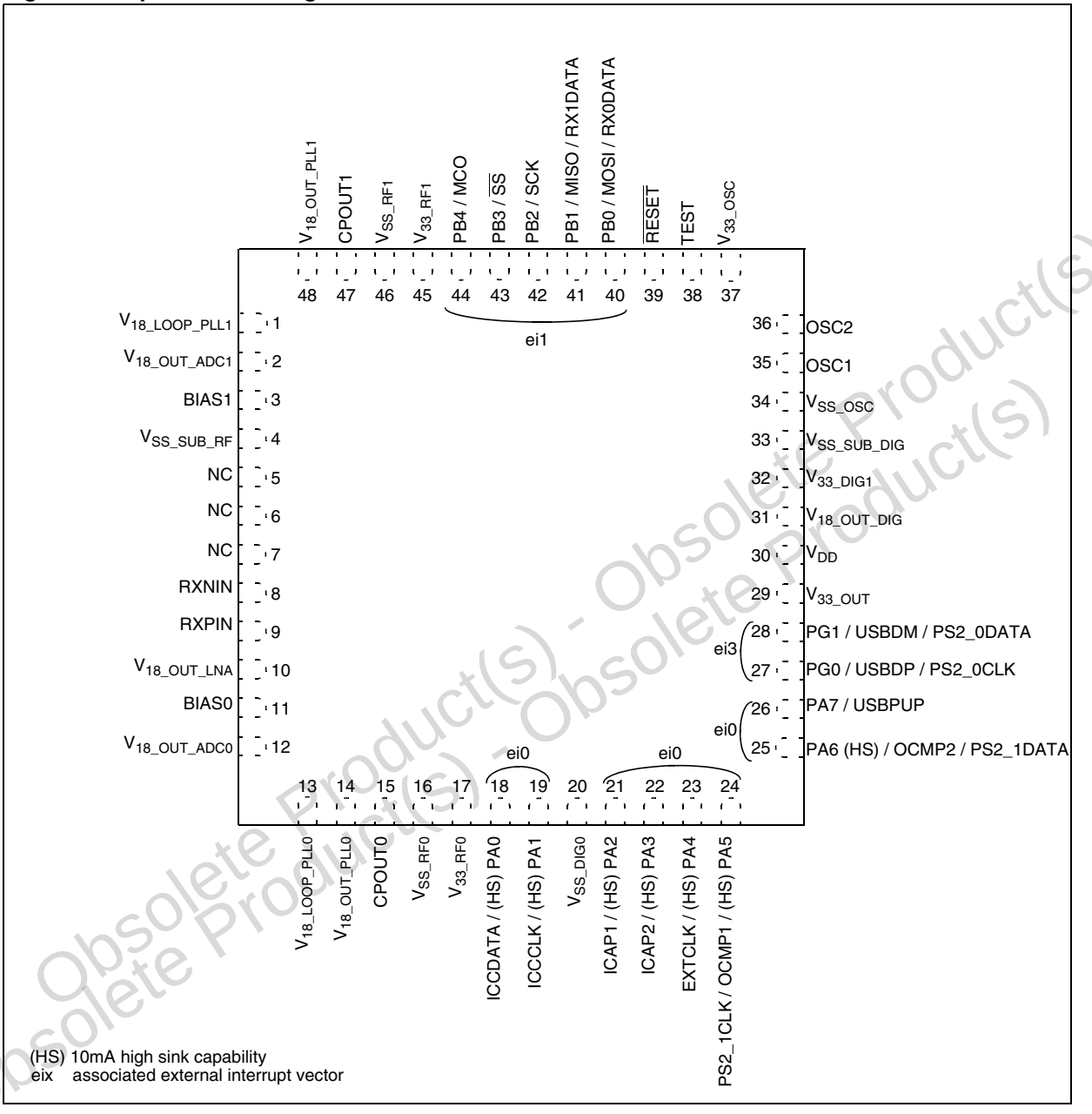
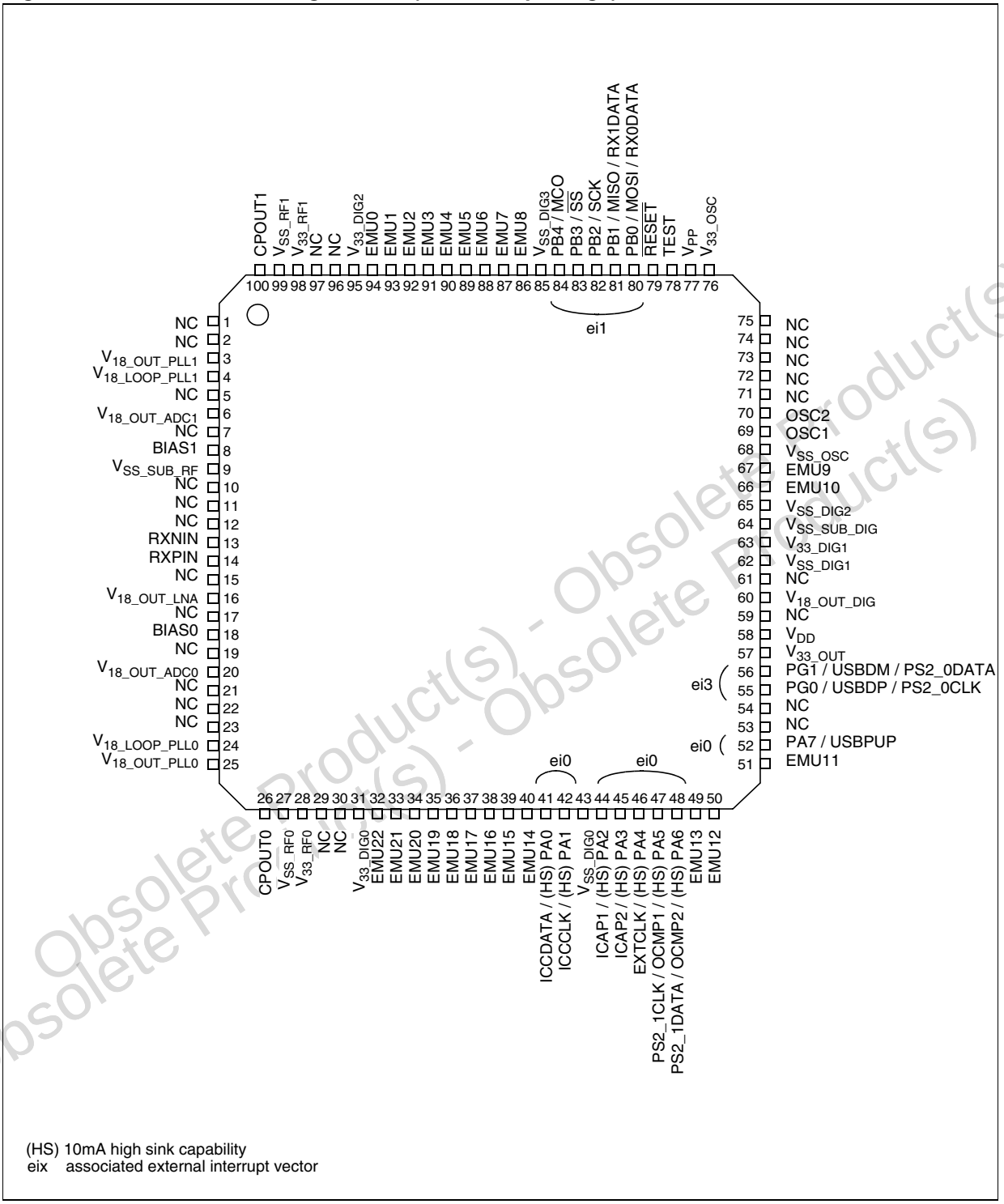


Figure 3. 100-Pin LQFP Package Pinout (EMUCHIP package)



(HS) 10mA high sink capability
eix associated external interrupt vector

Note: For further information on the emulator, please refer to the ST7WInD probe user guide.

PIN DESCRIPTION (Cont'd)

For supply pin connection guidelines, refer to the SUPPLY MANAGEMENT Section.

Legend / Abbreviations for Table 2:

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

In/Output level: C_T = CMOS $0.3V_{33}/0.7V_{33}$ with input trigger

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

Port and control configuration:

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

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

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

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

Table 2. Device Pin Description

Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Functions
						Input				Output			
LQFP100	QFN48			Input	Output	float	wpu	int[2]	ana	OD	PP		
1		NC										Not Connected	
2		NC										Not Connected	
3	48	$V_{18\_OUT\_PLL1}$	S									PLL1 1.8V Voltage Regulator Output	
4	1	$V_{18\_LOOP\_PLL1}$	S									PLL1 Filter Supply	
5		NC										Not Connected	
6	2	$V_{18\_OUT\_ADC1}$	S									ADC1 1.8V Voltage Regulator Output	
7		NC										Not Connected	
8	3	BIAS1	I									RX1 Bias Resistor	
9	4	$V_{SS\_SUB\_RF}$	S									RF Substrate Ground	
10	5	NC	O									Not Connected	
11	6	NC	O									Not Connected	
12	7	NC	O									Not Connected	
13	8	RXNIN	I									RX Antenna Negative Input	
14	9	RXPIN	I									RX Antenna Positive Input	
15		NC										Not Connected	
16	10	$V_{18\_OUT\_LNA}$	S									Low Noise Amplifier 1.8V Voltage Regulator Output	
17		NC										Not Connected	
18	11	BIAS0	I									RX0 Bias Resistor	
19		NC										Not Connected	
20	12	$V_{18\_OUT\_ADC0}$	S									ADC0 1.8V Voltage Regulator Output	
21		NC										Not Connected	
22		NC										Not Connected	
23		NC										Not Connected	
24	13	$V_{18\_LOOP\_PLL0}$	S									PLL0 Filter Supply	

Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Functions	
						Input				Output				
LQFP100	QFN48			Input	Output	float	wpu	int[2]	ana	OD	PP			
25	14	$V_{18\_OUT\_PLL0}$	S									PLL0 1.8V Voltage Regulator Output		
26	15	CPOUT0	I									PLL0 Filter		
27	16	V_{SS_RF0}	S									RF0 Common (PLL, ADC and LNA) Ground		
28	17	V_{33_RF0}	S									RF0 3.3V Power Supply		
29		NC										Not Connected		
30		NC										Not Connected		
31		V_{33_DIG0}	S									Digital 3.3V Power Supply		
32		EMU22	I/O	C_T								Emulator interface 22		
33		EMU21	I/O	C_T								Emulator interface 21		
34		EMU20	I/O	C_T								Emulator interface 20		
35		EMU19	I/O	C_T								Emulator interface 19		
36		EMU18	I/O	C_T								Emulator interface 18		
37		EMU17	I/O	C_T								Emulator interface 17		
38		EMU16	I/O	C_T								Emulator interface 16		
39		EMU15	I/O	C_T								Emulator interface 15		
40		EMU14	I/O	C_T								Emulator interface 14		
41	18	PA0/ICCDATA [6]	I/O	C_T	HS	**X**	X	ei0		X	X	Port A0	ICC Data	
42	19	PA1/ICCCLK [3][6]	I/O	C_T	HS	X	**X**	ei0		X	X	Port A1	ICC Clock	
43	20	V_{SS_DIG0}	S									Digital Ground		
44	21	PA2/ICAP1 [6]	I/O	C_T	HS	**X**	X	ei0		X	X	Port A2	Timer Input Capture 1	
45	22	PA3/ICAP2 [6]	I/O	C_T	HS	**X**	X	ei0		X	X	Port A3	Timer Input Capture 2	
46	23	PA4/EXTCLK [6]	I/O	C_T	HS	**X**	X	ei0		X	X	Port A4	Timer External Clock Source	
47	24	PA5/PS2_1CLK/ OCMP1 [6]	I/O	C_T	HS	**X**	X[4]	ei0		X	X	Port A5	PS2 1 Clock	Timer Output Compare 1
48	25	PA6/ PS2_1DATA/ OCMP2 [6]	I/O	C_T	HS	**X**	X[4]	ei0		X	X	Port A6	PS2 1 Data	Timer Output Compare 2
49		EMU13	I/O	C_T								Emulator interface 13		
50		EMU12	I/O	C_T								Emulator interface 12		
51		EMU11	I/O	C_T								Emulator interface 11		
52	26	PA7/USBPUP [6]	I/O	C_T		**X**	X[5]	ei0		X	X	Port A7	USB Pull-up	
53		NC										Not Connected		
54		NC										Not Connected		
55	27	PG0/USBDP/ PS2_0CLK [6]	I/O	C_T		**X**	X[4]	ei3		X		Port G0	PS2 0 Clock	USB bidirectional data (data +)

Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Functions	
						Input				Output				
LQFP100	QFN48			Input	Output	float	wpu	int[2]	ana	OD	PP			
56	28	PG1/USBDM/ PS2_0DATA [6]	I/O	C_T		**X**	X[4]	ei3		X		Port G1	PS2 0 Data	USB bidirectional data (data -)
57	29	V_{33_OUT}	S									3.3V Voltage Regulator Output		
58	30	V_{DD}	S									+5V Power Supply		
59		NC										Not Connected		
60	31	$V_{18\_OUT\_DIG}$	S									Digital 1.8V Voltage Regulator Output		
61		NC										Not Connected		
62		V_{SS_DIG1}	S									Digital Ground		
63	32	V_{33_DIG1}	S									Digital 3.3V Power Supply		
64	33	$V_{SS\_SUB\_DIG}$	S									Digital Substrate Ground		
65		V_{SS_DIG2}	S									Digital Ground		
66		EMU10	I/O	C_T								Emulator interface 10		
67		EMU9	I/O	C_T								Emulator interface 9		
68	34	V_{SS_OSC}	S									Oscillator Ground		
69	35	OSC1	I									External clock input or oscillator inverter input		
70	36	OSC2	O									Oscillator inverter output		
71		NC										Not Connected		
72		NC										Not Connected		
73		NC										Not Connected		
74		NC										Not Connected		
75		NC										Not Connected		
76	37	V_{33_OSC}	S									Oscillator 3.3V Power Supply		
77		V_{PP}	I	C_T			X					V_{PP} Mode Selection Pin. Must be pulled-low.		
78	38	TEST	I	C_T			X					Test Mode Selection Pin. Must be tied low.		
79	39	$\overline{\text{RESET}}$	I/O	C_T			X			X		Top Priority non maskable Interrupt (active low)		
80	40	PB0/MOSI/ RX0DATA	I/O	C_T		**X**	X	ei1		X	X	Port B0	SPI Master Out / Slave In Data	RX0DATA output
81	41	PB1/MISO/ RX1DATA	I/O	C_T		**X**	X	ei1		X	X	Port B1	SPI Master In / Slave Out Data	RX1DATA output
82	42	PB2/SCK	I/O	C_T		**X**	X	ei1		X	X	Port B2	SPI Serial Clock	
83	43	PB3/$\overline{\text{SS}}$	I/O	C_T		**X**	X	ei1		X	X	Port B3	SPI Slave Select (active low)	
84	44	PB4/MCO	I/O	C_T		**X**	X	ei1		X	X	Port B4	Master Clock Output	
85		V_{SS_DIG3}	S									Digital Ground		
86		EMU8	I/O	C_T								Emulator interface 8		
87		EMU7	I/O	C_T								Emulator interface 7		

Pin n°		Pin Name	Type	Level		Port / Control						Main Function (after reset)	Alternate Functions
						Input				Output			
LQFP100	QFN48			Input	Output	float	wpu	int[2]	ana	OD	PP		
88		EMU6	I/O	C_T								Emulator interface 6	
89		EMU5	I/O	C_T								Emulator interface 5	
90		EMU4	I/O	C_T								Emulator interface 4	
91		EMU3	I/O	C_T								Emulator interface 3	
92		EMU2	I/O	C_T								Emulator interface 2	
93		EMU1	I/O	C_T								Emulator interface 1	
94		EMU0	I/O	C_T								Emulator interface 0	
95		V_{33_DIG2}	S									Digital 3.3V Power Supply	
96		NC										Not Connected	
97		NC										Not Connected	
98	45	V_{33_RF1}	S									RF1 3.3V Power Supply	
99	46	V_{SS_RF1}	S									RF1 Common (PLL and ADC) Ground	
100	47	CPOUT1	I									PLL1 Filter	

Notes:

1. I/O ports with less than 8 bits may have unbonded pads present internally. Pads that are not bonded to external pins are in input pull-up configuration after reset. The configuration of these pads must be kept at reset state to avoid added current consumption.
2. In the interrupt input column, “eiX” defines the associated external interrupt vector.
3. During normal operation, this pin must be pulled-up, internally or externally, to avoid entering ICC mode unexpectedly during a reset.
4. These I/Os feature a dedicated 4.7KΩ pull-up resistor in addition to the standard weak pull-up.
5. This I/O features a dedicated switch allowing control of 1.5KΩ external USB pull-up resistor.
6. Port A and Port G I/Os are 5V tolerant.

3 REGISTER & MEMORY MAP

As shown in Figure 4, the ST7 CPU embedded in the Device is capable of addressing 64 KBytes of memories and hardware registers.

The available memory locations consist of 128 bytes of register locations, 24Kbytes of ROM and 2.5 Kbytes of RAM. Additional hardware registers are accessible through register paging. The RAM space includes 256 bytes of stack and 48 bytes of USB data buffer. The reset vector address is FFFFh.

Important: Memory locations noted “Reserved” must never be accessed. Accessing a reserved area can have unpredictable effects on the Device.

Figure 4. Memory Map

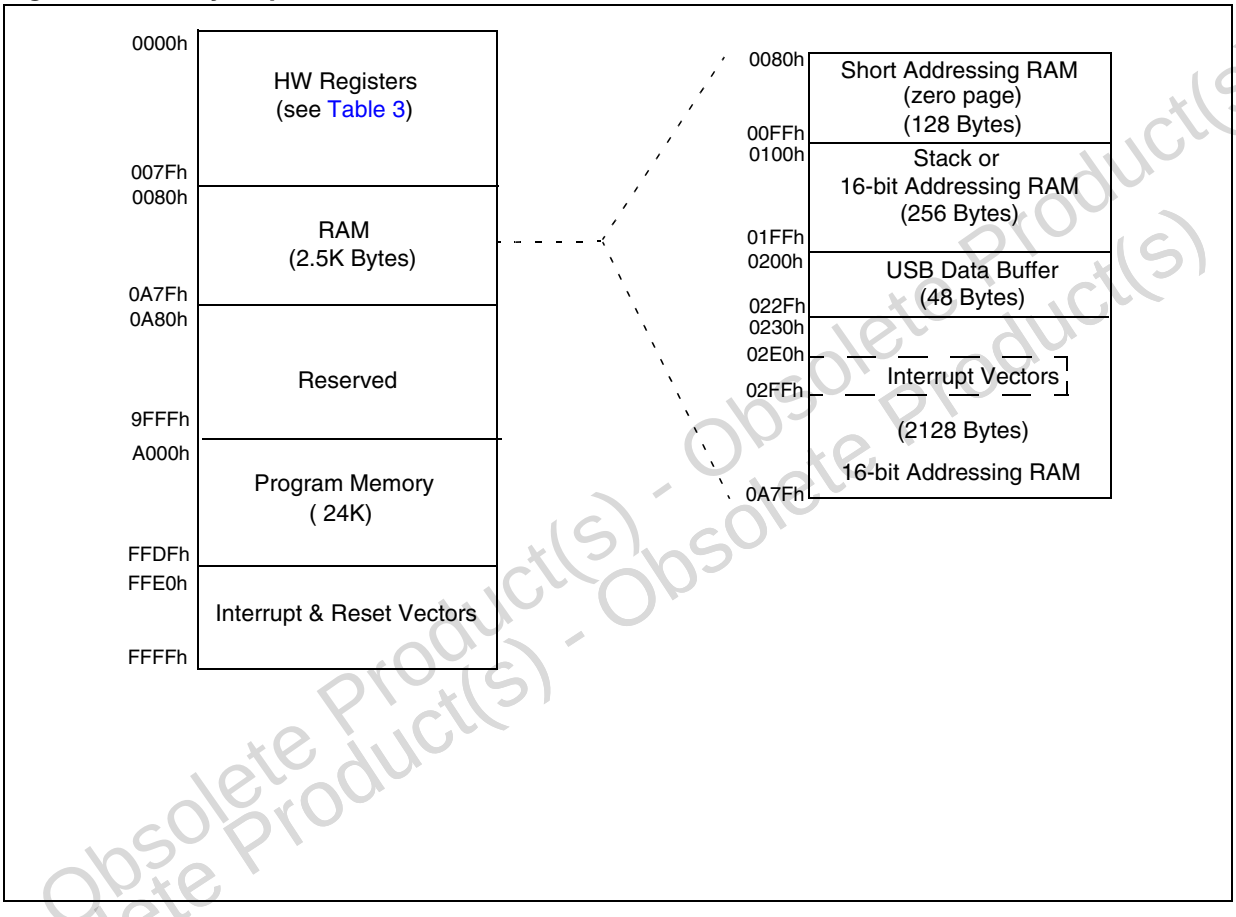


Table 3. Hardware Register Memory Map

Address	Block	Register Label	Register name	Reset Status[2)]	Remarks
0000h	Port A	PADR	Port A Data Register	00h[1)]	R/W
0001h		PADDR	Port A Data Direction Register	00h	R/W
0002h		PAOR	Port A Option Register	02h	R/W
0003h	Port B	PBDR	Port B Data Register	00h[1)]	R/W[2)]
0004h		PBDDR	Port B Data Direction Register	00h	R/W[2)]
0005h		PBOR	Port B Option Register	00h	R/W[2)]
0006h to 0011h	Reserved Area (12 Bytes)				
0012h	WATCH-DOG	WDGCR	Watchdog Control Register	7Fh	R/W
0013h		WDGCSR	Watchdog Control/Status Register	00Fh	R/W
0014h to 0016h	Reserved Area (3 Bytes)				
0017h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
0018h		SPICR	SPI Control Register	0xh	R/W
0019h		SPICSR	SPI Control/Status Register	00h	R/W
001Ah	16-BIT TIMER	T16CR2	Timer Control Register 2	00h	R/W
001Bh		T16CR1	Timer Control Register 1	00h	R/W
001Ch		T16CSR	Timer Control/Status Register	xxh	R/W
001Dh		T16IC1HR	Timer Input Capture 1 High Register	xxh	Read Only
001Eh		T16IC1LR	Timer Input Capture 1 Low Register	xxh	Read Only
001Fh		T16OC1HR	Timer Output Compare 1 High Register	80h	R/W
0020h		T16OC1LR	Timer Output Compare 1 Low Register	00h	R/W
0021h		T16CHR	Timer Counter High Register	FFh	Read Only
0022h		T16CLR	Timer Counter Low Register	FCh	Read Only
0023h		T16ACHR	Timer Alternate Counter High Register	FFh	Read Only
0024h		T16ACLR	Timer Alternate Counter Low Register	FCh	Read Only
0025h		T16IC2HR	Timer Input Capture 2 High Register	xxh	Read Only
0026h		T16IC2LR	Timer Input Capture 2 Low Register	xxh	Read Only
0027h		T16OC2HR	Timer Output Compare 2 High Register	80h	R/W
0028h		T16OC2LR	Timer Output Compare 2 Low Register	00h	R/W
0029h	TBU	TBUCVR	TBU Counter Value Register	00h	R/W
002Ah		TBUCSR	TBU Control/Status Register	00h	R/W
002Bh	Reserved Area (1 Byte)				
002Ch	Clocks	CMCR0	Clock Management Control Register 0	00h	R/W
002Dh		CMCR1	Clock Management Control Register 1	00h	R/W
002Eh		CMR	Clock Mode Register	14h	R/W
002Fh	Reserved Area (1 Byte)				
0030h	ITC	ISPR0	Interrupt Software Priority Register 0	FFh	R/W
0031h		ISPR1	Interrupt Software Priority Register 1	FFh	R/W
0032h		ISPR2	Interrupt Software Priority Register 2	FFh	R/W
0033h		ISPR3	Interrupt Software Priority Register 3	FFh	R/W
0034h		EICR	External Interrupt Control Register	00h	R/W
0035h		PAEIENR	Port A External Interrupt Enable Register	00h	R/W
0036h		PBEIENR	Port B External Interrupt Enable Register	00h	R/W
0037h	Reserved Area (1 Byte)				

Address	Block	Register Label	Register name	Reset Status[2)]	Remarks
0038h	ITC	PAEISR	Port A External Interrupt Status Register	00h	R/W
0039h		PBEISR	Port B External Interrupt Status Register	00h	R/W
003Ah	Reserved Area (1 Byte)				
003Bh	AWU	AWUCSR	Auto Wake Up From Halt Control/Status Reg.	00h	R/W
003Ch		AWUPR	Auto Wake Up From Halt Prescaler	FFh	R/W
003Dh	USB	USBISTR	USB Interrupt Status Register	00h	R/W
003Eh		USBIMR	USB Interrupt Mask Register	00h	R/W
003Fh		USBCTLR	USB Control Register	16h	R/W
0040h		USBDADDR	USB Device Address Register	00h	R/W
0041h		USBSR0	USB Status Register	00h	Read Only
0042h		USBSR1	USB Error Status Register	00h	Read Only
0043h		USBEP0R	USB Endpoint 0 Register	00h	R/W
0044h		USBCNT0RXR	USB EP0 Reception Counter Register	00h	R/W
0045h		USBCNT0TXR	USB EP0 Transmission Counter Register	00h	R/W
0046h		USBEP1RXR	USB EP1 Reception Register	00h	R/W
0047h		USBCNT1RXR	USB EP1 Reception Counter Register	00h	R/W
0048h		USBEP1TXR	USB EP1 Transmission Register	00h	R/W
0049h		USBCNT1TXR	USB EP1 Transmission Counter Register	00h	R/W
004Ah		USBEP2RXR	USB EP2 Reception Register	00h	R/W
004Bh		USBCNT2RXR	USB EP2 Reception Counter Register	00h	R/W
004Ch		USBEP2TXR	USB EP2 Transmission Register	00h	R/W
004Dh		USBCNT2TXR	USB EP2 Transmission Counter Register	00h	R/W
004Eh	PS/2 0	PS2C0R	PS/2 Driver Control 0 Register	00h	R/W
004Fh		PS2CS0R	PS/2 Control Status 0 Register	00h	R/W
0050h		PS2D0R	PS/2 Data 0 Register	00h	R/W
0051h	PS/2 1	PS2C1R	PS/2 Control 1 Register	00h	R/W
0052h		PS2CS1R	PS/2 Control Status 1 Register	00h	R/W
0053h		PS2D1R	PS/2 Data 1 Register	00h	R/W
0054h	Port G	PAGPUCR	Port A and G Pull-Up Control Register	00h	R/W
0055h		PGDR	Port G Data Register	00h	R/W
0056h		PGDDR	Port G Data Direction Register	00h	R/W
0057h		PGOR	Port G Option Register	00h	R/W
0058h	ITC	PGEICR	Port G External Interrupt Control Register	00h	R/W
0059h		PGEIENR	Port G External Interrupt Enable Register	00h	R/W
005Ah		PGEISR	Port G External Interrupt StatusRegister	00h	R/W
005Bh to 005Fh	Reserved Area (5 Bytes)				
0060h	RF Pages	RFPAGER	RF Page Selection Register	00h	R/W
0061h	RF Page 0	RFCSR	RF Control/Status Register	00h	R/W
0062h		RX0CSR	RX0 Control/Status Register	00h	R/W
0063h		RX0RSSHR	RX0 Receive Strength High Register	00h	Read Only
0064h		RX0RSSLR	RX0 Receive Strength Low Register	00h	Read Only
0065h		RX1CSR	RX1 Control/Status Register	00h	R/W
0066h		RX1RSSHR	RX1 Receive Strength High Register	00h	Read Only
0067h		RX1RSSLR	RX1 Receive Strength Low Register	00h	Read Only
0068h to 006Ah	Reserved Area (3 Bytes)				

Address	Block	Register Label	Register name	Reset Status[2)]	Remarks
0061h 0062h 0063h	RF Page 1 Common	RFREGCR0 RFREGCR1 RFSYNR	RF Regulator Control Register 0 RF Regulator Control Register 1 RF Synthesizer Register	00h 00h 00h	R/W R/W R/W
0064h to 0066h		Reserved Area (3 Bytes)			
0067h		RFTSWR	RF Track Slicer Weighting Register	00h	R/W
0068h to 006Ah		Reserved Area (3 Bytes)			
0061h 0062h 0063h 0064h	RF Page 2 RX0	RX0CFHR RX0CFLR RX0OFFHR RX0OFFLR	RX0 Carrier Frequency High Register RX0 Carrier Frequency Low Register RX0 Offset High Register RX0 Offset Low Register	00h 00h 00h 00h	R/W R/W R/W R/W
0065h 0066h		Reserved Area (2 Bytes)			
0067h 0068h 0069h		RX0DRR RX0SLHR RX0SLLR	RX0 Data Rate Register RX0 Slicer level High Register RX0 Slicer level Low Register	00h 00h 00h	R/W R/W R/W
006Ah		Reserved Area (1 Byte)			
0061h 0062h 0063h 0064h	RF Page 3 RX1	RX1CFHR RX1CFLR RX1OFFHR RX1OFFLR	RX1 Carrier Frequency High Register RX1 Carrier Frequency Low Register RX1 Offset High Register RX1 Offset Low Register	00h 00h 00h 00h	R/W R/W R/W R/W
0065h 0066h		Reserved Area (2 Bytes)			
0067h 0068h 0069h		RX1DRR RX1SLHR RX1SLLR	RX1 Data Rate Register RX1 Slicer level High Register RX1 Slicer level Low Register	00h 00h 00h	R/W R/W R/W
006Ah		Reserved Area (1 Byte)			
006Bh to 0077h	Reserved Area (13 Bytes)				
0078h 0079h 007Ah 007Bh 007Ch 007Dh 007Eh	Debug Module [3)]	DMCR1 DMCSR DMBK1HR DMBK1LR DMBK2HR DMBK2LR DMCR2	Debug Module Control Register 1 Debug Module Control Status Register Debug Module Breakpoint 1 High Register Debug Module Breakpoint 1 Low Register Debug Module Breakpoint 2 High Register Debug Module Breakpoint 2 Low Register Debug Module Control Register 2	00h 10h FFh FFh FFh FFh 00h	R/W R/W R/W R/W R/W R/W R/W
007Fh	Reserved Area (1 Byte)				

Legend: x=undefined, R/W=read/write

Notes:

1. The contents of the I/O port DR registers are readable only in output configuration. In input configuration, the values of the I/O pins are returned instead of the DR register contents.

2. The bits associated with unavailable pins must always be kept at their reset value.

3. For a description of the Debug Module registers, see ICC reference manual.

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 six CPU registers shown in Figure 5 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)

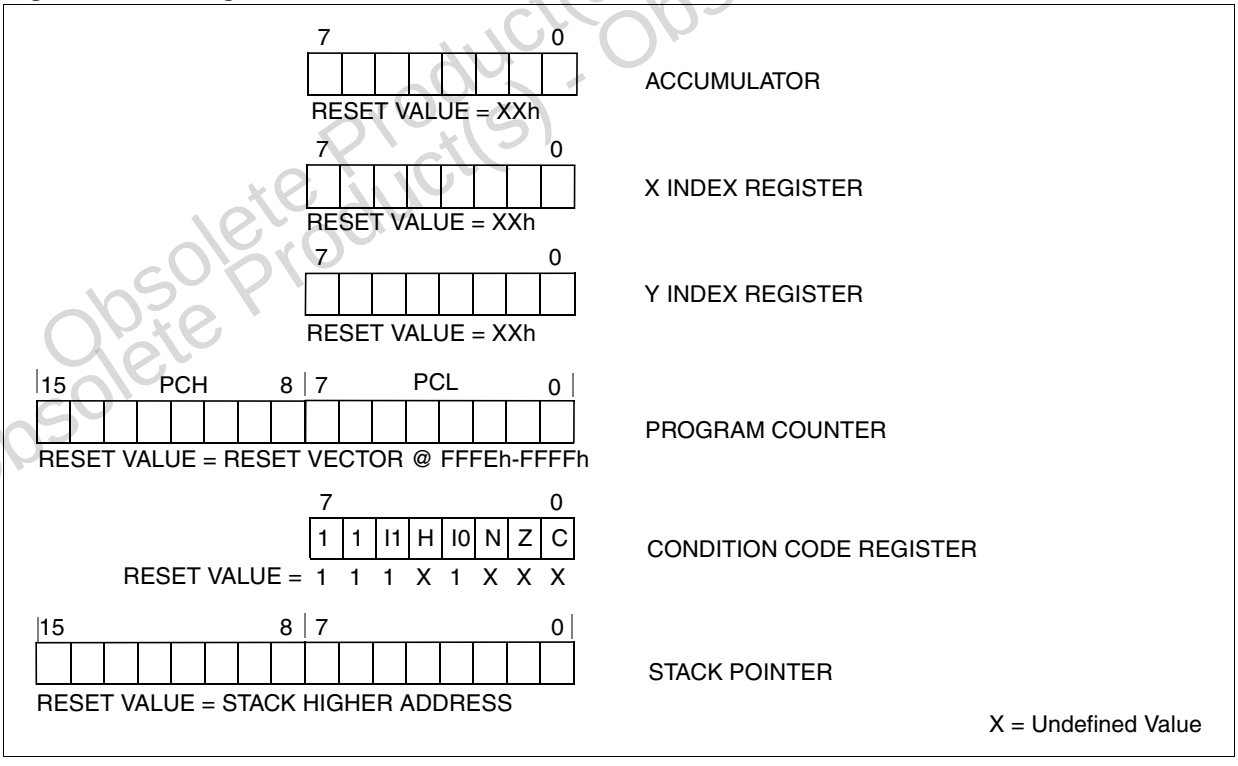
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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 5. 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 (that is, the most significant bit is a logic 1).

This bit is accessed by the JRMI and JRPL instructions.

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

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

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

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

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

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

Interrupt Management Bits

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

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

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

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

See the interrupt management chapter for more details.

CENTRAL PROCESSING UNIT (Cont'd)

STACK POINTER (SP)

Read/Write

Reset Value: 01FFh

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 6).

Since the stack is 256 bytes deep, the 8 most significant bits are forced by hardware. Following a CPU Reset, or after a Reset Stack Pointer instruction (RSP), the Stack Pointer contains its reset value (the SP7 to SP0 bits are set) which is the stack higher address.

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

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

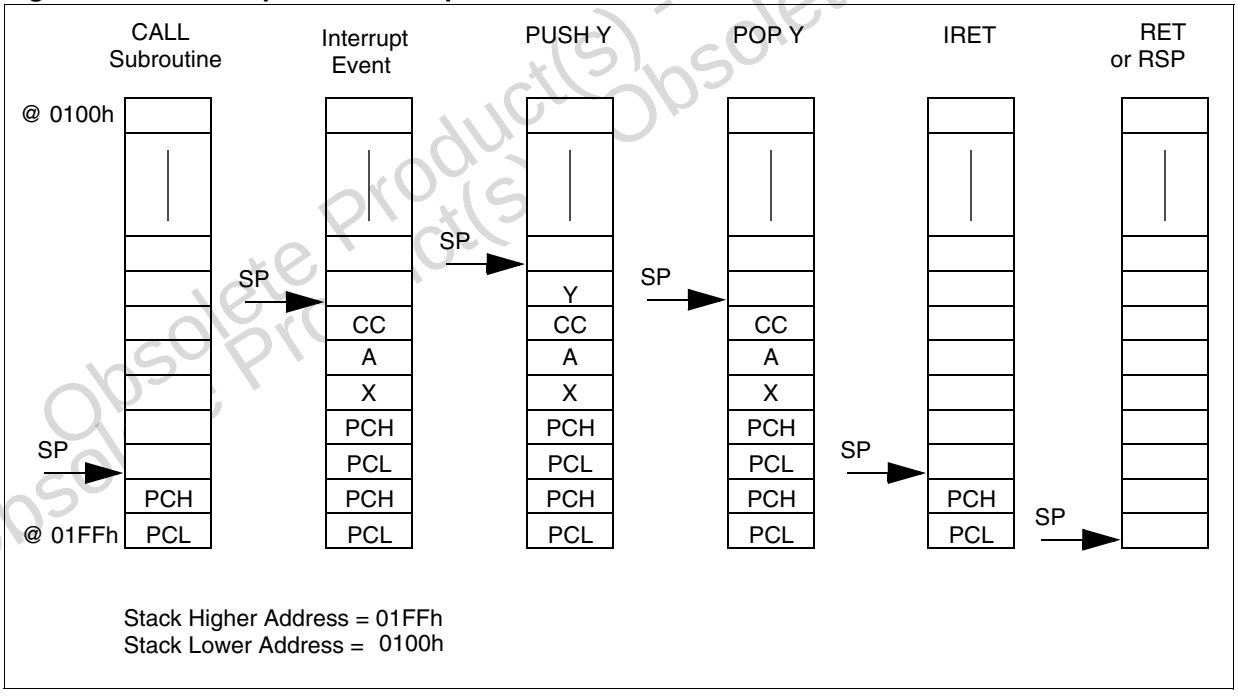
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 6.

– 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 6. Stack Manipulation Example



5 CLOCK, RESET AND SUPPLY MANAGEMENT

5.1 CLOCK SYSTEM

The main clock of the Device can be generated by three different source types:

- an external source
- a 12MHz crystal oscillator (main oscillator)
- a 6MHz internal RC oscillator

The associated hardware configurations are shown in Table 4. Refer to the electrical characteristics section for more details. On reset, the selected Device clock source is the crystal oscillator.

External Clock Source

An external clock source can be used to drive the Device. The clock signal (square, sinus or triangle) with ~50% duty cycle has to be input on the OSC1 pin while OSC2 pin is unconnected.

Crystal Oscillator

The internal oscillator is designed to operate with a 12MHz AT-cut parallel resonant quartz.

The crystal and associated components should be installed as close as possible to the input pins in order to minimize output distortion and start-up stabilisation time.

This oscillator is not stopped during the reset phase to avoid losing time in the oscillator start-up phase.

Important note:

To guarantee a correct operation of the oscillator, a serial resistor Rs of 100Ω must be placed in the external oscillator circuitry. See Table 4.

Internal RC Oscillators

The Device contains an internal RC oscillator for low power operation at a frequency of 6 MHz. A characteristic of RC oscillators is that their frequency varies from one Device to another and with temperature and voltage. For this reason, a feature allows the RC to be calibrated using the main oscillator. Calibration can be performed at any time by setting the CALRC bit in the CMR register. This also calibrates the AWU oscillator used in Auto-wakeup from Halt mode (see Section 7.5).

Table 4. Device Clock Sources

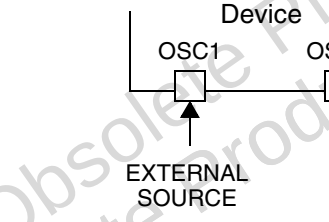
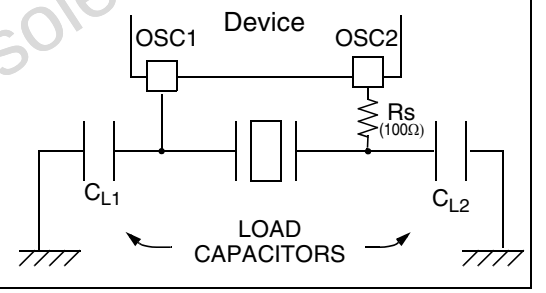
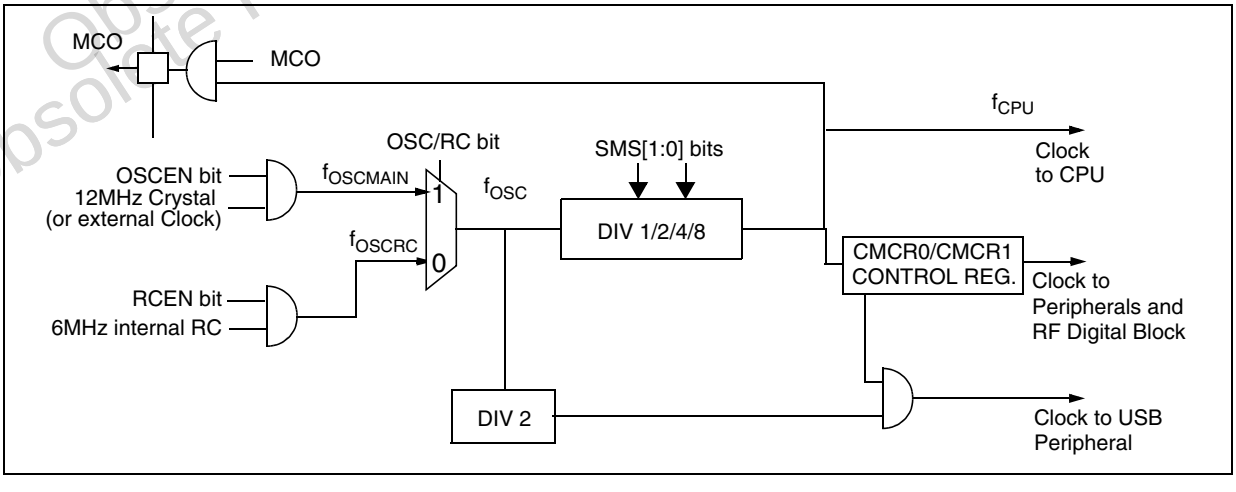
	Hardware Configuration
External Clock	
Crystal Oscillator	

Figure 7. Clock Control Block Diagram



5.2 CLOCK MANAGEMENT

To save power, the application can switch back and forth dynamically between the 12 MHz crystal oscillator or the 6 MHz RC oscillator while still staying in Run mode. This feature provides two types of run mode:

- Main Run mode (with 12 MHz clock source) for full operation with USB and RF communication
- Secondary Run mode (with 6 MHz clock source) for fast wakeup from Halt mode, but without USB or RF communication.

Control bits are also provided to enable or disable the clock to individual on-chip peripherals.

In addition the application software can put the Device in SLOW, SLOW/WAIT, Wait, HALT or Auto Wakeup from Halt mode as described in Section 7 "POWER SAVING MODES" on page 41.

5.2.1 Software Examples

Switching from Secondary to Main Run Mode

```
; SWITCH to 12 MHz
LD      A,#$1C   ; enable OSC 12 and select
it
LD      CMR, A
LD      A,#$14   ; disable RC 6MHz
LD      CMR, A
LD      A, CMR
```

Switching from Main to Secondary Run Mode

```
; SWITCH to 6 MHz
LD      A,#$0C    ; enable RC 6 and select
it
LD      CMR, A
LD      A,#$08    ; disable OSC 12MHz
LD      CMR, A
LD      A, CMR
```

5.3 CLOCK-OUT CAPABILITY

The clock-out capability is an alternate function of an I/O port pin that outputs the f_{CPU} clock to drive external devices. It is controlled by the MCO bit in the CMCR1 register.

5.3.1 Register Description

CLOCK MANAGEMENT CONTROL 0 REGISTER (CMCR0)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
RX1	RX0	RF-DSP	0	TBU	SPI	T16	USB

Bit 7 = **RX1** *RX1 clock enable*
This bit enables the clock of the RX1. It is set and cleared by software.
0: RX1 clock disabled
1: RX1 clock enabled

Bit 6 = **RX0** *RX0 clock enable*
This bit enables the clock of the RX0. It is set and cleared by software.
0: RX0 clock disabled
1: RX0 clock enabled

Bit 5 = **RF-DSP** *RF DSP clock enable*
This bit enables the clocks driving the RF and DSP. It is set and cleared by software.
0: RF DSP clocks disabled
1: RF DSP clocks enabled

Bit 4 = Reserved, must be kept cleared.

Bit 3 = **TBU** *TBU clock enable*
This bit enables the clock of the TBU. It is set and cleared by software.
0: TBU clock disabled
1: TBU clock enabled

Bit 2 = **SPI** *SPI clock enable*
This bit enables the clock of the SPI. It is set and cleared by software.
0: SPI clock disabled
1: SPI clock enabled

CLOCK MANAGEMENT (Cont'd)

Bit 1 = **T16** *16-bit timer clock enable*
This bit enables the clock of the 16-bit timer. It is set and cleared by software.
0: 16-bit timer clock disabled
1: 16-bit timer clock enabled

Bit 0 = **USB** *USB clock enable*
This bit enables the clock of the USB. It is set and cleared by software.
0: USB clock disabled
1: USB clock enabled

CLOCK MANAGEMENT CONTROL 1 REGISTER (CMCR1)
Read/Write
Reset Value: 0000 0000 (00h)

7 0

MCO	PS2-1	PS2-0	DM	0	0	0	0

Bit 7 = **MCO** *Main clock out selection*
This bit enables the MCO alternate function. It is set and cleared by software.
0: MCO alternate function disabled (I/O pin free for general-purpose I/O)
1: MCO alternate function enabled (f_{CPU} on I/O port)

Bit 6 = **PS2-1** *PS2-1 clock enable*
This bit enables the clock of the PS2-1. It is set and cleared by software.
0: PS2-1 clock disabled
1: PS2-1 clock enabled

Bit 5 = **PS2-0** *PS2-0 clock enable*
This bit enables the clock of the PS2-0. It is set and cleared by software.
0: PS2-0 clock disabled
1: PS2-0 clock enabled

Bit 4 = **DM** *Debug Module Clock enable*
This bit enables the clock of the DM. It is set by software. Once this bit is set, it cannot be cleared except by a reset.
0: DM clock disabled
1: DM clock enabled

Bits 3:2 = Reserved. Must be kept cleared.

Bits 1:0 = Reserved. Must be kept cleared.

CLOCK MODE REGISTER (CMR)
Read/Write
Reset Value: 0001 0100 (14h)

7 0

CALF	EN-CAL	STAB	OSC/RC	RC-EN	OSC-EN	SMS1	SMS0

Bit 7 = **CALF** *Calibration flag*
This bit indicates the end of the 6 MHz and auto wake-up RC calibration. It is set by hardware and cleared when setting the ENCAL bit.
0: Calibration on-going
1: Calibration finished

Bit 6 = **ENCAL** *Enable Calibration*
This bit enables the calibration of the 6 MHz and auto wake-up RC oscillator. It is set by software and cleared by hardware. The RCEN bit does not need to be set to perform a calibration.
0: Stop the calibration
1: Start the calibration

Bit 5 = **STAB** *Stabilization flag*
This bit indicates the end of the oscillator stabilization. It is cleared when either RCEN or OSCEN bit is set and it is set after a fixed delay.
0: Oscillator stabilization on-going
1: Oscillator stabilization complete
When the 6MHz RC oscillator is enabled, the STAB delay is of 11.2µs wheras it is of 1.36ms when the 12MHz oscillator is enabled. This bit does not guarantee that the oscillator is stable when STAB bit is set. It can depend on oscillator characteristics.

CLOCK MANAGEMENT (Cont'd)

Bit 4 = **OSC/RC** *Oscillator / RC selection*
This bit selects the oscillator type: main oscillator or RC. It is set and cleared by software and set by hardware when entering Halt mode. It can be used to switch from Main Run Mode to Secondary Run Mode if the corresponding oscillator is enabled (OSCEN or RCEN bits)
0: 6 MHz RC selected (if RCEN=1)
1: 12 MHz oscillator selected (if OSCEN=1)

Bit 3 = **RCEN** *6MHz RC enable*
This bit enables the RC 6MHz oscillator. It can be set by software anytime and cleared by software while OSC/RC=1. It is set by hardware after a wake up from Halt mode by external interrupt.
0: 6 MHz RC oscillator disabled
1: 6 MHz RC oscillator enabled

Bit 2 = **OSCEN** *12 MHz oscillator enable*
This bit enables the 12 MHz oscillator (or external clock). It can be set by software anytime and cleared by software while OSC/RC=0. It is cleared by hardware when entering Halt mode and set by hardware after a reset.
0: 12 MHz Oscillator disabled
1: 12 MHz Oscillator enabled

Bits 1:0 = **SMS[1:0]** *Slow mode select*
These two bits are set and cleared by software to select the slow mode division factor (as explained below:

SMS1	SMS0	division factor
0	0	1
0	1	2
1	0	4
1	1	8

Note: Software modification of the OSC/RC, RCEN and OSCEN bits is subject to the following conditions imposed by hardware logic:

1. It is not possible to modify OSCEN and RCEN at the same time (previous values are kept unchanged).
2. It is not possible to select a clock if the corresponding clock is disabled (previous value is kept unchanged).
3. It is not possible to disable RCEN if 6 MHz RC oscillator is selected.
4. It is not possible to disable OSCEN if 12MHz oscillator is selected.
5. Whatever the clock selected by the RC/OSC bit, when entering HALT mode, RC oscillator is selected, RC oscillator is disabled and 12MHz oscillator is disabled
6. The 6 MHz RC oscillator is automatically selected when waking up from HALT mode.

Caution: When OSC/RC is switched, several f_{CPU} pulses are lost because of switching from one clock source to another. This must be taken into account when using MCO or timer functions.

Table 5. Clock, Reset and Supply Control/Status Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
002Ch	**CMCR0**	RX1	RX0	RF-DSP	0	TBU	SPI	T16	USB
	Reset Value	0	0	0	0	0	0	0	0
002Dh	**CMCR1**	MCO	PS2-1	PS2-0	DM	0	0	0	0
	Reset Value	0	0	0	0	0	0	0	0
002Eh	**CMR**	CALF	ENCAL	STAB	OSC/RC	RCEN	OSCEN	SMS1	SMS0
	Reset Value	0	0	0	1	0	1	0	0

5.4 RESET SEQUENCE MANAGER (RSM)

5.4.1 Introduction

The reset sequence manager includes four RESET sources as shown in Figure 9:

- External $\overline{\text{RESET}}$ source pulse
- Internal LVD RESET (for supervising the 5V supply)
- Internal POR RESET (for supervising the power-on of the 1.8V digital parts)
- Internal WATCHDOG RESET

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to Section 10.2.1 on page 140 for further details.

These sources act on the $\overline{\text{RESET}}$ pin and it is always kept low during the delay phase.

The RESET service routine vector is fixed at addresses FFFEh-FFFFh in the Device memory map.

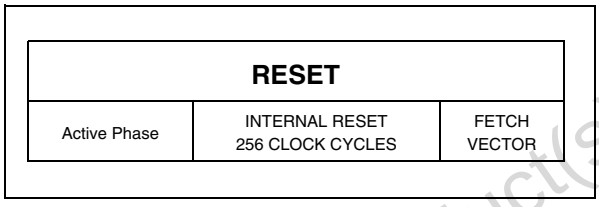
The basic RESET sequence consists of 3 phases as shown in Figure 8:

- Active Phase depending on the RESET source
- 256 CPU clock cycle delay
- RESET vector fetch

The 256 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state.

The RESET vector fetch phase duration is 2 clock cycles.

Figure 8. RESET Sequence Phases

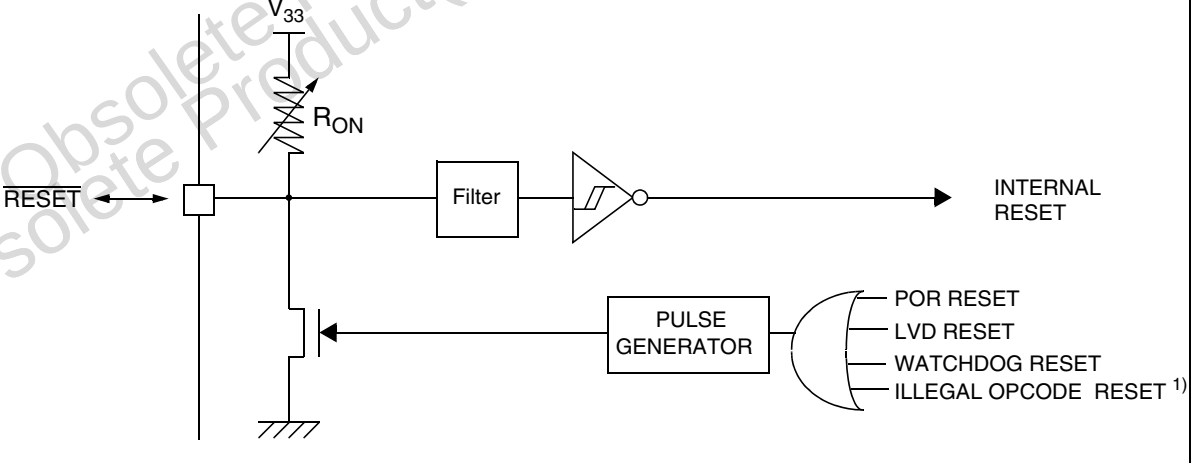


5.4.2 Asynchronous External $\overline{\text{RESET}}$ pin

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

A RESET signal originating from an external source must have a duration of at least $t_{h(RSTL)in}$ in order to be recognized (see Figure 10). This detection is asynchronous and therefore the Device can enter reset state even in HALT mode.

Figure 9. Reset Block Diagram



Note 1: See “Illegal Opcode Reset” on page 140. for more details on illegal opcode reset conditions.

RESET SEQUENCE MANAGER (Cont'd)

The $\overline{\text{RESET}}$ pin is an asynchronous signal which plays a major role in EMS performance. In a noisy environment, it is recommended to follow the guidelines mentioned in the electrical characteristics section.

5.4.3 Power-on reset (POR) RESET

The POR ensures a safe power-on sequence for the 1.8V volt digital parts of the Device.

5.4.4 Internal Low Voltage Detector (LVD) RESET

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

- Power-On RESET
- Voltage Drop RESET

The Device $\overline{\text{RESET}}$ pin acts as an output that is pulled low when $V_{DD}<V_{IT+}$ (rising edge) or $V_{DD}<V_{IT-}$ (falling edge) as shown in Figure 10.

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

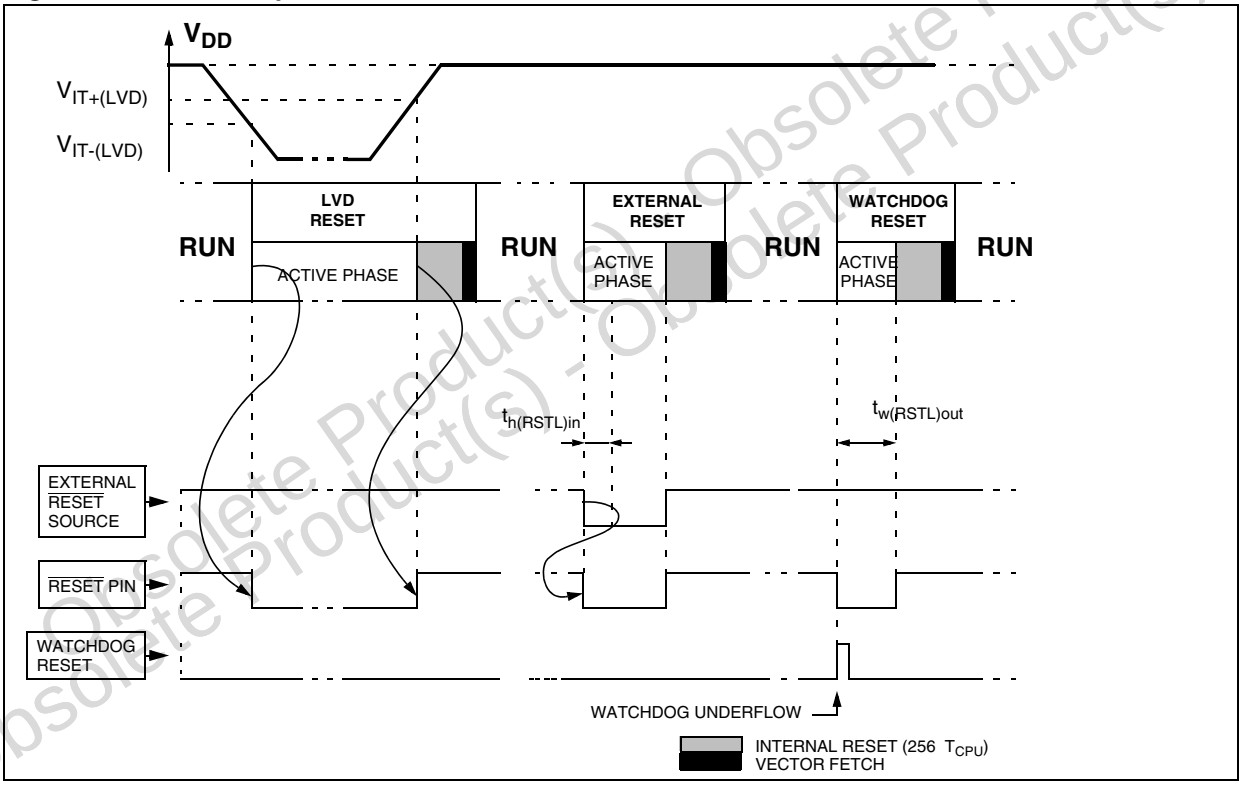
Note: it is recommended to make sure that the V_{DD} supply voltage rises monotonously when the device is exiting from RESET, to ensure the application functions properly.

5.4.5 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 10.

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

Figure 10. RESET Sequences



Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to Section 10.2.1 on page 140 for further details.

5.5 SUPPLY MANAGEMENT

The Device operates with a single external 5V power source (V_{DD}). This 5V supply is converted to 3.3V by one internal voltage regulator (REG33).

This 3.3V supply is then converted to 1.8V by several voltage regulators:

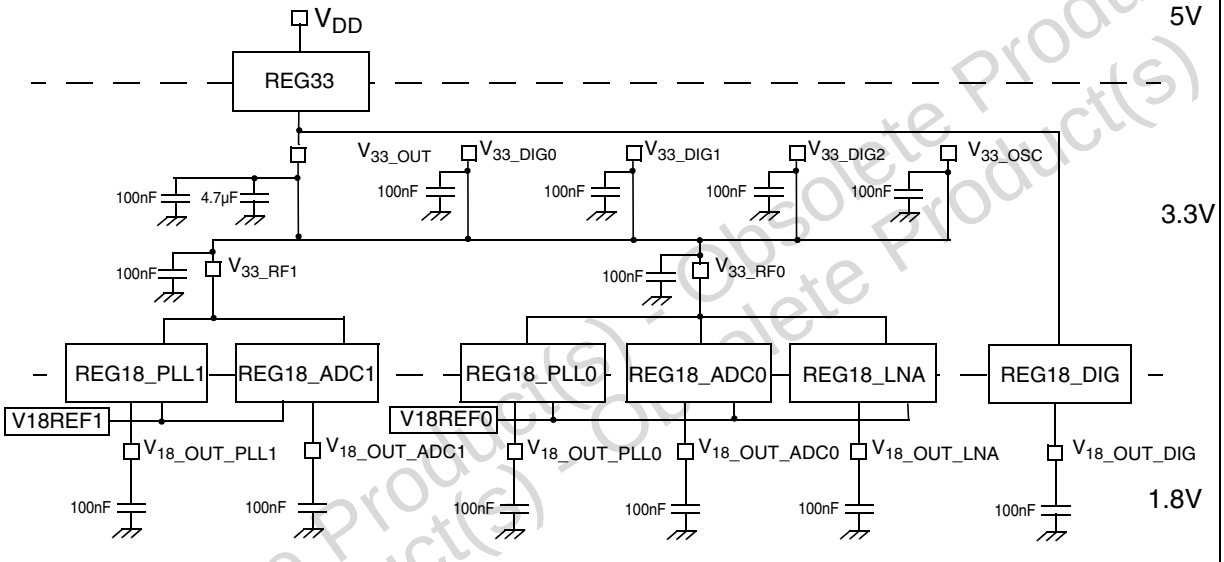
– REG18_PLL0
– REG18_ADC0
– REG18_LNA
– REG18_PLL1
– REG18_ADC1
– REG18_DIG

To reduce the Device power consumption, each of the RF regulators can be powered off individually through the RFREGCR*n* register.

The regulators REG33 and REG18_DIG cannot be powered off.

The V_{SS} connections (not shown in Figure 11) can be connected to a common ground plane. $V_{SS\_SUB\_RF}$ and $V_{SS\_SUB\_DIG}$ are the shielding ground connections for the RF analog and digital parts respectively.

Figure 11. RF Voltage Regulator Interconnections



Note: Refer to Figure 56 on page 92 for more information on RF power supply decoupling and external components.

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 6). The processing flow is shown in Figure 12.

When an interrupt request has to be serviced:

– Normal processing is suspended at the end of the current instruction execution.

– The PC, X, A and CC registers are saved onto the stack.

– I1 and I0 bits of CC register are set according to the corresponding values in the ISPRx registers of the serviced interrupt vector.

– The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to “Interrupt Mapping” table for vector addresses).

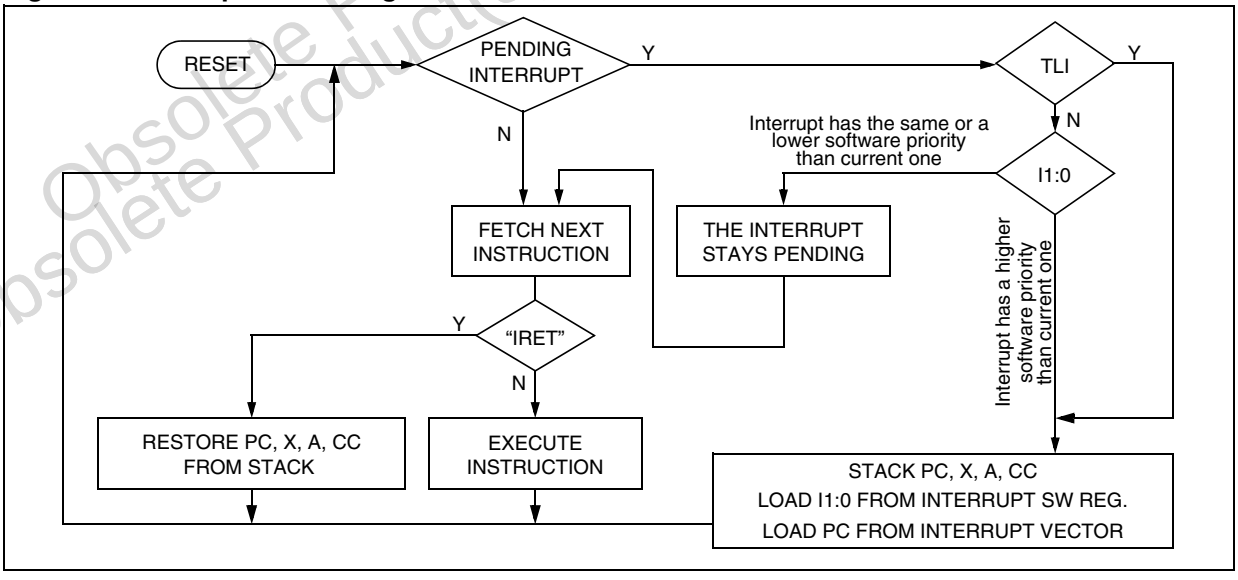
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

Note: As a consequence of the IRET instruction, the I1 and I0 bits will be restored from the stack and the program in the previous level will resume.

Table 6. Interrupt Software Priority Levels

Interrupt software priority	Level	I1	I0
Level 0 (main)	Low	1	0
Level 1	↓	0	1
Level 2	↓	0	0
Level 3 (= interrupt disable)	High	1	1

Figure 12. Interrupt Processing Flowchart



INTERRUPTS (Cont'd)

Servicing Pending Interrupts

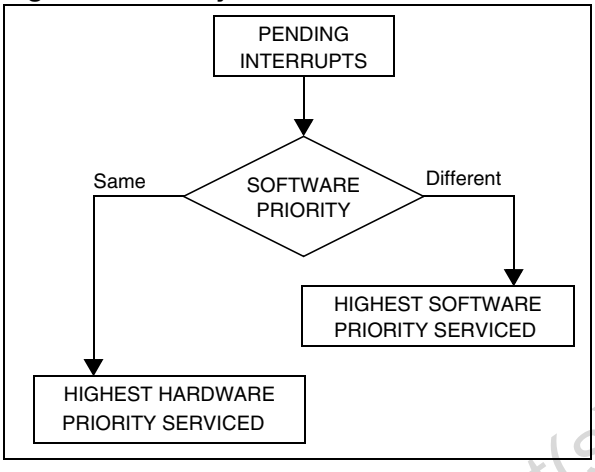
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– 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 13 describes this decision process.

Figure 13. 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, 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 12). 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 is generated when the Watchdog Time-out Flag is set.
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 12 as a TLI.
Caution: TRAP cannot 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 External Interrupt Control register (EICR).
External interrupt triggered on edge will be latched and the interrupt request automatically cleared upon entering the interrupt service routine.
If several input pins of a group connected to the same interrupt line are selected simultaneously, these will be logically 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 13.

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 14 and Figure 15 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 15. 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 14. Concurrent Interrupt Management

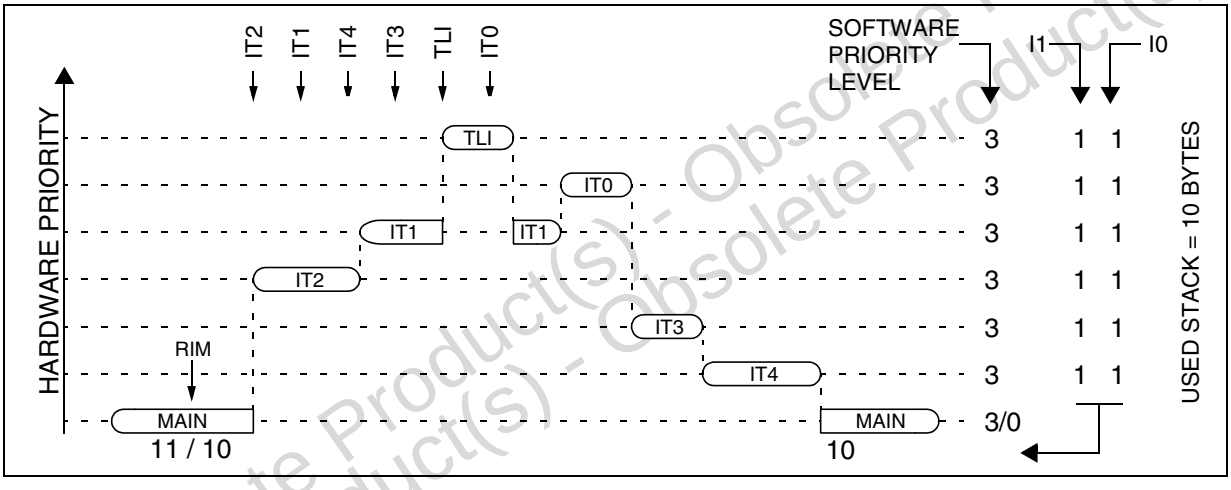
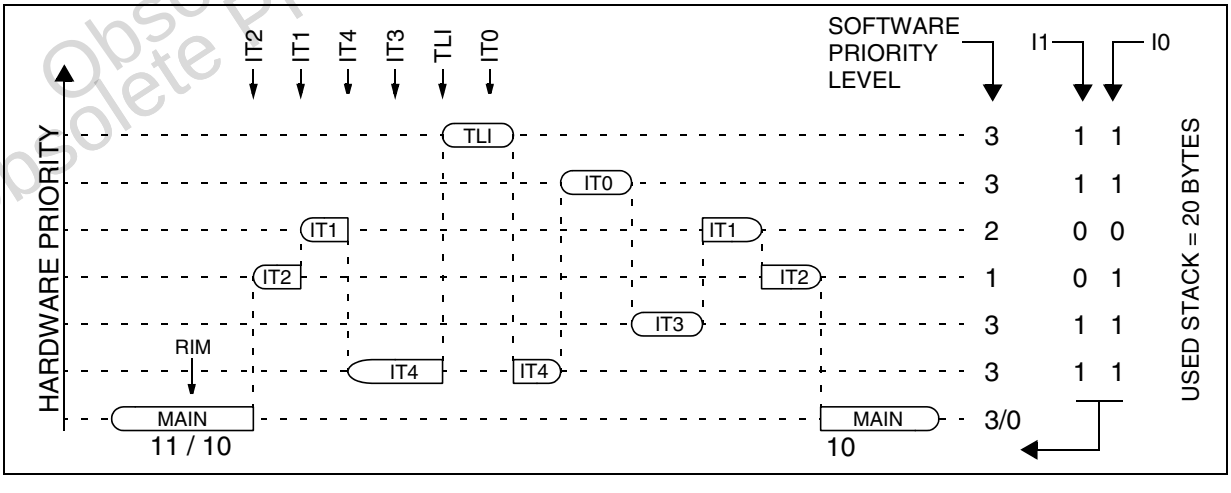


Figure 15. Nested Interrupt Management



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).

INTERRUPTS (Cont'd)

Table 7. Dedicated Interrupt Instruction Set

Instruction	New Description	Function/Example	I1	H	I0	N	Z	C
HALT	Entering Halt mode		1		0			
IRET	Interrupt routine return	Pop CC, A, X, PC	I1	H	I0	N	Z	C
JRM	Jump if I1:0=11	I1:0=11 ?						
JRNM	Jump if I1:0<>11	I1:0<>11 ?						
POP CC	Pop CC from the Stack	Mem => CC	I1	H	I0	N	Z	C
RIM	Enable interrupt (level 0 set)	Load 10 in I1:0 of CC	1		0			
SIM	Disable interrupt (level 3 set)	Load 11 in I1:0 of CC	1		1			
TRAP	Software trap	Software NMI	1		1			
WFI	Wait for interrupt		1		0			

Note: During the execution of an interrupt routine, the HALT, POPCC, RIM, SIM and WFI instructions change the current software priority up to the next IRET instruction or one of the previously mentioned instructions.
In order not to lose the current software priority level, the RIM, SIM, HALT, WFI and POP CC instructions should never be used in an interrupt routine.

DM CONTROL REGISTER (DMCR1)

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

7							0
0	MTR	0	0	0	0	0	0

Bit 7 = Reserved, must be kept cleared.

Bit 6 = **MTR** *Monitor Control.*
This bit must be set to access all DM registers, if this bit is cleared all DM registers except MTR bit are write protected. This bit is set by software or by hardware at the beginning of ICC Monitor execution. It is cleared by hardware at the end of the ICC Monitor.

0: ICC Monitor program is not running
1: ICC Monitor program is running

Bit 5:0 = Reserved, must be kept cleared.

DM CONTROL REGISTER 2 (DMCR2)

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

7							0
0	0	0	0	0	0	0	SVR

Bit 7:1= Reserved, must be kept cleared

Bit 0 = **SVR** *Switch Interrupt Vectors to RAM*

This bit is set and cleared by software. It switches the interrupt vector table location to RAM
0: Interrupt vector table located in ROM
1: Interrupt vector table located in RAM

6.6 Interrupt Vector Table Management

For added flexibility, the ST7WInD21 features two interrupt vector table modes. After reset, the interrupt vectors are located in ROM. The application can switch the vectors to RAM by executing the sequence in the procedure given below. Prior to switching the vectors to RAM, the RAM area must be initialised.

Procedure:

To switch the vectors from ROM to RAM:

1. Initialise the RAM area with the correct interrupt vectors
2. Set the MTR bit in the DMCR1 register
3. Set the SVR bit in the DMCR2 register to enable the interrupt vector table in RAM

Table 8. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from HALT	Vector Address (ROM)	Vector Address (RAM)
	RESET	Reset	N/A	Highest Priority	yes	FFFEh-FFFFh	
	TRAP	Software interrupt			no	FFFCh-FFFDh	02FCh-02FDh
0	WDG	Watchdog TLI interrupt	WDGCSR		no	FFFAh-FFFBh	02FAh-02FBh
1	USB	USB end of suspend	USBISTR		yes	FFF8h-FFF9h	02F8h-02F9h
2	ei0 / AWU	Port A external interrupts/ Auto Wake up From HALT	PAEISR / AWUCSR		yes	FFF6h-FFF7h	02F6h-02F7h
3	ei1	Port B external interrupts	PBEISR		yes	FFF4h-FFF5h	02F4h-02F5h
4	reserved	reserved	reserved		no	FFF2h-FFF3h	02F2h-02F3h
5	reserved	reserved	reserved		no	FFF0h-FFF1h	02F0h-02F1h
6	reserved	reserved	reserved		no	FFEEh-FFEFh	02EEh-02EFh
7	reserved	reserved	reserved		no	FFECh-FFEDh	02ECh-02EDh
8	PS/2 0 ei3	PS/2 0 interrupt Port G external interrupt	PS2CS0R PGEISR		no yes	FFEAh-FFEBh	02EAh-02EBh
9	PS/2 1	PS/2 1 interrupt	PS2CS1R		no	FFE8h-FFE9h	02E8h-02E9h
10	USB	USB interrupt	USBISTR		no	FFE6h-FFE7h	02E6h-02E7h
11	16-bit Timer	16-bits Timer interrupt	T16CSR		no	FFE4h-FFE5h	02E4h-02E5h
12	TBU	Time Base Unit interrupt	TBUCSR	Lowest Priority	no	FFE2h-FFE3h	02E2h-02E3h
13	SPI	SPI interrupt	SPISR		yes	FFE0h-FFE1h	02E0h-02E1h

INTERRUPTS (Cont'd)

6.7 EXTERNAL INTERRUPTS

When an event occurs on an I/O port, this incoming signal is interpreted as an external interrupt. This signal can also be used to wake up the Device from HALT. There are several controlling factors for external interrupts:

– Priority (Hardware and Software)

– Enable/Disable control bits

– Sensitivity Control

– Status Flag

Up to 8 signals on 8 ports can share one external interrupt. For example, ei0 is shared on all 8 ports of Port A.

6.7.1 Software and Hardware Priorities

External interrupts have default priorities associated with them. They are as listed in the Interrupt Mapping table. These are the hardware priorities and are unchangeable.

Software priorities are user assigned by programming the appropriate bits in the Interrupt Software Priority register (ISPRx) for a given external interrupt. The whole external interrupt group will have the same priority. For example, ISPR0 bits[4:5] control the software priority for Port A's external interrupt, ei0.

These two types of priorities are important to manage because they function in the same manner as other interrupts for concurrent and nested modes.

6.7.2 Enable and Sensitivity Controls

At an external interrupt event, for the interrupt to be acknowledged, it must be enabled. There is a control bit for each external interrupt. They are found in the External Interrupt Enable Port x register (EINENxR).

The external interrupt sensitivity is controlled by the ISxx bits in the EICR (Figure 16). This control allows to have up to 4 fully independent external interrupt source sensitivities. Port B is divided on two different controls, to allow more programming flexibility.

Each external interrupt source can be generated on four different events on the pin:

- Falling edge
- Rising edge
- Falling and rising edge
- Falling edge and low level

To guarantee correct functionality, the sensitivity bits in the EICR register can be modified only when the I1 and I0 bits of the CC register are both set to 1 (level 3).

6.7.3 Status Flag

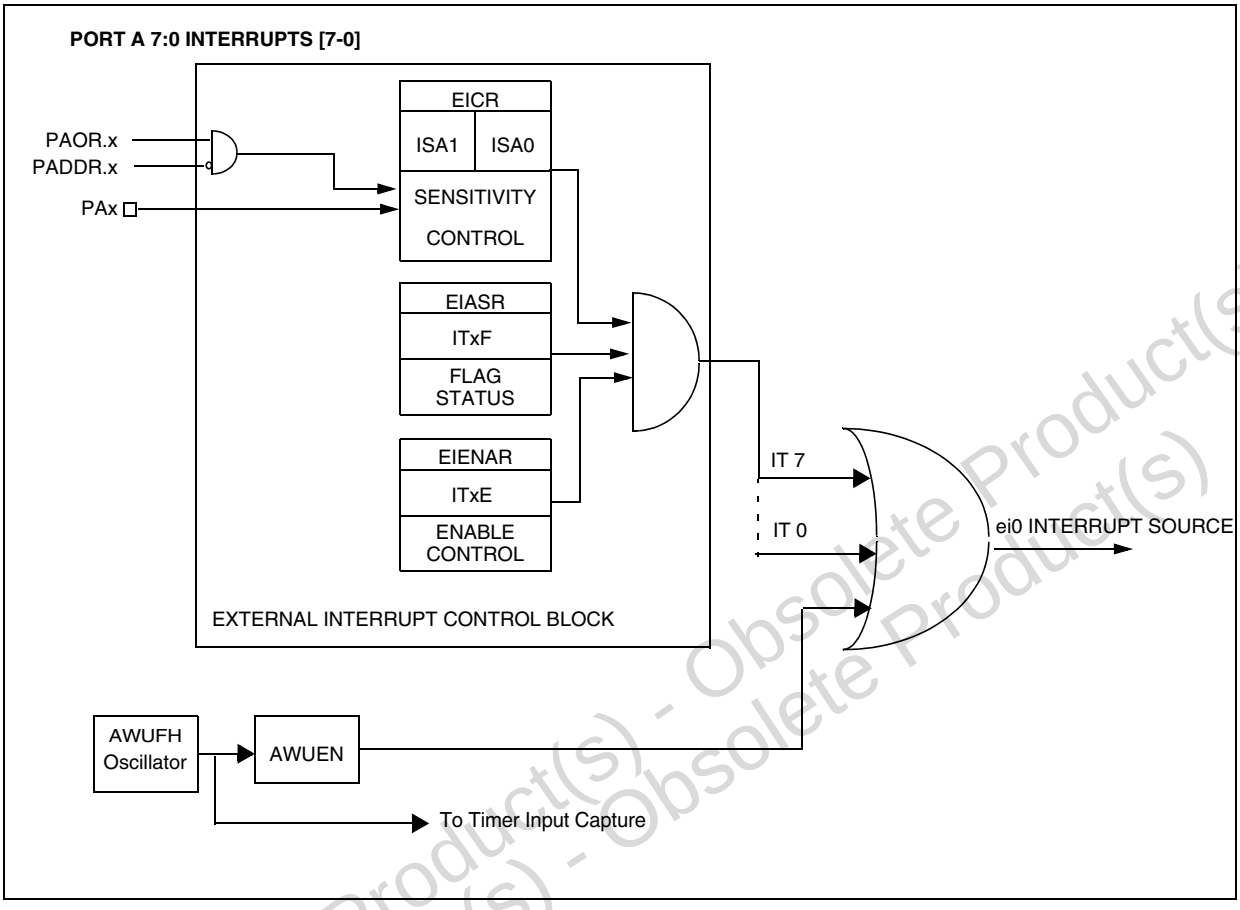
When an event occurs signalling that an external interrupt is requested, a flag is set by hardware. This flag informs the user which external interrupt has occurred. Each external interrupt has its own specific flag. They are found in the External Interrupt Port x register (EIxSR). If the corresponding external interrupt is enabled when this flag is set, the external interrupt is serviced.

If several interrupts are pending, the interrupts are serviced according to their priority (software and or hardware, according to which interrupt mode is being employed).

If there is an unwanted pending interrupt, it can be cleared by writing a different value in the ISx[1:0] in the EICR.

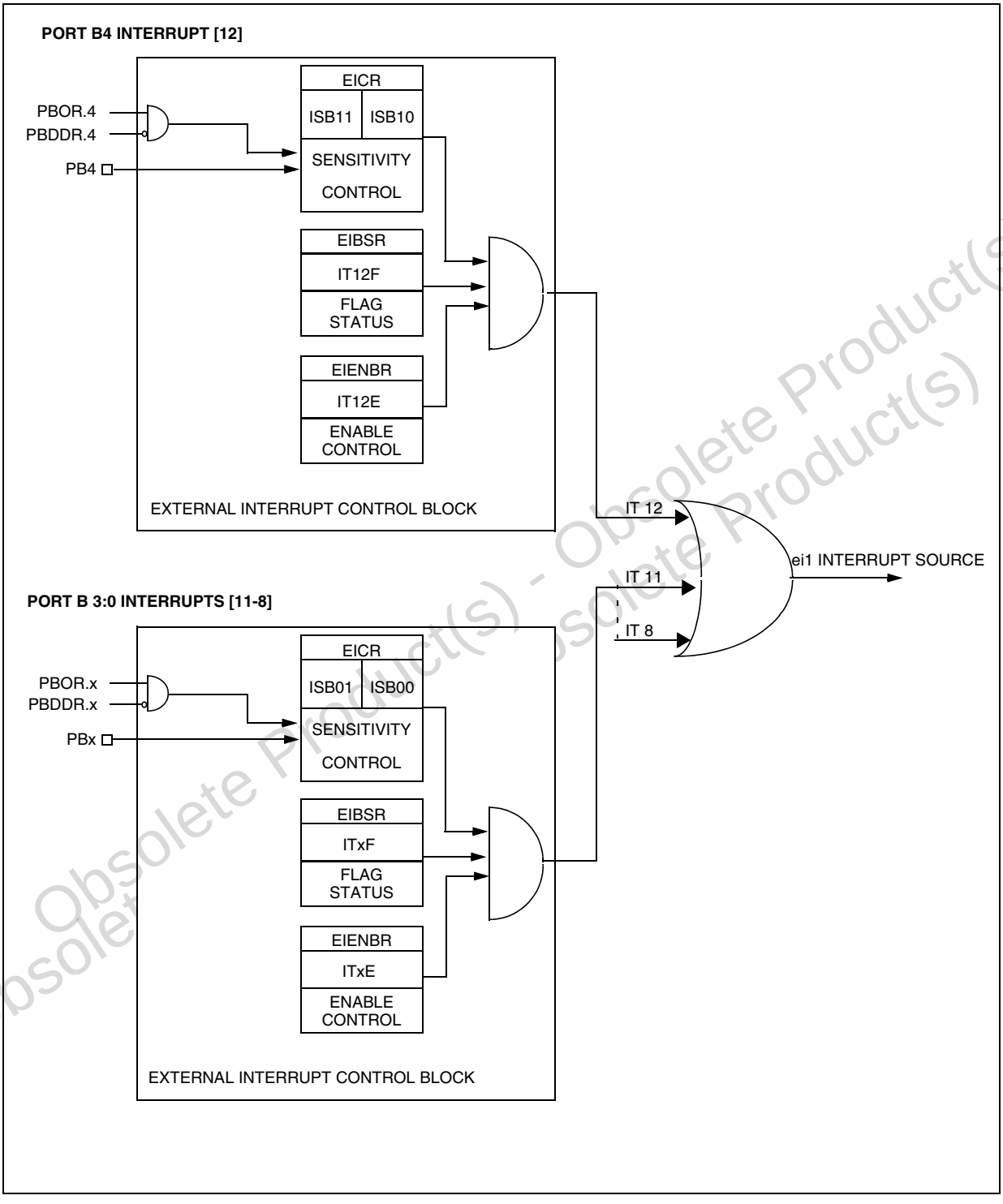
INTERRUPTS (Cont'd)

Figure 16. Port A External Interrupt Control bits



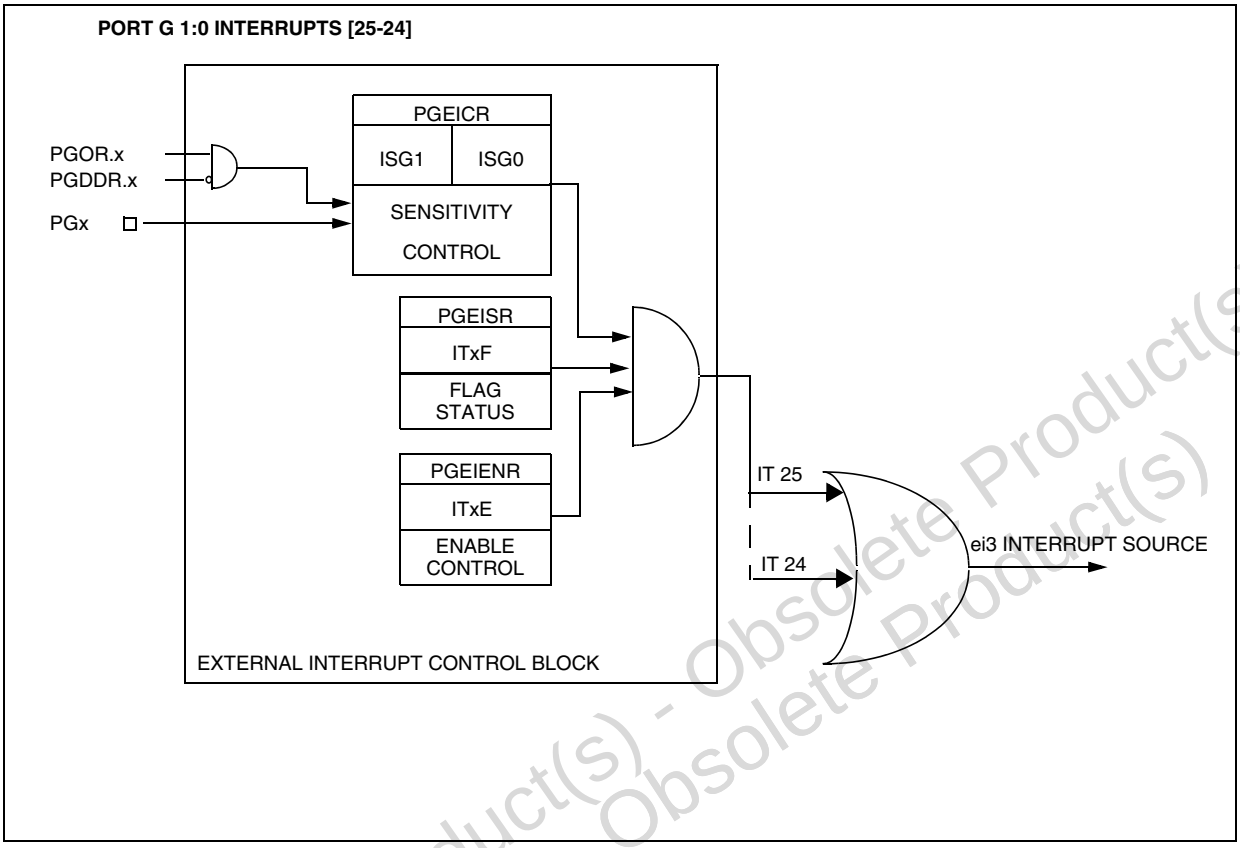
INTERRUPTS (Cont'd)

Figure 17. Port B External Interrupt Control bits



INTERRUPTS (Cont'd)

Figure 18. Port G External Interrupt Control bits



INTERRUPTS (Cont'd)

6.7.4 Register Description

EXTERNAL INTERRUPT CONTROL REGISTER (EICR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	ISB11	ISB10	ISB01	ISB00	ISA1	ISA0

Bits 7:6 = Reserved, must be kept cleared.

Bits 5:4 = **ISB1[1:0]** *Port B4 ei1 sensitivity IT12*
The interrupt sensitivity, defined using the ISB1[1:0] bits, is applied to the ei1 external interrupts:

ISB11	ISB10	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

These 2 bits can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

Bits 3:2 = **ISB0[1:0]** *Port B(3:0) ei1 sensitivity-IT[11-8]*
The interrupt sensitivity, defined using the ISB0[1:0] bits, is applied to the ei1 external interrupts:

ISB01	ISB00	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

These 2 bits can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

Bits 1:0 = **ISA[1:0]** *Port A ei0 sensitivity IT[7-0}*
The interrupt sensitivity, defined using the ISA[1:0] bits, is applied to the ei0 external interrupts:

ISA1	ISA0	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

These 2 bits can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

PORT A EXTERNAL INTERRUPT ENABLE REGISTER (PAEIENR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
IT7E	IT6E	IT5E	IT4E	IT3E	IT2E	IT1E	IT0E

Bits 7:0 = **ITxE** *Port A interrupt enable*

These bits are set and cleared by software.
0: ITx external interrupt disabled.
1: ITx external interrupt enabled.

PORT B EXTERNAL INTERRUPT ENABLE REGISTER (PBEIENR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	IT12E	IT11E	IT10E	IT9E	IT8E

Bits 7:5 = Reserved, must be kept cleared.

Bits 4:0 = **ITxE** *Port B interrupt enable*

These bits are set and cleared by software.
0: ITx external interrupt disabled.
1: ITx external interrupt enabled.

INTERRUPTS (Cont'd)

PORT A EXTERNAL INTERRUPT STATUS REGISTER (PAEISR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
IT7F	IT6F	IT5F	IT4F	IT3F	IT2F	IT1F	IT0F

Bits 7:0 = **ITxF** *Port A interrupt flag*

These bits are set by hardware and cleared by software (by writing 0).
0: ITx external interrupt not requested.
1: ITx external interrupt requested.

PORT B EXTERNAL INTERRUPT STATUS REGISTER (PBEISR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	IT12F	IT11F	IT10F	IT9F	IT8F

Bits 7:5 = Reserved, must be kept cleared.

Bits 4:0 = **ITxF** *Port B interrupt flag*

These bits are set by hardware and cleared by software (by writing 0).
0: ITx external interrupt not requested.
1: ITx external interrupt requested.

INTERRUPTS (Cont'd)

PORT G EXTERNAL INTERRUPT CONTROL REGISTER (PGEICR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	ISG1	ISG0

Bits 7:2 = Reserved.

Bit 1:0 = **ISG[1:0]** *Interrupt sensitivity Port G1:0*
The interrupt sensitivity, defined using the ISG[1:0] bits, is applied to the ei3 external interrupts:

ISG1	ISG0	External Interrupt Sensitivity
0	0	Falling edge & low level
0	1	Rising edge only
1	0	Falling edge only
1	1	Rising and falling edge

These 2 bits can be written only when I1 and I0 of the CC register are both set to 1 (level 3).

PORT G EXTERNAL INTERRUPT ENABLE REGISTER (PGEIENR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	IT25E	IT24E

Bits 7:2 = Reserved, must be kept cleared.

Bit 1 = **IT25E** *ei3 external interrupt enable*
0: ei3 external interrupt disabled on port G1.
1: ei3 external interrupt enabled on port G1.

Bit 0 = **IT24** *ei3 external interrupt enable*
0: ei3 external interrupt disabled on port G0.
1: ei3 external interrupt enabled on port G0.

PORT G EXTERNAL INTERRUPT STATUS REGISTER (PGEISR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	IT25F	IT24F

Bits 7:2 = Reserved, must be kept cleared.

Bit 1 = **IT25F** *ei3 external interrupt flag*
0: ei3 external interrupt not requested on port G1.
1: ei3 external interrupt requested on port G1.

Bit 0 = **IT24F** *ei3 external interrupt flag*
0: ei3 external interrupt not requested on port G0.
1: ei3 external interrupt requested on port G0.

INTERRUPTS (Cont'd)

Table 9. Nested Interrupts Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0030h		ei1		ei0/AWUFH		USB EOSuspend		WDG	
	ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1		
	Reset Value	1	1	1	1	1	1	1	1
0031h		reserved		reserved		reserved		reserved	
	ISPR1	I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4	I0_4
	Reset Value	1	1	1	1	1	1	1	1
0032h		T16		USB		PS2_1		PS2_0/ei3	
	ISPR2	I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8	I0_8
	Reset Value	1	1	1	1	1	1	1	1
0033h						SPI		TBU	
	ISPR3					I1_13	I0_13	I1_12	I0_12
	Reset Value	1	1	1	1	1	1	1	1
0034h	**EICR**	0	0	ISB11	ISB10	ISB01	ISB00	ISA1	ISA0
	Reset Value	0	0	0	0	0	0	0	0
0035h	**PAEIENR**	IT7E	IT6E	IT5E	IT4E	IT3E	IT2E	IT1E	IT0E
	Reset Value	0	0	0	0	0	0	0	0
0036h	**PBEIENR**				IT12E	IT11E	IT10E	IT9E	IT8E
	Reset Value	0	0	0	0	0	0	0	0
0037h	**reserved**								
	Reset Value	0	0	0	0	0	0	0	0
0038h	**PAEISR**	IT7F	IT6F	IT5F	IT4F	IT3F	IT2F	IT1F	IT0F
	Reset Value	0	0	0	0	0	0	0	0
0039h	**PBEISR**				IT12F	IT11F	IT10F	IT9F	IT8F
	Reset Value	0	0	0	0	0	0	0	0
003Ah	**reserved**								
	Reset Value	0	0	0	0	0	0	0	0
0058h	**PGEICR**							ISG1	ISG0
	Reset Value	0	0	0	0	0	0	0	0
0059h	**PGEIENR**							IT25E	IT24E
	Reset Value	0	0	0	0	0	0	0	0
005Ah	**PGEISR**							IT25F	IT24F
	Reset Value	0	0	0	0	0	0	0	0
0078h	**DMCR1**	0	MTR	0	0	0	0	0	0
	Reset Value		0						
007Eh	**DMCR2**	0	0	0	0	0	0	0	SVR
	Reset Value								0

7 POWER SAVING MODES

7.1 INTRODUCTION

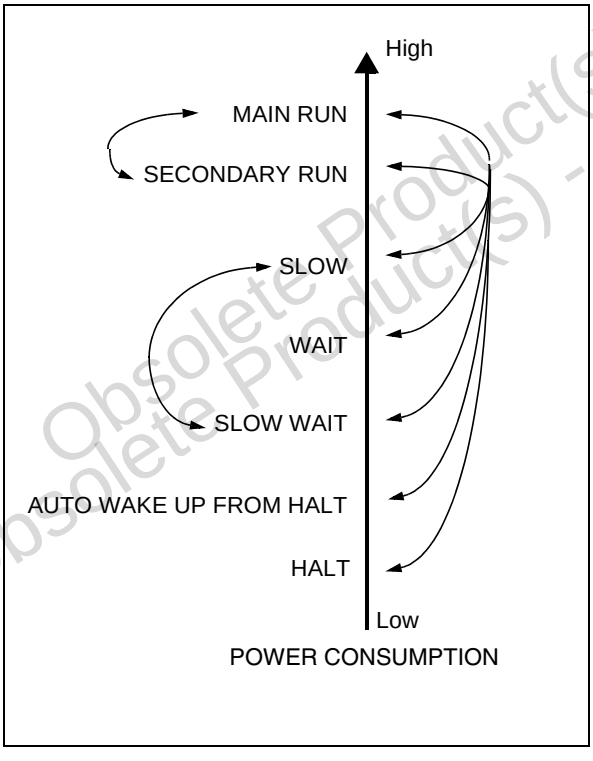
To give a large measure of flexibility to the application in terms of power consumption, five main power saving modes are implemented in the Device (see Figure 19):

- Secondary Run Mode (See Section 5.1 on page 20
- Slow
- Wait (and Slow-Wait)
- Auto Wake up From Halt (AWUFH)
- Halt

After a RESET the main operating mode is selected by default . This mode drives the Device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency ($f_{OSCMAIN}$).

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

Figure 19. Power Saving Mode Transitions



7.2 SLOW MODE

This mode has two targets:

– to reduce power consumption by decreasing the internal clock in the Device,

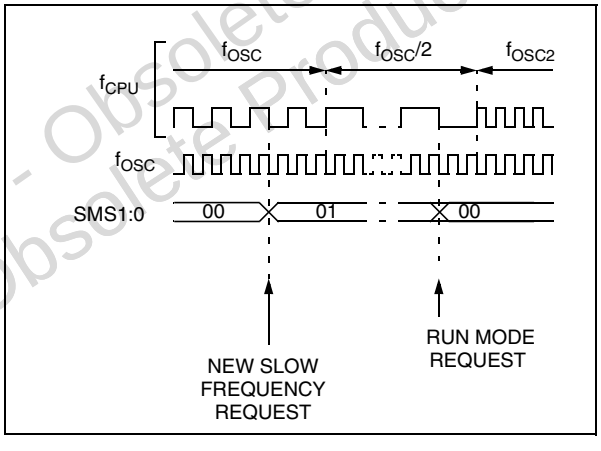
– to adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

SLOW mode is controlled by two bits in the CMR register which select the internal slow frequency (f_{CPU}).

In this mode, the master clock frequency (f_{OSC}) can be divided by 1, 2, 4, or 8. The CPU and peripherals are clocked at this lower frequency (f_{CPU}).

Note: SLOW-WAIT mode is activated by entering WAIT mode while the Device is in SLOW mode.

Figure 20. SLOW Mode Clock Transitions



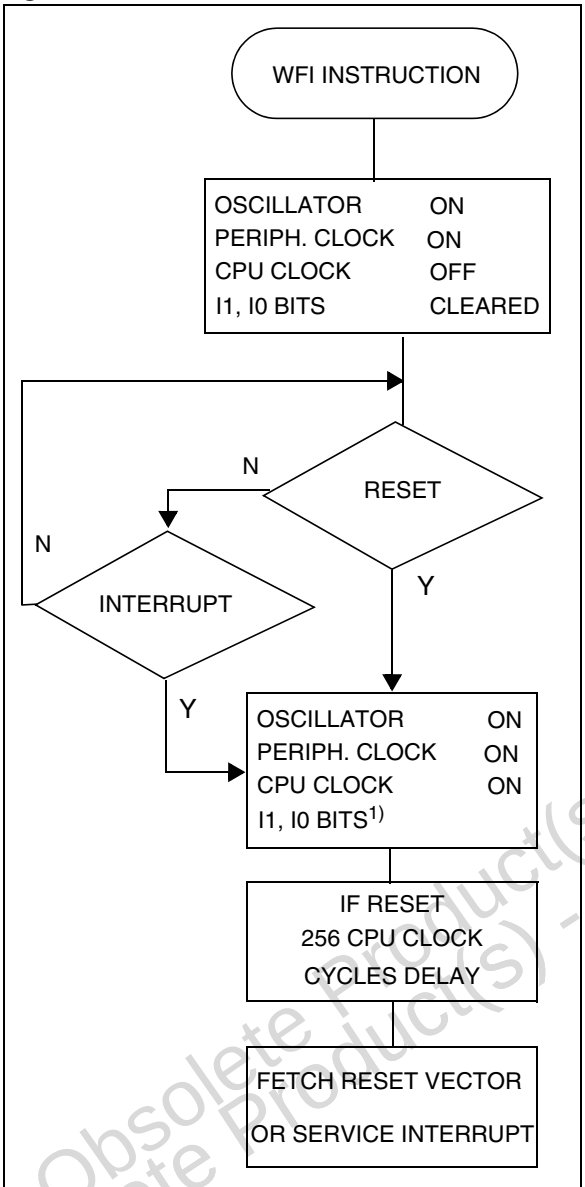
7.3 WAIT MODE

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

This power saving mode is selected by executing the “WFI” CPU instruction.

All peripherals remain active. During WAIT mode, the I bits in the CC register are forced to 0, enabling all interrupts. All other registers and memory remain unchanged. The Device remains in WAIT mode until an interrupt or reset occurs. If the event is an interrupt, the program counter immediately branches to the starting address of the interrupt or reset service routine. If the wake up event is a reset, before fetching the reset vector, there is a 256 CPU clock cycle delay to allow for stabilization. Refer to Figure 21.

Figure 21. WAIT Mode Flow Chart



Note: 1) Before servicing an interrupt, the CC register is pushed on the stack. The I0 and I1 bit values for each interrupt are predefined by the user in the ISPRx register. During the interrupt routine these values are loaded into I0 and I1 bits and cleared when the CC register is popped.

POWER SAVING MODES (Cont'd)

7.4 HALT MODE

HALT mode is the lowest power consumption mode. HALT mode is entered by executing the HALT instruction. The all the internal oscillators are stopped, causing all internal processing to be stopped, including the operation of the on-chip peripherals. Only peripherals with an external oscillator continue running.

Entering HALT mode clears the I bits in the CC register, enabling interrupts. If an interrupt is pending, the Device wakes up immediately. Not all interrupts will wake up the Device from HALT, only those listed in the Interrupt Mapping Table in the Interrupt section allow wake-up.

These specific interrupts (as described in Table 8) or a reset wakes up the Device from HALT mode.

– If a reset is the wake-up event, the main oscillator is immediately turned on and a 256 CPU cycle delay is used to stabilize the oscillator. After the start up delay, the CPU resumes operation by fetching the reset vector.

– If an interrupt is the wakeup event, the RC oscillator is immediately turned on and the CPU immediately resumes operation by servicing the interrupt which woke it up. The user can choose to switch to the main oscillator by programming the CMR register.

Refer to Figure 22 for more details.

7.4.1 HALT MODE RECOMMENDATIONS

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

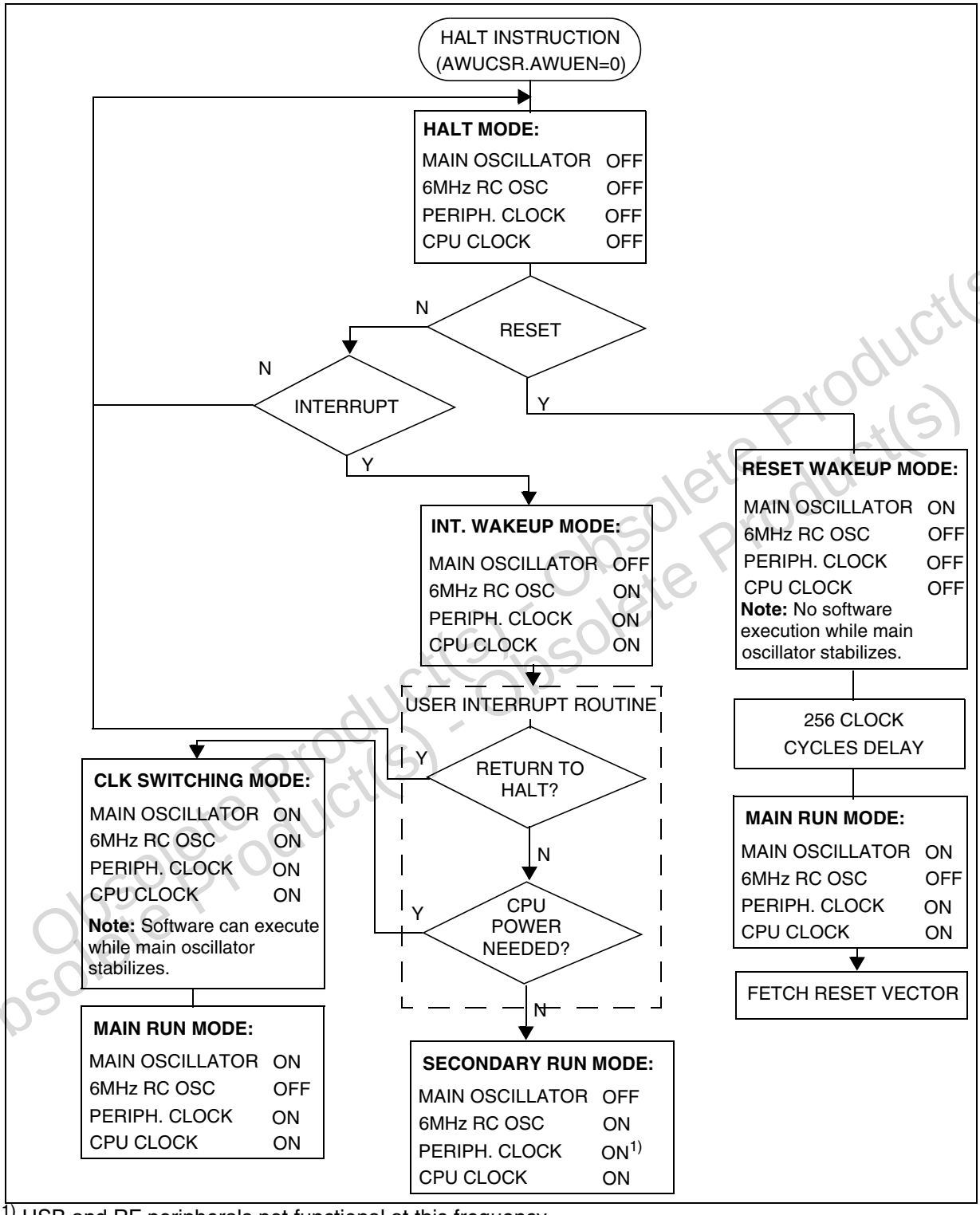
– When using an external interrupt to wake up the Device, reinitialize the corresponding I/O as "Input Pull-up with Interrupt" before executing the HALT instruction. The main reason for this is that the I/O may be wrongly configured due to external interference or by an unforeseen logical condition.

– For the same reason, reinitialize the level sensitiveness of each external interrupt as a precautionary measure.

– The opcode for the HALT instruction is 0x8E. To avoid an unexpected HALT instruction due to a program counter failure, it is advised to clear all occurrences of the data value 0x8E from memory. For example, avoid defining a constant in ROM with the value 0x8E.

– As the HALT instruction clears the I bits 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).

Figure 22. HALT Mode Flow Chart



1) USB and RF peripherals not functional at this frequency.

POWER SAVING MODES (Cont'd)

7.5 AUTO WAKE UP FROM HALT MODE

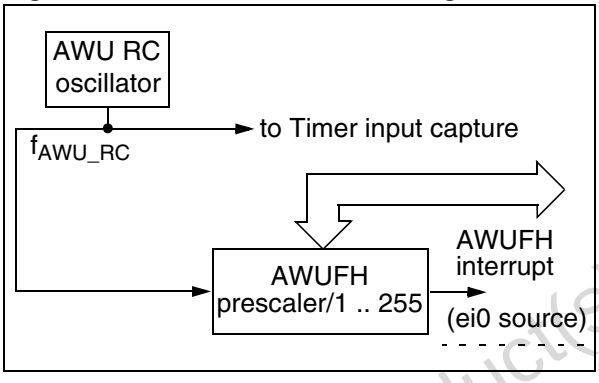
If the Auto Wake Up From Halt feature is enabled, a dedicated internal RC time base wakes up the Device from HALT mode after a programmable delay (generating an ei0 interrupt).

For more information, see Auto Wake Up From Halt Interrupt section in the Interrupts chapter.

Auto Wake Up From Halt (AWUFH) mode is similar to HALT mode with the addition of an internal RC oscillator for wake-up.

This mode is entered by executing the HALT instruction when the AWUEN bit in the AWUCSR register has been set.

Figure 23. AWUFH Mode Block Diagram



As soon as HALT mode is entered, and if the AWUEN bit has been set in the AWUCSR register, the AWU RC oscillator provides a clock signal (f_{AWU_RC}). Its frequency is divided by a fixed divider and a programmable prescaler controlled by the AWUPR register. The output of this prescaler provides the delay time. When the delay has elapsed the AWU flag is set by hardware and an interrupt wakes-up the Device from Halt mode. At the same time the main oscillator is immediately turned on. After this start-up delay, the CPU resumes operation by servicing the AWUFH interrupt. The AWU flag and its associated interrupt are cleared by software reading the AWUCSR register.

To compensate for any frequency dispersion of the AWU RC oscillator, it can be calibrated by measuring the clock frequency f_{AWU_RC} and then calculating the right prescaler value. Measurement mode is enabled by setting the AWUM bit in the AWUCSR register in Run mode. This connects f_{AWU_RC} to the ICAP1 input of the 16-bit timer, allowing the f_{AWU_RC} to be measured using the main oscillator clock as a reference timebase.

Similarities with Halt mode

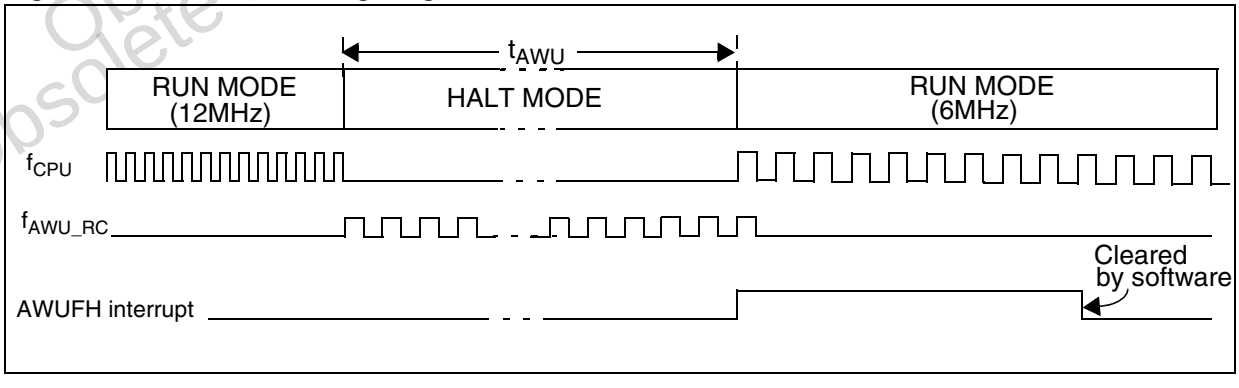
The following AWUFH mode behaviour is the same as normal Halt mode:

– The Device can exit AWUFH mode by means of any interrupt with exit from Halt capability or a reset (see Section 7.4 "HALT MODE" on page 43).

– When entering AWUFH mode, the I[1:0] bits in the CC register are forced to 10b to enable interrupts. Therefore, if an interrupt is pending, the Device wakes up immediately.

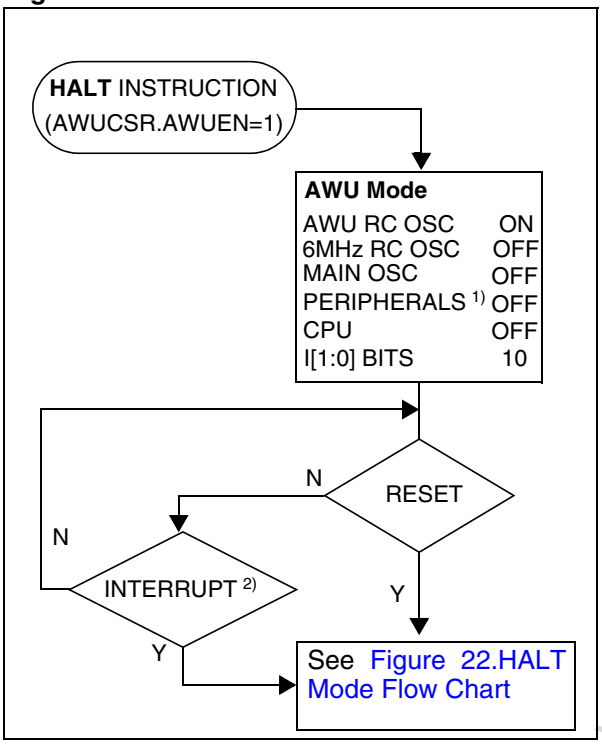
– In AWUFH mode, the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. None of the peripherals are clocked except those which get their clock supply from another clock generator (such as an external or auxiliary oscillator like the AWU oscillator).

Figure 24. AWUF Halt Timing Diagram



POWER SAVING MODES (Cont'd)

Figure 25. AWUFH Mode Flow-chart



Notes:

1. Peripheral clocked with an external clock source can still be active.

2. Only an AWUFH interrupt and some specific interrupts can exit the Device from HALT mode. Refer to Table 8, "Interrupt Mapping," on page 32 for more details.

3. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped.

POWER SAVING MODES (Cont'd)

7.5.0.1 Register Description

AWUFH CONTROL/STATUS REGISTER (AWUCSR)
Read/Write (except bit 2 read only)
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	AWUF	AWUM	AWUEN

Bits 7:3 = Reserved.

Bit 2 = **AWUF** *Auto Wake Up Flag*
This bit is set by hardware when the AWU module generates an interrupt and cleared by software on reading AWUCSR.
0: No AWU interrupt occurred
1: AWU interrupt occurred

Bit 1 = **AWUM** *Auto Wake Up Measurement*
This bit enables the AWU RC oscillator and connects its output to the ICAP1 input of the 16-bit timer. This allows the timer to be used to measure the AWU RC oscillator dispersion and then compensate this dispersion by providing the right value in the AWUPR register.
0: Measurement disabled
1: Measurement enabled

Bit 0 = **AWUEN** *Auto Wake Up From Halt Enabled*
This bit enables the Auto Wake Up From Halt feature: once HALT mode is entered, the AWUFH wakes up the Device after a time delay defined by the AWU prescaler value. It is set and cleared by software.
0: AWUFH (Auto Wake Up From Halt) mode disabled
1: AWUFH (Auto Wake Up From Halt) mode enabled

AWUFH PRESCALER REGISTER (AWUPR)
Read/Write
Reset Value: 1111 1111 (FFh)

7							0
AWUPR7	AWUPR6	AWUPR5	AWUPR4	AWUPR3	AWUPR2	AWUPR1	AWUPR0

Bits 7:0 = **AWUPR[7:0]** *Auto Wake Up Prescaler*
These 8 bits define the AWUPR Dividing factor (as explained below):

AWUPR[7:0]	Dividing factor
00h	Forbidden (See note)
01h	1
...	...
FEh	254
FFh	255

In AWU mode, the period that the MCU stays in Halt Mode (t_{AWU} in Figure 24 on page 45) is defined by

$$t_{AWU} = AWUPR \times \frac{1}{f_{AWURC}} + t_{SU(AWURC)}$$

This prescaler register can be programmed to modify the time that the MCU stays in Halt mode before waking up automatically.

Note: If 00h is written to AWUPR, either an interrupt is generated immediately after a HALT instruction, or the AWUPR remains unchanged.

Table 10. AWU Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
003Bh	**AWUCSR** Reset Value	0	0	0	0	0	AWUF	AWUM	AWUEN
003Ch	**AWUPR** Reset Value	AWUPR7 1	AWUPR6 1	AWUPR5 1	AWUPR4 1	AWUPR3 1	AWUPR2 1	AWUPR1 1	AWUPR0 1

8 I/O PORTS

8.1 INTRODUCTION

The I/O ports allow data transfer. An I/O port can contain up to 8 pins. Each pin can be programmed independently either as a digital input or digital output. In addition, specific pins may have several other functions. These functions can include external interrupt, alternate signal input/output for on-chip peripherals or analog input.

8.2 FUNCTIONAL DESCRIPTION

A Data Register (DR) and a Data Direction Register (DDR) are always associated with each port. The Option Register (OR), which allows input/output options, may or may not be implemented. The following description takes into account the OR register. Refer to the Port Configuration table for Device specific information.

An I/O pin is programmed using the corresponding bits in the DDR, DR and OR registers: bit x corresponding to pin x of the port.

Figure 26 shows the generic I/O block diagram.

8.2.1 Input Modes

Clearing the DDRx bit selects input mode. In this mode, reading its DR bit returns the digital value from that I/O pin.

If an OR bit is available, different input modes can be configured by software: floating or pull-up. Refer to I/O Port Implementation section for configuration.

Note: Writing to the DR modifies the latch value but does not change the state of the input pin.

External Interrupt Function

In input mode, external interrupts can be enabled by setting the corresponding bit in the PxEIENR register.

Falling or rising edge sensitivity is programmed independently for each interrupt vector. The External Interrupt Control Register (EICR) controls this sensitivity.

Several pins may be tied to one external interrupt vector. Refer to Pin Description to see which ports have external interrupts.

External interrupts are hardware interrupts. Fetching the corresponding interrupt vector automatically clears the request latch. Modifying the sensitivity bits will clear any pending interrupts.

8.2.2 Output Modes

Setting the DDRx bit selects output mode. Writing to the DR bits applies a digital value to the I/O through the latch. Reading the DR bits returns the previously stored value.

If an OR bit is available, different output modes can be selected by software: push-pull or open-drain. Refer to I/O Port Implementation section for configuration.

Table 11. DR value and output pin status

DR	Push-Pull	Open-Drain
0	V_{OL}	V_{OL}
1	V_{OH}	Floating

Note: When switching from input to output mode, first set the DR bit to set the correct level to be applied on the pin, then write the DDR to configure the pin as an output.

8.2.3 Alternate Functions

Many I/Os of the Device have one or more alternate functions to output. This may include output signals from, or input signals to, on-chip peripherals. The Device Pin Description table describes which peripheral signals can be input/output to which ports.

A signal coming from an on-chip peripheral can be output on an I/O. To do this, enable the on-chip peripheral as an output (enable bit in the peripheral's control register). The peripheral configures the I/O as an output and takes priority over standard I/O programming. The I/O's state is readable by addressing the corresponding I/O data register.

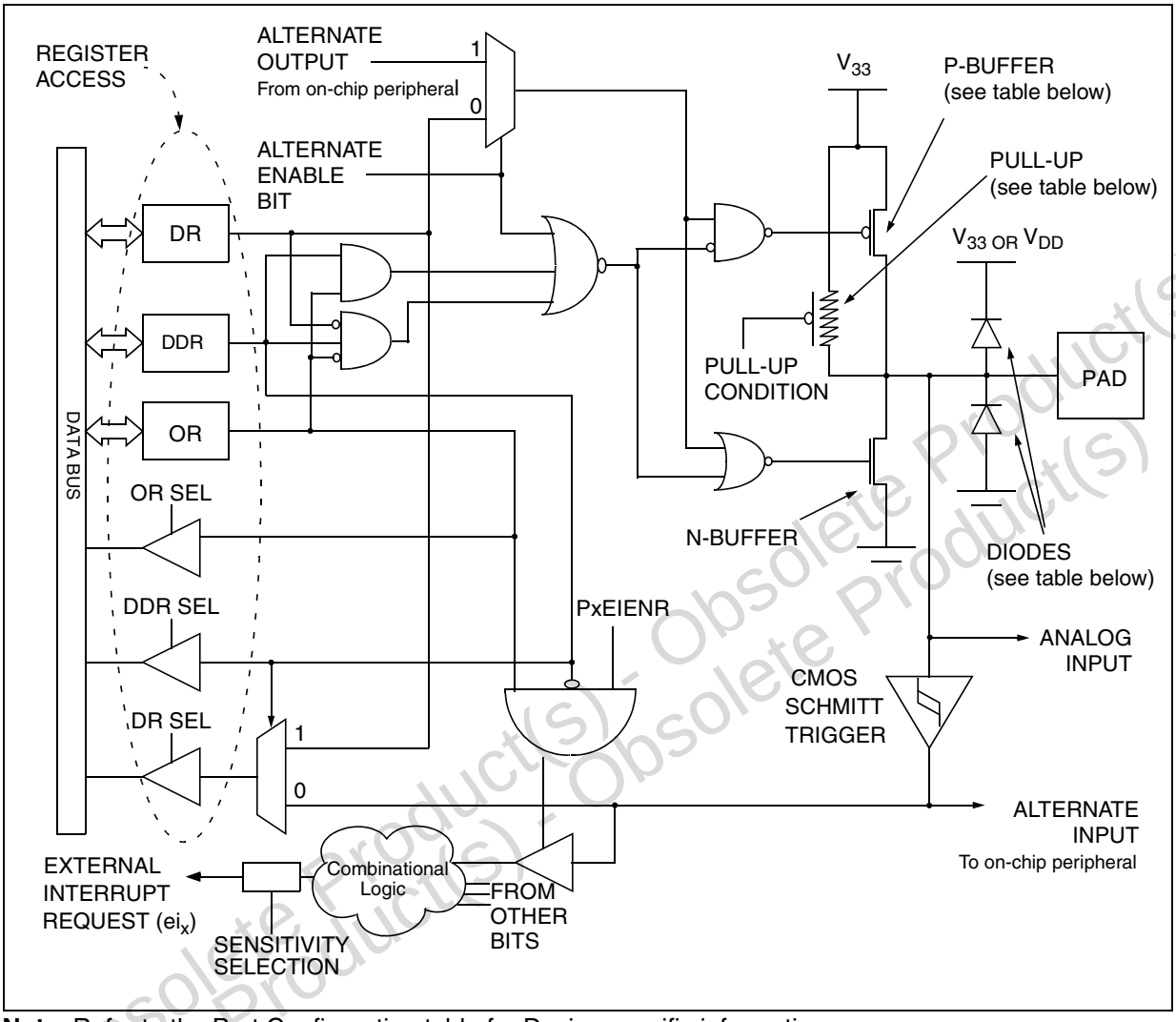
Configuring an I/O as floating enables alternate function input. It is not recommended to configure an I/O as pull-up as this will increase current consumption. Before using an I/O as an alternate input, configure it without interrupt. Otherwise spurious interrupts can occur.

Configure an I/O as input floating for an on-chip peripheral signal which can be input and output.

Caution:
I/Os which can be configured as both an analog and digital alternate function need special attention. The user must control the peripherals so that the signals do not arrive at the same time on the same pin. If an external clock is used, only the clock alternate function should be employed on that I/O pin and not the other alternate function.

I/O PORTS (Cont'd)

Figure 26. I/O Port General Block Diagram



Note: Refer to the Port Configuration table for Device specific information.

Table 12. I/O Port Mode Options

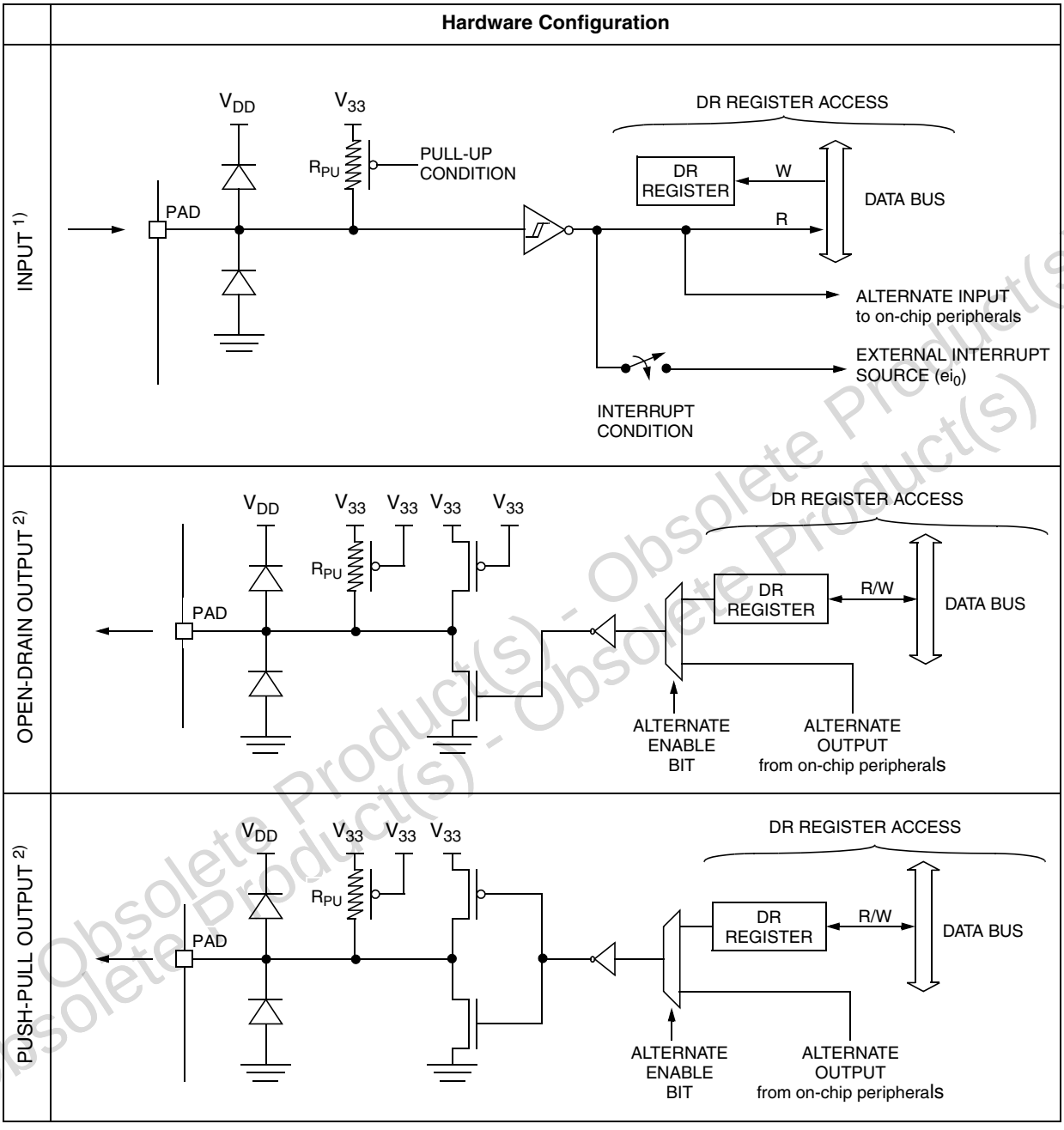
Configuration Mode		Pull-Up	P-Buffer	Diodes	
				to $V_{33\ OR}\ V_{DD}$	to V_{SS}
Input	Floating with/without Interrupt	Off	Off	On	On
	Pull-up with/without Interrupt	On			
Output	Push-pull	Off	On		
	Open Drain (logic level)		Off		
	True Open Drain	NI	NI	NI (see note)	

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

Note: The diode to $V_{33\ OR}\ V_{DD}$ V_{DD} is not implemented in the true open drain pads. A local protection between the pad and V_{OL} is implemented to protect the device against possible stress.

I/O PORTS (Cont'd)

Figure 27. I/O Port PA4:0 Configurations

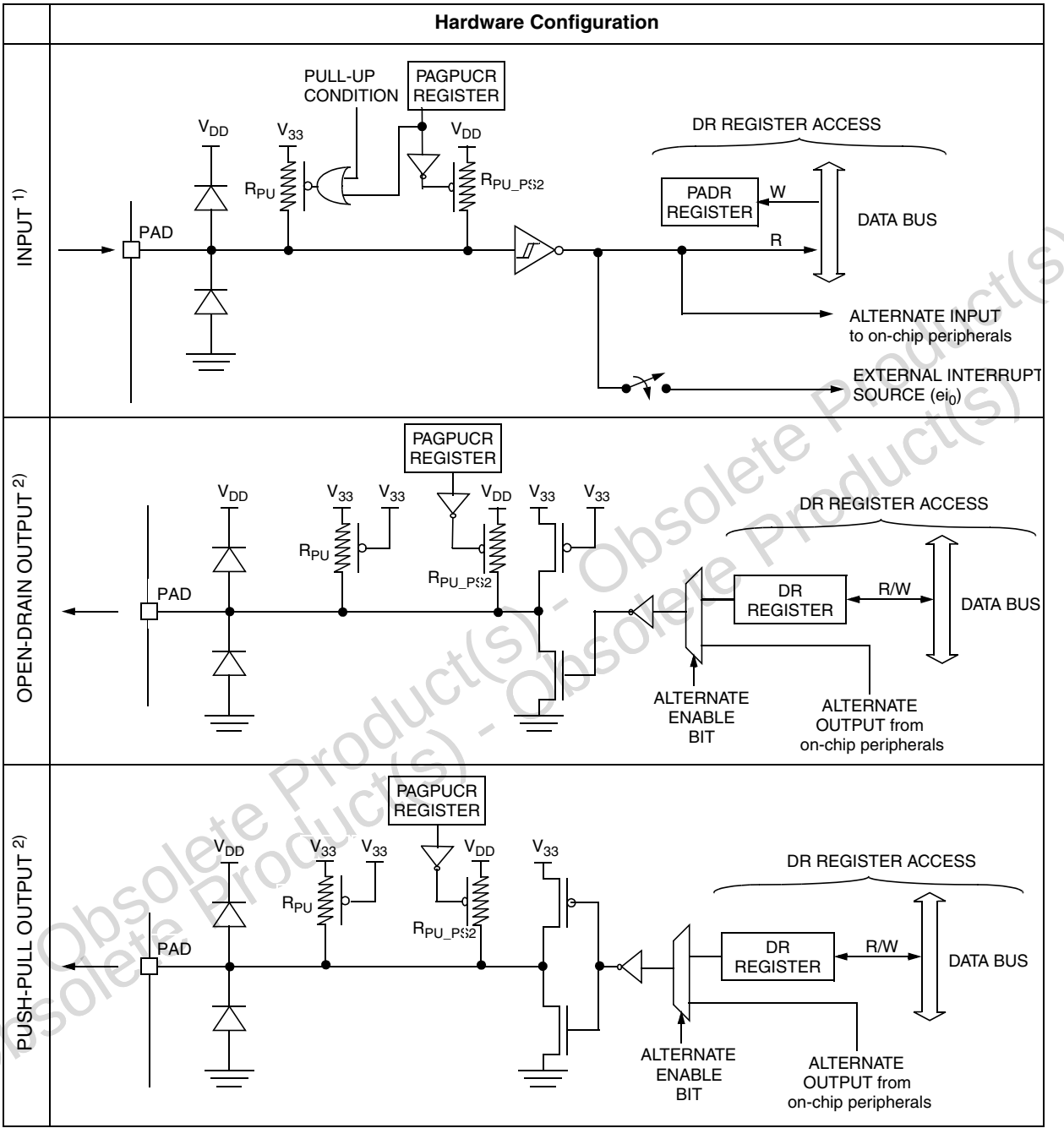


Notes:

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

I/O PORTS (Cont'd)

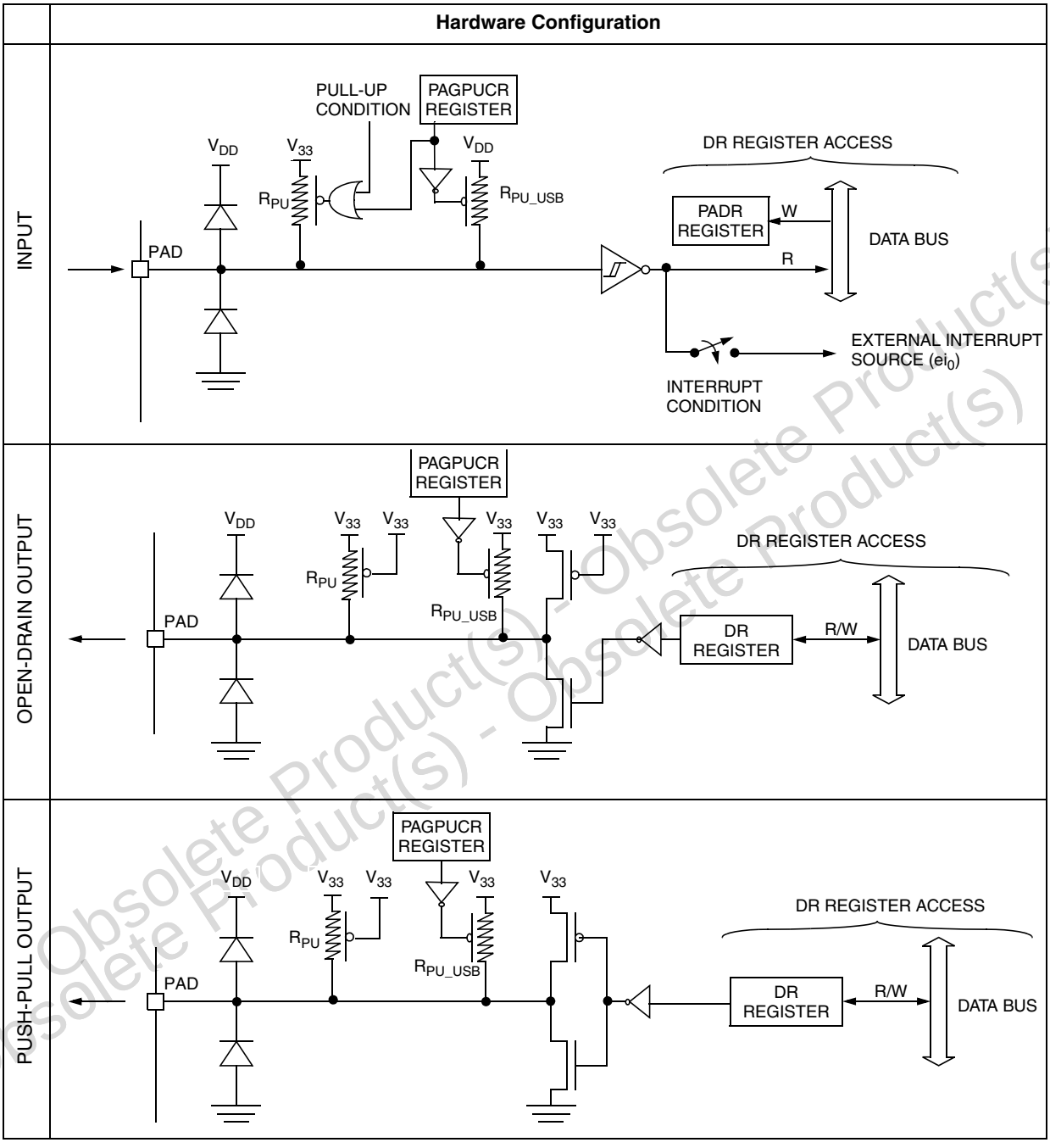
Figure 28. I/O Port PA6:5 Configurations with Special Pull-ups



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

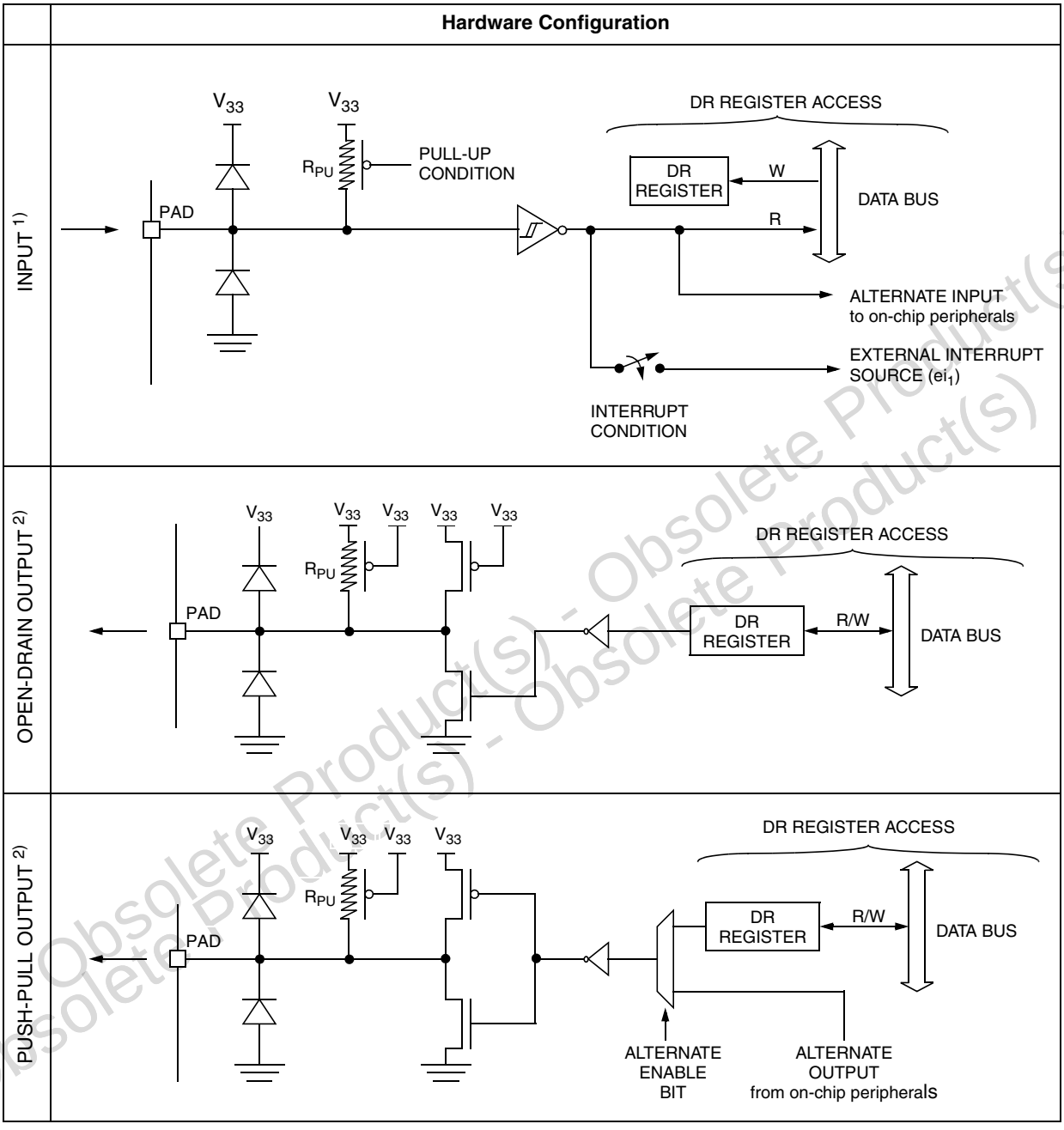
I/O PORTS (Cont'd)

Figure 29. I/O Port PA7 Configurations with Special Pull-ups



I/O PORTS (Cont'd)

Figure 30. I/O Port PB4:0 Configurations

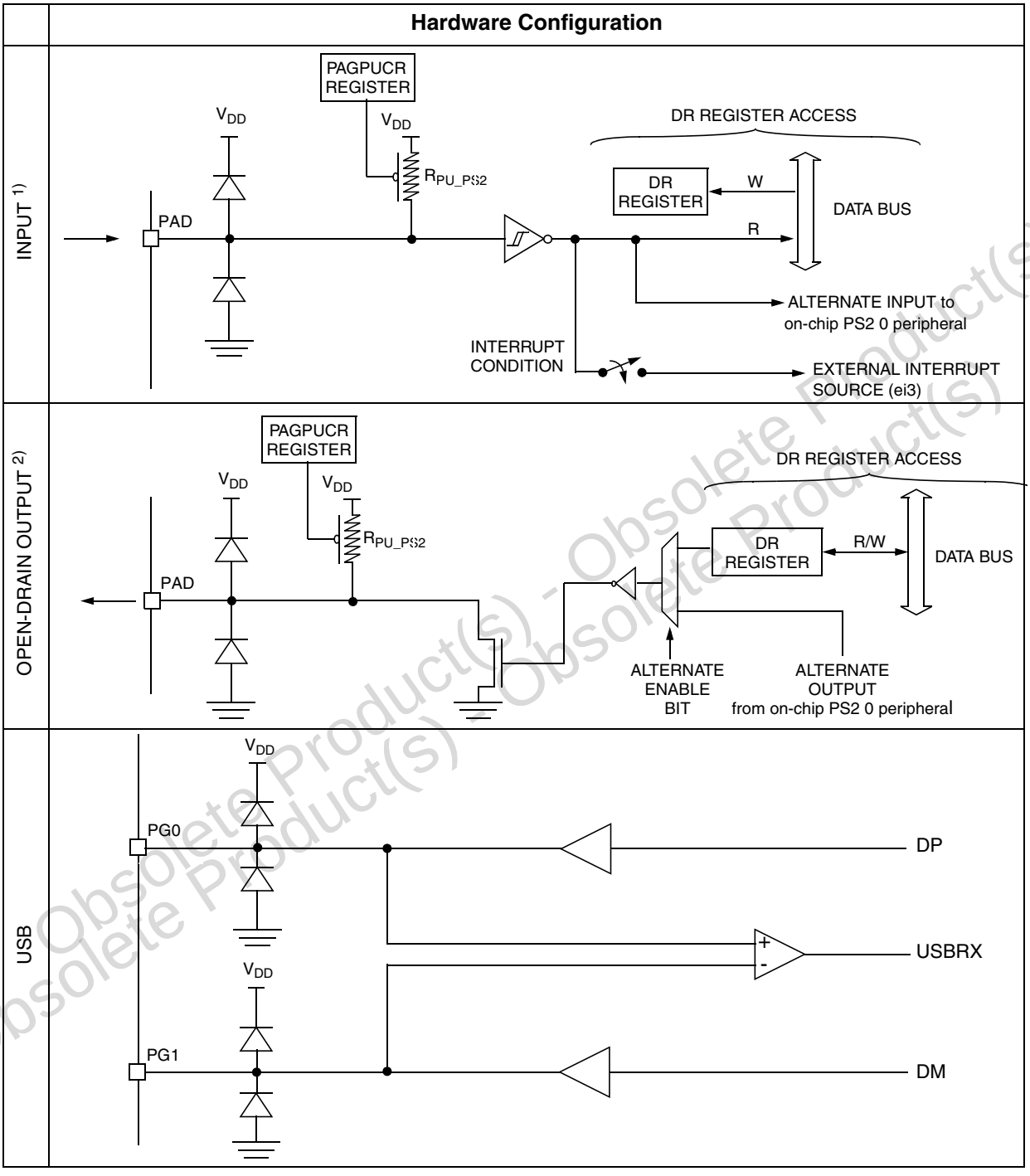


Notes:

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

I/O PORTS (Cont'd)

Figure 31. I/O Port G Configurations



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

I/O PORTS (Cont'd)

8.3 I/O PORT IMPLEMENTATION

The hardware implementation on each I/O port depends on the settings in the DDR and OR registers and specific I/O port features such as open drain.

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects.

8.4 UNUSED I/O PINS

Unused I/O pins must be connected to fixed voltage levels. Refer to the Electrical Characteristics Section.

8.5 LOW POWER MODES

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

8.6 INTERRUPTS

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and PxEIENR registers and if the I bit in the CC register is cleared (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
External interrupt on selected external event	-	DDRx PxEIENRx	Yes	Yes

I/O PORTS (Cont'd)

8.7 Ports with Special Pull-ups

I/O Ports G0, G1, A5 and A6 have special pull-ups for PS/2 compatibility. Port PA7 has a special pull-up for USB compatibility. These are enabled through the PAGPUCR register.

PORT A and G PULL UP CONTROL REGISTER (PAGPUCR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	0	0	USB-PUP	PUPA6	PUPA5	PUPG1	PUPG0

Bits 7:5 = Reserved, must be kept cleared.

Bit 4 = **USBPUP** *PA7 USB Pull-Up resistor enable*
0: USB pull-up disabled on PA7.
1: USB pull-up enabled on PA7. This replaces the PA7 OR pull-up function, if enabled.

Bit 3 = **PUPA6** *PA6 Pull-Up resistor enable*
0: Special pull-up disabled on PA6.
1: Special pull-up enabled on PA6. This replaces the PA6 OR pull-up function, if enabled.

Bit 2 = **PUPA5** *PA5 Pull-Up resistor enable*
0: Special pull-up disabled on PA5.
1: Special pull-up enabled on PA5. This replaces the PA5 OR pull-up function, if enabled.

Bit 1 = **PUPG1** *PG1 Pull-Up resistor enable*
0: Special pull-up disabled on PG1.
1: Special pull-up enabled on PG1.

Bit 0 = **PUPG0** *PG0 Pull-Up resistor enable*
0: Special pull-up disabled on PG0.
1: Special pull-up enabled on PG0.

I/O PORTS (Cont'd)

Table 13. Port Configuration

Port	Pin name	Input		Output	
		OR = 0	**OR = 1**	**OR = 0**	**OR = 1**
Port A	PA7:5	floating	pull-up	open drain	push-pull
		Special pullup if enabled in PAGPUCR register			
	PA4:0	floating	pull-up	open drain	push-pull
Port B	PB4:0	floating	pull-up	open drain	push-pull
Port G	PG1:0	floating	floating	open drain	
		Special pullup if enabled in PAGPUCR register			

Table 14. 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 except where noted		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PAOR**	0	0	0	0	0	0	1	0
0003h	**PBDR**	MSB							LSB
0004h	**PBDDR**								
0005h	**PBOR**								
0054h	**PAGPUCR**				USBPUP	PUPA6	PUPA5	PUPG1	PUPG0
0055h	**PGDR**	MSB							LSB
0056h	**PGDDR**								
0057h	**PGOR**								

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 a Device reset or a Top Level Interrupt (TLI) 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

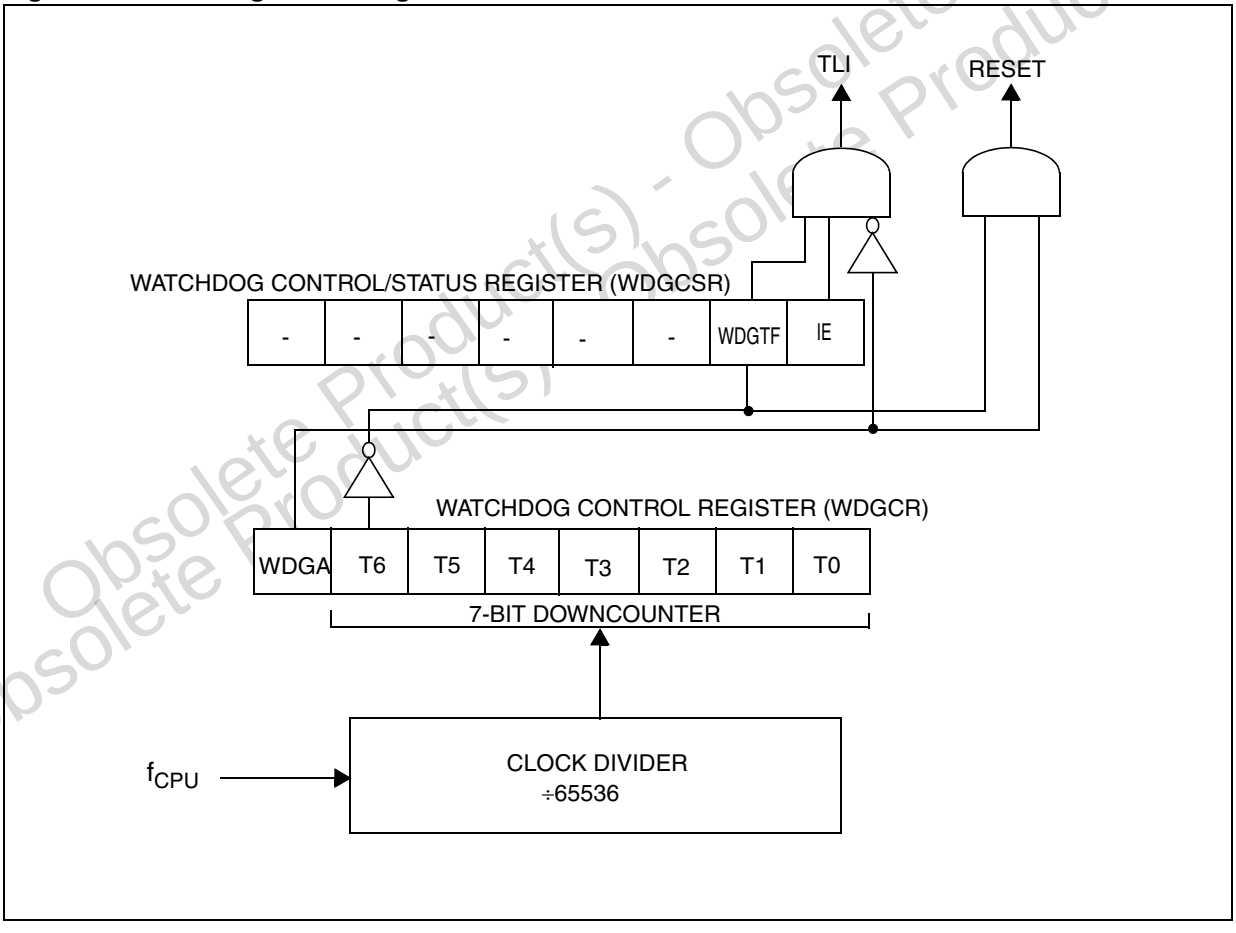
- Programmable free-running downcounter (64 increments of 65536 CPU cycles)
- Watchdog event configurable as Reset or TLI
- Watchdog event (if the WDGA or IE bit is set) when the T6 bit reaches zero
- Hardware Watchdog event selectable by option byte

9.1.3 Functional Description

The counter value stored in the WDGCR register (bits T[6:0]), is decremented every 65536 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 or IE bit is set) and when the 7-bit timer (bits T[6:0]) rolls over from 40h to 3Fh (T6 becomes cleared), it generates a Watchdog event.

Figure 32. Watchdog Block Diagram



WATCHDOG TIMER (Cont'd)

The application program must write in the WDGCR register at regular intervals during normal operation to prevent a Watchdog event. This downcounter is free-running: it counts down even if the watchdog is disabled.

9.1.4 Reset Mode

The value to be stored in the WDGCR register must be between FFh and C0h (see Table 15):

– The WDGA bit is set (watchdog reset enabled)

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

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

9.1.5 Interrupt Mode

The value to be stored in the WDGCR register must be between 7Fh and 40h (see Table 15):

– The IE bit in the WDGCSR register is set (watchdog interrupt enabled)

– The WDGA bit is cleared (watchdog reset disabled)

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

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

Table 15.Watchdog Timing (f_{CPU} = 12 MHz)

	CR Register initial value with WDGA=1	CR Register initial value with WDGA=0	WDG timeout period (ms)
Max	FFh	7Fh	349.525
Min	C0h	40h	5.461

Notes: Following a reset, the watchdog is disabled. Once activated it cannot be disabled, except by a reset.

9.1.6 Generating a Software reset

The T6 bit can be used to generate a software reset (the WDGA bit is set and the T6 bit is cleared).

9.1.7 Hardware Watchdog Options

Two options in the option byte can be used to enable hardware watchdog with Reset or TLI. If both are enabled the Reset event has priority.

Refer to the Option Byte description.

WDGHWR option	WDGHWI option	Behaviour when a Watchdog timeout occurs
0	0	If WDGA=1, a reset is generated. If WDGA=0 and IE=1, a TLI is generated.
0	1	If WDGA=1 a reset is generated. If WDGA=0, a TLI is generated.
1	x	A reset is generated.

Note: 0= option disabled, 1= option enabled

9.1.8 Low Power Modes

Mode	Description
WAIT	No effect on Watchdog
HALT	Watchdog counter frozen

9.1.9 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
TLI	WDGTF	IE	Yes	No

WATCHDOG TIMER (Cont'd)

9.1.10 Register Description

CONTROL REGISTER (WDGCR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
WDGA	T6	T5	T4	T3	T2	T1	T0

Bit 7 = **WDGA** *Watchdog Reset Activation bit*.
This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog generates a reset when T6 reaches 0.
0: Watchdog Reset disabled
1: Watchdog Reset enabled

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

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

CONTROL/STATUS REGISTER (WDGCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	WDG TF	IE

Bit 7:2 = must be kept cleared.

Bit 1 = **WDGTF** *Watchdog Timeout Flag* (read/clear).
This bit is set when T6 reaches 0. A TLI is generated when IE is set. This bit is reset by hardware when the WDGCSR register is read by software.
0: No Watchdog Timeout has occured
1: A Watchdog Timeout has occured

Note: This bit is not used if the watchdog is configured to generate a reset (WDGA=1 or hardware watchdog reset is selected by option byte).

Bit 0 = **IE** *Watchdog Interrupt Enable* (read/set).
This bit enables the generation of a Top Level interrupt when WDGTF is set. This bit is set by software and only cleared by hardware after a reset. This bit has no effect if the WDGA bit is set.
0: Watchdog Interrupt disabled
1: Watchdog Interrupt enabled

Note: This bit is forced to 1 if the Hardware Watchdog Interrupt option is chosen (by option byte).

Table 16. Watchdog Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0012h	**WDGCR**	WDGA	T6	T5	T4	T3	T2	T1	T0
	Reset Value	0	1	1	1	1	1	1	1
0013h	**WDGCSR**	-	-	-	-	-	-	WDGTF	IE
	Reset Value	0	0	0	0	0	0	0	0

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. 5.46ms (at 12MHz 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 TBUCVR register. The TBUCVR 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 TBUCVR register.

9.2.4 Programming Example

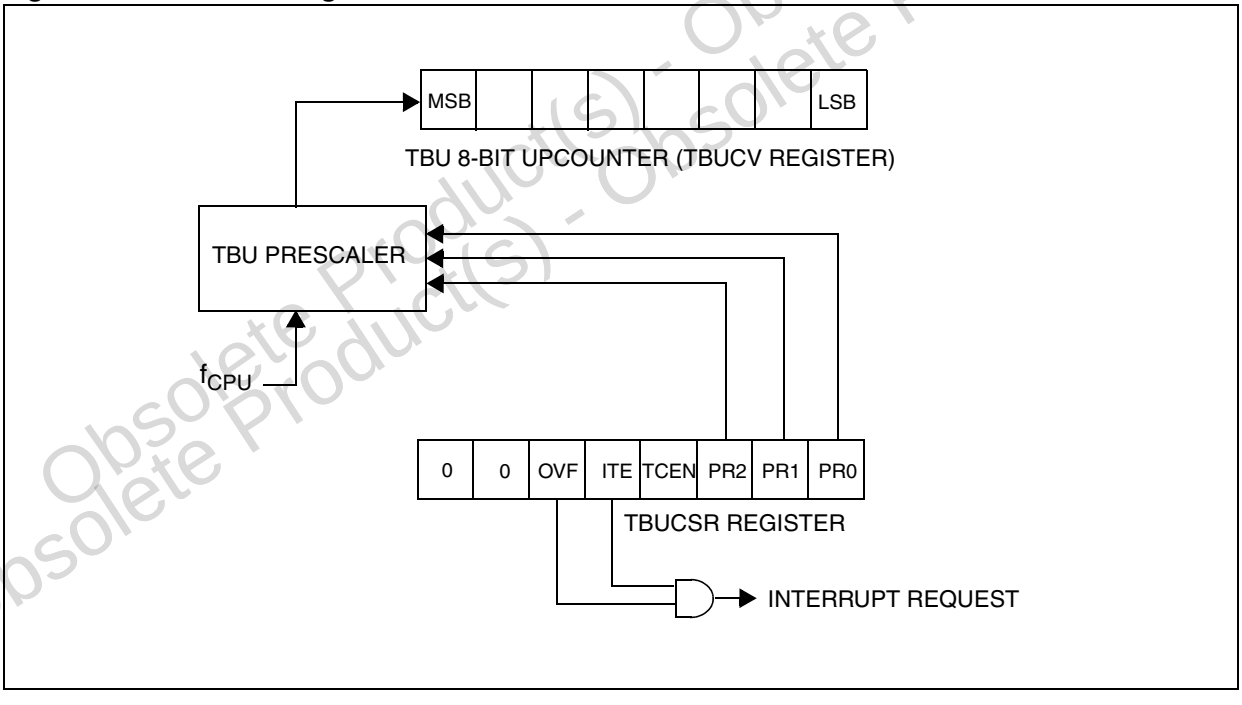
In this example, timer is required to generate an interrupt after a delay of 1 ms.

Assuming that f_{CPU} is 12 MHz and a prescaler division factor of 256 will be programmed using the PR[2:0] bits in the TBUCSR register, 1 ms = 47 TBU timer ticks.

In this case, the initial value to be loaded in the TBUCVR must be (256-47) = 209 (D1h).

```
ld A, D1h
ld TBUCVR, A  ; Initialize counter value
ld A 1Fh      ;
ld TBUCSR, A  ; Prescaler factor = 256,
              ; interrupt enable,
              ; TBU enable
```

Figure 33. 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 (TBUCVR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
CV7	CV6	CV5	CV4	CV3	CV2	CV1	CV0

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

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

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

7							0
0	0	OVF	ITE	TCEN	PR2	PR1	PR0

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

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

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

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

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

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

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

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

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

PR2	PR1	PR0	Prescaler Division Factor
0	0	0	2
0	0	1	4
0	1	0	8
0	1	1	16
1	0	0	32
1	0	1	64
1	1	0	128
1	1	1	256

TIMEBASE UNIT (Cont'd)

Table 17. TBU Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0029h	**TBUCVR**	CV7	CV6	CV5	CV4	CV3	CV2	CV1	CV0
	Reset Value	0	0	0	0	0	0	0	0
002Ah	**TBUCSR**	0	0	OVF	ITE	TCEN	PR2	PR1	PR0
	Reset Value	0	0	0	0	0	0	0	0

9.3 16-BIT TIMER

9.3.1 Introduction

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

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

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

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

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

9.3.2 Main Features

- Programmable prescaler: f_{CPU} divided by 2, 4 or 8.
- Overflow status flag and maskable interrupt
- External clock input (must be at least 4 times slower than the CPU clock speed) with the choice of active edge
- Output compare functions with
  - 2 dedicated 16-bit registers
  - 2 dedicated programmable signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Input capture functions with
  - 2 dedicated 16-bit registers
  - 2 dedicated active edge selection signals
  - 2 dedicated status flags
  - 1 dedicated maskable interrupt
- Pulse width modulation mode (PWM)
- One pulse mode
- Reduced Power Mode
- 5 alternate functions on I/O ports (ICAP1, ICAP2, OCMP1, OCMP2, EXTCLK)*

The Block Diagram is shown in Figure 34.

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

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

9.3.3 Functional Description

9.3.3.1 Counter

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

Counter Register (CR):

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

Alternate Counter Register (ACR)

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

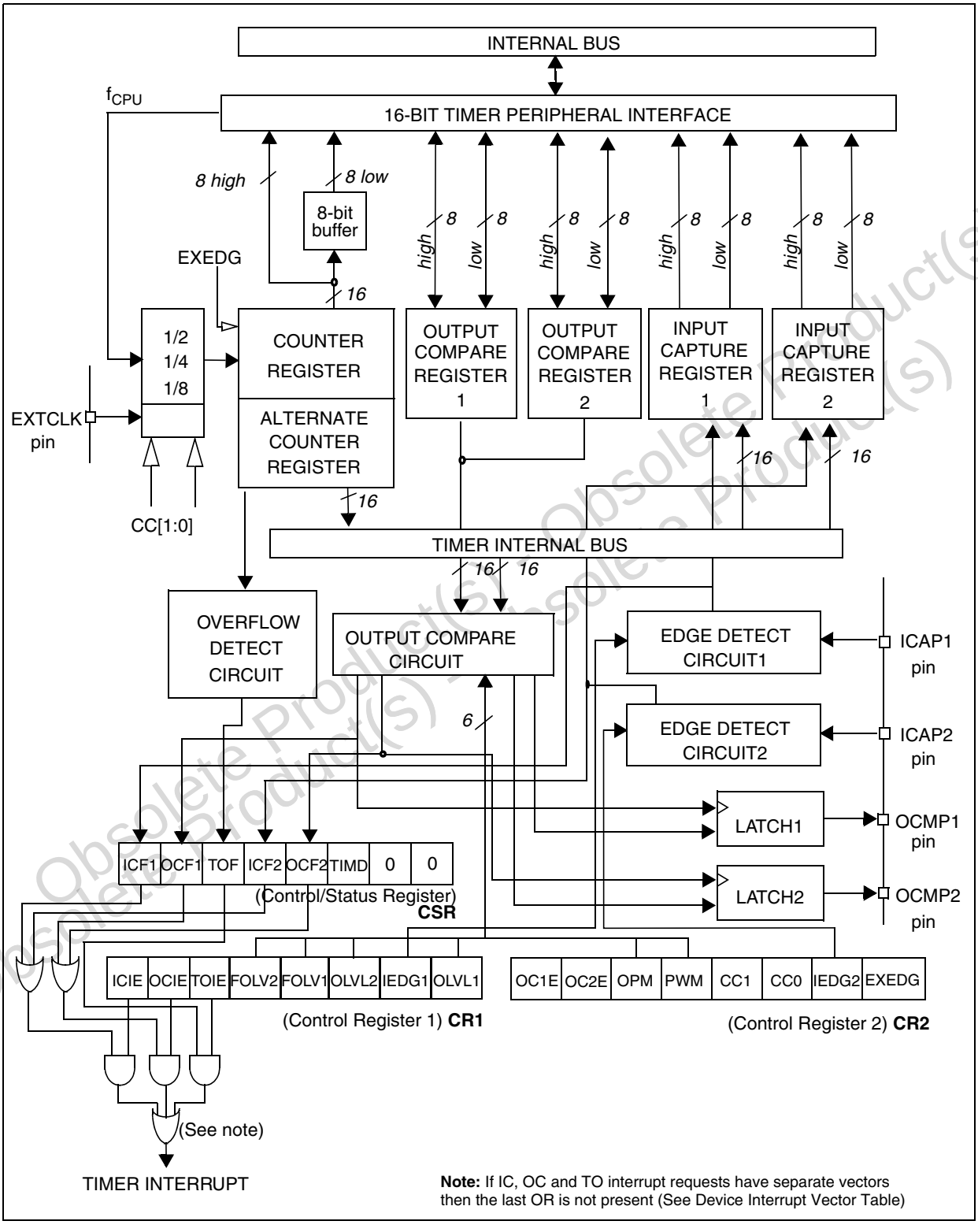
These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR), (see note at the end of paragraph titled 16-bit read sequence).

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

The timer clock depends on the clock control bits of the CR2 register, as illustrated in Table 18. The value in the counter register repeats every 131 072, 262 144 or 524 288 CPU clock cycles depending on the CC[1:0] bits.
The timer frequency can be $f_{CPU}/2$, $f_{CPU}/4$, $f_{CPU}/8$ or an external frequency.

16-BIT TIMER (Cont'd)

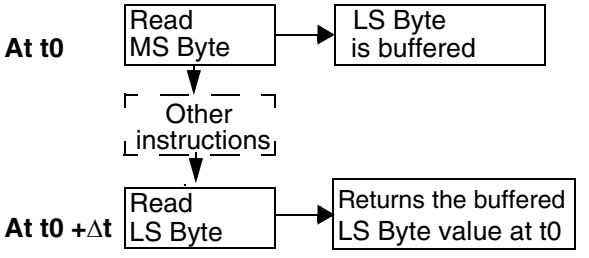
Figure 34. Timer Block Diagram



16-BIT TIMER (Cont'd)

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

Beginning of the sequence



Sequence completed

The user must read the MS Byte first, then the LS Byte value is buffered automatically.

This buffered value remains unchanged until the 16-bit read sequence is completed, even if the user reads the MS Byte several times.

After a complete reading sequence, if only the CLR register or ACLR register are read, they return the LS Byte of the count value at the time of the read.

Whatever the timer mode used (input capture, output compare, one pulse mode or PWM mode) an overflow occurs when the counter rolls over from FFFFh to 0000h then:

– The TOF bit of the SR register is set.

– A timer interrupt is generated if:

  – TOIE bit of the CR1 register is set and

  – I bit of the CC register is cleared.

If one of these conditions is false, the interrupt remains pending to be issued as soon as they are both true.

Clearing the overflow interrupt request is done in two steps:

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

Notes: The TOF bit is not cleared by accesses to ACLR register. The advantage of accessing the ACLR register rather than the CLR register is that it allows simultaneous use of the overflow function and reading the free running counter at random times (for example, to measure elapsed time) without the risk of clearing the TOF bit erroneously.

The timer is not affected by WAIT mode.

In HALT mode, the counter stops counting until the mode is exited. Counting then resumes from the previous count (Device awakened by an interrupt) or from the reset count (Device awakened by a Reset).

9.3.3.2 External Clock

The external clock (where available) is selected if CC0=1 and CC1=1 in CR2 register.

The status of the EXEDG bit in the CR2 register determines the type of level transition on the external clock pin EXTCLK that will trigger the free running counter.

The counter is synchronised with the falling edge of the internal CPU clock.

A minimum of four falling edges of the CPU clock must occur between two consecutive active edges of the external clock; thus the external clock frequency must be less than a quarter of the CPU clock frequency.

16-BIT TIMER (Cont'd)

Figure 35. Counter Timing Diagram, internal clock divided by 2

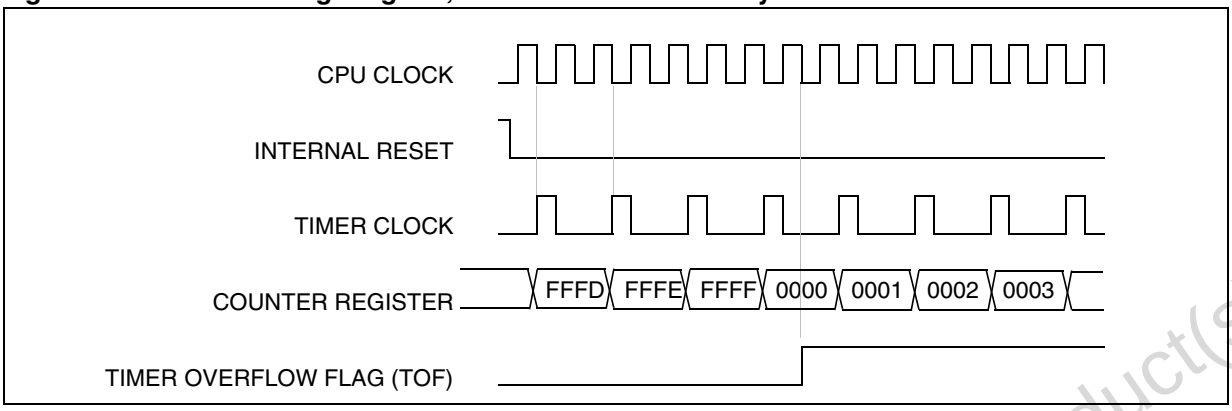


Figure 36. Counter Timing Diagram, internal clock divided by 4

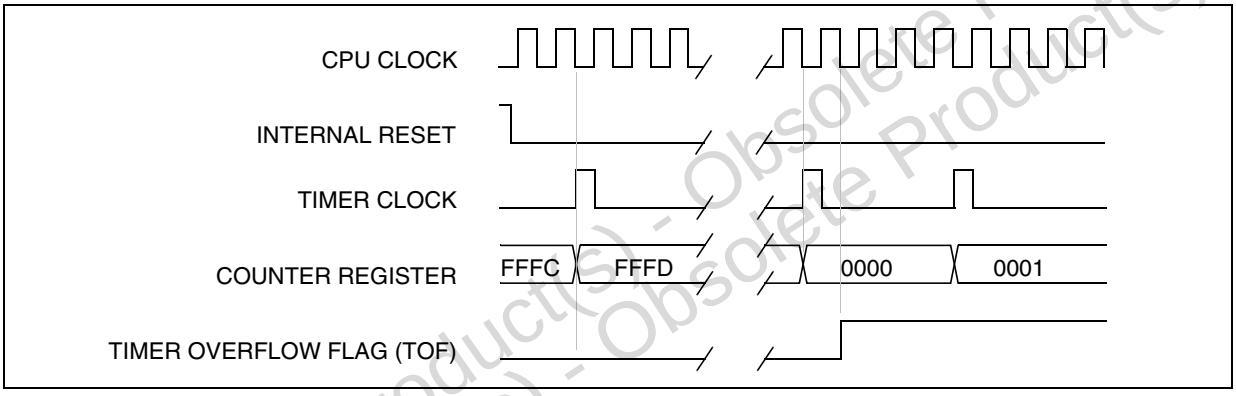
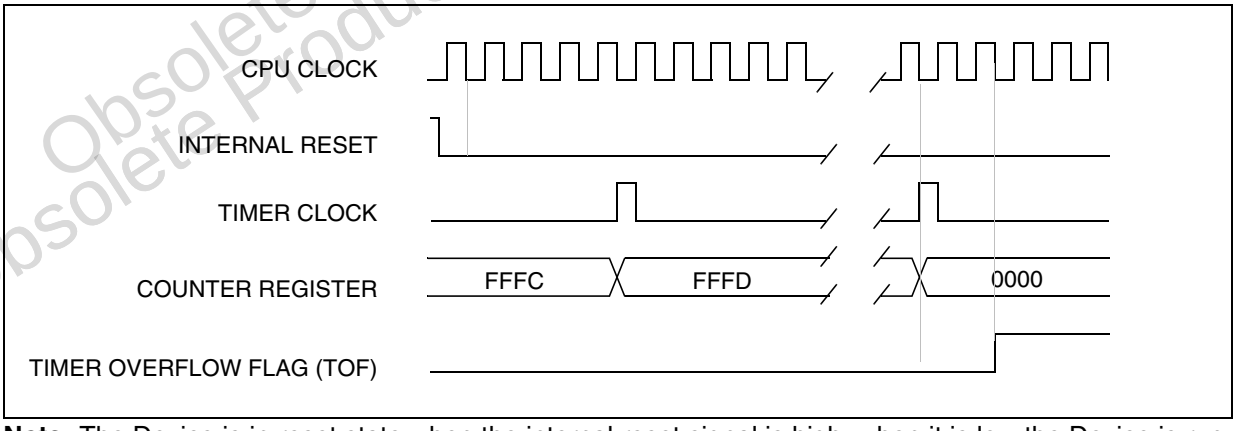


Figure 37. Counter Timing Diagram, internal clock divided by 8



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

16-BIT TIMER (Cont'd)

9.3.3.3 Input Capture

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

The two input capture 16-bit registers (IC1R and IC2R) are used to latch the value of the free running counter after a transition detected by the ICAP*i* pin (see figure 5).

	MS Byte	LS Byte
ICiR	IC*i*HR	IC*i*LR

IC*i*R register is a read-only register.

The active transition is software programmable through the IEDG*i* bit of Control Registers (CR*i*).

Timing resolution is one count of the free running counter: (f_{CPU}/CC[1:0]).

Procedure:

To use the input capture function select the following in the CR2 register:

– Select the timer clock (CC[1:0]) (see Table 18).

– Select the edge of the active transition on the ICAP2 pin with the IEDG2 bit (the ICAP2 pin must be configured as floating input).

And select the following in the CR1 register:

– Set the ICIE bit to generate an interrupt after an input capture coming from either the ICAP1 pin or the ICAP2 pin

– Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1pin must be configured as floating input).

When an input capture occurs:

– ICF*i* bit is set.

– The IC*i*R register contains the value of the free running counter on the active transition on the ICAP*i* pin (see Figure 39).

– A timer interrupt is generated if the ICIE bit is set and the I bit is cleared in the CC register. Otherwise, the interrupt remains pending until both conditions become true.

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

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

Notes:

1. After reading the IC*i*HR register, transfer of input capture data is inhibited and ICF*i* will never be set until the IC*i*LR register is also read.
2. The IC*i*R register contains the free running counter value which corresponds to the most recent input capture.
3. The 2 input capture functions can be used together even if the timer also uses the 2 output compare functions.
4. In One pulse Mode and PWM mode only the input capture 2 can be used.
5. The alternate inputs (ICAP1 & ICAP2) are always directly connected to the timer. So any transitions on these pins activate the input capture function.
   Moreover if one of the ICAP*i* pin is configured as an input and the second one as an output, an interrupt can be generated if the user toggle the output pin and if the ICIE bit is set.
   This can be avoided if the input capture function *i* is disabled by reading the IC*i*HR (see note 1).
6. The TOF bit can be used with interrupt in order to measure event that go beyond the timer range (FFFFh).

16-BIT TIMER (Cont'd)

Figure 38. Input Capture Block Diagram

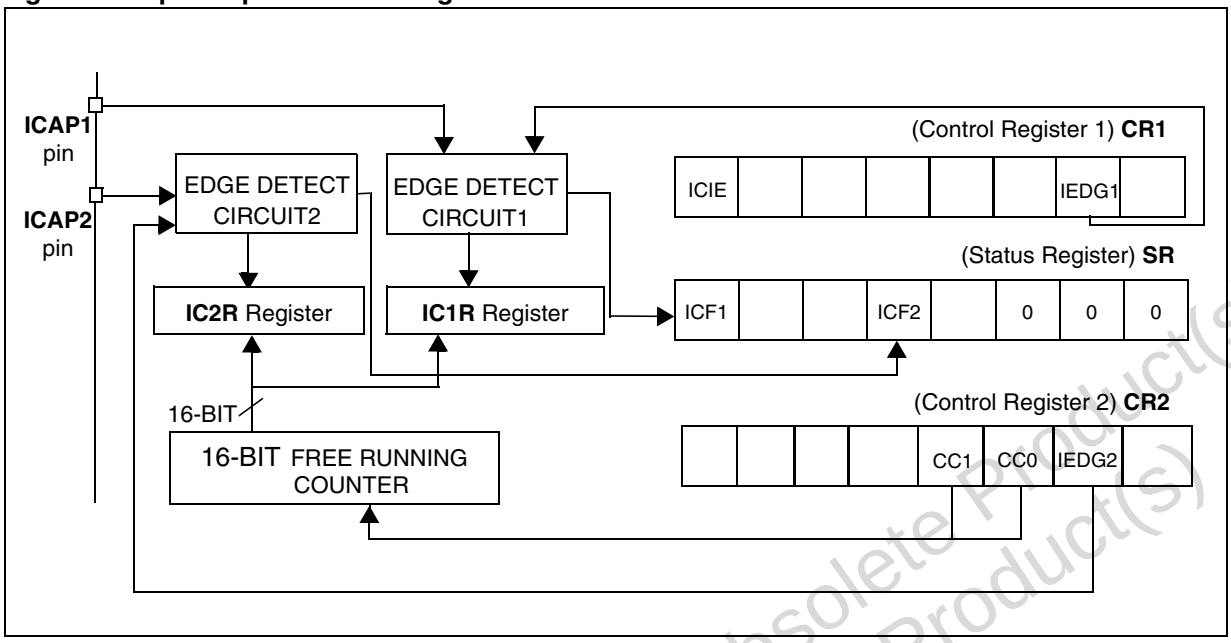
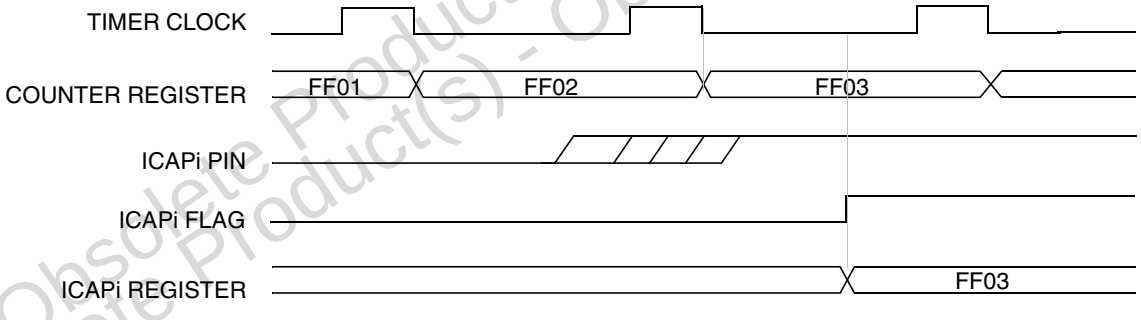


Figure 39. Input Capture Timing Diagram



Note: The active edge is the rising edge.

Note: The time between an event on the ICAPi pin and the appearance of the corresponding flag is from 2 to 3 CPU clock cycles. This depends on the moment when the ICAP event happens relative to the timer clock.

16-BIT TIMER (Cont'd)

9.3.3.4 Output Compare

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

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

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

- Assigns pins with a programmable value if the OCIE bit is set
- Sets a flag in the status register
- Generates an interrupt if enabled

Two 16-bit registers Output Compare Register 1 (OC1R) and Output Compare Register 2 (OC2R) contain the value to be compared to the counter register each timer clock cycle.

	MS Byte	LS Byte
OC*i*R	OC*i*HR	OC*i*LR

These registers are readable and writable and are not affected by the timer hardware. A reset event changes the OC*i*R value to 8000h.

Timing resolution is one count of the free running counter: ($f_{CPU/CC[1:0]}$).

Procedure:

To use the output compare function, select the following in the CR2 register:

- Set the OC*i*E bit if an output is needed then the OCMP*i* pin is dedicated to the output compare *i* signal.
- Select the timer clock (CC[1:0]) (see Table 18).

And select the following in the CR1 register:

- Select the OLVL*i* bit to applied to the OCMP*i* pins after the match occurs.
- Set the OCIE bit to generate an interrupt if it is needed.

When a match is found between OCRi register and CR register:

- OCF*i* bit is set.
- The OCMP*i* pin takes OLVL*i* bit value (OCMP*i* pin latch is forced low during reset).
- A timer interrupt is generated if the OCIE bit is set in the CR2 register and the I bit is cleared in the CC register (CC).

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

$$\Delta \text{ OC}i\text{R} = \frac{\Delta t * f_{CPU}}{\text{PRESC}}$$

Where:

Δt = Output compare period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 18)

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

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

Where:

Δt = Output compare period (in seconds)

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

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

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

The following procedure is recommended to prevent the OCF*i* bit from being set between the time it is read and the write to the OC*i*R register:

- Write to the OC*i*HR register (further compares are inhibited).
- Read the SR register (first step of the clearance of the OCF*i* bit, which may be already set).
- Write to the OC*i*LR register (enables the output compare function and clears the OCF*i* bit).

16-BIT TIMER (Cont'd)

Notes:

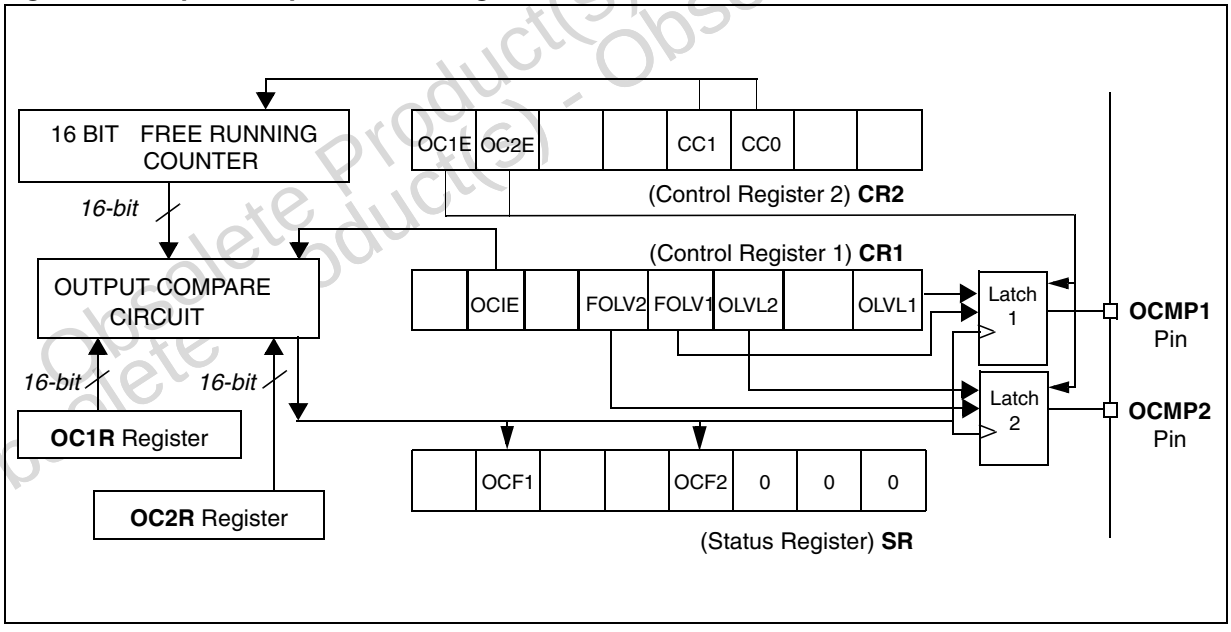
1. After a processor write cycle to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.
2. If the OC*i*E bit is not set, the OCMP*i* pin is a general I/O port and the OLVL*i* bit will not appear when a match is found but an interrupt could be generated if the OCIE bit is set.
3. When the timer clock is $f_{CPU}/2$, OCF*i* and OCMP*i* are set while the counter value equals the OC*i*R register value (see Figure 41 on page 73). This behaviour is the same in OPM or PWM mode.
   When the timer clock is $f_{CPU}/4$, $f_{CPU}/8$ or in external clock mode, OCF*i* and OCMP*i* are set while the counter value equals the OC*i*R register value plus 1 (see Figure 42 on page 73).
4. The output compare functions can be used both for generating external events on the OCMP*i* pins even if the input capture mode is also used.
5. The value in the 16-bit OC*i*R register and the OLV*i* bit should be changed after each successful comparison in order to control an output waveform or establish a new elapsed timeout.

Forced Compare Output capability

When the FOLV*i* bit is set by software, the OLVL*i* bit is copied to the OCMP*i* pin. The OLV*i* bit has to be toggled in order to toggle the OCMP*i* pin when it is enabled (OC*i*E bit=1). The OCF*i* bit is then not set by hardware, and thus no interrupt request is generated.

FOLVL*i* bits have no effect in both one pulse mode and PWM mode.

Figure 40. Output Compare Block Diagram



16-BIT TIMER (Cont'd)

Figure 41. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/2$

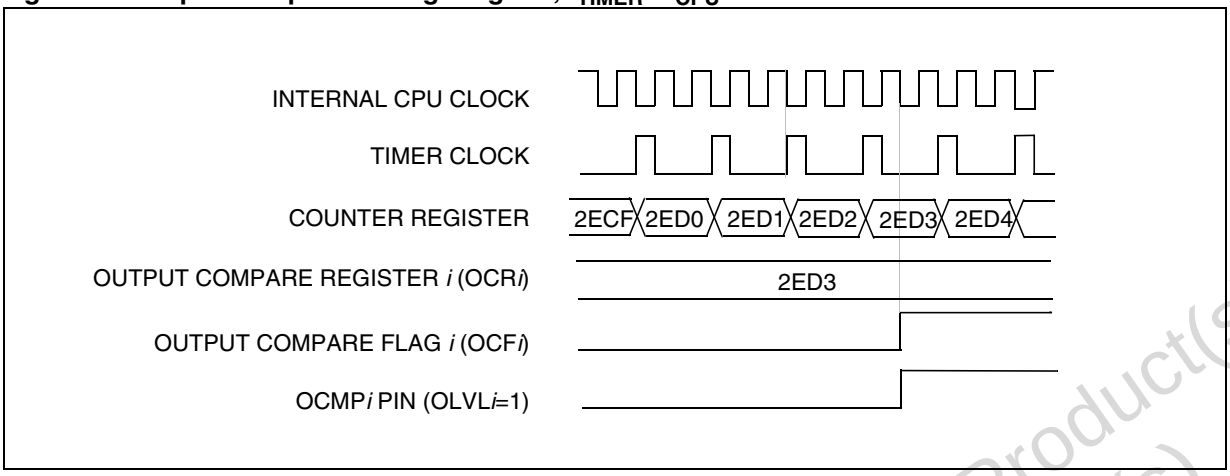
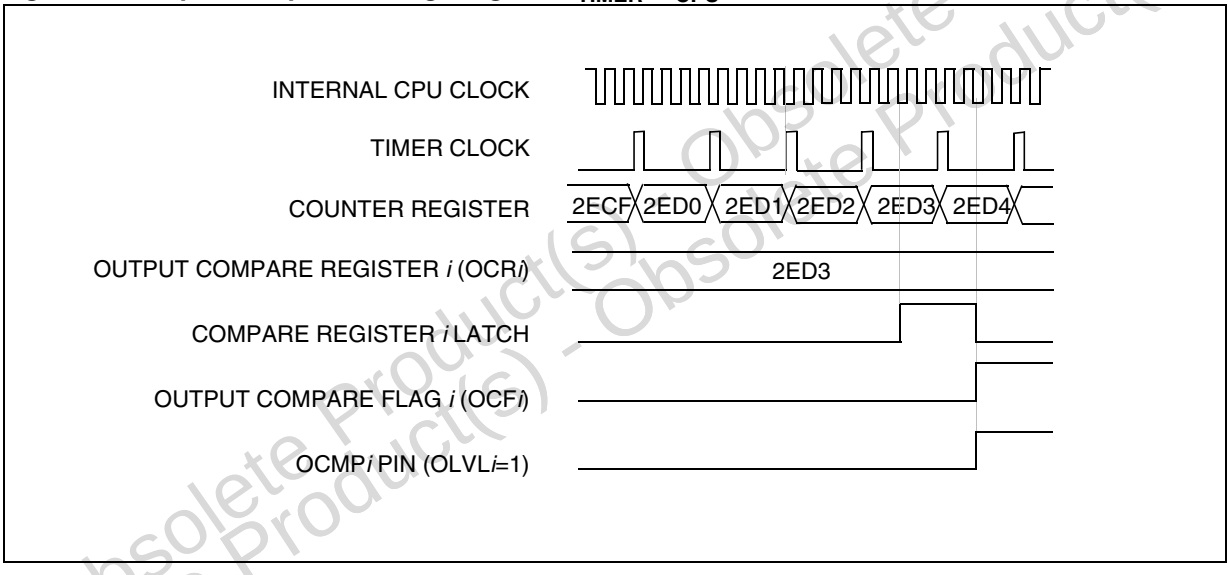


Figure 42. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/4$



16-BIT TIMER (Cont'd)

9.3.3.5 One Pulse Mode

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

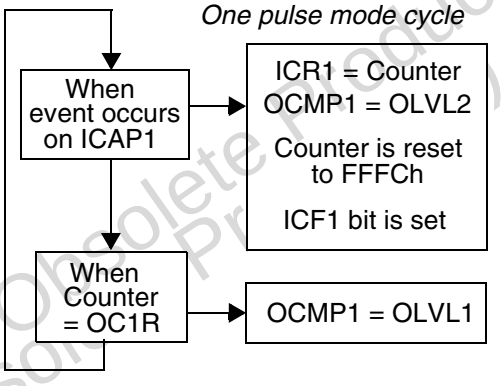
The one pulse mode uses the Input Capture1 function and the Output Compare1 function.

Procedure:

To use one pulse mode:

1. Load the OC1R register with the value corresponding to the length of the pulse (see the formula in the opposite column).
2. Select the following in the CR1 register:
   - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after the pulse.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin during the pulse.
   - Select the edge of the active transition on the ICAP1 pin with the IEDG1 bit (the ICAP1 pin must be configured as floating input).
3. Select the following in the CR2 register:
   - Set the OC1E bit, the OCMP1 pin is then dedicated to the Output Compare 1 function.
   - Set the OPM bit.
   - Select the timer clock CC[1:0] (see Table 18).

One pulse mode cycle



When a valid event occurs on the ICAP1 pin, the counter value is loaded in the ICR1 register. The counter is then initialized to FFFCh, the OLVL2 bit is output on the OCMP1 pin and the ICF1 bit is set.

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

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

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

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

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

Where:

t = Pulse period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on the CC[1:0] bits, see Table 18)

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

$$OCiR = t * f_{EXT} - 5$$

Where:

t = Pulse period (in seconds)

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

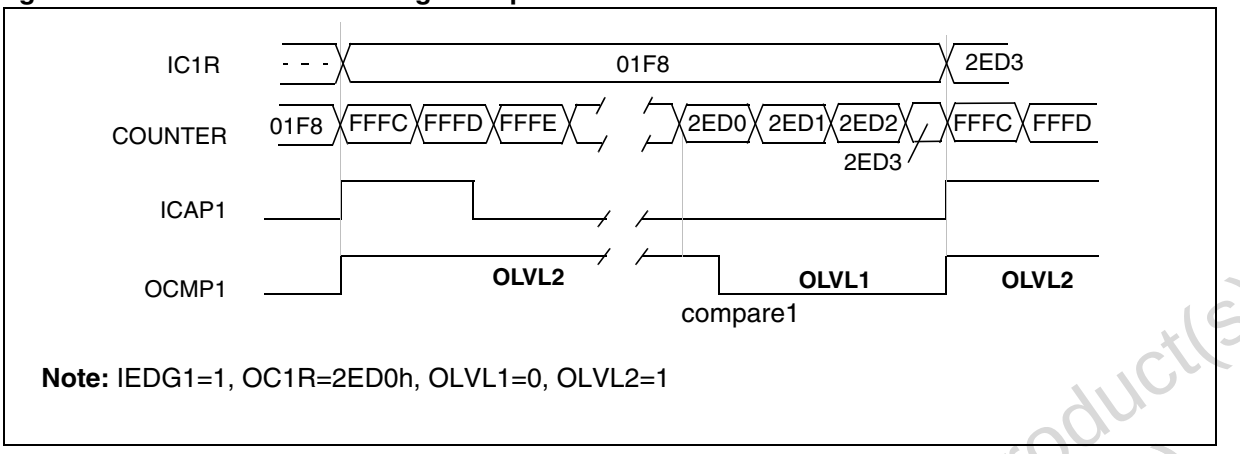
When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin, (See Figure 43).

Notes:

1. The OCF1 bit cannot be set by hardware in one pulse mode but the OCF2 bit can generate an Output Compare interrupt.
2. When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.
3. If OLVL1=OLVL2 a continuous signal will be seen on the OCMP1 pin.
4. The ICAP1 pin can not be used to perform input capture. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each time a valid edge occurs on the ICAP1 pin and ICF1 can also generates interrupt if ICIE is set.
5. When one pulse mode is used OC1R is dedicated to this mode. Nevertheless OC2R and OCF2 can be used to indicate a period of time has been elapsed but cannot generate an output waveform because the level OLVL2 is dedicated to the one pulse mode.

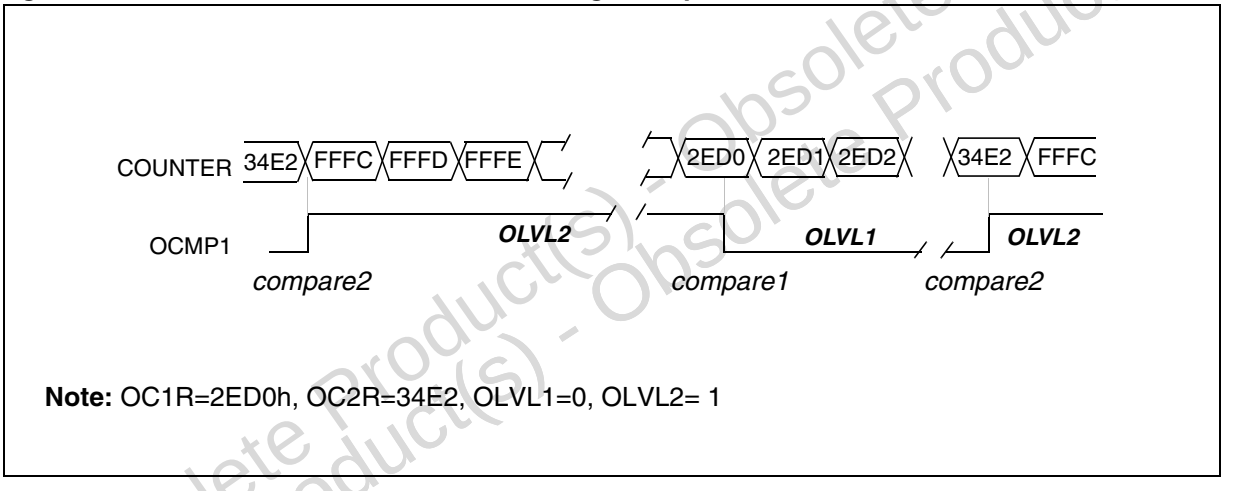
16-BIT TIMER (Cont'd)

Figure 43. One Pulse Mode Timing Example



Note: IEDG1=1, OC1R=2ED0h, OLVL1=0, OLVL2=1

Figure 44. Pulse Width Modulation Mode Timing Example



Note: OC1R=2ED0h, OC2R=34E2, OLVL1=0, OLVL2= 1

16-BIT TIMER (Cont'd)

9.3.3.6 Pulse Width Modulation Mode

Pulse Width Modulation (PWM) mode enables the generation of a signal with a frequency and pulse length determined by the value of the OC1R and OC2R registers.

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

In PWM mode, double buffering is implemented on the output compare registers. Any new values written in the OC1R and OC2R registers are loaded in their respective shadow registers (double buffer) only at the end of the PWM period (OC2) to avoid spikes on the PWM output pin (OCMP1). The shadow registers contain the reference values for comparison in PWM "double buffering" mode.

Note: There is a locking mechanism for transferring the OCiR value to the buffer. After a write to the OCiHR register, transfer of the new compare value to the buffer is inhibited until OCiLR is also written.

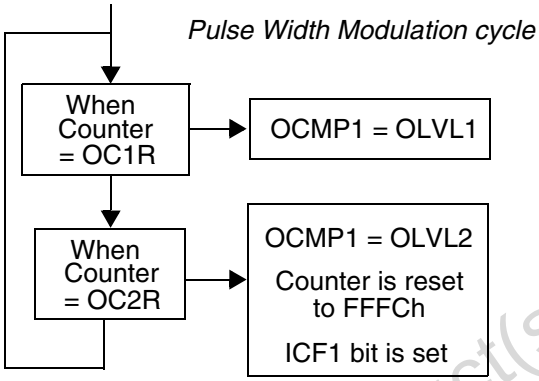
Unlike in Output Compare mode, the compare function is always enabled in PWM mode.

Procedure

To use pulse width modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if (OLVL1=0 and OLVL2=1) using the formula in the opposite column.
3. Select the following in the CR1 register:
   - Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC1R register.
   - Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC2R register.
4. Select the following in the CR2 register:
   - Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.
   - Set the PWM bit.
   - Select the timer clock (CC[1:0]) (see Table 18).

Pulse Width Modulation cycle



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

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

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

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

Where:

t = Signal or pulse period (in seconds)

f_{CPU} = CPU clock frequency (in hertz)

PRESC = Timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see Table 18)

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

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

Where:

t = Signal or pulse period (in seconds)

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

The Output Compare 2 event causes the counter to be initialized to FFFCh (See Figure 44)

Notes:

1. The OCF1 and OCF2 bits cannot be set by hardware in PWM mode therefore the Output Compare interrupt is inhibited.
2. The ICF1 bit is set by hardware when the counter reaches the OC2R value and can produce a timer interrupt if the ICIE bit is set and the I bit is cleared.

16-BIT TIMER (Cont'd)

3. In PWM mode the ICAP1 pin can not be used to perform input capture because it is disconnected to the timer. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each period and ICF1 can also generates interrupt if ICIE is set.
4. When the Pulse Width Modulation (PWM) and One Pulse Mode (OPM) bits are both set, the PWM mode is the only active one.

9.3.4 Low Power Modes

Mode	Description
WAIT	No effect on 16-bit Timer. Timer interrupts cause the Device to exit from WAIT mode.
HALT	16-bit Timer registers are frozen. In HALT mode, the counter stops counting until Halt mode is exited. Counting resumes from the previous count when the Device is woken up by an interrupt with "exit from HALT mode" capability or from the counter reset value when the Device is woken up by a RESET. If an input capture event occurs on the ICAP*i* pin, the input capture detection circuitry is armed. Consequently, when the Device is woken up by an interrupt with "exit from HALT mode" capability, the ICF*i* bit is set, and the counter value present when exiting from HALT mode is captured into the IC*i*R register.

9.3.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Input Capture 1 event/Counter reset in PWM mode	ICF1	ICIE	Yes	No
Input Capture 2 event	ICF2		Yes	No
Output Compare 1 event (not available in PWM mode)	OCF1	OCIE	Yes	No
Output Compare 2 event (not available in PWM mode)	OCF2		Yes	No
Timer Overflow event	TOF	TOIE	Yes	No

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

9.3.6 Summary of Timer modes

MODES	AVAILABLE RESOURCES			
	Input Capture 1	Input Capture 2	Output Compare 1	Output Compare 2
Input Capture (1 and/or 2)	Yes	Yes	Yes	Yes
Output Compare (1 and/or 2)	Yes	Yes	Yes	Yes
One Pulse Mode	No	Not Recommended[1)]	No	Partially [2)]
PWM Mode	No	Not Recommended[3)]	No	No

1) See note 4 in Section 9.3.3.5 "One Pulse Mode" on page 74

2) See note 5 in Section 9.3.3.5 "One Pulse Mode" on page 74

3) See note 4 in Section 9.3.3.6 "Pulse Width Modulation Mode" on page 76

16-BIT TIMER (Cont'd)

9.3.7 Register Description

Each Timer is associated with three control and status registers, and with six pairs of data registers (16-bit values) relating to the two input captures, the two output compares, the counter and the alternate counter.

CONTROL REGISTER 1 (CR1)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1

Bit 7 = **ICIE** *Input Capture Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the ICF1 or ICF2 bit of the SR register is set.

Bit 6 = **OCIE** *Output Compare Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is generated whenever the OCF1 or OCF2 bit of the SR register is set.

Bit 5 = **TOIE** *Timer Overflow Interrupt Enable.*
0: Interrupt is inhibited.
1: A timer interrupt is enabled whenever the TOF bit of the SR register is set.

Bit 4 = **FOLV2** *Forced Output Compare 2.*
This bit is set and cleared by software.
0: No effect on the OCMP2 pin.
1: Forces the OLVL2 bit to be copied to the OCMP2 pin, if the OC2E bit is set and even if there is no successful comparison.

Bit 3 = **FOLV1** *Forced Output Compare 1.*
This bit is set and cleared by software.
0: No effect on the OCMP1 pin.
1: Forces OLVL1 to be copied to the OCMP1 pin, if the OC1E bit is set and even if there is no successful comparison.

Bit 2 = **OLVL2** *Output Level 2.*
This bit is copied to the OCMP2 pin whenever a successful comparison occurs with the OC2R register and OCxE is set in the CR2 register. This value is copied to the OCMP1 pin in One Pulse Mode and Pulse Width Modulation mode.

Bit 1 = **IEDG1** *Input Edge 1.*
This bit determines which type of level transition on the ICAP1 pin will trigger the capture.
0: A falling edge triggers the capture.
1: A rising edge triggers the capture.

Bit 0 = **OLVL1** *Output Level 1.*
The OLVL1 bit is copied to the OCMP1 pin whenever a successful comparison occurs with the OC1R register and the OC1E bit is set in the CR2 register.

16-BIT TIMER (Cont'd)

CONTROL REGISTER 2 (CR2)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG

Bit 7 = **OC1E** *Output Compare 1 Pin Enable.*
This bit is used only to output the signal from the timer on the OCMP1 pin (OLV1 in Output Compare mode, both OLV1 and OLV2 in PWM and one-pulse mode). Whatever the value of the OC1E bit, the Output Compare 1 function of the timer remains active.
0: OCMP1 pin alternate function disabled (I/O pin free for general-purpose I/O).
1: OCMP1 pin alternate function enabled.

Bit 6 = **OC2E** *Output Compare 2 Pin Enable.*
This bit is used only to output the signal from the timer on the OCMP2 pin (OLV2 in Output Compare mode). Whatever the value of the OC2E bit, the Output Compare 2 function of the timer remains active.
0: OCMP2 pin alternate function disabled (I/O pin free for general-purpose I/O).
1: OCMP2 pin alternate function enabled.

Bit 5 = **OPM** *One Pulse Mode.*
0: One Pulse Mode is not active.
1: One Pulse Mode is active, the ICAP1 pin can be used to trigger one pulse on the OCMP1 pin; the active transition is given by the IEDG1 bit. The length of the generated pulse depends on the contents of the OC1R register.

Bit 4 = **PWM** *Pulse Width Modulation.*
0: PWM mode is not active.
1: PWM mode is active, the OCMP1 pin outputs a programmable cyclic signal; the length of the pulse depends on the value of OC1R register; the period depends on the value of OC2R register.

Bit 3, 2 = **CC[1:0]** *Clock Control.*

The timer clock mode depends on these bits:

Table 18. Clock Control Bits

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

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

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

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

16-BIT TIMER (Cont'd)

CONTROL/STATUS REGISTER (CSR)

Read Only

Reset Value: 0000 0000 (00h)

The three least significant bits are not used.

7							0
ICF1	OCF1	TOF	ICF2	OCF2	TIMD	0	0

Bit 7 = **ICF1** *Input Capture Flag 1.*
0: No input capture (reset value).
1: An input capture has occurred on the ICAP1 pin or the counter has reached the OC2R value in PWM mode. To clear this bit, first read the SR register, then read or write the low byte of the IC1R (IC1LR) register.

Bit 6 = **OCF1** *Output Compare Flag 1.*
0: No match (reset value).
1: The content of the free running counter has matched the content of the OC1R register. To clear this bit, first read the SR register, then read or write the low byte of the OC1R (OC1LR) register.

Bit 5 = **TOF** *Timer Overflow Flag.*
0: No timer overflow (reset value).
1: The free running counter rolled over from FFFFh to 0000h. To clear this bit, first read the SR register, then read or write the low byte of the CR (CLR) register.

Note: Reading or writing the ACLR register does not clear TOF.

Bit 4 = **ICF2** *Input Capture Flag 2.*
0: No input capture (reset value).
1: An input capture has occurred on the ICAP2 pin. To clear this bit, first read the SR register, then read or write the low byte of the IC2R (IC2LR) register.

Bit 3 = **OCF2** *Output Compare Flag 2.*
0: No match (reset value).
1: The content of the free running counter has matched the content of the OC2R register. To clear this bit, first read the SR register, then read or write the low byte of the OC2R (OC2LR) register.

Bit 2 = **TIMD** *Timer disable.*
This bit is set and cleared by software. When set, it freezes the timer prescaler and counter and disabled the output functions (OCMP1 and OCMP2 pins) to reduce power consumption. Access to the timer registers is still available, allowing the timer configuration to be changed while it is disabled.
0: Timer enabled
1: Timer prescaler, counter and outputs disabled

Bits 1:0 = Reserved, must be kept cleared.

16-BIT TIMER (Cont'd)

INPUT CAPTURE 1 HIGH REGISTER (IC1HR)

Read Only
Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the input capture 1 event).

7							0
MSB							LSB

INPUT CAPTURE 1 LOW REGISTER (IC1LR)

Read Only
Reset Value: Undefined

This is an 8-bit read only register that contains the low part of the counter value (transferred by the input capture 1 event).

7							0
MSB							LSB

OUTPUT COMPARE 1 HIGH REGISTER (OC1HR)

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

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

7							0
MSB							LSB

OUTPUT COMPARE 1 LOW REGISTER (OC1LR)

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

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

7							0
MSB							LSB

16-BIT TIMER (Cont'd)

OUTPUT COMPARE 2 HIGH REGISTER (OC2HR)

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

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

7							0
MSB							LSB

OUTPUT COMPARE 2 LOW REGISTER (OC2LR)

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

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

7							0
MSB							LSB

COUNTER HIGH REGISTER (CHR)

Read Only
Reset Value: 1111 1111 (FFh)

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

7							0
MSB							LSB

COUNTER LOW REGISTER (CLR)

Read Only
Reset Value: 1111 1100 (FCh)

This is an 8-bit register that contains the low part of the counter value. A write to this register resets the counter. An access to this register after accessing the CSR register clears the TOF bit.

7							0
MSB							LSB

ALTERNATE COUNTER HIGH REGISTER (ACHR)

Read Only
Reset Value: 1111 1111 (FFh)

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

7							0
MSB							LSB

ALTERNATE COUNTER LOW REGISTER (ACLR)

Read Only
Reset Value: 1111 1100 (FCh)

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

7							0
MSB							LSB

INPUT CAPTURE 2 HIGH REGISTER (IC2HR)

Read Only
Reset Value: Undefined

This is an 8-bit read only register that contains the high part of the counter value (transferred by the Input Capture 2 event).

7							0
MSB							LSB

INPUT CAPTURE 2 LOW REGISTER (IC2LR)

Read Only
Reset Value: Undefined

This is an 8-bit read only register that contains the low part of the counter value (transferred by the Input Capture 2 event).

7							0
MSB							LSB

16-BIT TIMER (Cont'd)

Table 19. 16-Bit Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
001Ah	**CR2**	OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG
	Reset Value	0	0	0	0	0	0	0	0
001Bh	**CR1**	ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
	Reset Value	0	0	0	0	0	0	0	0
001Ch	**CSR**	ICF1	OCF1	TOF	ICF2	OCF2	TIMD	-	-
	Reset Value	-	-	-	-	-	0	-	-
001Dh	**IC1HR**	MSB							LSB
	Reset Value	-	-	-	-	-	-	-	-
001Eh	**IC1LR**	MSB							LSB
	Reset Value	-	-	-	-	-	-	-	-
001Fh	**OC1HR**	MSB							LSB
	Reset Value	1	0	0	0	0	0	0	0
0020h	**OC1LR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0
0021h	**CHR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	1	1
0022h	**CLR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	0	0
0023h	**ACHR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	1	1
0024h	**ACLR**	MSB							LSB
	Reset Value	1	1	1	1	1	1	0	0
0025h	**IC2HR**	MSB							LSB
	Reset Value	-	-	-	-	-	-	-	-
0026h	**IC2LR**	MSB							LSB
	Reset Value	-	-	-	-	-	-	-	-
0027h	**OC2HR**	MSB							LSB
	Reset Value	1	0	0	0	0	0	0	0
0028h	**0C2LR**	MSB							LSB
	Reset Value	0	0	0	0	0	0	0	0

9.4 RF INTERFACE

9.4.1 Introduction

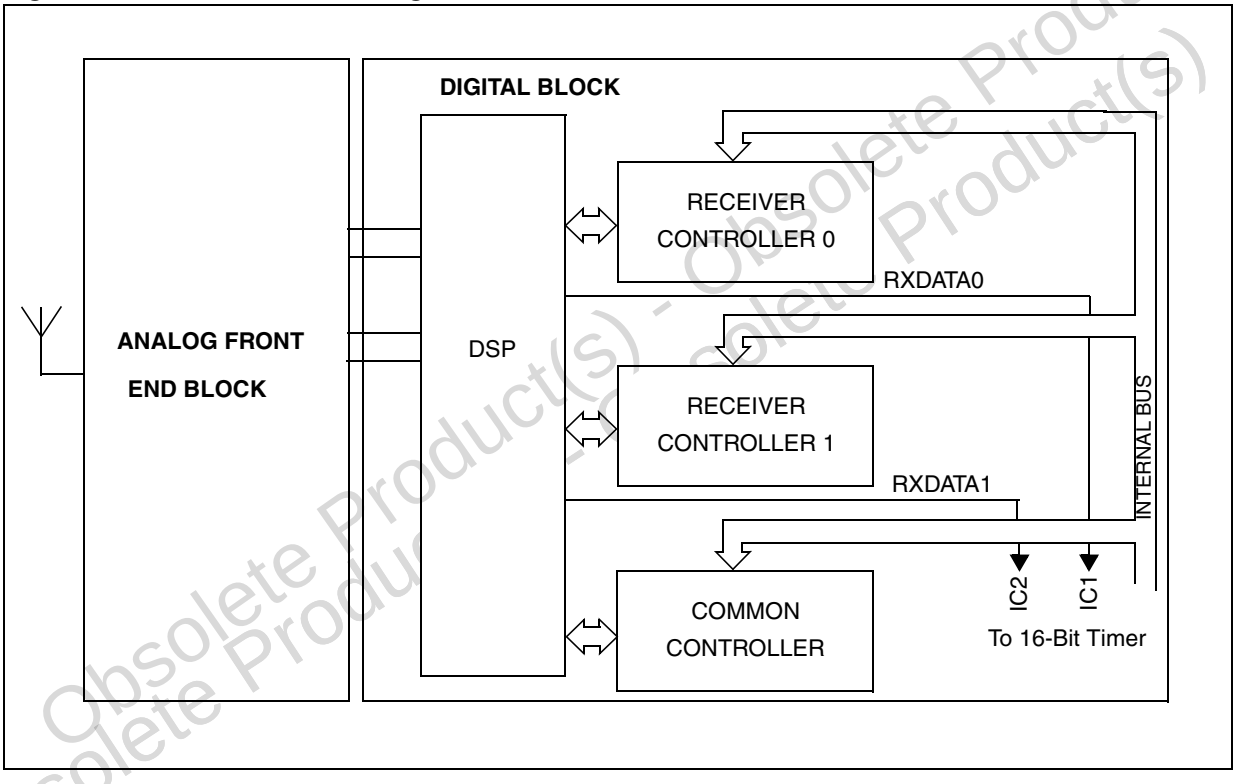
The RF Communication block is a fully integrated dual receiver. It operates in the 27 MHz band covering worldwide usage. It supports NRZ data and offers a cost-saving solution requiring a minimum of external components: crystal, PLL loop filter or biasing resistors.

It contains two receivers which can operate simultaneously(Refer to Figure 45).

9.4.2 Main Features

- Single 12 MHz crystal oscillator system
- 27 MHz Band PLL Synthesizer
  - Channel frequency: 26.905 to 27.305 MHz in 10KHz steps
- FSK demodulation
- Compatible with NRZ data (1.5K to 5Kbits/s data rate supported)
- Flexible data decoding using the 16-bit timer input capture function
- Receiver Modulation Index capability:1.9 to 3.2 at 2.5 Kbits/s; 1.2 to 2.1 at 5 Kbits/s
- 5 μV input sensitivity with BER=10^{-4}
- 36 dB of adjacent channel rejection (at 50 KHz)

Figure 45. RF General Block Diagram



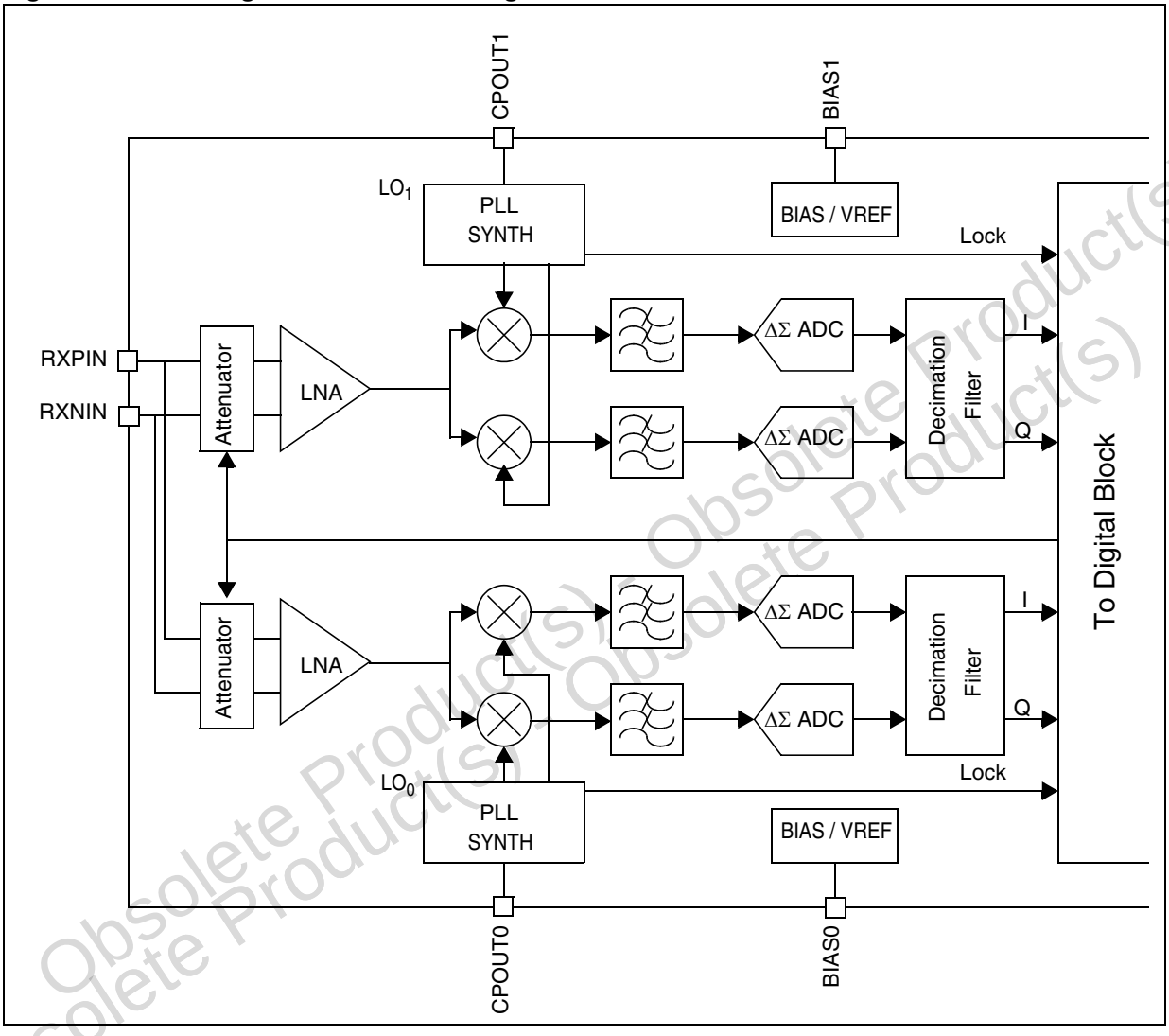
RF INTERFACE (Cont'd)

9.4.3 Functional Description

Figure 46 shows the block diagram of the RF analog front end section.

The demodulator principle is I/Q low IF (intermediate frequency) with a 15kHz offset.

Figure 46. RF Analog Front End Block Diagram



RF INTERFACE (Cont'd)

9.4.3.1 Receiver Overview

RF Analog Front End

The RF analog front end of the receiver is an LNA (Low Noise Amplifier), including an attenuator, which first amplifies the signal. The signal is then down-mixed into the low Intermediate Frequency range (15kHz). The low-pass filter keeps this signal in the bandwidth from DC up to 100kHz. The delta-sigma ADC and the decimator filter convert the analog signal into digital words.

The signal is digitally down-mixed to DC frequency, and the digital demodulator provides the data stream to the digital block (refer to Figure 47.Digital Receiver Block Diagram).

The PLL synthesizer must be programmed to select the channel to be demodulated (with a 15kHz frequency offset).

The Electrical Characteristics section provides the data rate and frequency deviation allowed for the receiver.

RF Digital block

The digital block is based on a DSP with embedded firmware which generates the data stream.

The DSP functions for the receiver are shown in Figure 47:

– Intermediate Frequency Local Oscillator (IFLO) Mixer

– Channel Filter & Decimation

– FM Discriminator

– Baseband Filter

– Slicer

The mixer function mixes the I and Q signals coming from the analog front end with the output of a baseband complex oscillator (IFLO) to mix from the IF down to DC. The IFLO has a programmable frequency allowing small frequency offsets.

Once the signal is mixed, the DSP works at baseband frequency range.

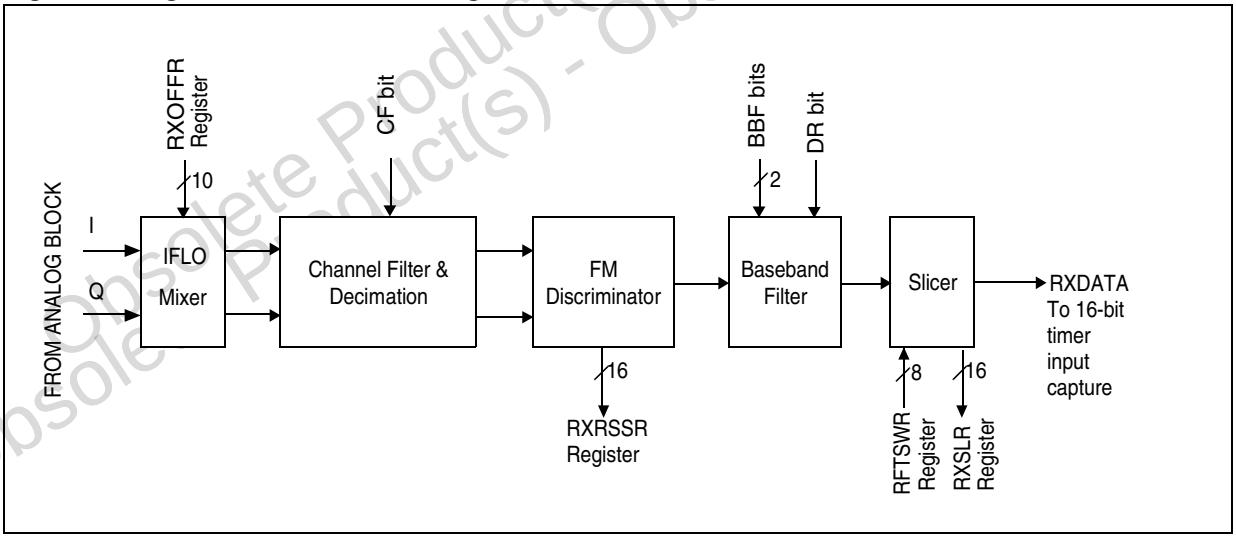
The channel filter isolates the wanted channel and the result is decimated.

The channel filter & decimation output is converted in the FM discriminator from cartesian to polar. The resulting magnitude is used as the signal strength indicator (RSS) and the angle derivative is used to calculate instantaneous frequency.

The slicer produces an oversampled version of binary data.

The RXDATA output is connected internally to the input capture of the 16-bit timer.

Figure 47. Digital Receiver Block Diagram



RF INTERFACE (Cont'd)

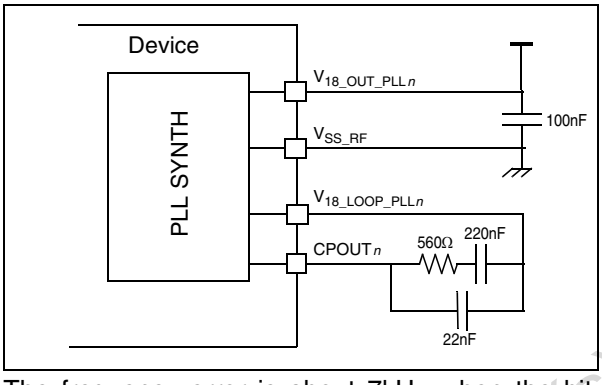
9.4.3.2 PLL Synthesizer

The Device includes an integrated PLL synthesizer which allows to demodulate any channel frequency from 26.905MHz up to 27.305MHz in steps of 10kHz.

The reference clock is provided by the 12MHz crystal oscillator, and then multiplied by the relevant factor defined using RX*n*CFR registers.

The PLL synthesizer requires only few external components for loop stabilization: two capacitors and one resistor, as shown in Figure 48 on page 87.

Figure 48. PLL Synthesizer Application

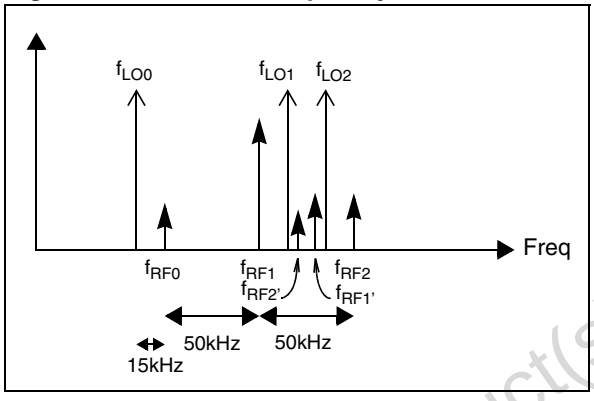


The frequency error is about 7kHz when the bit status LOCK*n* rises. This error decreases down to less than 1kHz (37ppm), after a further delay of about 100µs.

9.4.3.3 Receiver Frequency plan

Figure 49 shows the frequency plan of a typical receiver application involving the Device.

Figure 49. Receiver Frequency Plan



The receiver is designed to work on channels spaced 50kHz apart, with 15kHz of Intermediate Frequency.

Two-channel environment

The best way to configure a 2-channel environment is: f_{RF0} and f_{RF1} are the channels of a two channel system, f_{RF0} is lower than f_{RF1}. The f_{RF0} signal is demodulated with local oscillator 0 (f_{LO0}) and f_{RF1} with local oscillator 1 (f_{LO1}). f_{LO0} must be set at 15kHz lower than f_{RF0} and f_{LO1} at 15kHz higher than f_{RF1} by programming the RX*n*CF register.

For example, in order to select f_{RF0} at 27.055MHz and f_{RF1} at 27.105MHz, f_{LO0} may be set to 27.04MHz and f_{LO1} may be set to 27.12MHz with the RX*n*CF register. Then, the user must select the wanted channel, by programming the RX*n*OFF register (for example, $RXnOFFR_0$ may be set to +15kHz, $RXnOFFR_1$ may be set to -15kHz).

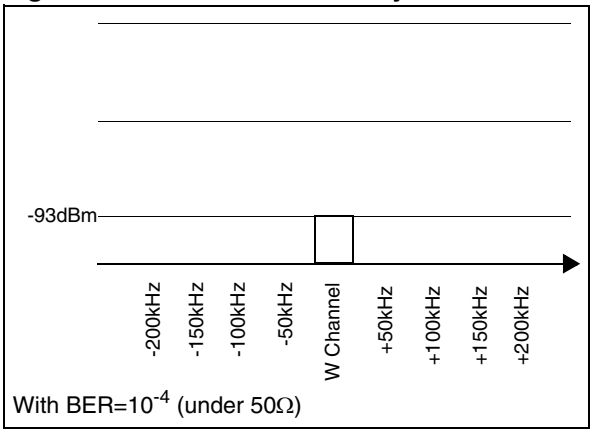
RF INTERFACE (Cont'd)

9.4.3.4 Receiver characteristics

Sensitivity

The "sensitivity" level is associated with a single signal to be received. In order to reach a targeted BER (Bit Error Rate) for the demodulator, the input signal needs to be at or above a minimum level. This minimum level is the "sensitivity" level. Figure 50 shows the RF receiver sensitivity level.

Figure 50. Reference Sensitivity Level



Adjacent channel rejection (ACR)

A signal in the adjacent channel (unwanted signal at +/-50kHz of frequency offset) may affect the BER if the level of this unwanted signal is above the adjacent channel rejection specification limit (see Figure 51).

The receiver is able to receive the wanted channel (W Channel) until the power level of the unwanted adjacent channel (UW Chan.) is less than 36dB above the wanted channel. This is assuming that the wanted signal is 3dB over the reference sensitivity level, as defined above.

The RF demodulator template of adjacent channel rejection is shown in Figure 51.

Notes:

1. One of the limiting factors is the phase noise of the receiver and the phase noise of the interfering signal.

2. It is assumed that the interfering signal (single tone generator) has phase noise characteristics better than 100dBc/Hz @ 50kHz.

RF INTERFACE (Cont'd)

Figure 51. Reference In-Band Interferer Level

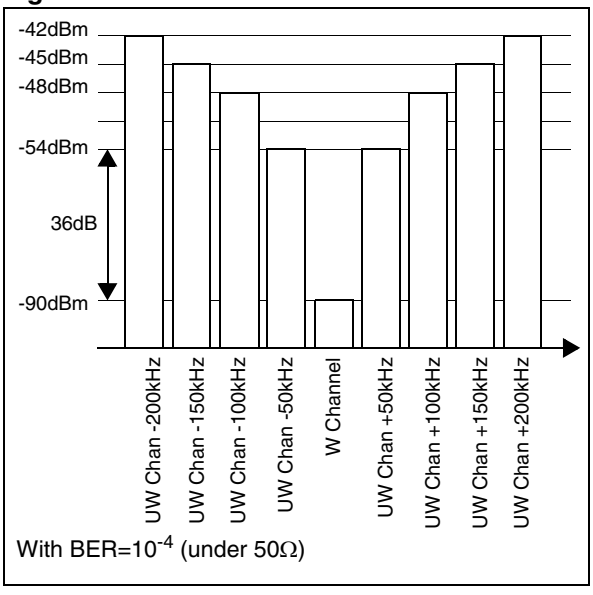


Image suppression (IMR)

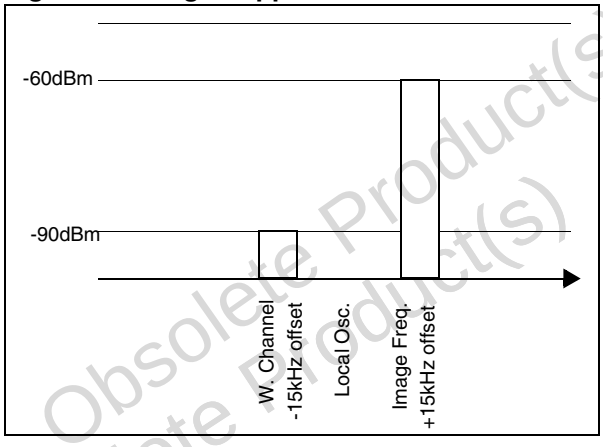
As a consequence, setting the frequency offset will automatically define where the image frequency will be: -(RX*n*OFFR). The receiver is sensitive to that frequency, with (at least) 30dB of attenuation.

The receiver is able to receive the wanted channel (W Channel) until the power level of the unwanted adjacent channel (in the Image frequency) is less than 30dB above the wanted channel. This is assuming that the wanted signal is 3dB above the reference sensitivity level, as defined in Figure 50.

The RF demodulator template of image suppression channel rejection is shown in Figure 52.

For example, if RX*n*OFFR is set to -15kHz, then the image frequency is set to +15kHz. An unwanted signal in the image frequency is attenuated of 30dB with respect to the wanted signal.

Figure 52. Image Suppression Level

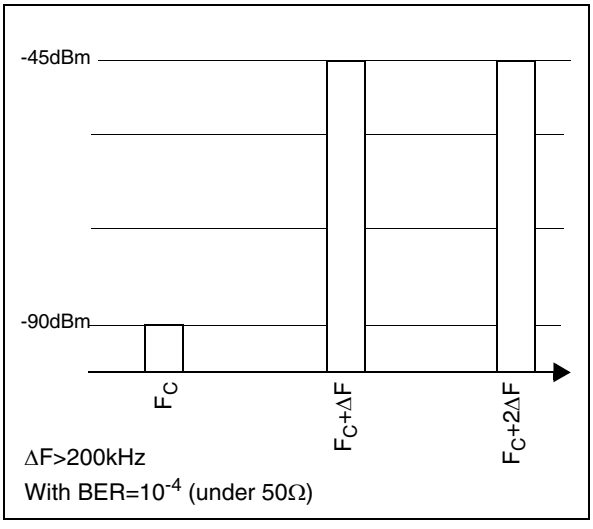


Out-of-Band Interferer Reference Level

A dual single tone signal at frequencies $F_C+\Delta F$ and $F_C+2\Delta F$ can affect the bit error rate on the wanted signal at frequency F_C (due to 3rd order inter-modulation distortion). The limit of these two interferer signals are specified as shown in Figure 53.

RF INTERFACE (Cont'd)

Figure 53. Out-of-Band Interferer Reference Level



This dual channel emission can be either on upper frequencies (as shown in Figure 53), or on lower frequencies (F_C; F_C-ΔF; F_C-2ΔF)

Input Attenuator

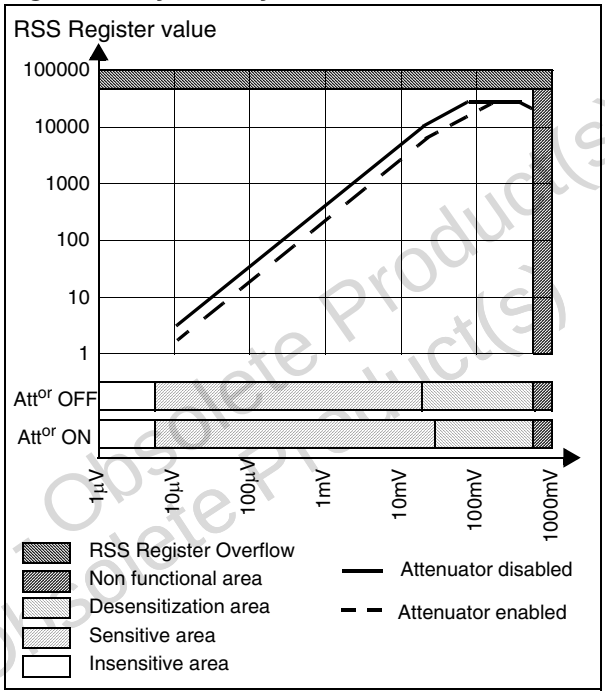
The analog RF chain includes an input attenuator. It can be set or reset using the RXATN bit.

When the received signal is high (close to the saturation of the chain), the attenuator can be used to reduce the gain of the whole analog chain (both receivers are then attenuated). The user can choose to enable or disable this attenuator depending on the value of the RX*n*RSS register, which provides the strength of the demodulated signal in the demodulation channel.

Receiver Compression / Saturation

Figure 54 shows the maximum input signal level allowed before saturation of the analog chain at 10mV (-27dBm) with attenuator disabled and 50mV (-13dBm) with attenuator enabled.

Figure 54. System Dynamic



Receiver Desensitization

This phenomenon is directly linked to compression (see above).

When the received signals become large enough at the input of the LNA, the receiver chain (LNA, mixers, low pass filter and ADC) saturates as shown in Figure 54. The total gain of this chain is then reduced.

If the chain receives an (unwanted) strong signal and a wanted signal with a lower amplitude, then the level of the wanted signal is decreased.

This is due to the loss of gain due to the strong signal which saturates the chain.

RF INTERFACE (Cont'd)

9.4.3.5 Device Power Supply Decoupling

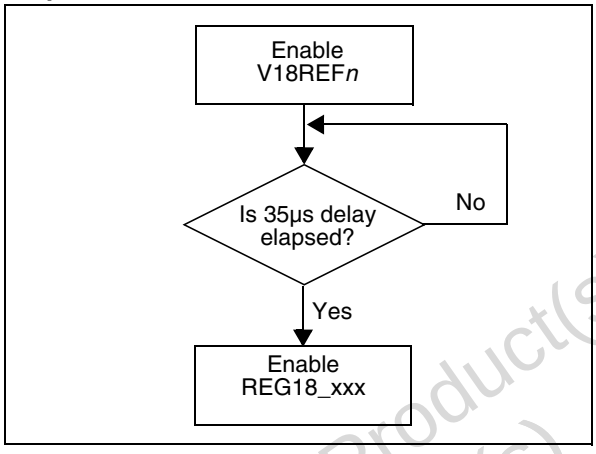
The RF Analog Front End is divided into 5 different blocks in order to guarantee optimum radio characteristics (power supply decoupling). Figure 56 shows how to decouple these power supplies.

The noise level on these power supplies must be checked properly, in order to guarantee the overall RF performance.

See also Electrical Characteristics section: Section 11.11.1.1 "Characterization Conditions" on page 162.

Important note: each of the RF regulators can be powered on/off individually through the RFREGCR*n* register. The sequence used to enable RF regulators is shown in Figure 55.

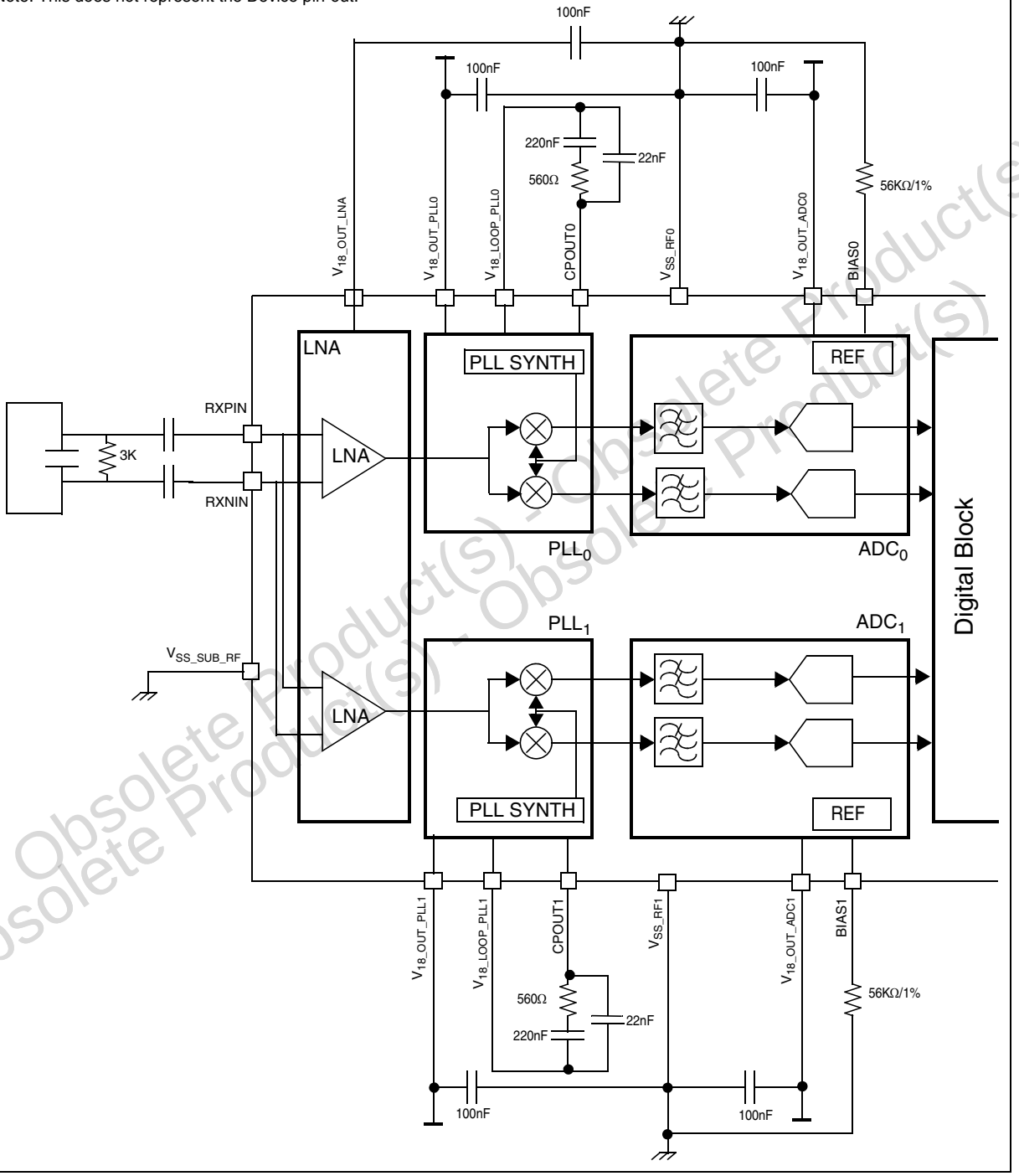
Figure 55. RF Voltage Regulator Enable Sequence



RF INTERFACE (Cont'd)

Figure 56. Device Power Supply Decoupling and External Components

V_{18_XXX} are power supplies generated through internal 1.8V regulators. They must not be shorted together.
V_{SS_XXX} must be star routed as cleanly as possible.
Note: This does not represent the Device pin-out.

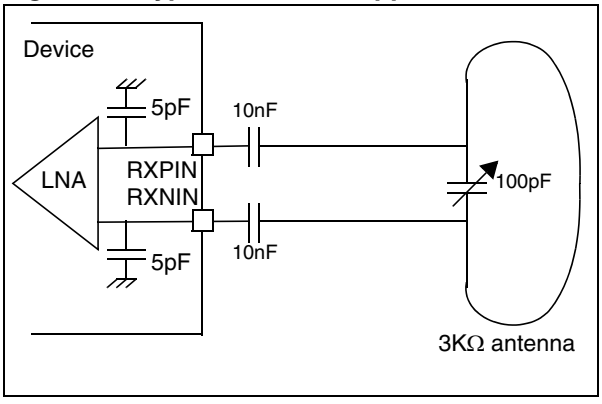


RF INTERFACE (Cont'd)

9.4.3.6 Application Examples

Figure 57 shows a typical application schematic, with a characteristic impedance of 3KΩ for the antenna and the input (off-chip resistor).

Figure 57. Typical Receiver Application



Note: the antenna frequency response should be stable within the operating band. The antenna should attenuate outband signals to improve overall system performance.

9.4.4 Low Power Modes

Mode	Description
WAIT	No effect on RF Communication peripheral.
HALT	RF Communication peripheral registers are frozen. In halt mode, the RF Communication peripheral is inactive. Before entering halt mode, the receivers must be disabled. Before resuming operations, the 12MHz oscillator must be selected.

9.4.5 Interrupts

The RF Communication peripheral does not manage any interrupt sources as data decoding is performed using the 16-bit timer.

RF INTERFACE (Cont'd)

9.4.6 Register Description

The register map of the RF peripheral is divided into pages. Each page allows access to a dedicated controller which is selected through the RFPAGER register. Each controller provides the user with control registers for setting RF parameters.

These parameters must be static during operations, but may be changed while the receiver is disabled. All RF parameters are taken into account by the DSP when the receiver is enabled.

9.4.6.1 RF Page Selector

RF PAGE SELECTION REGISTER (RFPAGER)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
0	RXT	0	0	0	0	RFP1	RFP0

Bit 7 = Reserved, must be kept cleared.

Bit 6 = **RXT** *RX Test*
This bit is set and cleared by software. It enables the output of RX*n*DATA on PB0 and PB1 pins.
0: No effect on PB0 and PB1 pins (used as standard I/O pins)
1: RX*n* DATA output available on PB0 and PB1 pins

Bit 5:2 = Reserved, must be kept cleared.

Bits 1:0 = **RFP[1:0]** *RF Page Selection*
These bits are set and cleared by software. They select the RF page as shown in the table below.

RFP1 bit	RFP0 bit	Selected RF Page	Peripheral
0	0	0	RFCSR, RX*n*CSR and RX*n*RSSR
0	1	1	RF Common Controller
1	0	2	RF Receiver Controller 0
1	1	3	RF Receiver Controller 1

Table 20. RF Page Selector Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
60h	RFPAGER	0	RXT	0	0	0	0	RFP1	RFP0
	Reset Value	0	0	0	0	0	0	0	0

RF INTERFACE (Cont'd)

9.4.6.2 RF Common Controller

The RF Common Controller provides the user with control registers for setting the RF parameters that are common to the two channel receivers.

CONTROL/STATUS REGISTER (RFCSR)
Read/Write (except bits 2 and 3)
Reset value: 0000 0000 (00h)

7							0
0	0	0	0	LOCK 1	LOCK 0	RX INV	RX ATN

Bits 7:4 = Reserved, must be kept cleared.

Bit 3 = **LOCK1** *In-lock Indicator for RX1 Synthesizer (Read only)*
This bit is set and cleared by hardware.
0: RX1 synthesizer not locked
1: RX1 synthesizer locked

Bit 2 = **LOCK0** *In-lock Indicator for RX0 Synthesizer (Read only)*
This bit is set and cleared by hardware.
0: RX0 synthesizer not locked
1: RX0 synthesizer locked

Bit 1 = **RXINV** *Inverter enable*
This bit is set and cleared by software. It is used to invert the receiver I and Q signals.
0: Inverter disabled
1: Inverter enabled

Bit 0 = **RXATN** *Attenuator enable*
This bit is set and cleared by software. It is used to control the 14 dB receiver attenuator.
0: Attenuator disabled
1: Attenuator enabled

SYNTHESIZER REGISTER (RFSYNR)
Read/Write
Reset value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0 = **D[7:0]** *Synthesizer Reference Divider*
This register configures the reference frequency to be used by both synthesizers.

Software must write 96h in this register to set the reference frequency to 80 kHz.

RF REGULATOR CONTROL REGISTER 0 (RFREGCR0)
Read/Write
Reset value: 0000 0000 (00h)

7							0
0	0	0	REG 04	REG 03	REG 02	REG 01	0

Bits 7:5 = Reserved, must be kept cleared.

Bits 4:1 = **REG0[4:1]** *Regulator control bits*
These bits are set and cleared by software. They can be used to individually power on and power off the various regulators. Refer to the Supply Management Chapter for a description of the regulators.
0: Regulator off
1: Regulator on

REG0 bits	Meaning
REG01	V18REF0
REG02	REG18_LNA
REG03	REG18_ADC0
REG04	REG18_PLL0

Bit 0 = Reserved, must be kept cleared.

RF COMMUNICATION (Cont'd)

RF REGULATOR CONTROL REGISTER 1 (RFREGCR1)
Read/Write
Reset value: 0000 0000 (00h)

7							0
0	0	0	REG 14	REG 13	0	REG 11	0

Bits 7:5 = Reserved, must be kept cleared.

Bits 4:3 = **REG1[4:3]** *Regulator control bits*
These bits are set and cleared by software. They can be used to individually power on and power off the regulators. Refer to the Supply Management Chapter for a description of the regulators.
0: Regulator off
1: Regulator on

REG1 bits	Meaning
REG11	V18REF1
REG13	REG18_ADC1
REG14	REG18_PLL1

Bit 2 = Reserved, must be kept cleared.

Bit 1 = **REG11** *Regulator control bit*
Refer to the above table.

Bit 0 = Reserved, must be kept cleared.

TRACK SLICER WEIGHTING REGISTER (RFTSWR)
Read/Write
Reset value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0= **D[7:0]** *Slicer Weighting*

This register defines the slicer time constant. Authorized values are listed in the following table:

D[7:0]	Slicer time constant (ms)
00h	Infinite
01h	699.051
02h	349.525
03h	233.017
04h	174.763
05h	139.810
06h	116.508
07h	99.864
08h	87.381
09h	77.672
0Ah	69.905
0Bh	63.550
0Ch	58.254
0Dh	53.773
0Eh	49.932
0Fh	46.603
10h	43.691
11h	41.121
12h	38.836
13h	36.792
14h	34.953
15h	33.288
16h	31.775
17h	30.394
18h	29.127
19h	27.962
1Ah	26.887
1Bh	25.891
1Ch	24.966
1Dh	24.105
1Eh	23.302
1Fh	22.550
20h	21.845
21h	21.183
22h	20.560

D[7:0]	Slicer time constant (ms)
23h	19.973
24h	19.418
25h	18.893
26h	18.396
27h	17.924
28h	17.476
29h	17.050
2Ah	16.644
2Bh	16.257
2Ch	15.888
2Dh	15.535
2Eh	15.197
2Fh	14.873
30h	14.564
31h	14.266
32h	13.981
33h	13.707
34h	13.443
35h	13.190
36h	12.945
37h	12.710
38h	12.483
39h	12.264
3Ah	12.053
3Bh	11.848
3Ch	11.651
3Dh	11.460
3Eh	11.275
3Fh	11.096
50h	10.923
51h	10.280
52h	9.709
53h	9.198
54h	8.738
55h	8.322
56h	7.944
57h	7.598
58h	7.282
59h	6.991
5Ah	6.722
5Bh	6.473
5Ch	6.241
5Dh	6.026
5Eh	5.825
5Fh	5.638
60h	5.461
61h	5.296
62h	5.140

D[7:0]	Slicer time constant (ms)
63h	4.993
64h	4.855
65h	4.723
66h	4.599
67h	4.481
68h	4.369
69h	4.262
6Ah	4.161
6Bh	4.064
6Ch	3.972
6Dh	3.884
6Eh	3.799
6Fh	3.718
70h	3.641
71h	3.567
72h	3.495
73h	3.427
74h	3.361
75h	3.297
76h	3.236
77h	3.178
78h	3.121
79h	3.066
7Ah	3.013
7Bh	2.962
7Ch	2.913
7Dh	2.865
7Eh	2.819
7Fh	2.774
90h	2.731
91h	2.570
92h	2.427
93h	2.300
94h	2.185
95h	2.080
96h	1.986
97h	1.900
98h	1.820
99h	1.748
9Ah	1.680
9Bh	1.618
9Ch	1.560
9Dh	1.507
9Eh	1.456
9Fh	1.409
A0h	1.365
A1h	1.324
A2h	1.285

D[7:0]	Slicer time constant (ms)
A3h	1.248
A4h	1.214
A5h	1.181
A6h	1.150
A7h	1.120
A8h	1.092
A9h	1.066
AAh	1.040
ABh	1.016
ACh	0.993
ADh	0.971
AEh	0.950
AFh	0.929
B0h	0.910
B1h	0.892
B2h	0.874
B3h	0.857
B4h	0.840
B5h	0.824
B6h	0.809
B7h	0.794
B8h	0.780
B9h	0.767
BAh	0.753
BBh	0.740
BCh	0.728
BDh	0.716
BEh	0.705
BFh	0.694
D0h	0.683
D1h	0.643
D2h	0.607
D3h	0.575
D4h	0.546
D5h	0.520
D6h	0.496
D7h	0.475
D8h	0.455
D9h	0.437
DAh	0.420
DBh	0.404
DCh	0.390
DDh	0.377
DEh	0.364
DFh	0.352
E0h	0.341
E1h	0.331
E2h	0.321

D[7:0]	Slicer time constant (ms)
E3h	0.312
E4h	0.303
E5h	0.295
E6h	0.287
E7h	0.280
E8h	0.273
E9h	0.266
EAh	0.260
EBh	0.254
ECh	0.248
EDh	0.243
EEh	0.237
EFh	0.232
F0h	0.228
F1h	0.223
F2h	0.218
F3h	0.214
F4h	0.210
F5h	0.206
F6h	0.202
F7h	0.199
F8h	0.195
F9h	0.192
FAh	0.188
FBh	0.185
FCh	0.182
FDh	0.179
FEh	0.176
FFh	0.173

RF INTERFACE (Cont'd)

Table 21. RF Common Controller Register Map and Reset Values (Page 0)

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
61h	RFCSR	0	0	0	0	LOCK1	LOCK0	RXINV	RXATN
	Reset Value	0	0	0	0	0	0	0	0

Table 22. RF Common Controller Register Map and Reset Values (Page 1)

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
61h	RFREGCR0	0	0	0	REG04	REG03	REG02	REG01	0
	Reset Value	0	0	0	0	0	0	0	0
62h	RFREGCR1	0	0	0	REG14	REG13	0	REG11	0
	Reset Value	0	0	0	0	0	0	0	0
63h	RFSYNR	D7	D6	D5	D4	D3	D2	D1	D0
	Reset Value	0	0	0	0	0	0	0	0
64h to 66h	Reserved area (3 Bytes)								
67h	RFTSWR	D7	D6	D5	D4	D3	D2	D1	D0
	Reset Value	0	0	0	0	0	0	0	0
68h to 6Ah	Reserved area (3 Bytes)								

RF INTERFACE (Cont'd)

9.4.6.3 RF Receiver Controller

The RF Receiver Controller provides the user with control registers for setting parameters of the receiver block.

CONTROL STATUS REGISTER (RX*n*CSR)
Read /Write
Reset value: 0000 0000 (00h)

7							0
RXD OEN	0	0	RX SEL	RX EN	CF	BB F1	BB F0

Bits 7 = **RXDOEN** *Receiver Data Output Enable*

This bit is used to enable or disable the receiver data stream output to the timer.
0: Disabled
1: Enabled

Bit 6:5 = Reserved, must be kept cleared

Bit 4 = **RXSEL** *Capture Mode selection*

This bit is used to select the capture mode .
0: Connect I/O Port PAx to 16-bit timer ICx function
1: Connect RXDATA internally to 16-bit timer ICx function

Bit 3 = **RXEN** *Receiver enable*
This bit is used to enable and disable the receiver.
0: Disabled
1: Enabled

Bit 2 = **CF** *Channel filter bandwidth*
This bit is used to select the channel filter bandwidth.
0: Standard bandwidth (16 kHz)
1: High bandwidth (22 kHz)

Note: The standard bandwidth channel filter should be used for data rates lower than 4 kbits/s.

Bits 1:0 = **BBF[1:0]** *Baseband filter bandwidth*

These bits are used to select one of the eight possible low pass filters depending on the protocol and data rate used.

BBF1	BBF0	Baseband Filter Bandwidth (kHz)	
		RX*n*DRR ≤ 4.8 Kbits/s	RX*n*DRR > 4.8 Kbits/s
0	0	1.2 kHz	3.2 kHz
0	1	1.7 kHz	4.8 KHz
1	0	2.4 kHz	6.4 KHz
1	1	3.2 kHz	9.6 KHz

CARRIER FREQUENCY REGISTER LOW (RX*n*CFLR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0

Bits 7:0 = **CF[7:0]** Carrier *Frequency LSB*
This register, used with the RX*n*CFHR register, programs the receiver synthesizer. It contains the LSB.

CARRIER FREQUENCY REGISTER HIGH (RX*n*CFHR)
Read/Write
Reset value: 0000 0000 (00h)

7							0
0	0	0	0	CF11	CF10	CF9	CF8

Bits 7:4= Reserved, must be kept cleared

Bits 3:0= **CF[11:8]** *Carrier Frequency MSB*
This register, used with the RX*n*CFLR register, programs the receiver synthesizer with a resolution of 10 kHz. It contains the MSB.

CF[11:0]	Carrier Frequency (MHz)
0A82h	26.90 MHz
0AA6h	27.26 MHz
..	..

RF INTERFACE (Cont'd)

FREQUENCY OFFSET LOW REGISTER (RX*n*OFFLR)
Read/Write
Reset value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0= **RXOFFSET[7:0]** *Frequency Offset*
This register, used with the RX*n*OFFHR register, programs the receiver frequency offset with a resolution of 100 Hz. It contains the LSB.

FREQUENCY OFFSET HIGH REGISTER (RX*n*OFFHR)
Read/Write
Reset value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	D9	D8

Bits 7:2 = Reserved, must be kept cleared.

Bits 1:0= **RXOFFSET[9:8]** *Frequency Offset*
This register, used with the RX*n*OFFLR register, programs the receiver frequency offset with a resolution of 100 Hz. It contains the MSB.

RX*n*OFFxR registers act as a 16-bit signed register and use 2's complement encoding:

RXOFFSET [9:0]	Frequency Offset (kHz)
00 96h	+15 kHz
03 6Ah	-15 kHz

DATA RATE REGISTER (RX*n*DRR)
Read/Write
Reset value: 0000 0000 (00h)

7							0
0	DR	0	0	0	0	0	0

Bit 7 = Reserved, must be kept cleared.

Bit 6 = **DR** *Data rate*
This bit is used to select the receiver NRZ data rate.
0: NRZ data rate ≤ 4.8 kbits/s
1: NRZ data rate > 4.8 kbits/s

Bits 5:0 = Reserved, must be kept cleared.

Note: NRZ data rate value must be within the 1.5 K to 5 Kbits/s range.

RF INTERFACE (Cont'd)

RECEIVER STRENGTH LOW REGISTER (RX*n*RSSLR)
Read only
Reset value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0= **RXRSS[7:0]** *Receiver Signal Strength*
This 8-bit register is used with the RX*n*RSSHR register. It contains the LSB of the receiver signal magnitude.

Table 23. RSS register value versus input level with attenuator disabled

	Typ. (μV_{RMS})	RXRSS[15:0]
Sensitive area	6	0005h
	7	0006h
	8	0007h
		...
	14	000Ch
	15	000Dh
	17	000Eh
		...
	107	005Ah
		...
	1000	0345h
		...
	10055	20E4h
	13502	2C57h
	15150	31A6h
Desensitization area		...
	21399	3A03h
	26940	3B9Bh
	30227	3C10h
		...
	107251	3D56h
	213993	3D61h
	603115	3D57h

RECEIVER STRENGTH HIGH REGISTER (RX*n*RSSHR)
Read only
Reset value: 0000 0000 (00h)

7							0
D15	D14	D13	D12	D11	D10	D9	D8

Bits 7:0= **RXRSS[15:8]** *Receiver Signal Strength*
This 8-bit register is used with the RX*n*RSSHR register. It contains the MSB of the receiver signal magnitude. It provides a value indicating the input level in the selected channel. This register is useful for determining if the system is saturated during demodulation.

Table 24. RSS register value versus input level with attenuator enabled

	Typ. (μV_{RMS})	RXRSS[15:0]
Sensitive area	4	0001h
	8	0002h
	12	0003h
		...
	107	001Bh
		...
	1001	00FBh
		...
	9559	0967h
	10725	0A89h
	12034	0BD3h
		...
Desensitization area	38054	2562h
	47907	2EF5h
	53753	33CEh
		...
	60311	3786h
	67671	3979h
	75928	3AB1h
	213993	3D3Eh
	302273	3D48h
	537527	3D4Fh

Caution: Due to the attenuation of the channel filter (50dB out of channel bandwidth), the RSS value gives the signal level of an unwanted signal attenuated by 50dB.

RF INTERFACE (Cont'd)

RECEIVER SLICER LEVEL LOW REGISTER (RX*n*SLLR)
Read only
Reset value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bits 7:0= **RXSL[7:0]** *Slicer Level LSB*
This 8-bit register is used with the RX*n*SLHR register. It contains the LSB of the slicer level.

RECEIVER SLICER LEVEL HIGH REGISTER (RX*n*SLHR)
Read only
Reset value: 0000 0000 (00h)

7							0
D15	D14	D13	D12	D11	D10	D9	D8

Bits 7:0= **RXSL[15:8]** *Slicer Level MSB*
This 8-bit register is used with the RX*n*SLLR register. It contains the MSB of the slicer level using a resolution of 0.715Hz.

RX*n*SLxR registers act as a 16-bit signed register and use 2's complement encoding.

This register is useful to indicate the frequency error between the receiver and the transmitter, assuming that the transmitted signal is DC balanced. The value read can be used to adjust RX*n*OFFR register to compensate frequency errors.

RF INTERFACE (Cont'd)

Table 25. RF Receiver Controller Register Map and Reset Values (Page 0)

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
62h	**RX0CSR**	RXDOEN	0	0	RXSEL	RXEN	CF	BBF1	BBF0
	Reset Value	0	0	0	0	0	0	0	0
63h	**RX0RSSHR**	D15	D14	D13	D12	D11	D10	D9	D8
	Reset Value	0	0	0	0	0	0	0	0
64h	**RX0RSSLR**	D7	D6	D5	D4	D3	D2	D1	D0
	Reset Value	0	0	0	0	0	0	0	0
65h	**RX1CSR**	RXDOEN	0	0	RXSEL	RXEN	CF	BBF1	BBF0
	Reset Value	0	0	0	0	0	0	0	0
66h	**RX1RSSHR**	D15	D14	D13	D12	D11	D10	D9	D8
	Reset Value	0	0	0	0	0	0	0	0
67h	**RX1RSSLR**	D7	D6	D5	D4	D3	D2	D1	D0
	Reset Value	0	0	0	0	0	0	0	0

Table 26. RF Receiver Controller Register Map and Reset Values (RX0=Page 2, RX1=Page 3)

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
61h	**RX*n*CFHR**	0	0	0	0	CF11	CF10	CF9	CF8
	Reset Value	0	0	0	0	0	0	0	0
62h	**RX*n*CFLR**	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
	Reset Value	0	0	0	0	0	0	0	0
63h	**RX*n*OFFHR**	0	0	0	0	0	0	D9	D8
	Reset Value	0	0	0	0	0	0	0	0
64h	**RX*n*OFFLR**	D7	D6	D5	D4	D3	D2	D1	D0
	Reset Value	0	0	0	0	0	0	0	0
65h to 66h	Reserved area (2 Bytes)								
67h	**RX*n*DRR**	0	DR	0	0	0	0	0	0
	Reset Value	0	0	0	0	0	0	0	0
68h	**RX*n*SLHR**	D15	D14	D13	D12	D11	D10	D9	D8
	Reset Value	0	0	0	0	0	0	0	0
69h	**RX*n*SLLR**	D7	D6	D5	D4	D3	D2	D1	D0
	Reset Value	0	0	0	0	0	0	0	0
6Ah	Reserved area (1 Byte)								

9.5 USB INTERFACE (USB)

9.5.1 Introduction

The USB Interface implemented allows to connect the Device to a USB host in low-speed mode. It is a highly integrated block which includes a USB transceiver, the Serial Interface Engine (SIE) and the USB data buffer interface. No external components are needed apart from the external pull-up on USBDM for low-speed recognition by the USB host.

9.5.2 Main Features

- USB Specification Version 2.0 Compliant
- Supports Low-Speed USB Protocol
- Three Endpoints (including default endpoint)
- CRC generation/checking, NRZI encoding/decoding and bit-stuffing
- USB Suspend/Resume operations
- On-Chip USB Transceiver

9.5.3 Functional Description

The block diagram in Figure 58, 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 Device is ready to transmit/receive, and how many bytes need to be transmitted.

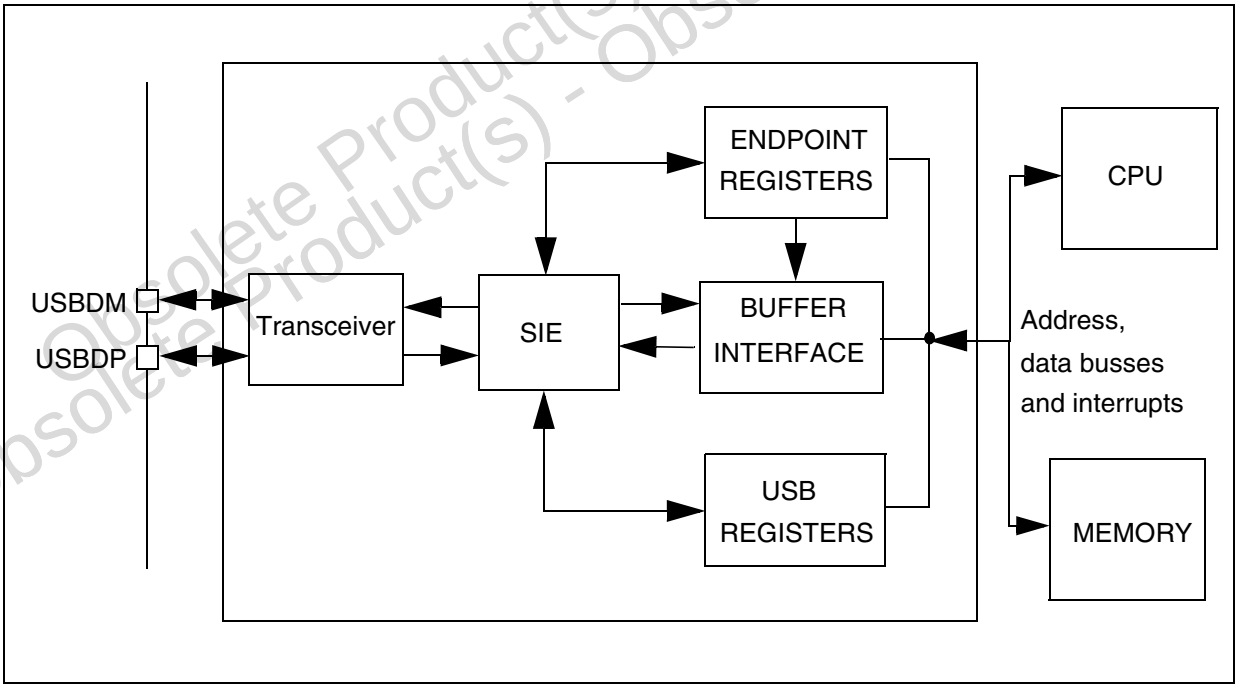
Data Transfer to/from USB Data Buffer Memory

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

Interrupts

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

Figure 58. USB Block Diagram



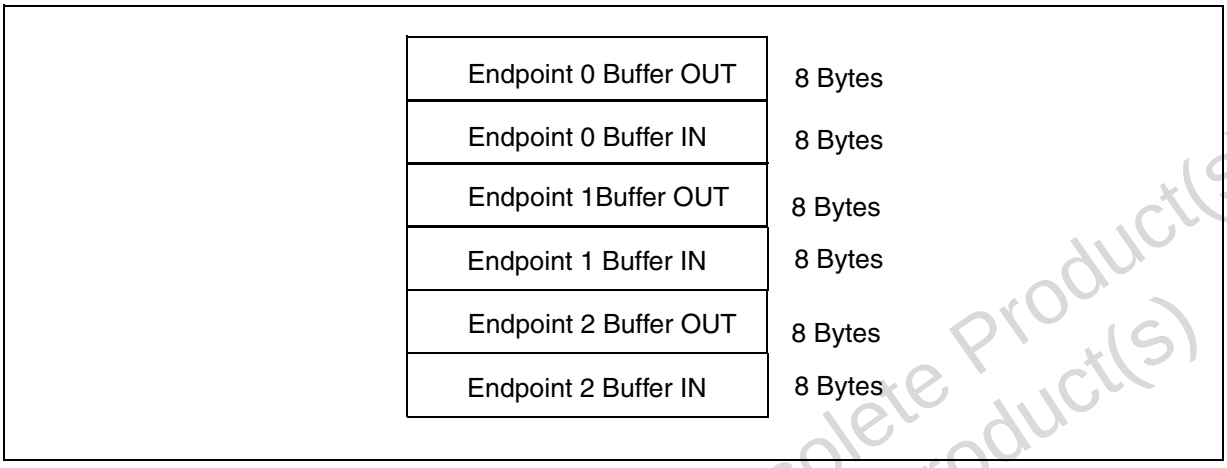
USB INTERFACE (Cont'd)

USB Endpoint RAM Buffers

There are three Endpoints including one bidirectional control Endpoint (Endpoint 0), two IN or OUT Endpoints (Endpoint 1 and 2).

All endpoints are 8 bytes in size.

Figure 59. Endpoint Buffer Size



USB INTERFACE (Cont'd)

9.5.4 Low Power modes

Mode	Description
WAIT	No effect on USB. USB interrupt events cause the Device to exit from WAIT mode.
HALT	USB registers are frozen. In halt mode, the USB is inactive. USB operations resume when the Device is woken up by an interrupt with "exit from halt capability" or by an event on the USB line in case of suspend. This event will generate an ESUSP interrupt which will wake-up from halt mode.

9.5.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit From Wait	Exit From Halt
Correct TRansfer	CTR	CTRM	Yes	No
Setup OVeRrun	SOVR	SOVRM	Yes	No
ERROR	ERR	ERRM	Yes	No
Suspend Mode Request	SUSP	SUSPM	Yes	No
End of SUSPend mode.	ESUSP	ESUSPM	Yes	Yes
USB RESET	RESET	RESETM	Yes	No
Start Of Frame	SOF	SOFM	Yes	No

Note: The USB end of suspend interrupt event is connected to a single interrupt vector (USB ESUSP) with the exit from halt capability (wake-up). All the other interrupt events are connected to another interrupt vector: USB interrupt (USB). They generate an interrupt if the corresponding enable control bit is set and the interrupt mask bits (I0, I1) in CC register are reset (RIM instruction).

USB INTERFACE (Cont'd)

9.5.6 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 USBEP0R register and CTR_RX and CTR_TX in the USBEPnRXR and USBEPnTXR 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[2:0] 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 overrun, bit stuffing or framing errors.

Bit 6 = Reserved, forced by hardware to 0.

Bit 5 = **SOVR** Setup Overrun.
This bit is set by hardware when a correct Setup transfer operation is performed while the software is servicing an interrupt which occurred on the same Endpoint (CTR0 bit in the USBEP0R register is still set when SETUP correct transfer occurs).
0: No SETUP overrun detected
1: SETUP overrun detected

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

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

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

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

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

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

This interrupt is serviced by a specific vector, in order to wake up the Device 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 USBDADDR, USBEP0R, USBEP1RXR, USBEP1TXR, USBEP2RXR and USBEP2TXR registers are reset by a USB reset.

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

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

USB INTERFACE (Cont'd)

INTERRUPT MASK REGISTER (USBIMR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CTRM	0	SOVRM	ERRM	SUSPM	ESUSPM	RESETM	SOFM

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

CONTROL REGISTER (USBCTLR)

Read/Write

Reset value: 0001 0110 (16h)

7							0
FSTAT1	FSTAT0	0	1	RESUME	PDWN	FSUSP	FRES

Bit 7:6 = **FSTAT[1:0]** *Forced state.*
This bits gives the state forced on USB ports when RESUME bit is set. Of course, the default value is K to force a RESUME event.

Note: This is a test feature because the only legal state which can be forced on USB port is the K state when the Device has the wake-up capability.

Table 27. Forced state encoding

FSTAT1	FSTAT0	Meaning
0	0	**K:** K state is forced on USB ports.
0	1	**J**: J state is forced on USB ports.
1	0	**SE0:** SE0 state forced on USB ports.
1	1	**SE1**: SE1 state forced on USB ports (not a valid USB state).

Bits [5:4] = Reserved.

Bit 3 = **RESUME** *Resume*.
This bit is set by software to wake-up the Host when the Device 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 transceiver.
0: Transceiver on
1: Transceiver off

Note: Do not forget to turn off the pull-up on DM before to set the PDWN bit.

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

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

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

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

USB INTERFACE (Cont'd)

DEVICE ADDRESS REGISTER (USBDADDR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0

Bit 7 = Reserved, forced by hardware to 0.

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

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

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

USB STATUS REGISTER 0 (USBSR0)

Read only

Reset Value: 0000 0000 (00h)

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

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

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

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

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

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

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

USB INTERFACE (Cont'd)

USB STATUS REGISTER 1 (USBSR1)

Read only

Reset Value: 0000 0000 (00h)

7							0
0	0	0	RSM	USB_RST	ERR2	ERR1	ERR0

Bits 7:5 = Reserved, forced by hardware to 0.

Bit 4= **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 3= **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 2:0 = **ERR[2:0]** *Error type*.

These bits identify the type of error which occurred.

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

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

USB INTERFACE (Cont'd)

ENDPOINT 0 REGISTER (USBEP0R)

Read/Write

Reset value: 0000 0000(00h)

7							0
CTR0	DTOG_TX	STAT_TX1	STAT_TX0	0	DTOG_RX	STAT_RX1	STAT_RX0

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

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

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

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

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

Table 28. Transmission Status Encoding

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

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

Bit 3 = Reserved, forced by hardware to 0.

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

USB INTERFACE (Cont'd)

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

Table 29. Reception Status Encoding

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

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

Note 1:

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

Note 2:

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

ENDPOINT TRANSMISSION REGISTER (USBEP1TXR, USBEP2TXR)

Read/Write

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	CTR_TX	DTOG_TX	STAT_TX1	STAT_TX0

This register is used for controlling Endpoint 1 or 2, transmission. DTOG_TX and STAT_TX bits are also reset by a USB reset, either received from the USB or forced through the FRES bit in the USBCTLR register.

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

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

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

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

Table 30. Transmission Status Encoding

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

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

USB INTERFACE (Cont'd)

ENDPOINT RECEPTION REGISTER (USBEP1RXR, USBEP2RXR)

Read/Write

Reset value: 0000 0000 (00h)

7							0
0	0	0	0	CTR_RX	DTOG_RX	STAT_RX1	STAT_RX0

This register is used for controlling Endpoint 1 and 2 reception. DTOG_RX and STAT_RX bits are also reset by a USB reset, either received from the USB or forced through the FRES bit in the USBCTLR register.

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

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

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

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

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

Table 31. Reception Status Encoding

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

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

RECEPTION COUNTER REGISTER (USBCNTn-RXR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	CNT3	CNT2	CNT1	CNT0

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

TRANSMISSION COUNTER REGISTER (USBCNTnTXR)

Read/Write

Reset Value 0000 0000 (00h)

7							0
0	0	0	0	CNT3	CNT2	CNT1	CNT0

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

Table 32. USB Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
003Dh	USBISTR Reset Value	CTR 0	0 0	SOVR 0	ERR 0	SUSP 0	ESUSP 0	RESET 0	SOF 0
003Eh	USBIMR Reset Value	CTRM 0	0 0	SOVRM 0	ERRM 0	SUSPM 0	ESUSPM 0	RESETM 0	SOFM 0
003Fh	USBCTLR Reset Value	FSTAT1 0	FSTAT0 0	0	1	RESUME 0	PDWN 1	FSUSP 1	FRES 0
0040h	USBDADDR Reset Value	0	ADD6 0	ADD5 0	ADD4 0	ADD3 0	ADD2 0	ADD1 0	ADD0 0
0041h	USBSR0 Reset Value	PID1 0	PID0 0	IN /OUT 0	0	0	EP2 0	EP1 0	EP0 0
0042h	USBSR1 Reset Value	0	0	0	RSM 0	USB_RST 0	ERR2 0	ERR1 0	ERR0 0
0043h	USBEP0R Reset Value	CTR0 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0	0 0	DTOG_RX 0	STAT_RX1 0	STAT_RX0 0
0044h	USBCNT0RXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
0045h	USBCNT0TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
0046h	USBEP1RXR Reset Value	0	0	0	0	CTR_RX 0	DTOG_RX 0	STAT_RX1 0	STATRX0 0
0047h	USBCNT1RXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
0048h	USBEP1TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
0049h	USBCNT1TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
004Ah	USBEP2RXR Reset Value	0	0	0	0	CTR_RX 0	DTOG_RX 0	STAT_RX1 0	STAT_RX0 0
004Bh	USBCNT2RXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0
004Ch	USBP2TXR Reset Value	0	0	0	0	CTR_TX 0	DTOG_TX 0	STAT_TX1 0	STAT_TX0 0
004Dh	USBCNT2TXR Reset Value	0	0	0	0	CNT3 0	CNT2 0	CNT1 0	CNT0 0

9.6 PS/2 INTERFACE

9.6.1 Introduction

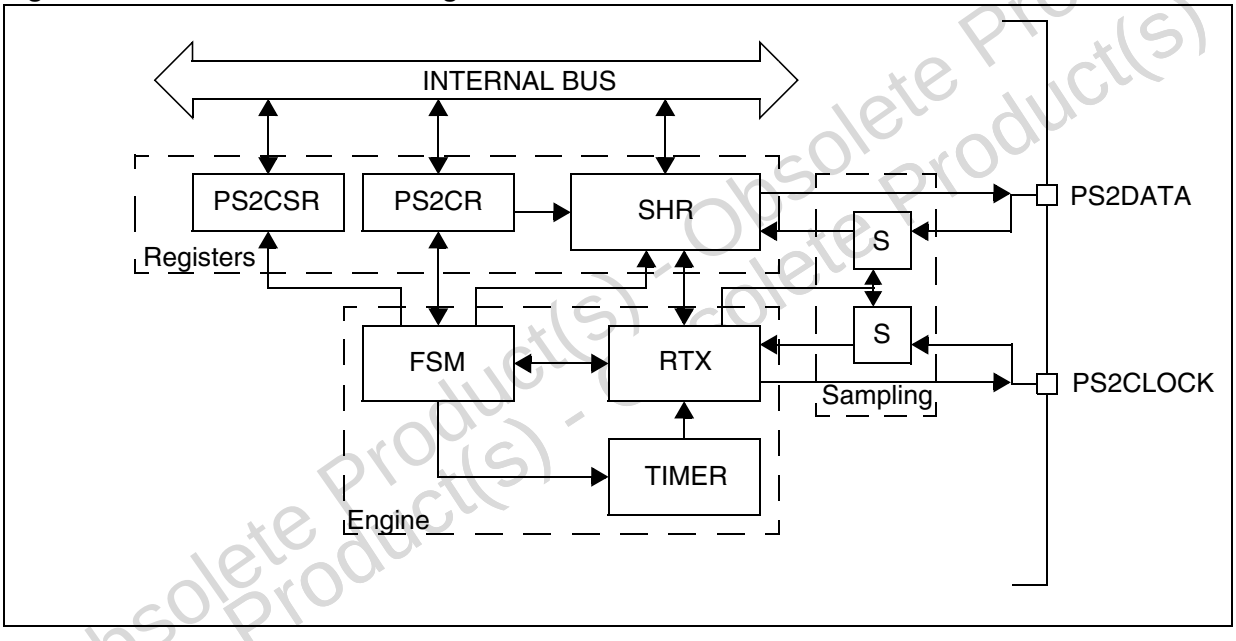
The PS/2 Interface allows communication between the Device and a PC using the PS/2 protocol.

The protocol used is based on the IBM specification.

9.6.2 Main Features

- Based on the IBM PS/2 specification
- Inter-Frame timing management
- Interrupt capability (4 interrupt sources)
  - Communication complete
  - Communication aborted
  - Host request to send
  - Reception error
- 3 Registers
  - Control register
  - Status register
  - Data register (for transmission and reception)

Figure 60. PS/2 Interface Block Diagram



PS/2 INTERFACE (Cont'd)

9.6.3 Protocol Description and Timings

The PS/2 Interface is designed to handle the basic PS/2 protocol.

Protocol

The communication is performed using two lines:
- PS2CLOCK
- PS2DATA

The clock is always generated by the PS/2 Interface (Device). The PC (Host) can only force it to zero in order to cancel the communication and/or inhibit the bus.

The data line is bidirectional, depending on the communication direction. A data frame is always composed of a start bit, 8 bits of data, a parity bit (odd parity) and a stop bit.

In host to Device communication, the host controls the data line. The Device can force the data line to zero to send an acknowledge after reception of a stop bit.

In Device to host communication, the PS/2 Interface controls the data line. If the host forces it to zero, it generates a Host Request To Send (HRTS).

Timings

The clock period is fixed at 80µs. Before its activation, the PS/2 Interface must be configured through the PS/2 Config/Status register (PS2CSR) to select the right ratio between the CPU frequency and the PS/2 Interface.

Host to Device

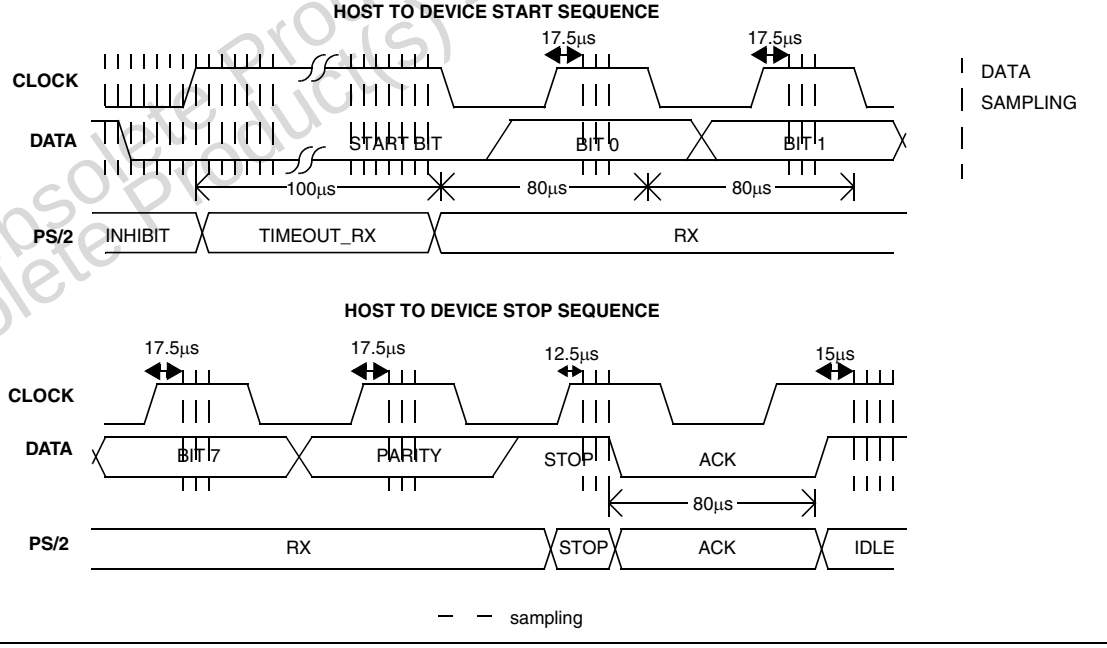
At the beginning of a Host to Device communication, the PS/2 Interface generates a fixed 100µs timeout duration between the release of the clock by the host and the generation of the first negative edge on the clock by the PS/2 Interface.

During the communication, the PS/2 Interface samples both clock and data lines three times at 2.5µs intervals. The acquired value is determined by these three samples.

A minimum delay is generated between an event (clock or data transition) and the first sampling. This is done in order to avoid corrupted sampling due to the line set-up time. This delay lasts from 12.5 to 17.5µs.

All the timings related to Host to Device communication are detailed in the Figure 61.

Figure 61. PS/2 Interface Timing: Host to Device communication



PS/2 INTERFACE (Cont'd)

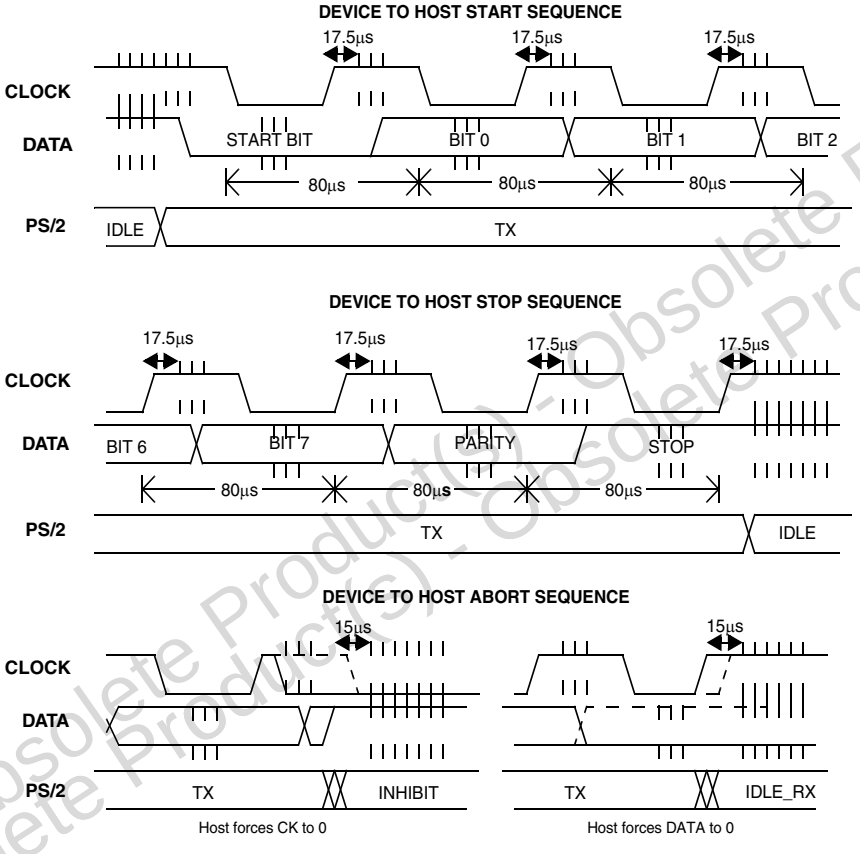
Device to Host

During a Device to Host communication, the PS/2 Interface samples both clock and data lines three times at 2.5µs intervals. This is shown in Figure 62.
A minimum delay is generated between an event (clock or data transition) and the first sampling. This is done in order to avoid corrupted sampling due to the line set-up time. This delay lasts from 15µs to 17.5µs.

Parity Calculation

The parity is calculated with an odd algorithm, meaning that the sum of the 8 data bits and the parity bit is odd.

Figure 62. PS/2 Interface Timing: Device to Host communication



PS/2 INTERFACE (Cont'd)

9.6.4 Functional Description

The basic PS/2 protocol is handled by the Device PS/2 Interface. Figure 63 illustrates the logic that is described below, following the Device from state to state.

Configuration

Before it is turned on, the PS/2 Interface must be configured through bits in the PS2CSR register and the PAGPUCR register (see Section 8). This step defines the pull-up activation, the operating frequency (linked to f_{CPU}) and the timing management.

This configuration can not be modified once the PS/2 Interface is turned on, i.e once the PS2ON bit in the PS/2 Control Register (PS2CR) is set.

Note: the PS/2 pull-ups must not be disabled while the PS/2 peripheral is on.

Timing management

If Timing Management is enabled i.e if the TOFF bit in the PS2CSR register is cleared, the PS/2 Interface can handle 3 different timeouts.

– Timeout TX which represents a 100µs inter-frame timing: no transmission will start before the end of this timeout.

– Timeout TX Inhibit which represents a 100µsec time between an inhibit and the start of a transmission.

– Timeout RX which represents the 100µs Device response time to a Host Request To Send (HRTS): no reception will start before the end of this timeout.

It is recommended to always use the timing management function provided. This saves the user from the necessity of using software to manage the timing.

Idle mode

After the PS/2 Interface is configured, it can be enabled by setting the PS2ON bit in the PS2CR register. Once on, its state goes from init to idle. In Idle the cell is waiting either for:

– a transmission start command from the Device

– an inhibit

– an HRTS.

After timeout, the PS/2 Interface will go into idle mode also.

The Device can not force the PS/2 Interface into receive mode when no HRTS has been transmitted.

Transmission mode

When the Device wants to send a byte to the host, it puts the PS/2 Interface into transmit mode by clearing the CD bit in the PS2CR register and writing the data in the PS2DR register.

As soon as the data is written and Timeout TX has expired (if timing management is enabled), the transmission starts and the RTXS bit in the PS2CR register is set by hardware. If timing management is not enabled, timing must be handled by software.

At the end of the transmission, the RTXS bit in the PS2CR register is cleared, and the RTXC flag in the PS2CSR register is set, both by hardware. A Transmission Complete interrupt will be generated if the RTXCIE bit in the PS2CR register is set.

A transmission can be aborted either by the user or by the host. To abort a transmission, the user can clear the RTXS bit in the PS2CR register. The host can also abort it by generating either an inhibit (ck at 0) or an HRTS (data at 0).

In each case, the RTXS bit in the PS2CR register is cleared and an abort flag is set by hardware (RTXA bit in the PS2CSR register). An Abort interrupt will be generated if the RTXAIE bit in the PS2CR register is set.

When the abort comes from an HRTS, the HRTS flag is set and an interrupt will be generated if the HRTS Interrupt Enable in the PS2CR is set.

No error can occur in transmission mode. A line error on the clock line will be interpreted as an inhibit. A line error on the data line will be interpreted as an HRTS.

PS/2 INTERFACE (Cont'd)

Reception mode

As soon as the PS/2 Interface receives an HRTS, which may or may not be preceded by an inhibit, the state goes into idle_rx mode after the Timeout RX delay, if timing management is enabled. The state change is immediate if timing management is not enabled, unless otherwise managed by software.

To start reception, the user must first switch the PS/2 Interface to reception mode by setting the CD bit in the PS2CR register. Then, the RTXS bit in the PS2CR register must be set.

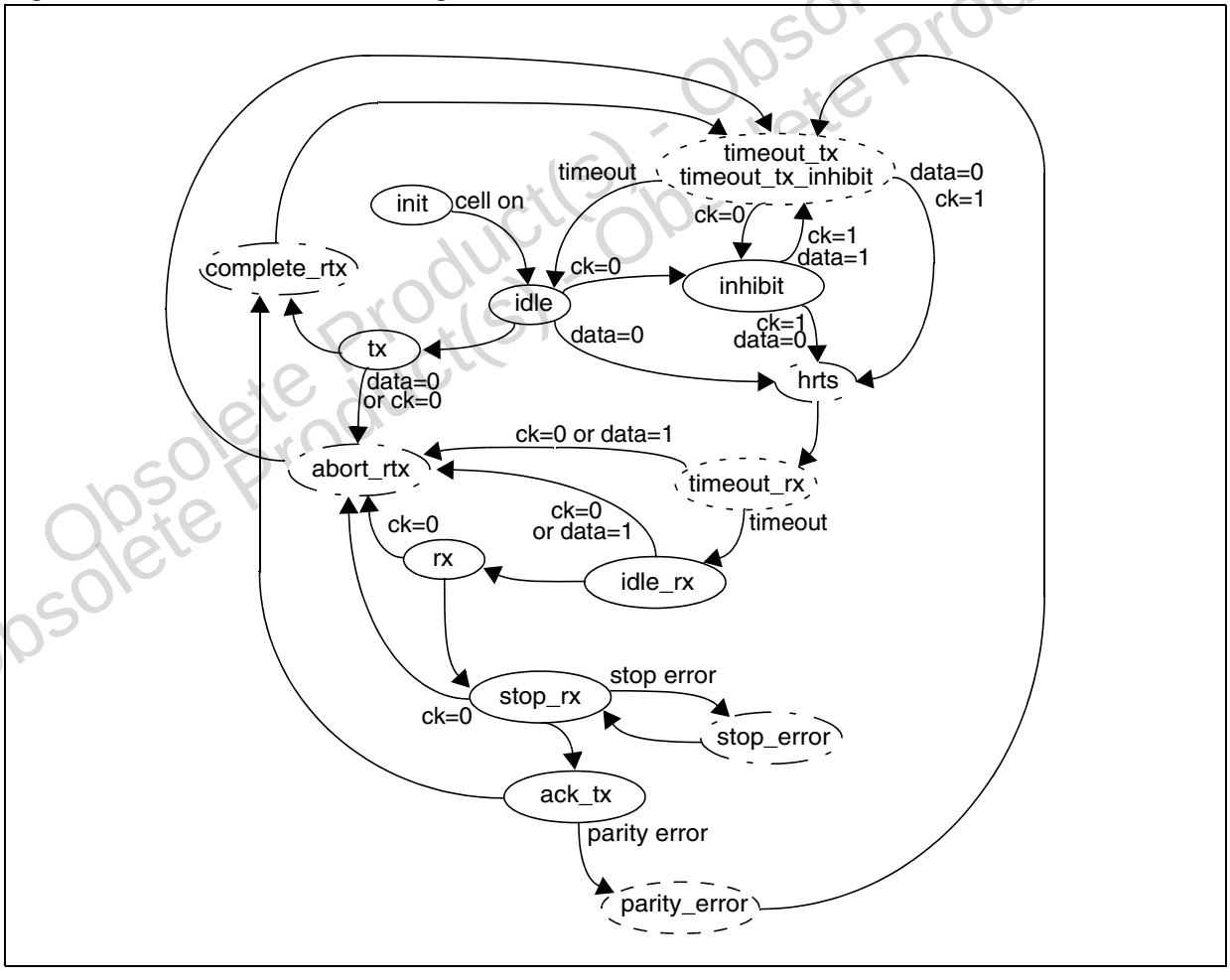
At the end of reception, the RTXS bit in the PS2CR register is cleared by hardware, signalling that reception is over. RTXC flag in the PS2CSR register is also set meaning that reception is complete. If the Reception Complete Interrupt is enabled (the RTXCIE bit in the PS2CR register), an interrupt is generated. An acknowledge is automatically sent to the host.

The reception can be aborted only by the host by generating either an inhibit or a Host Request To Send (HRTS).

If reception is aborted, the RTXS bit in the PS2CR register is cleared by hardware and the abort flag is set (the RTXA bit in the PS2CSR register). If the Abort Interrupt is enabled, an interrupt is generated (the RTXAIE bit in the PS2CR register).

When reception is aborted due to an HRTS flag, in order to receive data from the host, the HRTS Interrupt in the PS2CR register must be set. After the interrupt, and Timeout RX delay, if timing management is enabled, reception from the host occurs. If timing management is not used, reception configuration and start up must be managed by software.

Figure 63. PS/2 Interface State Diagram



PS/2 INTERFACE (Cont'd)

9.6.5 Error Management

The PS/2 Interface can generate 2 types of error during reception: parity error and stop bit error.

9.6.5.1 Parity Error

Parity error occurs when the parity of the host frame is wrong.

In case of parity error, the RXE bit in the PS2CSR register is set. At the same time the RTXS is cleared and RTXC is set, indicating the end of reception.

If the error interrupt is enabled, the RXEIE in the PS2CR register, an interrupt is generated.

An acknowledge is sent to the host after reception of the stop bit.

9.6.5.2 Stop bit Error

Stop bit error occurs if the stop bit Is missing in the host frame. The PS/2 Interface continues to output the clock until a stop bit is received.

At the first missing stop bit, the RXE flag in the PS2CSR register is set, even if an end of reception has not been received. Then, the RTXC flag in the PS2CSR register is set.

An interrupt is generated if the RXEIE in the PS2CR register is set.

The end of the reception is still managed as previously described. The RTXS will be cleared by hardware.

An acknowledge is sent to the host after the reception of the stop bit.

Note:

During transmission, no error can occur as:

– A clock line stuck at 0 is interpreted as an abort.

– A data line stuck at 0 is interpreted as a Host Request To Send (HRTS) thus, it will abort the current transmission and activate both Bit 1 and Bit 2 in the PS2CSR.

Lines stuck at 1 are ignored.

In case of error or abort, the data in the PS2DR register is lost, and the register returns to its reset state (00h).

9.6.6 Low Power Modes

Mode	Description
WAIT	No effect on PS/2. PS/2 interrupt events cause the Device to exit from WAIT mode.
HALT	PS/2 registers are frozen. In HALT mode, the PS/2 is inactive. PS/2 operation resumes when the Device is woken up by an interrupt with "exit from HALT mode" capability.

9.6.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
RTX Complete	RTXC	RTXCIE	Yes	No
RTX Aborted	RTXA	RTXAIE		
Host Request To Send	HRTS	HRTSIE		
RX Error	RXE	RXEIE		

Note: All the interrupts are connected to the same interrupt line.

PS/2 INTERFACE (Cont'd)

9.6.8 Register Description

PS/2 CONTROL REGISTER (PS2CR)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
-	RXEIE	HRT-SIE	RTXAIE	RTX-CIE	RTXS	CD	PS2ON

Bit 7 = Reserved, must be kept cleared.

Bit 6 = **RXEIE** *RX Error Interrupt Enable.*
This bit is set and cleared by software.
0: RX Error interrupt disabled
1: RX Error interrupt enabled

Bit 5 = **HRTSIE** *HRTS Interrupt Enable.*
This bit is set and cleared by software.
0: HRTS interrupt disabled
1: HRTS interrupt enabled

Bit 4 = **RTXAIE** *RX/TX Abort Interrupt Enable.*
This bit is set and cleared by software.
0: RX/TX Abort interrupt disabled
1: RX/TX Abort interrupt enabled

Bit 3 = **RTXCIE** *RX/TX Complete Interrupt Enable*.
This bit is set and cleared by software.
0: RX/TX Complete interrupt disabled
1: RX/TX Complete interrupt enabled

Bit 2 = **RTXS** *RX/TX Start.*
This bit is set by hardware by writing a value in the PS2DR register in transmit mode, and set by software after a Host Request To Send in receive mode.
This bit is cleared by software to abort a transmission. It is cleared by hardware when communication is over.
It is not possible to abort a reception.
0: RX/TX over / Abort TX
1: Start RX/TX

Bit 1 = **CD** *Communication Direction.*
This bit is set and cleared by software. It can only be set when no communication is active (RTXS bit is cleared).
0: TX
1: RX.

Bit 0 = **PS2ON** *PS/2 ON*
This bit is set and cleared by software.
0: PS/2 Interface is OFF
1: PS/2 Interface is ON

PS/2 INTERFACE (Cont'd)

PS/2 CONFIG/STATUS REGISTER (PS2CSR)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
0	FSEL1	FSEL0	TOFF	RXE	HRTS	RTXA	RTXC

Bit 7 = Reserved, must be kept cleared.

Bit 6:5 = **FSEL[1:0]** *Frequency Selection*
This bit is set and cleared by software when the PS/2 Interface is OFF.
It selects the division ratio according to the internal frequency f_{CPU}.
00: 12 MHz
01: 8 MHz
10: 6 MHz
11: 4 MHz

Bit 4 = **TOFF** *Timing OFF*
This bit is set and cleared by software when the PS/2 Interface is OFF.
0: Timing management enabled
1: Timing management disabled

Bit 3 = **RXE** *RX Error.*
This bit is set by hardware and cleared by software (by writing 1).
If the RXEIE bit in the PS2CR register is set, an interrupt is generated.
0: No Error
1: Reception Error

Bit 2 = **HRTS** *Host Request To Send.*
This bit is set by hardware and cleared by software (by writing 1).
If the HRTSIE bit in the PS2CR register is set, an interrupt is generated.
0: No request
1: Host request to send

Bit 1 = **RTXA** *RX/TX Abort.*
This bit is set by hardware and cleared by software (by writing 1).
If the RTXAIE bit in the PS2CR Register is set, an interrupt is generated.
0: No abort
1: RX or TX aborted

Bit 0 = **RTXC** *RX/TX Complete.*
This bit is set by hardware and cleared by software (by writing 1).
If the RTXCIE bit in the PS2CR Register is set, an interrupt is generated.
0: No communication / Communication ongoing
1: RX or TX complete

Note: In case of abort, the RTXC bit is not set.

PS/2 DATA REGISTER (PS2DR)
Read / Write
Reset Value: 0000 0000 (00h)

7							0
D7	D6	D5	D4	D3	D2	D1	D0

Bit 7:0 = **PS2D[7:0]** *PS/2 Data.*
These bits are written by software when no communication is in progress i.e the RTXS bit in the PS2CR register is cleared and the interface is configured in transmit mode, i.e the CD bit is set.
Writing a value in this register will automatically set the RTXS bit and start transmission.
These bits are written by hardware during reception.
They can be read only when no communication is in progress i.e the RTXS bit in the PS2CR register is cleared.

Note: During communication, these bits return the value 00h when read.

PS/2 INTERFACE (Cont'd)

Table 33. PS/2 Register Map and Reset Values

Address (Hex.)	Register Name	7	6	5	4	3	2	1	0
PS/2 0: 4Eh PS/2 1: 51h	PS2CnR Reset Value	0 0	RXEIE 0	HRTSIE 0	RTXAIE 0	RTXCIE 0	RTXS 0	CD 0	PS2ON
PS/2 0: 4Fh PS/2 1: 52h	PS2CSnR Reset Value	0 0	FSEL1 0	FSEL0 0	TOFF 0	RXE 0	HRTS 0	RTXA 0	RTXC 0
PS/2 0: 50h PS/2 1: 53h	PS2DnR Reset Value	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0	D1 0	D0 0

ON-CHIP PERIPHERALS (cont'd)

9.7 SERIAL PERIPHERAL INTERFACE (SPI)

9.7.1 Introduction

The Serial Peripheral Interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

9.7.2 Main Features

- Full duplex synchronous transfers (on three lines)
- Simplex synchronous transfers (on two lines)
- Master or slave operation
- 6 master mode frequencies (f_{CPU}/4 max.)
- f_{CPU}/2 max. slave mode frequency (see note)
- $\overline{SS}$ Management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, Master Mode Fault and Overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

9.7.3 General Description

Figure 64 on page 126 shows the serial peripheral interface (SPI) block diagram. There are three registers:

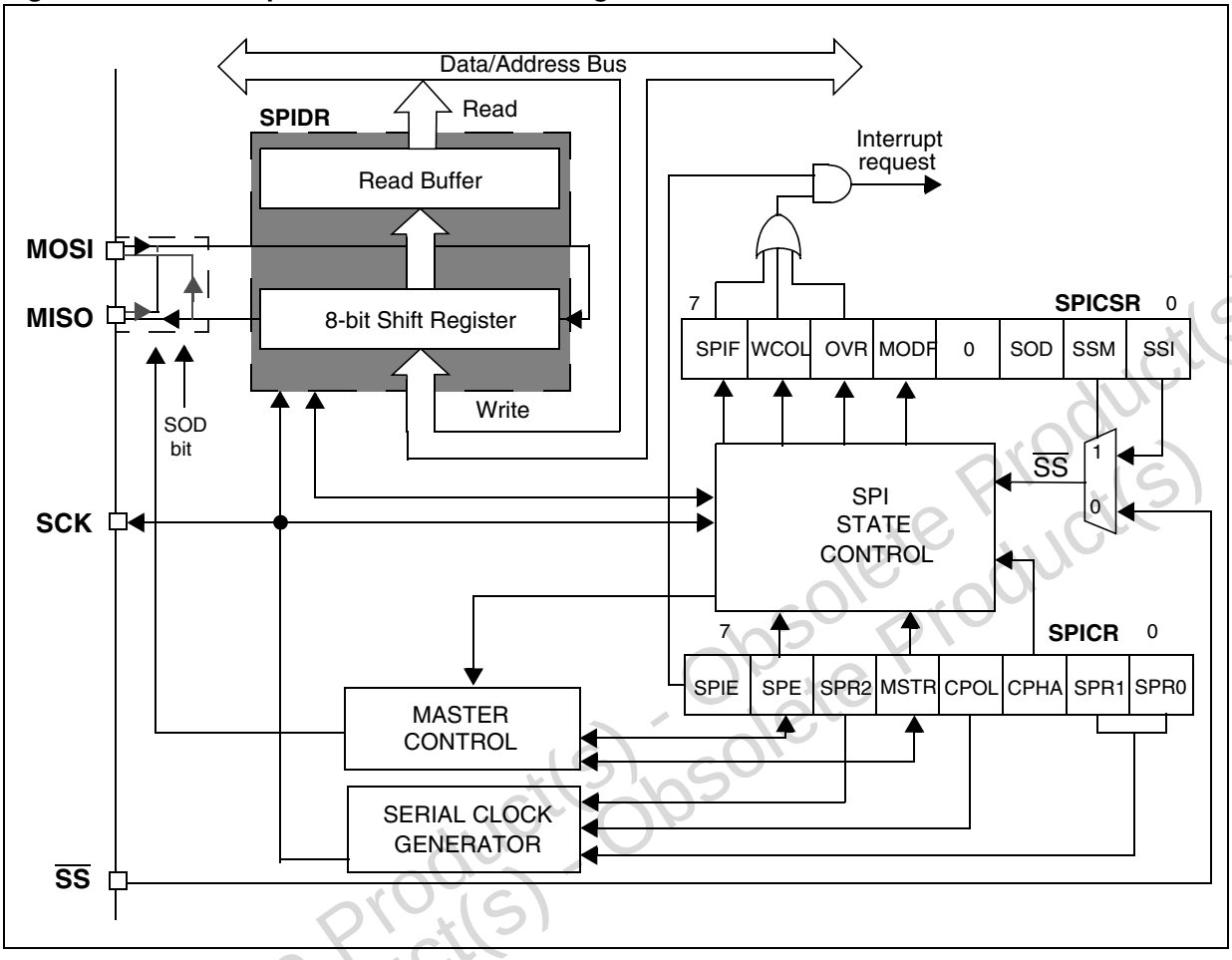
- SPI Control Register (SPICR)
- SPI Control/Status Register (SPICSR)
- SPI Data Register (SPIDR)

The SPI is connected to external devices through four pins:

- MISO: Master In / Slave Out data
- MOSI: Master Out / Slave In data
- SCK: Serial Clock out by SPI masters and input by SPI slaves
- $\overline{SS}$: Slave select:
  This input signal acts as a 'chip select' to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave $\overline{SS}$ inputs can be driven by standard I/O ports on the master Device.

SERIAL PERIPHERAL INTERFACE (SPI) (cont'd)

Figure 64. Serial Peripheral Interface Block Diagram



SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.3.1 Functional Description

A basic example of interconnections between a single master and a single slave is illustrated in Figure 65.

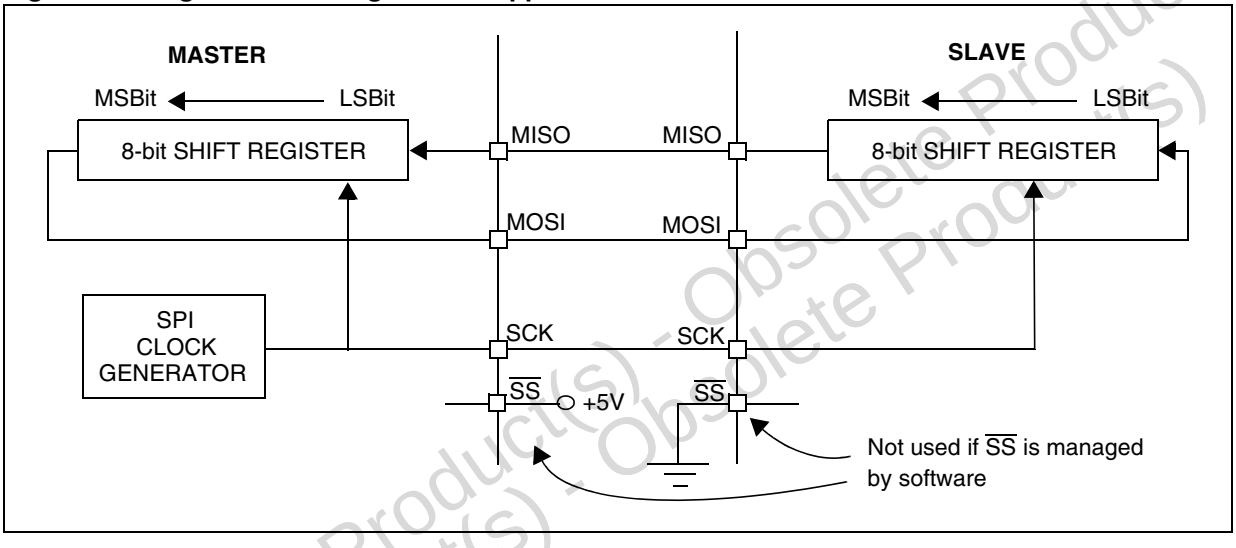
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

The communication is always initiated by the master. When the master device transmits data to a slave device via MOSI pin, the slave device responds by sending data to the master device via the MISO pin. This implies full duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

To use a single data line, the MISO and MOSI pins must be connected at each node (in this case only simplex communication is possible).

Four possible data/clock timing relationships may be chosen (see Figure 68 on page 130) but master and slave must be programmed with the same timing mode.

Figure 65. Single Master/ Single Slave Application



SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.3.2 Slave Select Management

As an alternative to using the $\overline{SS}$ pin to control the Slave Select signal, the application can choose to manage the Slave Select signal by software. This is configured by the SSM bit in the SPICSR register (see Figure 67).

In software management, the external $\overline{SS}$ pin is free for other application uses and the internal $\overline{SS}$ signal level is driven by writing to the SSI bit in the SPICSR register.

In Master mode:

– $\overline{SS}$ internal must be held high continuously

In Slave Mode:

There are two cases depending on the data/clock timing relationship (see Figure 66):

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

– $\overline{SS}$ internal must be held low during the entire transmission. This implies that in single slave applications the $\overline{SS}$ pin either can be tied to V_{SS}, or made free for standard I/O by managing the $\overline{SS}$ function by software (SSM = 1 and SSI = 0 in the in the SPICSR register)

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

– $\overline{SS}$ internal must be held low during byte transmission and pulled high between each byte to allow the slave to write to the shift register. If $\overline{SS}$ is not pulled high, a Write Collision error will occur when the slave writes to the shift register (see Section 9.7.5.3).

Figure 66. Generic $\overline{SS}$ Timing Diagram

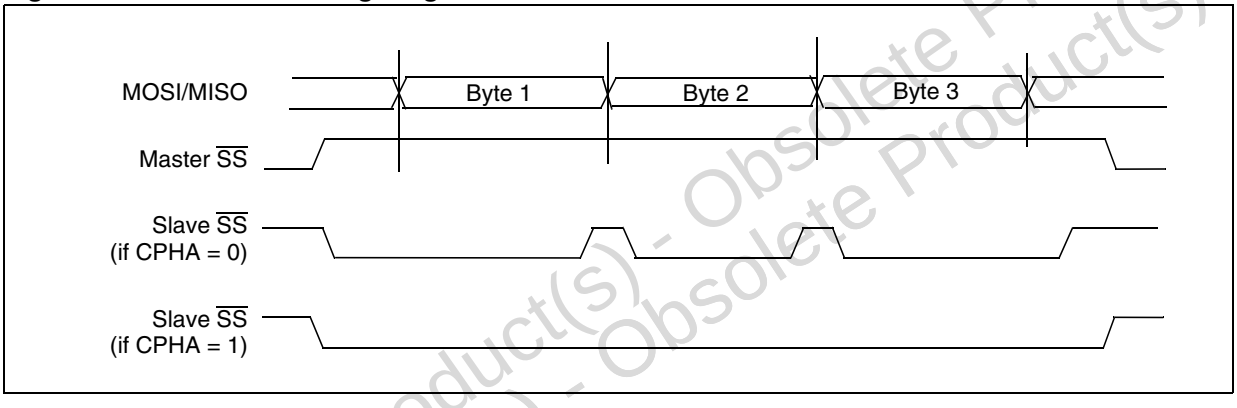
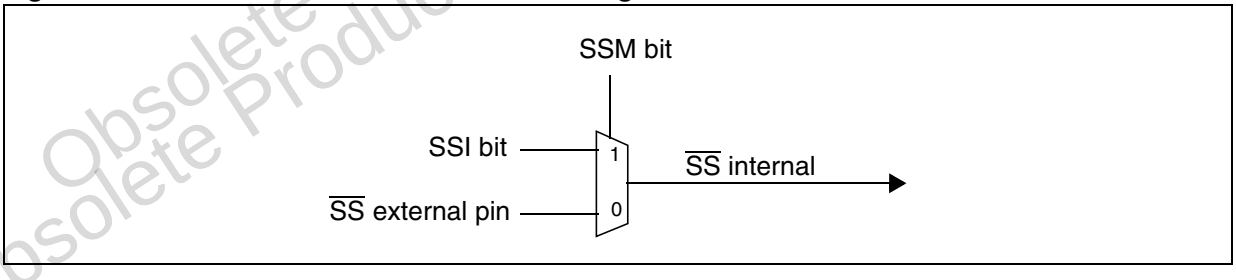


Figure 67. Hardware/Software Slave Select Management



SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.3.3 Master Mode Operation

In master mode, the serial clock is output on the SCK pin. The clock frequency, polarity and phase are configured by software (refer to the description of the SPICSR register).

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

How to operate the SPI in master mode

To operate the SPI in master mode, perform the following steps in order:

1. Write to the SPICR register:
   - Select the clock frequency by configuring the SPR[2:0] bits.
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits. Figure 68 shows the four possible configurations.
     Note: The slave must have the same CPOL and CPHA settings as the master.
2. Write to the SPICSR register:
   - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the SS pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
   - Set the MSTR and SPE bits
     Note: MSTR and SPE bits remain set only if SS is high).

Important note: if the SPICSR register is not written first, the SPICR register setting (MSTR bit) may be not taken into account.

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

9.7.3.4 Master Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if the SPIE bit is set and the interrupt mask in the CCR register is cleared.

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

1. An access to the SPICSR register while the SPIF bit is set
2. A read to the SPIDR register

Note: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

9.7.3.5 Slave Mode Operation

In slave mode, the serial clock is received on the SCK pin from the master device.

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see Figure 68).
     Note: The slave must have the same CPOL and CPHA settings as the master.
   - Manage the $\overline{SS}$ pin as described in Section 9.7.3.2 and Figure 66. If CPHA = 1 $\overline{SS}$ must be held low continuously. If CPHA = 0 $\overline{SS}$ must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

9.7.3.6 Slave Mode Transmit Sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if SPIE bit is set and interrupt mask in the CCR register is cleared.

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

1. An access to the SPICSR register while the SPIF bit is set
2. A write or a read to the SPIDR register

Notes: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

The SPIF bit can be cleared during a second transmission; however, it must be cleared before the second SPIF bit in order to prevent an Overrun condition (see Section 9.7.5.2).

SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.4 Clock Phase and Clock Polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 68).

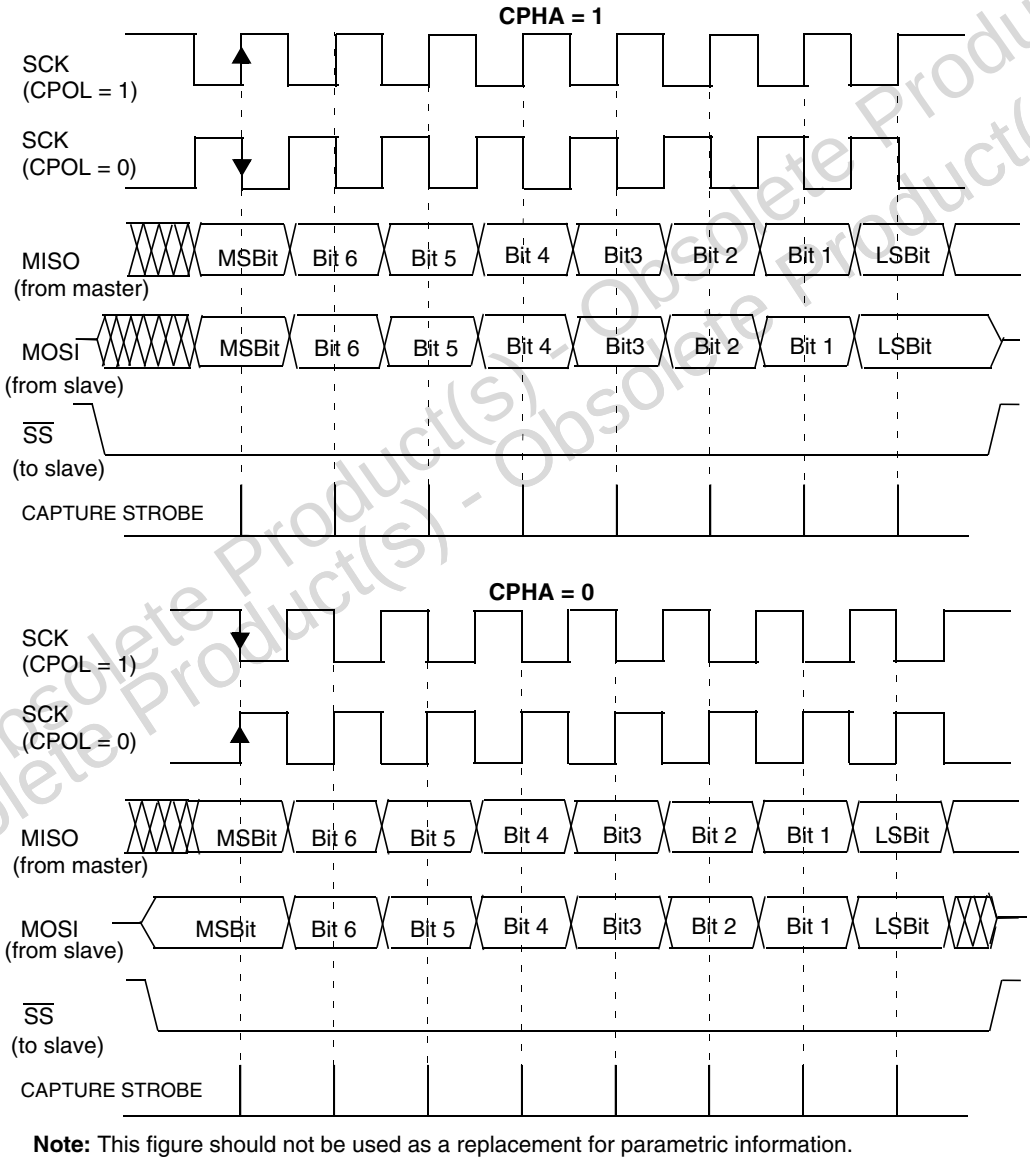
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL = 1 or pulling down SCK if CPOL = 0).

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

Figure 68 shows an SPI transfer with the four combinations of the CPHA and CPOL bits. The diagram may be interpreted as a master or slave timing diagram where the SCK pin, the MISO pin and the MOSI pin are directly connected between the master and the slave device.

Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.

Figure 68. Data Clock Timing Diagram



Note: This figure should not be used as a replacement for parametric information. Refer to the Electrical Characteristics chapter.

SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.5 Error Flags

9.7.5.1 Master Mode Fault (MODF)

Master mode fault occurs when the master device's $\overline{SS}$ pin is pulled low.

When a Master mode fault occurs:

– The MODF bit is set and an SPI interrupt request is generated if the SPIE bit is set.

– The SPE bit is reset. This blocks all output from the device and disables the SPI peripheral.

– The MSTR bit is reset, thus forcing the device into slave mode.

Clearing the MODF bit is done through a software sequence:

1. A read access to the SPICSR register while the MODF bit is set.

2. A write to the SPICR register.

Notes: To avoid any conflicts in an application with multiple slaves, the $\overline{SS}$ pin must be pulled high during the MODF bit clearing sequence. The SPE and MSTR bits may be restored to their original state during or after this clearing sequence.

Hardware does not allow the user to set the SPE and MSTR bits while the MODF bit is set except in the MODF bit clearing sequence.

In a slave device, the MODF bit can not be set, but in a multimaster configuration the device can be in slave mode with the MODF bit set.

The MODF bit indicates that there might have been a multimaster conflict and allows software to handle this using an interrupt routine and either perform a reset or return to an application default state.

9.7.5.2 Overrun Condition (OVR)

An overrun condition occurs when the master device has sent a data byte and the slave device has not cleared the SPIF bit issued from the previously transmitted byte.

When an Overrun occurs:

– The OVR bit is set and an interrupt request is generated if the SPIE bit is set.

In this case, the receiver buffer contains the byte sent after the SPIF bit was last cleared. A read to the SPIDR register returns this byte. All other bytes are lost.

The OVR bit is cleared by reading the SPICSR register.

9.7.5.3 Write Collision Error (WCOL)

A write collision occurs when the software tries to write to the SPIDR register while a data transfer is taking place with an external device. When this happens, the transfer continues uninterrupted and the software write will be unsuccessful.

Write collisions can occur both in master and slave mode. See also Section 9.7.3.2 "Slave Select Management" on page 128.

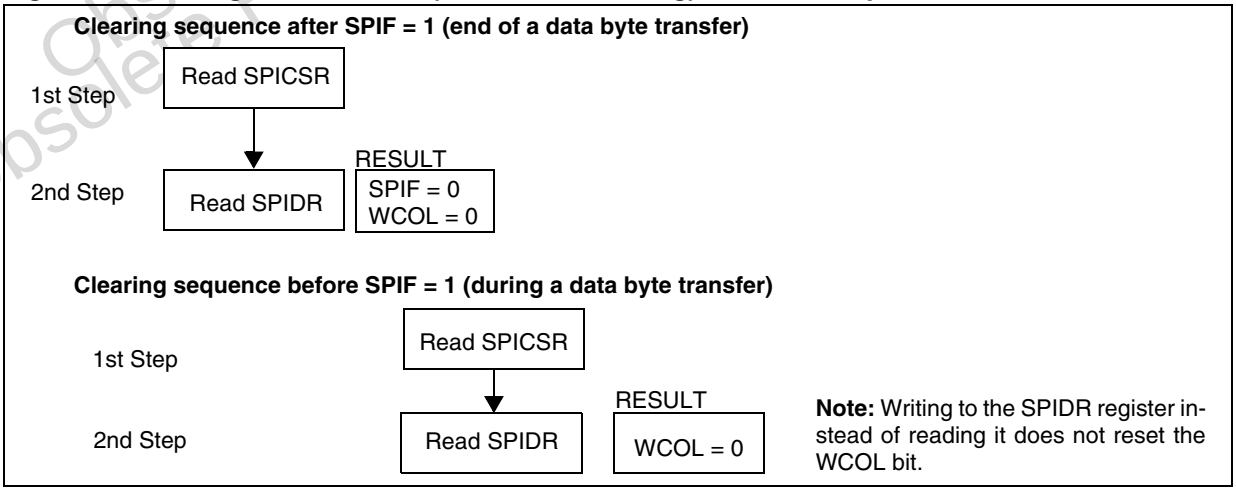
Note: A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

The WCOL bit in the SPICSR register is set if a write collision occurs.

No SPI interrupt is generated when the WCOL bit is set (the WCOL bit is a status flag only).

Clearing the WCOL bit is done through a software sequence (see Figure 69).

Figure 69. Clearing the WCOL Bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.5.4 Single Master and Multimaster Configurations

There are two types of SPI systems:

– Single Master System

– Multimaster System

Single Master System

A typical single master system may be configured using a device as the master and four devices as slaves (see Figure 70).

The master device selects the individual slave devices by using four pins of a parallel port to control the four $\overline{SS}$ pins of the slave devices.

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

Note: To prevent a bus conflict on the MISO line, the master allows only one active slave device during a transmission.

For more security, the slave device may respond to the master with the received data byte. Then the master will receive the previous byte back from the slave device if all MISO and MOSI pins are connected and the slave has not written to its SPIDR register.

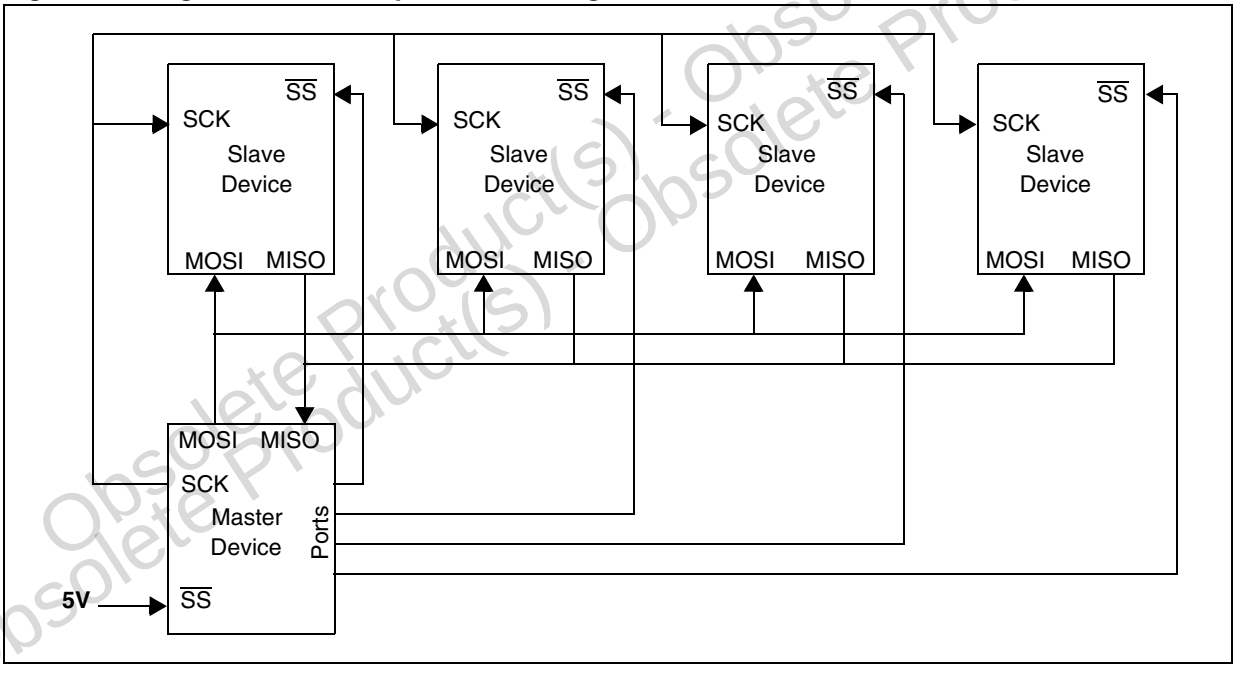
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

Multimaster System

A multimaster system may also be configured by the user. Transfer of master control could be implemented using a handshake method through the I/O ports or by an exchange of code messages through the serial peripheral interface system.

The multimaster system is principally handled by the MSTR bit in the SPICR register and the MODF bit in the SPICSR register.

Figure 70. Single Master / Multiple Slave Configuration



SERIAL PERIPHERAL INTERFACE (cont'd)

9.7.6 Low Power Modes

Mode	Description
WAIT	No effect on SPI. SPI interrupt events cause the device to exit from WAIT mode.
HALT	SPI registers are frozen. In HALT mode, the SPI is inactive. SPI operation resumes when the device is woken up by an interrupt with "exit from HALT mode" capability. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetching). If several data are received before the wake-up event, then an overrun error is generated. This error can be detected after the fetch of the interrupt routine that woke up the Device.

9.7.6.1 Using the SPI to wake up the device from Halt mode

In slave configuration, the SPI is able to wake up the device from HALT mode through a SPIF interrupt. The data received is subsequently read from the SPIDR register when the software is running (interrupt vector fetch). If multiple data transfers have been performed before software clears the SPIF bit, then the OVR bit is set by hardware.

Note: When waking up from HALT mode, if the SPI remains in Slave mode, it is recommended to perform an extra communications cycle to bring the SPI from HALT mode state to normal state. If the SPI exits from Slave mode, it returns to normal state immediately.

Caution: The SPI can wake up the device from HALT mode only if the Slave Select signal (external $\overline{SS}$ pin or the SSI bit in the SPICSR register) is low when the device enters HALT mode. So, if Slave selection is configured as external (see Section 9.7.3.2), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters HALT mode.

9.7.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
SPI End of Transfer Event	SPIF	SPIE	Yes	Yes
Master Mode Fault Event	MODF			No
Overrun Error	OVR			

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

9.7.8 Register Description

SPI CONTROL REGISTER (SPICR)

Read/Write

Reset Value: 0000 xxxx (0xh)

7							0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0

Bit 7 = **SPIE** *Serial Peripheral Interrupt Enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SPI interrupt is generated whenever an End of Transfer event, Master Mode Fault or Overrun error occurs (SPIF = 1, MODF = 1 or OVR = 1 in the SPICSR register)

Bit 6 = **SPE** *Serial Peripheral Output Enable*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see Section 9.7.5.1 "Master Mode Fault (MODF)" on page 131). The SPE bit is cleared by reset, so the SPI peripheral is not initially connected to the external pins.
0: I/O pins free for general purpose I/O
1: SPI I/O pin alternate functions enabled

Bit 5 = **SPR2** *Divider Enable*
This bit is set and cleared by software and is cleared by reset. It is used with the SPR[1:0] bits to set the baud rate. Refer to Table 34.
0: Divider by 2 enabled
1: Divider by 2 disabled

Note: This bit has no effect in slave mode.

Bit 4 = **MSTR** *Master Mode*
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see Section 9.7.5.1 "Master Mode Fault (MODF)" on page 131).
0: Slave mode
1: Master mode. The function of the SCK pin changes from an input to an output and the functions of the MISO and MOSI pins are reversed.

Bit 3 = **CPOL** *Clock Polarity*
This bit is set and cleared by software. This bit determines the idle state of the serial Clock. The CPOL bit affects both the master and slave modes.
0: SCK pin has a low level idle state
1: SCK pin has a high level idle state

Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.

Bit 2 = **CPHA** *Clock Phase*
This bit is set and cleared by software.
0: The first clock transition is the first data capture edge.
1: The second clock transition is the first capture edge.

Note: The slave must have the same CPOL and CPHA settings as the master.

Bits 1:0 = **SPR[1:0]** *Serial Clock Frequency*
These bits are set and cleared by software. Used with the SPR2 bit, they select the baud rate of the SPI serial clock SCK output by the SPI in master mode.

Note: These 2 bits have no effect in slave mode.

Table 34. SPI Master Mode SCK Frequency

Serial Clock	SPR2	SPR1	SPR0
$f_{CPU}/4$	1	0	0
$f_{CPU}/8$	0		
$f_{CPU}/16$			1
$f_{CPU}/32$	1	1	0
$f_{CPU}/64$	0		
$f_{CPU}/128$			1

SERIAL PERIPHERAL INTERFACE (cont'd)

SPI CONTROL/STATUS REGISTER (SPICSR)
Read/Write (some bits Read Only)
Reset Value: 0000 0000 (00h)

7							0
SPIF	WCOL	OVR	MODF	-	SOD	SSM	SSI

Bit 7 = **SPIF** *Serial Peripheral Data Transfer Flag (Read only)*
This bit is set by hardware when a transfer has been completed. An interrupt is generated if SPIE = 1 in the SPICR register. It is cleared by a software sequence (an access to the SPICSR register followed by a write or a read to the SPIDR register).
0: Data transfer is in progress or the flag has been cleared.
1: Data transfer between the device and an external device has been completed.

Note: While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Bit 6 = **WCOL** *Write Collision status (Read only)*
This bit is set by hardware when a write to the SPIDR register is done during a transmit sequence. It is cleared by a software sequence (see Figure 69).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = **OVR** S*PI Overrun error (Read only)*
This bit is set by hardware when the byte currently being received in the shift register is ready to be transferred into the SPIDR register while SPIF = 1 (See Section 9.7.5.2). An interrupt is generated if SPIE = 1 in the SPICR register. The OVR bit is cleared by software reading the SPICSR register.
0: No overrun error
1: Overrun error detected

Bit 4 = **MODF** *Mode Fault flag (Read only)*
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 9.7.5.1 "Master Mode Fault (MODF)" on page 131). An SPI interrupt can be generated if SPIE = 1 in the SPICR register. This bit is cleared by a software sequence (An access to the SPICSR register while MODF = 1 followed by a write to the SPICR register).
0: No master mode fault detected
1: A fault in master mode has been detected

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **SOD** *SPI Output Disable*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode / MISO in slave mode)
0: SPI output enabled (if SPE = 1)
1: SPI output disabled

Bit 1 = **SSM** $\overline{SS}$ *Management*
This bit is set and cleared by software. When set, it disables the alternate function of the SPI $\overline{SS}$ pin and uses the SSI bit value instead. See Section 9.7.3.2 "Slave Select Management" on page 128.
0: Hardware management ($\overline{SS}$ managed by external pin)
1: Software management (internal $\overline{SS}$ signal controlled by SSI bit. External $\overline{SS}$ pin free for general-purpose I/O)

Bit 0 = **SSI** $\overline{SS}$ *Internal Mode*
This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the $\overline{SS}$ slave select signal when the SSM bit is set.
0: Slave selected
1: Slave deselected

SPI DATA I/O REGISTER (SPIDR)
Read/Write
Reset Value: Undefined

7							0
D7	D6	D5	D4	D3	D2	D1	D0

The SPIDR register is used to transmit and receive data on the serial bus. In a master device, a write to this register will initiate transmission/reception of another byte.

Notes: During the last clock cycle the SPIF bit is set, a copy of the received data byte in the shift register is moved to a buffer. When the user reads the serial peripheral data I/O register, the buffer is actually being read.

While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

Warning: A write to the SPIDR register places data directly into the shift register for transmission.

A read to the SPIDR register returns the value located in the buffer and not the content of the shift register (see Figure 64).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 35. SPI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0017h	**SPIDR** Reset Value	MSB x	x	x	x	x	x	x	LSB x
0018h	**SPICR** Reset Value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
0019h	**SPICSR** Reset Value	SPIF 0	WCOL 0	OVR 0	MODF 0	0	SOD 0	SSM 0	SSI 0

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 36. 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 sub-modes:

Direct (short)

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

Direct (long)

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

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 37. 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 sub-modes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

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

INSTRUCTION SET OVERVIEW (Cont'd)

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.

10.2.1 Illegal Opcode Reset

In order to provide enhanced robustness to the device against unexpected behaviour, a system of illegal opcode detection is implemented. If a code to be executed does not correspond to any opcode or prebyte value, a reset is generated. This, combined with the Watchdog, allows the detection and recovery from an unexpected fault or interference.

Note: A valid prebyte associated with a valid opcode forming an unauthorized combination does not generate a reset.

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_{33}=3.3V. They are given only as design guidelines and are not tested.

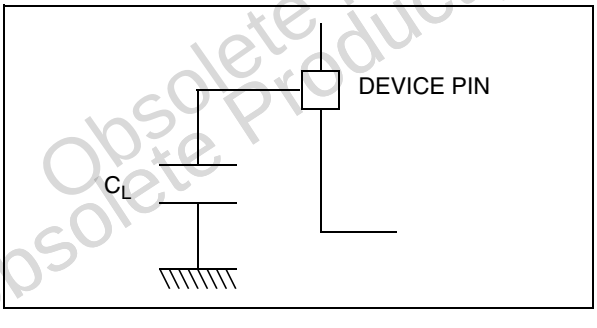
11.1.3 Typical curves

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 71.

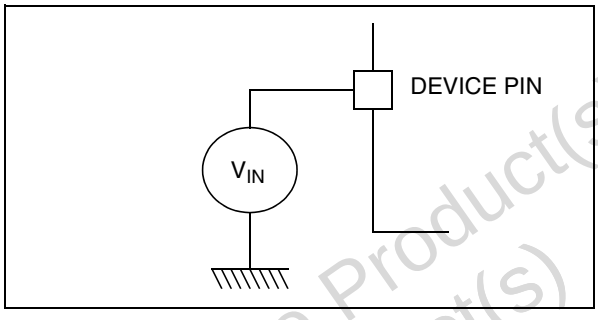
Figure 71. Pin loading conditions



11.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 72.

Figure 72. 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		5.6	V
V_{IN} 1) & 2)	Input Voltage on open drain pin		V_{SS}-0.3 to 5.6	V
	Input Voltage on other pins	Pins n°1 to 16 and 46 to 48	V_{SS}-0.3 to V_{18}+0.3	
		Pins n°17 to 45	V_{SS}-0.3 to V_{33}+0.3	
$\|\Delta V_{DDx}\|$ and $\|\Delta V_{SSx}\|$	Variations between different digital power pins		50	mV
$\|V_{SSA}$ - $V_{SSx}\|$	Variations between digital and analog ground pins		50	
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)		see Section 11.7 on page 153	
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine Model)			

11.2.2 Current Characteristics

Symbol	Ratings	Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) 3)	100	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink) 3)	100	
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	
$\Sigma I_{INJ(PIN)}$ 2)	Total injected current (sum of all I/O and control pins except Port A and G pins)5)	0	
$I_{INJ(PIN)}$ 2)	Injected current on 5V tolerant pins (Port A and G)	± 5	mA
$\Sigma I_{INJ(PIN)}$ 2)	Total injected current (Port A and G pins) 5)	± 15	

11.2.3 Thermal Characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_{JMAX}	Maximum junction temperature 4)	150	°C

Notes:

1. Directly connecting the $\overline{\text{RESET}}$ and I/O pins to V_{33} 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). For the same reason, unused I/O pins must not be directly tied to V_{33} or V_{SS}.
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 $V_{IN}>V_{33}$ while a negative injection is induced by $V_{IN}<V_{SS}$. For true open-drain pads, there is no positive injection current, and the corresponding V_{IN} maximum must always be respected.
3. All power supply (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
4. The average chip-junction temperature can be obtained from the formula $T_J = T_A + P_D$ x RthJA (see Section 12.2 on page 171).
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.

11.3 OPERATING CONDITIONS

11.3.1 General Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
f_{CPU}	Internal clock frequency		0	12	MHz
V_{DD}	Power Supply		4.0	5.6	V
T_A	Ambient temperature range		0	55	°C

11.4 SUPPLY CURRENT CHARACTERISTICS

The following current consumption specified for the Device 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	Max	Unit
$\Delta I_{DD(\Delta Ta)}$	Supply current variation vs. temperature	Constant V_{DD} and f_{CPU}	10	%

11.4.1 RUN and WAIT Modes

Symbol	Parameter	Conditions	Typ [1]	Max [2]	Unit
I_{DD}	Supply current in RUN mode [3][5]	f_{OSC}=12MHz, f_{CPU}=12MHz	1	3.5	mA
		f_{RC_6MHz}=6MHz, f_{CPU}=6MHz	1.5	2	
	Supply current in WAIT mode [4][5]	f_{OSC}=12MHz, f_{CPU}=12MHz	1	1.5	
		f_{RC_6MHz}=6MHz, f_{CPU}=6MHz	1	1.5	

Notes:

1. Typical data are based on T_A=25°C, V_{DD}=5.

2. Data based on characterization results, tested in production at V_{DD} max.

3. CPU running with memory access, all I/O pins in input mode with a static value at V_{33} or V_{SS} (no load), all peripherals in reset state; clock input (OSC1) driven by external square wave.

4. All I/O pins in input mode with a static value at V_{33} or V_{SS} (no load), all peripherals in reset state; clock input (OSC1) driven by external square wave.

5. The given consumption does not take into account the I_{ON} of the 5V/3.3V and 3.3V/1.8V regulators required to operate both the digital and USB peripherals.

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

11.4.2 HALT Modes

Symbol	Parameter	Conditions	Typ [1]	Max	Unit
I_{DD}	Supply current in HALT mode (suspend mode for USB) [2]	POR + 5V/3.3V REG. + 3.3V/1.8 REG., 0°C to 55°C	180	400	µA
		POR + 5V/3.3V REG. + 3.3V/1.8 REG., 0°C to 25°C		250	

11.4.3 Supply and Clock Managers

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

Symbol	Parameter	Conditions	Typ [1]	Max [3]	Unit
$I_{DD(CK)}$	Supply current of internal RC 6MHz oscillator		180	250	µA
	Supply current of crystal oscillator [4] & [5]		500	1000	

11.4.4 On-Chip Peripherals

Symbol	Parameter	Conditions	Typ	Unit
$I_{DD(T16)}$	Timer supply current [6]		210	µA
$I_{DD(TBU)}$	TBU supply current [6]		30	
$I_{DD(SPI)}$	SPI supply current [7]		120	
$I_{DD(PS/2)}$	PS/2 supply current [8]		1.120	mA

Notes:

1. Typical data are based on T_A=25°C and f_{CPU}=12MHz.
2. All I/O pins in input mode with a static value at V_{33} or V_{SS} (no load). Data based on characterization results, tested in production at V_{DD} max (pull-up consumption not included).
3. Data based on characterization results, not tested in production.
4. Data based on characterization results done with the external components specified in Section 11.5.4, not tested in production.
5. As the oscillator is based on a current source, the consumption does not depend on the voltage.
6. Data based on a differential I_{DD} measurement between reset configuration (timer stopped) and timer counter enabled (only TCE bit set).
7. Data based on a differential I_{DD} measurement between reset configuration (SPI disabled) and a permanent SPI master communication at maximum speed (data sent equal to 55h).This measurement includes the pad toggling consumption.
8. Data based on a differential I_{DD} measurement between reset configuration (PS/2 disabled) and a permanent PS/2 communication (data sent equal to 55h).This measurement includes the pad toggling consumption.

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

11.4.5 RF Peripheral

Table 38. Consumption in Reception mode

Symbol	Parameter	Conditions	Min [1]	Typ[1]	Max[1]	Unit
1 Receiver running						
I_{DD-PM}	Current consumption for Power Management	RX0 or RX1 voltage regulators only	1[1]	1.1[1]	1.3[1]	mA
I_{DD-RF}	RX0 or RX1 current consumption	RX0 or RX1 RF block only		7[1]	9[1]	mA
I_{DD}	Total Current consumption[2]	Reception with RX0 or RX1		13[1]	15[1]	mA
2 Receivers running						
$I_{DD-PM01}$	Current consumption for Power Management	RX0 & RX1 voltage regulators only	1.6[1]	1.8[1]	2[1]	mA
$I_{DD-RF01}$	Current consumption on RX0 & RX1	RX0 & RX1 RF blocks only		14[1]	15[1]	mA
I_{DD-01}	Total Current consumption[2]	Reception		21[1]	23[1]	mA

Notes:

1. Based on characterization results. Not tested in production.

2. CPU running with memory access, all I/O pins in input mode with static value at V_{33} or V_{SS} (no load), all peripherals in reset state (except RX block), oscillator driven by a crystal, associated power supply management.

11.5 CLOCK AND TIMING CHARACTERISTICS

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

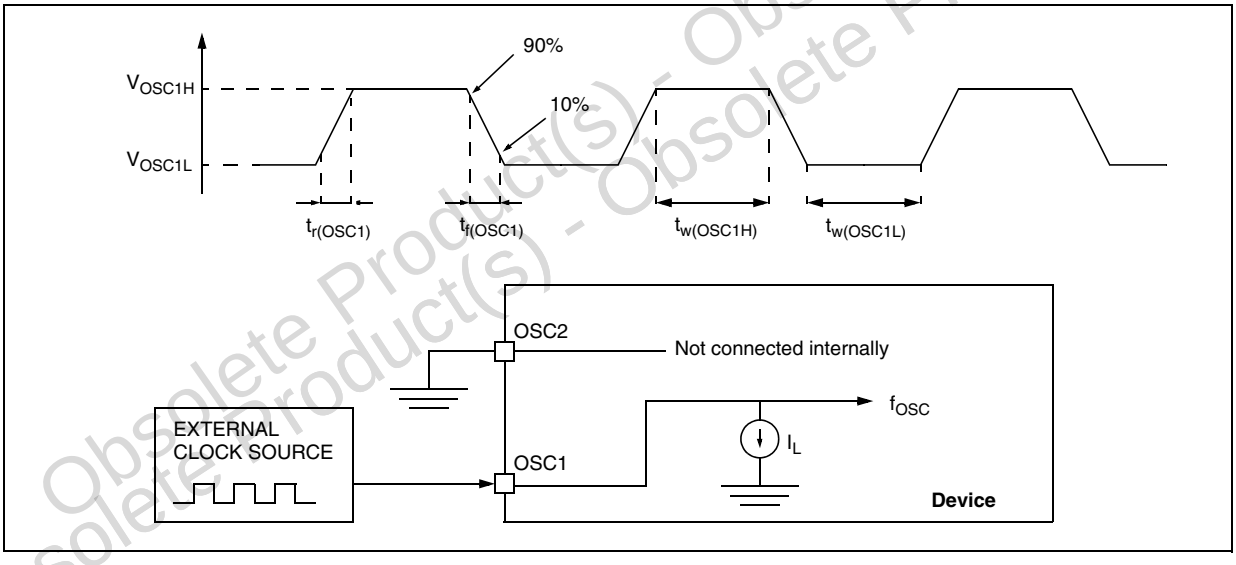
11.5.1 General Timings

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
$t_{c(INST)}$	Instruction cycle time		2	3	12	t_{CPU}
		f_{CPU}=12MHz	167	250	1000	ns
$t_{v(IT)}$	Interrupt reaction time [2] $t_{v(IT)} = \Delta t_{c(INST)} + 10$		10		22	t_{CPU}
		f_{CPU}=12MHz	0.833		1.833	µs

11.5.2 External Clock Source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSC1H}	OSC1 input pin high level voltage	see Figure 73	$0.7xV_{33}$		V_{33}	V
V_{OSC1L}	OSC1 input pin low level voltage		V_{SS}		$0.3xV_{33}$	
$t_{w(OSC1H)}$ $t_{w(OSC1L)}$	OSC1 high or low time [3]		15			ns
$t_{r(OSC1)}$ $t_{f(OSC1)}$	OSC1 rise or fall time [3]				15	
I_L	OSCx Input leakage current	$V_{SS} \le V_{IN} \le V_{33}$			±1	µA

Figure 73. Typical Application with an External Clock Source



Notes:

1. Data based on typical application software.
2. Time measured between interrupt event and interrupt vector fetch. $\Delta t_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.
3. Data based on design simulation and/or technology characteristics, not tested in production.

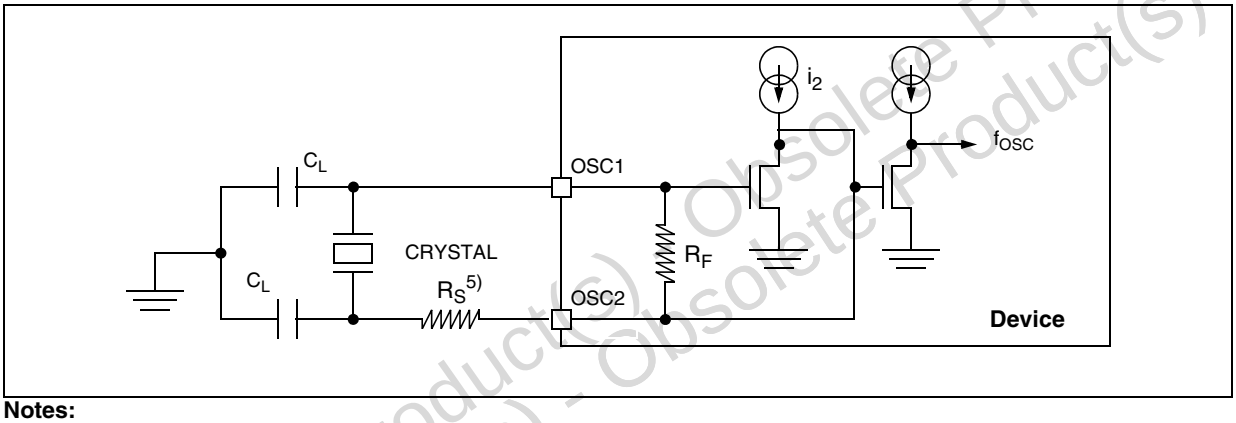
CLOCK AND TIMING CHARACTERISTICS (Cont'd)

11.5.3 Crystal Oscillator

The Device internal clock is supplied from a crystal oscillator. All the information given in this paragraph are based on characterization results with specified typical external components. In the application the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. Refer to the crystal manufacturer for more details (frequency, package, accuracy...).

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	Oscillator frequency [1) 2)]			12		MHz
G_{m_OSC}	Oscillator start-up G_m	V_{33}=3.3V	5.6			mA/V
		V_{33}=2V	3.8			
$t_{SU(OSC)}$	Start-up time [3)]	without serial resistor		0.6	1	ms
CK_{ACC}	Total quartz accuracy	abs. value + temp + aging			±60	ppm
α_{OSC}	Crystal oscillator duty cycle [4)]		45	50	55	%

Figure 74. Typical Application with a Crystal



Notes:

1. Refer to crystal manufacturer characteristics.

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

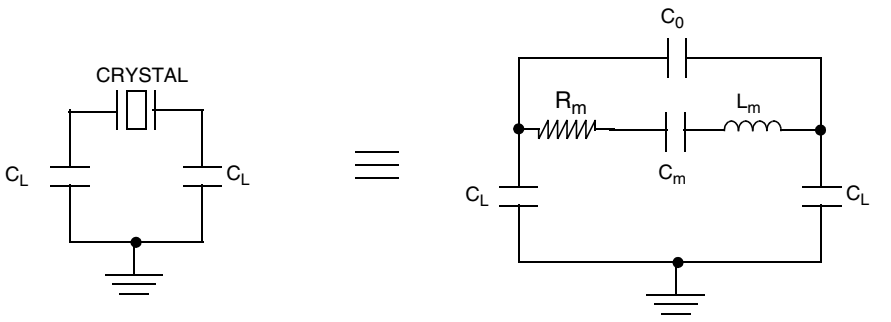
3. $t_{SU(OSC)}$ is the typical oscillator start-up time measured between V_{DD}= 4.0V and the fetch of the first instruction (with a quick V_{DD} ramp-up from 0 to V_{DD} (<50µs)).

4. The crystal oscillator duty cycle and frequency have to be adjusted trough the C_L capacitances. Refer to crystal manufacturer for more details.

5. A serial resistor R_s of 100Ω must be placed to guarantee a correct oscillator operation whatever the crystal power dissipation.

CLOCK CHARACTERISTICS (Cont'd)

Figure 75. Determining load capacitance value



$G_{m\_critic} = R_m \times (2 \times \Pi \times f_{OSC})^2 \times (2 \times C_0 + C_L)^2$

$G_{m\_critic} < (4 \times G_{m\_OSC})$

With :

R_m: crystal motional resistance

C_m : crystal motional capacitance

L_m : crystal motional inductance

C_L : load capacitance

C_0: shunt capacitance

11.5.4 6-MHz RC Oscillator

The Device internal clock can be supplied with a 6-MHz RC oscillator.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{RC}	Oscillator frequency before calibration[1)]		3.6	6	8.4	MHz
	Oscillator frequency after calibration	At a specific temperature and V_{18_DIG} power supply voltage.	5.4	6	6.6	MHz
		Including ±10% V_{18_DIG} power supply voltage variation only[1)]			±11	%
		Including ±20°C temperature variation only[1)]			±2	
		Including V_{18_DIG} power supply voltage and temperature variation over the product range[1)]			±23	
$t_{SU(RC)}$	Start-up time [2)]			4	10	µs

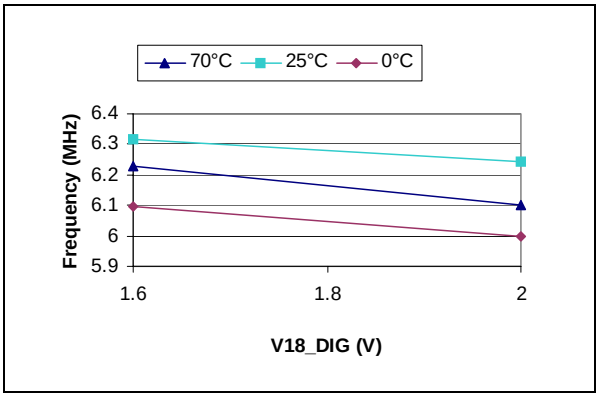
Notes:

1. Data based on characterization results.

2. $t_{SU(RC)}$ is the typical RC oscillator start-up time measured between V_{DD}= 4.0V and the fetch of the first instruction (with a quick V_{DD} ramp-up from 0 to V_{DD} (<50µs)).

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

Figure 76. Typical 6-MHz RC oscillator frequency



11.5.5 Auto Wake-up RC Oscillator

The Device internal clock can be supplied with an auto wake-up RC oscillator.

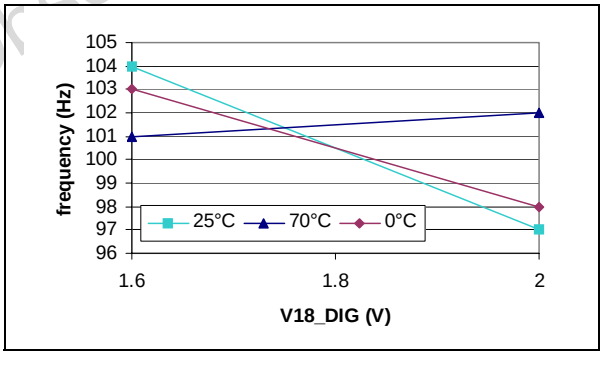
Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{AWURC}	Oscillator frequency before calibration[1)]		60	100	140	Hz
	Oscillator frequency after calibration	At a specific temperature and V_{18_DIG} power supply voltage.	90	100	110	Hz
		Including ±10% V_{18_DIG} power supply voltage variation only[1)]			±1	%
		Including ±20°C temperature variation only[1)]			±8	
		Including V_{18_DIG} power supply voltage and temperature variation over the product range[1)]			±19	
$t_{SU(AWURC)}$	Start-up time [2)]			4	10	µs

Notes:

1. Data based on characterization results.

2. $t_{SU(AWURC)}$ is the typical oscillator start-up time measured between V_{DD}= 4.0V and the fetch of the first instruction (with a quick V_{DD} ramp-up from 0 to V_{DD} (<50µs)).

Figure 77. Typical AWU RC oscillator frequency



11.6 MEMORY CHARACTERISTICS

Subject to general operating conditions for V_{DD}, f_{OSC}, 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)	0.8			V

Note:

1. Minimum V_{18_DIG} supply voltage without losing data stored in RAM (in HALT mode or under RESET) or in hardware registers (only in HALT mode). Guaranteed by construction, not tested in production.

11.7 EMC CHARACTERISTICS

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

11.7.1 Functional EMS (Electro Magnetic Susceptibility)

Based on a simple running application on the Device (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 Device 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}=12MHz, conforms to IEC 1000-4-2	3B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100pF on V_{DD} and V_{SS} pins to induce a functional disturbance	V_{DD}=5V, T_A=+25°C, f_{OSC}=12 MHz, conforms to IEC 1000-4-4	3B

EMC CHARACTERISTICS (Cont'd)

11.7.2 Electro Magnetic Interference (EMI)

Based on a simple application running on the Device (toggling 2 LEDs through the I/O ports), the Device 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
				12/12MHz	
S_{EMI}	Peak level	V_{DD}=5V, T_A=+25°C, QFN48 package conforming to SAE J 1752/3	0.1MHz to 30MHz	17	dBµV
			30MHz to 130MHz	25	
			130MHz to 1GHz	20	
			SAE EMI Level	3.5	-

Notes:

1. Data based on characterization results, not tested in production.
2. Refer to Application Note AN1709 for data on other package types.

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 Device 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). Two models can be simulated: Human Body Model and Machine Model. This test conforms to the JESD22-A114A/A115A 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
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine Model)	T_A=+25°C	200	
$V_{ESD(CDM)}$	Electro-static discharge voltage (charged device model)	T_A=+25°C	750	

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 Devices 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 Device and the Device 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 T_A=+50°C	A A
DLU	Dynamic latch-up class	V_{DD}=5V, f_{OSC}=12MHz, 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_{33}, f_{OSC}, and T_A unless otherwise specified.

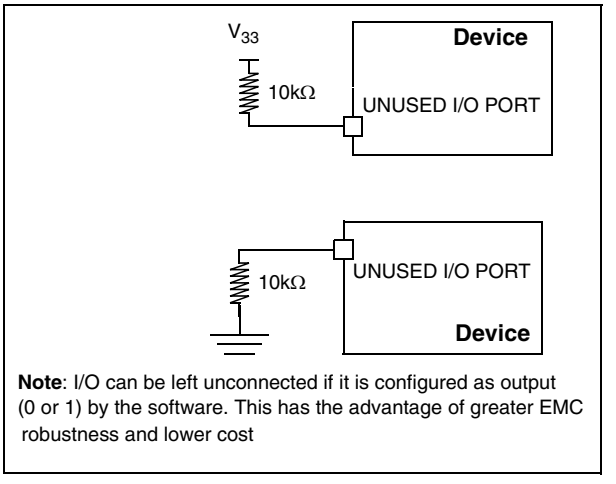
Symbol	Parameter	Conditions		Min	Typ [1]	Max	Unit
V_{IL}	Input low level voltage [2]	Port A[0:6], Port G				$0.2xV_{33}$	V
		Port B, PA7				$0.3xV_{33}$	
		CMOS ports				$0.3xV_{33}$	
V_{IH}	Input high level voltage [2]	CMOS ports		$0.7xV_{33}$			
V_{hys}	Schmitt trigger voltage hysteresis [3]				250		mV
I_L	Input leakage current	$V_{SS} \leq V_{IN} \leq V_{33}$				±15	µA
I_S	Static current consumption induced by each floating input pin[4]	Floating input mode				200	
R_{PU}	3.3V Weak pull-up equivalent resistor [5] (see Figure 80)	$V_{IN}=V_{SS}$	V_{33}=3.3V	33	60	110	kΩ
R_{PU_PS2}	5V pull-up equivalent resistor [8]	$V_{IN}=V_{SS}$	V_{DD}=5V	3.5	4.7	6.3	kΩ
R_{PU_USB}	Equivalent serial resistor	I=-250µA	V_{33}=3.3V		40		Ω
C_{IO}	I/O pin capacitance				5		pF
$t_{f(IO)out}$	Output high to low level fall time [6]	C_L=50pF Between 10% and 90%			10	16	ns
$t_{r(IO)out}$	Output low to high level rise time [6]				10	16	
$t_{w(IT)in}$	External interrupt pulse time [7]			1			t_{CPU}

Notes:

1. Unless otherwise specified, typical data are based on T_A=25°C and V_{DD}=5V.
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. 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 78). Static peak current value taken at a fixed V_{IN} value, based on design simulation and technology characteristics, not tested in production. This value depends on V_{DD} and temperature values.
5. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics). This data is based on characterization results, tested in production at V_{33} max.
6. Data based on characterization results, not tested in production.
7. 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.
8. $Rpu_{\_PS2}$ pull-up are only available on PA5, PA6, PG0 and PG1.

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 78. Two typical Applications with unused I/O Pin



Note: I/O can be left unconnected if it is configured as output (0 or 1) by the software. This has the advantage of greater EMC robustness and lower cost

Figure 79. Typical I_{PU} vs. V_{33} with $V_{IN}=V_{SS}$

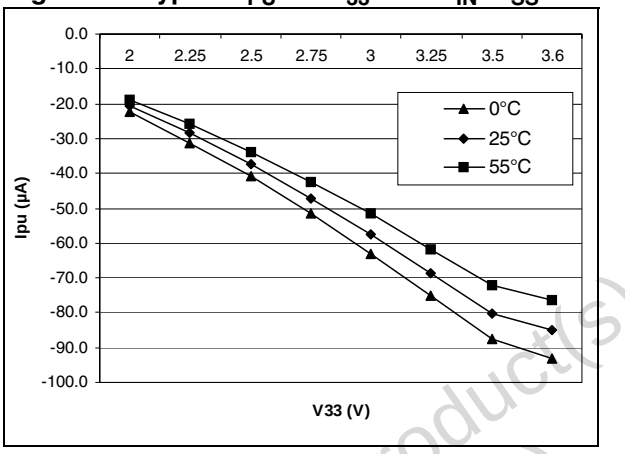
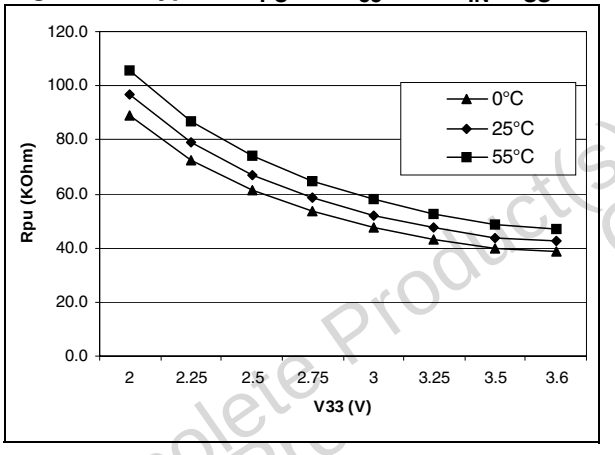


Figure 80. Typical R_{PU} vs. V_{33} with $V_{IN}=V_{SS}$



I/O PORT PIN CHARACTERISTICS (Cont'd)

11.8.2 Output Driving Current

Subject to general operating conditions for V_{33}, f_{OSC}, and T_A unless otherwise specified.

Symbol	Parameter	Conditions		Min	Max	Unit
V_{OL} 1)	Output low level voltage for a standard I/O pin when 8 pins are sunk at same time (see Figure 82 and Figure 83)	V_{33}=3.3V	I_{IO}=+5mA		0.6	V
			I_{IO}=+2mA		0.35	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see Figure 81)		I_{IO}=+20mA		1.2	
			I_{IO}=+8mA		0.75	
V_{OH} 2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 83)		I_{IO}=-5mA	2.6		
			I_{IO}=-2mA	2.8		
	Output high level voltage for PA7		I_{IO}=-5mA	2.4		
			I_{IO}=-2mA	2.6		

Figure 81. Typical V_{OL} vs. V_{33} (high-sink I/Os)

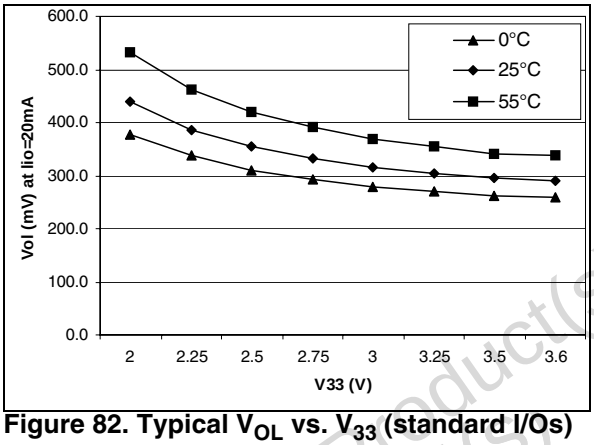
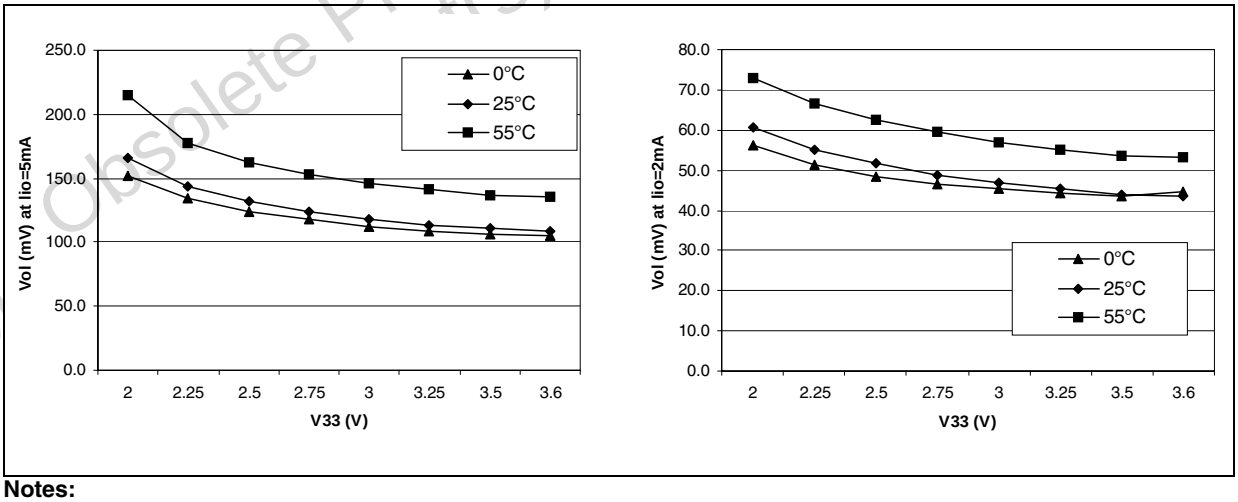


Figure 82. Typical V_{OL} vs. V_{33} (standard I/Os)

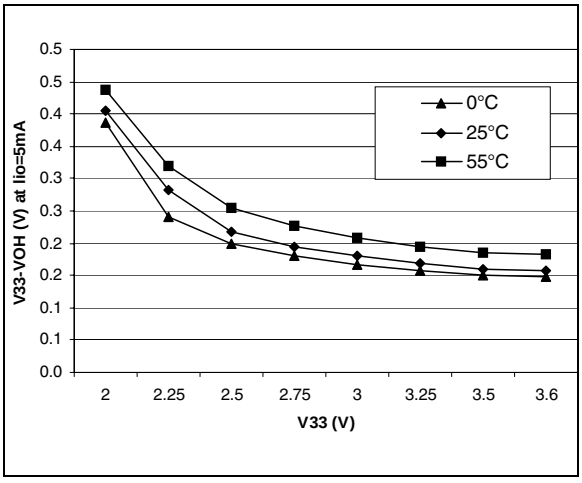


Notes:

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 11.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
2. The I_{IO} current sourced must always respect the absolute maximum rating specified in Section 11.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{V33}. True open drain I/O pins does not have V_{OH}.

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 83. Typical V_{33}-V_{OH} vs. V_{33} (standard I/Os)



11.9 CONTROL PIN CHARACTERISTICS

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

T_A = 0 to +55°C unless otherwise specified

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage[7]					0.16xV_{33}	V
V_{IH}	Input high level voltage			0.85xV_{33}			
V_{hys}	Schmitt trigger voltage hysteresis[1]				250		V
V_{OL}	Output low level voltage [2] [6]	V_{33}=3.3V	I_{IO}=+5mA			0.6	V
			I_{IO}=+2mA			0.35	
R_{ON}	Pull-up equivalent resistor [1][3][8]	V_{33}=3.3V		33	51	110	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Internal reset sources			30 + 256*f_{OSC}		µs
$t_{h(RSTL)in}$	External reset pulse hold time [4]			2.5			µs
$t_{g(RSTL)in}$	Filtered glitch duration [5]				200		ns

Notes:

1. Data based on characterization results, not tested in production.
2. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 11.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
3. The R_{ON} pull-up resistor is based on a true resistor.
4. To guarantee the reset of the Device, a minimum pulse has to be applied to the $\overline{\text{RESET}}$ pin. All short pulses applied on $\overline{\text{RESET}}$ pin with a duration below $t_{h(RSTL)in}$ can be ignored.
5. The reset network protects the device against parasitic resets.
6. The output of the external reset circuit must have an open-drain output to drive the Device reset pad. Otherwise the Device can be damaged when the CPU generates an internal reset.
7. Whatever the reset source is (internal or external), the user must ensure that the level on the $\overline{\text{RESET}}$ pin can go below the V_{IL} max. level specified in Section 11.9.1. Otherwise the reset will not be taken into account internally.
8. Because the reset circuit is designed to allow the internal RESET to be output in the $\overline{\text{RESET}}$ pin, the user must ensure that the current sunk on the $\overline{\text{RESET}}$ pin (by an external pull-up for example) is less than the absolute maximum value specified in Section 11.2.2 on page 144.

Figure 84. Typical I_{PU} on $\overline{\text{RESET}}$ pin

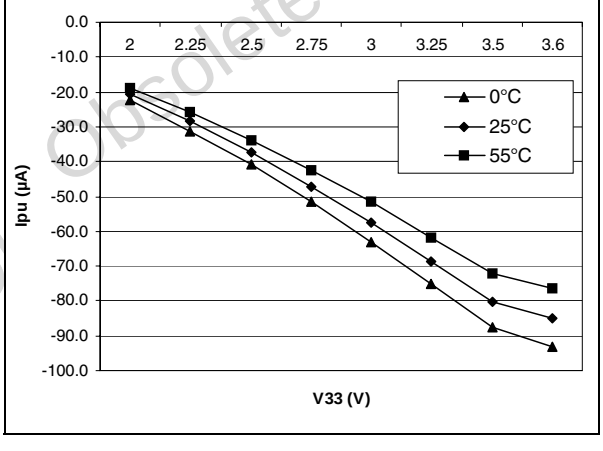
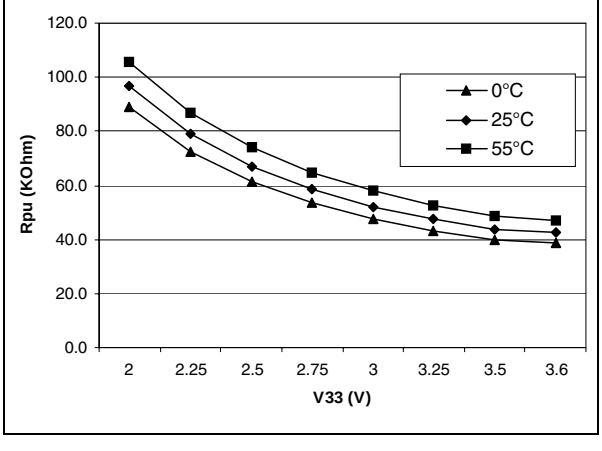


Figure 85. Typical R_{PU} on $\overline{\text{RESET}}$ pin



11.10 TIMER PERIPHERAL CHARACTERISTICS

Subject to general operating conditions for V_{33}, f_{OSC}, and T_A unless otherwise specified.

Refer to I/O port characteristics for more details on the input/output alternate function characteristics (output compare, input capture, external clock, PWM output...).

11.10.1 Watchdog Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{w(WDG)}$	Watchdog time-out duration		65 536		4 194 304	t_{CPU}
		f_{CPU} = 12MHz	5.461		349.525	ms

11.10.2 Time Base Unit Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{w(TBU)}$	TBU time-out duration	Standalone mode	2		65536	t_{CPU}
		Standalone mode f_{CPU} = 12MHz	0.166		5.461	ms

11.10.3 16-Bit Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{w(ICAP)in}$	Input capture pulse time		1			t_{CPU}
$t_{res(PWM)}$	PWM resolution time		2			t_{CPU}
		f_{CPU} = 12MHz	166.7			ns
		f_{CPU} = 6MHz	333.3			
f_{EXT}	Timer external clock frequency		0		$f_{CPU}/4$	MHz
f_{PWM}	PWM repetition rate		0		$f_{CPU}/4$	MHz
Res_{PWM}	PWM resolution				16	bit

11.10.4 Auto Wake-up from Halt Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{w(AWU)}$	AWUFH time-out duration		1		255	t_{AWU}
		f_{AWU} = 100Hz	10		2 550	ms

11.11 RF INTERFACE Electrical Characteristics

11.11.1 Radio Electrical Characteristics

11.11.1.1 Characterization Conditions

In order to reach all the electrical characteristics mentioned in Table 40 , the Device environment must comply with the conditions mentioned in the Table 39.

Table 39. Receiver Characterization Conditions

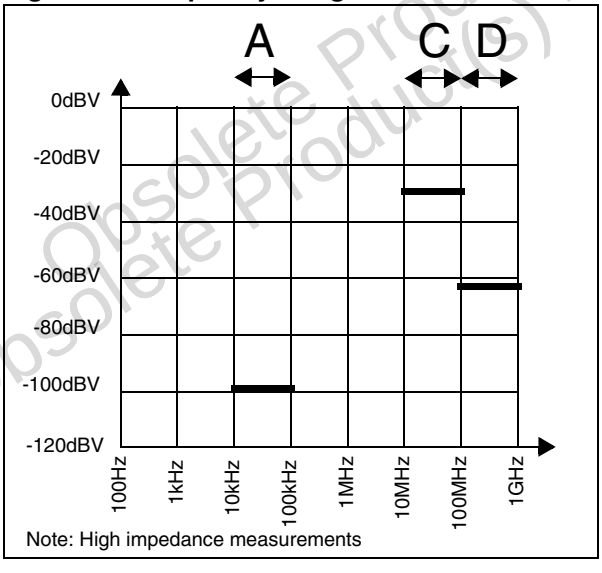
Parameter	Name	Conditions	Min	Typ	Max	Unit
Maximum Power Supply Noise Level	PSN_A	In frequency range “A” at 50kHz in +/-30kHz bandwidth referred to VSS_OUT_PLL*n*		50		μV_{RMS}
Maximum CPOUT Noise Level	$CPOUTN_A$	In frequency range “A” at 50kHz in +/-30kHz bandwidth referred to V18_OUT_PLL*n*		50		μV_{RMS}
Maximum Power Supply Noise Level	PSN_C	In frequency range “C” at 27MHz in +/-100kHz bandwidth referred to VSS_LNA		30		mV_{RMS}
Maximum Power Supply Noise Level	PSN_D	In frequency range “D” at 216MHz in +/-100kHz bandwidth referred to V18_OUT_PLL*n*		600		μV_{RMS}

The power supplies has to be decoupled in order to decrease noise amplitude on power supplies below the values provide by Table 39 and also in Figure 86.

These noise sources are the Device itself and also external noise conducted through USB / PS/2 plug or power supplies.

Figure 86 describes the frequency range definition, and shows the expected noise level on these frequency ranges during receiving.

Figure 86. Frequency Range Definition



RF INTERFACE Electrical Characteristics (Cont'd)

11.11.1.2 Characterization Information

Figure 87 shows the test board schematic used to measure the sensitivity. This uses a single ended to differential balun matched at 200Ω.

Figure 88 shows the test board schematic used with balun matched at 3.3KΩ.

Figure 87. 200Ω Input circuit for Sensitivity Measurements (wide band balun)

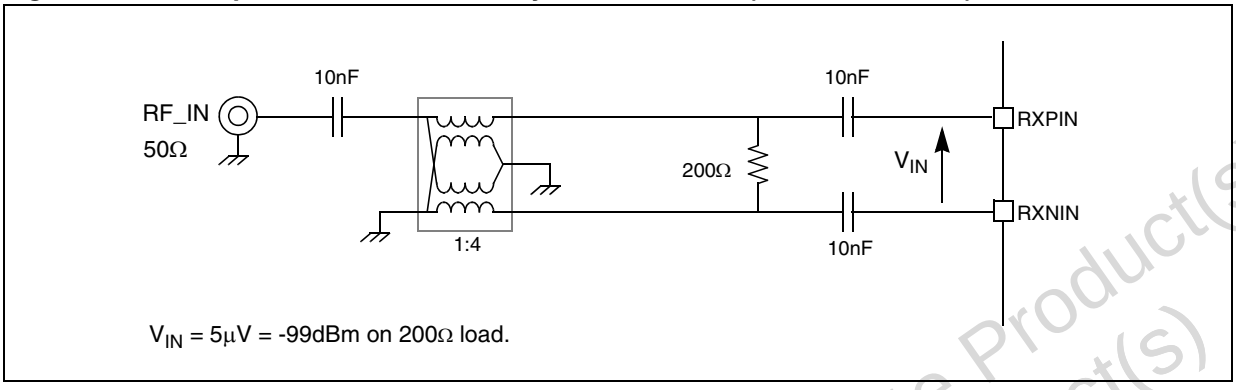
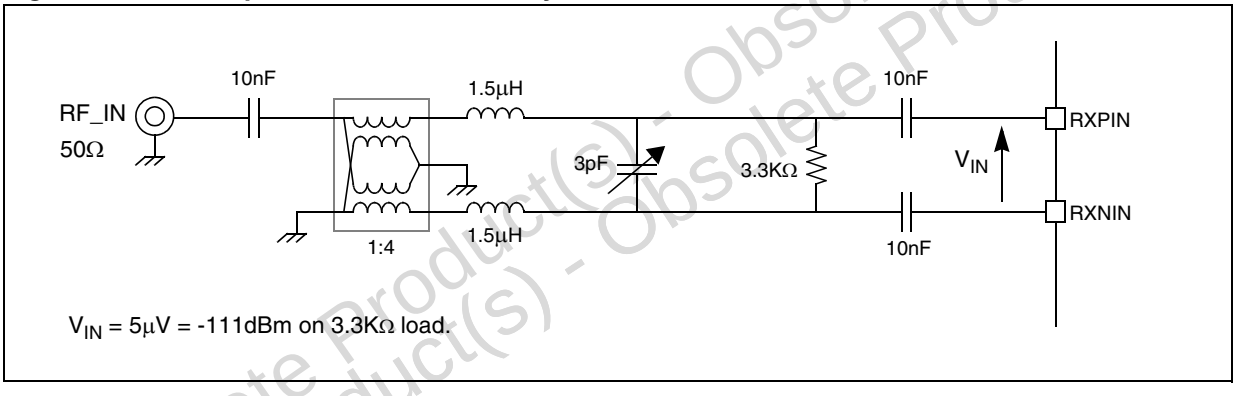


Figure 88. 3.3KΩ Input circuit for Sensitivity Measurements



RF INTERFACE Electrical Characteristics (Cont'd)

Table 40. Receiver Radio Characteristics

Unless otherwise specified, typical data are based on T_A=25°C and V_{18}=1.8V

Parameter	Name	Test Conditions	Min	Typ	Max	Unit
DEMODULATOR						
Channel Frequency Range	FR_{RANG}	Channel frequency	26,905		27,305	kHz
		Local oscillator frequency	26,890 + (10×N) (0 ≤ N ≤ 43)			kHz
Admissible Frequency Deviation	F_{DEV25}	Data Rate at 2.5Kbit/s	2.4	3.2	4.0	kHz peak
	F_{DEV50}	Data Rate at 5.0Kbit/s	2.9	4.1	5.4	kHz peak
Admissible Crystal tolerance	$XTAL_{TOL}$	Receiver and Transmitter contribution	-120		120	ppm
Input Capacitance	CIN			5		pF
Input Resistance	RIN_{ATTOFF}	Attenuator off. RXPIN/RXNIN res. (process and temp variation)	40	50	60	KΩ
	RIN_{ATTON}	Attenuator on. RXPIN/RXNIN res. (process and temp variation)	180	210	230	Ω
Maximum Input Signal	VMX_{ATTON}	Attenuator Enable (with 3.3KΩ antenna impedance)		500		mV
	VMX_{ATTOFF}	Attenuator Disable		500		mV
Compression Level	$CP1_{ATTON}$	Attenuator Enable (with 3.3KΩ antenna impedance)		160		mV
	$CP1_{ATTOFF}$	Attenuator Disable		100		mV
Sensitivity[1)]	IN_{SENS25}	2.5 Kbit/s; F_{dev}=2.5kHz		5	7	μV_{RMS}
	IN_{SENS50}	5.0 Kbit/s; F_{dev}=5kHz		6	9	μV_{RMS}
Adjacent Channel Rejection	ACR_{50}	50kHz freq. offset (see Figure 51)	33	36		dB
	ACR_{100}	100kHz freq. offset (see Figure 51)	39	42		dB
Dual Interferers Level	$DUAL_{INT}$	More than 200kHz frequency offset (see Figure 53)	-45			dBm
Image Suppression	RX_IM_{SUP}	At +/-30kHz frequency offset	27	30		dBc
Co-Channel Rejection	CO_{REJ}	Data Rate at 2.5Kbit/s	-11	-8		dBc
		Data Rate at 5.0Kbit/s	-11	-8		dBc
Analog Attenuator	ATT_{ANARF}	With 3.3KΩ antenna impedance	8	10	12	dB
Set-up Time	$WUdel_{RX}$	from RF Power Down mode		1	3	ms

Note:

1. Sensitivity is characterized for BER=10^{-4}, using NRZ coding with 3.3KΩ input circuit.

11.12 OTHER COMMUNICATION INTERFACE CHARACTERISTICS

11.12.1 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		V
Differential Common Mode Range	VCM	Includes VDI range	0.8	2.5	V
Single Ended Receiver Threshold	VSE		0.8	2.0	V
Static Output Low	VOL	RL [1] of 1.5K ohms to 3.6v		0.3	V
Static Output High	VOH	RL [1] of 15K ohms to V_{SS}	2.8	3.6	V
USBVCC: voltage level[3]	USBV	V_{DD}=5v	3.00	3.60	V

Figure 89. USB: Data Signal Rise and Fall Time

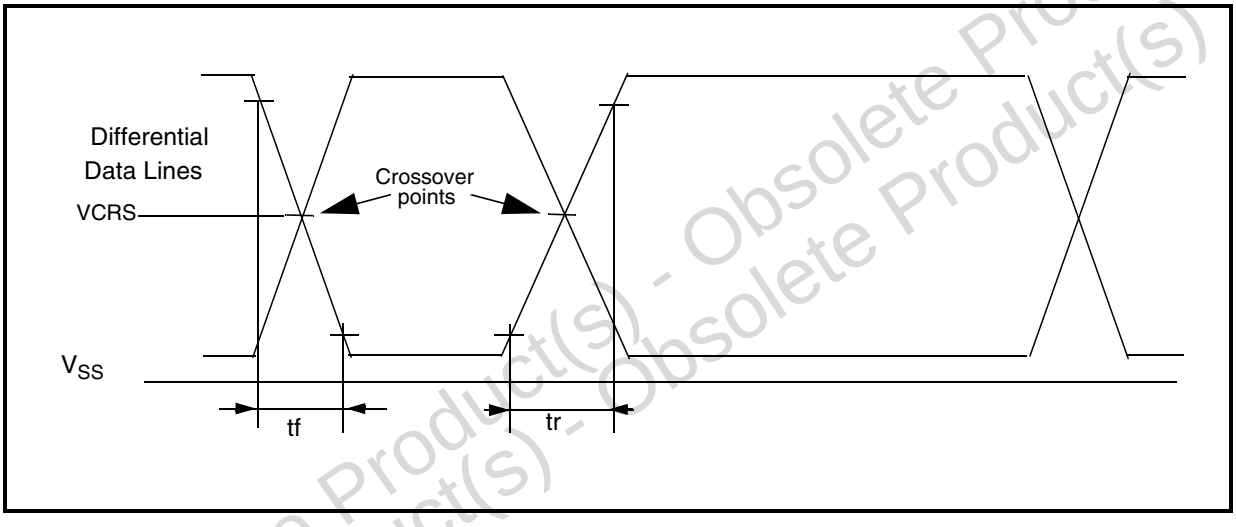


Table 41. USB: Low-speed Driver Electrical Characteristics

Parameter	Symbol	Conditions	Min	Max	Unit
Rise time [4]	tr	CL=50 pF	75		ns
		CL=600 pF		300	
Fall Time [4]	tf	CL=50 pF	75		
		CL=600 pF		300	
Rise/ Fall Time matching	trfm	tr/tf	80	120	%
Output signal Crossover Voltage	VCRS		1.3	2.0	V

Notes:

1. RL is the load connected on the USB drivers.
2. All the voltages are measured from the local ground potential.
3. To improve EMC performance (noise immunity), it is recommended to connect a 100nF capacitor to the USBVCC pin.
4. Measured from 10% to 90% of the data signal. For more detailed information, please refer to Chapter 7 (Electrical) of the USB specification (version 1.1).

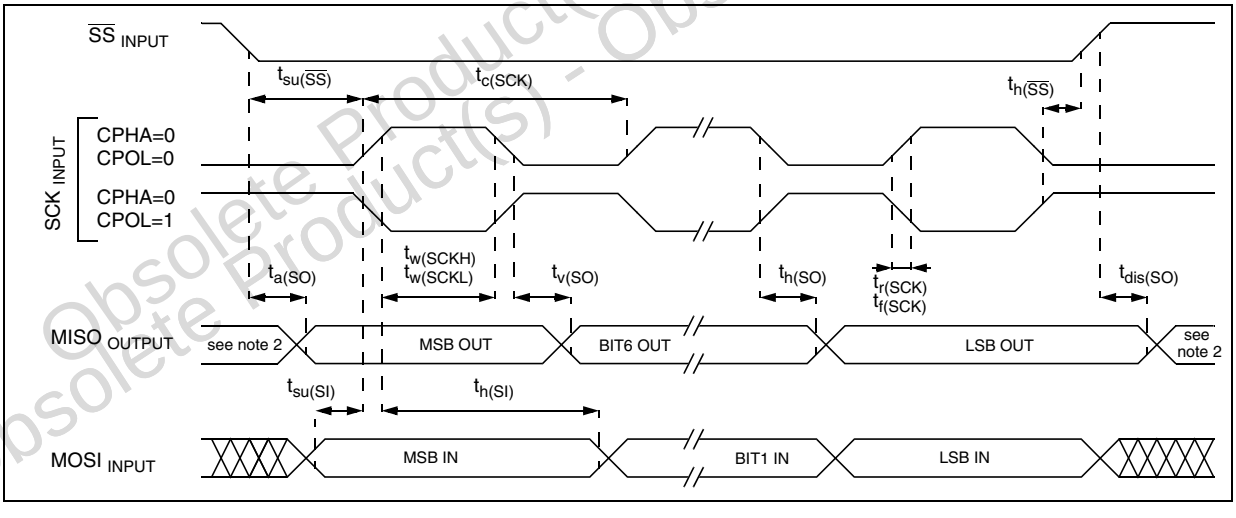
11.12.2 SPI - Serial Peripheral Interface

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

Refer to I/O port characteristics for more details on the input/output alternate function characteristics ($\overline{SS}$, SCK, MOSI, MISO).

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCK} $1/t_{c(SCK)}$	SPI clock frequency	Master f_{CPU}=12MHz	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave f_{CPU}=12MHz	0	$f_{CPU}/2$ 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see I/O port pin description		
$t_{su(\overline{SS})}$ 1)	$\overline{SS}$ setup time 4)	Slave	(4 x T_{CPU}) + 50		ns
$t_{h(\overline{SS})}$ 1)	$\overline{SS}$ hold time	Slave	120		
$t_{w(SCKH)}$ 1) $t_{w(SCKL)}$ 1)	SCK high and low time	Master Slave	100 90		
$t_{su(MI)}$ 1) $t_{su(SI)}$ 1)	Data input setup time	Master Slave	100 100		
$t_{h(MI)}$ 1) $t_{h(SI)}$ 1)	Data input hold time	Master Slave	100 100		
$t_{a(SO)}$ 1)	Data output access time	Slave	0	120	
$t_{dis(SO)}$ 1)	Data output disable time	Slave		240	
$t_{v(SO)}$ 1)	Data output valid time	Slave (after enable edge)		120	
$t_{h(SO)}$ 1)	Data output hold time		0		
$t_{v(MO)}$ 1)	Data output valid time	Master (after enable edge)		120	
$t_{h(MO)}$ 1)	Data output hold time		0		

Figure 90. SPI Slave Timing Diagram with CPHA=0 3)



Notes:

1. Data based on design simulation and/or characterisation results, not tested in production.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.
3. Measurement points are done at CMOS levels: $0.3xV_{33}$ and $0.7xV_{33}$.
4. Depends on f_{CPU}. For example, if f_{CPU}=8MHz, then T_{CPU} = $1/f_{CPU}$ =125ns and $t_{su(\overline{SS})}$=550ns

COMMUNICATION INTERFACE CHARACTERISTICS (Cont'd)

Figure 91. SPI Slave Timing Diagram with CPHA=1[1)]

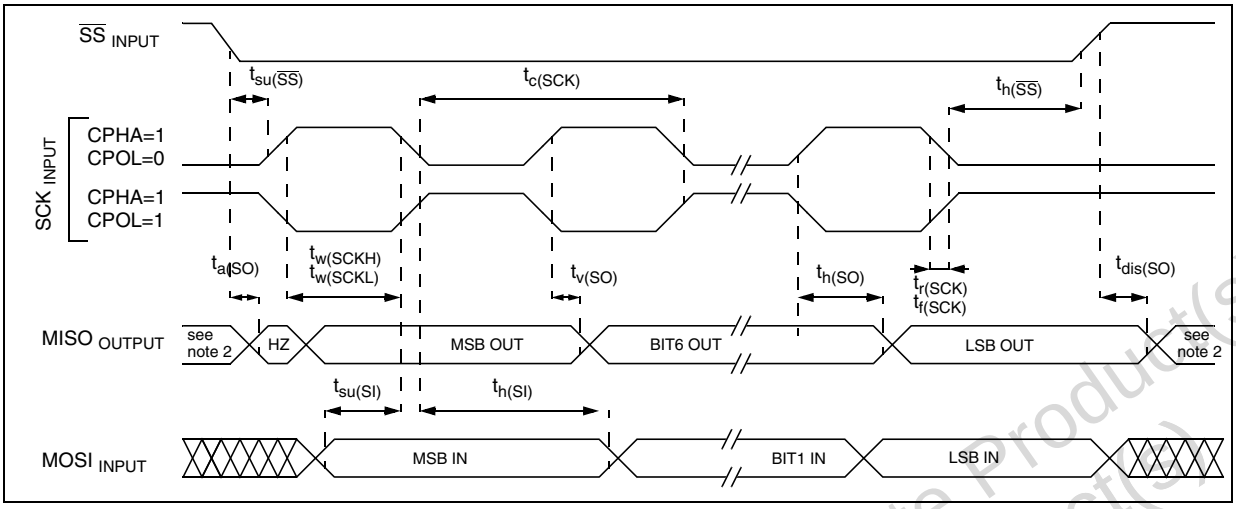
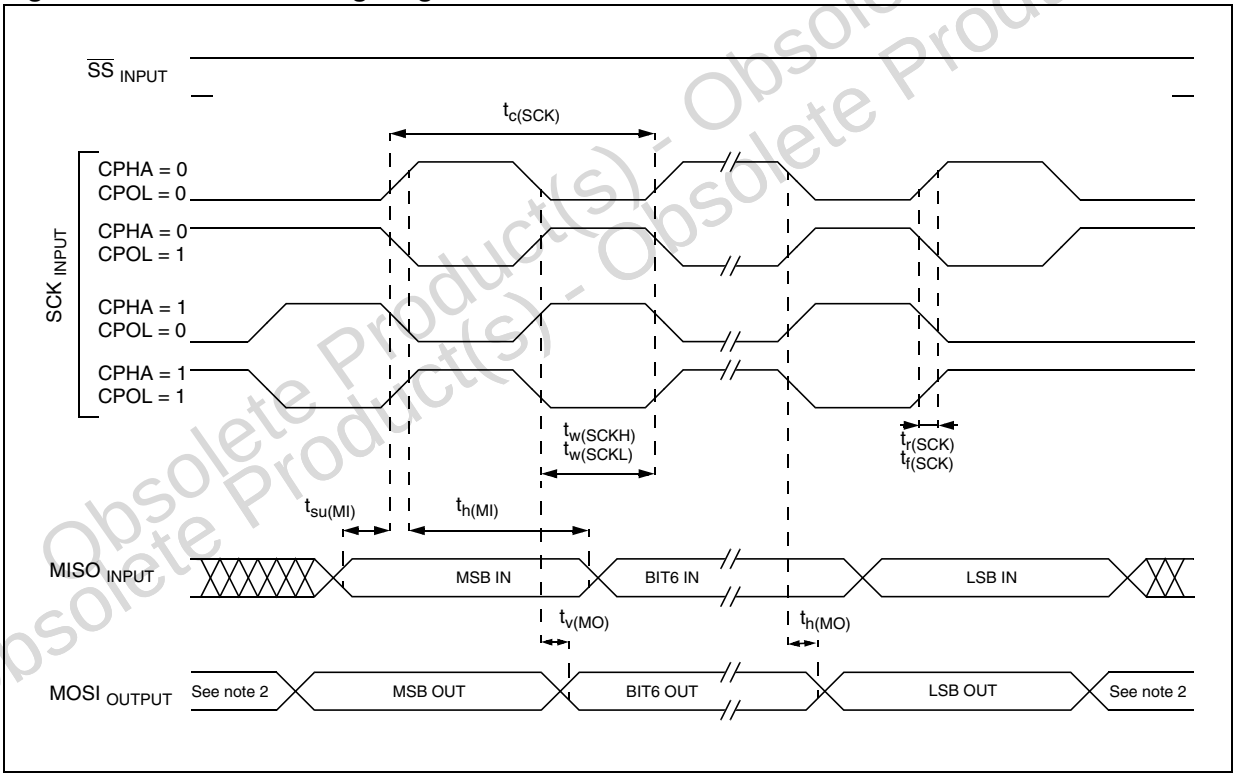


Figure 92. SPI Master Timing Diagram[1)]



Notes:

1. Measurement points are done at CMOS levels: $0.3xV_{33}$ and $0.7xV_{33}$.

2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends of the I/O port configuration.

11.13 POWER-ON-RESET AND LVD CHARACTERISTICS

Table 42. Power-on-reset characteristics

Symbol	Parameter	Conditions	Min	Typ[1)]	Max	Unit
V_{POR}	Threshold voltage		1.10	1.4	1.60	V
H_{POR}	Hysteresis			100		mV
I_{DD_POR}	Current consumption			8	10	µA
t_{POR}	POR pulse duration		200			nS
t_{glitch}	Glitch duration filtered	Triangular glitch-with 300mV height			80	nS

Table 43. Low Voltage Detector characteristics

Symbol	Parameter	Conditions	Min	Typ[1)]	Max	Unit
V_{IT+}	Reset release threshold (V_{DD} rising)		3.5		3.85	V
V_{IT-}	Reset generation threshold (V_{DD} falling)		3.4		3.75	V
V_{hys}	Hysteresis V_{IT+} - V_{IT-}			100		mV
V_{tPOR}	V_{DD} rise time rate		0.02		500	V/ms

Note:
1. Unless otherwise specified, typical data are based on T_A = 25°C and V_{33}-V_{SS} = 3.3V. They are given only as design guidelines and are not tested.

11.14 VOLTAGE REGULATORS CHARACTERISTICS

Table 44. 5V-3.3V Voltage Regulator characteristics

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
V_{IN}	Input voltage range		4.00	5.00	5.60	V
V_{OUT33}	Output voltage range		3.10	3.30	3.60	V
I_{OUT33}	Output current drive (including external current consumption)		30			mA
C_{33}	Decoupling capacitance	Multi layer / ceramic capacitor	4.7	10		µF
ESR	Equivalent series resistor		0	0.5	1	Ω
t_{ON}	Start-up delay	at I_{LOAD}=0mA & Power-down command			400	µs
		at I_{LOAD}=30mA & Power-down command			600	µs
I_{ON}	Current consumption				30	µA
I_{OFF}	I_{DD} current				1	µA

Table 45. 3.3V-1.8V Voltage Regulator characteristics [2]

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
V_{IN}	Input voltage range		3.10	3.30	3.60	V
V_{OUT18}	Output voltage range		1.65	1.80	1.95	V
I_{OUT18}	Output current drive	Suspend mode			250	µA
		Normal mode	15			mA
C_{18}	Decoupling capacitance	Multi layer / ceramic capacitor	100			nF
ESR	Equivalent series resistor		0		4	Ω
t_{ON}	Start-up delay	without output consumption			75	µs
I_{ON}	Current consumption	Suspend mode			75	µA
		Normal mode			300	
I_{OFF}	I_{DD} current				1	µA

Table 46. 1.8V Voltage Reference characteristics

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
I_{ON}	Current consumption		30	35	50	µA
t_{ON}	Start-up delay		10	25	40	µs
I_{OFF}	I_{DD} current				1	µA

Note:
1. Unless otherwise specified, typical data are based on T_A = 25°C and V_{33}-V_{SS} = 3.3V. They are given only as design guidelines and are not tested.
2. This voltage regulator is for internal supply only. It cannot be used to power external components.

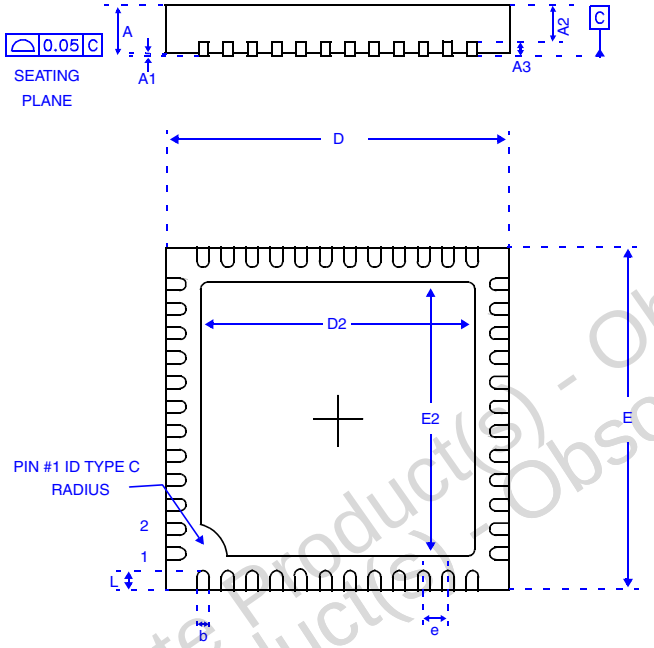
12 PACKAGE CHARACTERISTICS

In order to meet environmental requirements, ST offers these devices in ECOPACK® packages. These packages have a Lead-free second level interconnect. The category of second Level Interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label.

ECOPACK is an ST trademark. ECOPACK specifications are available at: www.st.com.

12.1 PACKAGE MECHANICAL DATA

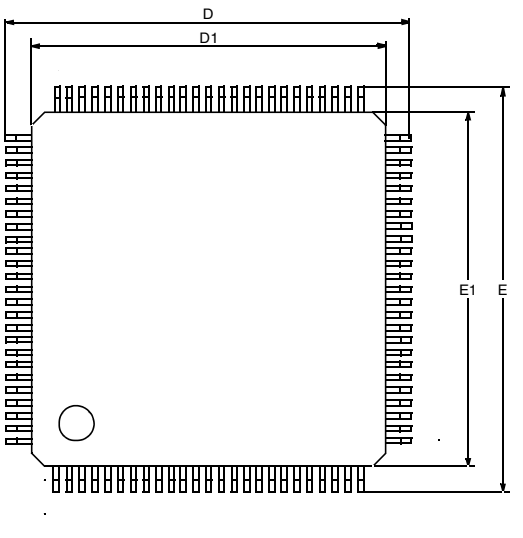
Figure 93. 48-Lead Very thin Fine pitch Quad Flat No-Lead Package



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			0.001	
A2		0.65			0.026	
A3		0.20			0.008	
b	0.18	0.25	0.30	0.007	0.010	0.012
D	6.85	7.00	7.15	0.270	0.276	0.281
D2	5.00	5.15	5.30	0.197	0.203	0.209
E	6.85	7.00	7.15	0.270	0.276	0.281
E2	5.00	5.15	5.30	0.197	0.203	0.209
e		0.50			0.020	
L	0.30	0.40	0.50	0.012	0.016	0.020
	Number of Pins					
N	48					

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

Figure 94. 100-Pin Low Profile Quad Flat Package



Dim.	mm			inches		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.063
A1	0.05		0.15	0.002		0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.17	0.22	0.27	0.007	0.009	0.011
C	0.09		0.20	0.004		0.008
D		16.00			0.630	
D1		14.00			0.551	
E		16.00			0.630	
E1		14.00			0.551	
e		0.50			0.020	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
	Number of Pins					
N	100					

12.2 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient) LQFP100	44	°C/W
	Package thermal resistance (junction to ambient) QFN48	29	
T_{Jmax}	Maximum junction temperature [1]	150	°C
P_{Dmax}	Power dissipation [2]	500	mW

Notes:

1. The maximum chip-junction temperature is based on technology characteristics.

2. The maximum power dissipation is obtained from the formula $P_D = (T_J - T_A) / R_{thJA}$.

The power dissipation of an application can be defined by the user with the formula: $P_D=P_{INT}+P_{PORT}$ where P_{INT} is the chip internal power ($I_{DD}xV_{DD}$) and P_{PORT} is the port power dissipation depending on the ports used in the application.

12.3 SOLDERING INFORMATION

In accordance with the RoHS European directive, all STMicroelectronics packages have been converted to lead-free technology, named ECOPACK™.

- ECOPACK™ packages are qualified according to the JEDEC STD-020C compliant soldering profile.
- Detailed information on the STMicroelectronics ECOPACK™ transition program is available on www.st.com/stonline/leadfree/, with specific technical Application notes covering the main technical aspects related to lead-free conversion (AN2033, AN2034, AN2035, AN2036).

Backward and forward compatibility:

The main difference between Pb and Pb-free soldering process is the temperature range.

– ECOPACK™ LQFP, SDIP, SO and QFN packages are fully compatible with Lead (Pb) containing soldering process (see application note AN2034)

– LQFP, SDIP and SO Pb-packages are compatible with Lead-free soldering process, nevertheless it's the customer's duty to verify that the Pb-packages maximum temperature (mentioned on the Inner box label) is compatible with their Lead-free soldering temperature.

Table 47. Soldering Compatibility (wave and reflow soldering process)

Package	Plating material devices	Pb solder paste	Pb-free solder paste
QFN	Sn (pure Tin)	Yes	Yes *
LQFP and SO	NiPdAu (Nickel-palladium-Gold)	Yes	Yes *

* Assemblers must verify that the Pb-package maximum temperature (mentioned on the Inner box label) is compatible with their Lead-free soldering process.

13 DEVICE CONFIGURATION

13.1 OPTION BYTE

The option byte allows the hardware configuration of the microcontroller to be selected. In masked ROM devices, the option bytes are fixed in hardware by the ROM code (see option list).

WDGHWR *Hardware Watchdog Reset*
This option permanently enables the watchdog reset.

WDGHWI *Hardware Watchdog Interrupt*
This option permanently enables the watchdog Top Level Interrupt (TLI) interrupt. Refer to Section 9.1 for more information in the WDGA and IE control bits in the WDGCSR register.

WDGHWR	WDGHWI	Behaviour when a Watchdog timeout occurs
0	0	If WDGA=1, a reset is generated. If WDGA=0 and IE=1, a TLI is generated.
0	1	If WDGA=1 a reset is generated. If WDGA=0, a TLI is generated.
1	x	A reset is generated.

Note: 0= option disabled, 1= option enabled

If both are enabled the Reset event has priority.

FMP_R *Full Memory Readout Protection*
This option protects the ROM program memory against readout. This protects the application-firmware against software piracy.

– Readout enabled

– Readout disabled

13.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

Customer code is made up of the ROM contents and the list of the selected options (if any). The ROM contents are to be sent on diskette, or by electronic means, with the S19 hexadecimal file generated by the development tool. All unused bytes must be set to FFh.

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

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

ST7WInD21 MICROCONTROLLER OPTION LIST
(last update: January 2007)

Customer . . . . . .
Address . . . . . .
. . . . . .
Contact . . . . . .
Phone No . . . . . .
Reference . . . . . .

Conditioning (check only one option):

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

Hardware Watchdog Reset: [ ] Enabled [ ] Disabled

Hardware Watchdog Interrupt [1] : [ ] Enabled [ ] Disabled

Full Memory Readout Protection: [ ] Enabled [ ] Disabled

Comments : . . . . . .
. . . . . .
. . . . . .
Supply Operating Range in the application: . . . . . .
Notes . . . . . .
. . . . . .
. . . . . .
Signature . . . . . .
Date . . . . . .

Note 1: This selection has no effect when the Watchdog Reset option has been enabled.

13.3 DEVELOPMENT TOOLS

Development tools for the ST7 microcontrollers include a complete range of hardware systems and software tools from STMicroelectronics and third-party tool suppliers. The range of tools includes solutions to help you evaluate microcontroller peripherals, develop and debug your application, and program your microcontrollers.

13.3.1 Evaluation boards

ST offers complete, affordable **evaluation kits**. Evaluation boards are complete, affordable hardware/software tool packages that include features and samples to help you quickly start developing your application.

13.3.2 Development and debugging tools

Application development for ST7 is supported by fully optimizing **C Compilers** and the **ST7 Assembler-Linker** toolchain, which are all seamlessly integrated in the ST7 integrated development environments in order to facilitate the debugging and fine-tuning of your application. The Cosmic C Compiler is available in a free version that outputs up to 16KBytes of code.

The hardware tool includes full-featured **ST7-EMU3 emulator**. This tool is supported by the **ST7 Toolset** from STMicroelectronics, which includes the STVD7 integrated development environment (IDE) with high-level language debugger, editor, project manager and integrated programming interface.

13.3.3 Order codes for development and evaluation tools

Table 48 below lists the ordering codes for the ST7WinD21 development and evaluation tools. For additional ordering codes for spare parts and accessories, refer to the online product selector at www.st.com/mcu.

Table 48. STMicroelectronics Development Tools

Supported Product	Emulator	Evaluation tool
ST7WInD21	ST7WInD-EMU3	ST7WInD-EVAL

13.4 ST7 APPLICATION NOTES

Table 49. 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
AN1812	A HIGH PRECISION, LOW COST, SINGLE SUPPLY ADC FOR POSITIVE AND NEGATIVE INPUT VOLTAGES
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

Table 49. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
AN1947	ST7MC PMAC SINE WAVE MOTOR CONTROL SOFTWARE LIBRARY
GENERAL PURPOSE	
AN1476	LOW COST POWER SUPPLY FOR HOME APPLIANCES
AN1526	ST7FLITE0 QUICK REFERENCE NOTE
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 RESONATOR
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
AN1946	SENSORLESS BLDC MOTOR CONTROL AND BEMF SAMPLING METHODS WITH ST7MC
AN1971	ST7LITE0 MICROCONTROLLED BALLAST
PROGRAMMING AND TOOLS	
AN 978	ST7 VISUAL DEVELOP SOFTWARE KEY DEBUGGING FEATURES
AN 983	KEY FEATURES OF THE COSMIC ST7 C-COMPILER PACKAGE
AN 985	EXECUTING CODE IN ST7 RAM
AN 986	USING THE INDIRECT ADDRESSING MODE WITH ST7
AN 987	ST7 SERIAL TEST CONTROLLER PROGRAMMING
AN 988	STARTING WITH ST7 ASSEMBLY TOOL CHAIN
AN 989	GETTING STARTED WITH THE ST7 HIWARE C TOOLCHAIN
AN1039	ST7 MATH UTILITY ROUTINES
AN1064	WRITING OPTIMIZED HIWARE C LANGUAGE FOR ST7

Table 49. ST7 Application Notes

IDENTIFICATION	DESCRIPTION
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
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
AN1900	HARDWARE IMPLEMENTATION FOR ST7DALI-EVAL
AN1904	ST7MC THREE-PHASE AC INDUCTION MOTOR CONTROL SOFTWARE LIBRARY
AN1905	ST7MC THREE-PHASE BLDC MOTOR CONTROL SOFTWARE LIBRARY
SYSTEM OPTIMIZATION	
AN1711	SOFTWARE TECHNIQUES FOR COMPENSATING ST7 ADC ERRORS
AN1827	IMPLEMENTATION OF SIGMA-DELTA ADC WITH ST7FLITE05/09
AN2009	PWM MANAGEMENT FOR 3-PHASE BLDC MOTOR DRIVES USING THE ST7FMC
AN2030	BACK EMF DETECTION DURING PWM ON TIME BY ST7MC

14 KNOWN LIMITATIONS

14.1 CLEARING ACTIVE INTERRUPTS OUTSIDE INTERRUPT ROUTINE

When an active interrupt request occurs at the same time as the related flag or interrupt mask is being cleared, the CC register may be corrupted.

Concurrent interrupt context

The symptom does not occur when the interrupts are handled normally, i.e. when:

- The interrupt request is cleared (flag reset or interrupt mask) within its own interrupt routine

- The interrupt request is cleared (flag reset or interrupt mask) within any interrupt routine

- The interrupt request is cleared (flag reset or interrupt mask) in any part of the code while this interrupt is disabled

If these conditions are not met, the symptom can be avoided by implementing the following sequence:

Perform SIM and RIM operation before and after resetting an active interrupt request

Ex:

```
SIM
reset flag or interrupt mask
RIM
```

14.2 16-bit Timer PWM Mode

In PWM mode, the first PWM pulse is missed after writing the value FFFCh in the OC1R register (OC1HR, OC1LR). It leads to either full or no PWM during a period, depending on the OLVL1 and OLVL2 settings.

15 REVISION HISTORY

Date	Revision	Main changes
06-February-2007	2	First release on intranet

Notes: