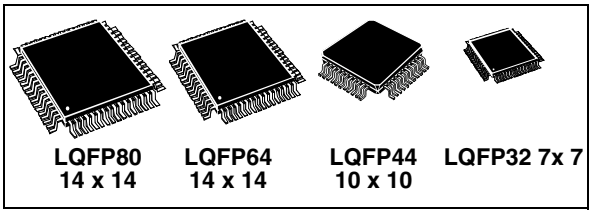
ST7MC1xx/ST7MC2xx

8-bit MCU with nested interrupts, Flash, 10-bit ADC, brushless motor control, five timers, SPI, LINSCI™

Features

- **Memories**
  - 8K to 60K dual voltage Flash Program memory or ROM with read-out protection capability, In-application programming and In-circuit programming.
  - 384 to 1.5K RAM
  - HDFlash endurance: 100 cycles, data retention: 40 years at 85°C
- **Clock, reset and supply management**
  - Enhanced reset system
  - Enhanced low voltage supervisor (LVD) for main supply and auxiliary voltage detector (AVD) with interrupt capability
  - Clock sources: crystal/ceramic resonator oscillators and by-pass for external clock, clock security system.
  - Four power saving modes: Halt, Active-halt, Wait and Slow
- **Interrupt management**
  - Nested interrupt controller
  - 14 interrupt vectors plus TRAP and RESET
  - MCES top level interrupt pin
  - 16 external interrupt lines (on 3 vectors)
- **Up to 60 I/O ports**
  - up to 60 multifunctional bidirectional I/O lines
  - up to 41 alternate function lines
  - up to 12 high sink outputs
- **5 timers**
  - Main clock controller with: Real-time base, Beep and clock-out capabilities
  - Configurable window watchdog timer
  - Two 16-bit timers with: 2 input captures, 2 output compares, external clock input, PWM and pulse generator modes
  - 8-bit PWM Auto-reload timer with: 2 input captures, 4 PWM outputs, output compare and time base interrupt, external clock with event detector



- **2 Communication interfaces**
  - SPI synchronous serial interface
  - LINSCI™ asynchronous serial interface
- **Brushless motor control peripheral**
  - 6 high sink PWM output channels for sine-wave or trapezoidal inverter control
  - Motor safety including asynchronous emergency stop and write-once registers
  - 4 analog inputs for rotor position detection (sensorless/hall/tacho/encoder)
  - Permanent magnet motor coprocessor including multiplier, programmable filters, blanking windows and event counters
  - Operational amplifier and comparator for current/voltage mode regulation and limitation
- **Analog peripheral**
  - 10-bit ADC with 16 input pins
- **In-circuit Debug**
- **Instruction set**
  - 8-bit data manipulation
  - 63 basic instructions with illegal opcode detection
  - 17 main Addressing modes
  - 8 x 8 unsigned multiply instruction
  - True bit manipulation
- **Development tools**
  - Full hardware/software development package

Table 1. Device summary

Features	ST7MC1K2 / ST7MC1K4		ST7MC2N6 [1] / ST7MC2S4 / ST7MC2S6 / ST7MC2S7 / ST7MC2S9 / ST7MC2R6 / ST7MC2R7 / ST7MC2R9 / ST7MC2M9			
Program memory - bytes	8K	16K	16K	32K	48K	60K
RAM (stack) - bytes	384 (256)	768 (256)	768 (256)	1024 (256)	1536 (256)	
Peripherals	Watchdog, 16-bit Timer A, LINSCI™, 10-bit ADC, MTC, 8-bit PWM ART, ICD					
	-		SPI, 16-bit Timer B			
Operating Supply vs. Frequency	4.5 to 5.5V with $f_{CPU} \leq 8MHz$					
Temperature Range	-40°C to +85°C /-40°C to +125°C	-40°C to +85°C	-40°C to +85°C -40°C to +125°C	-40°C to +85 °C		-40°C to +125°C
Package	LQFP32	LQFP32	LQFP44	SDIP56[1]/LQFP64	LQFP64/44	LQFP80/64 LQFP44

Note 1: For development only. No production

Table of Contents

1 INTRODUCTION 5
2 PIN DESCRIPTION 6
3 REGISTER & MEMORY MAP 17
4 FLASH PROGRAM MEMORY 22
4.1 INTRODUCTION 22
4.2 MAIN FEATURES 22
4.3 STRUCTURE 22
4.4 ICC INTERFACE 23
4.5 ICP (IN-CIRCUIT PROGRAMMING) 24
4.6 IAP (IN-APPLICATION PROGRAMMING) 24
4.7 RELATED DOCUMENTATION 24
4.8 REGISTER DESCRIPTION 24
5 CENTRAL PROCESSING UNIT 25
5.1 INTRODUCTION 25
5.2 MAIN FEATURES 25
5.3 CPU REGISTERS 25
6 SUPPLY, RESET AND CLOCK MANAGEMENT 28
6.1 OSCILLATOR 29
6.2 RESET SEQUENCE MANAGER (RSM) 30
6.3 SYSTEM INTEGRITY MANAGEMENT (SI) 32
6.4 MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK AND BEEPER (MCC/RTC) . 37
7 INTERRUPTS 40
7.1 INTRODUCTION 40
7.2 MASKING AND PROCESSING FLOW 40
7.3 INTERRUPTS AND LOW POWER MODES 42
7.4 CONCURRENT & NESTED MANAGEMENT 42
7.5 INTERRUPT REGISTER DESCRIPTION 43
7.6 EXTERNAL INTERRUPTS 45
7.7 EXTERNAL INTERRUPT CONTROL REGISTER (EICR) 47
8 POWER SAVING MODES 50
8.1 INTRODUCTION 50
8.2 SLOW MODE 50
8.3 WAIT MODE 51
8.4 ACTIVE-HALT AND HALT MODES 52
9 I/O PORTS 54
9.1 INTRODUCTION 54
9.2 FUNCTIONAL DESCRIPTION 54
9.3 I/O PORT IMPLEMENTATION 57
9.4 LOW POWER MODES 57
9.5 INTERRUPTS 57
10 ON-CHIP PERIPHERALS 60
10.1 WINDOW WATCHDOG (WWDG) 60

10.2 PWM AUTO-RELOAD TIMER (ART) 67
10.3 16-BIT TIMER 76
10.4 SERIAL PERIPHERAL INTERFACE (SPI) 95
10.5 LINSCI SERIAL COMMUNICATION INTERFACE (LIN MASTER/SLAVE) 107
10.6 MOTOR CONTROLLER (MTC) 138
10.7 OPERATIONAL AMPLIFIER (OA) 233
10.8 10-BIT A/D CONVERTER (ADC) 236
11 INSTRUCTION SET 241
11.1 CPU ADDRESSING MODES 241
11.2 INSTRUCTION GROUPS 244
12 ELECTRICAL CHARACTERISTICS 247
12.1 PARAMETER CONDITIONS 247
12.2 ABSOLUTE MAXIMUM RATINGS 248
12.3 OPERATING CONDITIONS 250
12.4 SUPPLY CURRENT CHARACTERISTICS 252
12.5 CLOCK AND TIMING CHARACTERISTICS 256
12.6 MEMORY CHARACTERISTICS 260
12.7 EMC CHARACTERISTICS 261
12.8 I/O PORT PIN CHARACTERISTICS 264
12.9 CONTROL PIN CHARACTERISTICS 267
12.10 TIMER PERIPHERAL CHARACTERISTICS 270
12.11 COMMUNICATION INTERFACE CHARACTERISTICS 271
12.12 MOTOR CONTROL CHARACTERISTICS 273
12.13 OPERATIONAL AMPLIFIER CHARACTERISTICS 280
12.14 10-BIT ADC CHARACTERISTICS 281
13 PACKAGE CHARACTERISTICS 285
13.1 PACKAGE MECHANICAL DATA 285
13.2 THERMAL CHARACTERISTICS 288
13.3 SOLDERING INFORMATION 289
14 ST7MC DEVICE CONFIGURATION AND ORDERING INFORMATION 290
14.1 FLASH OPTION BYTES 290
14.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE 292
14.3 DEVELOPMENT TOOLS 294
14.4 ST7 APPLICATION NOTES 296
15 IMPORTANT NOTES 299
15.1 FLASH/FASTROM DEVICES ONLY 299
15.2 CLEARING ACTIVE INTERRUPTS OUTSIDE INTERRUPT ROUTINE 300
15.3 TIMD SET SIMULTANEOUSLY WITH OC INTERRUPT 300
15.4 LINSCI LIMITATIONS 300
15.5 MISSING DETECTION OF BLDC “Z EVENT” 303
15.6 INJECTED CURRENT ON PD7 303

Table of Contents

15.7 RESET VALUE OF UNAVAILABLE PINS 303
15.8 MAXIMUM VALUES OF AVD THRESHOLDS 304
15.9 EXTERNAL INTERRUPT MISSED 304
16 REVISION HISTORY 307

To obtain the most recent version of this datasheet,
please check at www.st.com>products>technical literature>datasheet

Please also pay special attention to the Section “IMPORTANT NOTES” on page 299.

1 INTRODUCTION

The ST7MCx device is member of the ST7 microcontroller family designed for mid-range applications with a Motor Control dedicated peripheral.

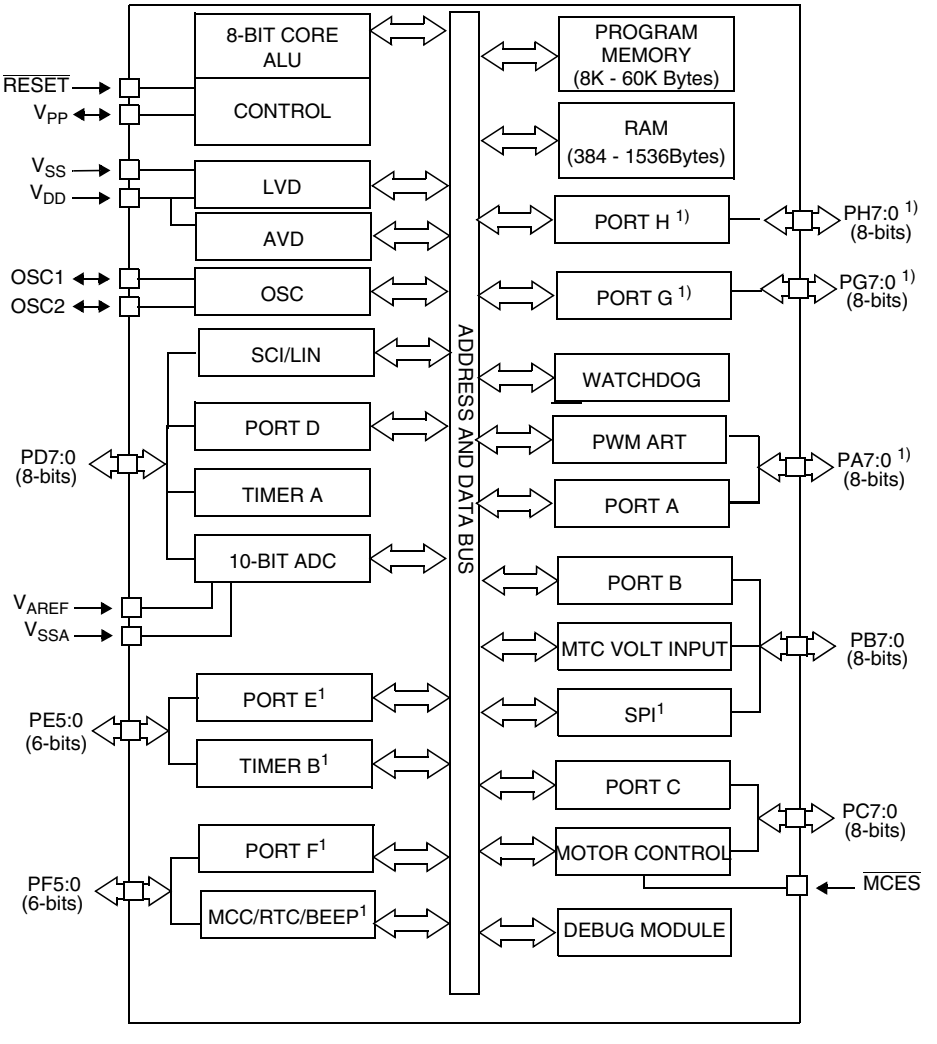
All devices are based on a common industry-standard 8-bit core, featuring an enhanced instruction set and are available with FLASH, ROM or FASTROM program memory.

Under software control, all devices can be placed in Wait, Slow, Active-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 ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned multiplication and indirect addressing modes.

The devices feature an on-chip Debug Module (DM) to support in-circuit debugging (ICD). For a description of the DM registers, refer to the ST7 ICC Protocol Reference Manual.

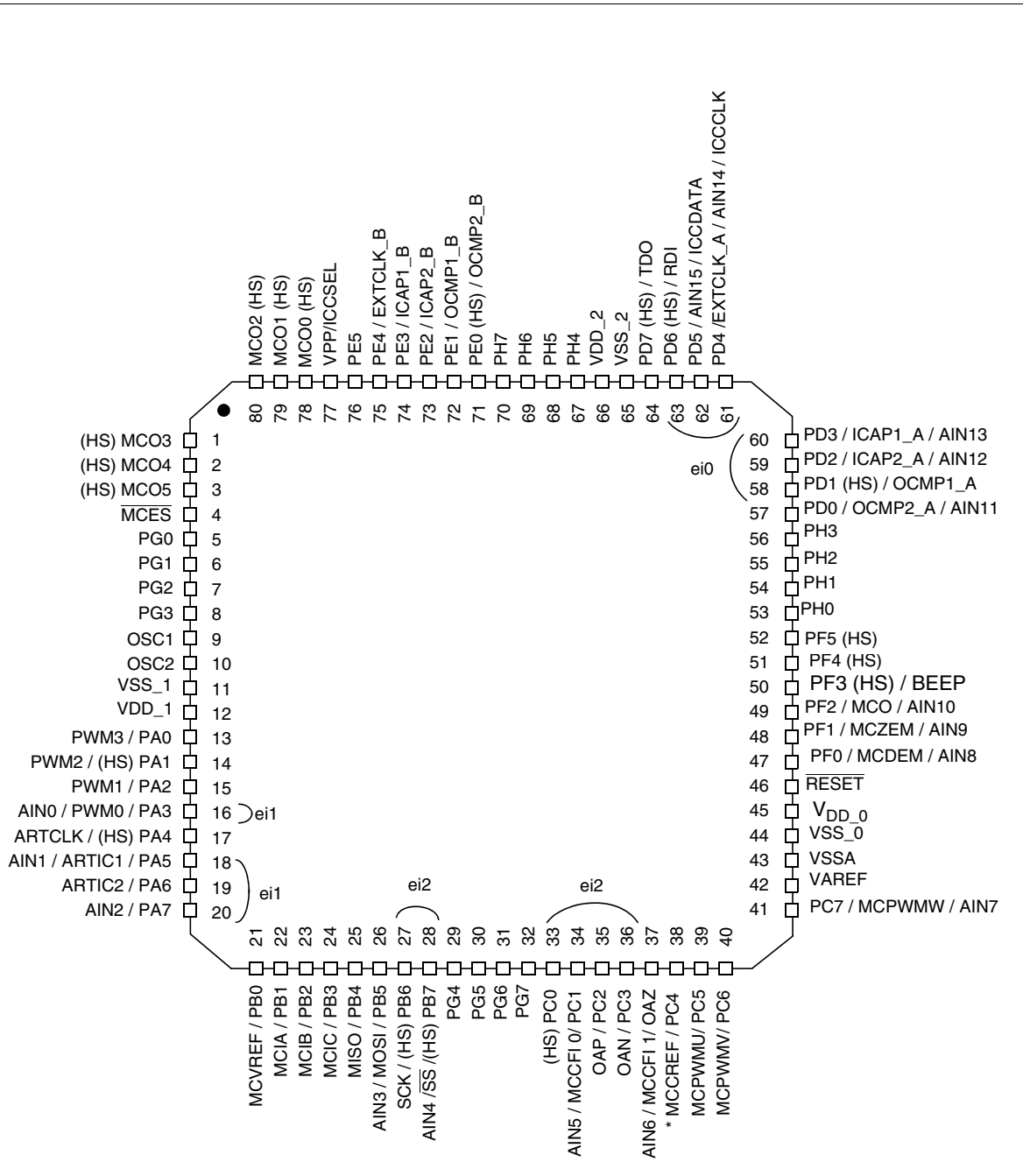
Figure 1. Device Block Diagram



On some devices only, see Table 1, “ST7MC Device Pin Description,” on page 12

2 PIN DESCRIPTION

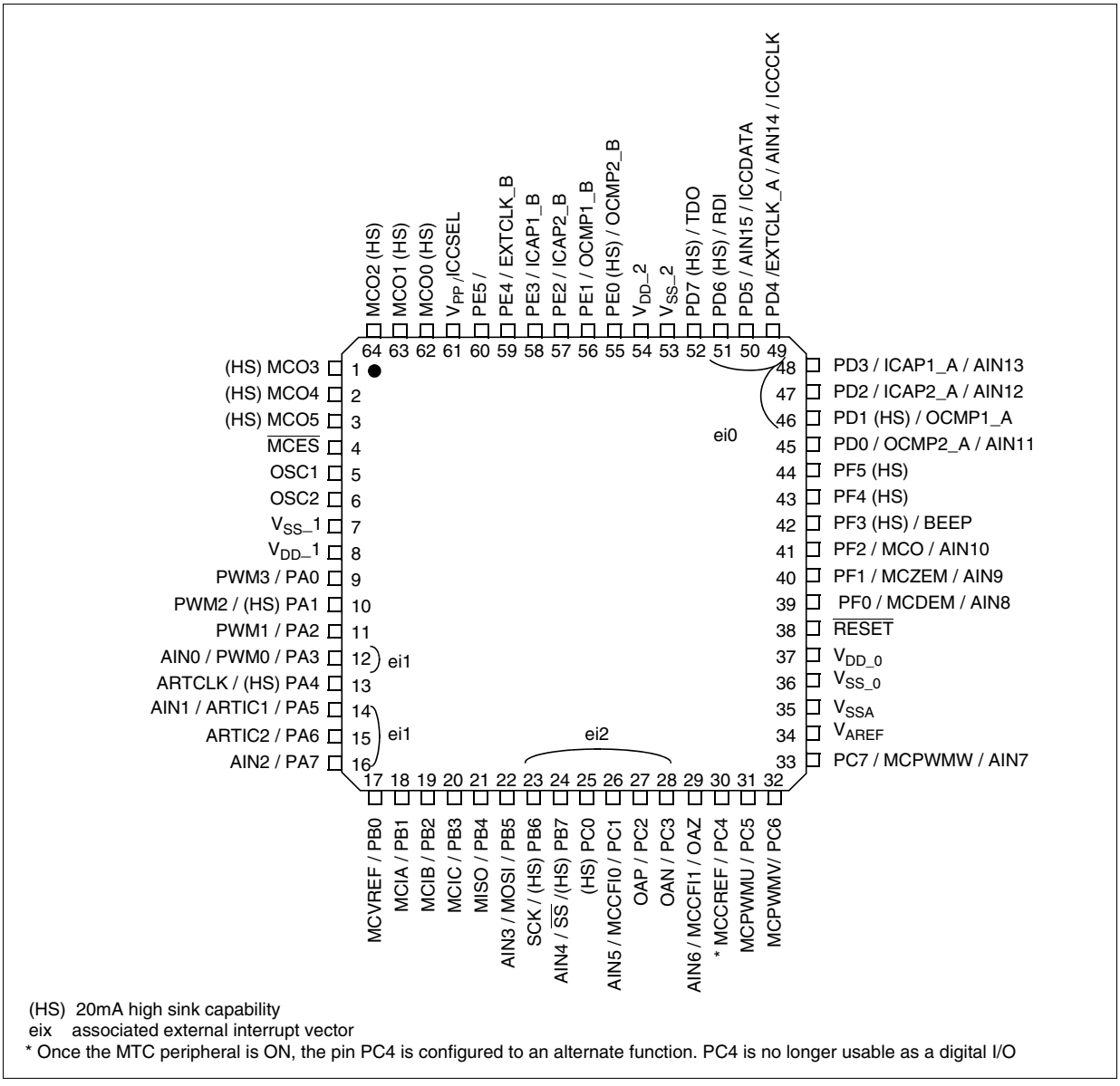
Figure 2. 80-Pin LQFP 14x14 Package Pinout



(HS) 20mA high sink capability
eix associated external interrupt vector
* Once the MTC peripheral is ON, the pin PC4 is configured to an alternate function. PC4 is no longer usable as a digital I/O

PIN DESCRIPTION (Cont'd)

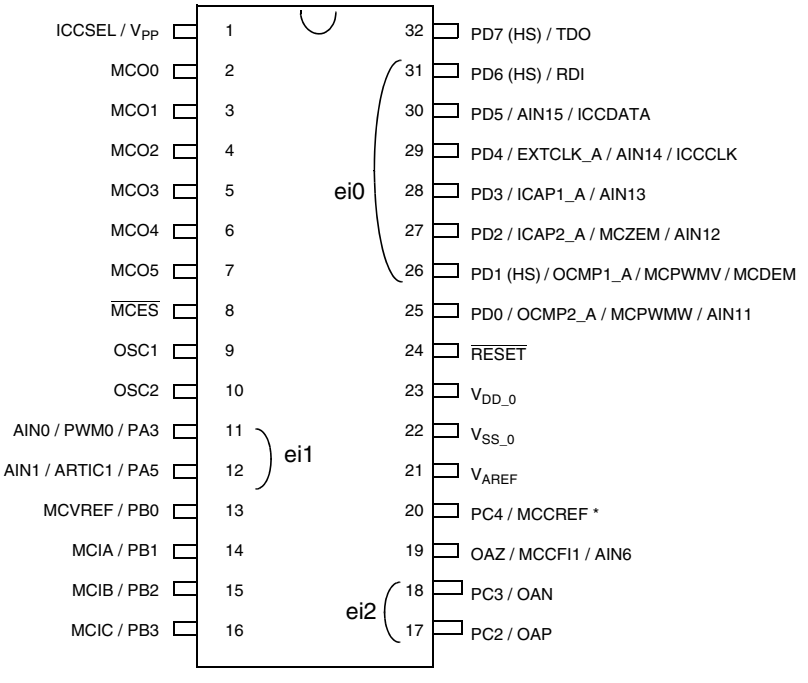
Figure 3. 64-Pin LQFP 14x14 Package Pinout



(HS) 20mA high sink capability
eix associated external interrupt vector
* Once the MTC peripheral is ON, the pin PC4 is configured to an alternate function. PC4 is no longer usable as a digital I/O

PIN DESCRIPTION (Cont'd)

Figure 4. 32-Pin SDIP Package Pinouts



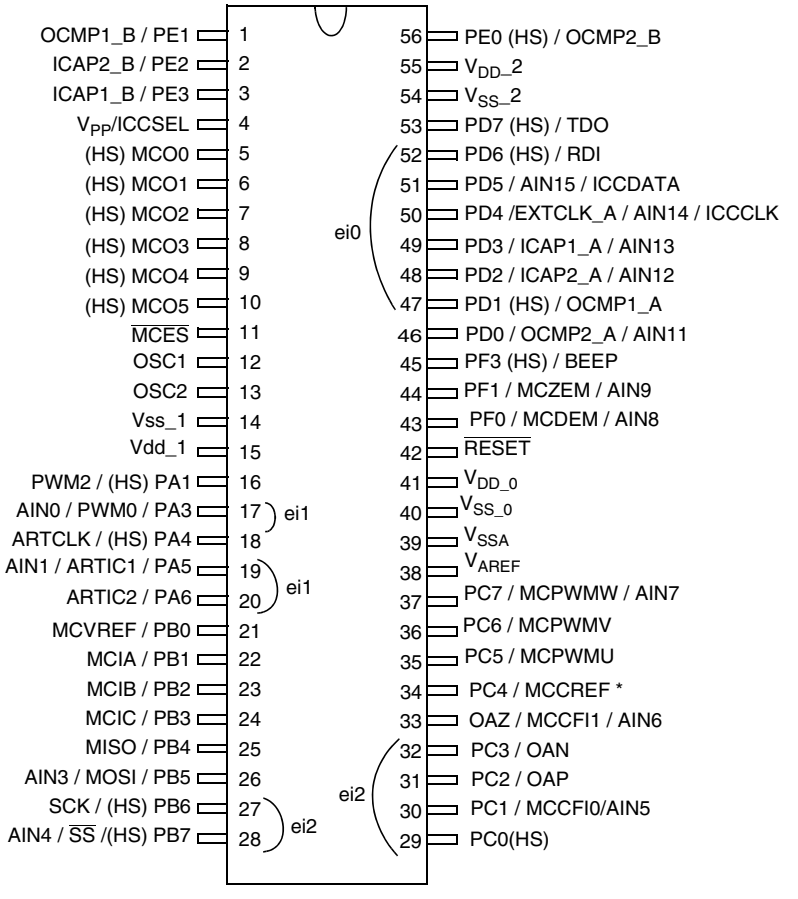
(HS) 20mA high sink capability

eix associated external interrupt vector

* Once the MTC peripheral is ON, the pin PC4 is configured to an alternate function. PC4 is no longer usable as a digital I/O

PIN DESCRIPTION (Cont'd)

Figure 5. 56-Pin SDIP Package Pinouts

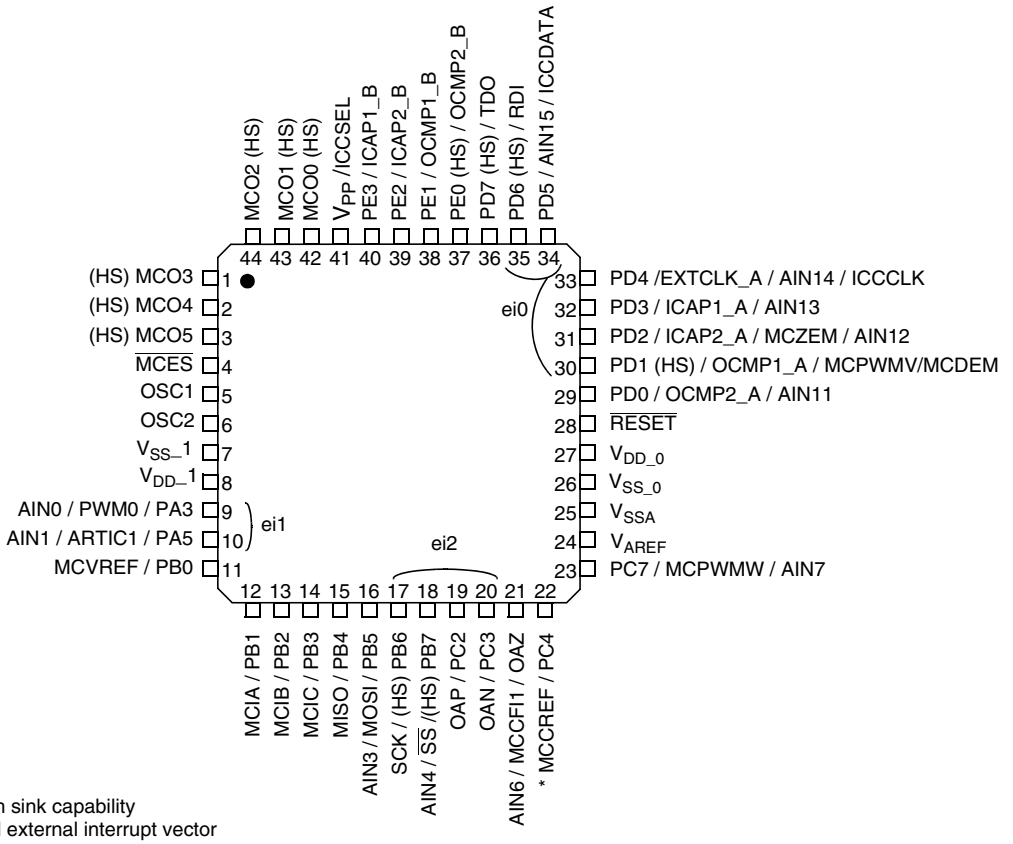


(HS) 20mA high sink capability
eix associated external interrupt vector

* Once the MTC peripheral is ON, the pin PC4 is configured to an alternate function. PC4 is no longer usable as a digital I/O

PIN DESCRIPTION (Cont'd)

Figure 6. 44-Pin LQFP Package Pinouts

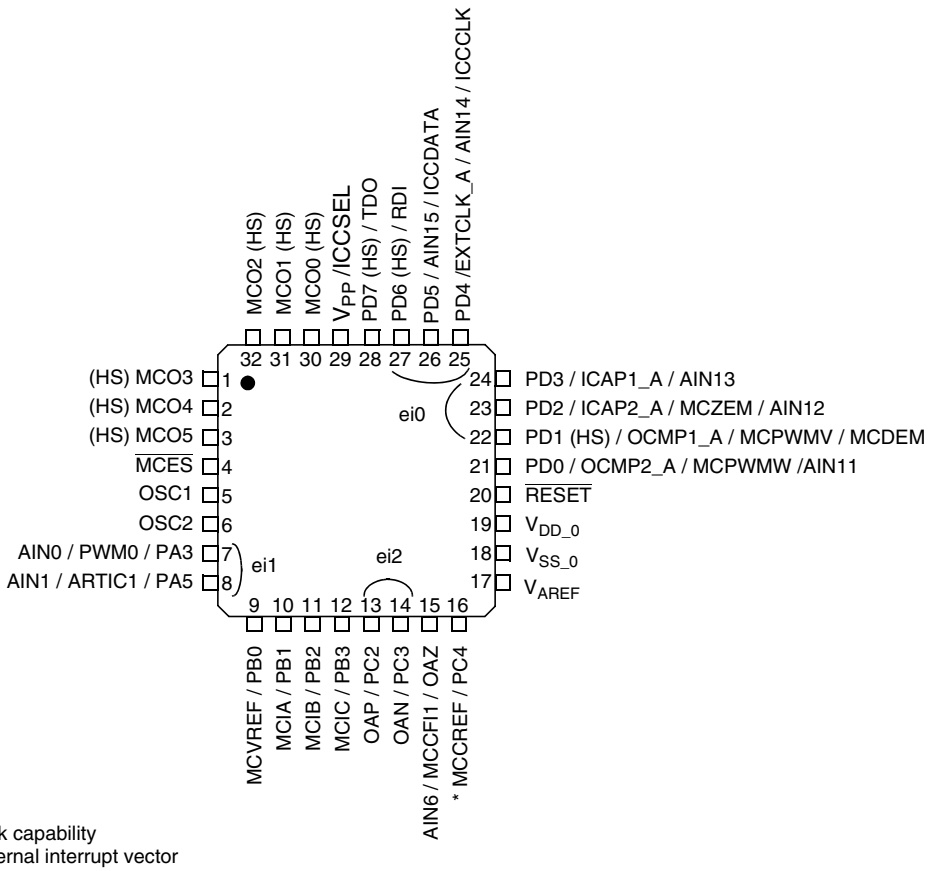


(HS) 20mA high sink capability
eix associated external interrupt vector

* Once the MTC peripheral is ON, the pin PC4 is configured to an alternate function. PC4 is no longer usable as a digital I/O

PIN DESCRIPTION (Cont'd)

Figure 7. 32-Pin LQFP 7x7 Package Pinout



(HS) 20mA high sink capability
eix associated external interrupt vector

* Once the MTC peripheral is ON, the pin PC4 is configured to an alternate function. PC4 is no longer usable as a digital I/O

PIN DESCRIPTION (Cont'd)

For external pin connection guidelines, See "ELECTRICAL CHARACTERISTICS" on page 247.

Legend / Abbreviations for Table 1:

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

Input level: A = Dedicated analog input

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

T_T= Refer to the G&H ports Characteristics in section 12.8.1 on page 264

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

Port and control configuration:

– Input: float = floating, wpu = weak pull-up, wpd = weak pull-down, int = interrupt [1], ana = analog

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

Refer to "I/O PORTS" on page 54 for more details on the software configuration of the I/O ports.

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

Table 1. ST7MC Device Pin Description

Pin n°						Pin Name	Type	Level		Port						Main function (after reset)	Alternate function [2]	
										Input [1]				Output				
LQFP80	LQFP64	SDIP56	LQFP44	SDIP32	LQFP32			Input	Output	float	wpu	int	ana	OD	PP			
1	1	8	1	5	1	MCO3 (HS)	O		HS						X	Motor Control Output 3		
2	2	9	2	6	2	MCO4 (HS)	O		HS						X	Motor Control Output 4		
3	3	10	3	7	3	MCO5 (HS)	O		HS						X	Motor Control Output 5		
4	4	11	4	8	4	$\overline{MCES}$ [3]	I	C_T		**X**						MTC Emergency Stop		
5	-	-	-	-	-	PG0	I/O	T_T		**X**	X			X	X	Port G0		
6	-	-	-	-	-	PG1	I/O	T_T		**X**	X			X	X	Port G1		
7	-	-	-	-	-	PG2	I/O	T_T		**X**	X			X	X	Port G2		
8	-	-	-	-	-	PG3	I/O	T_T		**X**	X			X	X	Port G3		
9	5	12	5	9	5	OSC1 [4]	I									External clock input or Resonator oscillator inverter input		
10	6	13	6	10	6	OSC2 [4]	I/O									Resonator oscillator inverter output		
11	7	14	7	-	-	V_{ss_1} [5]	S									Digital Ground Voltage		
12	8	15	8	-	-	V_{dd_1} [5]	S									Digital Main Supply Voltage		
13	9	-	-	-	-	PA0/PWM3	I/O	C_T		**X**	X			X	X	Port A0	PWM Output 3	
14	10	16	-	-	-	PA1/PWM2	I/O	C_T	HS	**X**	X			X	X	Port A1	PWM Output 2	
15	11	-	-	-	-	PA2PWM1	I/O	C_T		**X**	X			X	X	Port A2	PWM Output 1	
16	12	17	9	11	7	PA3/PWM0/ AIN0	I/O	C_T		**X**	ei1		X	X	X	Port A3	PWM Output 0	ADC Analog Input 0
17	13	18	-	-	-	PA4 (HS)/ARTCLK	I/O	C_T	HS	**X**	X			X	X	Port A4	PWM-ART External Clock	
18	14	19	10	12	8	PA5 / ARTIC1/ AIN1	I/O	C_T		**X**		ei1	X	X	X	Port A5	PWM-ART Input Capture 1	ADC Analog Input 1
19	15	20	-	-	-	PA6 / ARTIC2	I/O	C_T		**X**	ei1			X	X	Port A6	PWM-ART Input Capture 2	
20	16	-	-	-	-	PA7/AIN2	I/O	C_T		**X**		ei1	X	X	X	Port A7	ADC Analog Input 2	

Table 1. ST7MC Device Pin Description

Pin n°						Pin Name	Type	Level		Port						Main function (after reset)	Alternate function [2]	
										Input [1]				Output				
LQFP80	LQFP64	SDIP56	LQFP44	SDIP32	LQFP32			Input	Output	float	wpu	int	ana	OD	PP			
21	17	21	11	13	9	PB0/MCVREF	I/O	C_T		**X**	X		X	X	X	Port B0	MTC Voltage Reference	
22	18	22	12	14	10	PB1/MCIA	I/O	C_T		**X**	X		X	X	X	Port B1	MTC Input A	
23	19	23	13	15	11	PB2/MCIB	I/O	C_T		**X**	X		X	X	X	Port B2	MTC Input B	
24	20	24	14	16	12	PB3/MCIC	I/O	C_T		**X**	X		X	X	X	Port B3	MTC Input C	
25	21	25	15	-	-	PB4/MISO	I/O	C_T		**X**	X			X	X	Port B4	SPI Master In / Slave Out Data	
26	22	26	16	-	-	PB5/MOSI/ AIN3	I/O	C_T		**X**	X			X	X	Port B5	SPI Master Out / Slave In Data	ADC Analog Input 3
27	23	27	17	-	-	PB6/SCK	I/O	C_T	HS	**X**	ei2			X	X	Port B6	SPI Serial Clock	
28	24	28	18	-	-	PB7/$\overline{SS}$/AIN4	I/O	C_T	HS	**X**		ei2		X	X	Port B7	SPI Slave Select (active low)	ADC Analog Input 4
29	-	-	-	-	-	PG4	I/O	T_T		**X**	X			X	X	Port G4		
30	-	-	-	-	-	PG5	I/O	T_T		**X**	X			X	X	Port G5		
31	-	-	-	-	-	PG6	I/O	T_T		**X**	X			X	X	Port G6		
32	-	-	-	-	-	PG7	I/O	T_T		**X**	X			X	X	Port G7		
33	25	29	-	-	-	PC0	I/O	C_T	HS	**X**		ei2		X	X	Port C0		
34	26	30	-	-	-	PC1/MCCFI0[6] /AIN5	I/O	C_T		**X**	ei2		X	X	X	Port C1	MTC Current Feedback Input 0[6]	ADC Analog Input 5
35	27	31	19	17	13	PC2/OAP	I/O	C_T		**X**		ei2	X	X	X	Port C2	OPAMP Positive Input	
36	28	32	20	18	14	PC3/OAN	I/O	C_T		**X**	X	ei2	X	X	X	Port C3	OPAMP Negative Input	
37	29	33	21	19	15	OAZ/ MCCFI1[6]/ AIN6	I/O						X			Opamp Output	MTC Current Feedback Input 1[6]	ADC analog Input 6
38	30	34	22	20	16	PC4/MCCREF	I/O	C_T		**X**	X		X	X	X	Port C4	MTC Current Feedback Reference [9]	
39	31	35	-	-	-	PC5/MCPWMU	I/O	C_T		**X**	X			X	X	Port C5	MTC PWM Output U	
40	32	36	-	-	-	PC6/ MCPWMV[8]	I/O	C_T		**X**	X			X	X	Port C6	MTC PWM Output V[8]	
41	33	37	23	-	-	PC7/ MCPWMW[8]/ AIN7	I/O	C_T		**X**	X		X	X	X	Port C7	MTC PWM Output W[8]	ADC Analog Input 7
42	34	38	24	21	17	V_{AREF}	I									Analog Reference Voltage for ADC		
43	35	39	25	-	-	V_{SSA}[5]	S									Analog Ground Voltage		
44	36	40	26	22	18	V_{SS_0}[5]	S									Digital Ground Voltage		
45	37	41	27	23	19	V_{DD_0}[5]	S									Digital Main Supply Voltage		
46	38	42	28	24	20	$\overline{RESET}$	I/O	C_T								Top priority non maskable interrupt		

Table 1. ST7MC Device Pin Description

Pin n°						Pin Name	Type	Level		Port						Main function (after reset)	Alternate function [2]	
										Input [1]				Output				
LQFP80	LQFP64	SDIP56	LQFP44	SDIP32	LQFP32			Input	Output	float	wpu	int	ana	OD	PP			
47	39	43	-	-	-	PF0/ MCDEM[7]/ AIN8	I/O	C_T		**X**	X		X	X	X	Port F0	MTC Demagnetization Output[7]	ADC Analog Input 8
48	40	44	-	-	-	PF1/MCZEM[7]/ AIN9	I/O	C_T		**X**	X		X	X	X	Port F1	MTC BEMF Output[7]	ADC Analog Input 9
49	41	-	-	-	-	PF2/MCO/ AIN10	I/O	C_T		**X**	X		X	X	X	Port F2	Main Clock Out (f_{osc}/2)	ADC Analog Input 10
50	42	45	-	-	-	PF3/BEEP	I/O	C_T	HS	**X**	X			X	X	Port F3	Beep Signal Output	
51	43	-	-	-	-	PF4	I/O	C_T	HS	**X**	X			X	X	Port F4		
52	44	-	-	-	-	PF5	I/O	C_T	HS	**X**	X			X	X	Port F5		
53	-	-	-	-	-	PH0	I/O	T_T		**X**	X			X	X	Port H0		
54	-	-	-	-	-	PH1	I/O	T_T		**X**	X			X	X	Port H1		
55	-	-	-	-	-	PH2	I/O	T_T		**X**	X			X	X	Port H2		
56	-	-	-	-	-	PH3	I/O	T_T		**X**	X			X	X	Port H3		
57	45	46	29	25	21	PD0/ OCMP2_A/ MCPWMW[8]/ AIN11	I/O	C_T		**X**			X	X	X	Port D0	Timer A Output Compare 2 MTC PWM Output W[8] ADC Analog Input 11	
58	46	47	30	26	22	PD1 (HS)/ OCMP1_A/ MCPWMV[8]/ MCDEM[7]	I/O	C_T	HS	**X**		ei0		X	X	Port D1	Timer A Output Compare 1 MTC PWM Output V[8] MTC Demagnetization[7]	
59	47	48	31	27	23	PD2/ICAP2_A/ MCZEM[7] / AIN12	I/O	C_T		**X**	ei0		X	X	X	Port D2	Timer A Input Capture 2 MTC BEMF[7] ADC Analog Input 12	
60	48	49	32	28	24	PD3/ICAP1_A/ AIN13	I/O	C_T		**X**		ei0	X	X	X	Port D3	Timer A Input Capture 1	ADC Analog Input 13
61	49	50	33	29	25	PD4/ EXTCLK_A/ICCCLK/AIN14	I/O	C_T		**X**	ei0		X	X	X	Port D4	Timer A External Clock source ICC Clock Output ADC Analog Input 14	
62	50	51	34	30	26	PD5/ICCDATA/AIN15	I/O	C_T		**X**	ei0		X	X	X	Port D5	ICC Data Input ADC Analog Input 15	
63	51	52	35	31	27	PD6/RDI	I/O	C_T	HS	**X**		ei0		X	X	Port D6	SCI Receive Data In	
64	52	53	36	32	28	PD7/TDO	I/O	C_T	HS	**X**	X			X	X	Port D7	SCI Transmit Data Output	
65	53	54	-	-	-	V_{SS_2}	S									Digital Ground Voltage		
66	54	55	-	-	-	V_{DD_2}	S									Digital Main Supply Voltage		
67	-	-	-	-	-	PH4	I/O	T_T		**X**	X			X	X	Port H4		
68	-	-	-	-	-	PH5	I/O	T_T		**X**	X			X	X	Port H5		

Table 1. ST7MC Device Pin Description

Pin n°						Pin Name	Type	Level		Port						Main function (after reset)	Alternate function [2]
										Input [1]				Output			
LQFP80	LQFP64	SDIP56	LQFP44	SDIP32	LQFP32			Input	Output	float	wpu	int	ana	OD	PP		
69	-	-	-	-	-	PH6	I/O	T_T		**X**	X			X	X	Port H6	
70	-	-	-	-	-	PH7	I/O	T_T		**X**	X			X	X	Port H7	
71	55	56	37	-	-	PE0/ OCMP2_B	I/O	C_T	HS	**X**	X			X	X	Port E0	Timer B Output Compare 2
72	56	1	38	-	-	PE1/ OCMP1_B	I/O	C_T		**X**	X		X	X	X	Port E1	Timer B Output Compare 1
73	57	2	39	-	-	PE2/ICAP2_B	I/O	C_T		**X**	X			X	X	Port E2	Timer B Input Capture 2
74	58	3	40	-	-	PE3/ICAP1_B/	I/O	C_T		**X**	X		X	X	X	Port E3	Timer B Input Capture 1
75	59	-	-	-	-	PE4/ EXTCLK_B	I/O	C_T		**X**	X			X	X	Port E4	Timer B External Clock source
76	60	-	-	-	-	PE5	I/O	C_T		**X**	X		X	X	X	Port E5	
77	61	4	41	1	29	V_{PP}/ICCSEL	I									Must be tied low. In the programming mode when available, this pin acts as the programming voltage input V_{PP}./ ICC mode pin. See section 12.9.2 on page 269	
78	62	5	42	2	30	MCO0 (HS)	O		HS						X	MTC Output Channel 0	
79	63	6	43	3	31	MCO1 (HS)	O		HS						X	MTC Output Channel 1	
80	64	7	44	4	32	MCO2 (HS)	O		HS						X	MTC Output Channel 2	

Notes:
1. In the interrupt input column, “eiX” defines the associated external interrupt vector. If the weak pull-up column (wpu) is merged with the interrupt column (int), then the I/O configuration is pull-up interrupt input, else the configuration is floating interrupt input

2. If two alternate function outputs are enabled at the same time on a given pin (for instance, MCPWMV and MCDEM on PD1 on LQFP32), the two signals will be ORed on the output pin.

3. $\overline{\text{MCES}}$ is a floating input. To disable this function, a pull-up resistor must be used.

4. OSC1 and OSC2 pins connect a crystal/ceramic resonator or an external source to the on-chip oscillator; see Section 1 INTRODUCTION and Section 12.5 CLOCK AND TIMING CHARACTERISTICS for more details.

5. It is mandatory to connect all available V_{DD} and V_{DDA} pins to the supply voltage and all V_{SS} and V_{SSA} pins to ground.

6. MCCFI can be mapped on 2 different pins on 80 ,64 and 56-pin packages. This allows:

- either to use PC1 as a standard I/O and map MCCFI on OAZ (MCCFI1) with or without using the operational amplifier (selected case after reset),

- or to map MCCFI on PC1 (MCCFI0) and use the amplifier for another function.

The mapping can be selected in MREF register of motor control cell. See section MOTOR CONTROL for more details.

7. MCZEM is mapped on PF1 on 80, 64 and 56-pin packages and on PD2 on 44 and 32-pins.
MCDEM is mapped on PF0 on 80, 64 and 56-pin packages and on PD1 on 44 and 32-pin packages.

8. MCPWMV is mapped on PC6 on 80 and 64-pin packages and on PD1 on 44,and 32-pins packages.
MCPWMW is mapped on PC7 on 80, 64 and 44-pin packages and on PD0 on 32-pins package.

9. Once the MTC peripheral is ON (bits CKE=1 or DAC=1 in the register MCRA), the pin PC4 is configured

to an alternate function. PC4 is no longer usable as a digital I/O.l

10. On the chip, each I/O port has 8 pads. 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. Refer to section 15.7 on page 303

3 REGISTER & MEMORY MAP

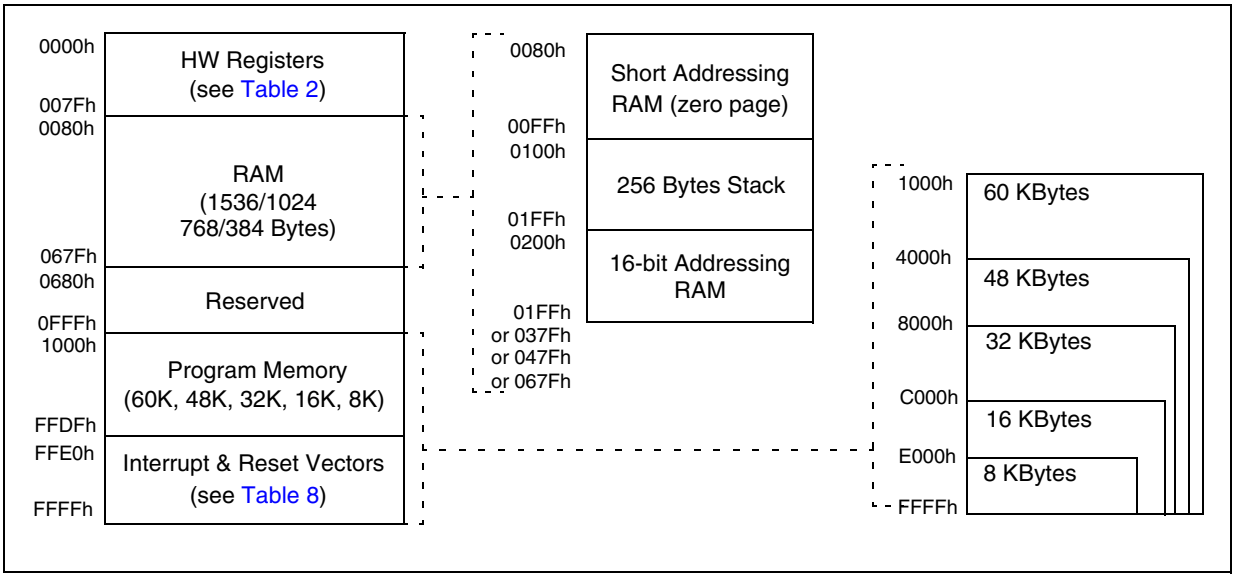
As shown in Figure 8, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register locations, up to 2Kbytes of RAM and up to 60Kbytes of user program memory. The RAM space includes up to 256 bytes for the stack from 0100h to 01FFh.

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

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

Figure 8. Memory Map



As shown in Figure 9, the MCU is capable of addressing 64K bytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register locations, up to 1536 bytes of RAM and up to 60 Kbytes of user program memory. The RAM space includes up to 256 bytes for the stack from 0100h to 01FFh.

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

Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	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	00h	R/W[2]
0003h	Port B	PBDR	Port B Data Register	00h[1]	R/W
0004h		PBDDR	Port B Data Direction Register	00h	R/W
0005h		PBOR	Port B Option Register	00h	R/W
0006h	Port C	PCDR	Port C Data Register	00h[1]	R/W
0007h		PCDDR	Port C Data Direction Register	00h	R/W
0008h		PCOR	Port C Option Register	00h	R/W
0009h	Port D	PDDR	Port D Data Register	00h[1]	R/W
000Ah		PDDDR	Port D Data Direction Register	00h	R/W
000Bh		PDOR	Port D Option Register	00h	R/W
000Ch	Port E	PEDR	Port E Data Register	00h[1]	R/W
000Dh		PEDDR	Port E Data Direction Register	00h	R/W[2]
000Eh		PEOR	Port E Option Register	00h	R/W[2]
000Fh	Port F	PFDR	Port F Data Register	00h[1]	R/W
0010h		PFDDR	Port F Data Direction Register	00h	R/W
0011h		PFOR	Port F Option Register	00h	R/W
0012h	Port G	PGDR	Port G Data Register	00h[1]	R/W
0013h		PGDDR	Port G Data Direction Register	00h	R/W
0014h		PGOR	Port G Option Register	00h	R/W
0015h	Port H	PHDR	Port H Data Register	00h[1]	R/W
0016h		PHDDR	Port H Data Direction Register	00h	R/W
0017h		PHOR	Port H Option Register	00h	R/W
0018h	LINSCI™	SCISR	SCI Status Register	C0h	Read Only
0019h		SCIDR	SCI Data Register	xxh	R/W
001Ah		SCIBRR	SCI Baud Rate Register	00h	R/W
001Bh		SCICR1	SCI Control Register 1	xxh	R/W
001Ch		SCICR2	SCI Control Register 2	00h	R/W
001Dh		SCICR3	SCI Control Register 3	00h	R/W
001Eh		SCIERPR	SCI Extended Receive Prescaler Register	00h	R/W
001Fh		SCIETPR	SCI Extended Transmit Prescaler Register	00h	R/W
0020h	Reserved Area (1 Byte)				
0021h	SPI	SPIDR	SPI Data I/O Register	xxh	R/W
0022h		SPICR	SPI Control Register	0xh	R/W
0023h		SPICSR	SPI Control/Status Register	00h	R/W

Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0024h	ITC	ITSPR0	Interrupt Software Priority Register 0	FFh	R/W
0025h		ITSPR1	Interrupt Software Priority Register 1	FFh	R/W
0026h		ITSPR2	Interrupt Software Priority Register 2	FFh	R/W
0027h		ITSPR3	Interrupt Software Priority Register 3	FFh	R/W
0028h		EICR	External Interrupt Control Register	00h	R/W
0029h	FLASH	FSCR	Flash Control/Status Register	00h	R/W
002Ah	WATCHDOG	WDGCR	Window Watchdog Control Register	7Fh	R/W
002Bh		WDGWR	Window Watchdog Window Register	7Fh	R/W
002Ch	MCC	MCCSR	Main Clock Control / Status Register	00h	R/W
002Dh		MCCBCR	Main Clock Controller: Beep Control Register	00h	R/W
002Eh	ADC	ADCCSR	Control/Status Register	00h	R/W
002Fh		ADCDRMSB	Data Register MSB	00h	Read Only
0030h		ADCDRLSB	Data Register LSB	00h	Read Only
0031h	TIMER A	TACR2	Timer A Control Register 2	00h	R/W
0032h		TACR1	Timer A Control Register 1	00h	R/W
0033h		TACSR	Timer A Control/Status Register	xxh	R/W
0034h		TAIC1HR	Timer A Input Capture 1 High Register	xxh	Read Only
0035h		TAIC1LR	Timer A Input Capture 1 Low Register	xxh	Read Only
0036h		TAOC1HR	Timer A Output Compare 1 High Register	80h	R/W
0037h		TAOC1LR	Timer A Output Compare 1 Low Register	00h	R/W
0038h		TACHR	Timer A Counter High Register	FFh	Read Only
0039h		TACLR	Timer A Counter Low Register	FCh	Read Only
003Ah		TAACHR	Timer A Alternate Counter High Register	FFh	Read Only
003Bh		TAACLR	Timer A Alternate Counter Low Register	FCh	Read Only
003Ch		TAIC2HR	Timer A Input Capture 2 High Register	xxh	Read Only
003Dh		TAIC2LR	Timer A Input Capture 2 Low Register	xxh	Read Only
003Eh		TAOC2HR	Timer A Output Compare 2 High Register	80h	R/W
003Fh		TAOC2LR	Timer A Output Compare 2 Low Register	00h	R/W
0040h	SIM	SICSR	System Integrity Control/Status Register	000x000x b	R/W
0041h	TIMER B	TBCR2	Timer B Control Register 2	00h	R/W
0042h		TBCR1	Timer B Control Register 1	00h	R/W
0043h		TBCSR	Timer B Control/Status Register	xxh	R/W
0044h		TBIC1HR	Timer B Input Capture 1 High Register	xxh	Read Only
0045h		TBIC1LR	Timer B Input Capture 1 Low Register	xxh	Read Only
0046h		TBOC1HR	Timer B Output Compare 1 High Register	80h	R/W
0047h		TBOC1LR	Timer B Output Compare 1 Low Register	00h	R/W
0048h		TBCHR	Timer B Counter High Register	FFh	Read Only
0049h		TBCLR	Timer B Counter Low Register	FCh	Read Only
004Ah		TBACHR	Timer B Alternate Counter High Register	FFh	Read Only
004Bh		TBACLR	Timer B Alternate Counter Low Register	FCh	Read Only
004Ch		TBIC2HR	Timer B Input Capture 2 High Register	xxh	Read Only
004Dh		TBIC2LR	Timer B Input Capture 2 Low Register	xxh	Read Only
004Eh		TBOC2HR	Timer B Output Compare 2 High Register	80h	R/W
004Fh		TBOC2LR	Timer B Output Compare 2 Low Register	00h	R/W

Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0050h	MTC (page 0)	MTIM	Timer Counter High Register	00h	R/W
0051h		MTIML	Timer Counter Low Register	00h	R/W
0052h		MZPRV	Capture Z_{n-1} Register	00h	R/W
0053h		MZREG	Capture Z_n Register	00h	R/W
0054h		MCOMP	Compare C_{n+1} Register	00h	R/W
0055h		MDREG	Demagnetization Register	00h	R/W
0056h		MWGHT	A_n Weight Register	00h	R/W
0057h		MPRSR	Prescaler & Sampling Register	00h	R/W
0058h		MIMR	Interrupt Mask Register	00h	R/W
0059h		MISR	Interrupt Status Register	00h	R/W
005Ah		MCRA	Control Register A	00h	R/W
005Bh		MCRB	Control Register B	00h	R/W
005Ch		MCRC	Control Register C	00h	R/W
005Dh		MPHST	Phase State Register	00h	R/W
005Eh		MDFR	D event Filter Register	0Fh	R/W
005Fh		MCFR	Current feedback Filter Register	00h	R/W
0060h		MREF	Reference Register	00h	R/W
0061h		MPCR	PWM Control Register	00h	R/W
0062h		MREP	Repetition Counter Register	00h	R/W
0063h		MCPWH	Compare Phase W Preload Register High	00h	R/W
0064h		MCPWL	Compare Phase W Preload Register Low	00h	R/W
0065h		MCPVH	Compare Phase V Preload Register High	00h	R/W
0066h		MCPVL	Compare Phase V Preload Register Low	00h	R/W
0067h		MCPUH	Compare Phase U Preload Register High	00h	R/W
0068h		MCPUL	Compare Phase U Preload Register Low	00h	R/W
0069h		MCP0H	Compare Phase 0 Preload Register High	0Fh	R/W
006Ah		MCP0L	Compare Phase 0 Preload Register Low	FFh	R/W
0050h	MTC (page 1)	MDTG	Dead Time Generator Enable	FFh	see MTC description
0051h		MPOL	Polarity Register	3Fh	
0052h		MPWME	PWM Register	00h	
0053h		MCONF	Configuration Register	02h	
0054h		MPAR	Parity Register	00h	
0055h		MZRF	Z event Filter Register	0Fh	
0056h		MSCR	Sampling Clock Register	00h	
0057h to 006Ah		Reserved Area (4 Bytes)			
006Bh	DM	DMCR	Debug Control Register	00h	R/W
006Ch		DMSR	Debug Status Register	10h	Read Only
006Dh		DMBK1H	Debug Breakpoint 1 MSB Register	FFh	R/W
006Eh		DMBK1L	Debug Breakpoint 1 LSB Register	FFh	R/W
006Fh		DMBK2H	Debug Breakpoint 2 MSB Register	FFh	R/W
0070h		DMBK2L	Debug Breakpoint 2 LSB Register	FFh	R/W

Table 2. Hardware Register Map

Address	Block	Register Label	Register Name	Reset Status	Remarks
0074h	PWM ART	PWMDCR3	PWM AR Timer Duty Cycle Register 3	00h	R/W
0075h		PWMDCR2	PWM AR Timer Duty Cycle Register 2	00h	R/W
0076h		PWMDCR1	PWM AR Timer Duty Cycle Register 1	00h	R/W
0077h		PWMDCR0	PWM AR Timer Duty Cycle Register 0	00h	R/W
0078h		PWMCR	PWM AR Timer Control Register	00h	R/W
0079h		ARTCSR	Auto-Reload Timer Control/Status Register	00h	R/W
007Ah		ARTCAR	Auto-Reload Timer Counter Access Register	00h	R/W
007Bh		ARTARR	Auto-Reload Timer Auto-Reload Register	00h	R/W
007Ch		ARTICCSR	AR Timer Input Capture Control/Status Reg.	00h	R/W
007Dh		ARTICR1	AR Timer Input Capture Register 1	00h	Read Only
007Eh		ARTICR2	AR Timer Input Capture Register 2	00h	Read Only
007Fh	OPAMP	OACSR	OPAMP Control/Status Register	00h	R/W

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

Notes:

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

2. The bits associated with unavailable pins must always keep their reset value.

4 FLASH PROGRAM MEMORY

4.1 INTRODUCTION

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

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

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

4.2 MAIN FEATURES

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

4.3 STRUCTURE

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

Depending on the overall Flash memory size in the microcontroller device, there are up to three user sectors (see Table 3). Each of these sectors can be erased independently to avoid unnecessary erasing of the whole Flash memory when only a partial erasing is required.

The first two sectors have a fixed size of 4 Kbytes (see Figure 9). They are mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

Table 3. Sectors available in Flash devices

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

4.3.1 Read-out Protection

Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller.

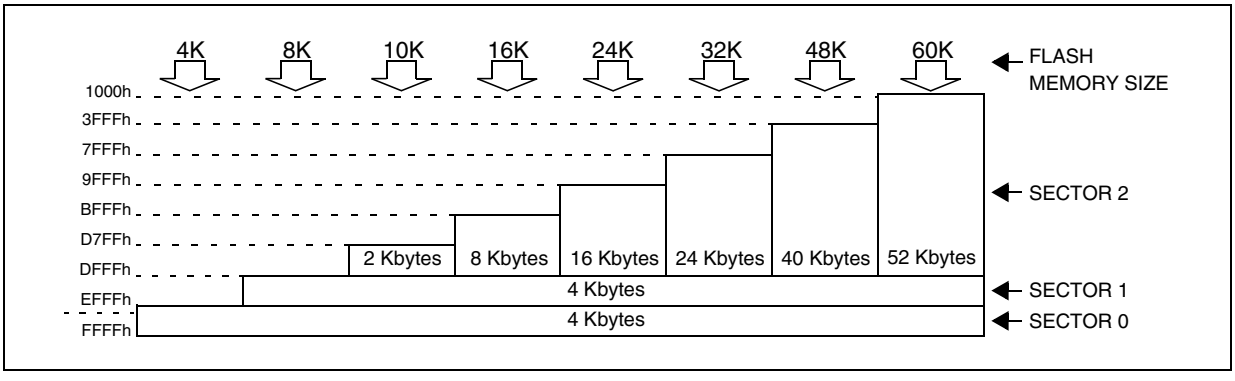
In Flash devices, this protection is removed by reprogramming the option. In this case, the entire program memory is first automatically erased and the device can be reprogrammed.

Read-out protection selection depends on the device type:

– In Flash devices it is enabled and removed through the FMP_R bit in the option byte.

– In ROM devices it is enabled by mask option specified in the Option List.

Figure 9. Memory Map and Sector Address



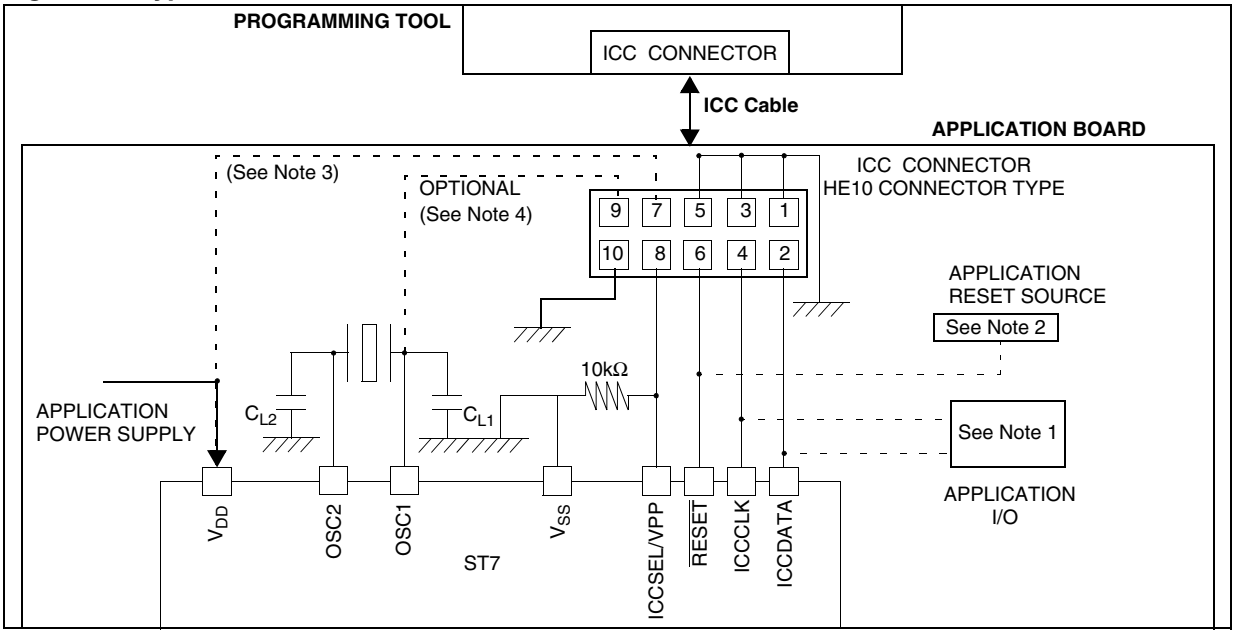
FLASH PROGRAM MEMORY (Cont'd)

4.4 ICC INTERFACE

ICC (In-Circuit Communication) needs a minimum of four and up to six pins to be connected to the programming tool (see Figure 10). These pins are:

- $\overline{\text{RESET}}$: device reset
- V_{SS}: device power supply ground
- ICCCLK: ICC output serial clock pin
- ICCDATA: ICC input/output serial data pin
- ICCSEL/V_{PP}: programming voltage
- OSC1(or OSCIN): main clock input for external source (optional)
- V_{DD}: application board power supply (see Figure 10, Note 3)

Figure 10. Typical ICC Interface



Notes:

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

2. During the ICC session, the programming tool must control the $\overline{\text{RESET}}$ pin. This can lead to conflicts between the programming tool and the application reset circuit if it drives more than 5mA at high level (push pull output or pull-up resistor<1K). A schottky diode can be used to isolate the application RESET circuit in this case. When using a classical RC network with R > 1K or a reset management IC with open drain output and pull-up resistor > 1K, no additional components are needed. In all cases the user must ensure that no external reset is generated by the application during the ICC session.

3. The use of Pin 7 of the ICC connector depends on the Programming Tool architecture. This pin must be connected when using most ST Programming Tools (it is used to monitor the application power supply). Please refer to the Programming Tool manual.

4. Pin 9 has to be connected to the OSC1 or OSCIN pin of the ST7 when the clock is not available in the application or if the selected clock option is not programmed in the option byte. ST7 devices with multi-oscillator capability need to have OSC2 grounded in this case.

FLASH PROGRAM MEMORY (Cont'd)

4.5 ICP (IN-CIRCUIT PROGRAMMING)

To perform ICP the microcontroller must be switched to ICC (In-Circuit Communication) mode by an external controller or programming tool.

Depending on the ICP code downloaded in RAM, Flash memory programming can be fully customized (number of bytes to program, program locations, or selection serial communication interface for downloading).

When using an STMicroelectronics or third-party programming tool that supports ICP and the specific microcontroller device, the user needs only to implement the ICP hardware interface on the application board (see Figure 10). For more details on the pin locations, refer to the device pinout description.

4.6 IAP (IN-APPLICATION PROGRAMMING)

This mode uses a BootLoader program previously stored in Sector 0 by the user (in ICP mode or by plugging the device in a programming tool).

This mode is fully controlled by user software. This allows it to be adapted to the user application, (user-defined strategy for entering programming mode, choice of communications protocol used to fetch the data to be stored, etc.). For example, it is possible to download code from the SPI, SCI or other type of serial interface and program it in the Flash. IAP mode can be used to program any of the Flash sectors except Sector 0, which is write/erase protected to allow recovery in case errors occur during the programming operation.

4.7 RELATED DOCUMENTATION

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

4.8 REGISTER DESCRIPTION

FLASH CONTROL/STATUS REGISTER (FCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	0	0

This register is reserved for use by Programming Tool software. It controls the Flash programming and erasing operations.

5 CENTRAL PROCESSING UNIT

5.1 INTRODUCTION

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

5.2 MAIN FEATURES

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

5.3 CPU REGISTERS

The six CPU registers shown in Figure 11 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)

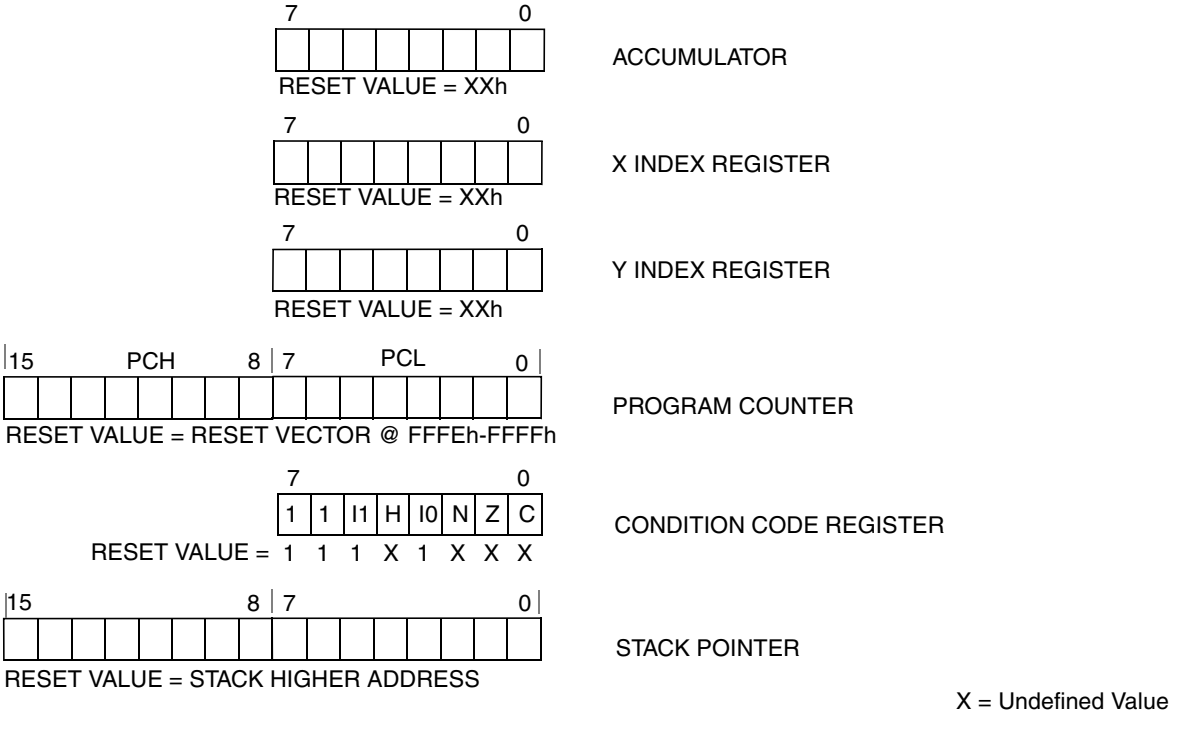
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation. (The Cross-Assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register.)

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 11. 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: 01 FFh

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 12).

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

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

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

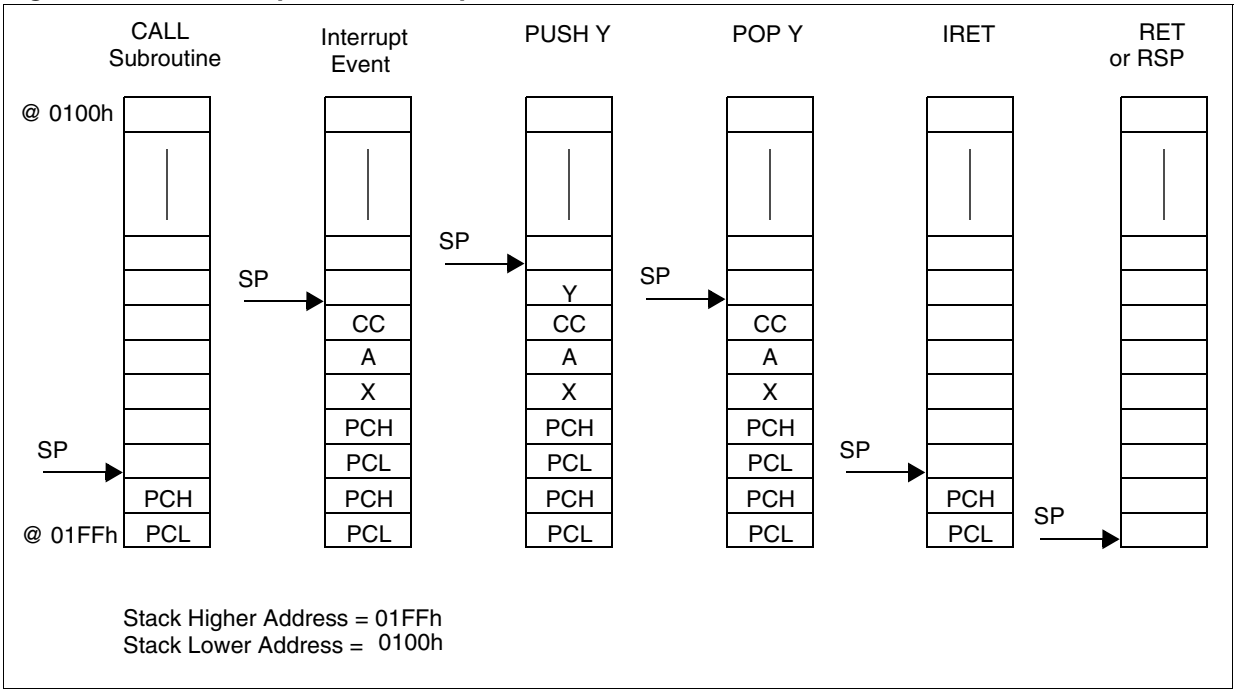
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the PUSH and POP instructions. In the case of an interrupt, the PCL is stored at the first location pointed to by the SP. Then the other registers are stored in the next locations as shown in Figure 12.

– 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 12. Stack Manipulation Example



6 SUPPLY, RESET AND CLOCK MANAGEMENT

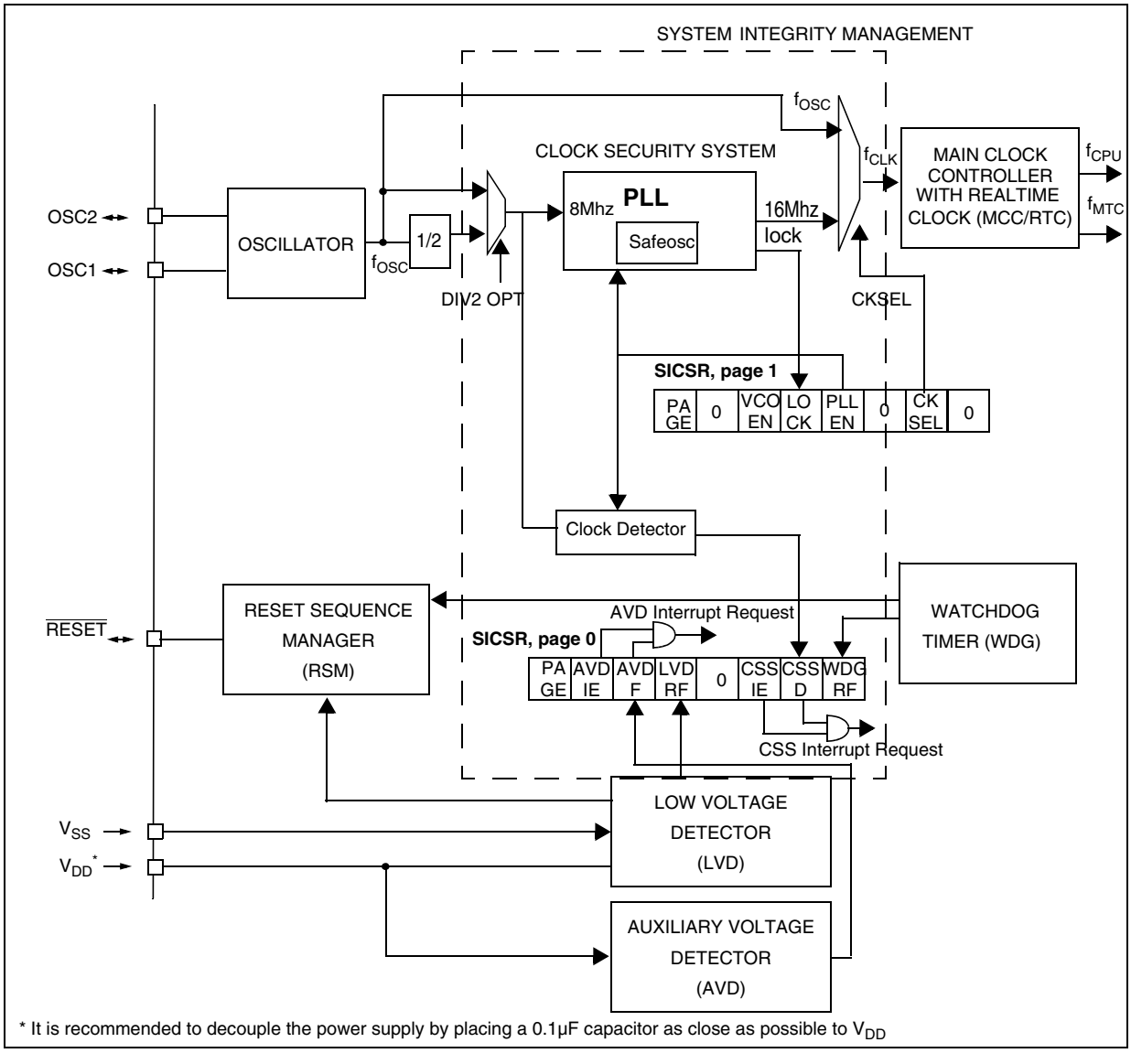
The device includes a range of utility features for securing the application in critical situations (for example in case of a power brown-out), and reducing the number of external components. An overview is shown in Figure 13.

For more details, refer to dedicated parametric section.

Main features

- Reset Sequence Manager (RSM)
- 1 Crystal/Ceramic resonator oscillator
- System Integrity Management (SI)
  - Main supply Low voltage detection (LVD)
  - Auxiliary Voltage detector (AVD) with interrupt capability for monitoring the main supply
  - Clock Security System (CSS) with the VCO of the PLL, providing a backup safe oscillator
  - Clock Detector
  - PLL which can be used to multiply the frequency by 2 if the clock frequency input is 8MHz

Figure 13. Clock, Reset and Supply Block Diagram



* It is recommended to decouple the power supply by placing a 0.1µF capacitor as close as possible to V_{DD}

6.1 OSCILLATOR

The main clock of the ST7 can be generated by a crystal or ceramic resonator oscillator or an external source.

The associated hardware configurations are shown in Table 4. Refer to the electrical characteristics section for more details.

External Clock Source

In this external clock mode, a clock signal (square, sinus or triangle) with ~50% duty cycle has to drive the OSC1 pin while the OSC2 pin is not connected.

Crystal/Ceramic Oscillators

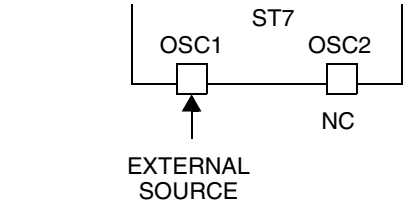
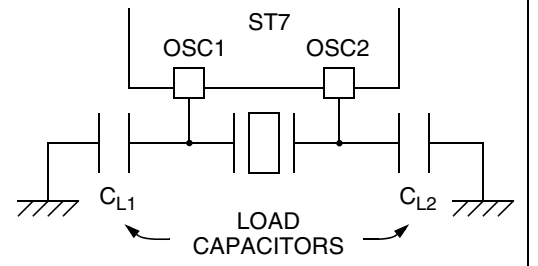
This family of oscillators has the advantage of producing a very accurate rate on the main clock of the ST7. In this mode, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time.

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

See Electrical Characteristics for more details.

Note: When crystal oscillator is used as a clock source, a risk of failure may exist if no series resistors are implemented.

Table 4. ST7 Clock Sources

	Hardware Configuration
External Clock	
Crystal/Ceramic Resonators	

6.2 RESET SEQUENCE MANAGER (RSM)

6.2.1 Introduction

The reset sequence manager includes three RESET sources as shown in Figure 15:

- External $\overline{\text{RESET}}$ source pulse
- Internal LVD RESET (Low Voltage Detection)
- Internal WATCHDOG RESET

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to section 11.2.1 on page 244 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 ST7 memory map.

The basic RESET sequence consists of 3 phases as shown in Figure 14:

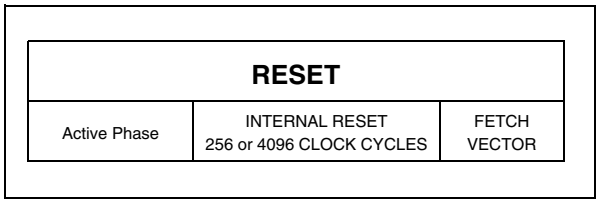
- Active Phase depending on the RESET source
- 256 or 4096 CPU clock cycle delay (selected by option byte)
- RESET vector fetch

Caution: When the ST7 is unprogrammed or fully erased, the Flash is blank and the RESET vector is not programmed. For this reason, it is recommended to keep the RESET pin in low state until programming mode is entered, in order to avoid unwanted behavior.

The 256 or 4096 CPU clock cycle delay allows the oscillator to stabilise and ensures that recovery has taken place from the Reset state. The shorter or longer clock cycle delay should be selected by option byte to correspond to the stabilization time of the external oscillator used in the application.

The RESET vector fetch phase duration is 2 clock cycles.

Figure 14. RESET Sequence Phases

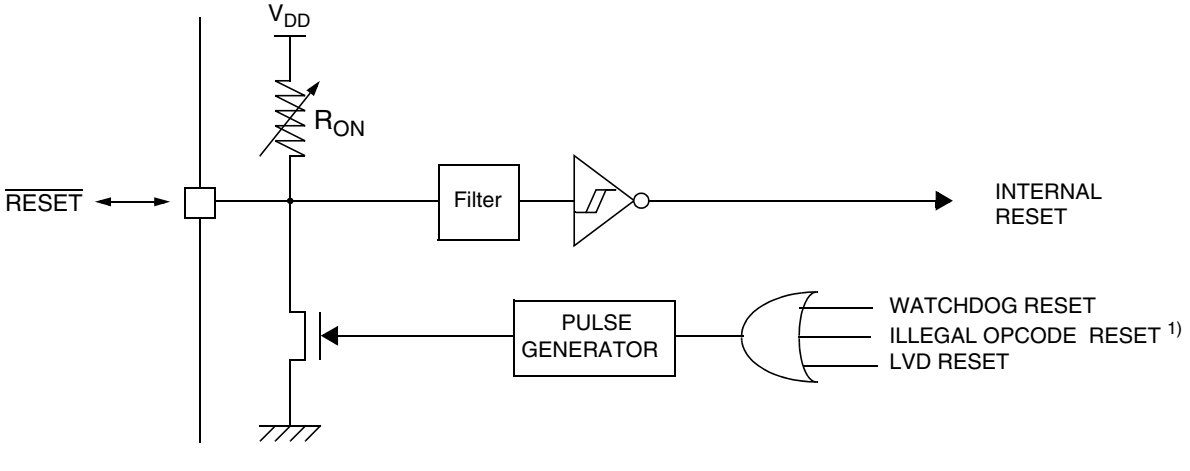


6.2.2 Asynchronous External $\overline{\text{RESET}}$ pin

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device. See Electrical 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 16). This detection is asynchronous and therefore the MCU can enter reset state even in Halt mode.

Figure 15. Reset Block Diagram



Note 1: See “Illegal Opcode Reset” on page 244. 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.

6.2.3 External Power-On RESET

If the LVD is disabled by option byte, to start up the microcontroller correctly, the user must ensure by means of an external reset circuit that the reset signal is held low until V_{DD} is over the minimum level specified for the selected f_{OSC} frequency.

A proper reset signal for a slow rising V_{DD} supply can generally be provided by an external RC network connected to the $\overline{\text{RESET}}$ pin.

6.2.4 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 16.

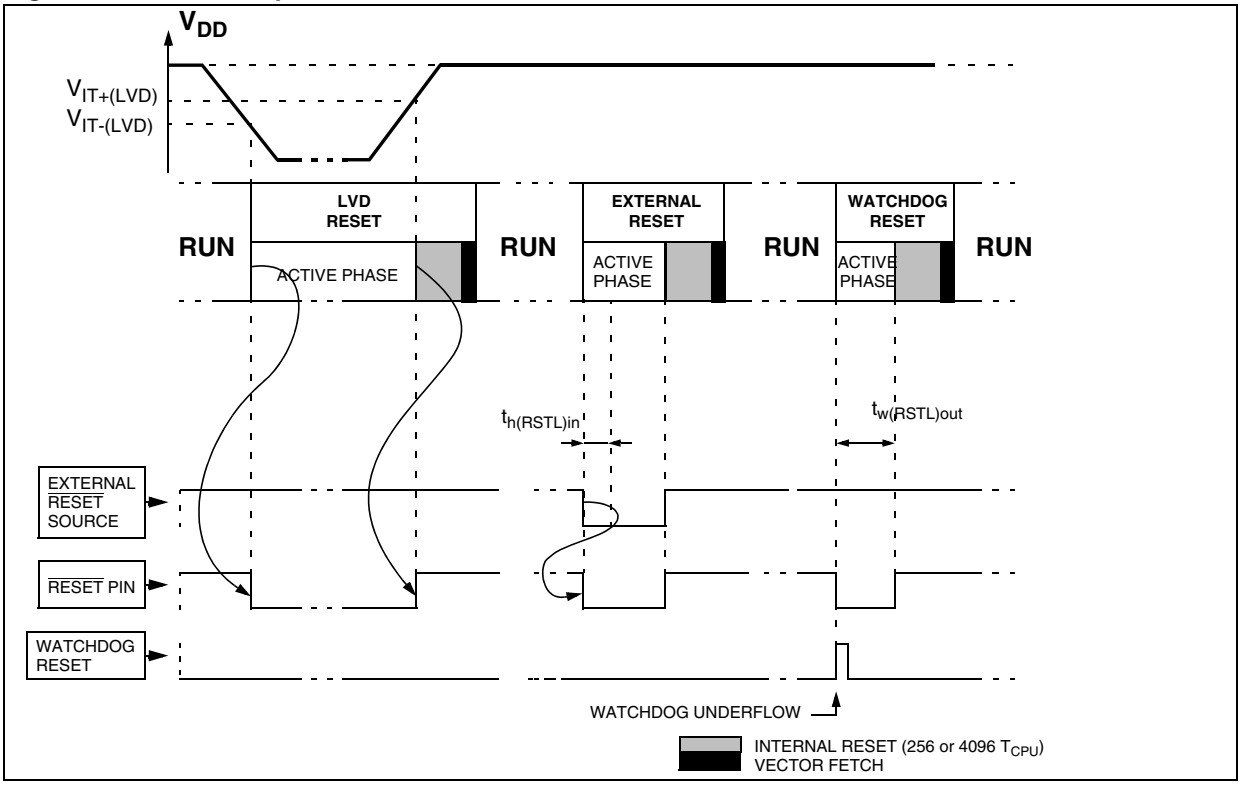
The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

6.2.5 Internal Watchdog RESET

The RESET sequence generated by a internal Watchdog counter overflow is shown in Figure 16.

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 16. RESET Sequences



6.3 SYSTEM INTEGRITY MANAGEMENT (SI)

The System Integrity Management block contains the Low Voltage Detector (LVD), Auxiliary Voltage Detector (AVD) and Clock Security System (CSS) functions. It is managed by the SICSR register.

Note: A reset can also be triggered following the detection of an illegal opcode or prebyte code. Refer to section 11.2.1 on page 244 for further details.

6.3.1 Low Voltage Detector (LVD)

The Low Voltage Detector function (LVD) generates a static reset when the V_{DD} supply voltage is below a V_{IT-} reference value. This means that it secures the power-up as well as the power-down keeping the ST7 in reset.

The V_{IT-} reference value for a voltage drop is lower than the V_{IT+} reference value for power-on in order to avoid a parasitic reset when the MCU starts running and sinks current on the supply (hysteresis).

The LVD Reset circuitry generates a reset when V_{DD} is below:

– V_{IT+} when V_{DD} is rising
– V_{IT-} when V_{DD} is falling

The LVD function is illustrated in Figure 17.

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

– under full software control
– in static safe reset

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

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

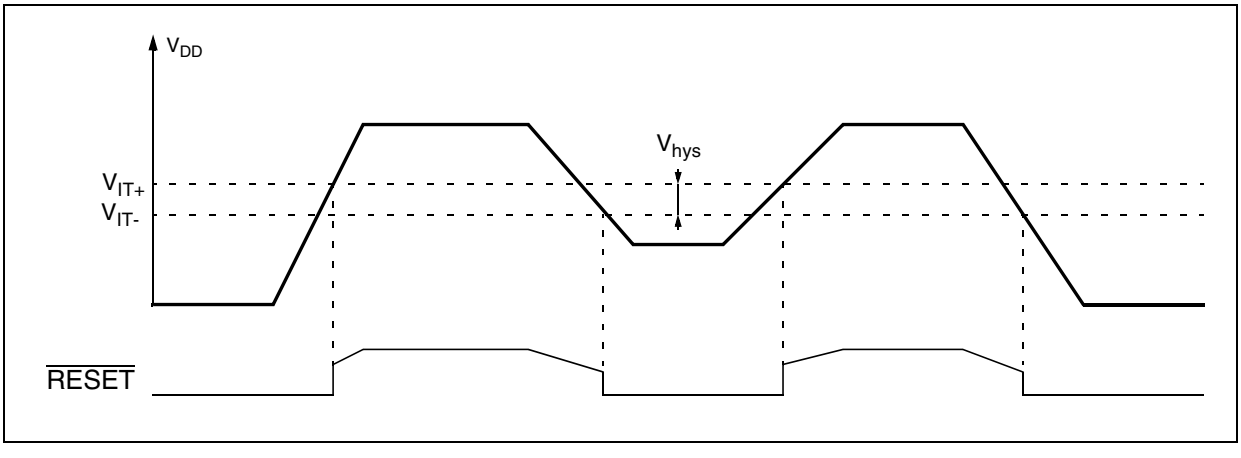
Notes:

The LVD allows the device to be used without any external RESET circuitry.

The LVD is an optional function which can be selected by option byte.

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

Figure 17. Low Voltage Detector vs Reset



SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.3.2 Auxiliary Voltage Detector (AVD)

The Voltage Detector function (AVD) is based on an analog comparison between a $V_{IT-(AVD)}$ and $V_{IT+(AVD)}$ reference value and the V_{DD} main supply. The V_{IT-} reference value for falling voltage is lower than the V_{IT+} reference value for rising voltage in order to avoid parasitic detection (hysteresis).

The output of the AVD comparator is directly readable by the application software through a real-time status bit (AVDF) in the SICSR register. This bit is read only.

Caution: The AVD function is active only if the LVD is enabled through the option byte (see section 14.1 on page 290).

6.3.2.1 Monitoring the V_{DD} Main Supply

If the AVD interrupt is enabled, an interrupt is generated when the voltage crosses the $V_{IT+(AVD)}$ or $V_{IT-(AVD)}$ threshold (AVDF bit toggles).

In the case of a drop in voltage, the AVD interrupt acts as an early warning, allowing software to shut down safely before the LVD resets the microcontroller. See Figure 18.

The interrupt on the rising edge is used to inform the application that the V_{DD} warning state is over.

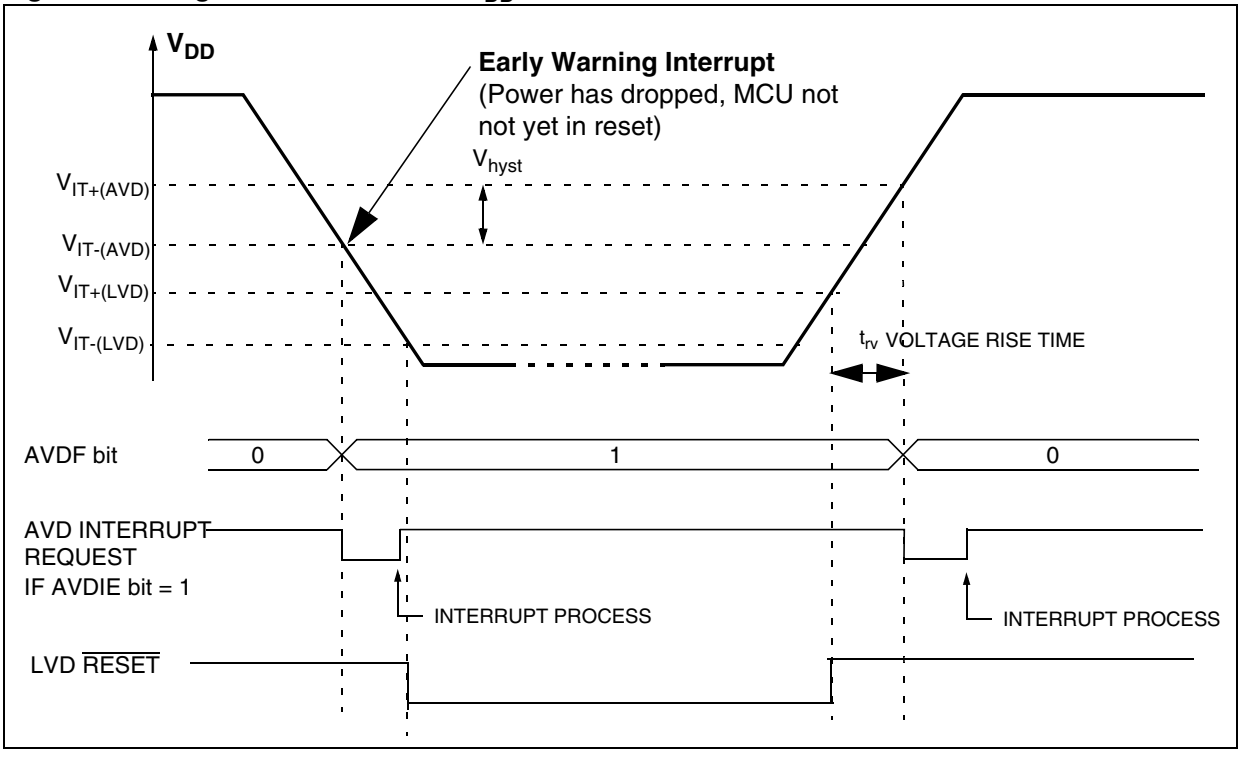
If the voltage rise time t_{rv} is less than 256 or 4096 CPU cycles (depending on the reset delay selected by option byte), no AVD interrupt will be generated when $V_{IT+(AVD)}$ is reached.

If t_{rv} is greater than 256 or 4096 cycles then:

– If the AVD interrupt is enabled before the $V_{IT+(AVD)}$ threshold is reached, then 2 AVD interrupts will be received: the first when the AVDIE bit is set, and the second when the threshold is reached.

– If the AVD interrupt is enabled after the $V_{IT+(AVD)}$ threshold is reached then only one AVD interrupt will occur.

Figure 18. Using the AVD to Monitor V_{DD}



SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.3.3 Clock Security System (CSS)

The Clock Security System (CSS) protects the ST7 against main clock problems. To allow the integration of the security features in the applications, it is based on a PLL which can provide a backup clock. The PLL can be enabled or disabled by option byte or by software. It requires an 8-MHz input clock and provides a 16-MHz output clock.

6.3.3.1 Safe Oscillator Control

The safe oscillator of the CSS block is made of a PLL.

If the clock signal disappears (due to a broken or disconnected resonator...) the PLL continues to provide a lower frequency, which allows the ST7 to perform some rescue operations.

Note: The clock signal must be present at start-up. Otherwise, the ST7MC will not start and will be maintained in RESET conditions.

6.3.3.2 Limitation detection

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

6.3.4 Low Power Modes

Mode	Description
Wait	No effect on SI. CSS and AVD interrupts cause the device to exit from Wait mode.
Halt	The CRSR register is frozen. The CSS (including the safe oscillator) is disabled until Halt mode is exited. The previous CSS configuration resumes when the MCU is woken up by an interrupt with "exit from Halt mode" capability or from the counter reset value when the MCU is woken up by a RESET. The AVD remains active, and an AVD interrupt can be used to exit from Halt mode.

6.3.4.1 Interrupts

The CSS or AVD interrupt events generate an interrupt if the corresponding Enable Control Bit (CSSIE or AVDIE) is set and the interrupt mask in the CC register is reset (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
CSS event detection (safe oscillator activated as main clock)	CSSD	CSSIE	Yes	No [1)]
AVD event	AVDF	AVDIE	Yes	Yes

Note 1: This interrupt allows to exit from Active-halt mode.

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

6.3.5 Register Description

SYSTEM INTEGRITY (SI) CONTROL/STATUS REGISTER (SICSR, page 0)

Read/Write

Reset Value: 000x 000x (00h)

7							0
PAGE	AVDIE	AVDF	LVDRF	0	CSSIE	CSSD	WDGRF

Bit 7 = **PAGE** *SICSR Register Page Selection*
This bit selects the SICSR register page. It is set and cleared by software
0: Access to SICSR register mapped in page 0.
1: Access to SICSR register mapped in page 1.

Bit 6 = **AVDIE** *Voltage Detector interrupt enable*
This bit is set and cleared by software. It enables an interrupt to be generated when the AVDF flag changes (toggles). The pending interrupt information is automatically cleared when software enters the AVD interrupt routine.
0: AVD interrupt disabled
1: AVD interrupt enabled

Bit 5 = **AVDF** *Voltage Detector flag*
This read-only bit is set and cleared by hardware. If the VDIE bit is set, an interrupt request is generated when the AVDF bit changes value.
0: V_{DD} over $V_{IT+ (AVD)}$ threshold
1: V_{DD} under $V_{IT-(AVD)}$ threshold

Bit 4 = **LVDRF** *LVD reset flag*
This bit indicates that the last Reset was generated by the LVD block. It is set by hardware (LVD reset) and cleared by software (writing zero). See WDGRF flag description for more details. When the LVD is disabled by OPTION BYTE, the LVDRF bit value is undefined.

Bit 3 = Reserved, must be kept cleared.

Bit 2 = **CSSIE** *Clock security syst interrupt enable*
This bit enables the interrupt when a disturbance is detected by the Clock Security System (CSSD bit set). It is set and cleared by software.
0: Clock security system interrupt disabled
1: Clock security system interrupt enabled
When the PLL is disabled (PLLEN=0), the CSSIE bit has no effect.

Bit 1 = **CSSD** *Clock security system detection*
This bit indicates a disturbance on the main clock signal (f_{OSC}): the clock stops (at least for a few cycles). It is set by hardware and cleared by reading the SICSR register when the original oscillator recovers.
0: Safe oscillator is not active
1: Safe oscillator has been activated
When the PLL is disabled (PLLEN=0), the CSSD bit value is forced to 0.

Bit 0 = **WDGRF** *Watchdog reset flag*
This bit indicates that the last Reset was generated by the Watchdog peripheral. It is set by hardware (watchdog reset) and cleared by software (writing zero) or an LVD Reset (to ensure a stable cleared state of the WDGRF flag when CPU starts).
Combined with the LVDRF flag information, the flag description is given by the following table.

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

Application notes

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

SYSTEM INTEGRITY MANAGEMENT (Cont'd)

SYSTEM INTEGRITY (SI) CONTROL/STATUS REGISTER (SICSR, page 1)

Reset Value: 00000000 (00h)

7							0
PAGE	0	VCOEN	LOCK	PLLEN	0	CKSEL	0

Bit 7 = **PAGE** *SICSR Register Page Selection*
This bit selects the SICSR register page. It is set and cleared by software
0: Access to SICSR register mapped in page 0.
1: Access to SICSR register mapped in page 1.

Bit 6 = Reserved, must be kept cleared.

Bit 5 = **VCOEN** *VCO Enable*
This bit is set and cleared by software.
0: VCO (Voltage Controlled Oscillator) connected to the output of the PLL charge pump (default mode), to obtain a 16-MHz output frequency (with an 8-MHz input frequency).
1: VCO tied to ground in order to obtain a 10-MHz frequency (f_{vco})

Notes:
1. During ICC session, this bit is set to 1 in order to have an internal frequency which does not depend on the input clock. Then, it can be reset in order to run faster with an external oscillator.

Bit 4 = **LOCK** *PLL Locked*
This bit is read only. It is set by hardware. It is set automatically when the PLL reaches its operating frequency.
0: PLL not locked
1: PLL locked

Bit 3 = **PLLEN** *PLL Enable*
This bit enables the PLL and the clock detector. It is set and cleared by software.
0: PLL and Clock Detector (CKD) disabled
1: PLL and Clock Detector (CKD) enabled

Notes:
1. During ICC session, this bit is set to 1.
2. PLL cannot be disabled if PLL clock source is selected (CKSEL= 1).

Bit 2 = Reserved, must be kept cleared.

Bit 1 = **CKSEL** *Clock Source Selection*
This bit selects the clock source: oscillator clock or clock from the PLL. It is set and cleared by software. It can also be set by option byte (PLL opt)
0: Oscillator clock selected
1: PLL clock selected

Notes:
1. During ICC session, this bit is set to 1. Then, CKSEL can be reset in order to run with f_{OSC}.
2. Clock from the PLL cannot be selected if the PLL is disabled (PLLEN =0)
3. If the clock source is selected by PLL option bit, CKSEL bit selection has no effect.

Bit 0 = Reserved, must be kept cleared.

6.4 MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK AND BEEPER (MCC/RTC)

The Main Clock Controller consists of three different functions:

- a programmable CPU clock prescaler
- a clock-out signal to supply external devices
- a real-time clock timer with interrupt capability

Each function can be used independently and simultaneously.

6.4.1 Programmable CPU Clock Prescaler

The programmable CPU clock prescaler supplies the clock for the ST7 CPU and its internal peripherals. It manages Slow power saving mode (See Section 8.2 SLOW MODE for more details).

The prescaler selects the f_{CPU} main clock frequency and is controlled by three bits in the MCCSR register: CP[1:0] and SMS.

6.4.2 Clock-out Capability

The clock-out capability is an alternate function of an I/O port pin that outputs a f_{OSC2} clock to drive external devices. It is controlled by the MCO bit in the MCCSR register.

CAUTION: When selected, the clock out pin suspends the clock during Active-halt mode.

6.4.3 Real-time Clock Timer (RTC)

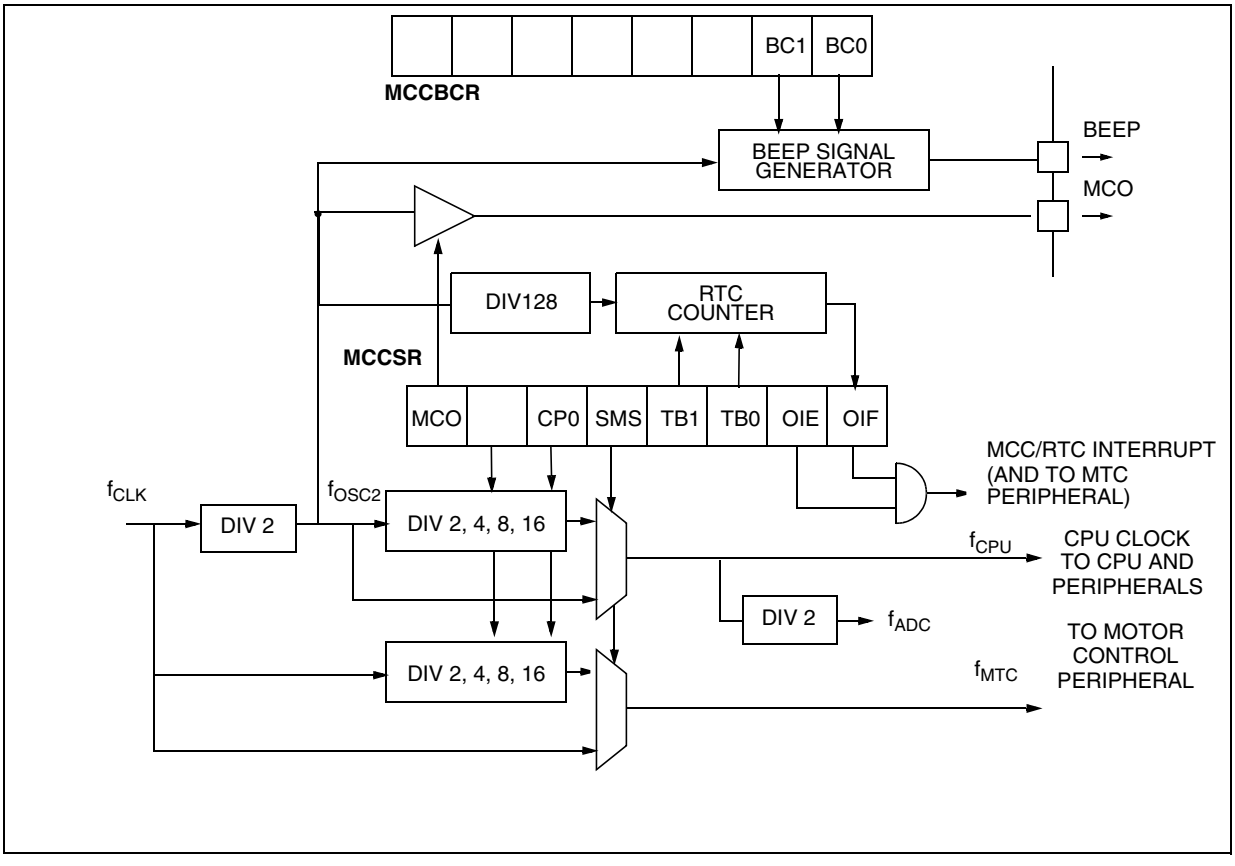
The counter of the real-time clock timer allows an interrupt to be generated based on an accurate real-time clock. Four different time bases depending directly on f_{OSC2} are available. The whole functionality is controlled by four bits of the MCCSR register: TB[1:0], OIE and OIF.

When the RTC interrupt is enabled (OIE bit set), the ST7 enters Active-halt mode when the HALT instruction is executed. See Section 8.4 ACTIVE-HALT AND HALT MODES for more details.

6.4.4 Beeper

The beep function is controlled by the MCCBCR register. It can output three selectable frequencies on the BEEP pin (I/O port alternate function).

Figure 19. Main Clock Controller (MCC/RTC) Block Diagram



MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK (Cont'd)

6.4.5 Low Power Modes

Mode	Description
Wait	No effect on MCC/RTC peripheral. MCC/RTC interrupt cause the device to exit from Wait mode.
Active-halt	No effect on MCC/RTC counter (OIE bit is set), the registers are frozen. MCC/RTC interrupt cause the device to exit from Active-halt mode.
Halt	MCC/RTC counter and registers are frozen. MCC/RTC operation resumes when the MCU is woken up by an interrupt with "exit from Halt" capability.

6.4.6 Interrupts

The MCC/RTC interrupt event generates an interrupt if the OIE bit of the MCCSR register is set and the interrupt mask in the CC register is not active (RIM instruction).

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Time base overflow event	OIF	OIE	Yes	No [1]

Note:

The MCC/RTC interrupt wakes up the MCU from Active-halt mode, not from Halt mode.

6.4.7 Register Description

MCC CONTROL/STATUS REGISTER (MCCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
MCO	CP1	CP0	SMS	TB1	TB0	OIE	OIF

Bit 7 = **MCO** *Main clock out selection*
This bit enables the MCO alternate function on the PF0 I/O port. 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_{OSC2} on I/O port)

Note: To reduce power consumption, the MCO function is not active in Active-halt mode.

Bit 6:5 = **CP[1:0]** *CPU clock prescaler*
These bits select the CPU clock prescaler which is applied in the different slow modes. Their action is conditioned by the setting of the SMS bit. These two bits are set and cleared by software

f_{CPU} in Slow mode	CP1	CP0
f_{OSC2} / 2	0	0
f_{OSC2} / 4	0	1
f_{OSC2} / 8	1	0
f_{OSC2} / 16	1	1

Bit 4 = **SMS** *Slow mode select*
This bit is set and cleared by software.
0: Normal mode. $f_{CPU} = f_{OSC2}$
1: Slow mode. f_{CPU} is given by CP1, CP0
See Section 8.2 SLOW MODE and Section 6.4 MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK AND BEEPER (MCC/RTC) for more details.

Bit 3:2 = **TB[1:0]** *Time base control*

These bits select the programmable divider time base. They are set and cleared by software.

Counter Prescaler	Time Base		TB1	TB0
	f_{OSC2} =4MHz	f_{OSC2}=8MHz		
16000	4ms	2ms	0	0
32000	8ms	4ms	0	1
80000	20ms	10ms	1	0
200000	50ms	25ms	1	1

A modification of the time base is taken into account at the end of the current period (previously set) to avoid an unwanted time shift. This allows to use this time base as a real-time clock.

Bit 1 = **OIE** *Oscillator interrupt enable*
This bit set and cleared by software.
0: Oscillator interrupt disabled
1: Oscillator interrupt enabled
This interrupt can be used to exit from Active-halt mode.
When this bit is set, calling the ST7 software HALT instruction enters the Active-halt power saving mode.

MAIN CLOCK CONTROLLER WITH REAL-TIME CLOCK (Cont'd)

Bit 0 = **OIF** *Oscillator interrupt flag*
This bit is set by hardware and cleared by software reading the CSR register. It indicates when set that the main oscillator has reached the selected elapsed time (TB1:0).
0: Timeout not reached
1: Timeout reached

CAUTION: The BRES and BSET instructions must not be used on the MCCSR register to avoid unintentionally clearing the OIF bit.

MCC BEEP CONTROL REGISTER (MCCBCR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	AD-STS	ADC IE	BC1	BC0

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

Bit 3 = **ADSTS** *A/D Converter Sample Time Stretch*
This bit is set and cleared by software to enable or disable the A/D Converter sample time stretch feature.
0: AD sample time stretch disabled (for standard impedance analog inputs)
1 AD sample time stretch enabled (for high impedance analog inputs)

Bit 2 = **ADCIE** *A/D Converter Interrupt Enable*
This bit is set and cleared by software to enable or disable the A/D Converter interrupt.
0: AD Interrupt disabled

1 AD Interrupt enabled

Bit 1:0 = **BC[1:0]** *Beep control*
These 2 bits select the PF1 pin beep capability.

BC1	BC0	Beep mode with f_{OSC2}=8MHz	
0	0	Off	
0	1	~2-KHz	Output Beep signal ~50% duty cycle
1	0	~1-KHz	
1	1	~500-Hz	

The beep output signal is available in Active-halt mode but has to be disabled to reduce the consumption.

Table 5. Main Clock Controller Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0040h	**SICSR, page0** Reset Value	PAGE 0	VDIE 0	VDF 0	LVDRF x	0	CFIE 0	CSSD 0	WDGRF x
0040h	**SICSR, page1** Reset Value	PAGE 0	0	VCOEN 0	LOCK x	PLLEN 0	0	CKSEL 0	0
002Ch	**MCCSR** Reset Value	MCO 0	CP1 0	CP0 0	SMS 0	TB1 0	TB0 0	OIE 0	OIF 0
002Dh	**MCCBCR** Reset Value	0	0	0	0	ADSTS 0	ADCIE 0	BC1 0	BC0 0

7 INTERRUPTS

7.1 INTRODUCTION

The ST7 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
  - 2 non maskable events: RESET, TRAP
  - 1 maskable top level event: MCES

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) ST7 interrupt controller.

7.2 MASKING AND PROCESSING FLOW

The interrupt masking is managed by the I1 and I0 bits of the CC register and the ISPRx registers which give the interrupt software priority level of each interrupt vector (see Table 6). The processing flow is shown in Figure 20

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).

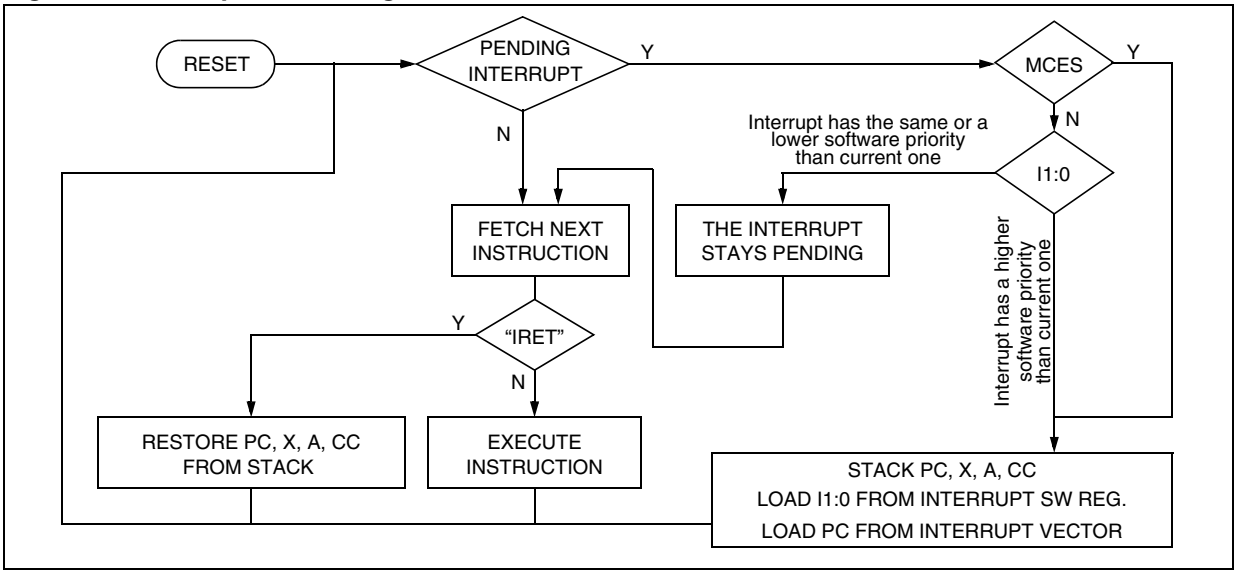
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

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 20. Interrupt Processing Flowchart



INTERRUPTS (Cont'd)

Servicing Pending Interrupts

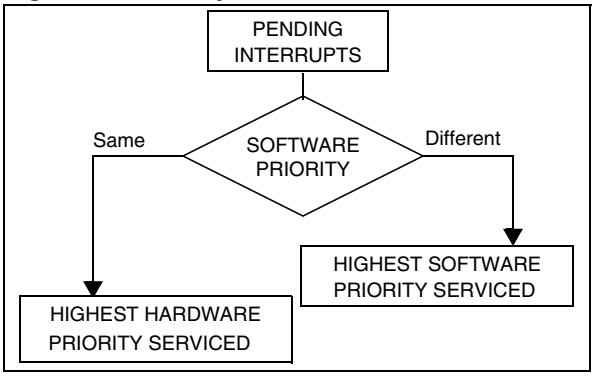
As several interrupts can be pending at the same time, the interrupt to be taken into account is determined by the following two-step process:

– 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 21 describes this decision process.

Figure 21. 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 MCES 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 ST7 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 20). 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.

- 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 20 as a MCES top level interrupt.

- RESET

The RESET source has the highest priority in the ST7. 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.

- MCES (MTC Emergency Stop)

This hardware interrupt occurs when a specific edge is detected on the dedicated $\overline{\text{MCES}}$ pin or when an error is detected by the micro in the motor speed measurement. The interrupt request is maintained as long as the $\overline{\text{MCES}}$ pin is low if the interrupt is enabled by the EIM bit in the MIMR register.

- 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 ORed.

- Peripheral Interrupts

Usually the peripheral interrupts cause the MCU to exit from Halt mode except those mentioned in the "Interrupt Mapping" table.
A peripheral interrupt occurs when a specific flag is set in the peripheral status registers and if the corresponding enable bit is set in the peripheral control register.
The general sequence for clearing an interrupt is based on an access to the status register followed by a read or write to an associated register.
Note: The clearing sequence resets the internal latch. A pending interrupt (i.e. waiting for being serviced) will therefore be lost if the clear sequence is executed.

INTERRUPTS (Cont'd)

7.3 INTERRUPTS AND LOW POWER MODES

All interrupts allow the processor to exit the Wait low power mode. On the contrary, only external and other specified interrupts allow the processor to exit from the Halt modes (see column "Exit from Halt" in "Interrupt Mapping" table). When several pending interrupts are present while exiting Halt mode, the first one serviced can only be an interrupt with exit from Halt mode capability and it is selected through the same decision process shown in Figure 21.

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

7.4 CONCURRENT & NESTED MANAGEMENT

The following Figure 22 and Figure 23 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 23. The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0, MCES. The software priority is given for each interrupt.

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

Figure 22. Concurrent Interrupt Management

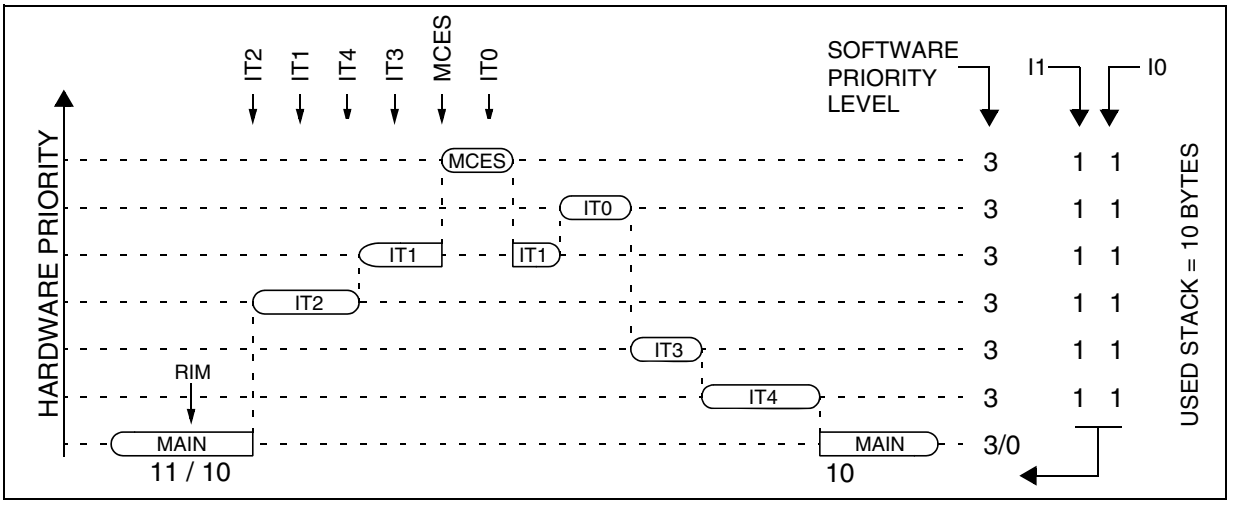
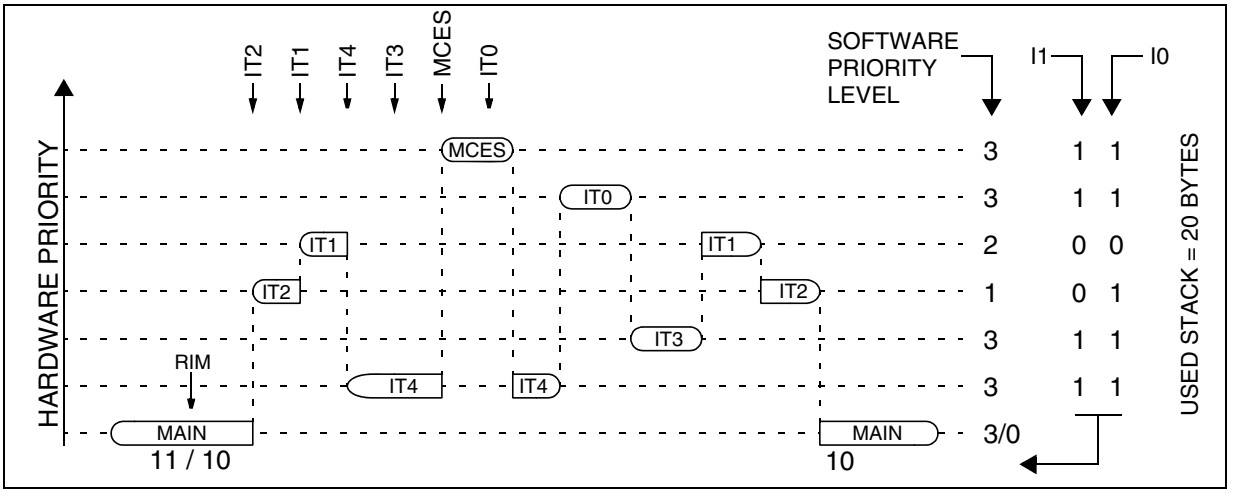


Figure 23. Nested Interrupt Management



INTERRUPTS (Cont'd)

7.5 INTERRUPT REGISTER DESCRIPTION

CPU CC REGISTER INTERRUPT BITS

Read/Write

Reset Value: 111x 1010 (xAh)

7							0
1	1	**I1**	H	**I0**	N	Z	C

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

These two bits indicate the current interrupt software priority.

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

These two bits are set/cleared by hardware when entering in interrupt. The loaded value is given by the corresponding bits in the interrupt software priority registers (ISPRx).

They can be also set/cleared by software with the RIM, SIM, HALT, WFI, IRET and PUSH/POP instructions (see "Interrupt Dedicated Instruction Set" table).

***Note**: MCES, 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 correspondence 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 MCES 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 MCES 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 (level 3)	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.

Table 8. Interrupt Mapping

N°	Source Block	Description	Register Label	Priority Order	Exit from Halt[1)]	Address Vector
	RESET	Reset	N/A	Highest Priority ↓ Lowest Priority	yes	FFFEh-FFFFh
	TRAP	Software interrupt			no	FFFCh-FFFDh
0	MCES	Motor Control Emergency Stop or Speed error interrupt	MISR MCRC		no	FFFAh-FFFBh
1	MCC/RTC CSS	Main clock controller time base interrupt Safe oscillator activation interrupt	MCCSR SICSR		yes	FFF8h-FFF9h
2	ei0	External interrupt port	N/A		yes	FFF6h-FFF7h
3	ei1	External interrupt port			yes	FFF4h-FFF5h
4	ei2	External interrupt port			yes	FFF2h-FFF3h
5	MTC	Event U or Current Loop or Sampling Out	MISR/MCONF		no	FFF0h-FFF1h
6		Event R or Event Z	MISR		no	FFEEh-FFEFh
7		Event C or Event D			no	FFECh-FFEDh
8	SPI	SPI peripheral interrupts	SPICSR		yes	FFEAh-FFEBh
9	TIMER A	TIMER A peripheral interrupts	TASR		no	FFE8h-FFE9h
10	TIMER B	TIMER B peripheral interrupts	TBSR		no	FFE6h-FFE7h
11	LINSCI™	LINSCI™ Peripheral interrupts	SCISR		no	FFE4h-FFE5h
12	AVD/ ADC	Auxiliary Voltage detector interrupt ADC End of conversion interrupt	SICSR ADCSR		yes	FFE2h-FFE3h
13	PWM ART	PWM ART overflow interrupt PWM ART input capture interrupts	ARTCSR ARTICCSR		no	FFE0h-FFE1h

Note 1. Valid for Halt and Active-halt modes except for the MCC/RTC or CSS interrupt source which exits from Active-halt mode only.

INTERRUPTS (Cont'd)

7.6 EXTERNAL INTERRUPTS

The pending interrupts are cleared writing a different value in the ISx[1:0], IPA or IPB bits of the EICR.

Note: External interrupts are masked when an I/O (configured as input interrupt) of the same interrupt vector is forced to V_{SS}.

7.6.1 I/O PORT INTERRUPT SENSITIVITY

The external interrupt sensitivity is controlled by the IPA, IPB and ISxx bits of the EICR register (Figure 24). This control allows to have up to 4 fully independent external interrupt source sensitivities.

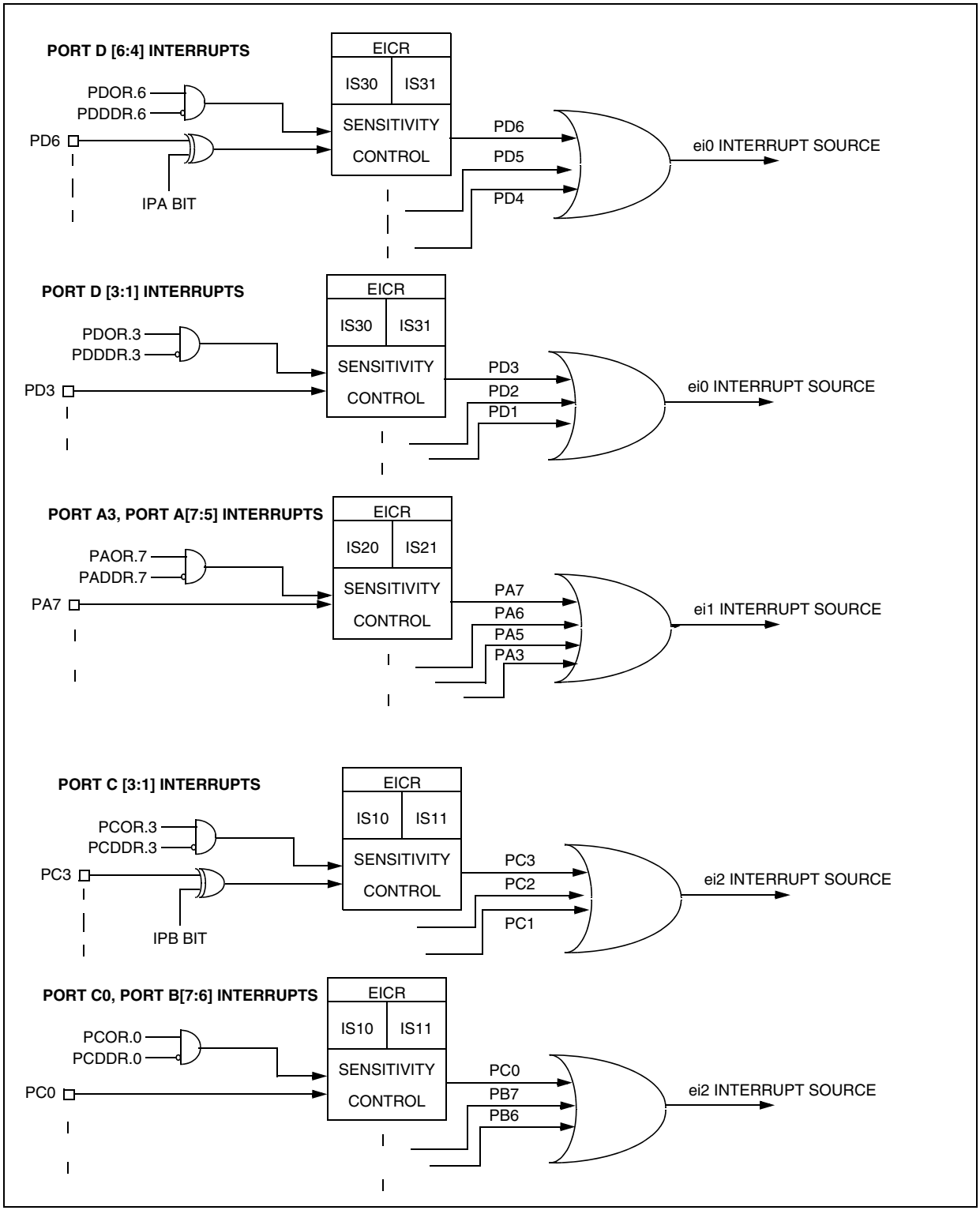
Each external interrupt source can be generated on four (or five) different events on the pin:

- Falling edge
- Rising edge
- Falling and rising edge
- Falling edge and low level
- Rising edge and high level (only for ei0 and ei2)

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).

INTERRUPTS (Cont'd)

Figure 24. External Interrupt Control bits



INTERRUPTS (Cont'd)

7.7 EXTERNAL INTERRUPT CONTROL REGISTER (EICR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
IS11	IS10	IPB	IS21	IS20	IS31	IS30	IPA

Bit 7:6 = **IS1[1:0]** *ei2 sensitivity*
The interrupt sensitivity, defined using the IS1[1:0] bits, is applied to the following external interrupts:
- ei2 (port C3..1)

IS11	IS10	External Interrupt Sensitivity	
		IPB bit =0	IPB bit =1
0	0	Falling edge & low level	Rising edge & high level
0	1	Rising edge only	Falling edge only
1	0	Falling edge only	Rising edge only
1	1	Rising and falling edge	

- ei2 (port C0, B7..6)

IS11	IS10	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).

Bit 5 = **IPB** *Interrupt polarity for port C*

This bit is used to invert the sensitivity of the port C[3:1] external interrupts. It can be set and cleared by software only when I1 and I0 of the CC register are both set to 1 (level 3).
0: No sensitivity inversion
1: Sensitivity inversion

Bit 4:3= **IS2[1:0]** *ei1sensitivity*
The interrupt sensitivity, defined using the IS2[1:0] bits, is applied to the following external interrupts:
- ei1 (port A3, A5...A7)

IS21	IS20	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

Bit 2:1= **IS3[1:0]** *ei0sensitivity*
The interrupt sensitivity, defined using the IS2[1:0] bits, is applied to the following external interrupts:

EXTERNAL INTERRUPT CONTROL REGISTER (EICR) (Cont'd)

- ei0 (port D6..4)

IS31	IS30	External Interrupt Sensitivity	
		IPA bit =0	IPA bit =1
0	0	Falling edge & low level	Rising edge & high level
0	1	Rising edge only	Falling edge only
1	0	Falling edge only	Rising edge only
1	1	Rising and falling edge	

- ei0 (port D3..1)

IS31	IS30	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).

Bit 0 = **IPA** *Interrupt polarity for port D*
This bit is used to invert the sensitivity of the port D [6:4] external interrupts. It can be set and cleared by software only when I1 and I0 of the CC register are both set to 1 (level 3).
0: No sensitivity inversion
1: Sensitivity inversion

INTERRUPTS (Cont'd)

Table 9. Nested Interrupts Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0024h		ei1		ei0		MCC + SI		MCES	
	ISPR0	I1_3	I0_3	I1_2	I0_2	I1_1	I0_1		
	Reset Value	1	1	1	1	1	1	1	1
0025h		MTC C/D		MTC R/Z		MTC U/CL		ei2	
	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
0026h		SCI		TIMER B		TIMER A		SPI	
	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
0027h						PWMART		AVD	
	ISPR3	I1_15	I0_15	I1_14	I0_14	I1_13	I0_13	I1_12	I0_12
	Reset Value	1	1	1	1	1	1	1	1
0028h	**EICR**	IS11	IS10	IPB	IS21	IS20	IPA	0	0
	Reset Value	0	0	0	0	0	0		

8 POWER SAVING MODES

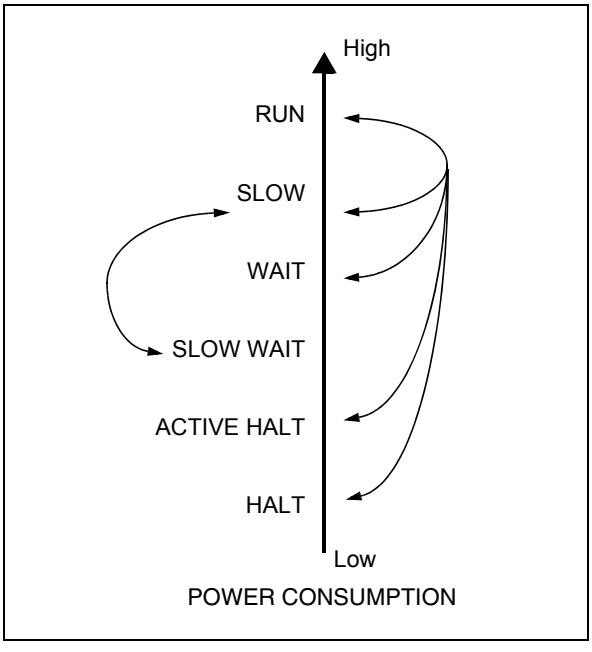
8.1 INTRODUCTION

To give a large measure of flexibility to the application in terms of power consumption, four main power saving modes are implemented in the ST7 (see Figure 25): Slow, Wait (Slow-wait), Active-halt and Halt.

After a RESET the normal operating mode is selected by default (Run mode). This mode drives the device (CPU and embedded peripherals) by means of a master clock which is based on the main oscillator frequency divided or multiplied by 2 (f_{OSC2}).

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

Figure 25. Power Saving Mode Transitions



8.2 SLOW MODE

This mode has two targets:

– To reduce power consumption by decreasing the internal clock in the device,

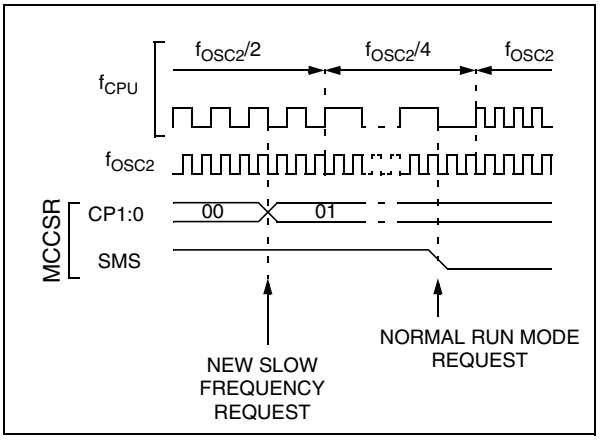
– To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

Slow mode is controlled by three bits in the MCCSR register: the SMS bit which enables or disables Slow mode and two CPx bits which select the internal slow frequency (f_{CPU}).

In this mode, the master clock frequency (f_{OSC2}) can be divided by 2, 4, 8 or 16. The CPU and peripherals are clocked at this lower frequency (f_{CPU}).

Note: Slow-wait mode is activated when entering the Wait mode while the device is already in Slow mode.

Figure 26. Slow Mode Clock Transitions



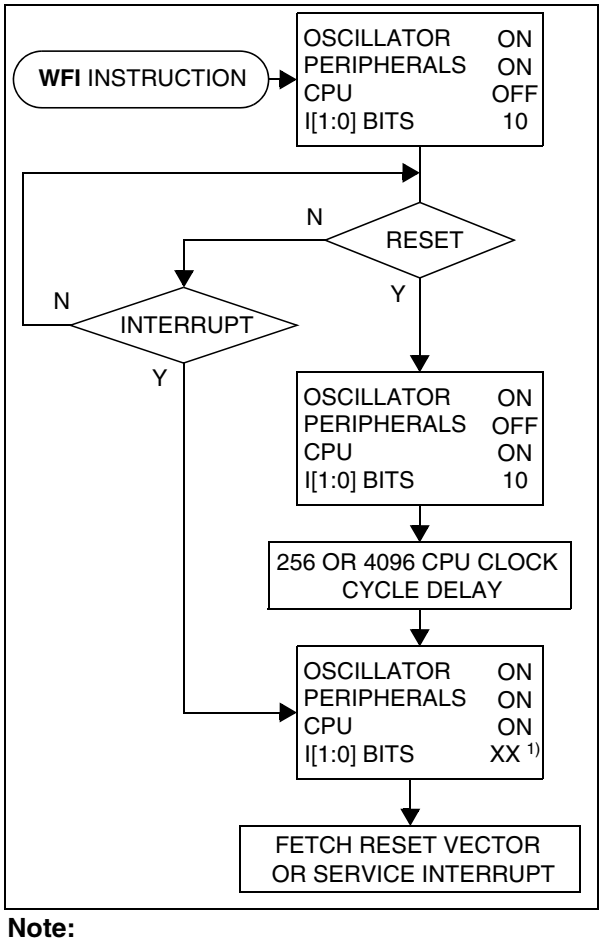
POWER SAVING MODES (Cont'd)

8.3 WAIT MODE

Wait mode places the MCU in a low power consumption mode by stopping the CPU.
This power saving mode is selected by calling the 'WFI' instruction.
All peripherals remain active. During Wait mode, the I[1:0] bits of the CC register are forced to '10', to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in Wait mode until an interrupt or RESET occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine.
The MCU will remain in Wait mode until a Reset or an Interrupt occurs, causing it to wake up.

Refer to Figure 27.

Figure 27. Wait Mode Flow-chart



Note:

1. 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)

8.4 ACTIVE-HALT AND HALT MODES

Active-halt and Halt modes are the two lowest power consumption modes of the MCU. They are both entered by executing the 'HALT' instruction. The decision to enter either in Active-halt or Halt mode is given by the MCC/RTC interrupt enable flag (OIE bit in MCCSR register).

MCCSR OIE bit	Power Saving Mode entered when HALT instruction is executed
0	Halt mode
1	Active-halt mode

8.4.1 ACTIVE-HALT MODE

Active-halt mode is the lowest power consumption mode of the MCU with a real-time clock available. It is entered by executing the 'HALT' instruction when the OIE bit of the Main Clock Controller Status register (MCCSR) is set (see section 6.4 on page 37 for more details on the MCCSR register).

The MCU can exit Active-halt mode on reception of either an MCC/RTC interrupt, a specific interrupt (see Table 8, "Interrupt Mapping," on page 44) or a RESET. When exiting Active-halt mode by means of an interrupt, no 256 or 4096 CPU cycle delay occurs. The CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 29).

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

In Active-halt mode, only the main oscillator and its associated counter (MCC/RTC) are running to keep a wake-up time base. All other peripherals are not clocked except those which get their clock supply from another clock generator (such as external or auxiliary oscillator).

The safeguard against staying locked in Active-halt mode is provided by the oscillator interrupt.

Note: As soon as the interrupt capability of one of the oscillators is selected (MCCSR.OIE bit set), entering Active-halt mode while the Watchdog is active does not generate a RESET.
This means that the device cannot spend more than a defined delay in this power saving mode.

Figure 28. Active-halt Timing Overview

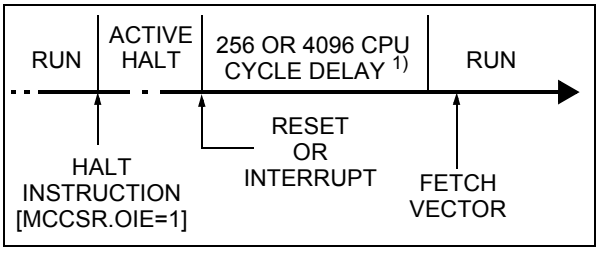
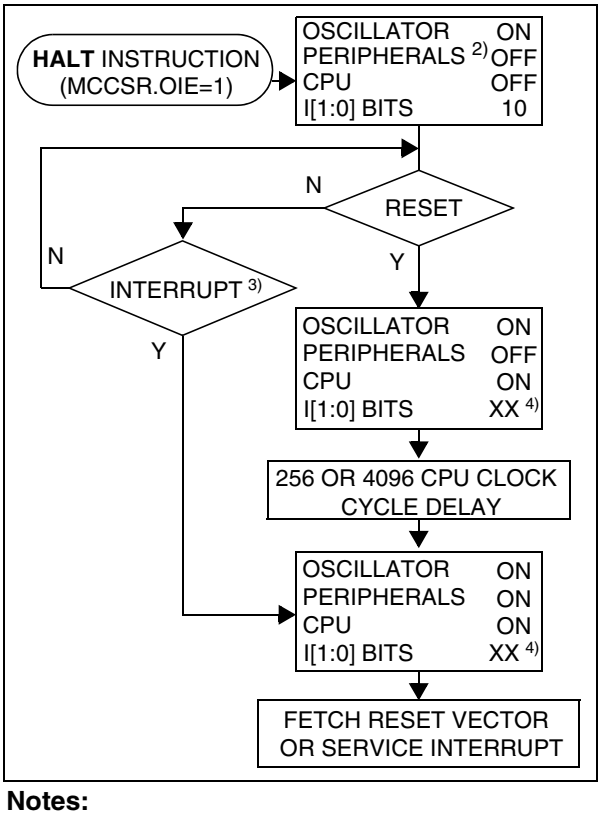


Figure 29. Active-halt Mode Flow-chart



Notes:

1. This delay occurs only if the MCU exits Active-halt mode by means of a RESET.

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

3. Only the MCC/RTC interrupt and some specific interrupts can exit the MCU from Active-halt mode (such as external interrupt). Refer to Table 8, "Interrupt Mapping," on page 44 for more details.

4. 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 restored when the CC register is popped.

POWER SAVING MODES (Cont'd)

8.4.2 HALT MODE

The Halt mode is the lowest power consumption mode of the MCU. It is entered by executing the 'HALT' instruction when the OIE bit of the Main Clock Controller Status register (MCCSR) is cleared (see section 6.4 on page 37 for more details on the MCCSR register).

The MCU can exit Halt mode on reception of either a specific interrupt (see Table 8, "Interrupt Mapping," on page 44) or a RESET. When exiting Halt mode by means of a RESET or an interrupt, the oscillator is immediately turned on and the 256 or 4096 CPU cycle delay is used to stabilize the oscillator. After the start up delay, the CPU resumes operation by servicing the interrupt or by fetching the reset vector which woke it up (see Figure 31).

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

In Halt mode, the main oscillator is turned off causing all internal processing to be stopped, including the operation of the on-chip peripherals. All peripherals are not clocked except the ones which get their clock supply from another clock generator (such as an external or auxiliary oscillator).

The compatibility of Watchdog operation with Halt mode is configured by the "WDGHALT" option bit of the option byte. The HALT instruction when executed while the Watchdog system is enabled, can generate a Watchdog RESET (see section 14.1 on page 290 for more details).

Figure 30. Halt Timing Overview

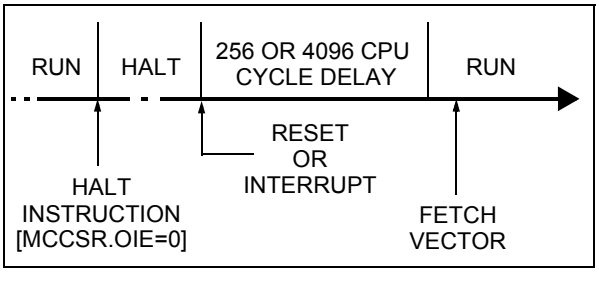
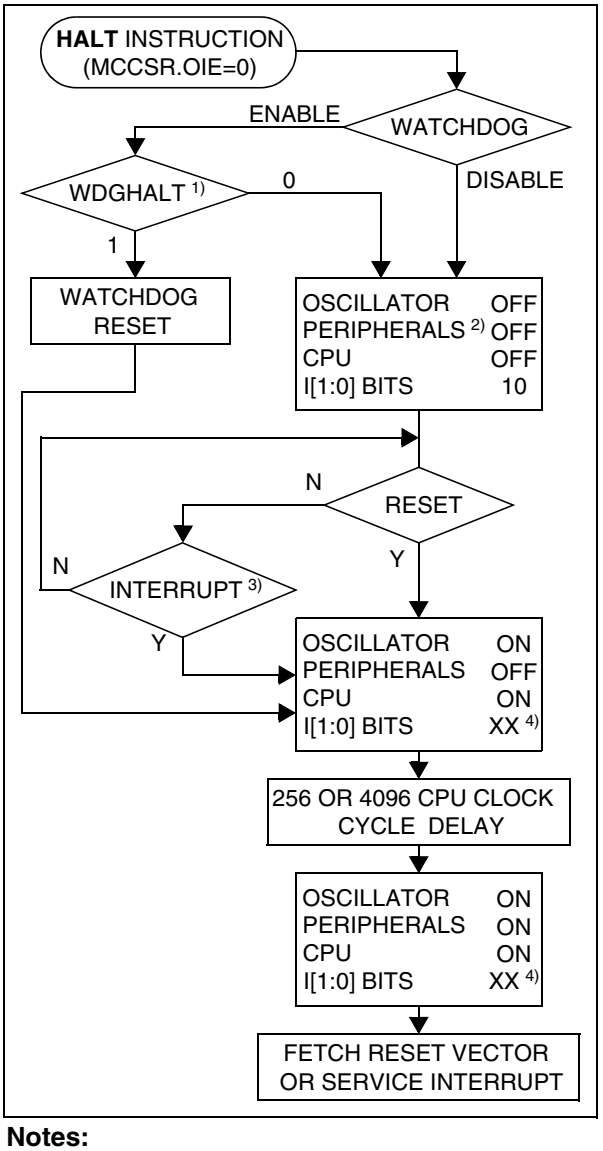


Figure 31. Halt Mode Flow-chart



Notes:

1. WDGHALT is an option bit. See option byte section for more details.

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

3. Only some specific interrupts can exit the MCU from Halt mode (such as external interrupt). Refer to Table 8, "Interrupt Mapping," on page 44 for more details.

4. 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.

9 I/O PORTS

9.1 INTRODUCTION

The I/O ports offer different functional modes:
– transfer of data through digital inputs and outputs

and for specific pins:
– external interrupt generation
– alternate signal input/output for the on-chip peripherals.

An I/O port contains up to 8 pins. Each pin can be programmed independently as digital input (with or without interrupt generation) or digital output.

9.2 FUNCTIONAL DESCRIPTION

Each port has two main registers:

– Data Register (DR)

– Data Direction Register (DDR)

and one optional register:

– Option Register (OR)

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

The following description takes into account the OR register, (for specific ports which do not provide this register refer to the I/O Port Implementation section). The generic I/O block diagram is shown in Figure 32

9.2.1 Input Modes

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

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

Different input modes can be selected by software through the OR register.

Notes:
1. Writing the DR register modifies the latch value but does not affect the pin status.
2. When switching from input to output mode, the DR register has to be written first to drive the correct level on the pin as soon as the port is configured as an output.
3. Do not use read/modify/write instructions (BSET or BRES) to modify the DR register as this might corrupt the DR content for I/Os configured as input.

External interrupt function

When an I/O is configured as Input with Interrupt, an event on this I/O can generate an external interrupt request to the CPU.

Each pin can independently generate an interrupt request. The interrupt sensitivity is independently programmable using the sensitivity bits in the EICR register.

Each external interrupt vector is linked to a dedicated group of I/O port pins (see pinout description and interrupt section). If several input pins are selected simultaneously as interrupt sources, these are first detected according to the sensitivity bits in the EICR register and then logically ORed.

The external interrupts are hardware interrupts, which means that the request latch (not accessible directly by the application) is automatically cleared when the corresponding interrupt vector is fetched. To clear an unwanted pending interrupt by software, the sensitivity bits in the EICR register must be modified.

9.2.2 Output Modes

The output configuration is selected by setting the corresponding DDR register bit. In this case, writing the DR register applies this digital value to the I/O pin through the latch. Then reading the DR register returns the previously stored value.

Two different output modes can be selected by software through the OR register: Output push-pull and open-drain.

DR register value and output pin status:

DR	Push-pull	Open-drain
0	V_{SS}	Vss
1	V_{DD}	Floating

9.2.3 Alternate Functions

When an on-chip peripheral is configured to use a pin, the alternate function is automatically selected. This alternate function takes priority over the standard I/O programming.

When the signal is coming from an on-chip peripheral, the I/O pin is automatically configured in output mode (push-pull or open drain according to the peripheral).

When the signal is going to an on-chip peripheral, the I/O pin must be configured in input mode. In this case, the pin state is also digitally readable by addressing the DR register.

Note: Input pull-up configuration can cause unexpected value at the input of the alternate peripheral input. When an on-chip peripheral use a pin as input and output, this pin has to be configured in input floating mode.

I/O PORTS (Cont'd)

Figure 32. I/O Port General Block Diagram

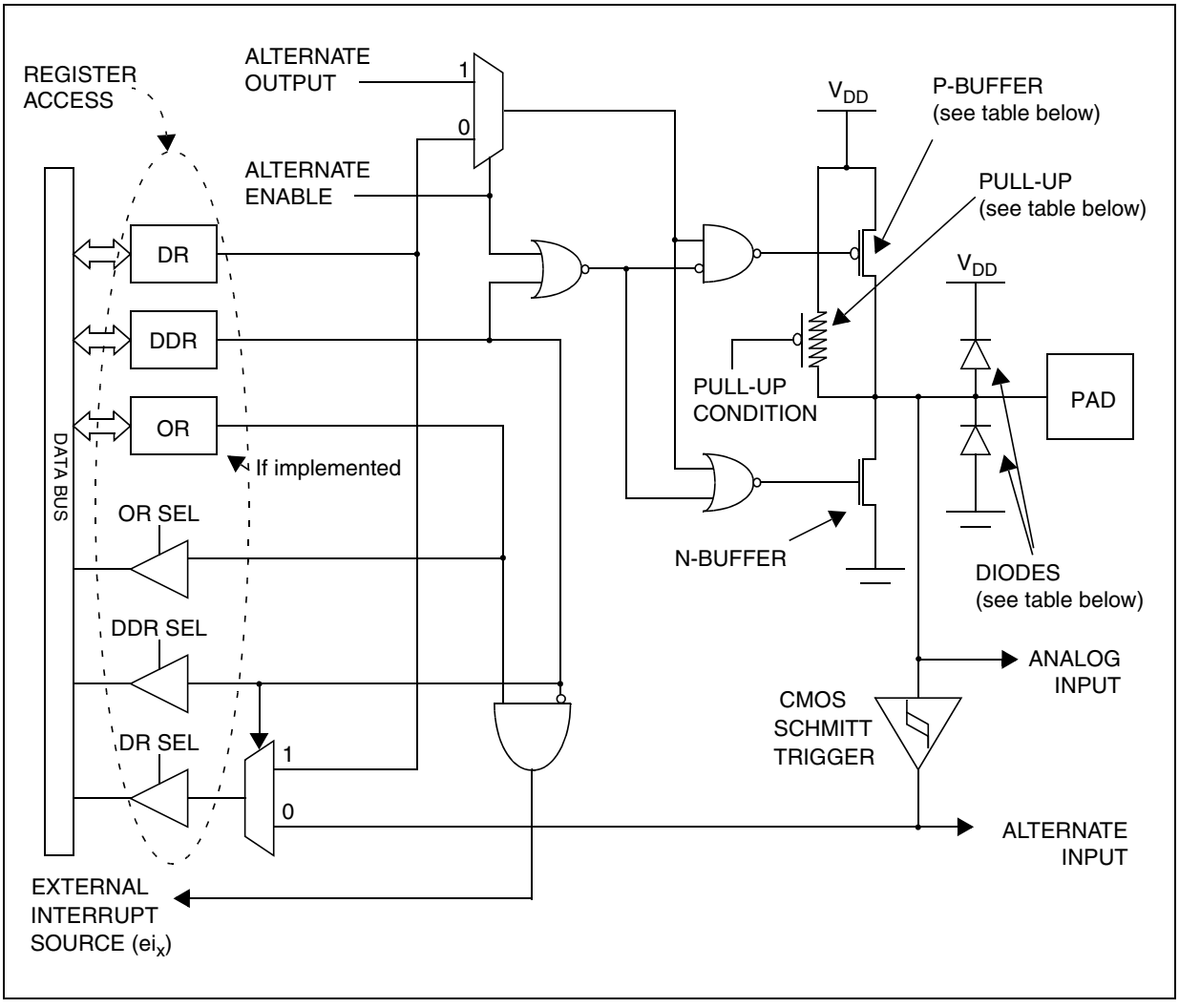


Table 10. I/O Port Mode Options

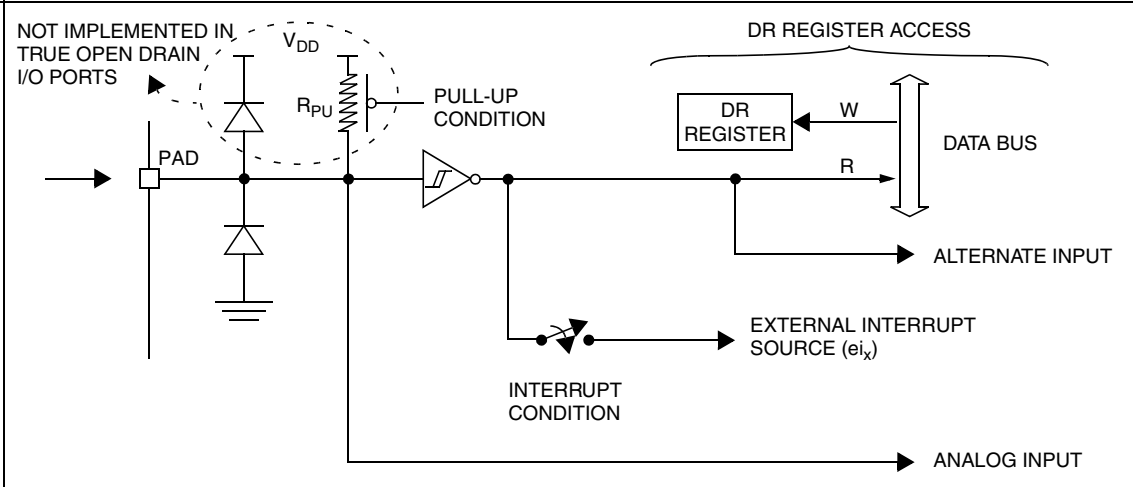
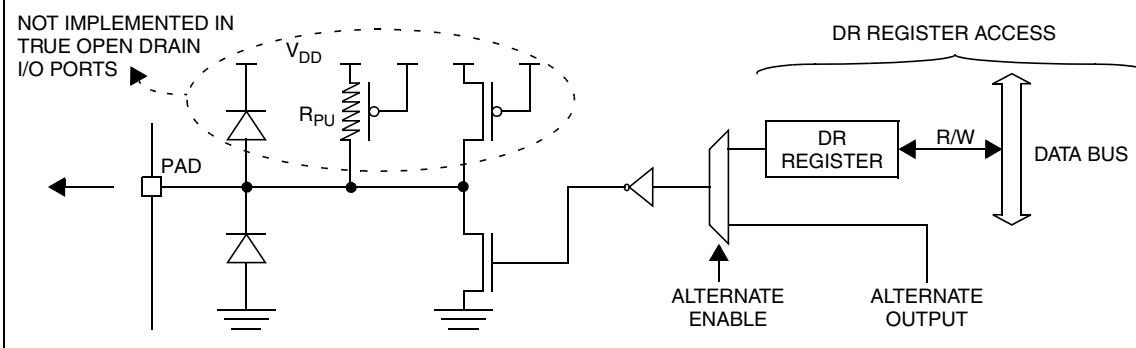
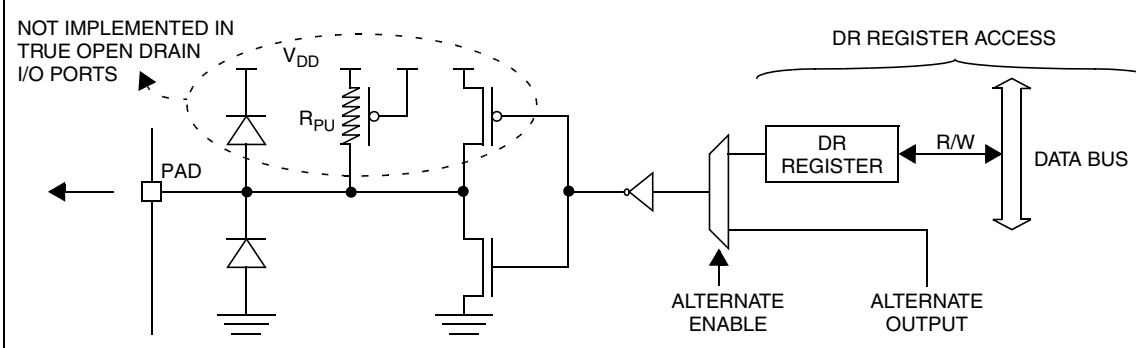
Configuration Mode		Pull-Up	P-Buffer	Diodes	
				to V_{DD}	to V_{SS}
Input	Floating with/without Interrupt	Off	Off	On	On
	Pull-up with/without Interrupt	On			
Output	Push-pull	Off	On		
	Open Drain (logic level)		Off		
	True Open Drain	NI	NI	NI (see note)	

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

Note: The diode to V_{DD} is not implemented in the true open drain pads. A local protection between the pad and V_{SS} is implemented to protect the device against positive stress.

I/O PORTS (Cont'd)

Table 11. I/O Port Configurations

	Hardware Configuration
INPUT [1]	
OPEN-DRAIN OUTPUT [2]	
PUSH-PULL OUTPUT [2]	

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)

CAUTION: The alternate function must not be activated as long as the pin is configured as input with interrupt, in order to avoid generating spurious interrupts.

Analog alternate function

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

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

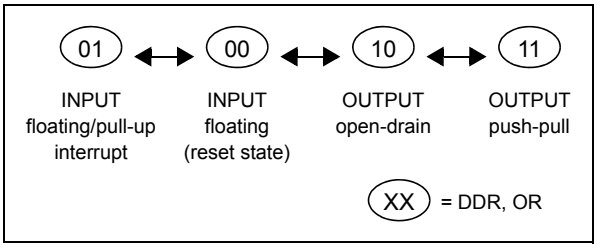
WARNING: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

9.3 I/O PORT IMPLEMENTATION

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

Switching these I/O ports from one state to another should be done in a sequence that prevents unwanted side effects. Recommended safe transitions are illustrated in Figure 33 on page 57. Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 33. Interrupt I/O Port State Transitions



9.4 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.

9.5 INTERRUPTS

The external interrupt event generates an interrupt if the corresponding configuration is selected with DDR and OR registers and the interrupt mask in the CC register is not active (RIM instruction).

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

I/O PORTS (Cont'd)

9.5.1 I/O Port Implementation

The I/O port register configurations are summarised as follows.

Standard Ports

PA4, PA2:0, PB5:0, PC7:4, PD7:6, PE5:0, PF5:0, PG7:0, PH7:0

MODE	DDR	OR
floating input	0	0
pull-up input	0	1
open drain output	1	0
push-pull output	1	1

Interrupt Ports

PA6, PA3, PB6, PC3, PC1, PD5, PD4, PD2 (with pull-up)

MODE	DDR	OR
floating input	0	0
pull-up interrupt input	0	1
open drain output	1	0
push-pull output	1	1

PA7, PA5, PB7, PC2, PC0, PD6, PD3, PD1 (without pull-up)

MODE	DDR	OR
floating input	0	0
floating interrupt input	0	1
open drain output	1	0
push-pull output	1	1

Table 12. Port Configuration

Port	Pin name	Input		Output	
		OR = 0	OR = 1	OR = 0	OR = 1
Port A	PA7, PA5	floating	floating interrupt	open drain	push-pull
	PA6, PA3	floating	pull-up interrupt	open drain	push-pull
	PA2:0	floating	pull-up	open drain	push-pull
Port B	PB7	floating	floating interrupt	open drain	push-pull
	PB6	floating	pull-up interrupt	open drain	push-pull
	PB5:0	floating	pull-up	open drain	push-pull
Port C	PC7:4	floating	pull-up	open drain	push-pull
	PC3, PC1	floating	pull-up interrupt	open drain	push-pull
	PC2, PC0	floating	floating interrupt	open drain	push-pull
Port D	PD7, PD0	floating	pull-up	open drain	push-pull
	PD6, PD3, PD1	floating	floating interrupt	open drain	push-pull
	PD5, PD4, PD2	floating	pull-up interrupt	open drain	push-pull
Port E	PE5:0	floating	pull-up	open drain	push-pull
Port F	PF5:0	floating	pull-up	open drain	push-pull
Port G	PG7:0	floating	pull-up	open drain	push-pull
Port H	PH7:0	floating	pull-up	open drain	push-pull

I/O PORTS (Cont'd)

Table 13. I/O Port Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
Reset Value of all I/O port registers		0	0	0	0	0	0	0	0
0000h	**PADR**	MSB							LSB
0001h	**PADDR**								
0002h	**PAOR**								
0003h	**PBDR**	MSB							LSB
0004h	**PBDDR**								
0005h	**PBOR**								
0006h	**PCDR**	MSB							LSB
0007h	**PCDDR**								
0008h	**PCOR**								
0009h	**PDDR**	MSB							LSB
000Ah	**PDDDR**								
000Bh	**PDOR**								
000Ch	**PEDR**	MSB							LSB
000Dh	**PEDDR**								
000Eh	**PEOR**								
000Fh	**PFDR**	MSB							LSB
0010h	**PFDDR**								
0011h	**PFOR**								
0012h	**PGDR**	MSB							LSB
0013h	**PGDDR**								
0014h	**PGOR**								
0015h	**PHDR**	MSB							LSB
0016h	**PHDDR**								
0017h	**PHOR**								

10 ON-CHIP PERIPHERALS

10.1 WINDOW WATCHDOG (WWDG)

10.1.1 Introduction

The Window Watchdog is used to detect the occurrence of a software fault, usually generated by external interference or by unforeseen logical conditions, which causes the application program to abandon its normal sequence. The Watchdog circuit generates an MCU reset on expiry of a programmed time period, unless the program refreshes the contents of the downcounter before the T6 bit becomes cleared. An MCU reset is also generated if the 7-bit downcounter value (in the control register) is refreshed before the downcounter has reached the window register value. This implies that the counter must be refreshed in a limited window.

10.1.2 Main Features

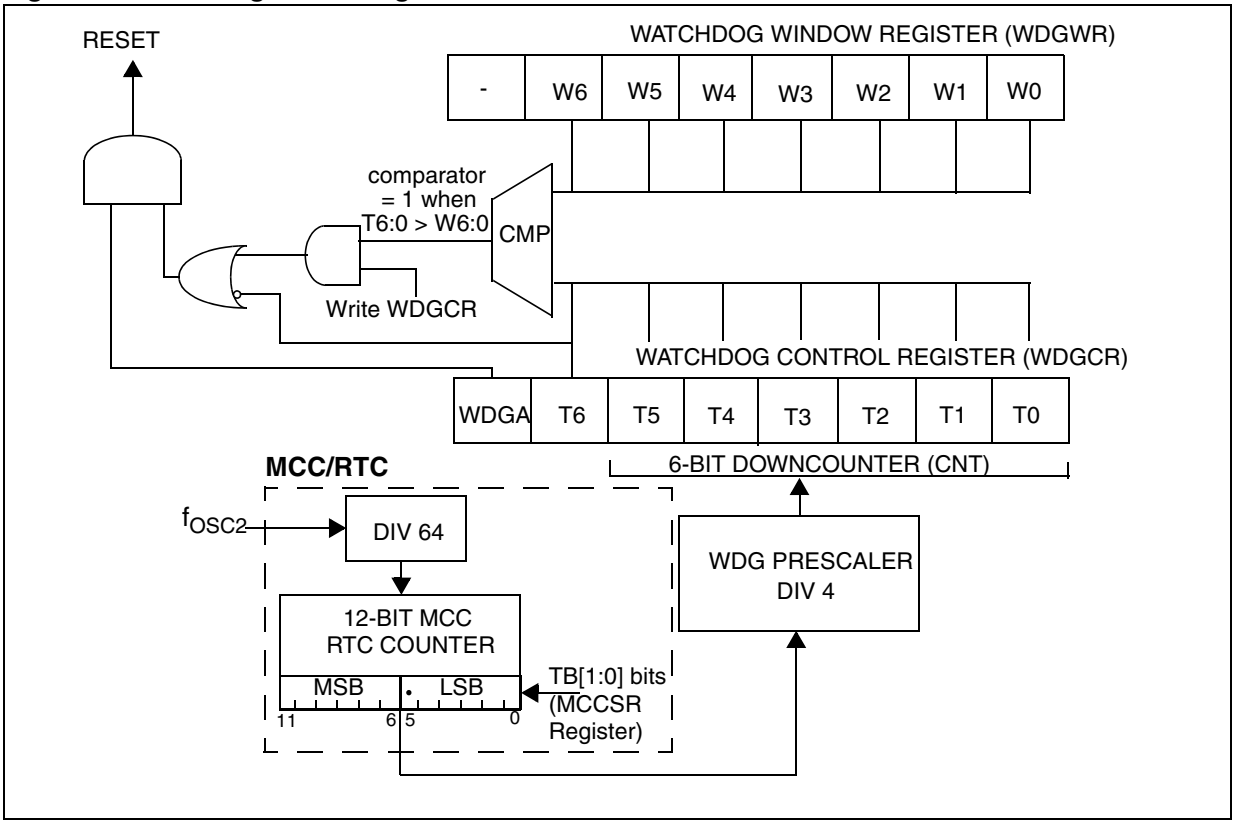
- Programmable free-running downcounter
- Conditional reset
  - Reset (if watchdog activated) when the downcounter value becomes less than 40h
  - Reset (if watchdog activated) if the downcounter is reloaded outside the window (see Figure 37)
- Hardware/Software Watchdog activation (selectable by option byte)
- Optional reset on HALT instruction (configurable by option byte)

10.1.3 Functional Description

The counter value stored in the WDGCR register (bits T[6:0]), is decremented every 16384 f_{OSC2} cycles (approx.), and the length of the timeout period can be programmed by the user in 64 increments.

If the watchdog is activated (the WDGA bit is set) and when the 7-bit downcounter (T[6:0] bits) rolls over from 40h to 3Fh (T6 becomes cleared), it initiates a reset cycle pulling low the reset pin for typically 30µs. If the software reloads the counter while the counter is greater than the value stored in the window register, then a reset is generated.

Figure 34. Watchdog Block Diagram



WINDOW WATCHDOG (Cont'd)

The application program must write in the WDGCR register at regular intervals during normal operation to prevent an MCU reset. This operation must occur only when the counter value is lower than the window register value. The value to be stored in the WDGCR register must be between FFh and C0h (see Figure 35):

– Enabling the watchdog:
When Software Watchdog is selected (by option byte), the watchdog is disabled after a reset. It is enabled by setting the WDGA bit in the WDGCR register, then it cannot be disabled again except by a reset.

When Hardware Watchdog is selected (by option byte), the watchdog is always active and the WDGA bit is not used.

– Controlling the downcounter:
This downcounter is free-running: It counts down even if the watchdog is disabled. When the watchdog is enabled, the T6 bit must be set to prevent generating an immediate reset.
The T[5:0] bits contain the number of increments which represents the time delay before the watchdog produces a reset (see Figure 35. Approximate Timeout Duration). The timing varies between a minimum and a maximum value due to the unknown status of the prescaler when writing to the WDGCR register (see Figure 36).

The window register (WDGWR) contains the high limit of the window: To prevent a reset, the downcounter must be reloaded when its value is lower than the window register value and greater than 3Fh. Figure 37 describes the window watchdog process.

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

– Watchdog Reset on Halt option
If the watchdog is activated and the watchdog reset on halt option is selected, then the HALT instruction will generate a Reset.

10.1.4 Using Halt Mode with the WDG

If Halt mode with Watchdog is enabled by option byte (no watchdog reset on HALT instruction), it is recommended before executing the HALT instruction to refresh the WDG counter, to avoid an unexpected WDG reset immediately after waking up the microcontroller.

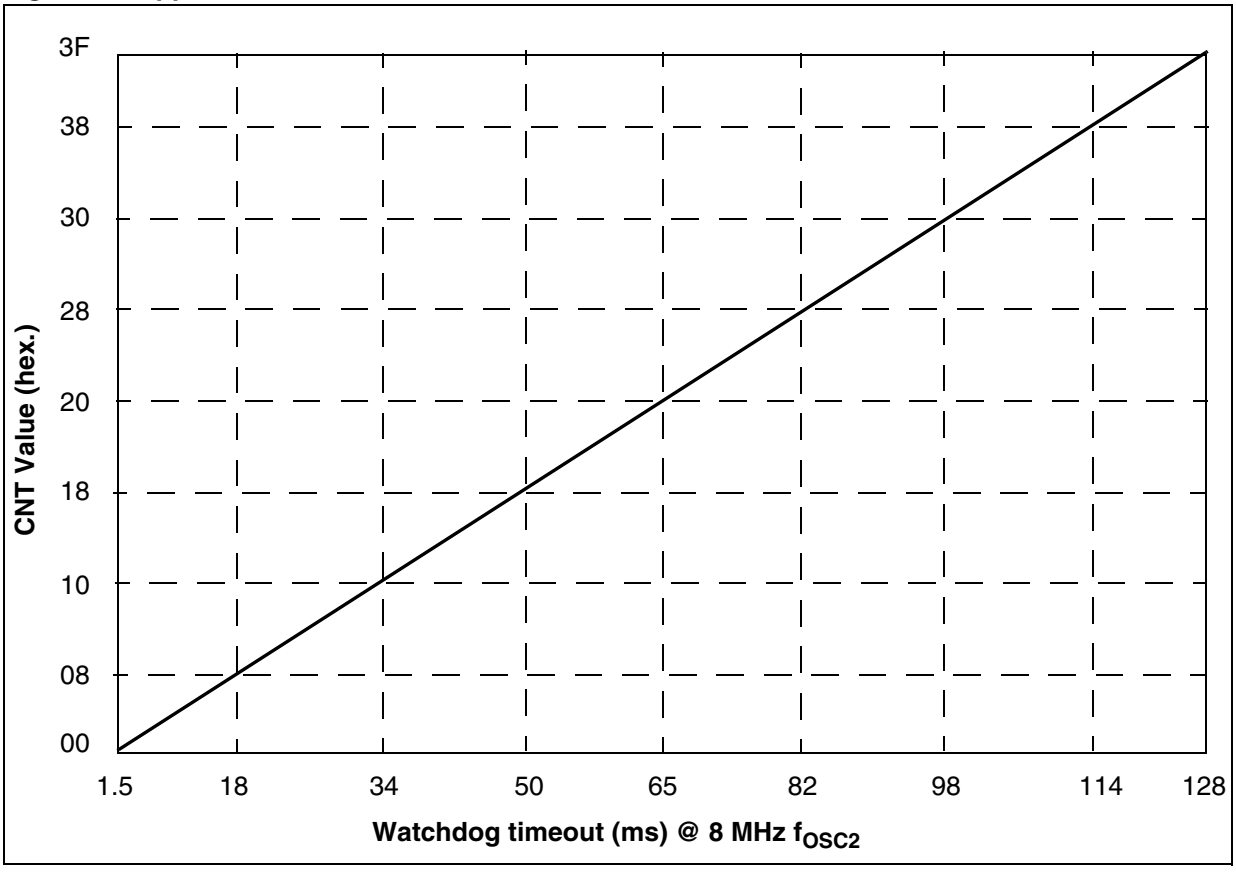
WINDOW WATCHDOG (Cont'd)

10.1.5 How to Program the Watchdog Timeout

Figure 35 shows the linear relationship between the 6-bit value to be loaded in the Watchdog Counter (CNT) and the resulting timeout duration in milliseconds. This can be used for a quick calculation without taking the timing variations into account. If more precision is needed, use the formulae in Figure 36.

Caution: When writing to the WDGCR register, always write 1 in the T6 bit to avoid generating an immediate reset.

Figure 35. Approximate Timeout Duration



WINDOW WATCHDOG (Cont'd)

Figure 36. Exact Timeout Duration (t_{min} and t_{max})

WHERE:

$t_{min0} = (LSB + 128) \times 64 \times t_{OSC2}$

$t_{max0} = 16384 \times t_{OSC2}$

$t_{OSC2} = 125ns$ if $f_{OSC2} = 8$ MHz

CNT = Value of T[5:0] bits in the WDGCR register (6 bits)
MSB and LSB are values from the table below depending on the timebase selected by the TB[1:0] bits in the MCCSR register

TB1 Bit (MCCSR Reg.)	TB0 Bit (MCCSR Reg.)	Selected MCCSR Timebase	MSB	LSB
0	0	2ms	4	59
0	1	4ms	8	53
1	0	10ms	20	35
1	1	25ms	49	54

To calculate the minimum Watchdog Timeout (t_{min}):

IF $CNT < \left\lceil \frac{MSB}{4} \right\rceil$ **THEN** $t_{min} = t_{min0} + 16384 \times CNT \times t_{osc2}$

ELSE $t_{min} = t_{min0} + \left[16384 \times \left(CNT - \left\lceil \frac{4CNT}{MSB} \right\rceil \right) + (192 + LSB) \times 64 \times \left\lceil \frac{4CNT}{MSB} \right\rceil \right] \times t_{osc2}$

To calculate the maximum Watchdog Timeout (t_{max}):

IF $CNT \leq \left\lceil \frac{MSB}{4} \right\rceil$ **THEN** $t_{max} = t_{max0} + 16384 \times CNT \times t_{osc2}$

ELSE $t_{max} = t_{max0} + \left[16384 \times \left(CNT - \left\lceil \frac{4CNT}{MSB} \right\rceil \right) + (192 + LSB) \times 64 \times \left\lceil \frac{4CNT}{MSB} \right\rceil \right] \times t_{osc2}$

Note: In the above formulae, division results must be rounded down to the next integer value.

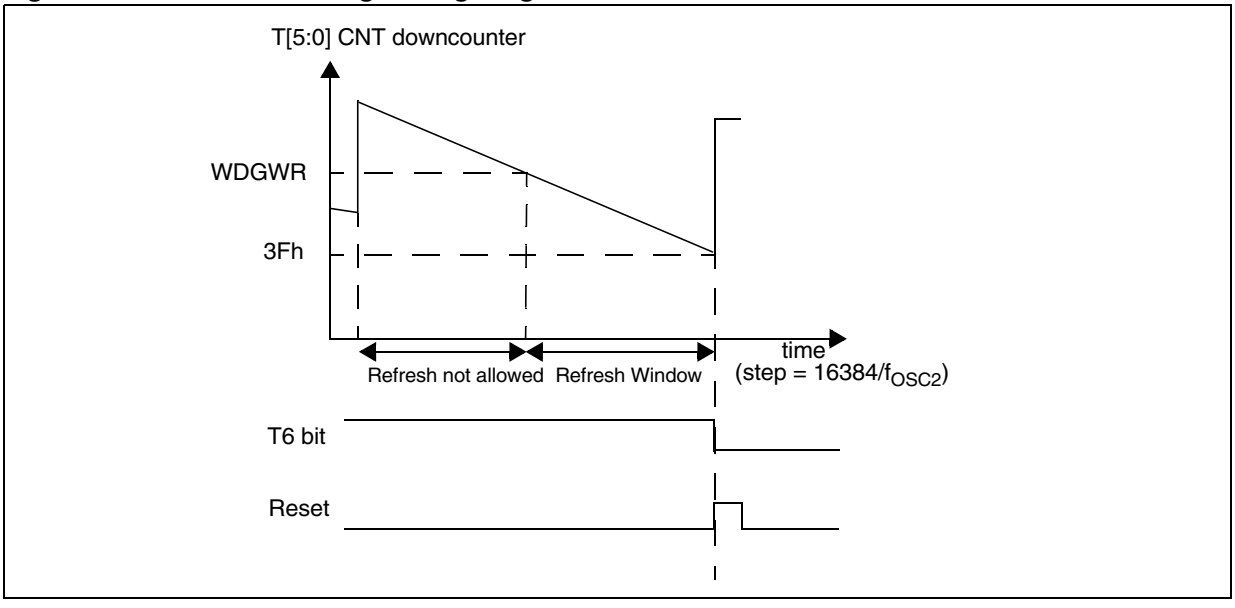
Example:

With 2ms timeout selected in MCCSR register

Value of T[5:0] Bits in WDGCR Register (Hex.)	Min. Watchdog Timeout (ms) t_{min}	Max. Watchdog Timeout (ms) t_{max}
00	1.496	2.048
3F	128	128.552

WINDOW WATCHDOG (Cont'd)

Figure 37. Window Watchdog Timing Diagram



10.1.6 Low Power Modes

Mode	Description		
Slow	No effect on Watchdog: The downcounter continues to decrement at normal speed.		
Wait	No effect on Watchdog: The downcounter continues to decrement.		
Halt	**OIE bit in MCCSR register**	**WDGHALT bit in Option Byte**	
	0	0	No Watchdog reset is generated. The MCU enters Halt mode. The Watchdog counter is decremented once and then stops counting and is no longer able to generate a watchdog reset until the MCU receives an external interrupt or a reset. If an interrupt is received (refer to interrupt table mapping to see interrupts which can occur in halt mode), the Watchdog restarts counting after 256 or 4096 CPU clocks. If a reset is generated, the Watchdog is disabled (reset state) unless Hardware Watchdog is selected by option byte. For application recommendations see Section 10.1.8 below.
	0	1	A reset is generated instead of entering halt mode.
Active-halt	1	x	No reset is generated. The MCU enters Active Halt mode. The Watchdog counter is not decremented. It stop counting. When the MCU receives an oscillator interrupt or external interrupt, the Watchdog restarts counting immediately. When the MCU receives a reset the Watchdog restarts counting after 256 or 4096 CPU clocks.

10.1.7 Hardware Watchdog Option

If Hardware Watchdog is selected by option byte, the watchdog is always active and the WDGA bit in the WDGCR is not used. Refer to the Option Byte description.

10.1.8 Using Halt Mode with the WDG (WDGHALT option)

The following recommendation applies if Halt mode is used when the watchdog is enabled.

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

WINDOW WATCHDOG (Cont'd)

10.1.9 Interrupts

None.

10.1.10 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** *Activation bit.*

This bit is set by software and only cleared by hardware after a reset. When WDGA = 1, the watchdog can generate a reset.

0: Watchdog disabled

1: Watchdog enabled

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

Bits 6:0 = **T[6:0]** *7-bit counter (MSB to LSB).*
These bits contain the value of the watchdog counter. It is decremented every 16384 f_{OSC2} cycles (approx.). A reset is produced when it rolls over from 40h to 3Fh (T6 becomes cleared).

WINDOW REGISTER (WDGWR)

Read/Write

Reset Value: 0111 1111 (7Fh)

7							0
-	W6	W5	W4	W3	W2	W1	W0

Bit 7 = Reserved

Bits 6:0 = **W[6:0]** *7-bit window value*
These bits contain the window value to be compared to the downcounter.

Table 14. Watchdog Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
002Ah	**WDGCR**	WDGA	T6	T5	T4	T3	T2	T1	T0
	Reset Value	0	1	1	1	1	1	1	1
002Bh	**WDGWR**	0	W6	W5	W4	W3	W2	W1	W0
	Reset Value	0	1	1	1	1	1	1	1

10.2 PWM AUTO-RELOAD TIMER (ART)

10.2.1 Introduction

The Pulse Width Modulated Auto-Reload Timer on-chip peripheral consists of an 8-bit auto reload counter with compare/capture capabilities and of a 7-bit prescaler clock source.

These resources allow five possible operating modes:

– Generation of up to 4 independent PWM signals

– Output compare and Time base interrupt

– Up to two input capture functions

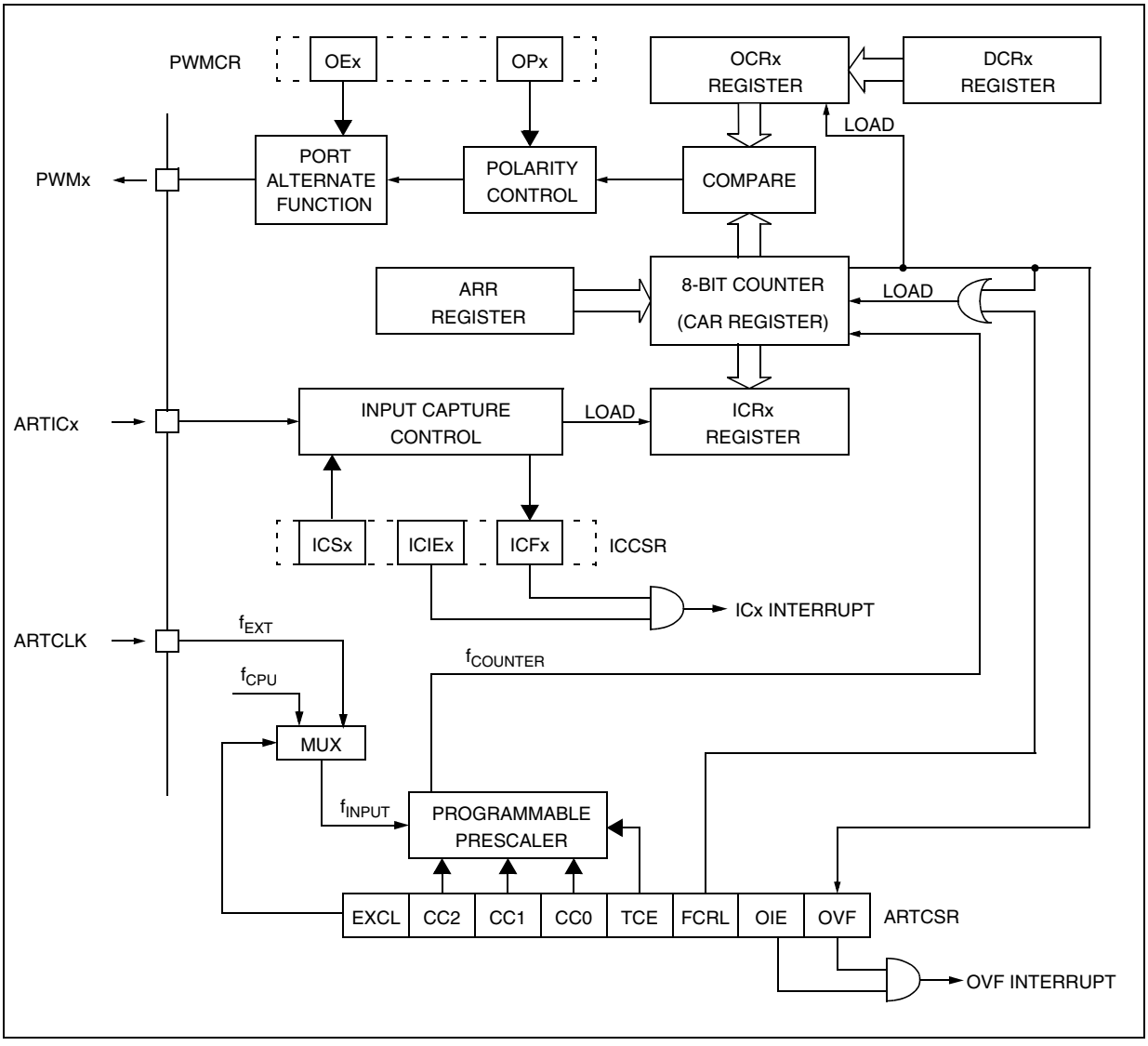
– External event detector

– Up to two external interrupt sources

The three first modes can be used together with a single counter frequency.

The timer can be used to wake up the MCU from Wait and Halt modes.

Figure 38. PWM Auto-Reload Timer Block Diagram



ON-CHIP PERIPHERALS (Cont'd)

10.2.2 Functional Description

Counter

The free running 8-bit counter is fed by the output of the prescaler, and is incremented on every rising edge of the clock signal.

It is possible to read or write the contents of the counter on the fly by reading or writing the Counter Access register (ARTCAR).

When a counter overflow occurs, the counter is automatically reloaded with the contents of the ARTARR register (the prescaler is not affected).

Counter clock and prescaler

The counter clock frequency is given by:

$$f_{COUNTER} = f_{INPUT} / 2^{CC[2:0]}$$

The timer counter's input clock (f_{INPUT}) feeds the 7-bit programmable prescaler, which selects one of the 8 available taps of the prescaler, as defined by CC[2:0] bits in the Control/Status Register (ARTCSR). Thus the division factor of the prescaler can be set to 2^n (where n = 0, 1,..7).

This f_{INPUT} frequency source is selected through the EXCL bit of the ARTCSR register and can be either the f_{CPU} or an external input frequency f_{EXT}.

The clock input to the counter is enabled by the TCE (Timer Counter Enable) bit in the ARTCSR register. When TCE is reset, the counter is stopped and the prescaler and counter contents are frozen. When TCE is set, the counter runs at the rate of the selected clock source.

Counter and Prescaler Initialization

After RESET, the counter and the prescaler are cleared and $f_{INPUT} = f_{CPU}$.

The counter can be initialized by:

– Writing to the ARTARR register and then setting the FCRL (Force Counter Re-Load) and the TCE (Timer Counter Enable) bits in the ARTCSR register.

– Writing to the ARTCAR counter access register,

In both cases the 7-bit prescaler is also cleared, whereupon counting will start from a known value.

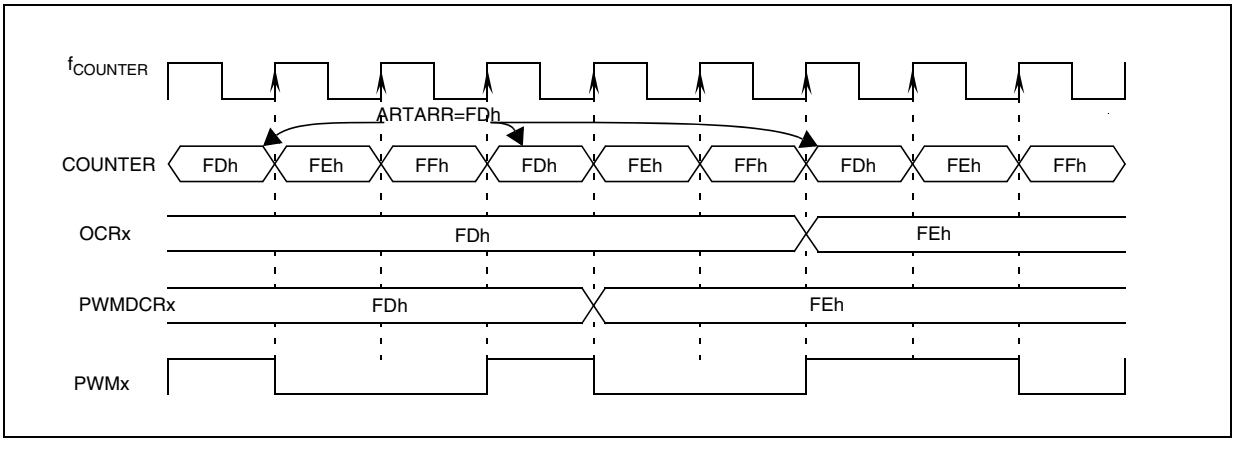
Direct access to the prescaler is not possible.

Output compare control

The timer compare function is based on four different comparisons with the counter (one for each PWMx output). Each comparison is made between the counter value and an output compare register (OCRx) value. This OCRx register can not be accessed directly, it is loaded from the duty cycle register (PWMDCRx) at each overflow of the counter.

This double buffering method avoids glitch generation when changing the duty cycle on the fly.

Figure 39. Output compare control



ON-CHIP PERIPHERALS (Cont'd)

Independent PWM signal generation

This mode allows up to four Pulse Width Modulated signals to be generated on the PWMx output pins with minimum core processing overhead. This function is stopped during Halt mode.

Each PWMx output signal can be selected independently using the corresponding OEx bit in the PWM Control register (PWMCR). When this bit is set, the corresponding I/O pin is configured as output push-pull alternate function.

The PWM signals all have the same frequency which is controlled by the counter period and the ARTARR register value.

$$f_{PWM} = f_{COUNTER} / (256 - ARTARR)$$

When a counter overflow occurs, the PWMx pin level is changed depending on the corresponding OPx (output polarity) bit in the PWMCR register. When the counter reaches the value contained in one of the output compare register (OCRx) the corresponding PWMx pin level is restored.

It should be noted that the reload values will also affect the value and the resolution of the duty cycle of the PWM output signal. To obtain a signal on a PWMx pin, the contents of the OCRx register must be greater than the contents of the ARTARR register.

The maximum available resolution for the PWMx duty cycle is:

$$Resolution = 1 / (256 - ARTARR)$$

Note: To get the maximum resolution (1/256), the ARTARR register must be 0. With this maximum resolution, 0% and 100% can be obtained by changing the polarity.

Figure 40. PWM Auto-reload Timer Function

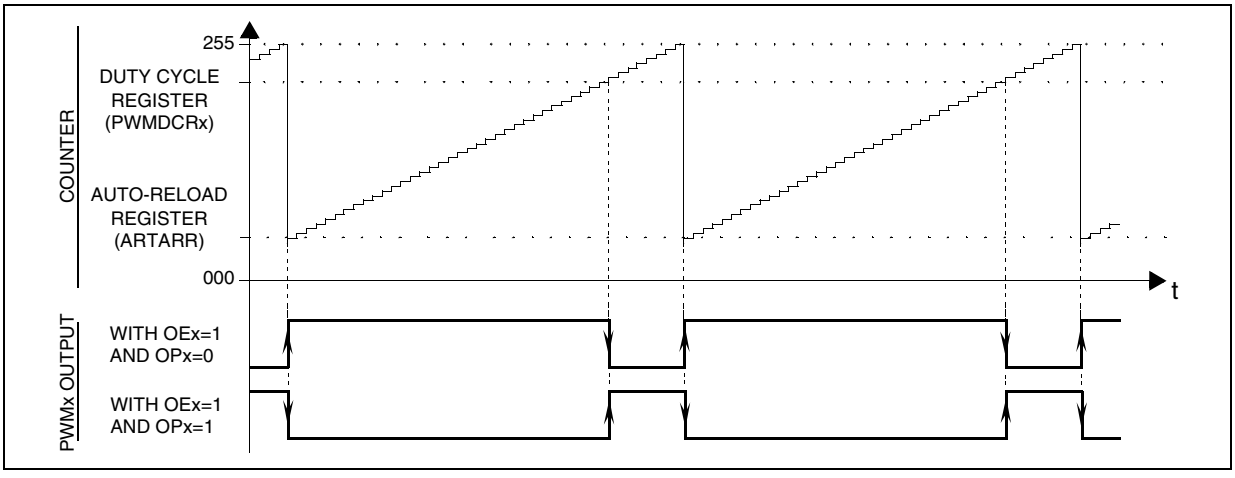
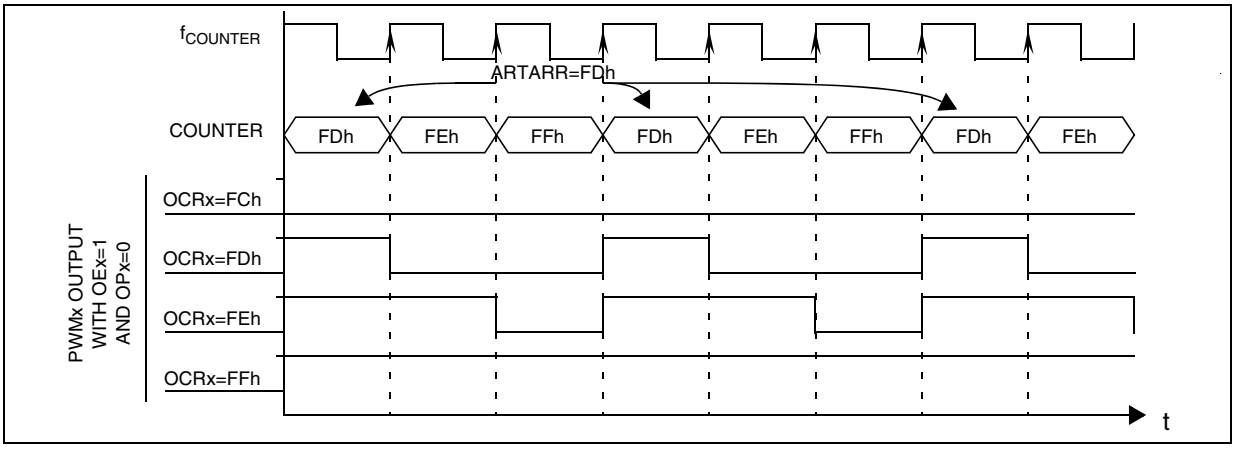


Figure 41. PWM Signal from 0% to 100% Duty Cycle



ON-CHIP PERIPHERALS (Cont'd)

Output compare and Time base interrupt

On overflow, the OVF flag of the ARTCSR register is set and an overflow interrupt request is generated if the overflow interrupt enable bit, OIE, in the ARTCSR register, is set. The OVF flag must be reset by the user software. This interrupt can be used as a time base in the application.

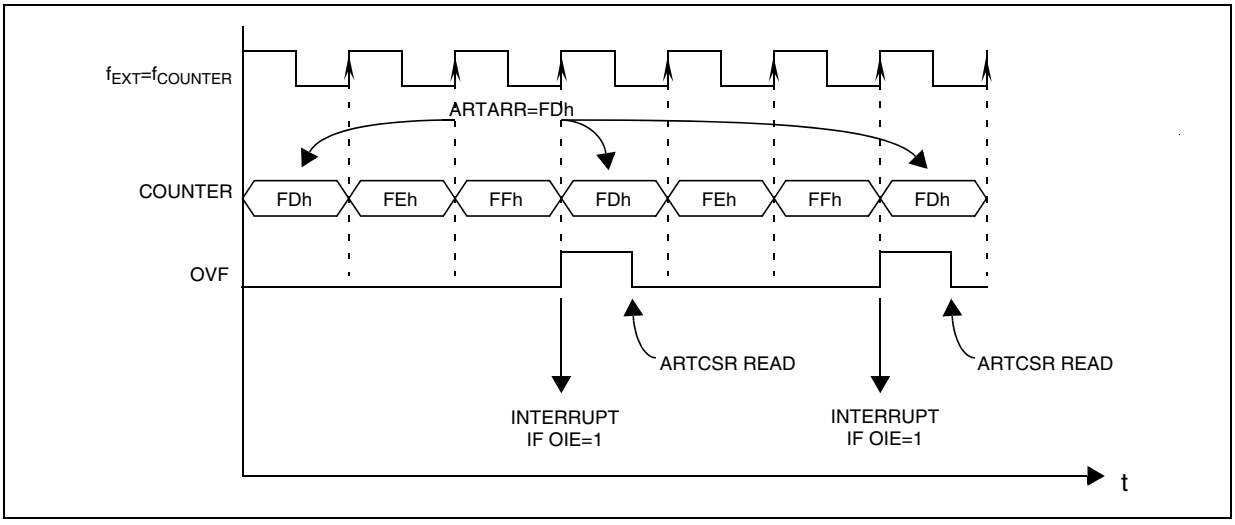
External clock and event detector mode

Using the f_{EXT} external prescaler input clock, the auto-reload timer can be used as an external clock event detector. In this mode, the ARTARR register is used to select the n_{EVENT} number of events to be counted before setting the OVF flag.

$$n_{EVENT} = 256 - ARTARR$$

Caution: The external clock function is not available in Halt mode. If Halt mode is used in the application, prior to executing the HALT instruction, the counter must be disabled by clearing the TCE bit in the ARTCSR register to avoid spurious counter increments.

Figure 42. External Event Detector Example (3 counts)



ON-CHIP PERIPHERALS (Cont'd)

Input capture function

This mode allows the measurement of external signal pulse widths through ARTICRx registers.

Each input capture can generate an interrupt independently on a selected input signal transition. This event is flagged by a set of the corresponding CFx bits of the Input Capture Control/Status register (ARTICCSR).

These input capture interrupts are enabled through the CIEx bits of the ARTICCSR register.

The active transition (falling or rising edge) is software programmable through the CSx bits of the ARTICCSR register.

The read only input capture registers (ARTICRx) are used to latch the auto-reload counter value when a transition is detected on the ARTICx pin (CFx bit set in ARTICCSR register). After fetching the interrupt vector, the CFx flags can be read to identify the interrupt source.

Note: After a capture detection, data transfer in the ARTICRx register is inhibited until it is read (clearing the CFx bit).
The timer interrupt remains pending while the CFx flag is set when the interrupt is enabled (CIEx bit set). This means, the ARTICRx register has to be read at each capture event to clear the CFx flag.

The timing resolution is given by auto-reload counter cycle time ($1/f_{COUNTER}$).

Note: During Halt mode, if both input capture and external clock are enabled, the ARTICRx register value is not guaranteed if the input capture pin and the external clock change simultaneously.

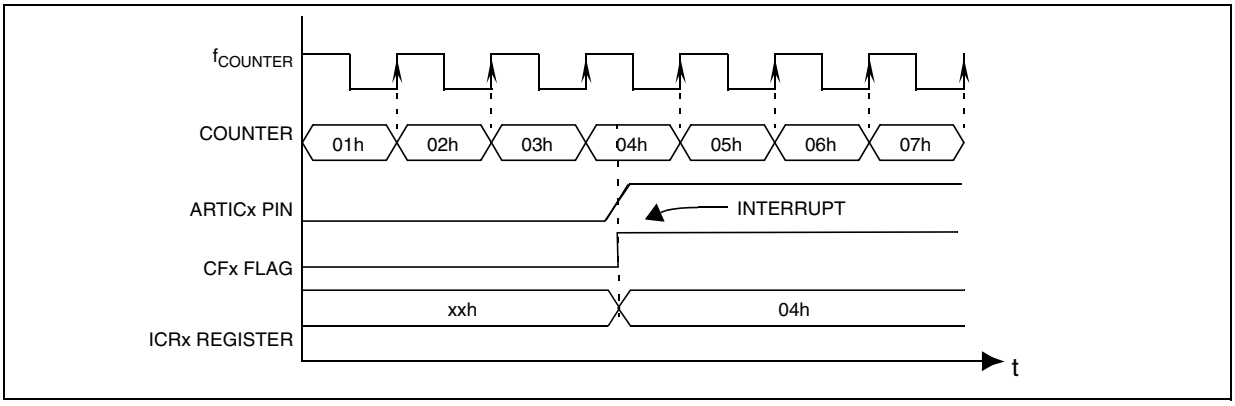
External interrupt capability

This mode allows the Input capture capabilities to be used as external interrupt sources. The interrupts are generated on the edge of the ARTICx signal.

The edge sensitivity of the external interrupts is programmable (CSx bit of ARTICCSR register) and they are independently enabled through CIEx bits of the ARTICCSR register. After fetching the interrupt vector, the CFx flags can be read to identify the interrupt source.

During Halt mode, the external interrupts can be used to wake up the micro (if the CIEx bit is set).

Figure 43. Input Capture Timing Diagram



ON-CHIP PERIPHERALS (Cont'd)

10.2.3 Register Description

CONTROL / STATUS REGISTER (ARTCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
EXCL	CC2	CC1	CC0	TCE	FCRL	OIE	OVF

Bit 7 = **EXCL** *External Clock*
This bit is set and cleared by software. It selects the input clock for the 7-bit prescaler.
0: CPU clock.
1: External clock.

Bit 6:4 = **CC[2:0]** *Counter Clock Control*
These bits are set and cleared by software. They determine the prescaler division ratio from f_{INPUT}.

$f_{COUNTER}$	With f_{INPUT}=8 MHz	CC2	CC1	CC0
f_{INPUT}	8 MHz	0	0	0
f_{INPUT} / 2	4 MHz	0	0	1
f_{INPUT} / 4	2 MHz	0	1	0
f_{INPUT} / 8	1 MHz	0	1	1
f_{INPUT} / 16	500 kHz	1	0	0
f_{INPUT} / 32	250 kHz	1	0	1
f_{INPUT} / 64	125 kHz	1	1	0
f_{INPUT} / 128	62.5 kHz	1	1	1

Bit 3 = **TCE** *Timer Counter Enable*
This bit is set and cleared by software. It puts the timer in the lowest power consumption mode.
0: Counter stopped (prescaler and counter frozen).
1: Counter running.

Bit 2 = **FCRL** *Force Counter Re-Load*
This bit is write-only and any attempt to read it will yield a logical zero. When set, it causes the contents of ARTARR register to be loaded into the counter, and the content of the prescaler register to be cleared in order to initialize the timer before starting to count.

Bit 1 = **OIE** *Overflow Interrupt Enable*
This bit is set and cleared by software. It allows to enable/disable the interrupt which is generated when the OVF bit is set.
0: Overflow Interrupt disable.
1: Overflow Interrupt enable.

Bit 0 = **OVF** *Overflow Flag*
This bit is set by hardware and cleared by software reading the ARTCSR register. It indicates the transition of the counter from FFh to the ARTARR value.
0: New transition not yet reached
1: Transition reached

COUNTER ACCESS REGISTER (ARTCAR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0

Bit 7:0 = **CA[7:0]** *Counter Access Data*

These bits can be set and cleared either by hardware or by software. The ARTCAR register is used to read or write the auto-reload counter "on the fly" (while it is counting).

AUTO-RELOAD REGISTER (ARTARR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0

Bit 7:0 = **AR[7:0]** *Counter Auto-Reload Data*

These bits are set and cleared by software. They are used to hold the auto-reload value which is automatically loaded in the counter when an overflow occurs. At the same time, the PWM output levels are changed according to the corresponding OPx bit in the PWMCR register.

This register has two PWM management functions:

- Adjusting the PWM frequency
- Setting the PWM duty cycle resolution

PWM Frequency vs Resolution:

ARTARR value	Resolution	f_{PWM}	
		Min	Max
0	8-bit	~0.244 kHz	31.25 kHz
[0..127]	> 7-bit	~0.244 kHz	62.5 kHz
[128..191]	> 6-bit	~0.488 kHz	125 kHz
[192..223]	> 5-bit	~0.977 kHz	250 kHz
[224..239]	> 4-bit	~1.953 kHz	500 kHz

ON-CHIP PERIPHERALS (Cont'd)

PWM CONTROL REGISTER (PWMCR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OE3	OE2	OE1	OE0	OP3	OP2	OP1	OP0

Bit 7:4 = **OE[3:0]** *PWM Output Enable*
These bits are set and cleared by software. They enable or disable the PWM output channels independently acting on the corresponding I/O pin.
0: PWM output disabled.
1: PWM output enabled.

Bit 3:0 = **OP[3:0]** *PWM Output Polarity*
These bits are set and cleared by software. They independently select the polarity of the four PWM output signals.

PWMx output level		OPx
Counter <= OCRx	Counter > OCRx	
1	0	0
0	1	1

Note: When an OPx bit is modified, the PWMx output signal polarity is immediately reversed.

DUTY CYCLE REGISTERS (PWMDCRx)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0

Bit 7:0 = **DC[7:0]** *Duty Cycle Data*

These bits are set and cleared by software.

A PWMDCRx register is associated with the OCRx register of each PWM channel to determine the second edge location of the PWM signal (the first edge location is common to all channels and given by the ARTARR register). These PWMDCR registers allow the duty cycle to be set independently for each PWM channel.

ON-CHIP PERIPHERALS (Cont'd)

INPUT CAPTURE CONTROL / STATUS REGISTER (ARTICCSR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	CS2	CS1	CIE2	CIE1	CF2	CF1

Bit 7:6 = Reserved, always read as 0.

Bit 5:4 = **CS[2:1]** *Capture Sensitivity*
These bits are set and cleared by software. They determine the trigger event polarity on the corresponding input capture channel.
0: Falling edge triggers capture on channel x.
1: Rising edge triggers capture on channel x.

Bit 3:2 = **CIE[2:1]** *Capture Interrupt Enable*
These bits are set and cleared by software. They enable or disable the Input capture channel interrupts independently.
0: Input capture channel x interrupt disabled.
1: Input capture channel x interrupt enabled.

Bit 1:0 = **CF[2:1]** *Capture Flag*
These bits are set by hardware and cleared by software reading the corresponding ARTICRx register. Each CFx bit indicates that an input capture x has occurred.
0: No input capture on channel x.
1: An input capture has occurred on channel x.

INPUT CAPTURE REGISTERS (ARTICRx)

Read only

Reset Value: 0000 0000 (00h)

7							0
IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0

Bit 7:0 = **IC[7:0]** *Input Capture Data*

These read only bits are set and cleared by hardware. An ARTICRx register contains the 8-bit auto-reload counter value transferred by the input capture channel x event.

PWM AUTO-RELOAD TIMER (Cont'd)

Table 15. PWM Auto-Reload Timer Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0074h	**PWMDCR3**	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
	Reset Value	0	0	0	0	0	0	0	0
0075h	**PWMDCR2**	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
	Reset Value	0	0	0	0	0	0	0	0
0076h	**PWMDCR1**	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
	Reset Value	0	0	0	0	0	0	0	0
0077h	**PWMDCR0**	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
	Reset Value	0	0	0	0	0	0	0	0
0078h	**PWMCR**	OE3	OE2	OE1	OE0	OP3	OP2	OP1	OP0
	Reset Value	0	0	0	0	0	0	0	0
0079h	**ARTCSR**	EXCL	CC2	CC1	CC0	TCE	FCRL	OIE	OVF
	Reset Value	0	0	0	0	0	0	0	0
007Ah	**ARTCAR**	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0
	Reset Value	0	0	0	0	0	0	0	0
007Bh	**ARTARR**	AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
	Reset Value	0	0	0	0	0	0	0	0
007Ch	**ARTICCSR**			CS2	CS1	CIE2	CIE1	CF2	CF1
	Reset Value	0	0	0	0	0	0	0	0
007Dh	**ARTICR1**	IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0
	Reset Value	0	0	0	0	0	0	0	0
007Eh	**ARTICR2**	IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0
	Reset Value	0	0	0	0	0	0	0	0

10.3 16-BIT TIMER

10.3.1 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).

10.3.2 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 44.

***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'.

10.3.3 Functional Description

10.3.3.1 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).

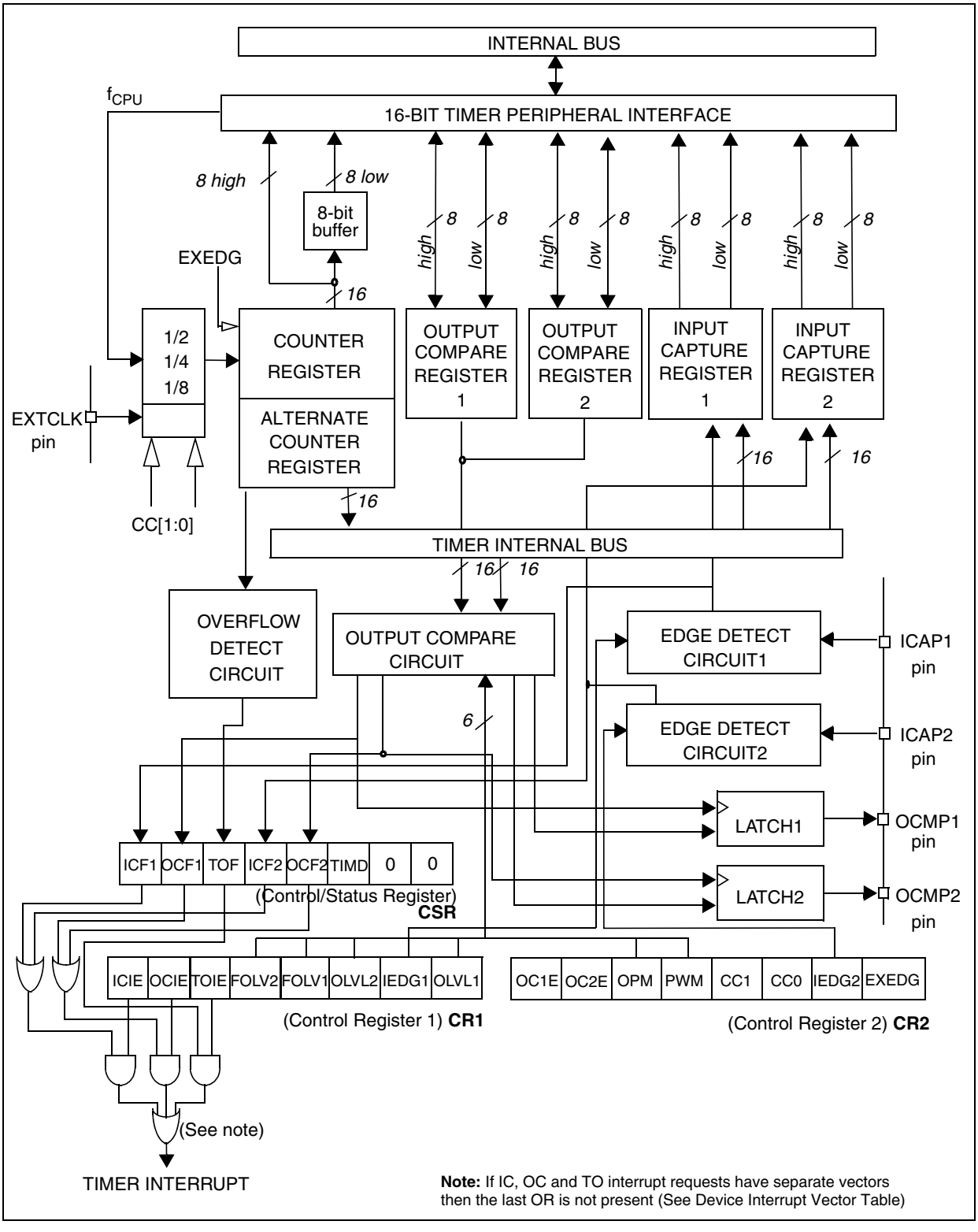
These two read-only 16-bit registers contain the same value but with the difference that reading the ACLR register does not clear the TOF bit (Timer overflow flag), located in the Status register, (SR), (see note at the end of paragraph titled 16-bit read sequence).

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 16 Clock Control Bits. 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.

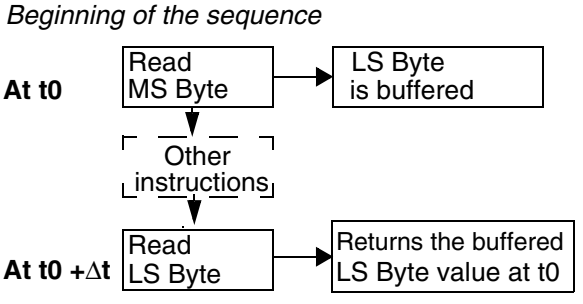
16-BIT TIMER (Cont'd)

Figure 44. Timer Block Diagram



16-BIT TIMER (Cont'd)

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



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).

10.3.3.2 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 45. Counter Timing Diagram, internal clock divided by 2

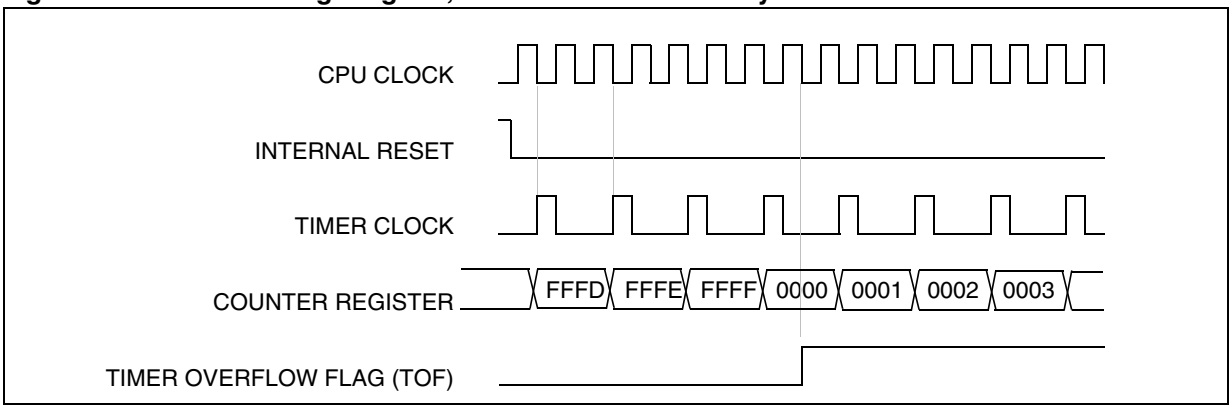


Figure 46. Counter Timing Diagram, internal clock divided by 4

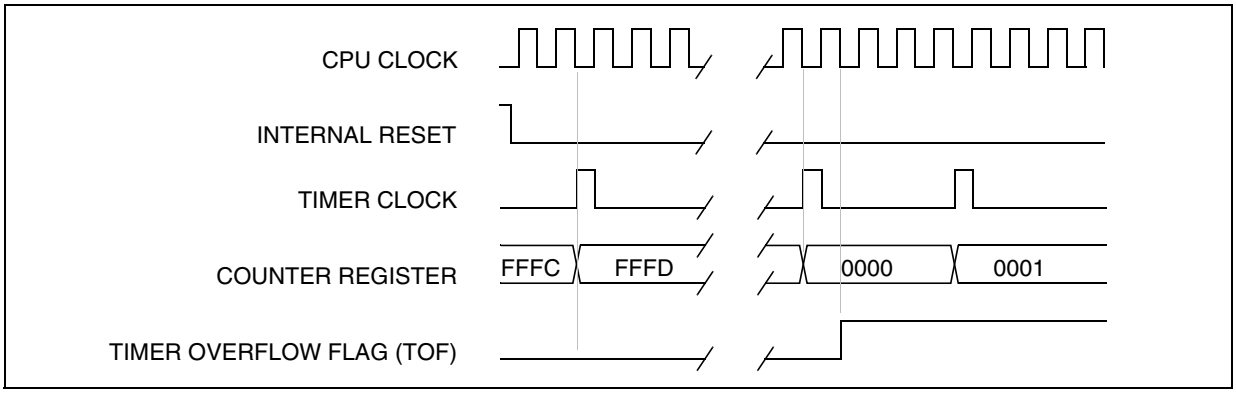
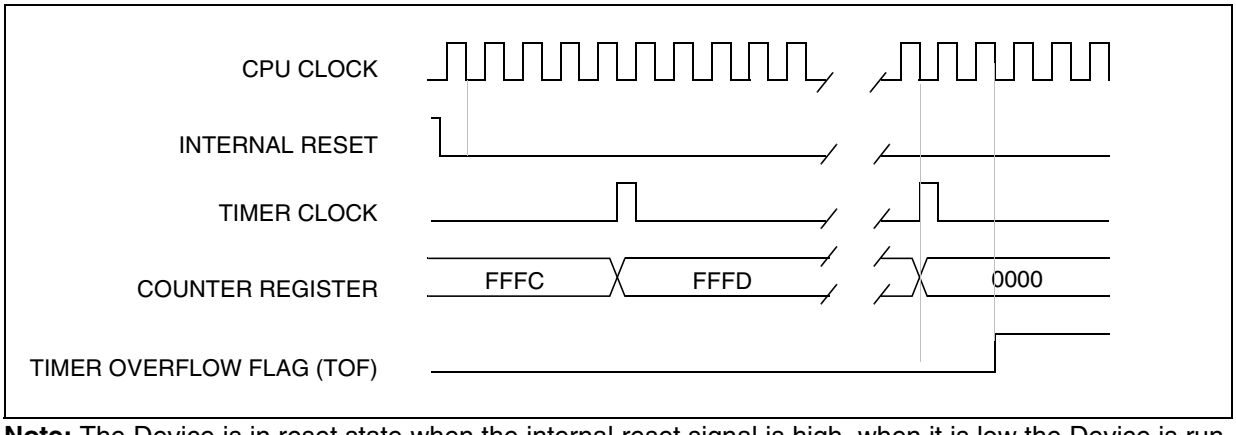


Figure 47. Counter Timing Diagram, internal clock divided by 8



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)

10.3.3.3 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 16 Clock Control Bits).

– 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 49).

– 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 48. Input Capture Block Diagram

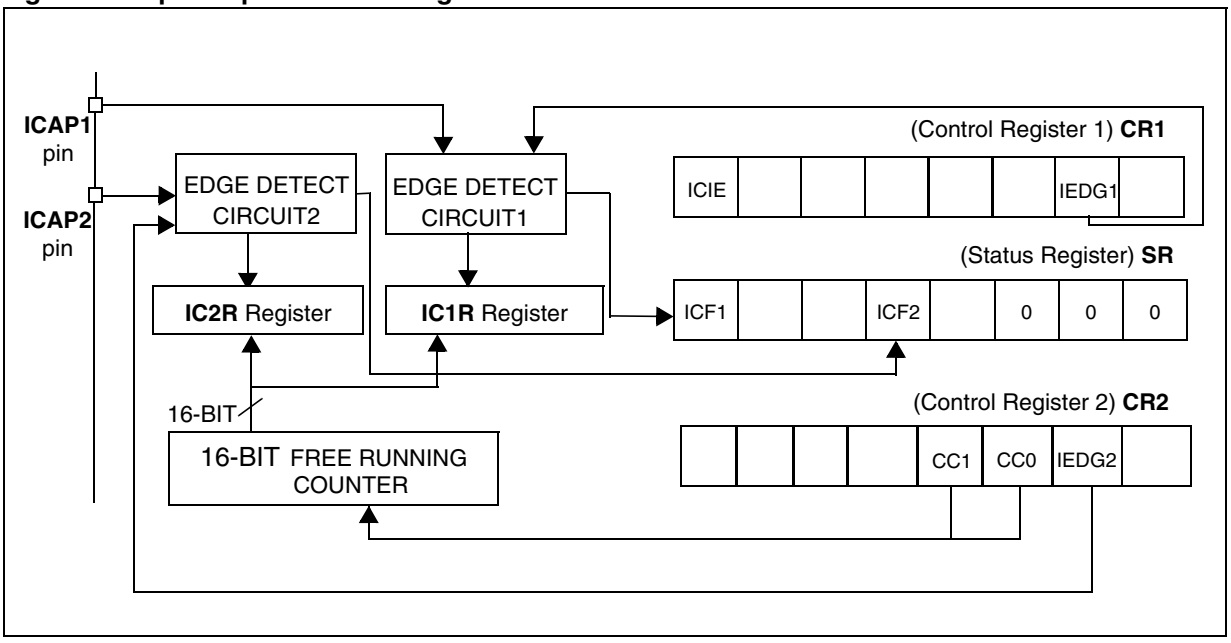
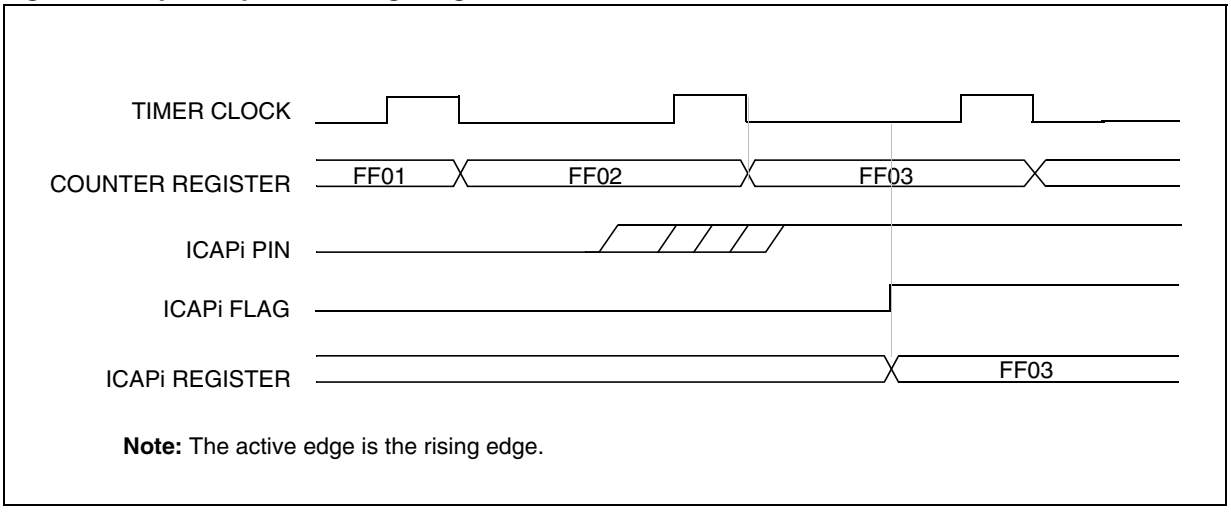


Figure 49. Input Capture Timing Diagram



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)

10.3.3.4 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 16 Clock Control Bits).

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 16 Clock Control Bits)

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:

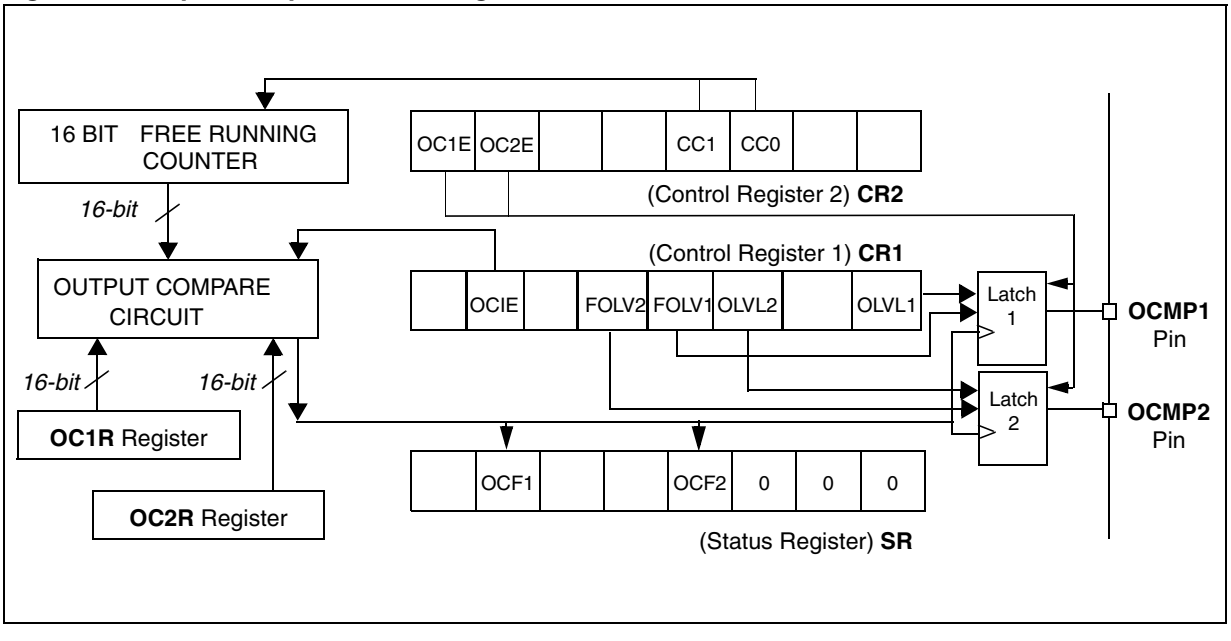
1. After a processor write cycle to the OC*i*HR register, the output compare function is inhibited until the OC*i*LR register is also written.
2. If the OC*i*E bit is not set, the OCMP*i* pin is a general I/O port and the OLVL*i* bit will not appear when a match is found but an interrupt could be generated if the OCIE bit is set.
3. In both internal and external clock modes, OCFi and OCMPi are set while the counter value equals the OCiR register value (see Figure 51 for an example with $f_{CPU/2}$ and Figure 52 for an example with $f_{CPU/4}$). This behavior is the same in OPM or PWM mode.
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 50. Output Compare Block Diagram



16-BIT TIMER (Cont'd)

Figure 51. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/2$

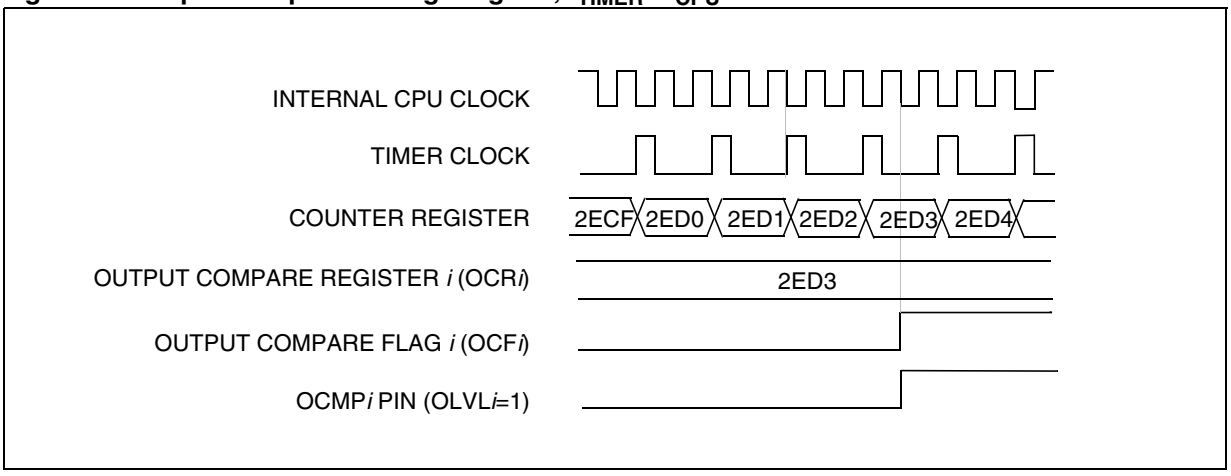
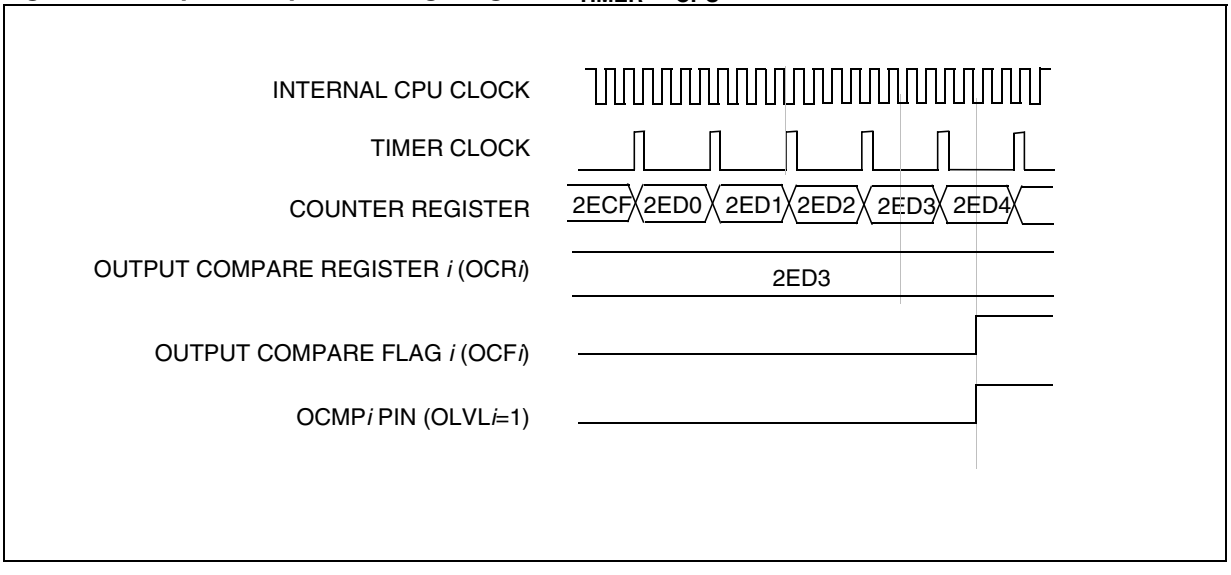


Figure 52. Output Compare Timing Diagram, $f_{TIMER} = f_{CPU}/4$



16-BIT TIMER (Cont'd)

10.3.3.5 One Pulse Mode

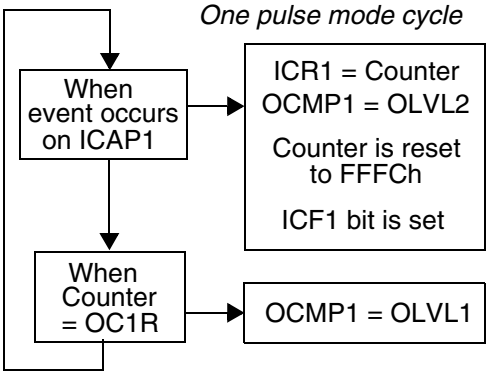
One Pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

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 16 Clock Control Bits).



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 16 Clock Control Bits)

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)

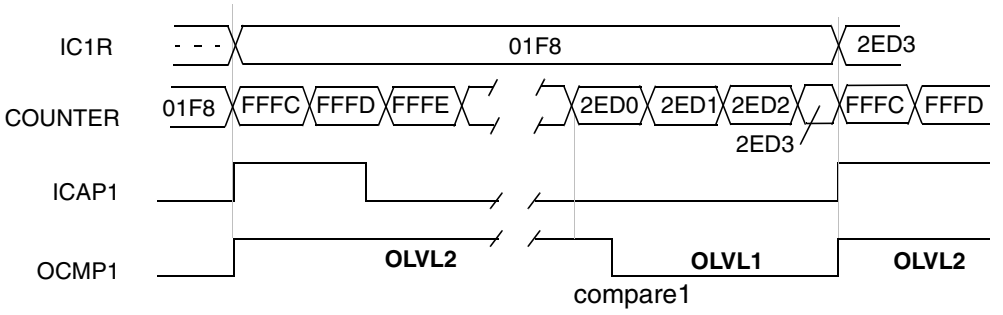
When the value of the counter is equal to the value of the contents of the OC1R register, the OLVL1 bit is output on the OCMP1 pin, (See Figure 53).

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.

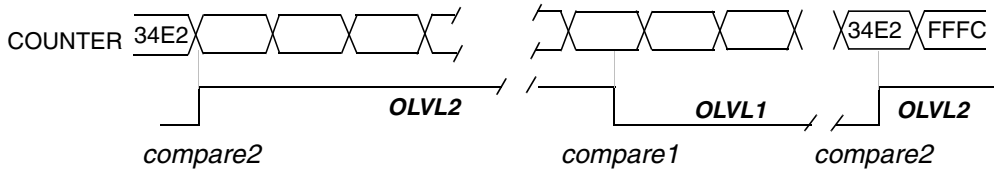
16-BIT TIMER (Cont'd)

Figure 53. One Pulse Mode Timing Example



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

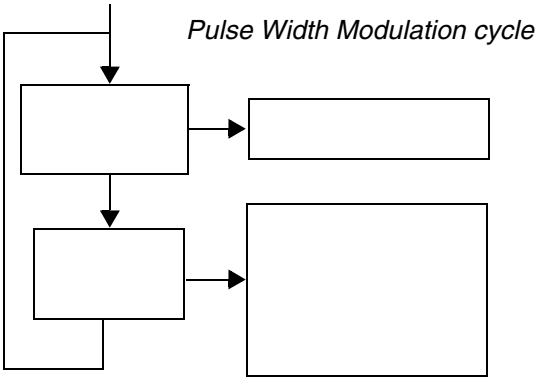
Figure 54. Pulse Width Modulation Mode Timing Example



Note:

16-BIT TIMER

10.3.3.6 Pulse Width Modulation Mode



Note:

i

i

Procedure

i

Notes:

16-BIT TIMER

10.3.4 Low Power Modes

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.

10.3.5 Interrupts

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:

10.3.6 Summary of Timer modes

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

[3)] See note 4 in Section 10.3.3.6 Pulse Width Modulation Mode

(Cont'd)

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.

Read/Write

Reset Value: 0000 0000 (00h)

7							0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1

Bit 7 = *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 = *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 = *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 = *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 = *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 = *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 = *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 = *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.

(Cont'd)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
OC1E	OC2E	OPM	PWM	CC1	CC0	IEDG2	EXEDG

Bit 7 = *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 = *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 = *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 = *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 = *Clock Control.*
The timer clock mode depends on these bits:

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

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

Bit 1 = *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 = *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.

(Cont'd)

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 =
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 =
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 =
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.

Reading or writing the ACLR register does not clear TOF.

Bit 4 =
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 =
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.

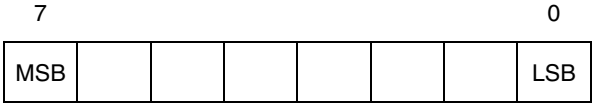
Bit 2 =
This bit is set and cleared by software. When set, it freezes the timer prescaler and counter and disabled the output functions (OCMP1 and OCMP2 pins) to reduce power consumption. Access to the timer registers is still available, allowing the timer configuration to be changed while it is disabled.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

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

(Cont'd)

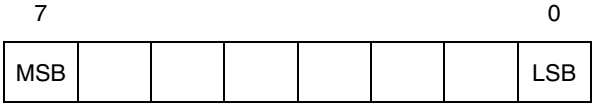
Read Only
Reset Value: Undefined

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



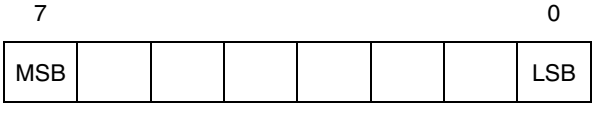
Read/Write
Reset Value: 1000 0000 (80h)

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



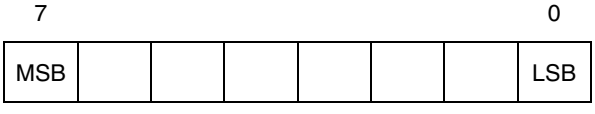
Read Only
Reset Value: Undefined

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



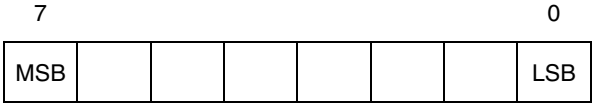
Read/Write
Reset Value: 0000 0000 (00h)

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



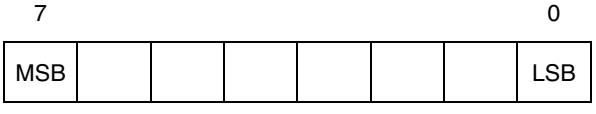
Read/Write
Reset Value: 1000 0000 (80h)

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



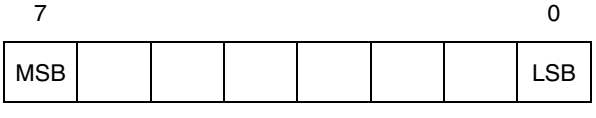
Read/Write
Reset Value: 0000 0000 (00h)

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



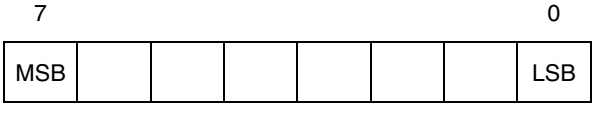
Read Only
Reset Value: 1111 1111 (FFh)

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



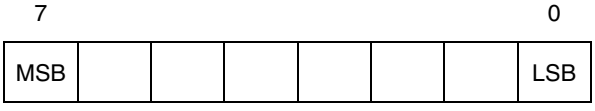
Read Only
Reset Value: 1111 1100 (FCh)

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.



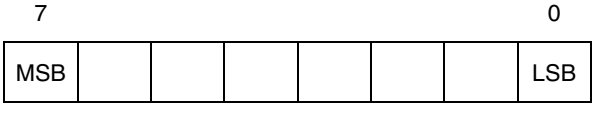
Read Only
Reset Value: 1111 1111 (FFh)

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



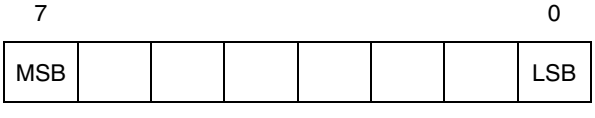
Read Only
Reset Value: 1111 1100 (FCh)

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.



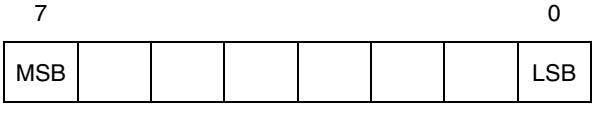
Read Only
Reset Value: Undefined

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



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).



(Cont'd)

Timer A: 32 Timer B: 42	Reset Value	ICIE 0	OCIE 0	TOIE 0	FOLV2 0	FOLV1 0	OLVL2 0	IEDG1 0	OLVL1 0
Timer A: 31 Timer B: 41	Reset Value	OC1E 0	OC2E 0	OPM 0	PWM 0	CC1 0	CC0 0	IEDG2 0	EXEDG 0
Timer A: 33 Timer B: 43	Reset Value	ICF1 0	OCF1 0	TOF 0	ICF2 0	OCF2 0	TIMD 0	- 0	- 0
Timer A: 34 Timer B: 44	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 35 Timer B: 45	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 36 Timer B: 46	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 37 Timer B: 47	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3E Timer B: 4E	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3F Timer B: 4F	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 38 Timer B: 48	Reset Value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 39 Timer B: 49	Reset Value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3A Timer B: 4A	Reset Value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 3B Timer B: 4B	Reset Value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3C Timer B: 4C	Reset Value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3D Timer B: 4D	Reset Value	MSB -	-	-	-	-	-	-	LSB -

(cont'd)

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.

- 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

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.

Figure 55 on page 96 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:

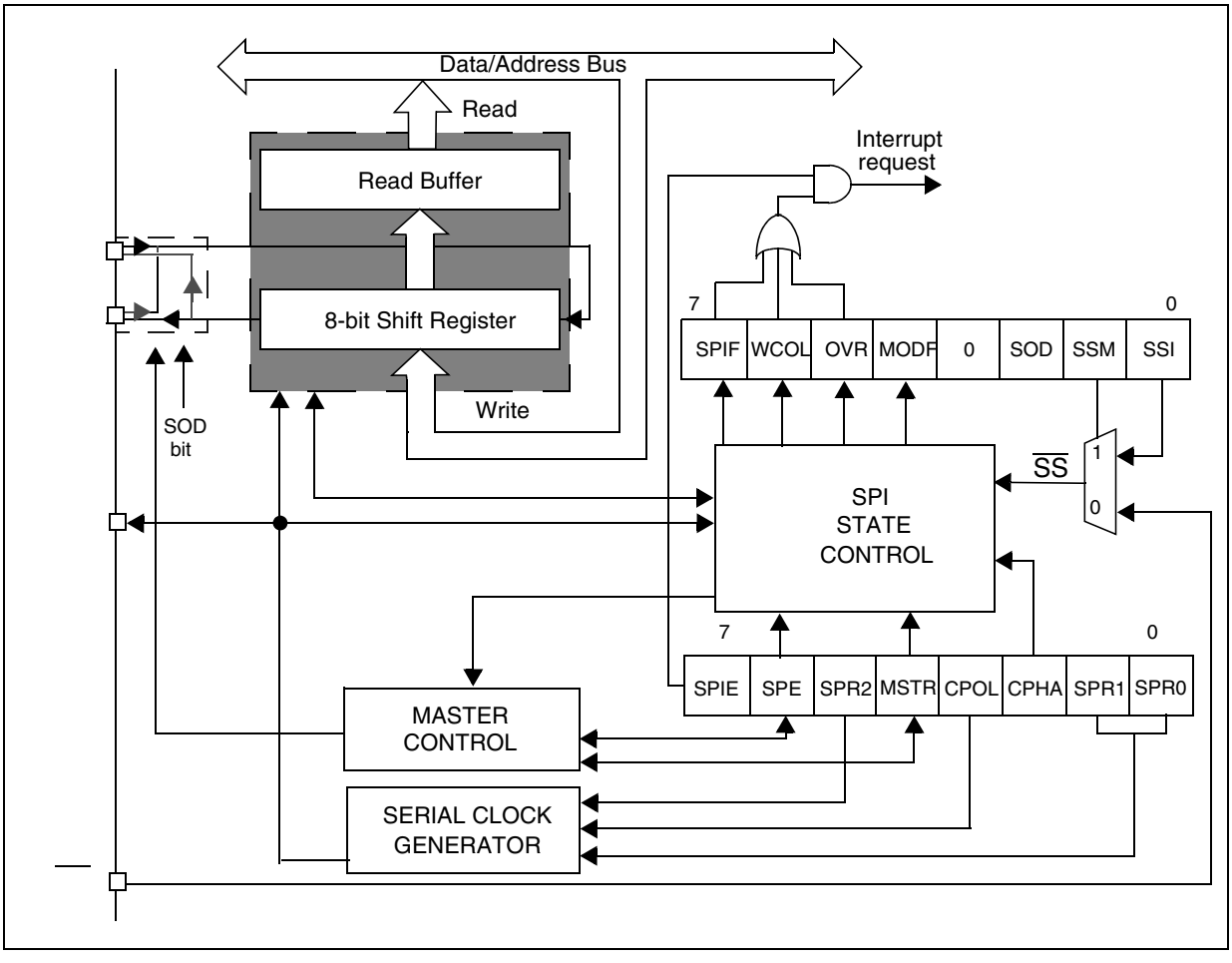
– MISO: Master In / Slave Out data

– 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.

(cont'd)


Data/Address Bus
Read
Read Buffer
8-bit Shift Register
Write
SOD
bit
Interrupt
request
7
0
SPIF
WCOL
OVR
MODF
0
SOD
SSM
SSI
SS
1
0
SPI
STATE
CONTROL
7
0
SPIE
SPE
SPR2
MSTR
CPOL
CPHA
SPR1
SPR0
MASTER
CONTROL
SERIAL CLOCK
GENERATOR

(cont'd)

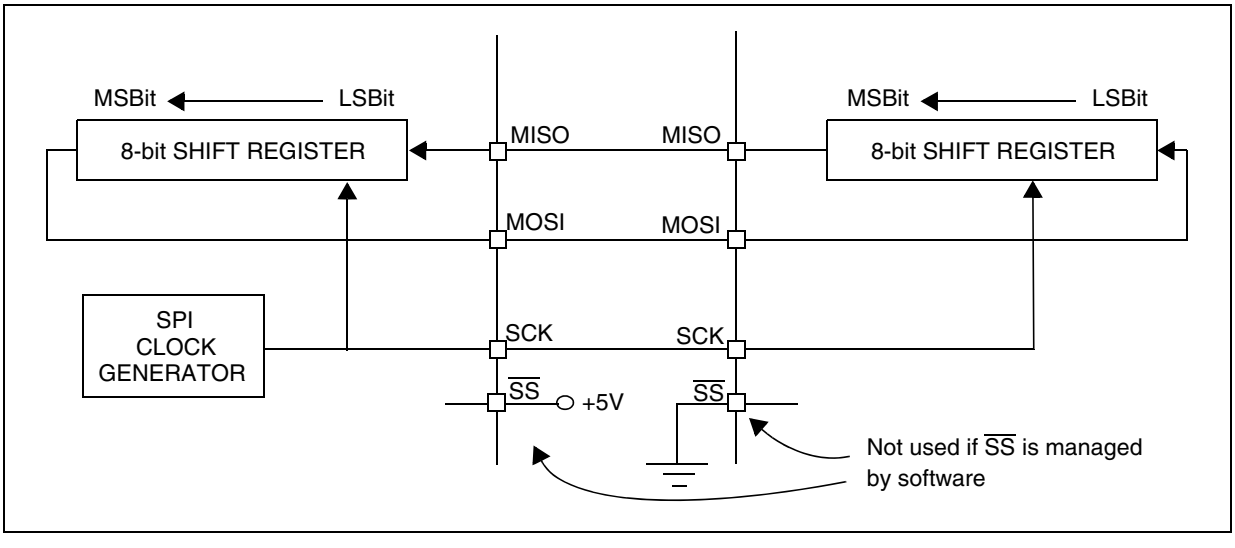
A basic example of interconnections between a single master and a single slave is illustrated in Figure 56.

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 59 on page 100) but master and slave must be programmed with the same timing mode.



(cont'd)

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 58).

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.

– $\overline{SS}$ internal must be held high continuously

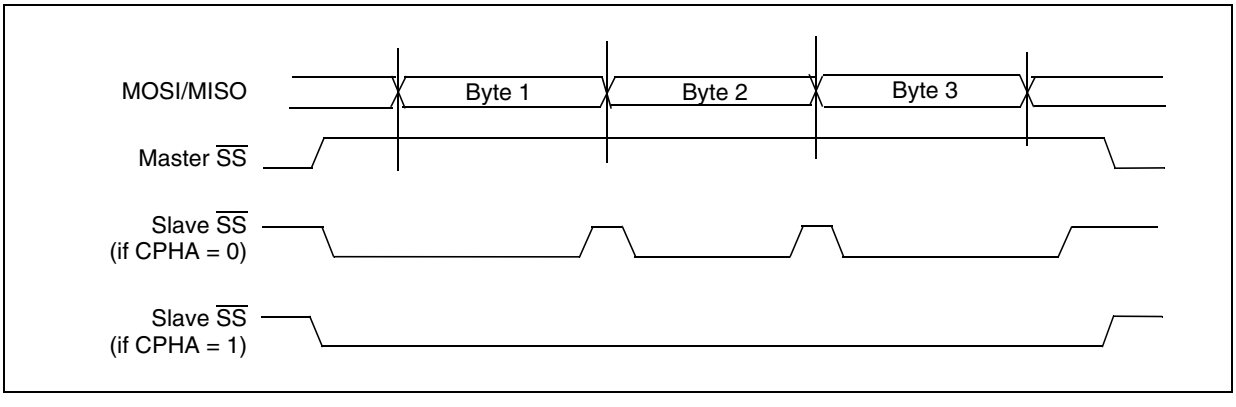
There are two cases depending on the data/clock timing relationship (see Figure 57):

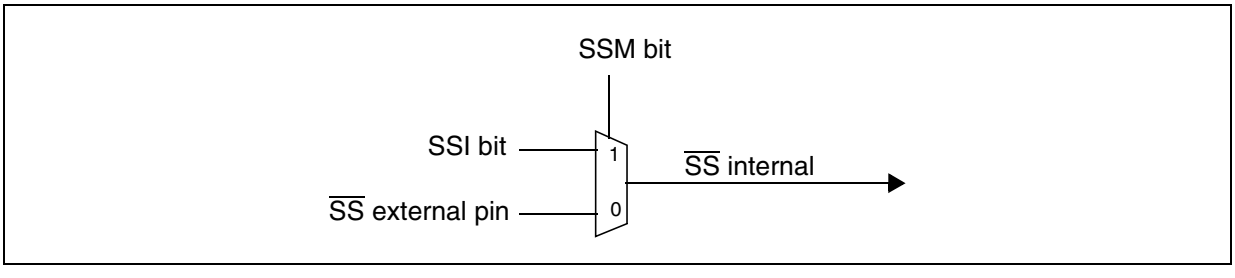
If CPHA = 1 (data latched on second clock edge):

– $\overline{SS}$ 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 10.4.5.3).





(cont'd)

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).

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).

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 59 shows the four possible configurations.
     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
     MSTR and SPE bits remain set only if SS is high).

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.

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

While the SPIF bit is set, all writes to the SPIDR register are inhibited until the SPICSR register is read.

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 59).
     The slave must have the same CPOL and CPHA settings as the master.
   - Manage the SS pin as described in Section 10.4.3.2 and Figure 57. If CPHA = 1 SS must be held low continuously. If CPHA = 0 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.

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

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 10.4.5.2).

(cont'd)

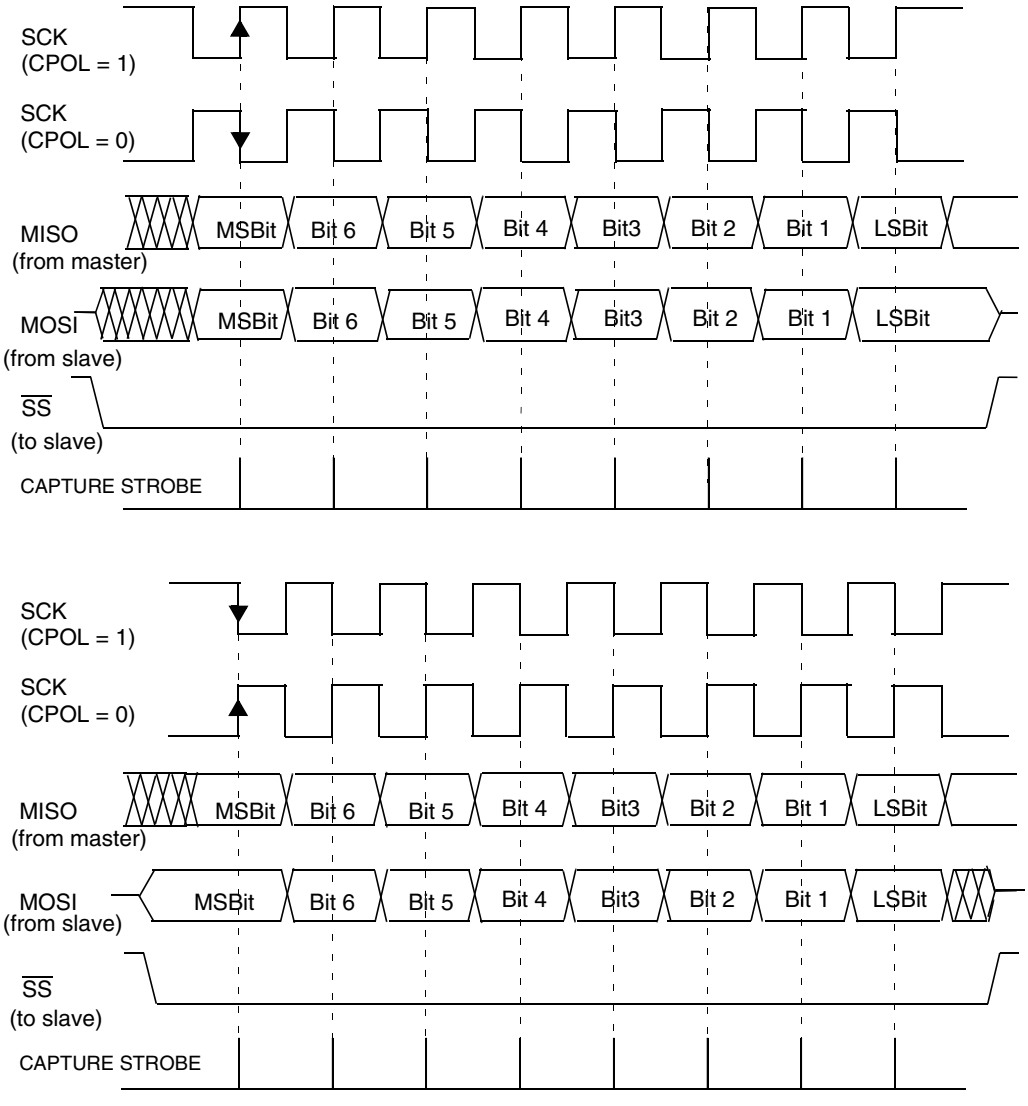
Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (See Figure 59).

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 59 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.

: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.



Note: This figure should not be used as a replacement for parametric information. Refer to the Electrical Characteristics chapter.

SERIAL PERIPHERAL INTERFACE (cont'd)

10.4.5 Error Flags

10.4.5.1 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.

10.4.5.2 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.

10.4.5.3 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 10.4.3.2 Slave Select Management.

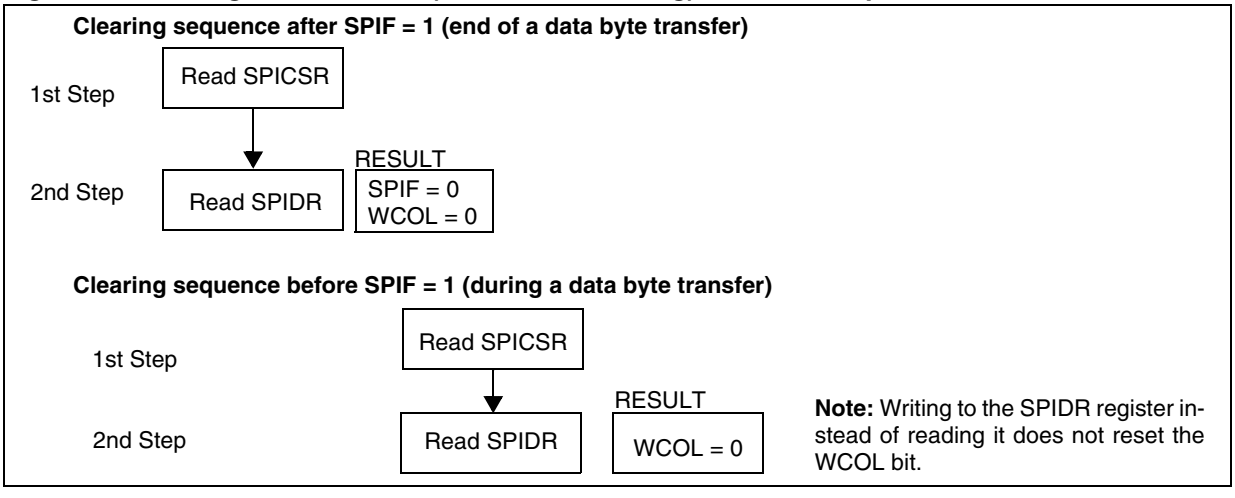
Note: A "read collision" will never occur since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

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 60).

Figure 60. Clearing the WCOL Bit (Write Collision Flag) Software Sequence



SERIAL PERIPHERAL INTERFACE (cont'd)

10.4.5.4 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 61).

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.

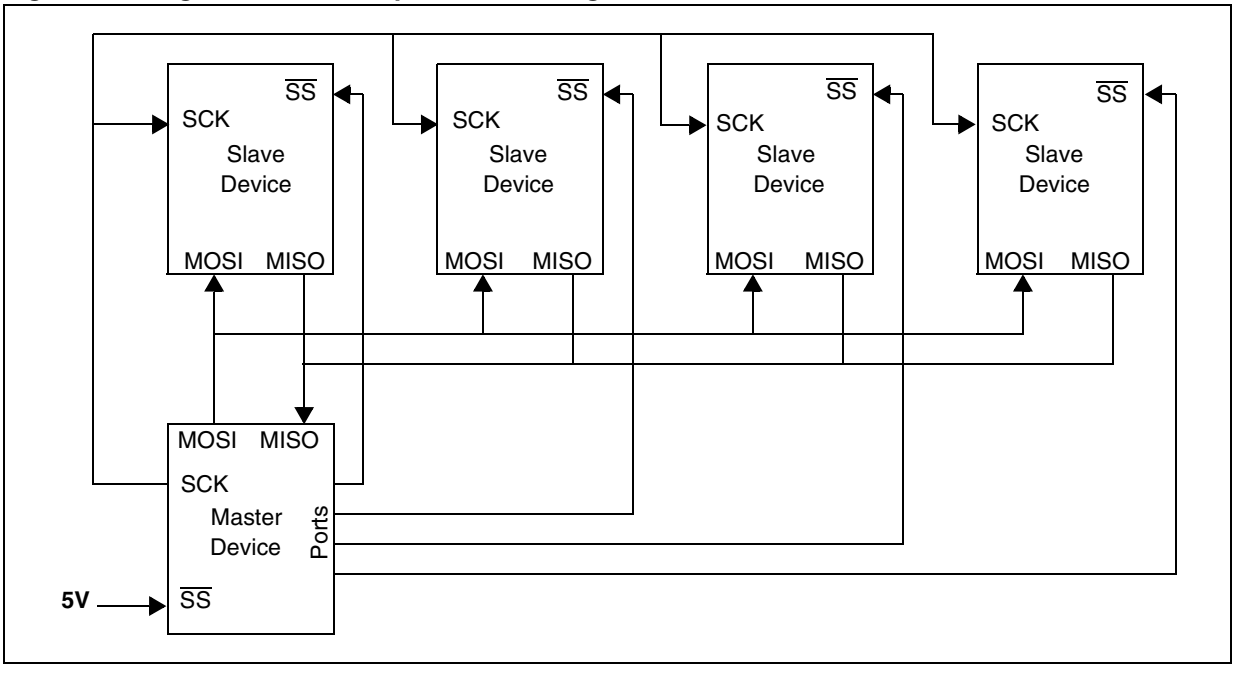
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

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 61. Single Master / Multiple Slave Configuration



SERIAL PERIPHERAL INTERFACE (cont'd)

10.4.6 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.

10.4.6.1 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 10.4.3.2), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters Halt mode.

10.4.7 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).

10.4.8 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**
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**
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see Section 10.4.5.1 Master Mode Fault (MODF)). 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**
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 18 SPI Master Mode SCK Frequency.
0: Divider by 2 enabled
1: Divider by 2 disabled

Note: This bit has no effect in slave mode.

Bit 4 = **MSTR**
This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see Section 10.4.5.1 Master Mode Fault (MODF)).
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**
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**
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]**
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 18. 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**

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**
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 60).
0: No write collision occurred
1: A write collision has been detected

Bit 5 = **OVR** S
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 10.4.5.2). 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**
This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see Section 10.4.5.1 Master Mode Fault (MODF)). 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**
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**
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 10.4.3.2 Slave Select Management.
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**
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 55).

SERIAL PERIPHERAL INTERFACE (Cont'd)

Table 19. SPI Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
0021h	**SPIDR**	MSB							LSB
	Reset Value	x	x	x	x	x	x	x	x
0022h	**SPICR**	SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR1	SPR0
	Reset Value	0	0	0	0	x	x	x	x
0023h	**SPICSR**	SPIF	WCOL	OVR	MODF		SOD	SSM	SSI
	Reset Value	0	0	0	0	0	0	0	0

10.5 LINSCI SERIAL COMMUNICATION INTERFACE (LIN MASTER/SLAVE)

10.5.1 Introduction

The Serial Communications Interface (SCI) offers a flexible means of full-duplex data exchange with external equipment requiring an industry standard NRZ asynchronous serial data format. The SCI offers a very wide range of baud rates using two baud rate generator systems.

The LIN-dedicated features support the LIN (Local Interconnect Network) protocol for both master and slave nodes.

This chapter is divided into SCI Mode and LIN mode sections. For information on general SCI communications, refer to the SCI mode section. For LIN applications, refer to both the SCI mode and LIN mode sections.

10.5.2 SCI Features

- Full duplex, asynchronous communications
- NRZ standard format (Mark/Space)
- Independently programmable transmit and receive baud rates up to 500K baud
- Programmable data word length (8 or 9 bits)
- Receive buffer full, Transmit buffer empty and End of Transmission flags
- 2 receiver wake-up modes:
  - Address bit (MSB)
  - Idle line
- Muting function for multiprocessor configurations
- Separate enable bits for Transmitter and Receiver
- Overrun, Noise and Frame error detection
- 6 interrupt sources
  - Transmit data register empty
  - Transmission complete
  - Receive data register full
  - Idle line received
  - Overrun error
  - Parity interrupt
- Parity control:
  - Transmits parity bit
  - Checks parity of received data byte
- Reduced power consumption mode

10.5.3 LIN Features

– LIN Master
  – 13-bit LIN Synch Break generation
– LIN Slave
  – Automatic Header Handling
  – Automatic baud rate resynchronization based on recognition and measurement of the LIN Synch Field (for LIN slave nodes)
  – Automatic baud rate adjustment (at CPU frequency precision)
  – 11-bit LIN Synch Break detection capability
  – LIN Parity check on the LIN Identifier Field (only in reception)
  – LIN Error management
  – LIN Header Timeout
  – Hot plugging support

LINSCI™ SERIAL COMMUNICATION INTERFACE (cont'd)

10.5.4 General Description

The interface is externally connected to another device by two pins:

– TDO: Transmit Data Output. When the transmitter is disabled, the output pin returns to its I/O port configuration. When the transmitter is enabled and nothing is to be transmitted, the TDO pin is at high level.

– RDI: Receive Data Input is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

Through these pins, serial data is transmitted and received as characters comprising:

– An Idle Line prior to transmission or reception

– A start bit

– A data word (8 or 9 bits) least significant bit first

– A Stop bit indicating that the character is complete

This interface uses three types of baud rate generator:

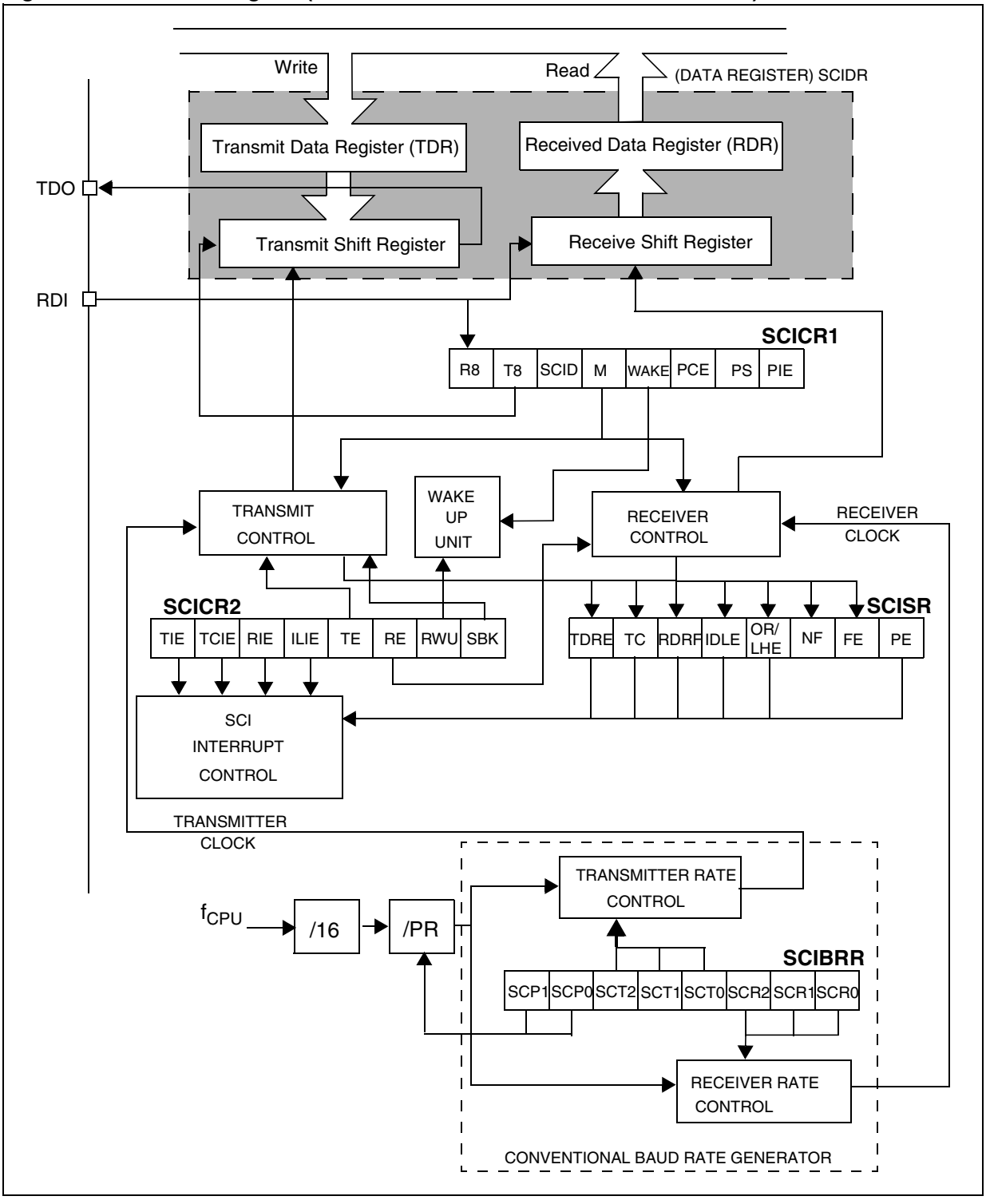
– A conventional type for commonly-used baud rates

– An extended type with a prescaler offering a very wide range of baud rates even with non-standard oscillator frequencies

– A LIN baud rate generator with automatic resynchronization

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

Figure 62. SCI Block Diagram (in Conventional Baud Rate Generator Mode)



LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.5 SCI Mode - Functional Description

Conventional Baud Rate Generator Mode

The block diagram of the Serial Control Interface in conventional baud rate generator mode is shown in Figure 62.

It uses four registers:

– 2 control registers (SCICR1 and SCICR2)

– A status register (SCISR)

– A baud rate register (SCIBRR)

Extended Prescaler Mode

Two additional prescalers are available in extended prescaler mode. They are shown in Figure 64.

– An extended prescaler receiver register (SCIERPR)

– An extended prescaler transmitter register (SCIETPR)

10.5.5.1 Serial Data Format

Word length may be selected as being either 8 or 9 bits by programming the M bit in the SCICR1 register (see Figure 63).

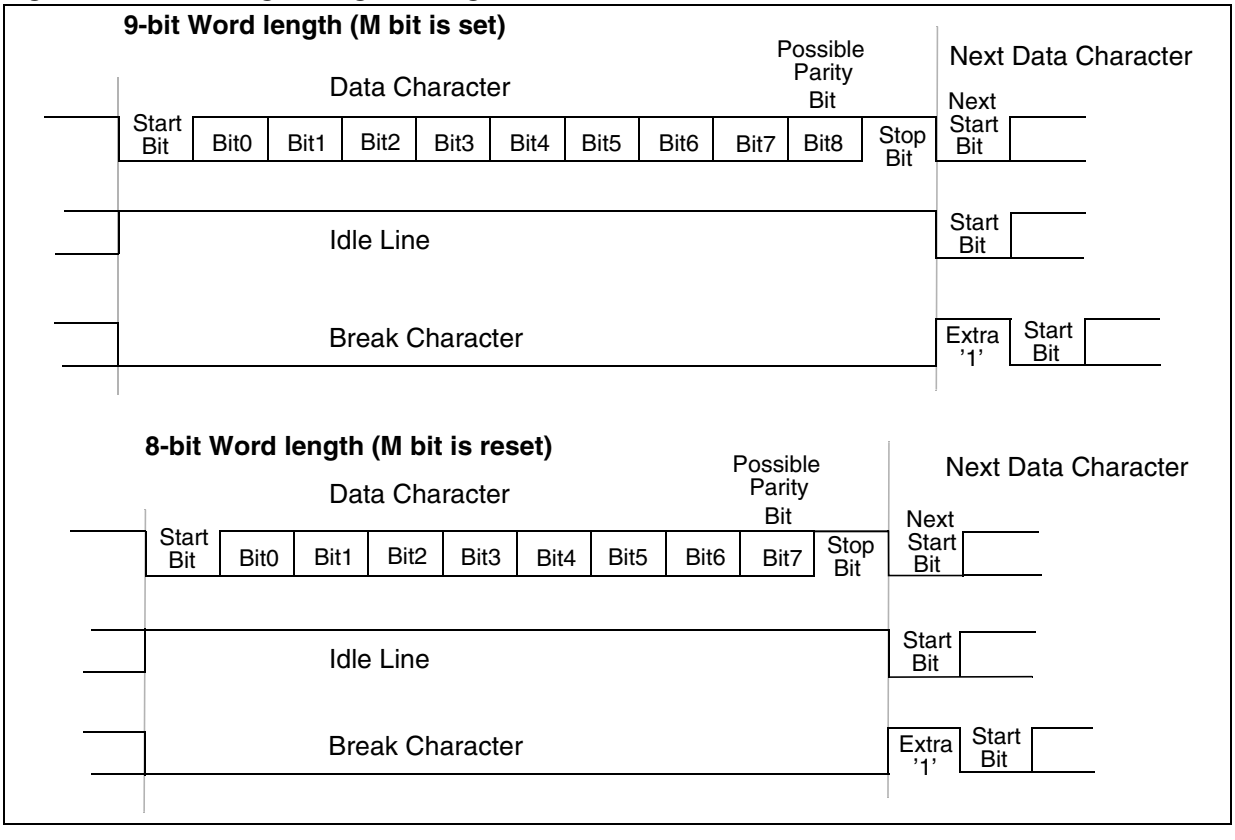
The TDO pin is in low state during the start bit.

The TDO pin is in high state during the stop bit.

An Idle character is interpreted as a continuous logic high level for 10 (or 11) full bit times.

A Break character is a character with a sufficient number of low level bits to break the normal data format followed by an extra "1" bit to acknowledge the start bit.

Figure 63. Word Length Programming



LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.5.2 Transmitter

The transmitter can send data words of either 8 or 9 bits depending on the M bit status. When the M bit is set, word length is 9 bits and the 9th bit (the MSB) has to be stored in the T8 bit in the SCICR1 register.

Character Transmission

During an SCI transmission, data shifts out least significant bit first on the TDO pin. In this mode, the SCIDR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 62).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the SCIBRR and the SCIETPR registers.

– Set the TE bit to send a preamble of 10 (M = 0) or 11 (M = 1) consecutive ones (Idle Line) as first transmission.

– Access the SCISR register and write the data to send in the SCIDR register (this sequence clears the TDRE bit). Repeat this sequence for each data to be transmitted.

Clearing the TDRE bit is always performed by the following software sequence:
1. An access to the SCISR register
2. A write to the SCIDR register

The TDRE bit is set by hardware and it indicates:

– The TDR register is empty.

– The data transfer is beginning.

– The next data can be written in the SCIDR register without overwriting the previous data.

This flag generates an interrupt if the TIE bit is set and the I[l1:0] bits are cleared in the CCR register.

When a transmission is taking place, a write instruction to the SCIDR register stores the data in the TDR register and which is copied in the shift register at the end of the current transmission.

When no transmission is taking place, a write instruction to the SCIDR register places the data directly in the shift register, the data transmission starts, and the TDRE bit is immediately set.

When a character transmission is complete (after the stop bit) the TC bit is set and an interrupt is generated if the TCIE is set and the I[1:0] bits are cleared in the CCR register.

Clearing the TC bit is performed by the following software sequence:
1. An access to the SCISR register
2. A write to the SCIDR register

Note: The TDRE and TC bits are cleared by the same software sequence.

Break Characters

Setting the SBK bit loads the shift register with a break character. The break character length depends on the M bit (see Figure 63).

As long as the SBK bit is set, the SCI sends break characters to the TDO pin. After clearing this bit by software, the SCI inserts a logic 1 bit at the end of the last break character to guarantee the recognition of the start bit of the next character.

Idle Line

Setting the TE bit drives the SCI to send a preamble of 10 (M = 0) or 11 (M = 1) consecutive '1's (idle line) before the first character.

In this case, clearing and then setting the TE bit during a transmission sends a preamble (idle line) after the current word. Note that the preamble duration (10 or 11 consecutive '1's depending on the M bit) does not take into account the stop bit of the previous character.

Note: Resetting and setting the TE bit causes the data in the TDR register to be lost. Therefore the best time to toggle the TE bit is when the TDRE bit is set, that is, before writing the next byte in the SCIDR.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.5.3 Receiver

The SCI can receive data words of either 8 or 9 bits. When the M bit is set, word length is 9 bits and the MSB is stored in the R8 bit in the SCICR1 register.

Character reception

During a SCI reception, data shifts in least significant bit first through the RDI pin. In this mode, the SCIDR register consists or a buffer (RDR) between the internal bus and the received shift register (see Figure 62).

Procedure

– Select the M bit to define the word length.

– Select the desired baud rate using the SCIBRR and the SCIERPR registers.

– Set the RE bit, this enables the receiver which begins searching for a start bit.

When a character is received:

– The RDRF bit is set. It indicates that the content of the shift register is transferred to the RDR.

– An interrupt is generated if the RIE bit is set and the I[1:0] bits are cleared in the CCR register.

– The error flags can be set if a frame error, noise or an overrun error has been detected during reception.

Clearing the RDRF bit is performed by the following software sequence done by:

1. An access to the SCISR register

2. A read to the SCIDR register.

The RDRF bit must be cleared before the end of the reception of the next character to avoid an overrun error.

Idle Line

When an idle line is detected, there is the same procedure as a data received character plus an interrupt if the ILIE bit is set and the I[1:0] bits are cleared in the CCR register.

Overrun Error

An overrun error occurs when a character is received when RDRF has not been reset. Data can not be transferred from the shift register to the TDR register as long as the RDRF bit is not cleared.

When an overrun error occurs:

– The OR bit is set.

– The RDR content will not be lost.

– The shift register will be overwritten.

– An interrupt is generated if the RIE bit is set and the I[1:0] bits are cleared in the CCR register.

The OR bit is reset by an access to the SCISR register followed by a SCIDR register read operation.

Noise Error

Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.

When noise is detected in a character:

– The NF bit is set at the rising edge of the RDRF bit.

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The NF bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

Framing Error

A framing error is detected when:

– The stop bit is not recognized on reception at the expected time, following either a desynchronization or excessive noise.

– A break is received.

When the framing error is detected:

– the FE bit is set by hardware

– Data is transferred from the Shift register to the SCIDR register.

– No interrupt is generated. However this bit rises at the same time as the RDRF bit which itself generates an interrupt.

The FE bit is reset by a SCISR register read operation followed by a SCIDR register read operation.

Break Character

– When a break character is received, the SCI handles it as a framing error. To differentiate a break character from a framing error, it is necessary to read the SCIDR. If the received value is 00h, it is a break character. Otherwise it is a framing error.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.5.4 Conventional Baud Rate Generation

The baud rates for the receiver and transmitter (Rx and Tx) are set independently and calculated as follows:

$$Tx = \frac{f_{CPU}}{(16*PR)*TR} \qquad Rx = \frac{f_{CPU}}{(16*PR)*RR}$$

with:

PR = 1, 3, 4 or 13 (see SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64,128

(see SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64,128

(see SCR[2:0] bits)

All these bits are in the SCIBRR register.

Example: If f_{CPU} is 8 MHz (normal mode) and if PR = 13 and TR = RR = 1, the transmit and receive baud rates are 38400 baud.

Note: The baud rate registers MUST NOT be changed while the transmitter or the receiver is enabled.

10.5.5.5 Extended Baud Rate Generation

The extended prescaler option gives a very fine tuning on the baud rate, using a 255 value prescaler, whereas the conventional Baud Rate Generator retains industry standard software compatibility.

The extended baud rate generator block diagram is described in Figure 64.

The output clock rate sent to the transmitter or to the receiver will be the output from the 16 divider divided by a factor ranging from 1 to 255 set in the SCIERPR or the SCIETPR register.

Note: The extended prescaler is activated by setting the SCIETPR or SCIERPR register to a value other than zero. The baud rates are calculated as follows:

$$Tx = \frac{f_{CPU}}{16*ETPR*(PR*TR)} \qquad Rx = \frac{f_{CPU}}{16*ERPR*(PR*RR)}$$

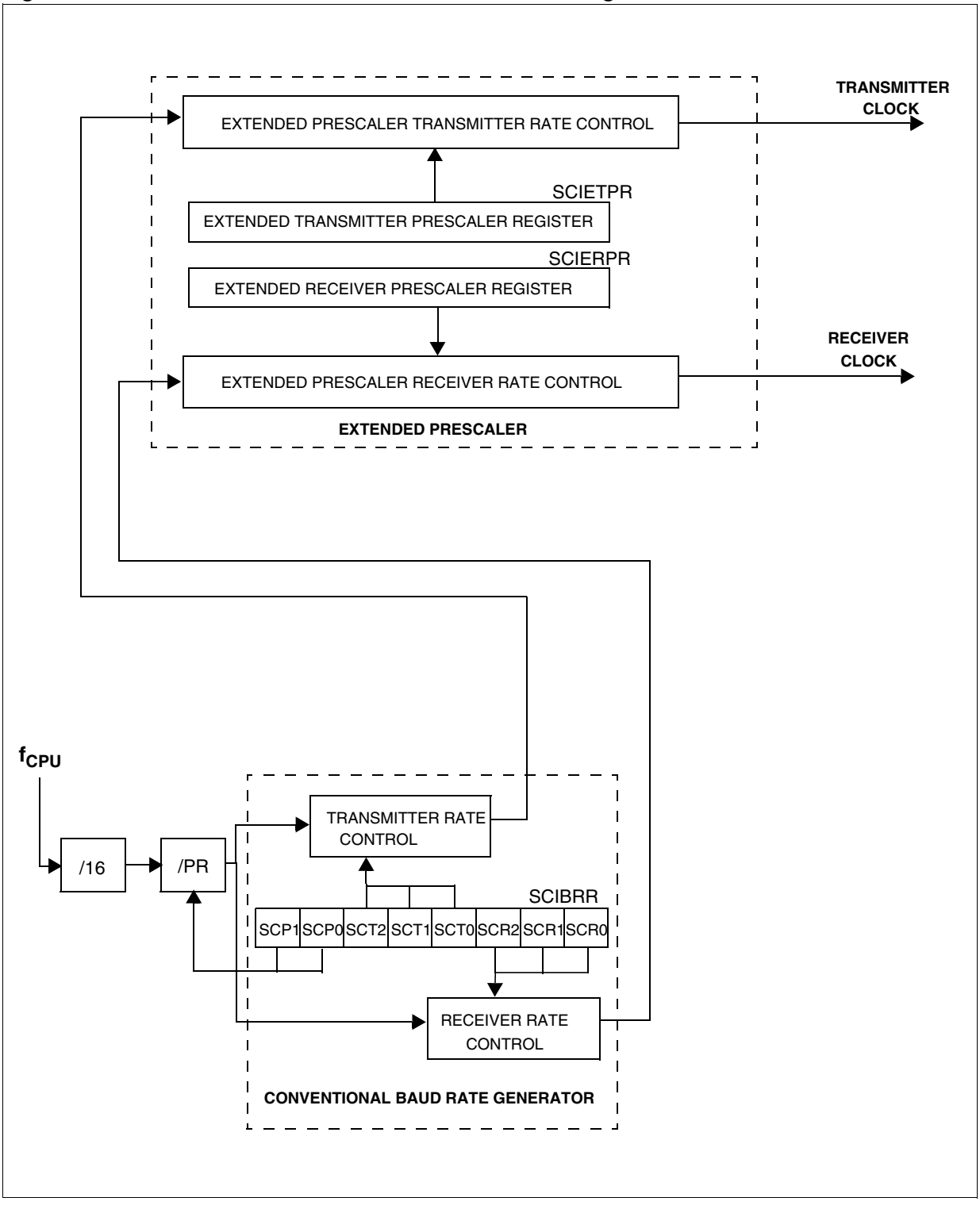
with:

ETPR = 1, ..., 255 (see SCIETPR register)

ERPR = 1, ..., 255 (see SCIERPR register)

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

Figure 64. SCI Baud Rate and Extended Prescaler Block Diagram



LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.5.6 Receiver Muting and Wake-up Feature

In multiprocessor configurations it is often desirable that only the intended message recipient should actively receive the full message contents, thus reducing redundant SCI service overhead for all non-addressed receivers.

The non-addressed devices may be placed in sleep mode by means of the muting function.

Setting the RWU bit by software puts the SCI in sleep mode:

All the reception status bits can not be set.

All the receive interrupts are inhibited.

A muted receiver may be woken up in one of the following ways:

– by Idle Line detection if the WAKE bit is reset,

– by Address Mark detection if the WAKE bit is set.

Idle Line Detection

Receiver wakes up by Idle Line detection when the Receive line has recognized an Idle Line. Then the RWU bit is reset by hardware but the IDLE bit is not set.

This feature is useful in a multiprocessor system when the first characters of the message determine the address and when each message ends by an idle line: As soon as the line becomes idle, every receivers is waken up and analyse the first characters of the message which indicates the addressed receiver. The receivers which are not addressed set RWU bit to enter in mute mode. Consequently, they will not treat the next characters constituting the next part of the message. At the end of the message, an idle line is sent by the transmitter: this wakes up every receivers which are ready to analyse the addressing characters of the new message.

In such a system, the inter-characters space must be smaller than the idle time.

Address Mark Detection

Receiver wakes up by Address Mark detection when it received a "1" as the most significant bit of a word, thus indicating that the message is an address. The reception of this particular word wakes up the receiver, resets the RWU bit and sets the RDRF bit, which allows the receiver to receive this word normally and to use it as an address word.

This feature is useful in a multiprocessor system when the most significant bit of each character (except for the break character) is reserved for Address Detection. As soon as the receivers received an address character (most significant bit = '1'), the receivers are waken up. The receivers which are not addressed set RWU bit to enter in mute mode. Consequently, they will not treat the next characters constituting the next part of the message.

10.5.5.7 Parity Control

Hardware byte Parity control (generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PCE bit in the SCICR1 register. Depending on the character format defined by the M bit, the possible SCI character formats are as listed in Table 20.

Note: In case of wake-up by an address mark, the MSB bit of the data is taken into account and not the parity bit

Table 20. Character Formats

M bit	PCE bit	Character format
0	0	ǀ SB ǀ 8 bit data ǀ STB ǀ
	1	ǀ SB ǀ 7-bit data ǀ PB ǀ STB ǀ
1	0	ǀ SB ǀ 9-bit data ǀ STB ǀ
	1	ǀ SB ǀ 8-bit data ǀ PB ǀ STB ǀ

Legend: SB = Start Bit, STB = Stop Bit, PB = Parity Bit

Even parity: The parity bit is calculated to obtain an even number of "1s" inside the character made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit will be 0 if even parity is selected (PS bit = 0).

Odd parity: The parity bit is calculated to obtain an odd number of "1s" inside the character made of the 7 or 8 LSB bits (depending on whether M is equal to 0 or 1) and the parity bit.

Example: data = 00110101; 4 bits set => parity bit will be 1 if odd parity is selected (PS bit = 1).

Transmission mode: If the PCE bit is set then the MSB bit of the data written in the data register is not transmitted but is changed by the parity bit.

Reception mode: If the PCE bit is set then the interface checks if the received data byte has an even number of "1s" if even parity is selected (PS = 0) or an odd number of "1s" if odd parity is selected (PS = 1). If the parity check fails, the PE flag is set in the SCISR register and an interrupt is generated if PCIE is set in the SCICR1 register.

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.6 Low Power Modes

Mode	Description
Wait	No effect on SCI. SCI interrupts cause the device to exit from Wait mode.
Halt	SCI registers are frozen. In Halt mode, the SCI stops transmitting/receiving until Halt mode is exited.

10.5.7 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Transmit Data Register Empty	TDRE	TIE	Yes	No
Transmission Complete	TC	TCIE		
Received Data Ready to be Read	RDRF	RIE		
Overrun Error or LIN Synch Error Detected	OR/ LHE			
Idle Line Detected	IDLE	ILIE		
Parity Error	PE	PIE		
LIN Header Detection	LHDF	LHIE		

The SCI 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).

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

10.5.8 SCI Mode Register Description

STATUS REGISTER (SCISR)
Read Only
Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	OR[1)]	NF[1)]	FE[1)]	PE[1)]

Bit 7 = **TDRE**
This bit is set by hardware when the content of the TDR register has been transferred into the shift register. An interrupt is generated if the TIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).
0: Data is not transferred to the shift register
1: Data is transferred to the shift register

Bit 6 = **TC**
This bit is set by hardware when transmission of a character containing Data is complete. An interrupt is generated if TCIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a write to the SCIDR register).
0: Transmission is not complete
1: Transmission is complete

Note: TC is not set after the transmission of a Preamble or a Break.

Bit 5 = **RDRF**
This bit is set by hardware when the content of the RDR register has been transferred to the SCIDR register. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: Data is not received
1: Received data is ready to be read

Bit 4 = **IDLE**
This bit is set by hardware when an Idle Line is detected. An interrupt is generated if the ILIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Idle Line is detected
1: Idle Line is detected

Note: The IDLE bit will not be set again until the RDRF bit has been set itself (that is, a new idle line occurs).

Bit 3 = **OR**
The OR bit is set by hardware when the word currently being received in the shift register is ready to be transferred into the RDR register whereas RDRF is still set. An interrupt is generated if RIE = 1 in the SCICR2 register. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Overrun error
1: Overrun error detected

Note: When this bit is set, RDR register contents will not be lost but the shift register will be overwritten.

Bit 2 = **NF** *Character Noise flag*
This bit is set by hardware when noise is detected on a received character. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No noise
1: Noise is detected

Note: This bit does not generate interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt.

Bit 1 = **FE** *Framing error*
This bit is set by hardware when a desynchronization, excessive noise or a break character is detected. It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Framing error
1: Framing error or break character detected

Note: This bit does not generate an interrupt as it appears at the same time as the RDRF bit which itself generates an interrupt. If the word currently being transferred causes both a frame error and an overrun error, it will be transferred and only the OR bit will be set.

Bit 0 = **PE** *Parity error*
This bit is set by hardware when a byte parity error occurs (if the PCE bit is set) in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.
0: No parity error
1: Parity error detected

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

CONTROL REGISTER 1 (SCICR1)

Read/Write
Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE[1)]	PS	PIE

[1)]This bit has a different function in LIN mode, please refer to the LIN mode register description.

Bit 7 = **R8** *Receive data bit 8*
This bit is used to store the 9th bit of the received word when M = 1.

Bit 6 = **T8** *Transmit data bit 8*
This bit is used to store the 9th bit of the transmitted word when M = 1.

Bit 5 = **SCID** *Disabled for low power consumption*
When this bit is set the SCI prescalers and outputs are stopped and the end of the current byte transfer in order to reduce power consumption.This bit is set and cleared by software.
0: SCI enabled
1: SCI prescaler and outputs disabled

Bit 4 = **M** *Word length*
This bit determines the word length. It is set or cleared by software.
0: 1 Start bit, 8 Data bits, 1 Stop bit
1: 1 Start bit, 9 Data bits, 1 Stop bit

Note: The M bit must not be modified during a data transfer (both transmission and reception).

Bit 3 = **WAKE** *Wake-Up method*
This bit determines the SCI Wake-Up method, it is set or cleared by software.
0: Idle Line
1: Address Mark

Note: If the LINE bit is set, the WAKE bit is deactivated and replaced by the LHDM bit.

Bit 2 = **PCE** *Parity control enable*
This bit is set and cleared by software. It selects the hardware parity control (generation and detection for byte parity, detection only for LIN parity).
0: Parity control disabled
1: Parity control enabled

Bit 1 = **PS** *Parity selection*
This bit selects the odd or even parity when the parity generation/detection is enabled (PCE bit set). It is set and cleared by software. The parity will be selected after the current byte.
0: Even parity
1: Odd parity

Bit 0 = **PIE** *Parity interrupt enable*
This bit enables the interrupt capability of the hardware parity control when a parity error is detected (PE bit set). The parity error involved can be a byte parity error (if bit PCE is set and bit LPE is reset) or a LIN parity error (if bit PCE is set and bit LPE is set).
0: Parity error interrupt disabled
1: Parity error interrupt enabled

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

CONTROL REGISTER 2 (SCICR2)

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

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU[1)]	SBK[1)]

[1)]This bit has a different function in LIN mode, please refer to the LIN mode register description.

Bit 7 = **TIE** *Transmitter interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: In SCI interrupt is generated whenever TDRE = 1 in the SCISR register

Bit 6 = **TCIE** *Transmission complete interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever TC = 1 in the SCISR register

Bit 5 = **RIE** *Receiver interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever OR = 1 or RDRF = 1 in the SCISR register

Bit 4 = **ILIE** *Idle line interrupt enable*
This bit is set and cleared by software.
0: Interrupt is inhibited
1: An SCI interrupt is generated whenever IDLE = 1 in the SCISR register.

Bit 3 = **TE** *Transmitter enable*
This bit enables the transmitter. It is set and cleared by software.
0: Transmitter is disabled
1: Transmitter is enabled

Notes:

– During transmission, a "0" pulse on the TE bit ("0" followed by "1") sends a preamble (idle line) after the current word.

– When TE is set there is a 1 bit-time delay before the transmission starts.

Bit 2 = **RE** *Receiver enable*
This bit enables the receiver. It is set and cleared by software.
0: Receiver is disabled in the SCISR register
1: Receiver is enabled and begins searching for a start bit

Bit 1 = **RWU** *Receiver wake-up*
This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.
0: Receiver in active mode
1: Receiver in mute mode

Notes:

– Before selecting Mute mode (by setting the RWU bit) the SCI must first receive a data byte, otherwise it cannot function in Mute mode with wake-up by Idle line detection.

– In Address Mark Detection Wake-Up configuration (WAKE bit = 1) the RWU bit cannot be modified by software while the RDRF bit is set.

Bit 0 = **SBK** *Send break*
This bit set is used to send break characters. It is set and cleared by software.
0: No break character is transmitted
1: Break characters are transmitted

Note: If the SBK bit is set to "1" and then to "0", the transmitter will send a BREAK word at the end of the current word.

DATA REGISTER (SCIDR)

Read/Write
Reset Value: Undefined

Contains the Received or Transmitted data character, depending on whether it is read from or written to.

7							0
DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0

The Data register performs a double function (read and write) since it is composed of two registers, one for transmission (TDR) and one for reception (RDR).
The TDR register provides the parallel interface between the internal bus and the output shift register (see Figure 62).
The RDR register provides the parallel interface between the input shift register and the internal bus (see Figure 62).

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

BAUD RATE REGISTER (SCIBRR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0

Note: When LIN slave mode is disabled, the SCIBRR register controls the conventional baud rate generator.

Bits 7:6 = **SCP[1:0]** *First SCI Prescaler*
These 2 prescaling bits allow several standard clock division ranges:

PR Prescaling factor	SCP1	SCP0
1	0	0
3		1
4	1	0
13		1

Bits 5:3 = **SCT[2:0]** *SCI Transmitter rate divisor*
These 3 bits, in conjunction with the SCP1 and SCP0 bits define the total division applied to the bus clock to yield the transmit rate clock in conventional Baud Rate Generator mode.

TR dividing factor	SCT2	SCT1	SCT0
1	0	0	0
2			1
4		1	0
8			1
16	1	0	0
32			1
64		1	0
128			1

Bits 2:0 = **SCR[2:0]** *SCI Receiver rate divider*
These 3 bits, in conjunction with the SCP[1:0] bits define the total division applied to the bus clock to yield the receive rate clock in conventional Baud Rate Generator mode.

RR dividing factor	SCR2	SCR1	SCR0
1	0	0	0
2			1
4		1	0
8			1
16	1	0	0
32			1
64		1	0
128			1

LINSCI™ SERIAL COMMUNICATION INTERFACE (SCI Mode) (cont'd)

EXTENDED RECEIVE PRESCALER DIVISION REGISTER (SCIERPR)

Read/Write

Reset Value: 0000 0000 (00h)

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

Bits 7:0 = **ERPR[7:0]** *8-bit Extended Receive Prescaler Register*

The extended Baud Rate Generator is activated when a value other than 00h is stored in this register. The clock frequency from the 16 divider (see Figure 64) is divided by the binary factor set in the SCIERPR register (in the range 1 to 255).

The extended baud rate generator is not active after a reset.

EXTENDED TRANSMIT PRESCALER DIVISION REGISTER (SCIETPR)

Read/Write

Reset Value:0000 0000 (00h)

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

Bits 7:0 = **ETPR[7:0]** *8-bit Extended Transmit Prescaler Register*

The extended Baud Rate Generator is activated when a value other than 00h is stored in this register. The clock frequency from the 16 divider (see Figure 64) is divided by the binary factor set in the SCIETPR register (in the range 1 to 255).

The extended baud rate generator is not active after a reset.

Note: In LIN slave mode, the Conventional and Extended Baud Rate Generators are disabled.

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode)

10.5.9 LIN Mode - Functional Description.

The block diagram of the Serial Control Interface, in LIN slave mode is shown in Figure 66.

It uses six registers:

– 3 control registers: SCICR1, SCICR2 and SCICR3

– 2 status registers: the SCISR register and the LHLR register mapped at the SCIERPR address

– A baud rate register: LPR mapped at the SCIBRR address and an associated fraction register LPFR mapped at the SCIETPR address

The bits dedicated to LIN are located in the SCICR3. Refer to the register descriptions in Section 10.5.10for the definitions of each bit.

10.5.9.1 Entering LIN Mode

To use the LINSCI in LIN mode the following configuration must be set in SCICR3 register:

– Clear the M bit to configure 8-bit word length.

– Set the LINE bit.

Master

To enter master mode the LSLV bit must be reset In this case, setting the SBK bit will send 13 low bits.

Then the baud rate can programmed using the SCIBRR, SCIERPR and SCIETPR registers.

In LIN master mode, the Conventional and / or Extended Prescaler define the baud rate (as in standard SCI mode)

Slave

Set the LSLV bit in the SCICR3 register to enter LIN slave mode. In this case, setting the SBK bit will have no effect.

In LIN Slave mode the LIN baud rate generator is selected instead of the Conventional or Extended Prescaler. The LIN baud rate generator is common to the transmitter and the receiver.

Then the baud rate can be programmed using LPR and LPRF registers.

Note: It is mandatory to set the LIN configuration first before programming LPR and LPRF, because the LIN configuration uses a different baud rate generator from the standard one.

10.5.9.2 LIN Transmission

In LIN mode the same procedure as in SCI mode has to be applied for a LIN transmission.

To transmit the LIN Header the proceed as follows:

– First set the SBK bit in the SCICR2 register to start transmitting a 13-bit LIN Synch Break

– reset the SBK bit

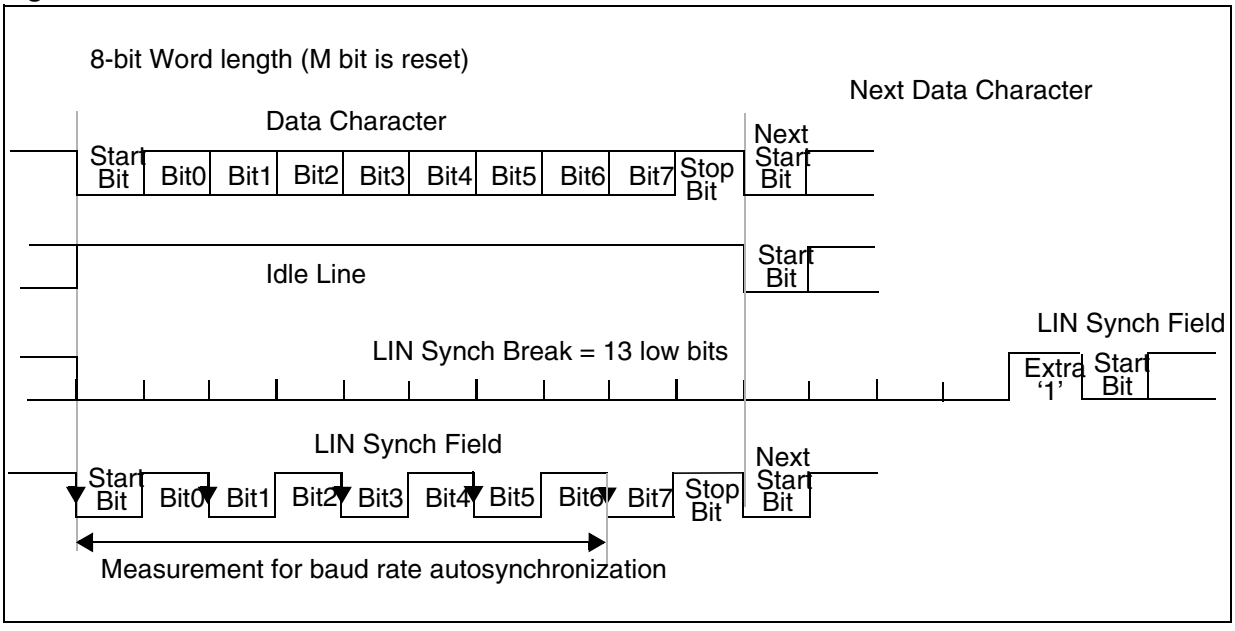
– Load the LIN Synch Field (0x55) in the SCIDR register to request Synch Field transmission

– Wait until the SCIDR is empty (TDRE bit set in the SCISR register)

– Load the LIN message Identifier in the SCIDR register to request Identifier transmission.

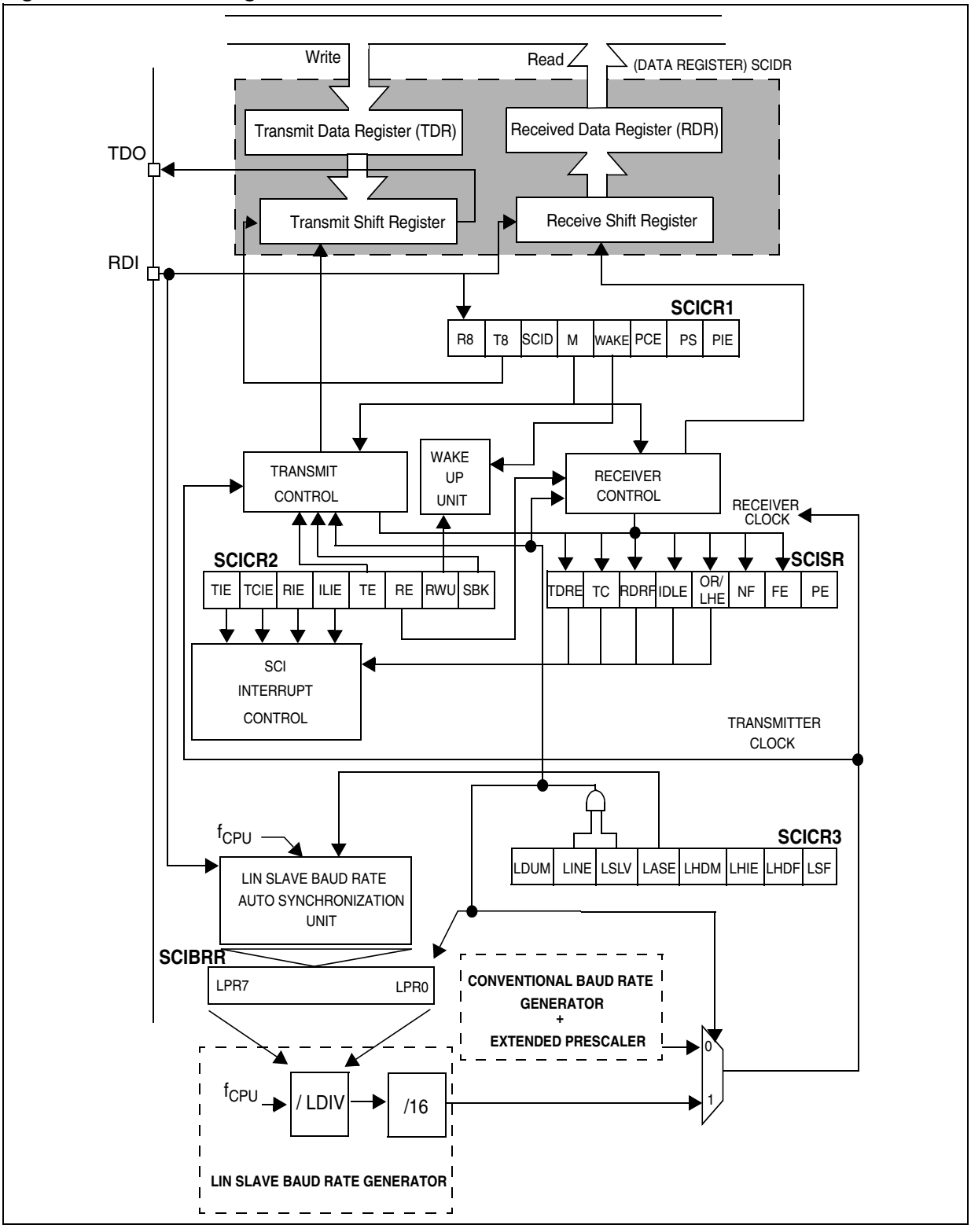
LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

Figure 65. LIN Characters



LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

Figure 66. SCI Block Diagram in LIN Slave Mode



LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

10.5.9.3 LIN Reception

In LIN mode the reception of a byte is the same as in SCI mode but the LINSCI has features for handling the LIN Header automatically (identifier detection) or semiautomatically (Synch Break detection) depending on the LIN Header detection mode. The detection mode is selected by the LHDM bit in the SCICR3.

Additionally, an automatic resynchronization feature can be activated to compensate for any clock deviation, for more details please refer to Section 10.5.9.5 LIN Baud Rate.

LIN Header Handling by a Slave

Depending on the LIN Header detection method the LINSCI will signal the detection of a LIN Header after the LIN Synch Break or after the Identifier has been successfully received.

Note:

It is recommended to combine the Header detection function with Mute mode. Putting the LINSCI in Mute mode allows the detection of Headers only and prevents the reception of any other characters.

This mode can be used to wait for the next Header without being interrupted by the data bytes of the current message in case this message is not relevant for the application.

Synch Break Detection (LHDM = 0):

When a LIN Synch Break is received:

– The RDRF bit in the SCISR register is set. It indicates that the content of the shift register is transferred to the SCIDR register, a value of 0x00 is expected for a Break.

– The LHDF flag in the SCICR3 register indicates that a LIN Synch Break Field has been detected.

– An interrupt is generated if the LHIE bit in the SCICR3 register is set and the I[1:0] bits are cleared in the CCR register.

– Then the LIN Synch Field is received and measured.

  – If automatic resynchronization is enabled (LASE bit = 1), the LIN Synch Field is not transferred to the shift register: There is no need to clear the RDRF bit.

  – If automatic resynchronization is disabled (LASE bit = 0), the LIN Synch Field is received as a normal character and transferred to the SCIDR register and RDRF is set.

Note:

In LIN slave mode, the FE bit detects all frame error which does not correspond to a break.

Identifier Detection (LHDM = 1):

This case is the same as the previous one except that the LHDF and the RDRF flags are set only after the entire header has been received (this is true whether automatic resynchronization is enabled or not). This indicates that the LIN Identifier is available in the SCIDR register.

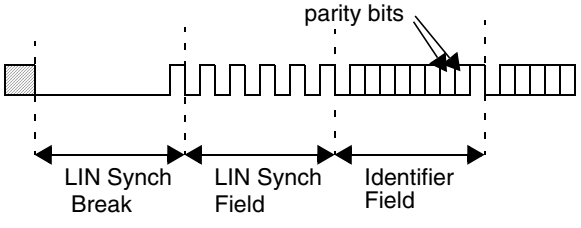
Notes:

During LIN Synch Field measurement, the SCI state machine is switched off: No characters are transferred to the data register.

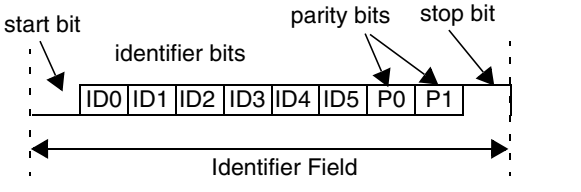
LIN Slave parity

In LIN Slave mode (LINE and LSLV bits are set) LIN parity checking can be enabled by setting the PCE bit.

In this case, the parity bits of the LIN Identifier Field are checked. The identifier character is recognized as the third received character after a break character (included):



The bits involved are the two MSB positions (7th and 8th bits if M = 0; 8th and 9th bits if M = 0) of the identifier character. The check is performed as specified by the LIN specification:



$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$

$$P1 = \overline{ID1 \oplus ID3 \oplus ID4 \oplus ID5}$$

M = 0

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

10.5.9.4 LIN Error Detection

LIN Header Error Flag

The LIN Header Error Flag indicates that an invalid LIN Header has been detected.

When a LIN Header Error occurs:

– The LHE flag is set

– An interrupt is generated if the RIE bit is set and the I[1:0] bits are cleared in the CCR register.

If autosynchronization is enabled (LASE bit = 1), this can mean that the LIN Synch Field is corrupted, and that the SCI is in a blocked state (LSF bit is set). The only way to recover is to reset the LSF bit and then to clear the LHE bit.

– The LHE bit is reset by an access to the SCISR register followed by a read of the SCIDR register.

LHE/OVR Error Conditions

When Auto Resynchronization is disabled (LASE bit = 0), the LHE flag detects:

– That the received LIN Synch Field is not equal to 55h.

– That an overrun occurred (as in standard SCI mode)

– Furthermore, if LHDM is set it also detects that a LIN Header Reception Timeout occurred (only if LHDM is set).

When the LIN auto-resynchronization is enabled (LASE bit = 1), the LHE flag detects:

– That the deviation error on the Synch Field is outside the LIN specification which allows up to +/-15.5% of period deviation between the slave and master oscillators.

– A LIN Header Reception Timeout occurred. If $T_{HEADER} > T_{HEADER_MAX}$ then the LHE flag is set. Refer to Figure 67. (only if LHDM is set to 1)

– An overflow during the Synch Field Measurement, which leads to an overflow of the divider registers. If LHE is set due to this error then the SCI goes into a blocked state (LSF bit is set).

– That an overrun occurred on Fields other than the Synch Field (as in standard SCI mode)

Deviation Error on the Synch Field

The deviation error is checking by comparing the current baud rate (relative to the slave oscillator) with the received LIN Synch Field (relative to the master oscillator). Two checks are performed in parallel:

– The first check is based on a measurement between the first falling edge and the last falling edge of the Synch Field. Let us refer to this period deviation as D:

If the LHE flag is set, it means that:

D > 15.625%

If LHE flag is not set, it means that:

D < 16.40625%

If 15.625% ≤D < 16.40625%, then the flag can be either set or reset depending on the dephasing between the signal on the RDI line and the CPU clock.

– The second check is based on the measurement of each bit time between both edges of the Synch Field: this checks that each of these bit times is large enough compared to the bit time of the current baud rate.

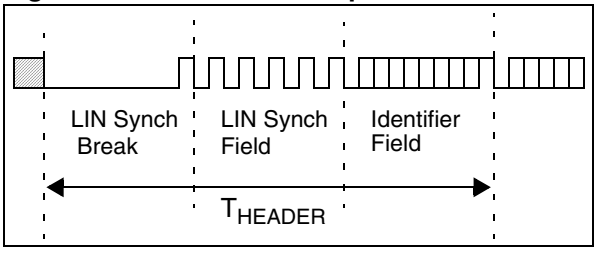
When LHE is set due to this error then the SCI goes into a blocked state (LSF bit is set).

LIN Header Time-out Error

When the LIN Identifier Field Detection Method is used (by configuring LHDM to 1) or when LIN auto-resynchronization is enabled (LASE bit = 1), the LINSCI automatically monitors the T_{HEADER_MAX} condition given by the LIN protocol.

If the entire Header (up to and including the STOP bit of the LIN Identifier Field) is not received within the maximum time limit of 57 bit times then a LIN Header Error is signalled and the LHE bit is set in the SCISR register.

Figure 67. LIN Header Reception Timeout



The time-out counter is enabled at each break detection. It is stopped in the following conditions:
- A LIN Identifier Field has been received
- An LHE error occurred (other than a timeout error).
- A software reset of LSF bit (transition from high to low) occurred during the analysis of the LIN Synch Field or

If LHE bit is set due to this error during the LIN Synchr Field (if LASE bit = 1) then the SCI goes into a blocked state (LSF bit is set).

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

If LHE bit is set due to this error during Fields other than LIN Synch Field or if LASE bit is reset then the current received Header is discarded and the SCI searches for a new Break Field.

Note on LIN Header Time-out Limit

According to the LIN specification, the maximum length of a LIN Header which does not cause a timeout is equal to 1.4 * (34 + 1) = 49 T_{BIT_MASTER}.

T_{BIT_MASTER} refers to the master baud rate.

When checking this timeout, the slave node is desynchronized for the reception of the LIN Break and Synch fields. Consequently, a margin must be allowed, taking into account the worst case: This occurs when the LIN identifier lasts exactly 10 T_{BIT_MASTER} periods. In this case, the LIN Break and Synch fields last 49 - 10 = 39T_{BIT_MASTER} periods.

Assuming the slave measures these first 39 bits with a desynchronized clock of 15.5%. This leads to a maximum allowed Header Length of:

39 x (1/0.845) T_{BIT_MASTER} + 10T_{BIT_MASTER}

= 56.15 T_{BIT_SLAVE}

A margin is provided so that the time-out occurs when the header length is greater than 57 T_{BIT_SLAVE} periods. If it is less than or equal to 57 T_{BIT_SLAVE} periods, then no timeout occurs.

LIN Header Length

Even if no timeout occurs on the LIN Header, it is possible to have access to the effective LIN header Length (T_{HEADER}) through the LHL register. This allows monitoring at software level the T_{FRAME_MAX} condition given by the LIN protocol.

This feature is only available when LHDM bit = 1 or when LASE bit = 1.

Mute Mode and Errors

In mute mode when LHDM bit = 1, if an LHE error occurs during the analysis of the LIN Synch Field or if a LIN Header Time-out occurs then the LHE bit is set but it does not wake up from mute mode. In this case, the current header analysis is discarded. If needed, the software has to reset LSF bit. Then the SCI searches for a new LIN header.

In mute mode, if a framing error occurs on a data (which is not a break), it is discarded and the FE bit is not set.

When LHDM bit = 1, any LIN header which respects the following conditions causes a wake-up from mute mode:

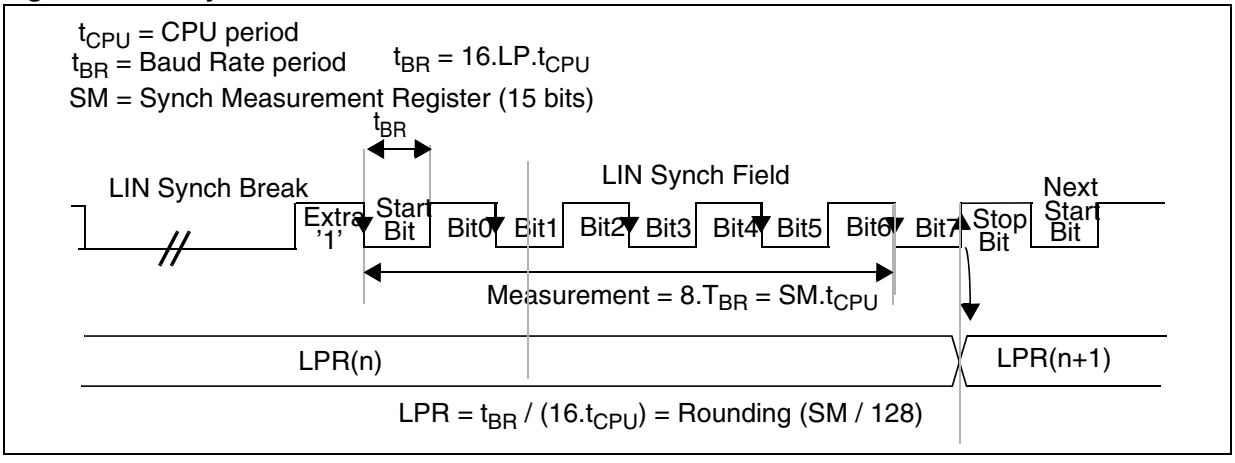
- A valid LIN Break Field (at least 11 dominant bits followed by a recessive bit)

- A valid LIN Synch Field (without deviation error)

- A LIN Identifier Field without framing error. Note that a LIN parity error on the LIN Identifier Field does not prevent wake-up from mute mode.

- No LIN Header Time-out should occur during Header reception.

Figure 68. LIN Synch Field Measurement



LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

10.5.9.5 LIN Baud Rate

Baud rate programming is done by writing a value in the LPR prescaler or performing an automatic resynchronization as described below.

Automatic Resynchronization

To automatically adjust the baud rate based on measurement of the LIN Synch Field:

– Write the nominal LIN Prescaler value (usually depending on the nominal baud rate) in the LPFR / LPR registers.

– Set the LASE bit to enable the Auto Synchronization Unit.

When Auto Synchronization is enabled, after each LIN Synch Break, the time duration between five falling edges on RDI is sampled on f_{CPU} and the result of this measurement is stored in an internal 15-bit register called SM (not user accessible) (See Figure 68). Then the LDIV value (and its associated LPFR and LPR registers) are automatically updated at the end of the fifth falling edge. During LIN Synch field measurement, the SCI state machine is stopped and no data is transferred to the data register.

10.5.9.6 LIN Slave Baud Rate Generation

In LIN mode, transmission and reception are driven by the LIN baud rate generator

Note: LIN Master mode uses the Extended or Conventional prescaler register to generate the baud rate.

If LINE bit = 1 and LSLV bit = 1 then the Conventional and Extended Baud Rate Generators are disabled: the baud rate for the receiver and transmitter are both set to the same value, depending on the LIN Slave baud rate generator:

$$Tx = Rx = \frac{f_{CPU}}{(16*LDIV)}$$

with:

LDIV is an unsigned fixed point number. The mantissa is coded on 8 bits in the LPR register and the fraction is coded on 4 bits in the LPFR register.

If LASE bit = 1 then LDIV is automatically updated at the end of each LIN Synch Field.

Three registers are used internally to manage the auto-update of the LIN divider (LDIV):

- LDIV_NOM (nominal value written by software at LPR/LPFR addresses)

- LDIV_MEAS (results of the Field Synch measurement)

- LDIV (used to generate the local baud rate)

The control and interactions of these registers is explained in Figure 69 and Figure 70. It depends on the LDUM bit setting (LIN Divider Update Method)

Note:

As explained in Figure 69 and Figure 70, LDIV can be updated by two concurrent actions: a transfer from LDIV_MEAS at the end of the LIN Sync Field and a transfer from LDIV_NOM due to a software write of LPR. If both operations occur at the same time, the transfer from LDIV_NOM has priority.

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

Figure 69. LDIV Read / Write Operations When LDUM = 0

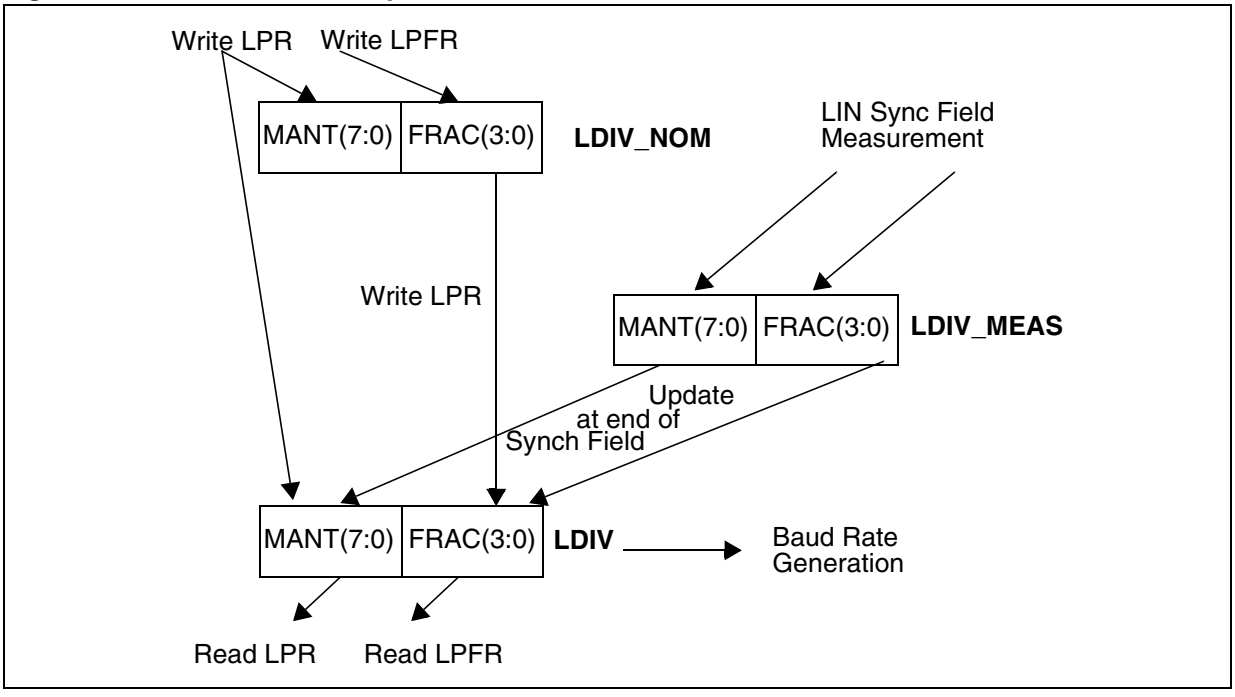
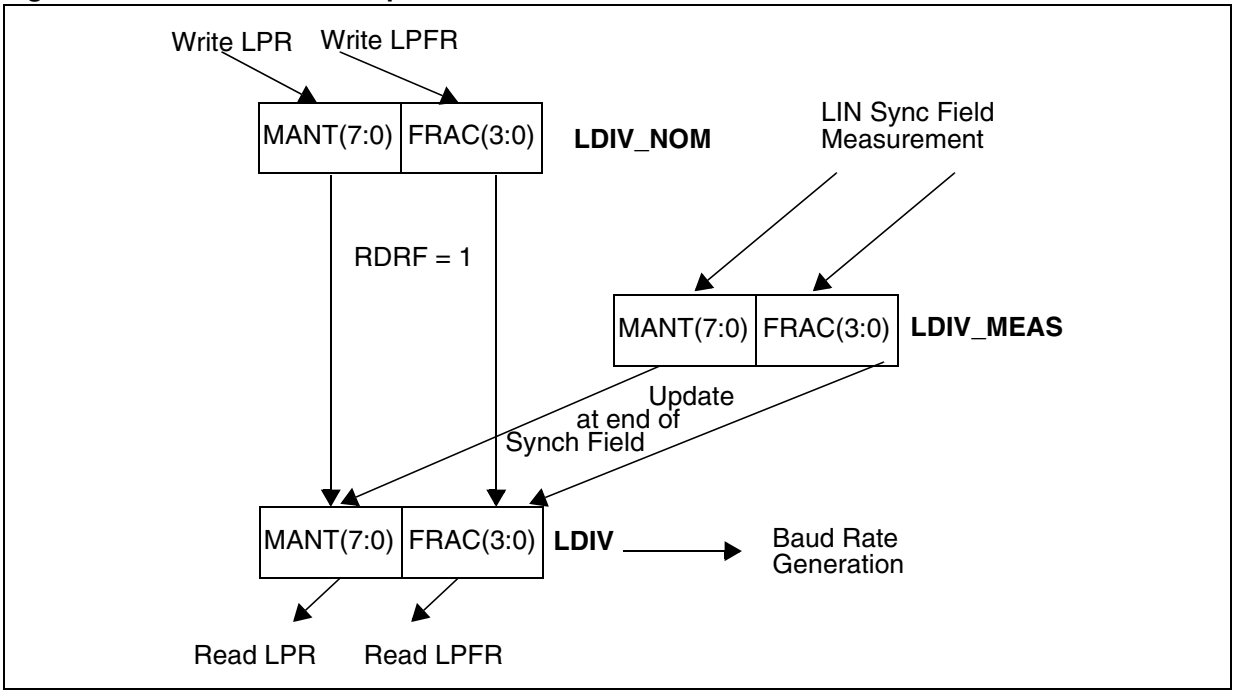


Figure 70. LDIV Read / Write Operations When LDUM = 1



LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

10.5.9.7 LINSCI Clock Tolerance

LINSCI Clock Tolerance when unsynchronized

When LIN slaves are unsynchronized (meaning no characters have been transmitted for a relatively long time), the maximum tolerated deviation of the LINSCI clock is +/-15%.

If the deviation is within this range then the LIN Synch Break is detected properly when a new reception occurs.

This is made possible by the fact that masters send 13 low bits for the LIN Synch Break, which can be interpreted as 11 low bits (13 bits -15% = 11.05) by a "fast" slave and then considered as a LIN Synch Break. According to the LIN specification, a LIN Synch Break is valid when its duration is greater than t_{SBRKTS} = 10. This means that the LIN Synch Break must last at least 11 low bits.

Note: If the period desynchronization of the slave is +15% (slave too slow), the character "00h" which represents a sequence of 9 low bits must not be interpreted as a break character (9 bits + 15% = 10.35). Consequently, a valid LIN Synch break must last at least 11 low bits.

LINSCI Clock Tolerance when Synchronized

When synchronization has been performed, following reception of a LIN Synch Break, the LINSCI, in LIN mode, has the same clock deviation tolerance as in SCI mode, which is explained below:

During reception, each bit is oversampled 16 times. The mean of the 8th, 9th and 10th samples is considered as the bit value.

Consequently, the clock frequency should not vary more than 6/16 (37.5%) within one bit.

The sampling clock is resynchronized at each start bit, so that when receiving 10 bits (one start bit, 1 data byte, 1 stop bit), the clock deviation should not exceed 3.75%.

10.5.9.8 Clock Deviation Causes

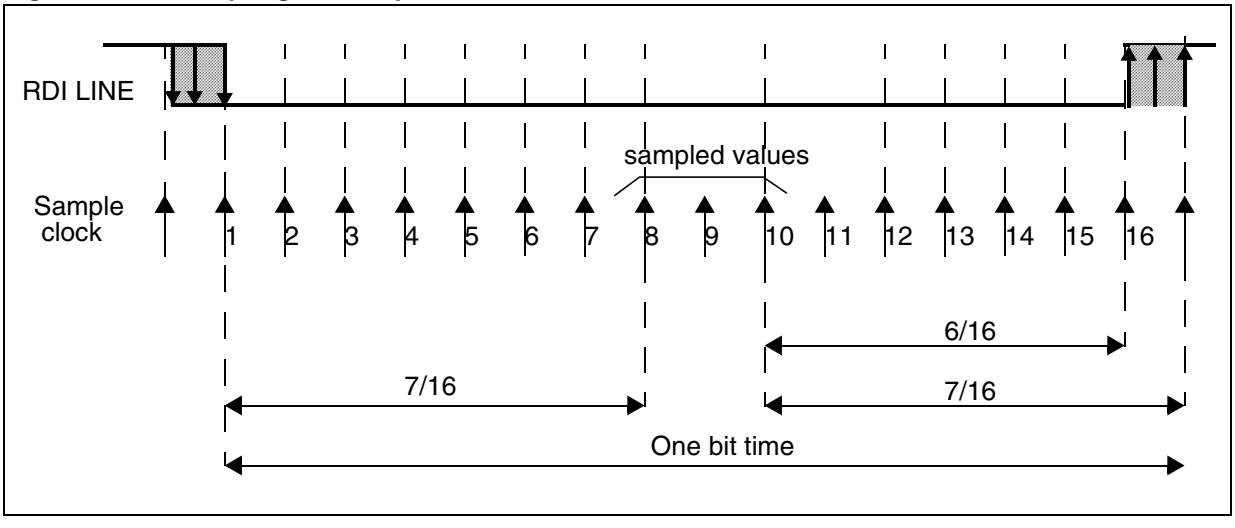
The causes which contribute to the total deviation are:

- D_{TRA}: Deviation due to transmitter error.
  Note: The transmitter can be either a master or a slave (in case of a slave listening to the response of another slave).
- D_{MEAS}: Error due to the LIN Synch measurement performed by the receiver.
- D_{QUANT}: Error due to the baud rate quantization of the receiver.
- D_{REC}: Deviation of the local oscillator of the receiver: This deviation can occur during the reception of one complete LIN message assuming that the deviation has been compensated at the beginning of the message.
- D_{TCL}: Deviation due to the transmission line (generally due to the transceivers)

All the deviations of the system should be added and compared to the LINSCI clock tolerance:

$D_{TRA} + D_{MEAS} + D_{QUANT} + D_{REC} + D_{TCL} < 3.75\%$

Figure 71.Bit Sampling in Reception Mode



LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

10.5.9.9 Error due to LIN Synch measurement

The LIN Synch Field is measured over eight bit times.

This measurement is performed using a counter clocked by the CPU clock. The edge detections are performed using the CPU clock cycle.

This leads to a precision of 2 CPU clock cycles for the measurement which lasts 16*8*LDIV clock cycles.

Consequently, this error (D_{MEAS}) is equal to:

$2 / (128*LDIV_{MIN})$.

$LDIV_{MIN}$ corresponds to the minimum LIN prescaler content, leading to the maximum baud rate, taking into account the maximum deviation of +/-15%.

10.5.9.10 Error due to Baud Rate Quantization

The baud rate can be adjusted in steps of 1 / (16 * LDIV). The worst case occurs when the "real" baud rate is in the middle of the step.

This leads to a quantization error (D_{QUANT}) equal to $1 / (2*16*LDIV_{MIN})$.

10.5.9.11 Impact of Clock Deviation on Maximum Baud Rate

The choice of the nominal baud rate ($LDIV_{NOM}$) will influence both the quantization error (D_{QUANT}) and the measurement error (D_{MEAS}). The worst case occurs for $LDIV_{MIN}$.

Consequently, at a given CPU frequency, the maximum possible nominal baud rate (LPR_{MIN}) should be chosen with respect to the maximum tolerated deviation given by the equation:

$D_{TRA} + 2 / (128*LDIV_{MIN}) + 1 / (2*16*LDIV_{MIN}) + D_{REC} + D_{TCL} < 3.75\%$

Example:

A nominal baud rate of 20Kbits/s at T_{CPU} = 125ns (8 MHz) leads to $LDIV_{NOM}$ = 25d.

$LDIV_{MIN} = 25 - 0.15*25 = 21.25$

$D_{MEAS} = 2 / (128*LDIV_{MIN}) * 100 = 0.00073\%$

$D_{QUANT} = 1 / (2*16*LDIV_{MIN}) * 100 = 0.0015\%$

LIN Slave systems

For LIN Slave systems (the LINE and LSLV bits are set), receivers wake up by LIN Synch Break or LIN Identifier detection (depending on the LHDM bit).

Hot Plugging Feature for LIN Slave Nodes

In LIN Slave Mute Mode (the LINE, LSLV and RWU bits are set) it is possible to hot plug to a network during an ongoing communication flow. In this case the SCI monitors the bus on the RDI line until 11 consecutive dominant bits have been detected and discards all the other bits received.

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

10.5.10 LIN Mode Register Description

STATUS REGISTER (SCISR)
Read Only
Reset Value: 1100 0000 (C0h)

7							0
TDRE	TC	RDRF	IDLE	LHE	NF	FE	PE

Bits 7:4 = Same function as in SCI mode, please refer to Section 10.5.8 SCI Mode Register Description.

Bit 3 = **LHE** *LIN Header Error*.
During LIN Header this bit signals three error types:

– The LIN Synch Field is corrupted and the SCI is blocked in LIN Synch State (LSF bit = 1).

– A timeout occurred during LIN Header reception

– An overrun error was detected on one of the header field (see OR bit description in Section 10.5.8 SCI Mode Register Description)).

An interrupt is generated if RIE = 1 in the SCICR2 register. If blocked in the LIN Synch State, the LSF bit must first be reset (to exit LIN Synch Field state and then to be able to clear LHE flag). Then it is cleared by the following software sequence: An access to the SCISR register followed by a read to the SCIDR register.
0: No LIN Header error
1: LIN Header error detected

Note:

Apart from the LIN Header this bit signals an Overrun Error as in SCI mode, (see description in Section 10.5.8 SCI Mode Register Description)

Bit 2 = **NF** *Noise flag*

In LIN Master mode (LINE bit = 1 and LSLV bit = 0) this bit has the same function as in SCI mode, please refer to Section 10.5.8 SCI Mode Register Description

In LIN Slave mode (LINE bit = 1 and LSLV bit = 1) this bit has no meaning.

Bit 1 = **FE** *Framing error.*
In LIN slave mode, this bit is set only when a real framing error is detected (if the stop bit is dominant (0) and at least one of the other bits is recessive (1). It is not set when a break occurs, the LHDF bit is used instead as a break flag (if the LHDM bit = 0). It is cleared by a software sequence (an access to the SCISR register followed by a read to the SCIDR register).
0: No Framing error
1: Framing error detected

Bit 0 = **PE** *Parity error.*
This bit is set by hardware when a LIN parity error occurs (if the PCE bit is set) in receiver mode. It is cleared by a software sequence (a read to the status register followed by an access to the SCIDR data register). An interrupt is generated if PIE = 1 in the SCICR1 register.
0: No LIN parity error
1: LIN Parity error detected

CONTROL REGISTER 1 (SCICR1)
Read/Write
Reset Value: x000 0000 (x0h)

7							0
R8	T8	SCID	M	WAKE	PCE	PS	PIE

Bits 7:3 = Same function as in SCI mode, please refer to Section 10.5.8 SCI Mode Register Description.

Bit 2 = **PCE** *Parity control enable.*
This bit is set and cleared by software. It selects the hardware parity control for LIN identifier parity check.
0: Parity control disabled
1: Parity control enabled

When a parity error occurs, the PE bit in the SCISR register is set.

Bit 1 = Reserved

Bit 0 = Same function as in SCI mode, please refer to Section 10.5.8 SCI Mode Register Description.

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

CONTROL REGISTER 2 (SCICR2)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK

Bits 7:2 Same function as in SCI mode, please refer to Section 10.5.8 SCI Mode Register Description.

Bit 1 = **RWU** *Receiver wake-up.*
This bit determines if the SCI is in mute mode or not. It is set and cleared by software and can be cleared by hardware when a wake-up sequence is recognized.
0: Receiver in active mode
1: Receiver in mute mode

Notes:

– Mute mode is recommended for detecting only the Header and avoiding the reception of any other characters. For more details please refer to Section 10.5.9.3 LIN Reception.

– In LIN slave mode, when RDRF is set, the software can not set or clear the RWU bit.

Bit 0 = **SBK** *Send break.*
This bit set is used to send break characters. It is set and cleared by software.
0: No break character is transmitted
1: Break characters are transmitted

Note: If the SBK bit is set to "1" and then to "0", the transmitter will send a BREAK word at the end of the current word.

CONTROL REGISTER 3 (SCICR3)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
LDUM	LINE	LSLV	LASE	LHDM	LHIE	LHDF	LSF

Bit 7 = **LDUM** *LIN Divider Update Method.*
This bit is set and cleared by software and is also cleared by hardware (when RDRF = 1). It is only used in LIN Slave mode. It determines how the LIN Divider can be updated by software.
0: LDIV is updated as soon as LPR is written (if no Auto Synchronization update occurs at the same time).
1: LDIV is updated at the next received character (when RDRF = 1) after a write to the LPR register

Notes:

- If no write to LPR is performed between the setting of LDUM bit and the reception of the next character, LDIV will be updated with the old value.

- After LDUM has been set, it is possible to reset the LDUM bit by software. In this case, LDIV can be modified by writing into LPR / LPFR registers.

Bits 6:5 = **LINE, LSLV** *LIN Mode Enable Bits.*
These bits configure the LIN mode:

LINE	LSLV	Meaning
0	x	LIN mode disabled
1	0	LIN Master Mode
	1	LIN Slave Mode

The LIN Master configuration enables:

The capability to send LIN Synch Breaks (13 low bits) using the SBK bit in the SCICR2 register.

The LIN Slave configuration enables:

– The LIN Slave Baud Rate generator. The LIN Divider (LDIV) is then represented by the LPR and LPFR registers. The LPR and LPFR registers are read/write accessible at the address of the SCIBRR register and the address of the SCIETPR register

– Management of LIN Headers.

– LIN Synch Break detection (11-bit dominant).

– LIN Wake-Up method (see LHDM bit) instead of the normal SCI Wake-Up method.

– Inhibition of Break transmission capability (SBK has no effect)

– LIN Parity Checking (in conjunction with the PCE bit)

Bit 4 = **LASE** *LIN Auto Synch Enable.*
This bit enables the Auto Synch Unit (ASU). It is set and cleared by software. It is only usable in LIN Slave mode.

0: Auto Synch Unit disabled
1: Auto Synch Unit enabled.

Bit 3 = **LHDM** *LIN Header Detection Method*
This bit is set and cleared by software. It is only usable in LIN Slave mode. It enables the Header Detection Method. In addition if the RWU bit in the

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

SCICR2 register is set, the LHDM bit selects the Wake-Up method (replacing the WAKE bit).
0: LIN Synch Break Detection Method
1: LIN Identifier Field Detection Method

Bit 2 = **LHIE** *LIN Header Interrupt Enable*
This bit is set and cleared by software. It is only usable in LIN Slave mode.
0: LIN Header Interrupt is inhibited.
1: An SCI interrupt is generated whenever LHDF = 1.

Bit 1 = **LHDF** *LIN Header Detection Flag*
This bit is set by hardware when a LIN Header is detected and cleared by a software sequence (an access to the SCISR register followed by a read of the SCICR3 register). It is only usable in LIN Slave mode.
0: No LIN Header detected.
1: LIN Header detected.

Notes: The header detection method depends on the LHDM bit:

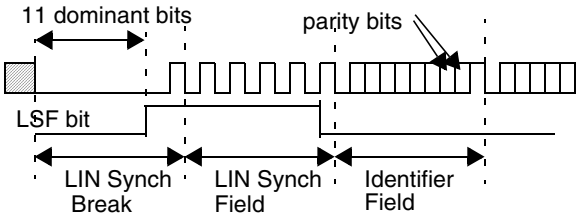
- If LHDM = 0, a header is detected as a LIN Synch Break.
- If LHDM = 1, a header is detected as a LIN Identifier, meaning that a LIN Synch Break Field + a LIN Synch Field + a LIN Identifier Field have been consecutively received.

Bit 0 = **LSF** *LIN Synch Field State*

This bit indicates that the LIN Synch Field is being analyzed. It is only used in LIN Slave mode. In Auto Synchronization Mode (LASE bit = 1), when the SCI is in the LIN Synch Field State it waits or counts the falling edges on the RDI line.

It is set by hardware as soon as a LIN Synch Break is detected and cleared by hardware when the LIN Synch Field analysis is finished (See Figure 72). This bit can also be cleared by software to exit LIN Synch State and return to idle mode.
0: The current character is not the LIN Synch Field
1: LIN Synch Field State (LIN Synch Field undergoing analysis)

Figure 72. LSF Bit Set and Clear



LIN DIVIDER REGISTERS

LDIV is coded using the two registers LPR and LPFR. In LIN Slave mode, the LPR register is accessible at the address of the SCIBRR register and the LPFR register is accessible at the address of the SCIETPR register.

LIN PRESCALER REGISTER (LPR)

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

7							0
LPR7	LPR6	LPR5	LPR4	LPR3	LPR2	LPR1	LPR0

LPR[7:0] *LIN Prescaler (mantissa of LDIV)*

These 8 bits define the value of the mantissa of the LIN Divider (LDIV):

LPR[7:0]	Rounded Mantissa (LDIV)
00h	SCI clock disabled
01h	1
...	...
FEh	254
FFh	255

Caution: LPR and LPFR registers have different meanings when reading or writing to them. Consequently bit manipulation instructions (BRES or BSET) should never be used to modify the LPR[7:0] bits, or the LPFR[3:0] bits.

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

LIN PRESCALER FRACTION REGISTER (LPFR)

Read/Write

Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	LPFR 3	LPFR 2	LPFR 1	LPFR 0

Bits 7:4 = Reserved.

Bits 3:0 = **LPFR[3:0]** *Fraction of LDIV*

These 4 bits define the fraction of the LIN Divider (LDIV):

LPFR[3:0]	Fraction (LDIV)
0h	0
1h	1/16
...	...
Eh	14/16
Fh	15/16

1. When initializing LDIV, the LPFR register must be written first. Then, the write to the LPR register will effectively update LDIV and so the clock generation.

2. In LIN Slave mode, if the LPR[7:0] register is equal to 00h, the transceiver and receiver input clocks are switched off.

Examples of LDIV coding:

Example 1: LPR = 27d and LPFR = 12d

This leads to:

Mantissa (LDIV) = 27d

Fraction (LDIV) = 12/16 = 0.75d

Therefore LDIV = 27.75d

Example 2: LDIV = 25.62d

This leads to:

LPFR = rounded(16*0.62d)

= rounded(9.92d) = 10d = Ah

LPR = mantissa (25.620d) = 25d = 1Bh

Example 3: LDIV = 25.99d

This leads to:

LPFR = rounded(16*0.99d)

= rounded(15.84d) = 16d

LINSCI™ SERIAL COMMUNICATION INTERFACE (LIN Mode) (cont'd)

LIN HEADER LENGTH REGISTER (LHLR)
Read Only
Reset Value: 0000 0000 (00h).

7							0
LHL7	LHL6	LHL5	LHL4	LHL3	LHL2	LHL1	LHL0

Note: In LIN Slave mode when LASE = 1 or LHDM = 1, the LHLR register is accessible at the address of the SCIERPR register.

Otherwise this register is always read as 00h.

Bits 7:0 = **LHL[7:0]** *LIN Header Length.*

This is a read-only register, which is updated by hardware if one of the following conditions occurs:
- After each break detection, it is loaded with "FFh".
- If a timeout occurs on T_{HEADER}, it is loaded with 00h.
- After every successful LIN Header reception (at the same time than the setting of LHDF bit), it is loaded with a value (LHL) which gives access to the number of bit times of the LIN header length (T_{HEADER}). The coding of this value is explained below:

LHL Coding:

T_{HEADER_MAX} = 57
LHL(7:2) represents the mantissa of (57 - T_{HEADER})
LHL(1:0) represents the fraction (57 - T_{HEADER})

LHL[7:2]	Mantissa (57 - T_{HEADER})	Mantissa (T_{HEADER})
0h	0	57
1h	1	56
...	...	...
39h	56	1
3Ah	57	0
3Bh	58	Never Occurs
...	...	...
3Eh	62	Never Occurs
3Fh	63	Initial value

LHL[1:0]	Fraction (57 - T_{HEADER})
0h	0
1h	1/4
2h	1/2
3h	3/4

Example of LHL coding:

Example 1: LHL = 33h = 001100 11b
LHL(7:3) = 1100b = 12d
LHL(1:0) = 11b = 3d
This leads to:
Mantissa (57 - T_{HEADER}) = 12d
Fraction (57 - T_{HEADER}) = 3/4 = 0.75
Therefore:
(57 - T_{HEADER}) = 12.75d
and T_{HEADER} = 44.25d

Example 2:
57 - T_{HEADER} = 36.21d
LHL(1:0) = rounded(4*0.21d) = 1d
LHL(7:2) = Mantissa (36.21d) = 36d = 24h
Therefore LHL(7:0) = 10010001 = 91h

Example 3:
57 - T_{HEADER} = 36.90d
LHL(1:0) = rounded(4*0.90d) = 4d
The carry must be propagated to the matissa:
LHL(7:2) = Mantissa (36.90d) + 1 = 37d =
Therefore LHL(7:0) = 10110000 = A0h

SERIAL COMMUNICATION INTERFACE (Cont'd)

Table 21. SCI Register Map and Reset Values

Addr. (Hex.)	Register Name	7	6	5	4	3	2	1	0
0018h	**SCI1SR**	TDRE	TC	RDRF	IDLE	OR/LHE	NF	FE	PE
	Reset Value	1	1	0	0	0	0	0	0
0019h	**SCI1DR**	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
	Reset Value	-	-	-	-	-	-	-	-
001Ah	**SCI1BRR**	SCP1	SCP0	SCT2	SCT1	SCT0	SCR2	SCR1	SCR0
	LPR (LIN Slave Mode)	LPR7	LPR6	LPR5	LPR4	LPR3	LPR2	LPR1	LPR0
	Reset Value	0	0	0	0	0	0	0	0
001Bh	**SCI1CR1**	R8	T8	SCID	M	WAKE	PCE	PS	PIE
	Reset Value	x	0	0	0	0	0	0	0
001Ch	**SCI1CR2**	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	Reset Value	0	0	0	0	0	0	0	0
001Dh	**SCI1CR3**	LDUM	LINE	LSLV	LASE	LHDM	LHIE	LHDF	LSF
	Reset Value	0	0	0	0	0	0	0	0
001Eh	**SCI1ERPR**	ERPR7	ERPR6	ERPR5	ERPR4	ERPR3	ERPR2	ERPR1	ERPR0
	LHLR (LIN Slave Mode)	LHL7	LHL6	LHL5	LHL4	LHL3	LHL2	LHL1	LHL0
	Reset Value	0	0	0	0	0	0	0	0
001Fh	**SCI1TPR**	ETPR7	ETPR6	ETPR5	ETPR4	ETPR3	ETPR2	ETPR1	ETPR0
	LPRF (LIN Slave Mode)	0	0	0	0	LPRF3	LPRF2	LPRF1	LPRF0
	Reset Value	0	0	0	0	0	0	0	0

10.6 MOTOR CONTROLLER (MTC)

10.6.1 Introduction

The ST7 Motor Controller (MTC) can be seen as a Three-Phase Pulse Width Modulator multiplexed on six output channels and a Back Electromotive Force (BEMF) zero-crossing detector for sensorless control of Permanent Magnet Direct Current (PM BLDC) brushless motors.

The MTC is particularly suited to driving brushless motors (either induction or permanent magnet types) and supports operating modes like:

– Commutation step control with motor voltage regulation and current limitation

– Commutation step control with motor current regulation, i.e. direct torque control

– Position Sensor or sensorless motor phase commutation control (six-step mode)

– BEMF zero-crossing detection with high sensitivity. The integrated phase voltage comparator is directly referred to the full BEMF voltage without any attenuation. A BEMF voltage down to 200 mV can be detected, providing high noise immunity and self-commutated operation in a large speed range.

– Realtime motor winding demagnetization detection for fine-tuning the phase voltage masking time to be applied before BEMF monitoring.

– Automatic and programmable delay between BEMF zero-crossing detection and motor phase commutation.

– PWM generation for three-phase sinewave or three-channel independent PWM signals.

Table 22. MTC Functional Blocks

Section	Page
Input Detection Block	146
Input Pins	146
Sensorless Mode	149
D Event detection	150
Z Event Detection	151
Demagnetization (D) Event	153
Z Event Generation (BEMF Zero Crossing)	155
Protection for ZH event detection	157
Position Sensor Mode	158
Sampling block	159
Commutation Noise Filter	162
Speed Sensor Mode	164
Tachogenerator Mode	164
Encoder Mode	165
Summary	166
Delay Manager	168
Switched Mode	169
Autoswitched Mode	171
Debug Option	172
Checks and Controls for simulated events	175
Speed Measurement Mode	180
Summary	185
PWM Manager	185
Voltage Mode	185
Over Current Handling in Voltage mode	186
Current Mode	186
Current Feedback Comparator	186
Current feedback amplifier	188
Measurement Window	188
Channel Manager	190
MPHST Phase State Register	191
Emergency Feature	191
Dead Time Generator	194
Programmable Chopper	199
PWM Generator Block	200
Main Features	200
Functional Description	201
Prescaler	201
PWM Operating mode	201
Repetition Down-Counter	205
PWM interrupt generation	205
Timer Re-synchronisation	206
PWM generator initialization and start-up	206

MOTOR CONTROLLER (Cont'd)

Table 23. MTC Registers

Register	Description	Register page (RPGS bit)	Page
MTIM	Timer Counter Register	0	207
MTIML	Timer LSB (mode dependent)	0	207
MZPRV	Capture Z_{n-1} Register	0	207
MZREG	Capture Z_n Register	0	207
MCOMP	Compare C_{n+1} Register	0	207
MDREG	Demagnetization Reg.	0	207
MWGHT	A_n Weight Register	0	208
MPRSR	Prescaler & Sampling Reg.	0	208
MIMR	Interrupt Mask Register	0	208
MISR	Interrupt Status Register	0	209
MCRA	Control Register A	0	210
MCRB	Control Register B	0	212
MCRC	Control Register C	0	213
MPHST	Phase State Register	0	214
MDFR	D Event Filter Register	0	216
MCFR	Current Feedback Filter Register	0	215
MREF	Reference register	0	217
MPCR	PWM Control Register	0	218
MREP	Repetition Counter Reg.	0	219
MCPWH	Compare W Register High	0	219
MCPWL	Compare W Register Low	0	219
MCPVH	Compare V Register High	0	219
MCPVL	Compare V Register Low	0	219
MCPUH	Compare U Register High	0	220
MCPUL	Compare U Register Low	0	220
MCP0H	Compare 0 Register High	0	220
MCP0L	Compare 0 Register Low	0	220
MDTG	Dead Time Generator reg.	1	221
MPOL	Polarity Register	1	222
MPWME	PWM register	1	223
MCONF	Configuration register	1	224
MPAR	Parity register	1	225
MZFR	Z Event Filter Register	1	226
MSCR	Sampling Clock Register	1	227

10.6.2 Main Features

- Two on-chip analog comparators, one for BEMF zero-crossing detection, the other for current regulation or limitation
- Seven selectable reference voltages for the hysteresis comparator (0.2 V, 0.6 V, 1 V, 1.5 V, 2 V, 2.5 V, 3.5 V) and the possibility to select an external reference pin (MCVREF).
- 8-bit timer (MTIM) with three compare registers and two capture features, which may be used as the Delay manager of a speed measurement unit
- Measurement window generator for BEMF zero-crossing detection
- Filter option for the zero-crossing detection.
- Auto-calibrated prescaler with 16 division steps
- 8x8-bit multiplier
- Phase input multiplexer
- Sophisticated output management:
  - The six output channels can be split into two groups (high & low)
  - The PWM signal can be multiplexed on high, low or both groups, alternatively or simultaneously, for six-step motor drives
  - 12-bit PWM generator with full modulation capability (0 and 100% duty cycle), edge or center-aligned patterns
  - Dedicated interrupt for PWM duty cycles updating and associated PWM repetition counter.
  - Programmable deadtime insertion unit.
  - Programmable High frequency Chopper insertion and high current PWM outputs for direct optocoupler drives.
  - The output polarity is programmable channel by channel.
  - A programmable bit (active low) forces the outputs in HiZ, Low or High state, depending on option byte 1 (refer to "ST7FMC Device Configuration And Ordering Information" section).
  - An "emergency stop" input pin (active low) asynchronously forces the outputs in HiZ, Low or High state, depending on option byte 1 (refer to "ST7FMC Device Configuration And Ordering Information" section).

MOTOR CONTROLLER (Cont'd)

10.6.3 Application Example: PM BLDC motor drive

This example shows a six-step command sequence for a 3-phase permanent magnet DC brushless motor (PM BLDC motor). Figure 74 shows the phase steps and voltage, while Table 24 shows the relevant phase configurations.

To run this kind of motor efficiently, an autoswitching mode has to be used, i.e. the position of the rotor must self-generate the powered winding commutation. The BEMF zero crossing (Z event) on the non-excited winding is used by the MTC as a rotor position sensor. The delay between this event and the commutation is computed by the MTC and the hardware commutation event C_n is automatically generated after this delay.

After the commutation occurs, the MTC waits until the winding is completely demagnetized by the free-wheeling diode: during this phase the winding is tied to 0V or to the HV high voltage rail and no BEMF can be read. At the end of this phase a new BEMF zero-crossing detection is enabled.

The end of demagnetization event (D), is also detected by the MTC or simulated with a timer compare feature when no detection is possible.

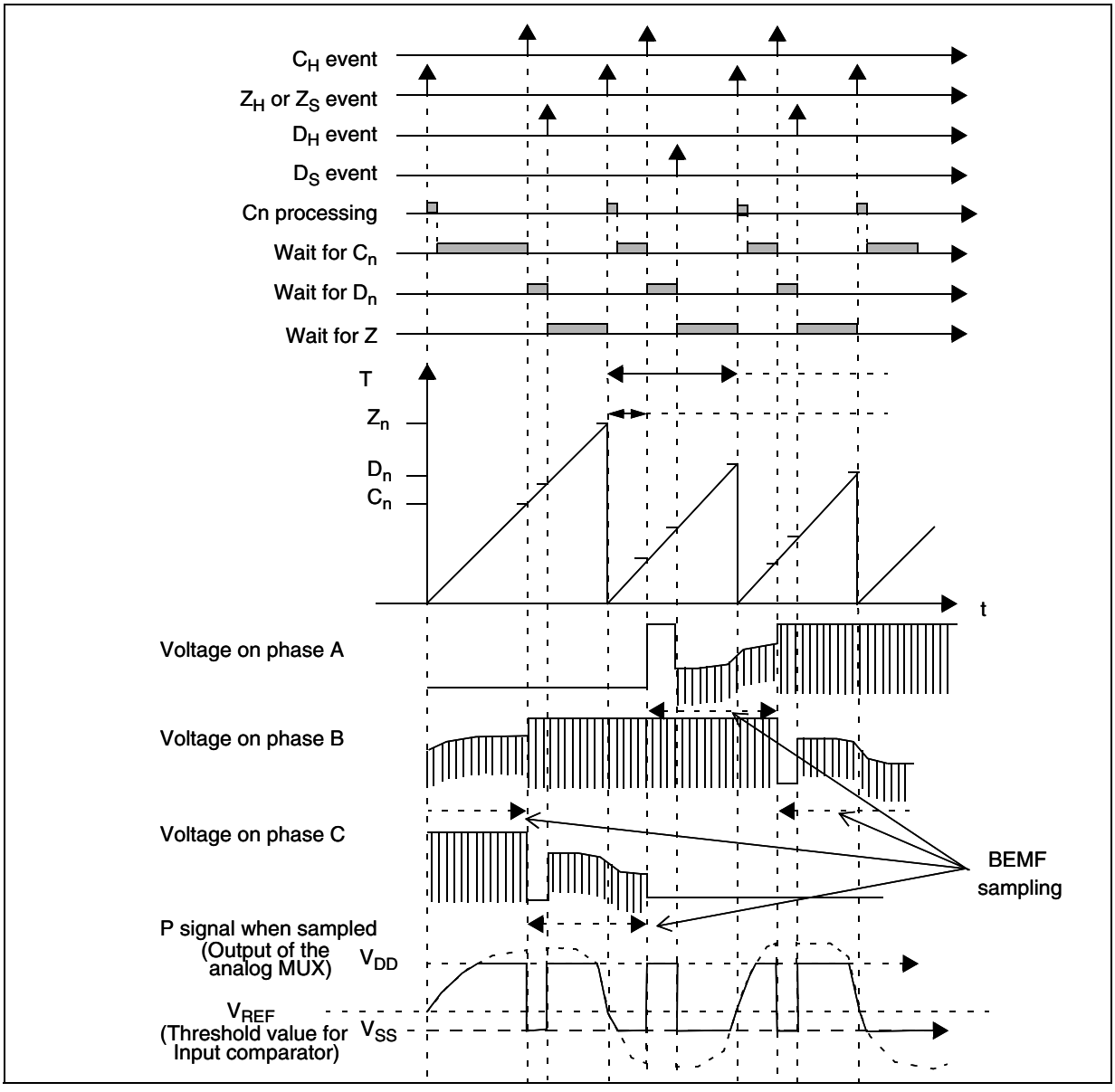
The MTC manages these three events always in the same order: Z generates C after a delay computed in realtime, then waits for D in order to enable the peripheral to detect another Z event.

The BEMF zero-crossing event (Z), can also be detected by the MTC or simulated with a timer compare feature when no detection is possible.

The speed regulation is managed by the microcontroller, by means of an adjustable reference current level in case of current control, or by direct PWM duty-cycle adjustment in case of voltage control.

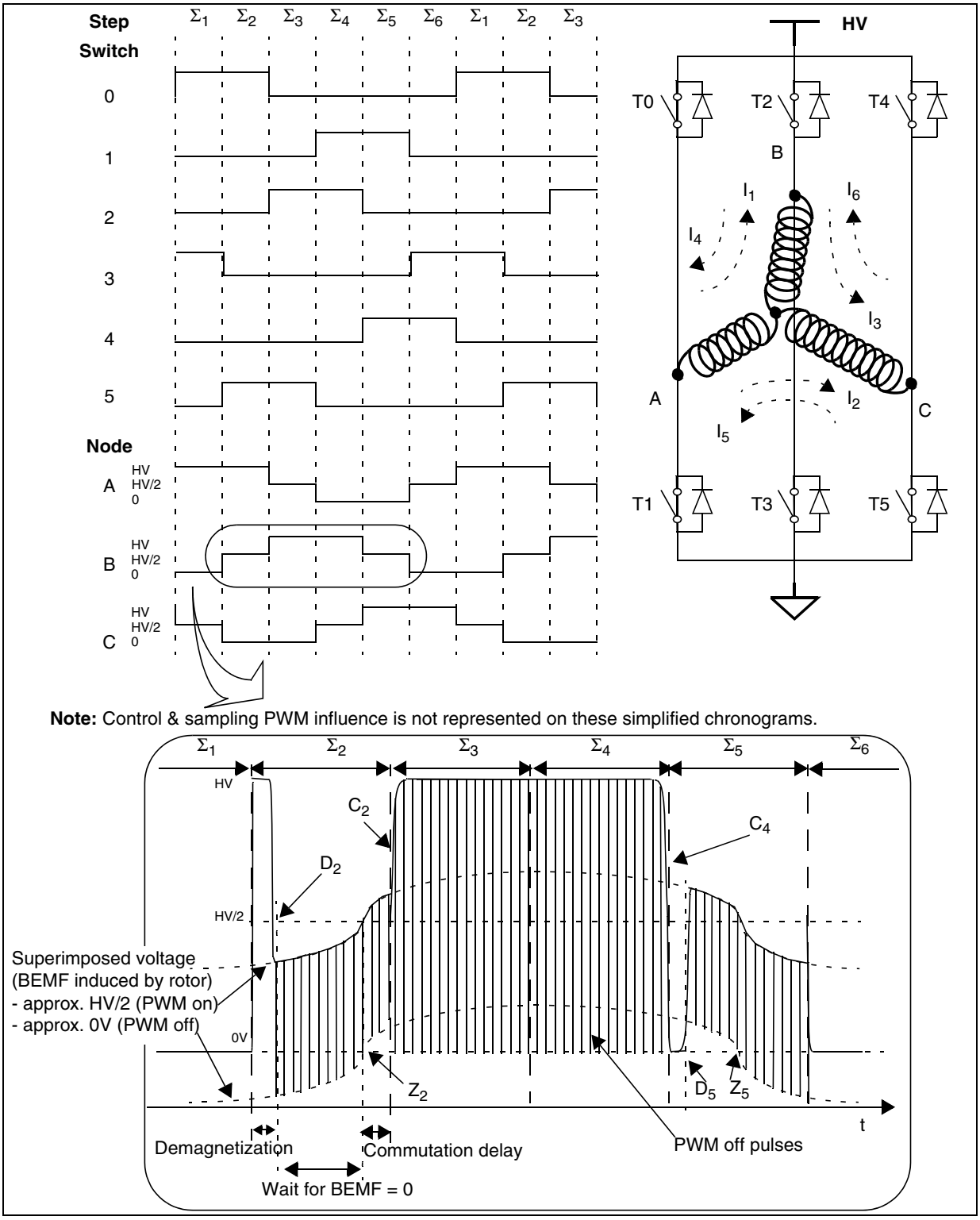
MOTOR CONTROLLER (Cont'd)

Figure 73. Chronogram of Events (in Autoswitched Mode)



MOTOR CONTROLLER (Cont'd)

Figure 74. Example of Command Sequence for 6-step Mode (typical 3-phase PM BLDC Motor Control)



MOTOR CONTROLLER (Cont'd)

All detections of Z_n events are done during a short measurement window while the high side switch is turned off. For this reason the PWM signal is applied on the high side switches.

When the high side switch is off, the high side winding is tied to 0V by the free-wheeling diode, the low side winding voltage is also held at 0V by the low side ON switch and the complete BEMF voltage is present on the third winding: detection is then possible.

Table 24. Step Configuration Summary

Configuration		Step					
		Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6
Phase state register	Current direction	A to B	A to C	B to C	B to A	C to A	C to B
	High side	T0	T0	T2	T2	T4	T4
	Low side	T3	T5	T5	T1	T1	T3
	OO[5:0] bits in MPHST register	001001	100001	100100	000110	010010	011000
BEMF input	Measurement done on:	MCIC	MCIB	MCIA	MCIC	MCIB	MCIA
	IS[1:0] bits in MPHST register	10	01	00	10	01	00
BEMF edge	Back EMF shape	Falling	Rising	Falling	Rising	Falling	Rising
	CPB bit in MCRB register (ZVD bit = 0)	0	1	0	1	0	1
Hardware or Hardware-simulated demagnetization	Voltage on measured point at the start of demagnetization	0V	HV	0V	HV	0V	HV
	HDM-SDM bits in MCRB register	10	11	10	11	10	11
Demagnetization switch	PWM side selection to accelerate demagnetization	Low Side	High Side	Low Side	High Side	Low Side	High Side
	Driver selection to accelerate demagnetization	T3	T0	T5	T2	T1	T4

For a detailed description of the MTC registers, see Section 10.6.13.

10.6.4 Application Example: AC Induction Motor Drive

Although the command sequence is rather different between a PM BLDC and an AC three-phase induction motor, the Motor Controller can be configured to generate three-phase sinusoidal voltages.

A timer with three independent PWM channels is available for this purpose. Based on each of the PWM reference signal, two complemented PWM signals with deadtime are generated on the output pins (6 in total), to drive directly an inverter with triple half bridge topology.

The variable voltage levels to be applied on the motor terminals come from continuously varying duty cycle, from one PWM period to the other (refer to Figure 75 on page 144). The PWM counter generates a dedicated Update event (U event) which:

– updates automatically the compare registers setting the duty cycle to avoid time critical issues and ensure glitchless PWM operation.

– generates a dedicated U interrupt in which the values for the next coming update event are loaded in compare preload registers.

The shape of the output voltage (voltage, frequency, sinewave, trapezoid, ...) is completely managed by the applicative software, in charge of computing the compare values to be loaded for a given PWM duty-cycle (refer to Figure 76).

MOTOR CONTROLLER (Cont'd)

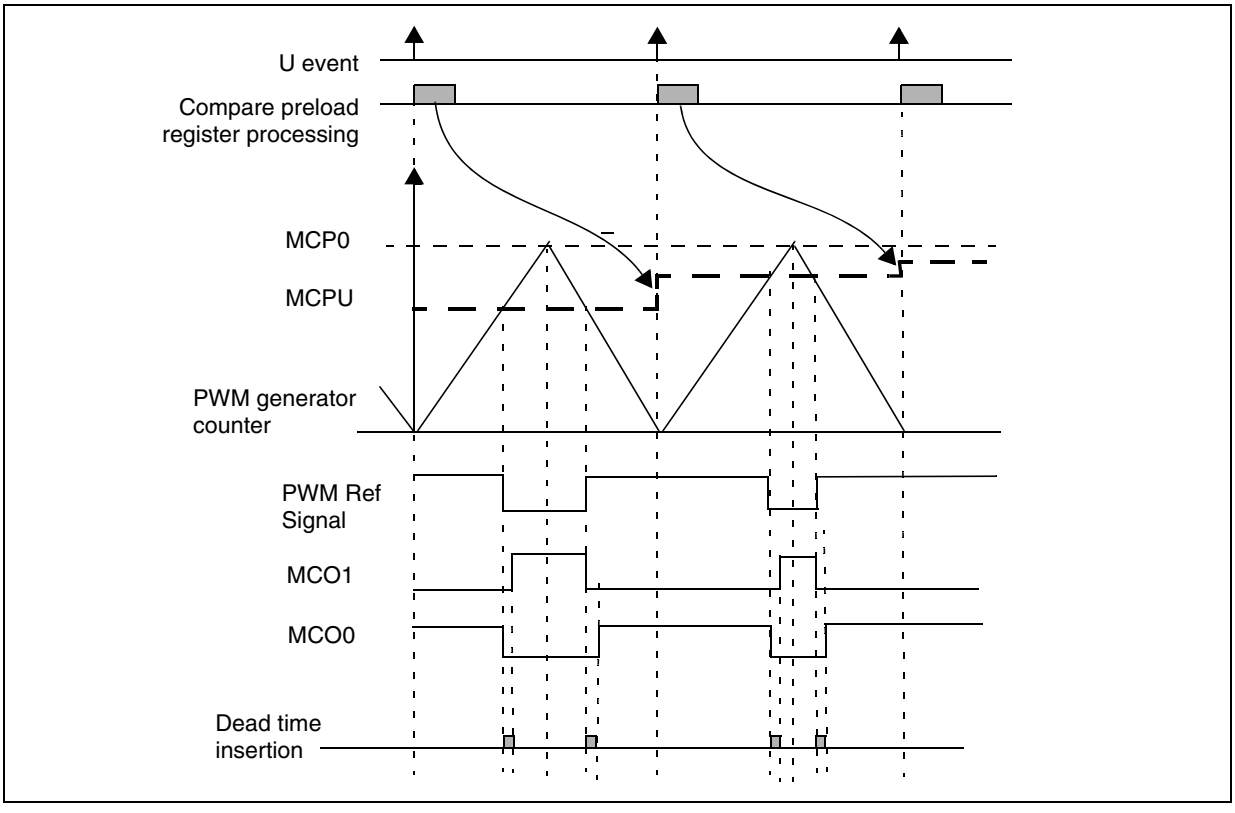
Finally, the PWM modulated voltage generated by the power stage is smoothed by the motor inductance to get sinusoidal currents in the stator windings.

The induction motor being asynchronous, there is no need to synchronize the rotor position to the sinewave generation phase in most of the applications.

Part of the MTC dedicated to delay computation and event sampling can thus be reconfigured to perform speed acquisition of the most common speed sensor, without the need of an additional standard timer.

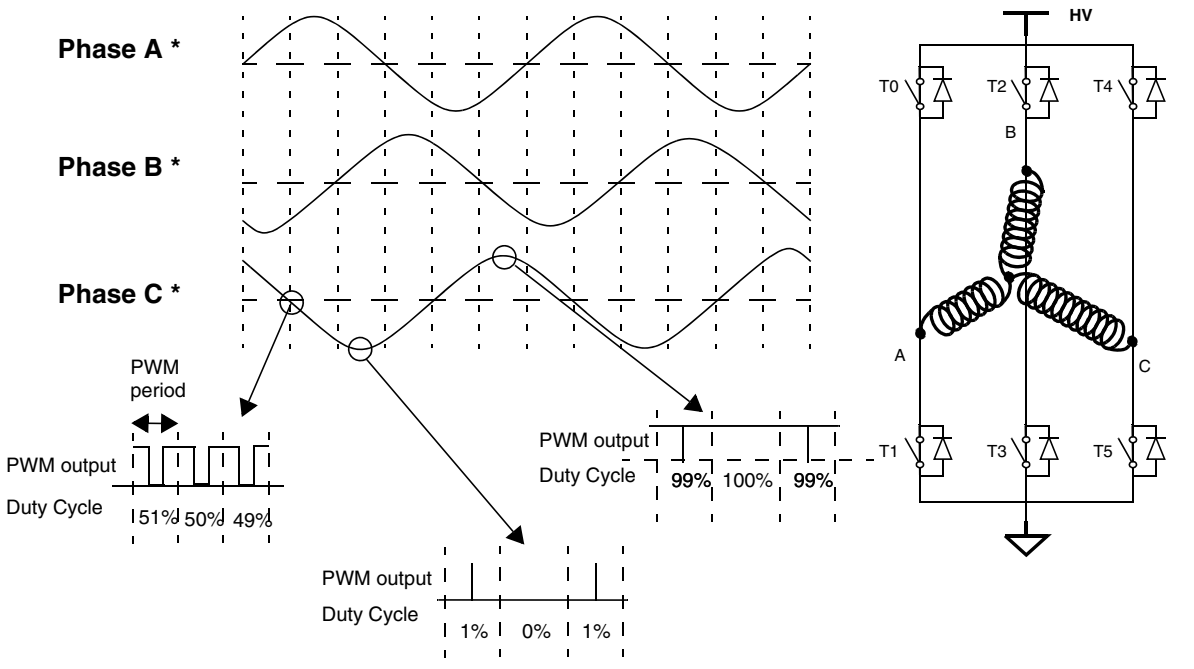
This speed measurement timer with clear-on-capture and clock prescaler auto-setting allows to keep the CPU load to a minimum level while taking benefit of the embedded input comparator and edge detector.

Figure 75. Complementary PWM generation for three-phase induction motor (1 phase represented)



MOTOR CONTROLLER (Cont'd)

Figure 76. Typical command signals of a three-phase induction motor



* These simplified chronograms represent the phase voltages after low-pass filtering of the PWM outputs reference signals

MOTOR CONTROLLER (Cont'd)

10.6.5 Functional Description

The MTC can be split into five main parts as shown in the simplified block diagram in Figure 77. Each of these parts may be configured for different purposes:

- INPUT DETECTION BLOCK with a comparator, an input multiplexer and an incremental encoder interface, which may work as:
  - A BEMF zero-crossing detector
  - A Speed Sensor Interface
- The DELAY MANAGER with an 8/16-bit timer and an 8x8 bit multiplier, which may work as a:
  - 8-bit delay manager
  - Speed Measurement unit
- The PWM MANAGER, including a measurement window generator, a mode selector and a current comparator.
- The CHANNEL MANAGER with the PWM multiplexer, polarity programming, deadtime insertion and high frequency chopping capability and emergency HiZ configuration input.
- The THREE-PHASE PWM GENERATOR with 12-bit free-running counter and repetition counter.

10.6.6 Input Detection Block

This block can operate in Position sensor mode, in sensorless mode or in Speed Sensor mode. The mode is selected via the SR bit in the MCRA register and the TES[1:0] bits in MPAR register (refer to Table 35 for set-up information). The block diagram is shown in Figure 78 for the Position Sensor/Sensorless modes (TES[1:0] = 00) and in Figure 88 for the Speed Sensor mode (TES[1:0] = 01, 10, 11).

10.6.6.1 Input Pins

The MCIA, MCIB and MCIC input pins can be used as analog or as digital pins.

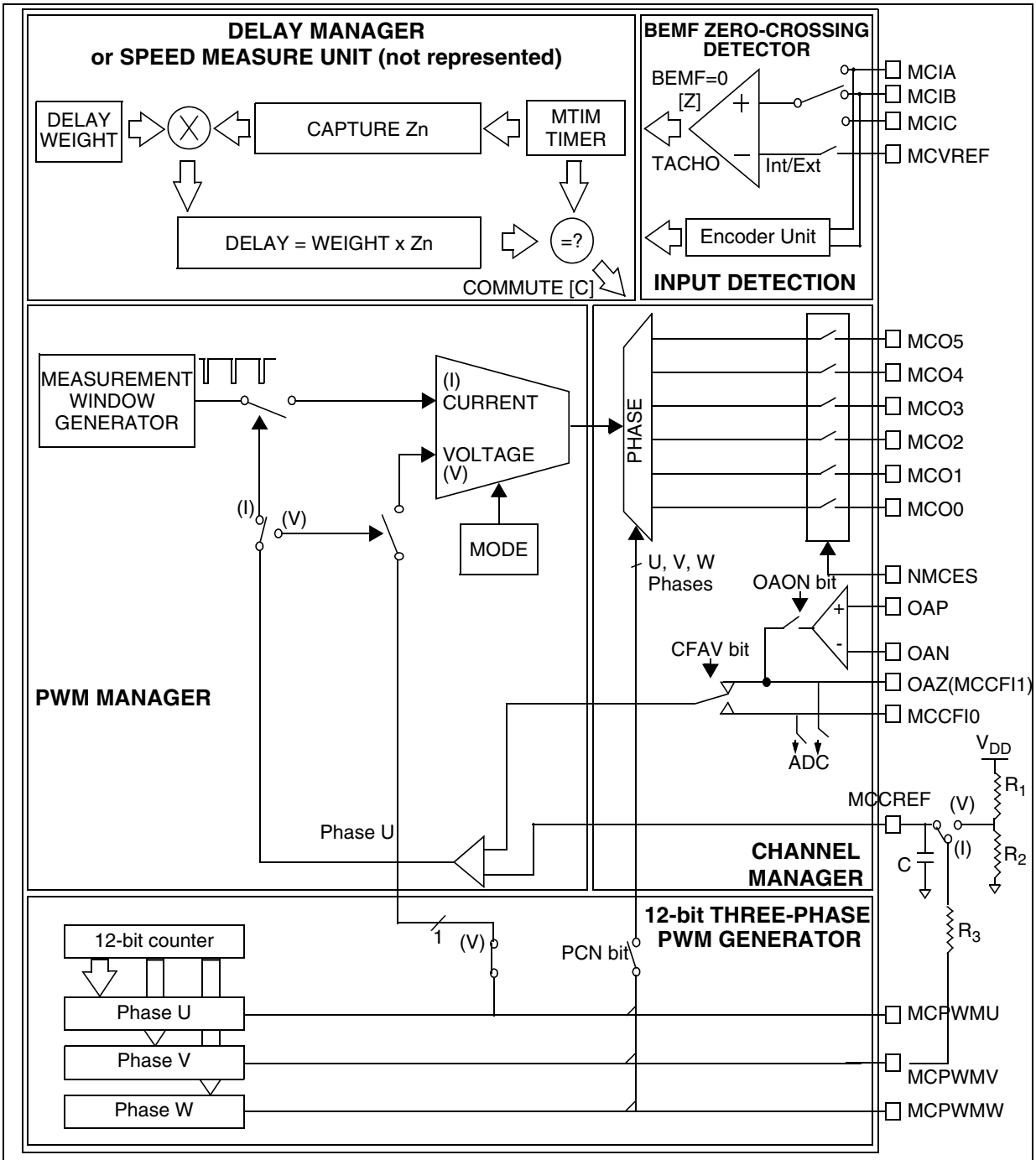
- In sensorless mode, the analog inputs are used to measure the BEMF zero crossing and to detect the end of demagnetization if required.
- In sensor mode, the analog inputs are used to get the Hall sensor information.
- In speed sensor mode (e.g. tachogenerator), the inputs are used as digital pins. When using an AC tachogenerator, a small external circuit may be needed to convert the incoming signal into a square wave signal which can be treated by the MTC.

Due to the presence of diodes, these pins can permanently support an input current of 5mA. In sensorless mode, this feature enables the inputs to be connected to each motor phase through a single resistor.

A multiplexer, programmed by the IS[1:0] bits in the MPHST register selects the input pins and connects them to the control logic in either sensorless or tachogenerator mode. In encoder mode, it is mandatory to connect sensor digital outputs to the MCIA and MCIB pins.

MOTOR CONTROLLER (Cont'd)

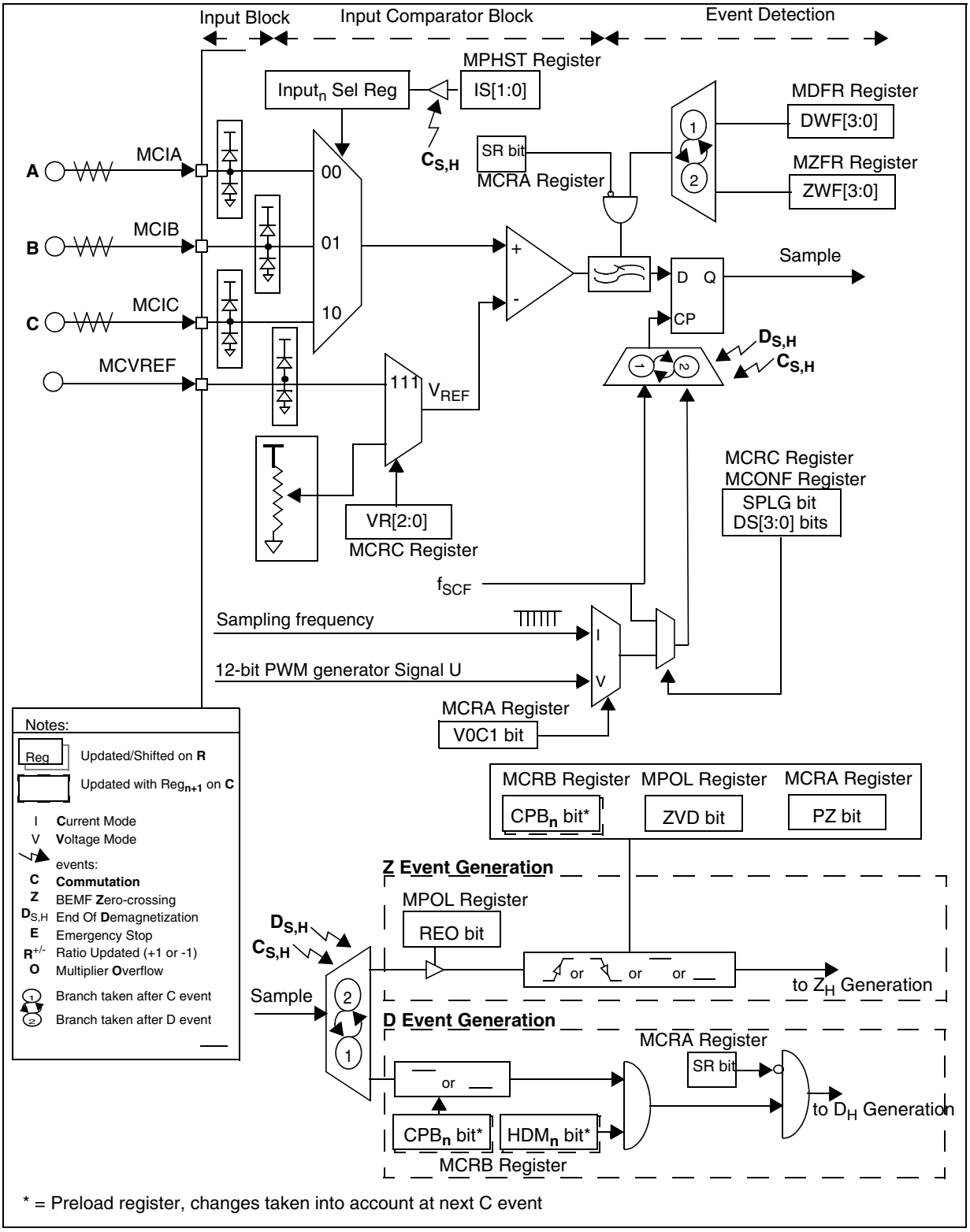
Figure 77. Simplified MTC Block Diagram



[Z] : Back EMF Zero-crossing event
Z_n : Time elapsed between two consecutive Z events
[C] : Commutation event
C_n : Time delayed after Z event to generate C event
(I): Current mode
(V): Voltage mode

MOTOR CONTROLLER (Cont'd)

Figure 78. Input Stage in Sensorless or Sensor Mode (bits TES[1:0] = 00)



MOTOR CONTROLLER (Cont'd)

10.6.6.2 Sensorless Mode

This mode is used to detect BEMF zero crossing and end of demagnetization events.

The analog phase multiplexer connects the non-excited motor winding to an analog 100mV hysteresis comparator referred to a selectable reference voltage.

IS[1:0] bits in MPHST register allow to select the input which will be drive to the comparator (either MCIA, B or C). Be careful that the comparator is OFF until CKE and/or DAC bit are set in MCRA register.

The VR[2:0] bits in the MCRC register select the reference voltage from seven internal values depending on the noise level and the application voltage supply. The reference voltage can also be set externally through the MCVREF pin when the VR[2:0] bits are set.

Table 25. Threshold voltage setting

VR2	VR1	VR0	Vref voltage threshold
1	1	1	Threshold voltage set by external MCVREF pin
1	1	0	3.5V*
1	0	1	2.5V*
1	0	0	2V*
0	1	1	1.5V*
0	1	0	1V*
0	0	1	0.6V*
0	0	0	0.2V*

*Typical value for V_{DD}=5V.

BEMF detections are performed during the measurement window, when the excited windings are free-wheeling through the low side switches and diodes. At this stage the common star connection voltage is near to ground voltage (instead of $V_{DD}/2$ when the excited windings are powered) and the complete BEMF voltage is present on the non-excited winding terminal, referred to the ground terminal.

The zero crossing sampling frequency is then defined, in current mode, by the measurement window generator frequency (SA[3:0] bits in the MPRSR register) or, in voltage mode, by the PWM generator frequency and phase U duty cycle.

During a short period after a phase commutation (C event), the winding where the back-emf will be read is no longer excited but needs a demagnetisation phase during which the BEMF cannot be read. A demagnetization current goes through the free-wheeling diodes and the winding voltage is stuck at the high voltage or to the ground terminal. For this reason an "end of demagnetization event" D must be detected on the winding before the detector can sense a BEMF zero crossing.

For the end-of-demagnetization detection, no special PWM configuration is needed, the comparator sensing is done at a selectable frequency (f_{SCF}), see Table 82.

So, the three events: C (commutation), D (demagnetization) and Z (BEMF zero crossing) must always occur in this order in autoswitched mode when hard commutation is selected.

The comparator output is processed by a detector that automatically recognizes the D or Z event, depending on the CPB or ZVD edge and level configuration bits as described in Table 30.

To avoid wrong detection of D and Z events, a blanking window filter is implemented for spike filtering. In addition, by means of an event counter, software can filter several consecutive events up to a programmed limit before generating the D or Z event internally. This is shown in Figure 79 and Figure 80.

MOTOR CONTROLLER (Cont'd)

10.6.6.3 D Event detection

In sensorless mode, the D Window Filter becomes active after each C event. It blanks out the D event during the time window defined by the DWF[3:0] bits in the MDFR register (see Table 26). The reset value is 200µs.

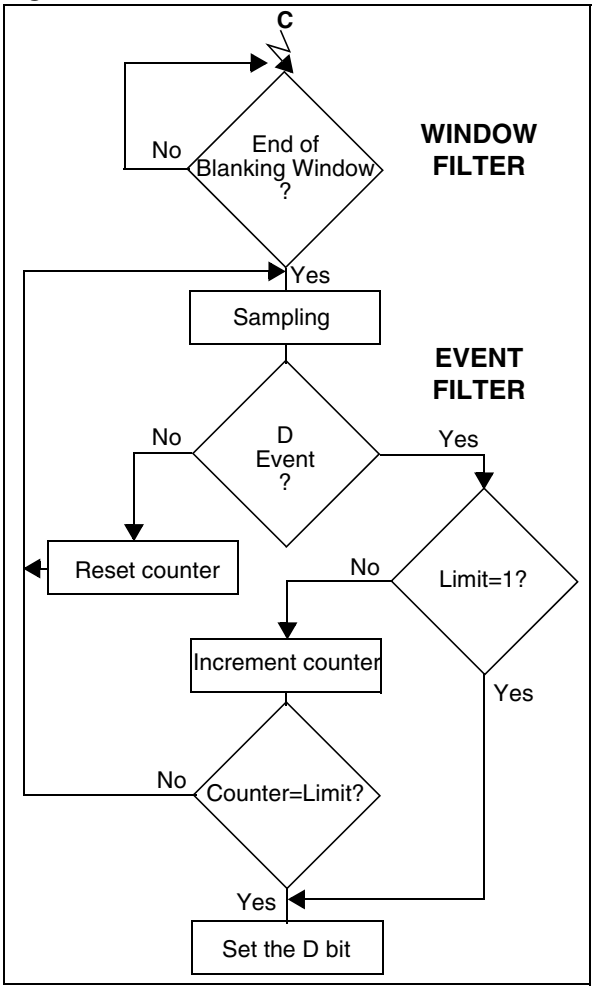
This Window Filter becomes active after both hardware and software C events.

The D Event Filter becomes active after the D Window Filter. It counts the number of consecutive D events up to a limit defined by the DEF[3:0] bits in the MDFR register. The reset value is 1. The D bit is set when the counter limit is reached.

Sampling is done at a selectable frequency (f_{SCF}), see Table 82.

The D event filter is active only for a hardware D event (D_H). For a simulated (D_S) event, it is forced to 1.

Figure 79. D Window and Event Filter Flowchart



DWF3	DWF2	DWF1	DWF0	C to D window filter in Sensorless Mode (SR=0)	SR=1
0	0	0	0	5 µs	No Window Filter after C event
0	0	0	1	10 µs	
0	0	1	0	15 µs	
0	0	1	1	20 µs	
0	1	0	0	25 µs	
0	1	0	1	30 µs	
0	1	1	0	35 µs	
0	1	1	1	40 µs	
1	0	0	0	60 µs	
1	0	0	1	80 µs	
1	0	1	0	100 µs	
1	0	1	1	120 µs	
1	1	0	0	140 µs	
1	1	0	1	160 µs	
1	1	1	0	180 µs	
1	1	1	1	200 µs	

Note: Times are indicated for 4 MHz f_{PERIPH}

Table 27. D Event filter Setting

DEF3	DEF2	DEF1	DEF0	D event Limit	SR=1
0	0	0	0	1	No D Event Filter
0	0	0	1	2	
0	0	1	0	3	
0	0	1	1	4	
0	1	0	0	5	
0	1	0	1	6	
0	1	1	0	7	
0	1	1	1	8	
1	0	0	0	9	
1	0	0	1	10	
1	0	1	0	11	
1	0	1	1	12	
1	1	0	0	13	
1	1	0	1	14	
1	1	1	0	15	
1	1	1	1	16	

MOTOR CONTROLLER (Cont'd)

10.6.6.4 Z Event Detection

In sensorless mode, the Z window filter becomes active after each D event. It blanks out the Z event during the time window defined by the ZWF[3:0] bits in the MZFR register (see Table 28). The reset value is 200µs. This Window Filter becomes active after both hardware and software D events.

The Z Event Filter becomes active after the Z Window Filter. It counts the number of consecutive Z events up to a limit defined by the ZEF[3:0] bits in the MZFR register. The reset value is 1. The Z bit is set when the counter limit is reached.

Sampling is done at a selectable frequency (f_{SCF}), see Table 82.

The Z event filter is active only for a hardware Z event (Z_H). For a simulated (Z_S) event, it is forced to 1. Z event filter is also active in sensor mode.

Figure 80. Z Window and Event Filter Flowchart

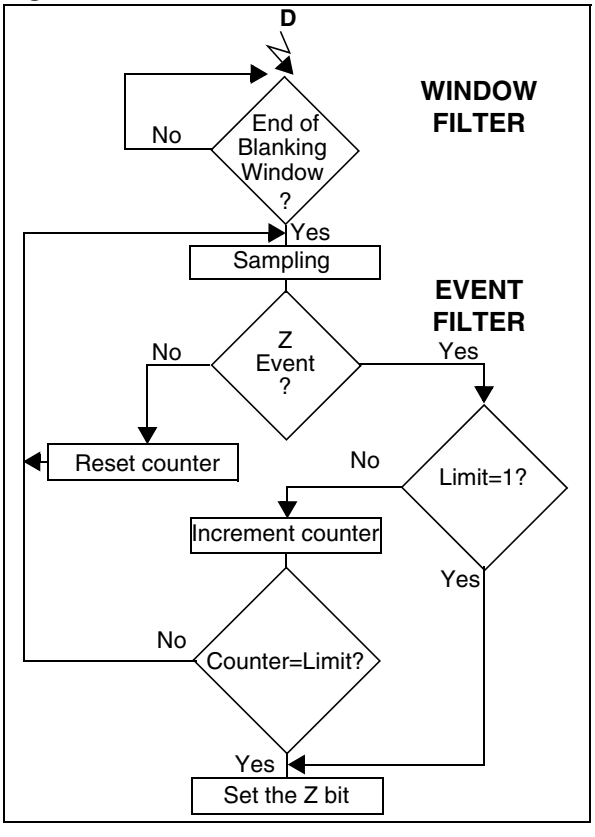


Table 28. Z Window filter Setting

ZWF3	ZWF2	ZWF1	ZWF0	D to Z window filter in Sensorless Mode (SR=0)	SR=1
0	0	0	0	5 µs	No Window Filter after D event
0	0	0	1	10 µs	
0	0	1	0	15 µs	
0	0	1	1	20 µs	
0	1	0	0	25 µs	
0	1	0	1	30 µs	
0	1	1	0	35 µs	
0	1	1	1	40 µs	
1	0	0	0	60 µs	
1	0	0	1	80 µs	
1	0	1	0	100 µs	
1	0	1	1	120 µs	
1	1	0	0	140 µs	
1	1	0	1	160 µs	
1	1	1	0	180 µs	
1	1	1	1	200 µs	

Note: Times are indicated for 4 MHz f_{PERIPH}

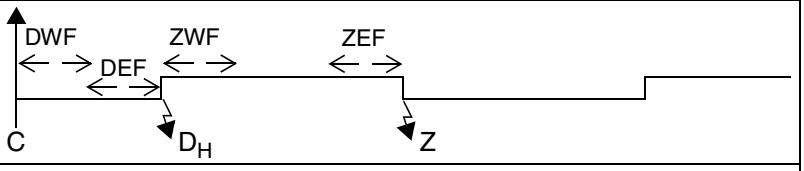
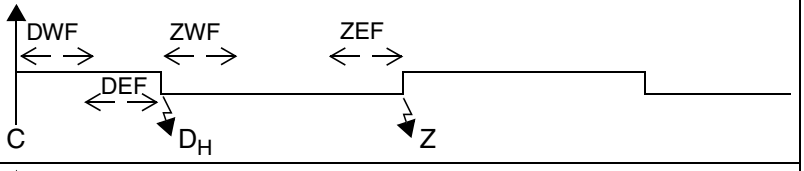
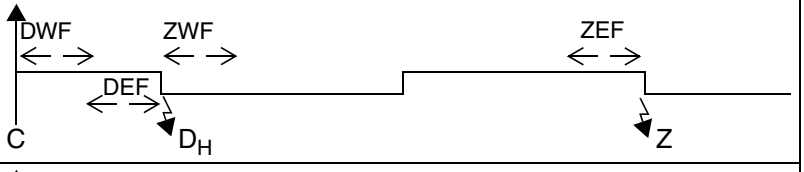
Table 29. Z Event filter Setting

ZEF3	ZEF2	ZEF1	ZEF0	Z event Limit
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14
1	1	1	0	15
1	1	1	1	16

MOTOR CONTROLLER (Cont'd)

Table 30 shows the event control selected by the ZVD and CPB bits. In most cases, the D and Z events have opposite edge polarity, so the ZVD bit is usually 0.

Table 30. ZVD and CPB Edge Selection Bits

ZVD bit	CPB bit	Event generation vs input data sampled
0	0	
0	1	
1	0	
1	1	
Note: The ZVD bit is located in the MPOL register, the CPB bit is in the MCRB register.		

Legend:

DWF= D window filter

DEF= D event filter

ZWF = Z window filter

ZEF = Z event filter

Refer also to Table 34 on page 162.

MOTOR CONTROLLER (Cont'd)

10.6.6.5 Demagnetization (D) Event

At the end of the demagnetization phase, current no longer goes through the free-wheeling diodes. The voltage on the non-excited winding terminal goes from one of the power rail voltages to the common star connection voltage plus the BEMF voltage. In some cases (if the BEMF voltage is positive and the free-wheeling diodes are at ground for example) this end of demagnetization can be seen as a voltage edge on the selected MCIx input and it is called a hardware demagnetization event D_H. See Table 30.

The D event filter can be used to select the number of consecutive D events needed to generate the D_H event.

If enabled by the HDM bit in the MCRB register, the current value of the MTIM timer is captured in register MDREG when this event occurs in order to be able to simulate the demagnetization phase for the next steps.

When enabled by the SDM bit in the MCRB register, demagnetization can also be simulated by comparing the MTIM timer with the MDREG register. This kind of demagnetization is called simulated demagnetization D_S.

If the HDM and SDM bits are both set, the first event that occurs, triggers a demagnetization event. For this to work correctly, a D_S event must not precede a D_H event because the latter could be detected as a Z event.

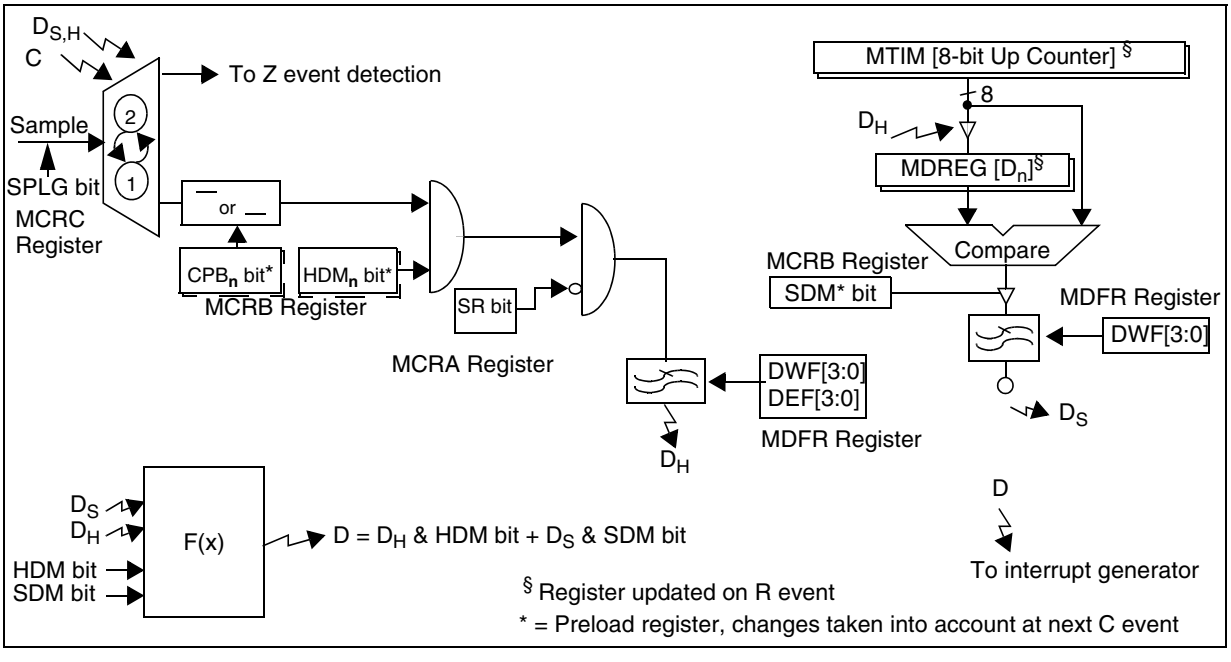
Simulated demagnetization can also be always used if the HDM bit is reset and the SDM bit is set. This mode works as a programmable masking time between the C_H and Z events. To drive the motor securely, the masking time must be always greater than the real demagnetization time in order to avoid a spurious Z event.

When an event occurs, (either D_H or D_S) the DI bit in the MISR register is set and an interrupt request is generated if the DIM bit of register MIMR is set.

Caution 1: Due to the alternate automatic capture and compare of the MTIM timer with MDREG register by D_H and D_S events, the MDREG register should be manipulated with special care.

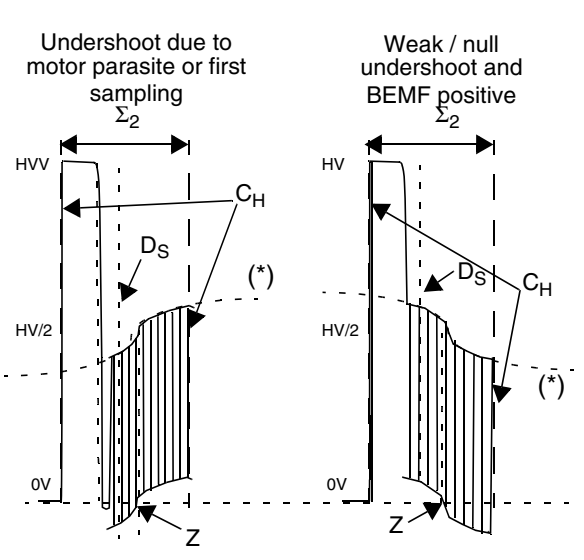
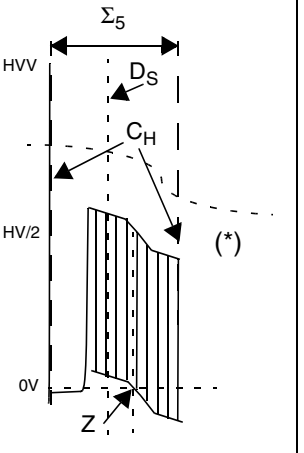
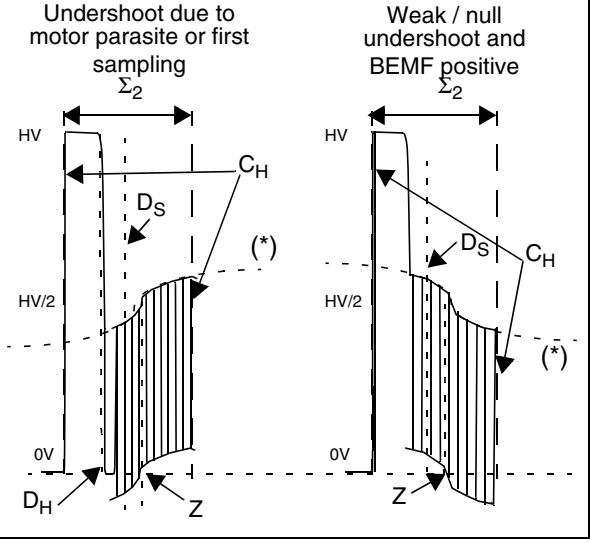
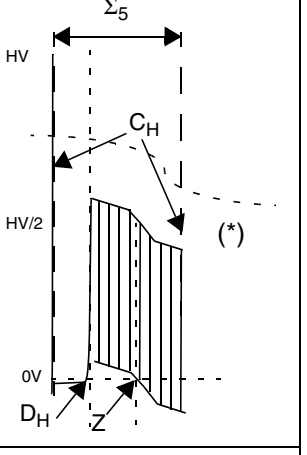
Caution 2: Due to the event generation protection in the MZREG, MCOMP and MDREG registers for Soft Event generation (See "Built-in Checks and Controls for simulated events" on page 175.), the value written in the MDREG register in soft demagnetisation mode (SDM=1) is checked by hardware after the C event. If this value is less than or equal to the MTIM counter value at this moment, the Software demagnetisation event is generated immediately and the MTIM current value overwrites the value in the MDREG register to be able to reuse the right demagnetisation time for another simulated event generation.

Figure 81. D Event Generation Mechanism



MOTOR CONTROLLER (Cont'd)

Table 31. Demagnetisation (D) Event Generation (example for ZVD=0)

<table>
<tr><th>HDM bit</th><th>Meaning</th><th>CPB bit = 1</th><th>CPB bit = 0</th></tr>
<tr><td rowspan="2">0</td><td rowspan="2">Simulated Mode
(SDM bit =1 and HDM bit = 0)</td><td colspan="2">D = D_S = Output Compare [MDREG, MTIM registers]</td></tr>
<tr><td> 
</td><td> 
</td></tr>
<tr><td>1</td><td>Hardware/Simulated Mode
(SDM bit = 1 and HDM bit = 1)</td><td>D = D_H + D_S
(Hardware detection or Output compare true)
 
</td><td>D = D_H
(Hardware detection only)
 
</td></tr>
<tr><td colspan="4">(*) Note: This is a zoom to the additional voltage induced by the rotor (Back EMF)</td></tr>
</table>

MOTOR CONTROLLER (Cont'd)

10.6.6.6 Z Event Generation (BEMF Zero Crossing)

When both C and D events have occurred, the PWM may be switched to another group of outputs (depending on the OS[2:0] bits in the MCRB register) and the real BEMF zero crossing sampling can start (see Figure 87). After Z event, the PWM can also be switched to another group of outputs before the next C event.

A BEMF voltage is present on the non-powered terminal but referred to the common star connection of the motor whose voltage is equal to $V_{DD}/2$.

When a winding is free-wheeling (during PWM off-time) its terminal voltage changes to the other power rail voltage, this means if the PWM is applied on the high side driver, free-wheeling will be done through the low side diode and the terminal will be 0V.

This is used to force the common star connection to 0V in order to read the BEMF referred to the ground terminal.

Consequently, BEMF reading (i.e. comparison with a voltage close to 0V) can only be done when the PWM is applied on the high side drivers. When the BEMF signal crosses the threshold voltage close to zero, it is called a hardware zero-crossing event Z_H. A filter can be implemented on the Z_H event detection (see Figure 83).

The Z event filter register (MZFR) is used to select the number of consecutive Z events needed to generate the Z_H event. Alternatively, the PZ bit can be used to enable protection as described in Figure 83. on page 157

For this reason the MTC outputs can be split in two groups called LOW and HIGH and the BEMF reading will be done only when PWM is applied on one of these two groups. The REO bit in the MPOL register is used to select the group to be used for BEMF sensing (high side group). It has to be configured whatever the sampling mode.

When enabled by the HZ bit in MCRC register, the current value of the MTIM timer is captured in register MZREG when this event occurs in order to be able to compute the real delay in the delay manager part for hardware commutation but also to be able to simulate zero-crossing events for other steps.

When enabled by the SZ bit set in the MCRC register, a zero-crossing event can also be simulated by comparing the MTIM timer value with the MZREG register. This kind of zero-crossing event is called simulated zero-crossing Z_S.

If both HZ and SZ bits are set in MCRC register, the first event that occurs, triggers a zero-crossing event.

Depending on the edge and level selection (ZVD and CPB) bits and when PWM is applied on the correct group, a BEMF zero crossing detection (either Z_H or Z_S) sets the ZI bit in the MISR register and generates an interrupt if the ZIM bit is set in the MIMR register.

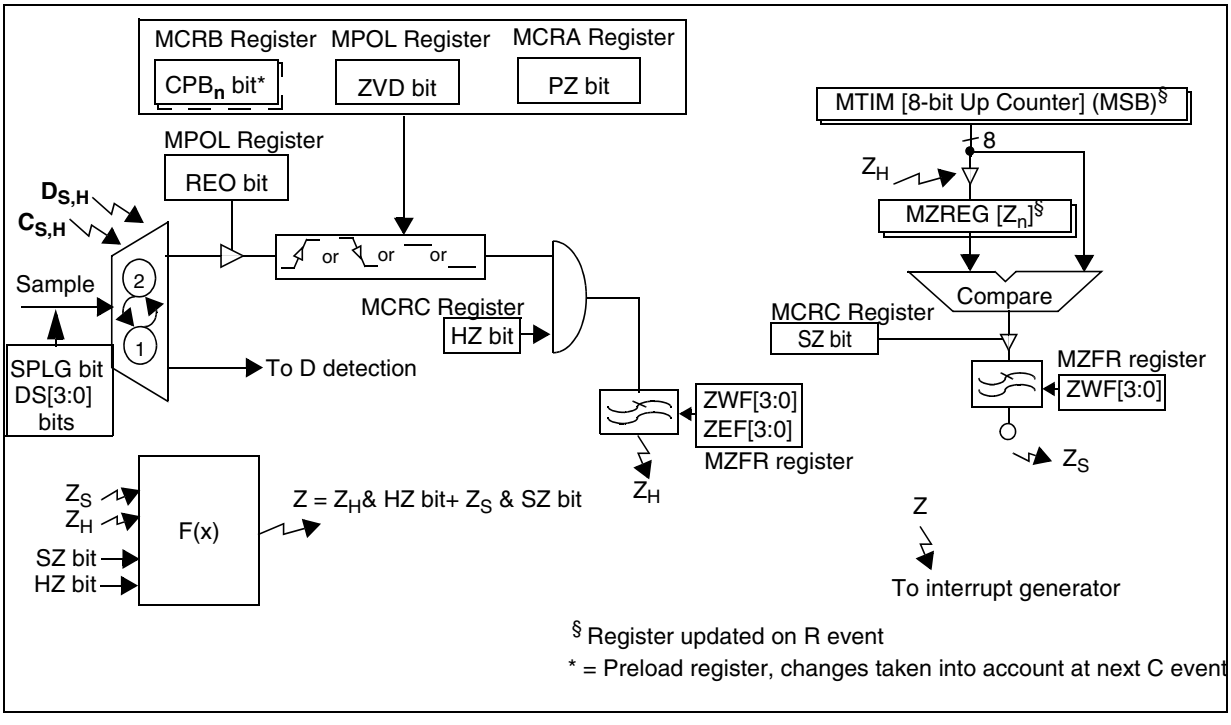
Caution 1: Due to the alternate automatic capture and compare of the MTIM timer with MZREG register by Z_H and Z_S events, the MZREG register should be manipulated with special care.

Caution 2: Due to the event generation protection in the MZREG, MCOMP and MDREG registers for Soft Event generation, the value written in the MZREG register in simuated zero-crossing mode (SZ=1) is checked by hardware after the D (either D_H or D_S) event. If this value is less than or equal to the MTIM counter value at this moment, the simulated zero-crossing event is generated immediately and the MTIM current value overwrites the value in the MZREG register. See "Built-in Checks and Controls for simulated events" on page 175.

The Z event also triggers some timer/multiplier operations, for more details see Section 10.6.7

MOTOR CONTROLLER (Cont'd)

Figure 82. Z Event Generation



MOTOR CONTROLLER (Cont'd)

10.6.6.7 Protection for Z_H event detection

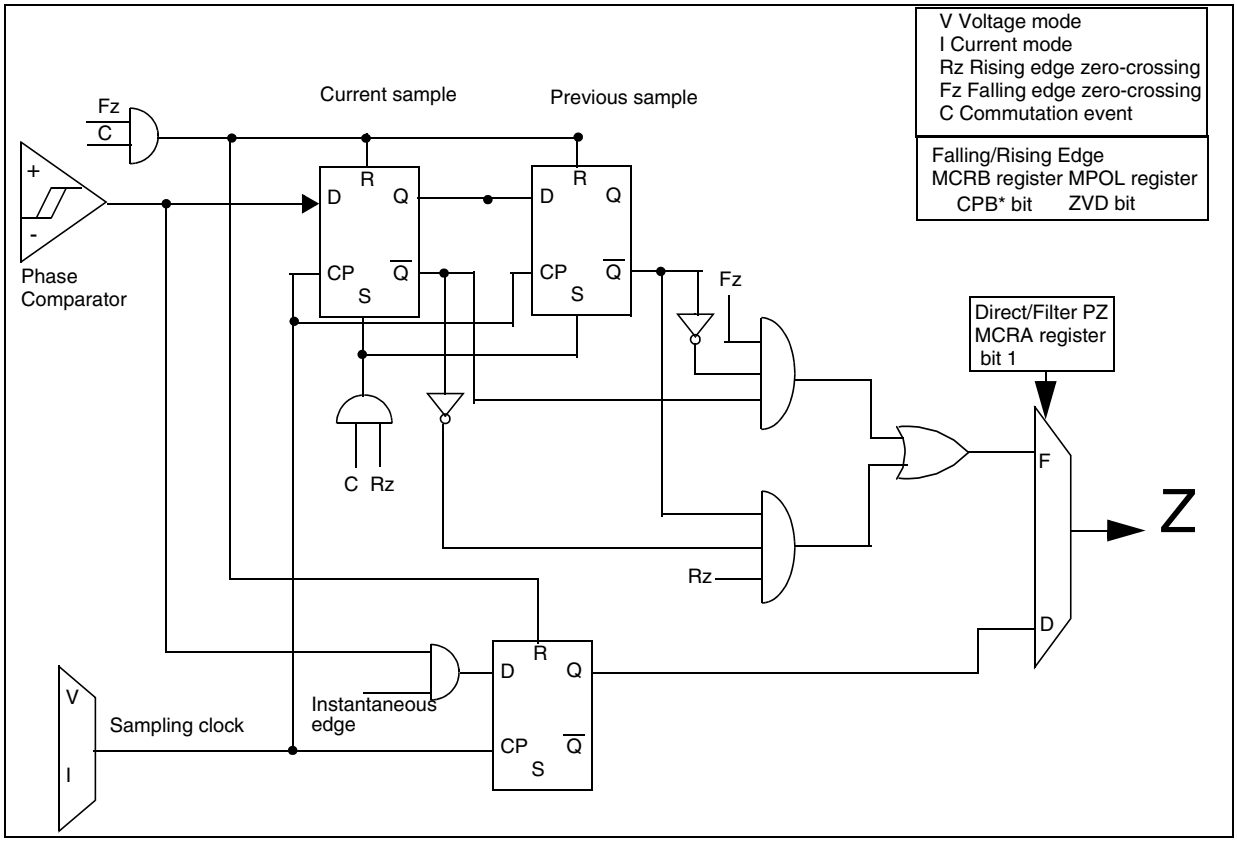
To avoid an erroneous detection of a hardware zero-crossing event, a filter can be enabled by setting the PZ bit in the MCRA register. This filter will ensure the detection of a Z_H event on an edge transition between D event and Z_H event.

Without this protection, Z_H event detection is done directly on the current sample in comparison with the expected state at the output of the phase comparator. For example, if a falling edge transition (meaning a transition from 1 to 0 at the output of the phase comparator) is configured for Z_H event through the CPB bit in MCRB register, then, the state 0 is expected at the comparator output and once this state is detected, the Z_H event is generated without any verification that the state at the comparator output of the previous sample was 1. The purpose of this protection filter is to be sure that the state of the comparator output at the sample before was really the opposite of the current state which is generating the Z_H event. With this filter, the Z_H event generation is done on edge transition level comparison.

This filter is not needed in sensor mode (SR=1) and for simulated zero-crossing event (Z_S) generation.

When the PZ bit is set, the Z event filter ZEF[3:0] in the MZFR register is ignored.

Figure 83. Protection of Z_H event detection



MOTOR CONTROLLER (Cont'd)

10.6.6.8 Position Sensor Mode

In position sensor mode (SR=1 in MCRA register), the rotor position information is given to the peripheral by means of logical data on the three inputs MCIA, MCIB and MCIC (Hall sensors).

For each step one of these three inputs is selected (IS[1:0] bits in register MPHST) in order to detect the Z event. Be careful that the phase comparator is OFF until CKE and /or DAC bits are set in MCRA register.

In sensor mode, Demagnetization and the related features (such as the special PWM configuration, D_S or D_H management, programmable filter) are not available (see Table 32)

Table 32. Demagnetisation access

SR bit MCRA register	Demagnetisation feature availabilty
1	NO
0	YES

In sensor mode configuration the rotor detection doesn't need a particular phase configuration to perform the measurement and a Z event can be read from any detection window. The sampling is done at a selectable frequency (f_{SCF}), see Table 82. This means that Z event position sensoring is more precise than it is in sensorless mode.

There is no minimum off time required for current control PWM in sensor mode so the minimum off time is set automatically to 0µs as soon as the SR bit is set in the MCRA register and a true 100% duty cycle can be set in the PWM compare U register for the PWM generation in voltage mode.

In Sensor mode, the ZEF[3:0] bits in the MZFR register are active and can be used to define the number of consecutive Z samples needed to generate the active event.

Procedure for reading sensor inputs in Direct Access mode: In Direct Access mode, the sensors can be read either when the clock are enabled or disabled (depending on CKE it in MCRA register). To read the sensor data the following steps have to be performed:

1. Select Direct Access Mode (DAC bit in MCRA register)
2. Select the appropriate MCIx input pin by means of the IS[1:0] bits in the MPHST register
3. Read the comparator output (HST bit in the MREF register)

MOTOR CONTROLLER (Cont'd)

10.6.6.9 Sampling block

For a full digital solution, the phase comparator output sampling frequency is the frequency of the PWM signal applied to the switches and the sampling for the Z event detection in sensorless mode is done at the end of the off time of this PWM signal to avoid to have to re-create a virtual ground because when the PWM signal is off, the star point is at ground due to the free-wheeling diode. That's why, the sampling for Z event detection is done by default during the OFF-state of the PWM signal and therefore at the PWM frequency.

In current mode, this PWM signal is generated by a combination of the output of the measurement window generator (SA[3:0] bits), the output of the current comparator and a minimum OFF time set by the OT[3:0] bits for system stabilisation.

In voltage mode, this PWM signal is generated by the 12-bit PWM generator signal in the compare U register with still a minimum OFF time required if the sampling is done at the end of the OFF time of the PWM signal for system stabilisation. The PWM signal is put OFF as soon as the current feedback reaches the current input limitation. This can add an OFF time to the one programmed with the 12-bit Timer.

For D event detection in sensorless mode, no specific PWM configuration is needed and the sampling frequency ($f_{SCF,}$ see Table 82) is completely independent from the PWM signal.

In sensor mode, the D event detection is not needed as the MCIA, MCIB and MCIC pins are the digital signals coming from the hall sensors so no specific PWM configuration is needed and the sampling for the Z detection event is done at f_{SCF}, completely independent from the PWM signal.

In sensorless mode, if a virtual ground is created by the addition of an external circuit, sampling for the Z event detection can be completely independent from the PWM signal applied to the switches. Setting the SPLG bit in the MCRC register allows a sampling frequency of f_{SCF} for Z event detection independent from the PWM signal after getting the D (end of demagnetisation) event. This means that the sampling order is given whatever the PWM signal (during the ON time or the OFF time). As soon as the SPLG bit is set in the MCRC register, the minimum OFF time needed for the PWM signal in current mode is set to 0µs and a true 100% duty cycle can be set in the 12-bit PWM generator compare register in voltage mode.

Specific applications can require sampling for the Z event detection only during the ON time of the PWM signal. This can happen when the PWM signal is applied only on the low side switches for Z event detection. In this case, during the OFF time of the PWM signal, the phase voltage is tied to the application voltage V and no back-EMF signal can be seen. During the ON time of the PWM signal, the phase voltage can be compared to the neutral point voltage and the Z event can be detected. Therefore, it is possible to add a programmable delay before sampling (which is normally done when the PWM signal is switched ON) to perform the sampling during the ON time of the PWM signal. This delay is set with the DS [3:0] bits in the MCONF register.

Table 33. Delay length before sampling

DS3	DS2	DS1	DS0	Delay added to sample at Ton
0	0	0	0	No delay added. Sample during Toff
0	0	0	1	2.5 µs
0	0	1	0	5 µs
0	0	1	1	7.5 µs
0	1	0	0	10 µs
0	1	0	1	12.5 µs
0	1	1	0	15 µs
0	1	1	1	17.5 µs
1	0	0	0	20 µs
1	0	0	1	22.5 µs
1	0	1	0	25 µs
1	0	1	1	27.5 µs
1	1	0	0	30 µs
1	1	0	1	32.5 µs
1	1	1	0	35 µs
1	1	1	1	37.5 µs

Note: Times are indicated for 4 MHz f_{PERIPH}

As soon as a delay is set in the DS[3:0] bits, the minimum OFF time for the PWM signal is no longer required and it is automatically set to 0µs in current mode in the internal sampling clock and a true 100% duty cycle can be set in the 12-bit PWM generator compare U register if needed.

MOTOR CONTROLLER (Cont'd)

Depending on the frequency and the duty cycle of the PWM signal, the delay inserted before sampling could cause it sample the signal OFF time instead of the ON time. In this case an interrupt can be generated and the sample will not be taken into acount. When a sample occurs outside the PWM signal ON time, the SOI bit in the MCONF register is set and an interrupt request is generated if the SOM bit is set in the MCONF register. This interrupt is enabled only if a delay value has been set in the DS[3:0] bits. In this case, the sampling is done at the PWM frequency but only during the ON time of the PWM signal. Figure 84 and Figure 85 shows in detail the generation of the sampling order when the delay is added.

For complete flexibility, the possibility of sampling at f_{SCF} high frequency during the ON time of the PWM signal is also available when the SPLG bit is set as if there is a delay value in the DS[3:0] bits. This means that when the sampling is to be performed, after the delay a sampling window at f_{SCF} frequency is opened until the next OFF time of the PWM signal. The Sampling Out interrupt will be generated if the delay added is longer than the duty cycle of the PWM signal. As the SPLG bit is set and a value has been put in the DS[3:0] bits, no minimum off time is required for the PWM signal and it is automatically set to 0µs in current mode. A true 100% duty cycle can be also set in the 12-bit Timer in voltage mode. Figure 86 shows in detail the sampling at f_{SCF} high frequency during ON time.

Figure 84. Adding the Delay to sample during ON time for Z detection

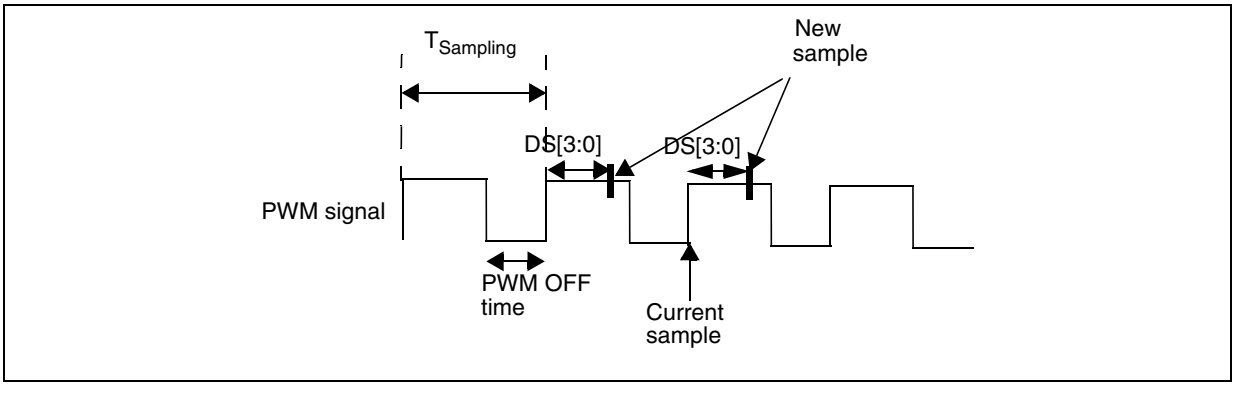
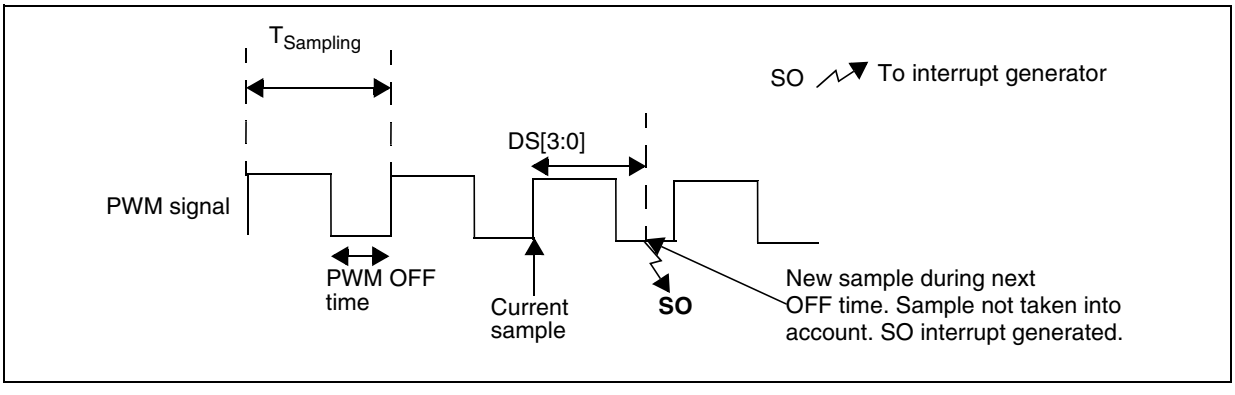


Figure 85. Sampling Out interrupt generation



MOTOR CONTROLLER (Cont'd)

In conclusion, there are 4 sampling types that are available for Z event detection in sensorless mode.

1. Sampling at the end of the OFF time of the PWM signal at the PWM frequency
2. Sampling, at a programmable frequency independent of the PWM state (during ON time or OFF time of the signal). Sampling is done at f_{SCF}, see Table 82.
3. Sampling during the ON time of the PWM signal by adding a delay at PWM frequency
4. Sampling, at a programmable frequency during the ON time (addition of a programmable delay) of the PWM signal. Sampling is done at f_{SCF}, see Table 82.

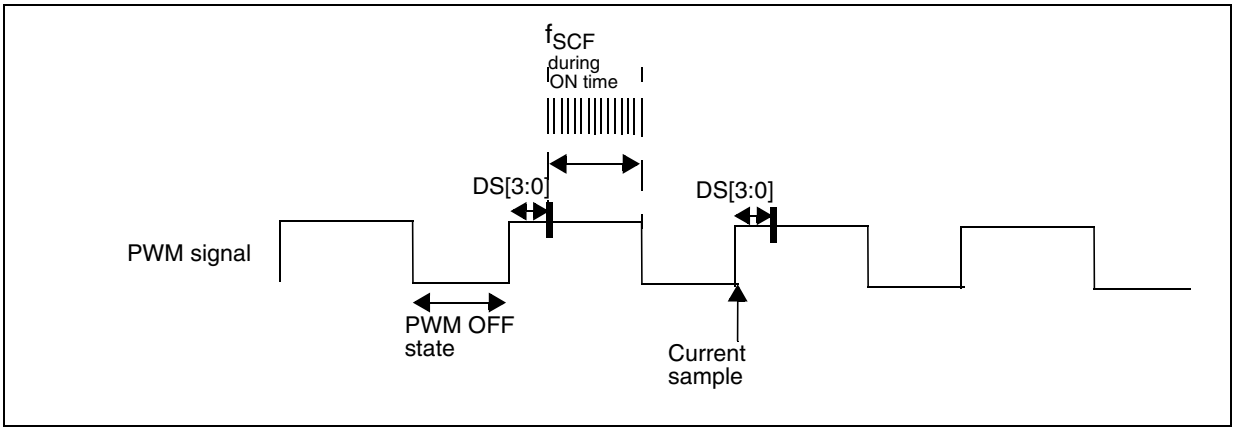
Note 1: The sampling type is applied only for Z event detection after the D event has occured. Whatever the sampling type for Z event detection, the sampling of the signal for D event detection is always done at the selected f_{SCF} frequency (see Table 82), independently of the PWM signal (either during ON or OFF time). Table 34 explains the different sampling types in sensorless and in sensor mode.

Note 2: When the MOE bit in the MCRA register is reset (MCOx outputs in reset state), and the SR bit in the MCRA register is reset (sensorless mode) and the SPLG bit in the MCRC register is reset (sampling at PWM frequency) then, depending on the state of the ZSV bit in the MSCR register, Z event sampling can run or be stopped (and D event is sampled).

Note 3: When BEMF sampling is performed at the end of the PWM signal off-time, the inputs in OFF-state are grounded or put in HiZ as selected by the DISS bit in the MSCR register.

Note 4: The ZEF[3:0] event counter in the MZFR register is active in all configurations.

Figure 86. Sampling during ON time at f_{SCF}



MOTOR CONTROLLER (Cont'd)

10.6.6.10 Commutation Noise Filter

For D event detection and for Z event detection (when SPLG bit is set while DS[3:0] bits are reset), sampling is done at f_{SCF} during the PWM ON or OFF time ("Sampling block" on page 159). To avoid any erroneous detection due to PWM commutation noise, an hardware filter of 1µs (for $f_{PER-IPH}$ = 4Mhz) when PWM is put ON and when PWM is put OFF has been implemented. This means that, with sampling at 1MHz (1µs), due to this filter, 1 sample are ignored directly after the commutation.

This filter is active all the time for the D event and it is active for the Z event when the SPLG bit is set and DS[3:0] bits are cleared (meaning that the Z event is sampled at high frequency during the PWM ON or OFF time).

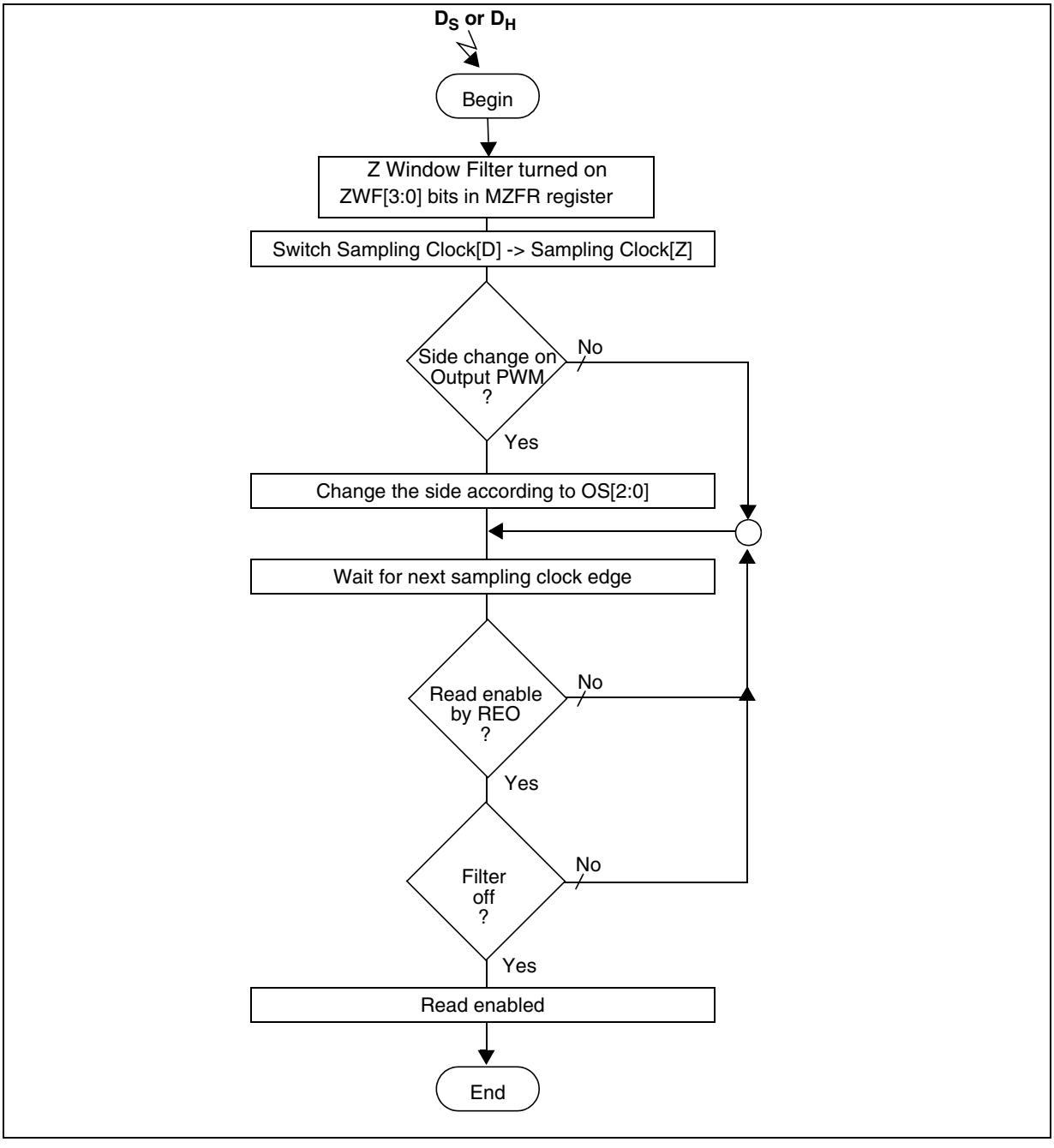
Table 34. Sensor/sensorless mode and D & Z event selection

SR bit	SPLG bit	DS[3:0] bits	Mode	OS[2:0] bits use	Event detection sampling clock	Sampling behaviour for Z event detection	Window and Event Filters	Behaviour of the output PWM
0	0	000	Sensors not used	Enabled	D: f_{SCF} Z: SA&OT config. PWM frequency	At the end of the off time of the PWM signal	D Window Filter DWF[3:0] after C event D Event Filter DEF[3:0] after DWF Z Window Filter ZWF[3:0] after D event Z Event Filter ZEF[3:0] after ZWF See Table 30 on page 152	"Before D" behaviour, "between D and Z" behaviour and "after Z" behaviour
0	1	000	Sensors not used	Enabled	D: f_{SCF} Z: f_{SCF}	During off time or ON time of the PWM signal		"Before D" behaviour, "between D and Z" behaviour and "after Z" behaviour
0	0	Not equal to 000	Sensors not used	Enabled	D: f_{SCF} Z: SA&OT config. PWM frequency	During ON time of the PWM signal		"Before D" behaviour, "between D and Z" behaviour and "after Z" behaviour
0	1	Not equal to 000	Sensors not used	Enabled	D: f_{SCF} Z: f_{SCF}	During ON time of the PWM signal		"Before D" behaviour, "between D and Z" behaviour and "after Z" behaviour
1	x	xxx	Position Sensors used	OS1 disabled	Z: f_{SCF}	During OFF time or ON time of the PWM signal	No Z Window Filter Only Z Event Filter is active in Sensor mode	"Before Z" behaviour and "after Z" behaviour

Note: For f_{SCF} selection, see Table 82

MOTOR CONTROLLER (Cont'd)

Figure 87. Functional Diagram of Z Detection after D Event



MOTOR CONTROLLER (Cont'd)

10.6.6.11 Speed Sensor Mode

This mode is entered whenever the Tacho Edge Selection bits in the MPAR register are not both reset (TES[1:0] = 01, 10 or 11). The corresponding block diagram is shown in Figure 88.

Either Incremental Encoder or Tachogenerator-type speed sensor can be selected with the IS[1:0] bits in the MPHST register.

10.6.6.12 Tachogenerator Mode (IS[1:0] = 00, 01 or 10)

Any of the MCIx input pins can be used as a tachogenerator input, with a digital signal (externally amplified for instance); the two remaining pins can be used as standard I/O ports.

A digital multiplexer connects the chosen MCIx input to an edge detection block. Input selection is done with the IS[1:0] bits in the MPHST register.

An edge selection block is used to select one of three ways to trigger capture events: rising edge, falling edge or both rising and falling edge sensitive; set-up is done with the TES[1:0] bits (keeping in mind that TES[1:0] = 00 configuration is reserved for Position Sensor / Sensorless Modes).

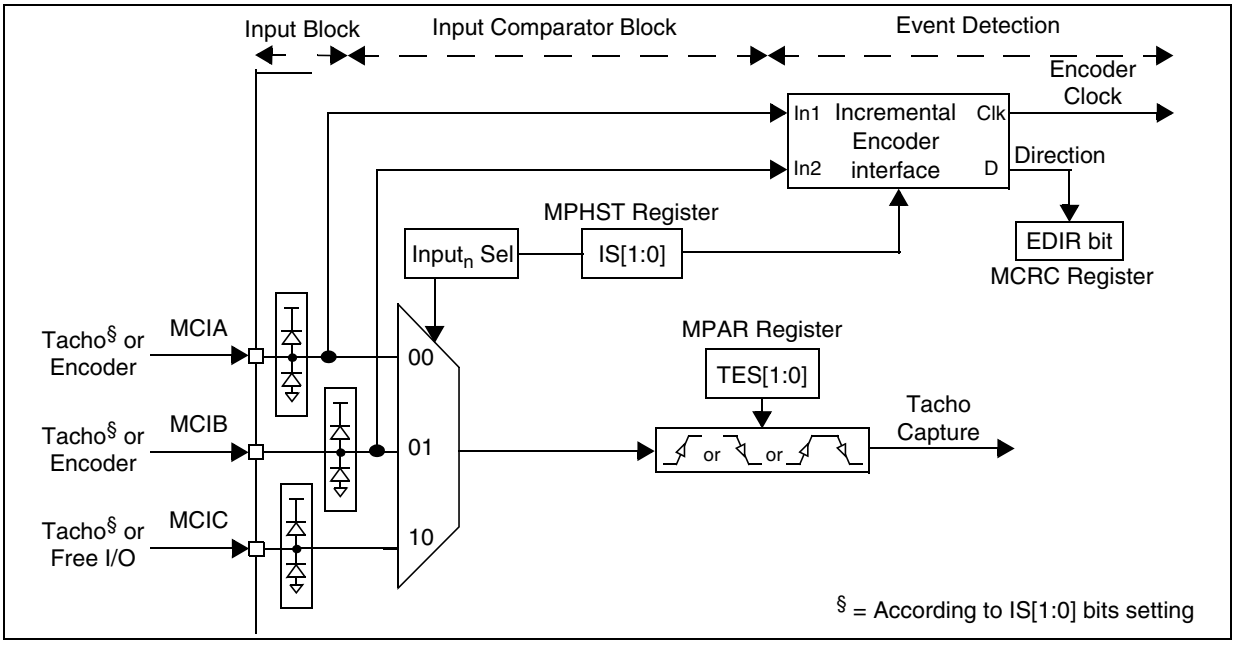
Having only one edge selected eliminates any incoming signal dissymmetry, which may due to pole-to-pole magnet dissymmetry or from a comparator threshold with low level signals.

Figure 89 presents the signals generated internally with different tacho input and TES bit settings.

Note on Hall Sensors: This configuration is also suitable for motors using 3 hall sensors for position detection and not driven in six-step mode (refer to "Speed Measurement Mode" on page 180).

Note on initializing the Input Stage: As the IS[1:0] bits in the MPHST register are preload bits (new values taken into account at C event), the initialization value of the IS[1:0] bits has to be entered in Direct Access mode. This is done by setting the DAC bit in the MCRA register during the speed sensor input initialization routine.

Figure 88. Input Stage in Speed Sensor Mode (TES[1:0] bits = 01, 10, 11)



MOTOR CONTROLLER (Cont'd)

10.6.6.13 Encoder Mode (IS[1:0] = 11)

Figure 90 shows the signals delivered by a standard digital incremental encoder and associated information:

- Two 90° phased square signals with variable frequency proportional to the speed; they must be connected to MCIA and MCIB input pins,
- Clock derived from incoming signal edges,
- Direction information determined by the relative phase shift of input signals (+ or -90°).

The Incremental Encoder Interface block aims at extracting these signals. As input logic is both rising and falling edge sensitive (independently from TES[1:0] bits setting), resulting clock frequency is four times the one of the input signals, thus increasing resolution for measurements.

It may be noticed that Direction bit (EDIR bit in MCRC register) is read only and that it does'nt affect counting direction of clocked timer (cf Section). As a result, one cannot extract position information from encoder inputs during speed reversal.

Figure 89. Tacho Capture events configured by the TES[1:0] bits

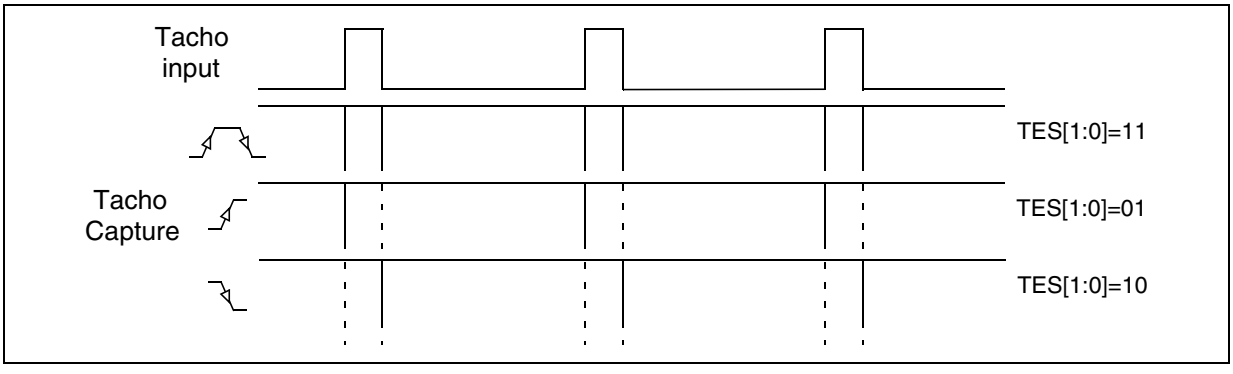
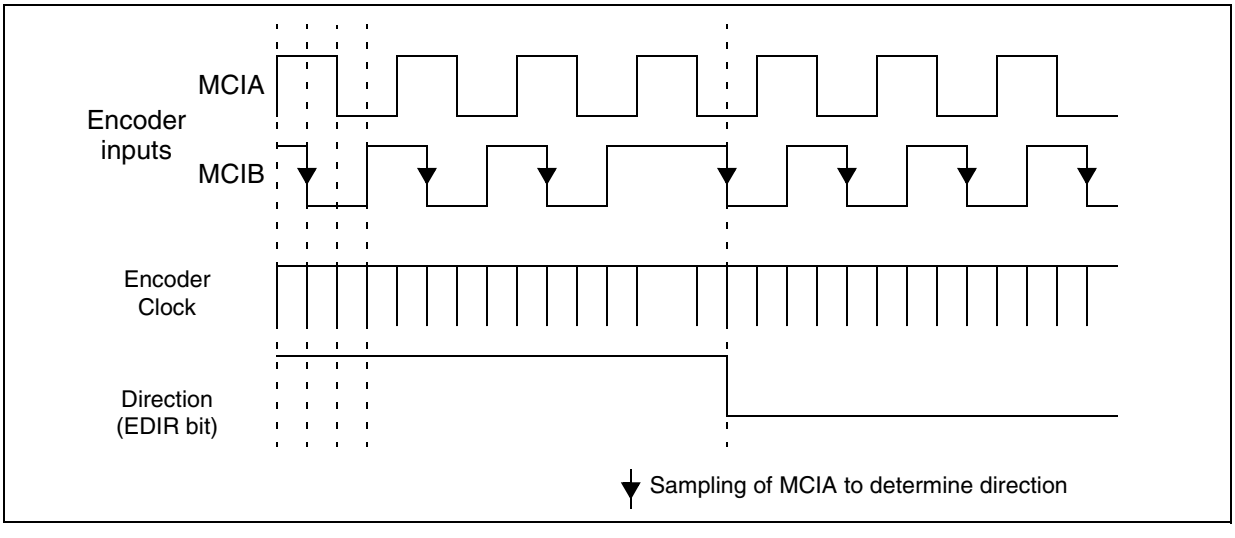


Figure 90. Incremental Encoder output signals and derived information



MOTOR CONTROLLER (Cont'd)

Note

If only one encoder output is available, it may be input either on MCIA or MCIB and an encoder clock signal will still be generated (in this case the frequency will be 50% less than with two inputs.

The state of EDIR bit will depend on signals present on MCIA and MCIB pins, the result will be given by the sampling of MCIA with MCIB falling edges.

10.6.6.14 Summary

Input Detection block set-up for the different available modes is summarized in the Table 35.

Table 35. Input Detection Block set-up

Input Detection Block Mode	Sensor Type	Edge sensitivity	SR bit	TES[1:0] bits (Tacho Edge Selection)	IS[1:0] bits (Input Selection)
Position Sensor	Hall, Optical,...	Both rising and falling edges	1	00	00 01 10
Sensorless	N/A	N/A	0	00	00 01 10
Speed Sensor	Incremental Encoder	Both rising and falling edges (imposed)	x	Any configuration different from 00: 01 10 11	11
	Tachogenerator, Hall, Optical...	Rising edge		01	00 01 10
		Falling edge		10	00 01 10
		Both rising and falling edges		11	00 01 10

MOTOR CONTROLLER (Cont'd)

Note on using the 3 MCIx pins as standard I/Os: When none of the MCIx pins are needed in the application (for instance when driving an induction motor in open loop), they can be used as standard I/O ports, by configuring the Motor Controller as follows: PCN=1, TES≠0 and IS=11. This disables the MCIx alternate functions and switches off the phase comparator. The state of the MCIx pins is summarized in Table 36.

Table 36. MCIx pin configuration summary

PCN	TES	SR	IS[1:0]	MCIA	MCIB	MCIC	Input Detection Block Mode	Comments
0	00	0	00	Analog Input	Hi-Z or GND	Hi-Z or GND	Sensorless	All MCIx pins are reserved for the MTC peripheral
			01	Hi-Z or GND	Analog Input	Hi-Z or GND		
			10	Hi-Z or GND	Hi-Z or GND	Analog input		
			11	NA	NA	NA	NA	
		1	00	Analog Input	Standard I/O	Standard I/O	Position Sensor	From 1 to 3 MCIx pins reserved depending on sensor
			01	Standard I/O	Analog Input	Standard I/O		
			10	Standard I/O	Standard I/O	Analog Input		
			11	Standard I/O	Standard I/O	Standard I/O	NA	All MCIx pins are standard I/Os. Phase comparator is OFF
	≠0	x	xx	NA	NA	NA	NA	
1	00	x	00	Analog Input	Standard I/O	Standard I/O	NA	Phase comparator is ON. The IS[1:0] bits must not be modified to avoid spurious event detection in Motor Controller
			01	Standard I/O	Analog Input	Standard I/O		
			10	Standard I/O	Standard I/O	Analog Input		
			11	Standard I/O	Standard I/O	Standard I/O	NA	All MCIx pins are standard I/Os. Recommended configuration: phase comparator OFF
	≠00	x	00	Digital Input	Standard I/O	Standard I/O	Speed Sensor Tachogenerator	Phase comparator is OFF
			01	Standard I/O	Digital Input	Standard I/O		
			10	Standard I/O	Standard I/O	Digital Input		
			11	Digital Input	Digital Input	Standard I/O	Speed Sensor Encoder	

*When PCN=0, TES=0 SR=0, inputs in OFF-state are put in HiZ or grounded depending on the value of the DISS bit in the MSCR register.

Notes:

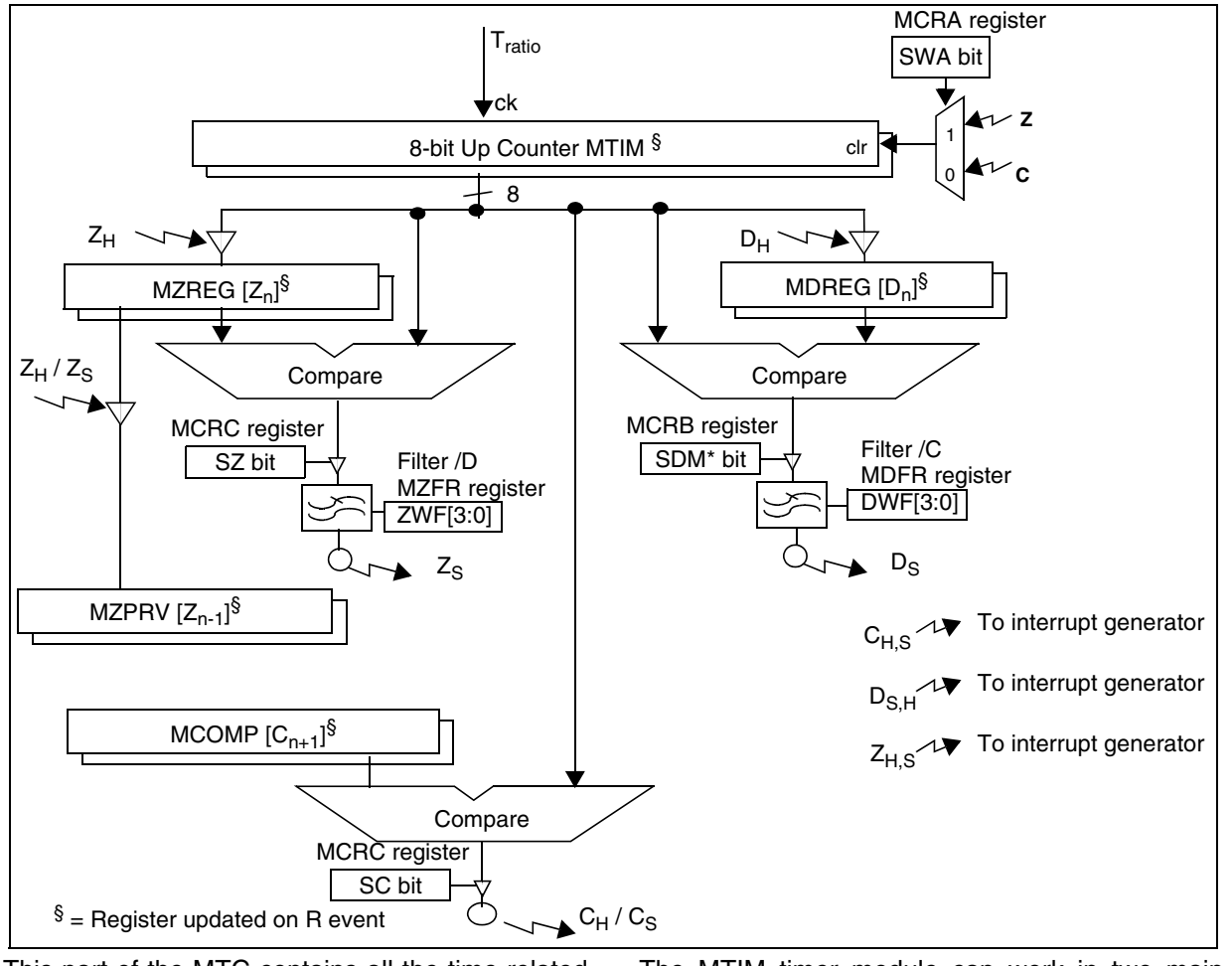
1. Analog input: Based on analog comparator and analog voltage reference. The corresponding digital I/O is disabled and data in the DR register is not representative of data on the input.

2. Digital input: Use of standard V_{IL}, V_{IH} I/O level. This input can also be read via the associated I/O port.

MOTOR CONTROLLER (Cont'd)

10.6.7 Delay Manager

Figure 91. Overview of MTIM Timer in Switched and Autoswitched Mode



This part of the MTC contains all the time-related functions, its architecture is based on an 8-bit shift left/shift right timer shown in Figure 91. The MTIM timer includes:

– An auto-updated prescaler

– A capture/compare register for simulated demagnetization simulation (MDREG)

– Two cascaded capture and one compare registers (MZREG and MZPRV) for storing the times between two consecutive BEMF zero crossings (Z_H events) and for zero-crossing event simulation (Z_S)

– An 8x8 bit multiplier for auto computing the next commutation time

– One compare register for phase commutation generation (MCOMP)

The MTIM timer module can work in two main modes when driving synchronous motors in six-steps mode.

In switched mode the user must process the step duration and commutation time by software.

In autoswitched mode the commutation action is performed automatically depending on the rotor position information and register contents. This is called the hardware commutation event C_H. When enabled by the SC bit in the MCRC register, commutation can also be simulated by writing a value directly in the MCOMP register that is compared with the MTIM value. This is called simulated commutation C_S (See "Built-in Checks and Controls for simulated events" on page 175.).

Both in switched mode and autoswitched mode , if the SC bit in the MCRC register is set (software commutation enabled), no comparison between

MOTOR CONTROLLER (Cont'd)

the MCOMP and MTIM register is enabled before a write access in the MCOMP register. This means that if the SC bit is set and no write access is done after in the MCOMP register, no C_S commutation event will occur.

In Speed Measurement mode, when using encoder or tachogenerator speed sensors (i.e. both TES[1:0] bits in the MPAR register are not reset

and the input detection block is set-up to process sensor signals), motor speed can be measured but it is not possible drive a motor in six-step mode, either sensored or sensorless.

Speed Measurement mode is useful for motors supplied with 3-phase sinewave-modulated PWM signals:

– AC induction motors,
– Permanent Magnet AC (PMAC) motors (although it needs three position sensors, they can be handled just like tachogenerator signals).

This mode uses only part of the Delay Manager's resources. For more details refer to "Speed Measurement Mode" on page 180.

Table 37. Switched and Autoswitched modes

SWA bit	Commutation Type	MCOMP User access
0	Switched mode	Read/Write
1	Autoswitched mode	Read/Write

10.6.7.1 Switched Mode

This feature allows the motor to be run step-by-step. This is useful when the rotor speed is still too low to generate a BEMF. It can also run other kinds of motor without BEMF generation such as induction motors or switch reluctance motors. This mode can also be used for autoswitching with all computation for the next commutation time done by software (hardware multiplier not used) and using the powerful interrupt set of the peripheral.

In this mode, the step time is directly written by software in the commutation compare register MCOMP. When the MTIM timer reaches this value a commutation occurs (C event) and the MTIM timer is reset.

At this time all registers with a preload function are loaded (registers marked with (*) in Section 10.6.13). The CI bit of MISR is set and if the CIM bit in the MIMR register is set an interrupt is generated.

The MTIM timer prescaler (Step ratio bits ST[3:0] in the MPRSR register) is user programmable. Access to this register is not allowed while the MTIM timer is running (access is possible only before the starting the timer by means of the CKE bit) but the prescaler contents can be incremented/decremented at the next commutation event by setting the RMI (decrement) or RPI (increment) bits in the MISR register. When this method is used, at the next commutation event the prescaler value will be updated but also all the MTIM timer-related registers will be shifted in the appropriate direction to keep their value. After it has been taken into account, (at commutation) the RPI or RMI bit is reset by hardware. See Table 38.

Only one update per step is allowed, so if both RPI and RMI bits are set together by software, this does not affect the MISR register: the write access to these two bits together is not taken into account and the previous state is kept. This means that if either RPI or RMI bit was set before the write access of both bits at the same time, this bit (RPI or RMI) is kept at 1. If none of them was set before the simultaneous write access, none of them will be set after the write access.

In switched mode, BEMF and demagnetization detection are already possible in order to pass in autoswitched mode as soon as possible but Z and D events do not affect the timer contents.

In this mode, if an MTIM overflow occurs, it restarts counting from 0x00h and the OI overflow flag in the MCRC register is set if the TES[1:0] bits = 00.

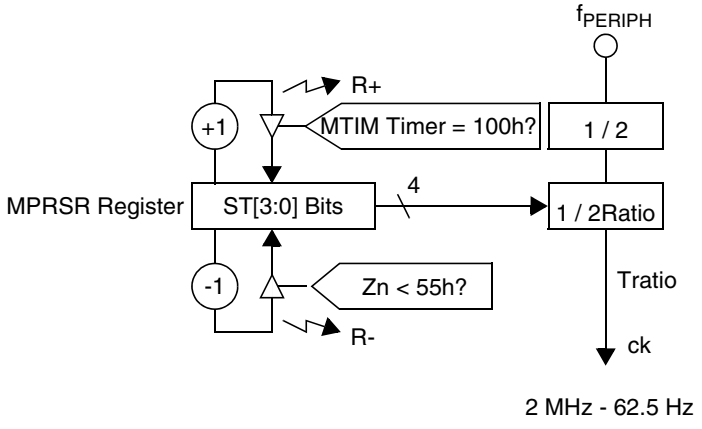
Caution: In this mode, MCOMP must never be written to 0.

Table 38. Step Update

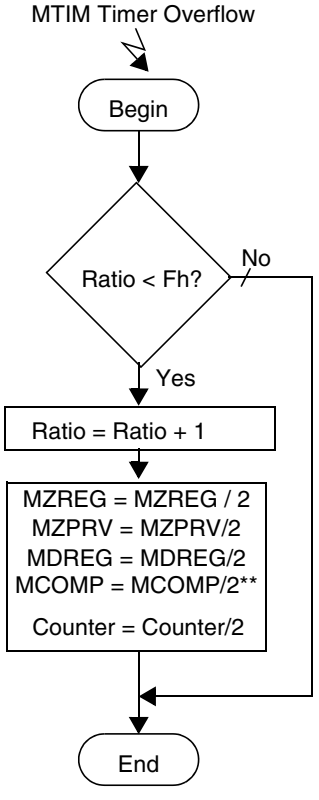
Mode	TES[1:0]	CKE bit	SWA bit	Clock State	Read	Ratio Increment (Slow Down)	Ratio Decrement (Speed-Up)
x	xx	0	x	Disabled	Always possible	Write the ST[3:0] value directly in the MPRSR register	
Switched	00	1	0	Enabled		Set RPI bit in the MISR register till next commutation	Set RMI bit in the MISR register till next commutation
Autoswitched	00	1	1	Enabled		Automatically updated according to MZREG value	
Speed measure	01 10 11	1	x	Enabled			

MOTOR CONTROLLER (Cont'd)

Figure 92. Step Ratio Functional Diagram

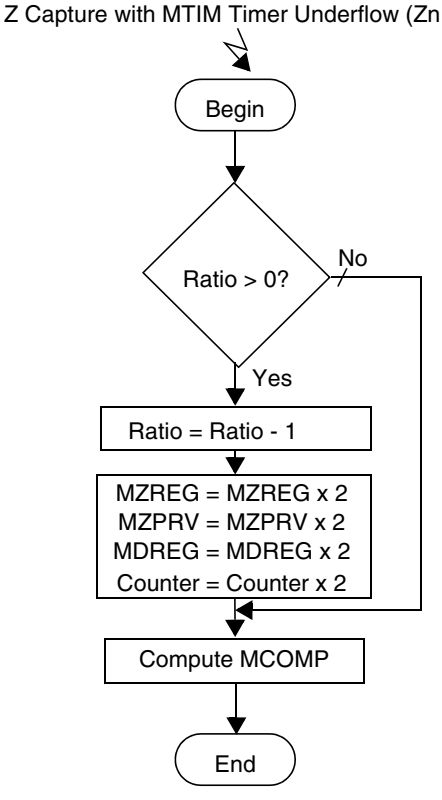


MTIM Timer control over T_{ratio} and register operation



Slow-down control

** Only in Auto-switched mode (SWA=1 in MCRA register)



Speed-up control

MOTOR CONTROLLER (Cont'd)

10.6.7.2 Autoswitched Mode

In this mode, using the hardware commutation event C_H (SC bit reset in MCRC register), the MCOMP register content is automatically computed in real-time as described below and in Figure 93.

The C (either C_S or C_H) event has no effect on the contents of the MTIM timer.

When a Z_H event occurs the MTIM timer value is captured in the MZREG register, the previous captured value is shifted into the MZPRV register and the MTIM timer is reset. See Figure 73.

When a Z_S event occurs, the value written in the MZREG register is shifted into the MZPRV register and the MTIM timer is reset.

One of these two registers, (when the SC bit = 0 in the MCRC register and depending on the DCB bit in the MCRA register), is multiplied with the contents of the MWGHT register and divided by 256. The result is loaded in the MCOMP compare register, which automatically triggers the next hardware commutation (C_H event).

Note: The result of the 8*8 bit multiplication, once written in the MCOMP register is compared with the current MTIM value to check that the MCOMP value is not already less than the MTIM value due to the multiplication time. If MCOMP<=MTIM, a C_H event is generated immediately and the MCOMP value is overwritten by the MTIM value.

Table 39. Multiplier Result

DCB bit	Commutation Delay
0	MCOMP = MWGHT x MZPRV / 256
1	MCOMP = MWGHT x MZREG / 256

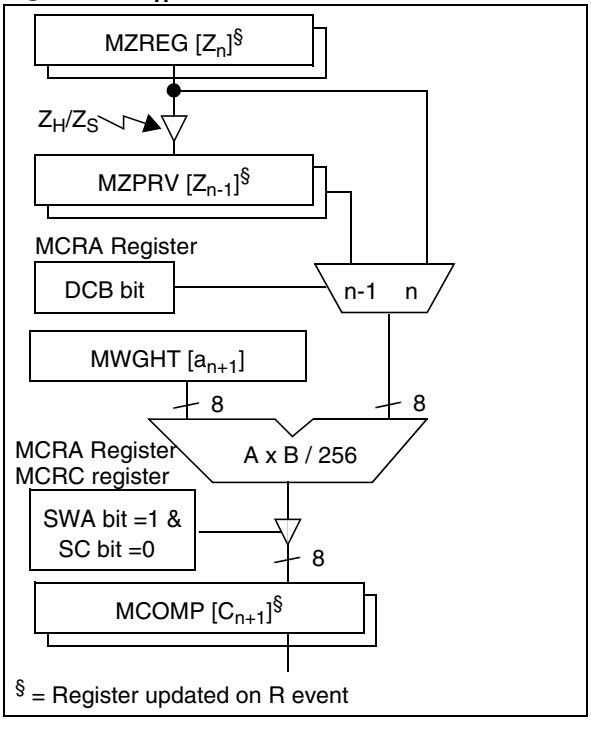
After each shift operation the multiply is recomputed for greater precision.

Using either the MZREG or MZPRV register depends on the motor symmetry and type.

The MWGHT register gives directly the phase shift between the motor driven voltage and the BEMF. This parameter generally depends on the motor and on the speed.

Setting the SC bit in the MCRC register enables the simulated commutation event (C_S) generation. This means that a write access is possible to the MCOMP register and the MTIM value will be compared directly with the value written by software in the MCOMP register to generate the C_S event. The comparison is enabled as soon as a write access is done to the MCOMP register. This means that if the SC bit is set and no write access is done to the MCOMP register, the C event will never occur because no comparison will be done between MCOMP and MTIM. Therefore, it is recommended in autoswitched mode, when using software commutation feature (SC bit is set) and for a normal event sequence, the corresponding value to be put in MCOMP has to be written during the Z interrupt routine (because MTIM has just been reset), so that there is no spurious comparison. If the SC bit is set during a Z event interrupt, then , the result of the 8*8 bits hardware multiplication can be overwritten by software in the MCOMP register. When simulated commutation mode is enabled, the event sequence is no longer respected, meaning that the peripheral will accept consecutive commutation events and not necessarily wait for a D event after a C_s event. In this case the MCOMP register can be written immediately after the previous C event, in the C interrupt service routine for example.

Figure 93. C_H Processor Block



Note 1: An overflow of the MTIM timer generates an RPI interrupt if the RIM bit is set.

Note 2: When simulated commutation mode is enabled, the D and Z event are not ignored by the peripheral, this means that if a Z event happens, the MTIM 8 bit internal counter will be reset.

Note 3: To generate consecutive simulated commutations (C_S), the successive value has to be written in the MCOMP register only after a C event

MOTOR CONTROLLER (Cont'd)

generation. Otherwise, the C event will never occur.

Note 4: When simulated commutation mode is enabled, the built-in check is active, so if the value written in the MCOMP register is less than or equal to MTIM, the C event is generated and the data in the MCOMP register are overwritten by the MTIM value.

Auto-updated Step Ratio Register:

a) **In switched mode:** the MTIM timer is driven by software only and any prescaler change has to be done by software (see page 169 for more details).

b) **In autoswitched mode:** an auto-updated prescaler always configures the MTIM timer for best accuracy. Figure 92 shows the process of updating the Step Ratio bits:

– When the MTIM timer value reaches 100h, the prescaler is automatically incremented in order to slow down the MTIM timer and avoid an overflow. To keep consistent values, the MTIM register and all the relevant registers are shifted right (divided by two). The RPI bit in the MISR register is set and an interrupt is generated (if RIM is set). The timer restarts counting from its median value 0x80h and if the TES[1:0] bits = 00, the OI bit in the MCRC register is set.

– When a Z-event occurs, if the MTIM timer value is below 55h, the prescaler is automatically decremented in order to speed up the MTIM timer and keep precision better than 1.2%. The MTIM register and all the relevant registers are shifted left (multiplied by two). The RMI bit in the MISR register is set and an interrupt is generated if RIM is set.

– If the prescaler contents reach the value 0, it can no longer be automatically decremented, the MTC continues working with the same prescaler value, i.e. with a lower accuracy. No RMI interrrupt can be generated.

– If the prescaler contents reach the value 15, it can no longer be automatically incremented. When the timer reaches the value FFh, the prescaler and all the relevant registers remain unchanged and no interrupt is generated, the timer restarts counting from 0x00h and if the TES[1:0] bits = 00, the OI bit in the MCRC register is set at each overflow (it has to be reset by software). The RPI bit is no longer set. The PWM is still generated and the D and Z detection circuitry still work, enabling the capture of the maximum timer value.

The automatically updated registers are: MTIM, MZREG, MZPRV, MCOMP and MDREG. Access to these registers is summarized in Table 41.

10.6.7.3 Debug Option

In both Switched Mode and Autoswitched Mode, setting the bit DG in MPWME register enables the Debug Option. This option consists of outputting the C, D and Z signals in real time on pins MCZEM and MCDEM. This is very useful during the debug phase of the application. Figure 94 shows the signals output on pins MCDEM and MCZEM with the debug option.

Note 1: When the delay coefficient equals 0/256 (C event immediately after Z event), a glitch appears on MCZEM pin to be able to see the event even in this case.

This option is also available in Speed measurement mode with different signal outputs (see Figure 94):

– MCDEM toggles when a capture event is generated,

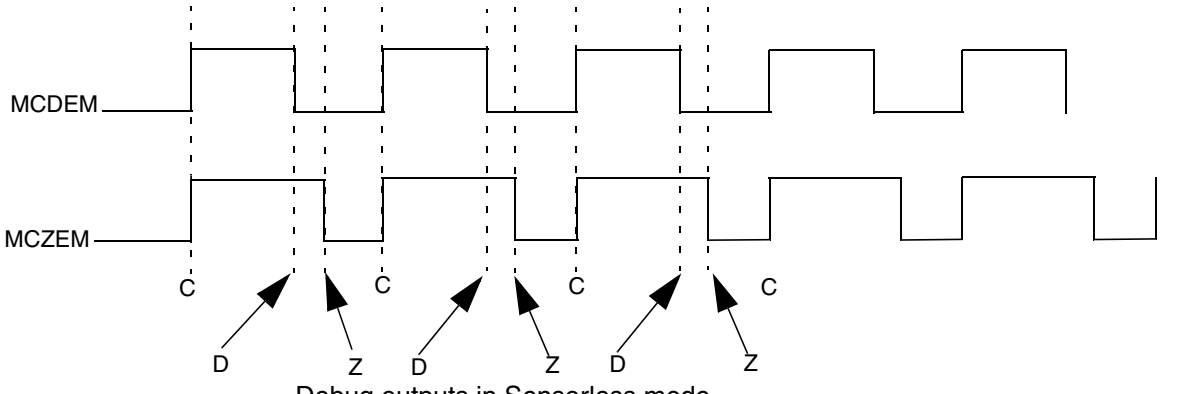
– MCZEM toggles every time a U event is generated.

These signals are only available if the TES[1:0] bits = 10, 01 or 11.

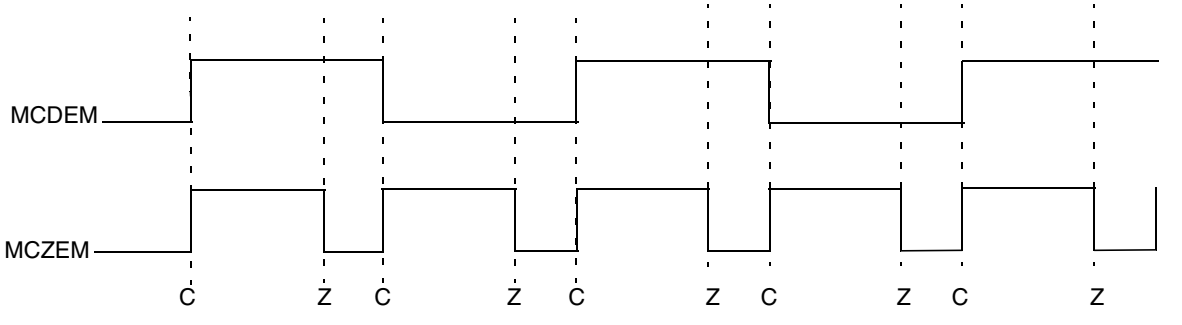
Note 2: In sensor mode, the MCDEM output pin toggles at each C event. The MCZEM pin outputs the Z event.

MOTOR CONTROLLER (Cont'd)

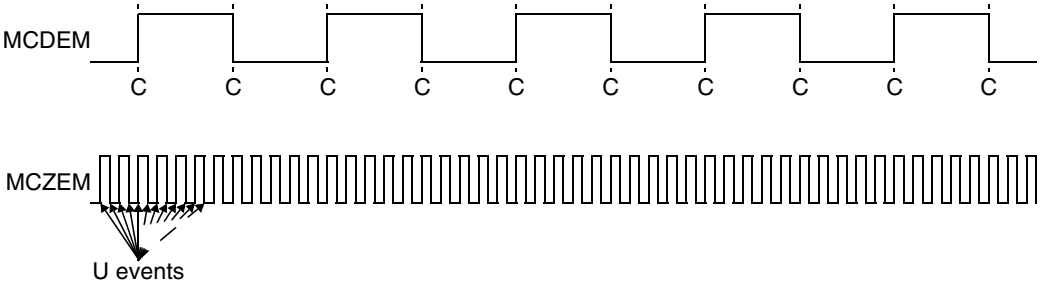
Figure 94. Output on pins MCDEM and MCZEM with debug option (DG bit=1)



Debug outputs in Sensorless mode



Debug outputs in Sensor mode



Debug outputs in Speed Measurement mode (TES[1:0] bits equal to 01, 10 or 11).

MOTOR CONTROLLER (Cont'd)

Note on using the auto-updated MTIM timer: The auto-updated MTIM timer works accurately within its operating range but some care has to be taken when processing timer-dependent data such as the step duration for regulation or demagnetization.

For example if an overflow occurs when calculating a simulated end of demagnetization (MCOMP+demagnetisation_time>FFh), the value that is stored in MDREG will be:
80h+(MCOMP+demagnetization_time-FFh)/2.

Note on commutation interrupts: It is good practice to modify the configuration for the next step as soon as possible, i.e within the commutation interrupt routine.

All registers that need to be changed at each step have a preload register that enables the modifications for a complete new configuration to be performed at the same time (at C event in normal mode or when writing the MPHST register in direct access mode).

These configuration bits are:

CPB, HDM, SDM and OS2 in the MCRB register and IS[1:0], OO[5:0] in the MPHST register.

Note on initializing the MTC: As shown in Table 41 all the MTIM timer registers are in read-write mode until the MTC clock is enabled (with the CKE bit). This allows the timer, prescaler and compare registers to be properly initialized for start-up.

In sensorless mode, the motor has to be started in switched mode until a BEMF voltage is present on the inputs. This means the prescaler ST[3:0] bits and MCOMP register have to be modified by software. When running the ST[3:0] bits can only be incremented / decremented, so the initial value is very important.

When starting directly in autoswitched mode (in sensor mode for example), write an appropriate value in the MZREG and MZPRV register to perform a step calculation as soon as the clock is enabled.

MOTOR CONTROLLER (Cont'd)

10.6.7.4 Built-in Checks and Controls for simulated events

As described in Figure 91. on page 168, MZREG, MDREG and MCOMP registers are capture/compare registers. The Compare registers are write accessible and can be used to generate simulated events. The value of the MTIM timer is compared with the value written in the registers and when the MTIM value reaches the corresponding register value, the simulated event is generated. Simulated event generation is enabled when the corresponding bits are set:

– In the MCRB register for simulated demagnetisation
  – SDM for simulated demagnetisation
– In the MCRC register for simulated zero-crossing and commutation.
  – SC for simulated commutation
  – SZ for simulated zero-crossing event.

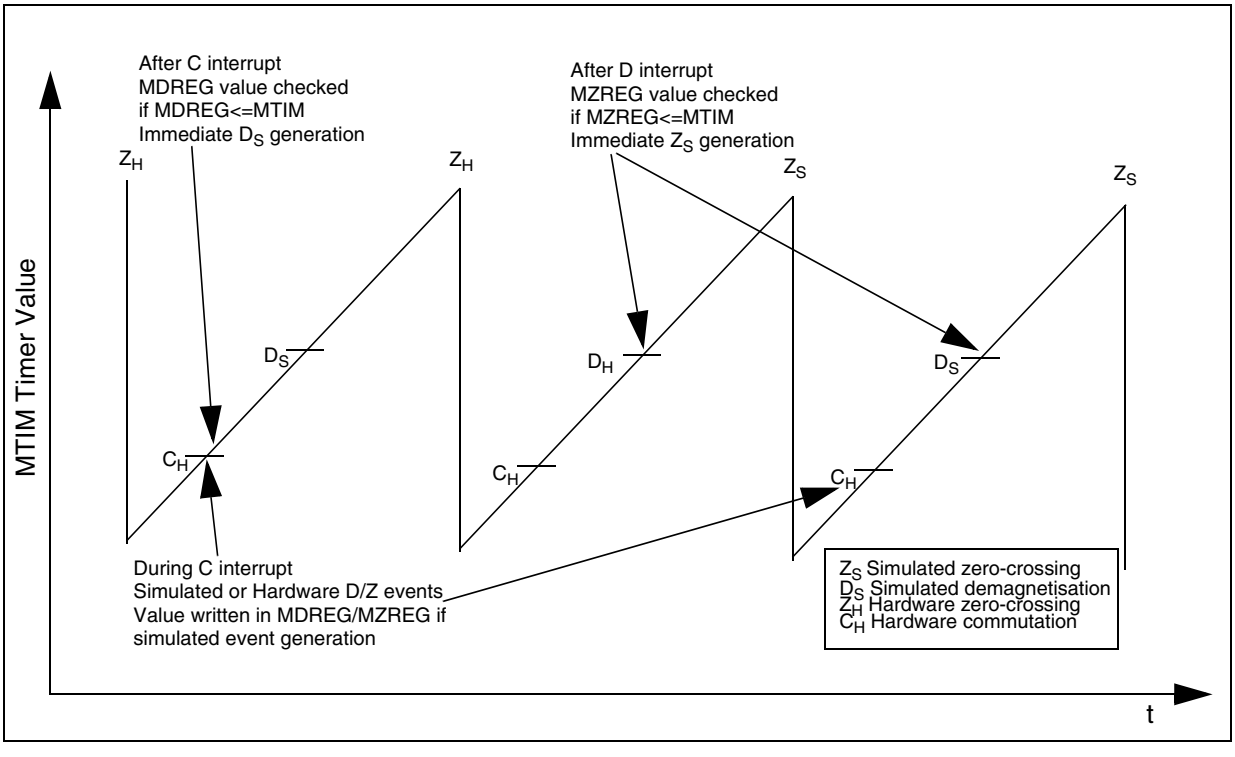
To avoid a system stop, special attention is needed when writing in the register to generate the corresponding simulated event. The value written in the register has to be greater than the current value of the MTIM timer when writing in the registers. If the value written in the registers (MDREG, MZREG or MCOMP) is already less than the current value of MTIM, the simulated event will never be generated and the system will be stopped.

For this reason, built-in checks and controls have been implemented in the MTIM timer.

If the value written in one of those registers in simulated event generation mode is less than or equal to the current value of the timer when it is compared, the simulated event is generated immediately and the value of the MTIM timer at the time the simulated event occurs overwrites the value in the registers. Like that the value in the register really corresponds to the simulated event generation and can be re-used to generate the next simulated event.

So, the value written in the registers able to generate simulated events is checked by hardware and compare to the current MTIM value to verify that it is greater.

Figure 95. Simulated demagnetisation / zero-crossing event generation (SC=0)



MOTOR CONTROLLER (Cont'd)

When using hardware commutation C_H, the sequence of events needed is C_H then D and finally Z events and the value written in the registers are checked at different times.

If SDM bit is set, meaning simulated demagnetisation, a value must be written in the MDREG register to generate the simulated demagnetisation. This value must be written after the C (either C_s or C_H) event preceding the simulated demagnetisation.

If SZ bit is set, meaning simulated zero-crossing event, a value must be written in the MZREG register to generate the simulated zero-crossing. This value must be written after the D event (D_H or D_S) preceding the simulated zero-crossing.

When using simulated commutation (C_S), the result of the 8*8 hardware multiplication of the delay manager is not taken in account and must be overwritten if the SC bit has been set in a Z event interrupt and the sequence of events is broken meaning that several consecutive simulated commutations can be implemented.

As soon as the SC bit is set in the MCRC register, the system won't necessarily expect a D event after a C event. This can be used for an application in sensor mode with only one Hall Effect sensor for example.

Be careful that the D and Z events are not ignored by the peripheral, this means that for example if a Z event occurs, the MTIM timer is reset. In Simulated Commutation mode, the sequence D -> Z is expected, and this order must be repected.

As the sequence of events may not be the same when using simulated commutation, as soon as the SC bit is set, the capture/compare feature and protection on MCOMP register is reestablished only after a write to the MCOMP register. This means that as soon as the SC bit is set, if no write access is done to the MCOMP register, no commutation event will be generated, whatever the value of MCOMP compared to MTIM at the time SC is set. This does not depend on the running mode: switched or autoswitched mode (SWA bit). If software commutation event is used with a normal sequence of events C-->D-->Z, it is recommended to write the MCOMP register during the Z interrupt routine to avoid any spurious comparison as several consecutive C_s events can be generated.

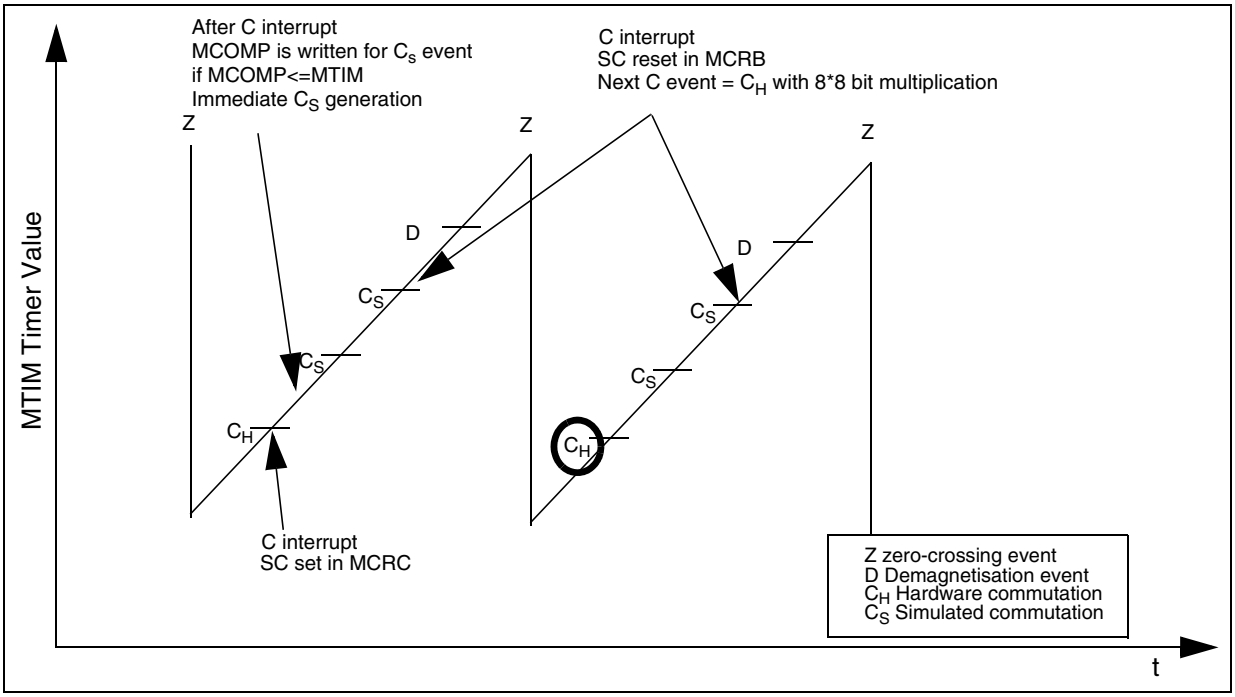
Note that two different simulated events can be used in the same step (like D_S followed by Z_S).

Note also that for more precision, it is recommended to use the value captured from the preceding hardware event to compute the value used to generate simulated events.

Figure 95, Figure 96 and Figure 97 shows details of simulated event generation.

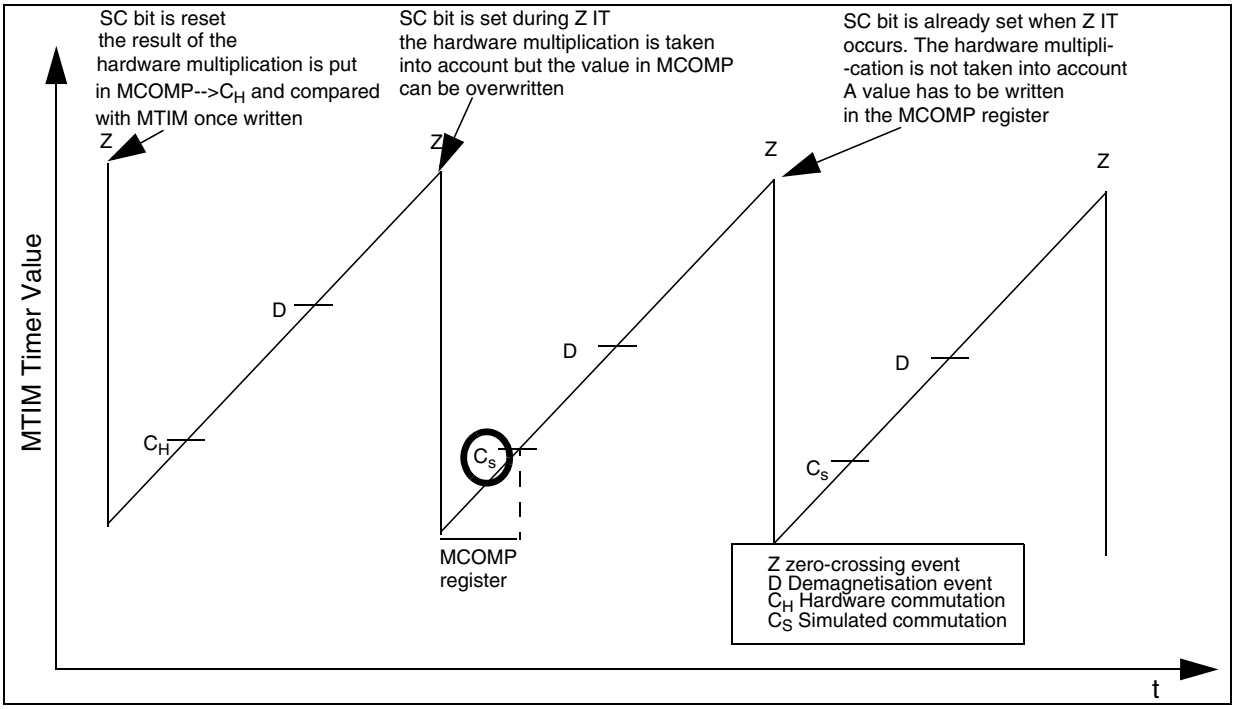
MOTOR CONTROLLER (Cont'd)

Figure 96. Simulated commutation event generation with only 1 Hall effect sensor (SC bit =1)



Note: If the SC bit is set during Z event interrupt, then the 8*8 bit hardware multiplication result must be overwritten in the MCOMP register. Otherwise, when the SC bit is set, the result of the multiplication is not taken into account after a Z event.

Figure 97. Simulated commutation and Z event



MOTOR CONTROLLER (Cont'd)

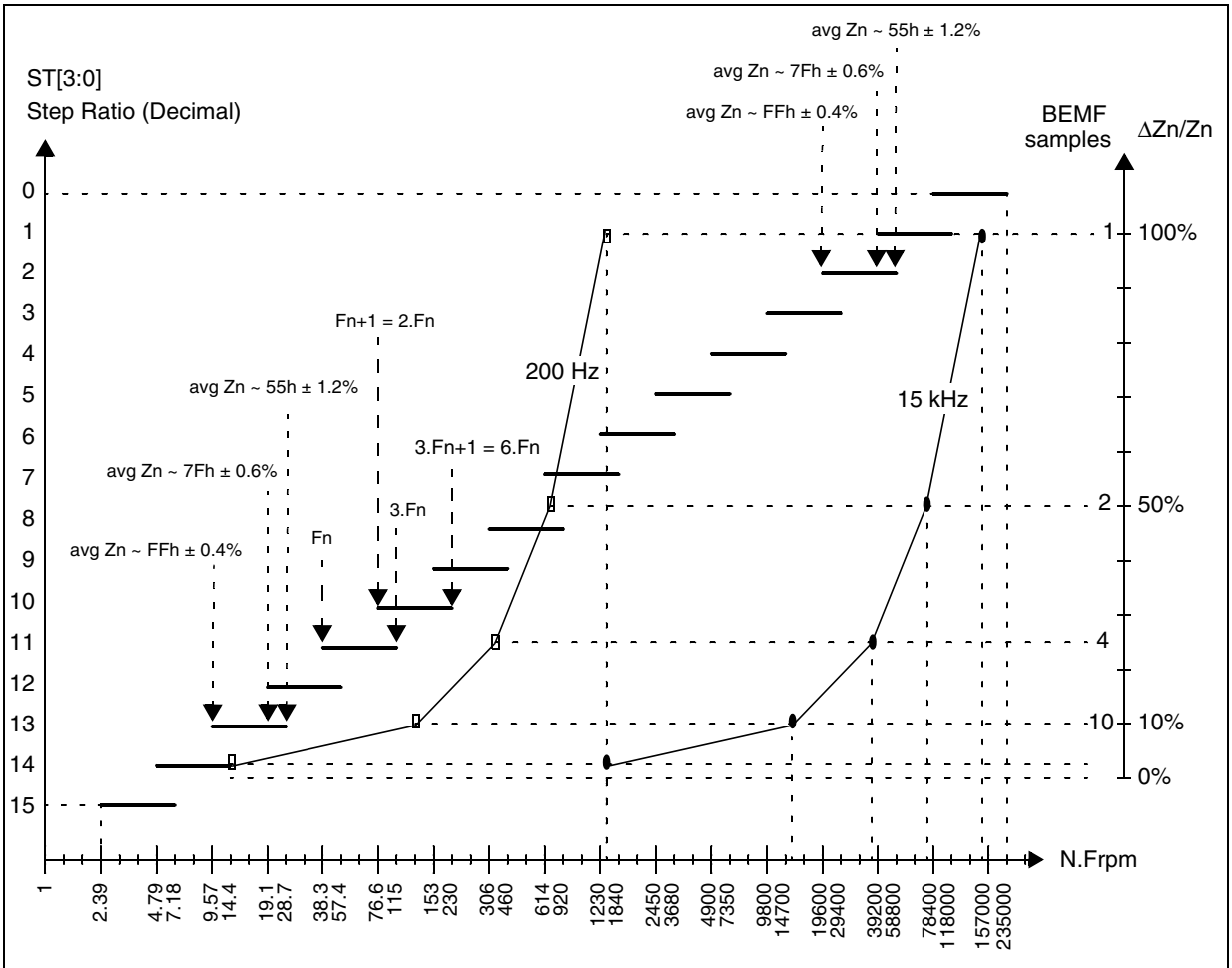
The Figure 98 gives the step ratio register value (left axis) and the number of BEMF sampling during one electrical step with the corresponding accuracy on the measure (right axis) as a function of the mechanical frequency.

For a given prescaler value (step ratio register) the mechanical frequency can vary between two fixed values shown on the graph as the segment ends. In autoswitched mode, this register is automatically incremented/decremented when the step frequency goes out of this segment.

At f_{MTC}=4MHz, the range covered by the Step Ratio mechanism goes from 2.39 to 235000 (pole pair x rpm) with a minimum accuracy of 1.2% on the step period.

To read the number of samples for Zn within one step (right Y axis), select the mechanical frequency on the X axis and the sampling frequency curve used for BEMF detection (PWM frequency or measurement window frequency). For example, for N.Frpm = 15,000 and a sampling frequency of 15kHz, there are approximately 10 samples in one step and there is a 10% error rate on the measurement.

Figure 98. Step Ratio Bits decoding and accuracy results and BEMF Sampling Rate



$$F_{step} = 6.N.F_{rpm} = N.F / 10 \Leftrightarrow N.F = 10.F_{step}$$

F_{step}: Electrical step frequency

N: Pole pair number

MOTOR CONTROLLER (Cont'd)

Table 40. Step Frequency/Period Range (4MHz)

Step Ratio Bits ST[3:0] in MPRSR Register	Maximum Step Frequency	Minimum Step Frequency	Minimum Step Period	Maximum Step Period
0000	23.5 kHz	7.85 kHz	42.5 µs	127.5 µs
0001	11.7 kHz	3.93 kHz	85 µs	255 µs
0010	5.88 kHz	1.96 kHz	170 µs	510 µs
0011	2.94 kHz	980 Hz	340 µs	1.02 ms
0100	1.47 kHz	490 Hz	680 µs	2.04 ms
0101	735 Hz	245 Hz	1.36 ms	4.08 ms
0110	367 Hz	123 Hz	2.72 ms	8.16 ms
0111	183 Hz	61.3 Hz	5.44 ms	16.32 ms
1000	91.9 Hz	30.7 Hz	10.9 ms	32.6 ms
1001	45.9 Hz	15.4 Hz	21.8 ms	65.2 ms
1010	22.9 Hz	7.66 Hz	43.6 ms	130 ms
1011	11.4 Hz	3.83 Hz	87 ms	261 ms
1100	5.74 Hz	1.92 Hz	174 ms	522 ms
1101	2.87 Hz	0.958 Hz	349 ms	1.04 s
1110	1.43 Hz	0.479 Hz	697 ms	2.08 s
1111	0.718 Hz	0.240 Hz	1.40 s	4.17 s

Table 41. Modes of Accessing MTIM Timer-Related Registers

State of MCRA / MCRB / MPAR Register Bits					Access to MTIM Timer Related Registers	
RST bit	TES[1:0]	SWA bit	CKE bit	Mode	Read Only Access	Read / Write Access
0	xx	x	0	Configuration Mode		MTIM, MTIML, MZPRV, MZREG, MCOMP, MDREG, ST[3:0]
0	00	0	1	Switched Mode	MTIM, ST[3:0]	MCOMP, MDREG, MZREG, MZPRV RMI bit of MISR: 0: No action 1: Decrement ST[3:0] RPI bit of MISR: 0: No action 1: Increment ST[3:0]
0	00	1	1	Autoswitched Mode	MTIM, ST[3:0]	MDREG, MCOMP, MZREG, MZPRV, RMI, RPI bit of MISR: Set by hardware, (increment ST[3:0]) Cleared by software
0	01 10 11	x	1	Speed Sensor Mode	MTIM, MTIML, ST[3:0]	MDREG,MZREG, MZPRV, RMI, RPI bit of MISR, : Set by hardware, (increment or decrement ST[3:0]), cleared by software.

MOTOR CONTROLLER (Cont'd)

10.6.7.5 Speed Measurement Mode

Motor speed can be measured using two methods depending on sensor type: period measurement or pulse counting. Typical sensor handling is described here.

Incremental encoders allows accurate speed measurement by providing a large number of pulses per revolution (ppr) with ppr rates up to several thousands; the higher the ppr rate, the higher the resolution. The proposed method consists of counting the number of clock cycles issued by the Incremental Encoder Interface (Encoder Clock) during a fixed time window (refer to Figure 100).

The tachogenerator has a much lower ppr rate than the encoder (typically factor 10). In this context, it is more meaningful to measure the period between Tacho Captures (i.e. relevant transitions of the incoming signals). Accuracy is imposed by the reference clock, i.e. the CPU clock (refer to Figure 99).

Figure 99. Tachogenerator period acquisition using MTIM timer

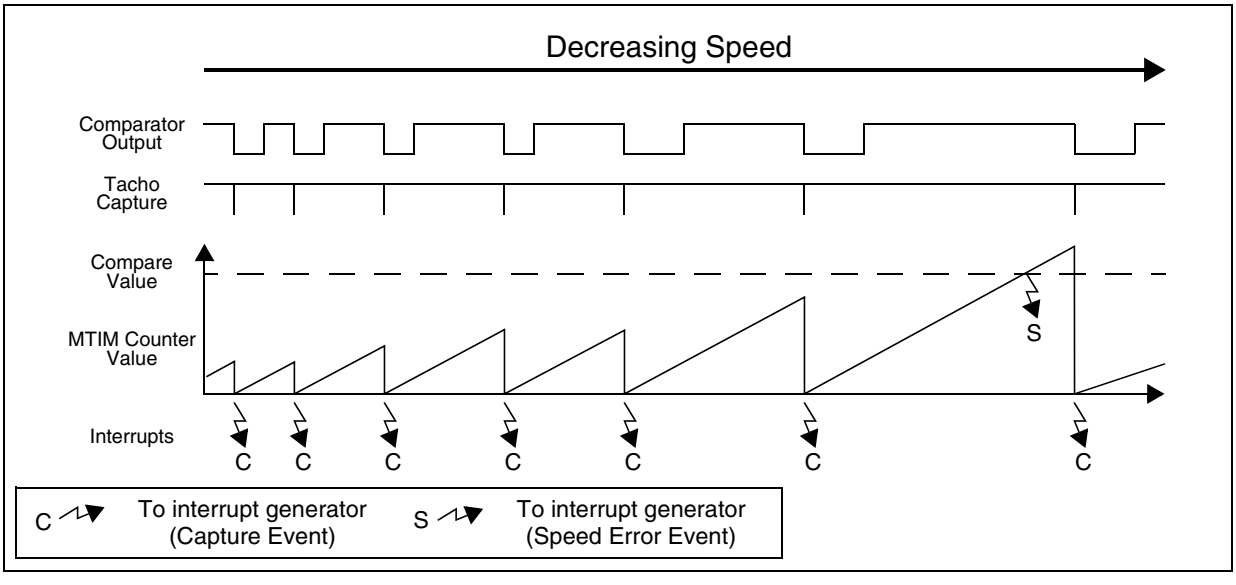
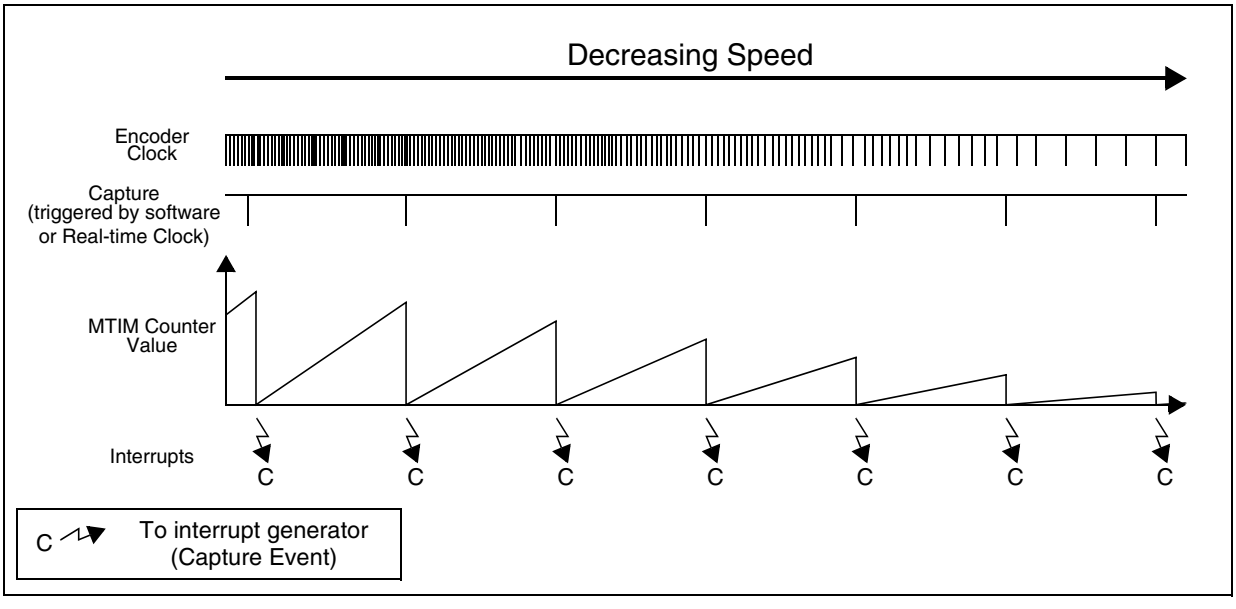


Figure 100. Encoder Clock frequency measure using MTIM timer



MOTOR CONTROLLER (Cont'd)

Hall sensors (or equivalent sensors providing position information) are widely used for motor control. There are two cases to be considered:

– BLDC motor or six-step synchronous motor drive; "Sensor Mode" is recommended in this case, as most tasks are performed by hardware in the Delay Manager

– BLAC, asynchronous or motors supplied with 3-phase sinewave-modulated PWM signals in general; in this case "Speed Sensor Mode" allows high accuracy speed measurement (the Sensor Mode of the Delay Manager being unsuitable for sinewave generation). Position information is handled by software to lock the statoric field to the rotoric one for driving synchronous motors.

Hall sensors are usually arranged in a 120° configuration. In that case they provide 3 ppr with both rising and falling edge triggering; the tachogenerator measurement method can therefore be applied. The main difference lies in the fact that one must use the position information they provide. This can be done using the three MCIx pins and the analog multiplexer to know which of the 3 sensors toggled; an interrupt is generated just after the expected transition (refer to Figure 101).

As described in Figure 102, the MTIM Timer is reconfigured depending on the selected sensor. This means that most of Delay Manager registers are used for a different purpose, with modified functionalities.

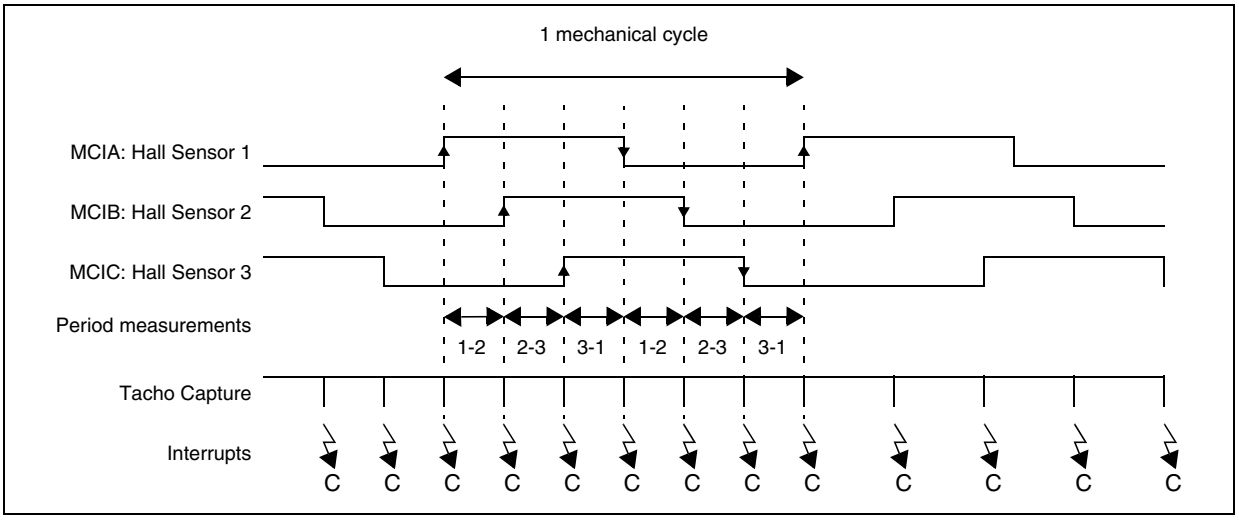
For greater precision, the MTIM Up-counter is extended to 16 bits using MTIM and an additional MTIML register. On a capture event, the current counter value is captured and the counter [MTIM:MTIML] is cleared. The counting direction is not affected by the EDIR bit when using an encoder sensor.

A 16-bit capture register is used to store the captured value of the extended MTIM counter: the speed result will be either a period in clock cycles or a number of encoder pulses. This 16-bit register is mapped in the MZREG and MZPRV register addresses. To ensure that the read value is not corrupted between the high and low byte accesses, a read access to the MSB of this register (MZREG) locks the LSB (ie MZPRV content is locked) until it is read and any other capture event in between these two accesses is discarded.

A compare unit allows a maximum value to be entered for the tacho periods. If the 16-bit counter [MTIM:MTIML] exceeds this value, a Speed Error interrupt is generated. This may be used to warn the user that the tachogenerator signal is lost (wires disconnected, motor stalled,...). As 8-bit accuracy is sufficient for this purpose, only the MSByte of the counter (i.e. MTIM) is compared to 8-bit compare register, mapped in the MDREG register location. The LSByte is nevertheless compared with a fixed FFh value. Available values for comparison are therefore FFFFh, FEFFh, FDFFh, ..., 01FFh, 00FFh.

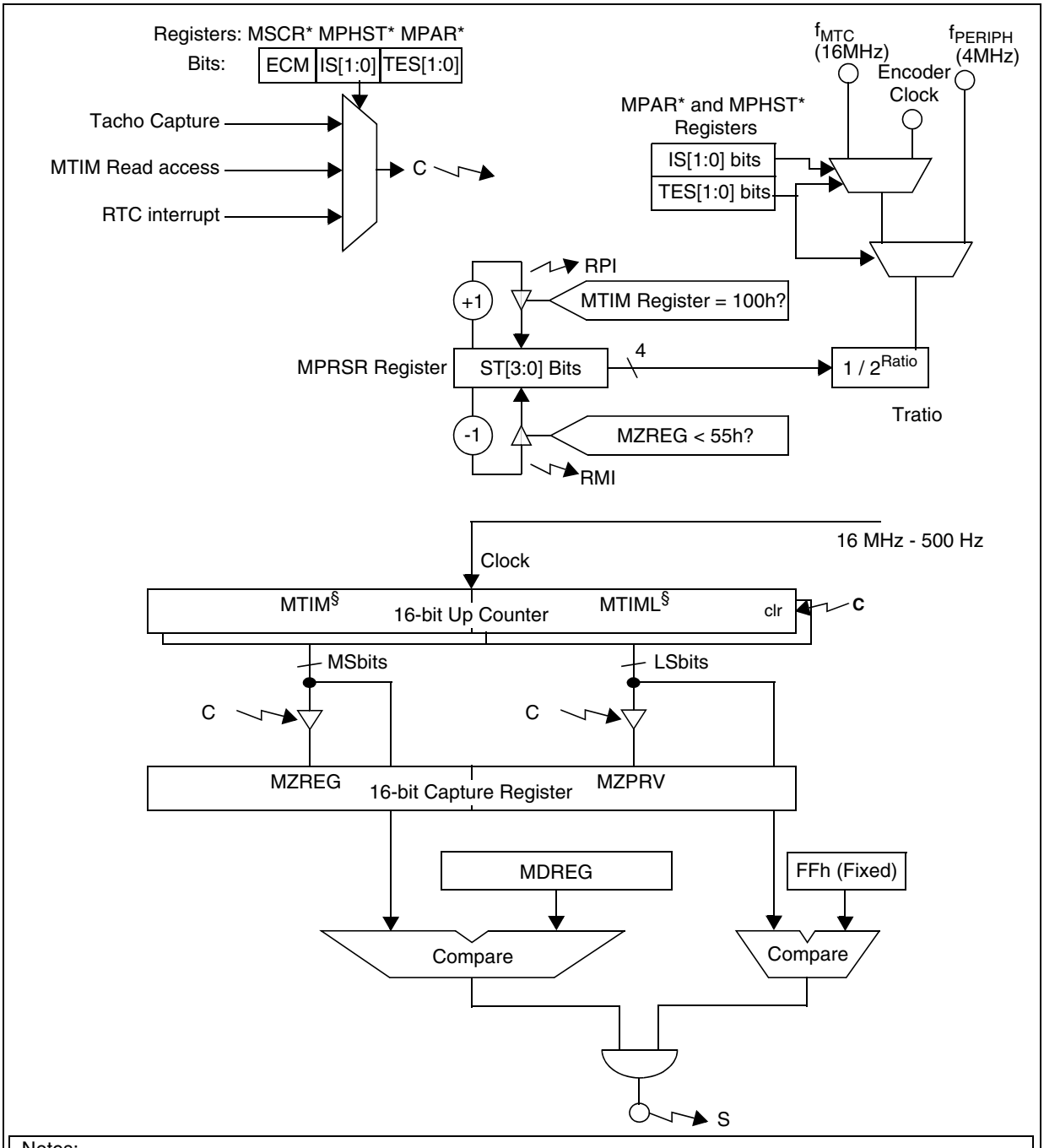
Note: This functionality is not useful when using an encoder. With an encoder, user must monitor the captured values by software during the periodic capture interrupts: for instance, when driving an AC motor, if the values are too low compared to the stator frequency, a software interrupt may be triggered.

Figure 101. Hall sensor period acquisition using MTIM timer



MOTOR CONTROLLER (Cont'd)

Figure 102. Overview of MTIM Timer in Speed Measurement Mode



Notes:

§ = Register updated on R event

* = Register set-up described in Speed Sensor Mode Section

C → To interrupt generator (Capture Event)

S → To interrupt generator (Speed Error Event)

RPI → To interrupt generator (Ratio Increment Event)

RMI → To interrupt generator (Ratio Decrement Event)

MOTOR CONTROLLER (Cont'd)

A logic block manages capture operations depending on the sensor type. A capture is initiated on an active edge ("Tacho capture" event) when using a tachogenerator.

If an encoder is used, the capture is triggered on two events depending on the Encoder Capture Mode bit (ECM) in the MZFR register:

– Reading the MSB of the counter in manual mode (ECM = 1)
– Interrupt from the Real-time Clock in automatic mode (ECM = 0)

The clock source of the counter is selected depending on sensor type:

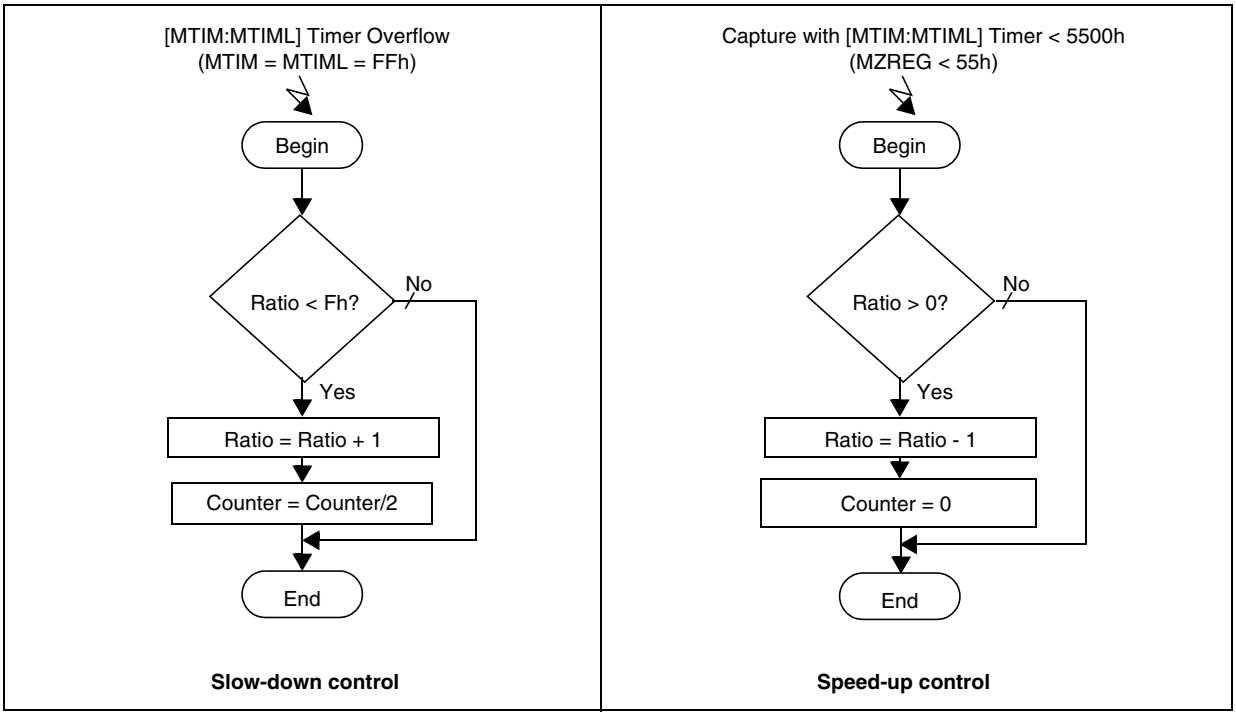
– Motor Control Peripheral clock (16 MHz) with tachogenerator or Hall sensors
– Encoder Clock

In order to optimize the accuracy of the measurement for a wide speed range, the auto-updated prescaler functionality is used with slight modifications compared to Sensor/Sensorless Modes (refer to Figure 103 and Table 38).

– When the [MTIM:MTIML] timer value reaches FFFFh, the prescaler is automatically incremented in order to slow down the counter and avoid an overflow. To keep consistent values, the MTIM and MTIML registers are shifted right (divided by two). The RPI bit in the MISR register is set and an interrupt is generated (if RIM is set).
– When a capture event occurs, if the [MTIM:MTIML] timer value is below 5500h, the prescaler is automatically decremented in order to speed up the counter and keep precision better than 0.005% (1/5500h). The MTIM and MTIML registers are shifted left (multiplied by two). The RMI bit in the MISR register is set and an interrupt is generated if RIM is set.
– If the prescaler contents reach the value 0, it can no longer be automatically decremented, the [MTIM:MTIML] timer continues working with the same prescaler value, i.e. with a lower accuracy. No RMI interrrupt can be generated.
– If the prescaler contents reach the value 15, it can no longer be automatically incremented. When the timer reaches the value FFFFh, the prescaler and all the relevant registers remain unchanged and no interrupt is generated, the timer clock is disabled, and its contents stay at FFFFh. The capture logic block still works, enabling the capture of the maximum timer value.

The only automatically updated registers for the Speed Sensor Mode are MTIM and MTIML. Access to Delay manager registers in Speed Sensor Mode is summarised in Table 41.

Figure 103. Auto-updated prescaler functional diagram



MOTOR CONTROLLER (Cont'd)

Three kinds of interrupt can be generated in Speed Sensor Mode, as summarized in Figure 104:

– C interrupt, when a capture event occurs; this interrupt shares resources (Mask bit and Flag) with the Commutation event in Switched/Autoswitched Mode, as these modes are mutually exclusive.

– RPI/RMI interrupts occur when the ST[3:0] bits of the MPSR register are changed, either automatically or by hardware.

– S interrupt occurs when a Speed Error happens (i.e. a successful comparison between [MTIM:MTIML] and [MDREG:FF]). This interrupt has the same channel as the Emergency Stop interrupt (MCES), as it also warns the user about abnormal system operation. The respective Flag bits have to be tested in the interrupt service routine to differentiate Speed Errors from Emergency Stop events.

These interrupts may be masked individually.

Note on Delay Manager Initialization in Speed Measurement Mode: In order to set-up the [MTIM:MTIML] counter properly before any speed measurement, the following procedure must be applied:

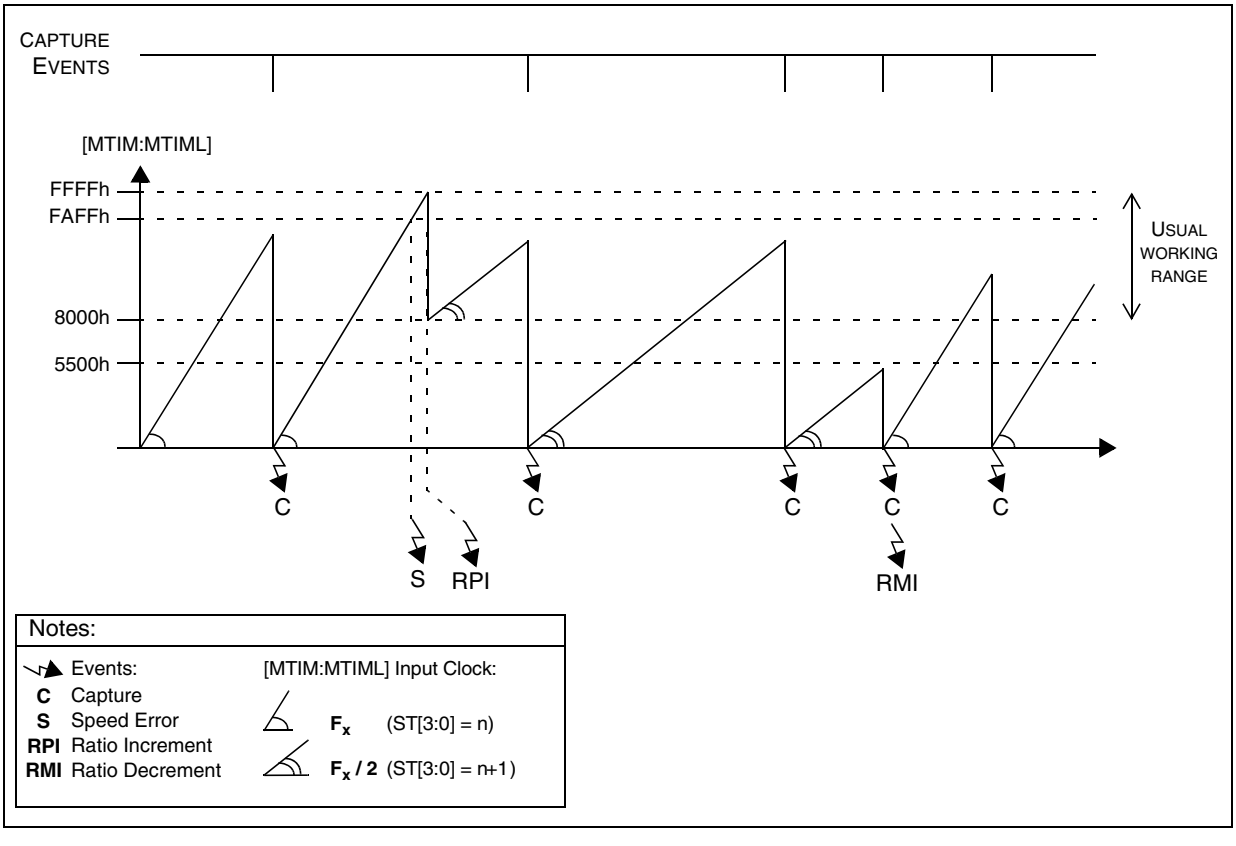
– The peripheral clock must be disabled (resetting the CKE bit in the MCRA register) to allow write access to ST[3:0], MTIM and MTIML (refer to Table 41),

– MTIM, MTIML must be reset and appropriate values must be written in the ST[3:0] prescaler adapt to the frequency of the signal being measured and to allow speed measurement with sufficient resolution.

Note on MTIML: The Least Significant Byte of the counter (MTIML) is not used when working in Position Sensor or Sensorless Modes.

Debug option: a signal reflecting the capture events may be output on a standard I/O port for debugging purposes. Refer to section10.6.7.3 on page 172 for more details.

Figure 104. Prescaler auto-change example



MOTOR CONTROLLER (Cont'd)

10.6.7.6 Summary

The use of the Delay manager registers for the various available modes is summarized in Table 42.

Table 42. MTIM Timer-related Registers

Name	Reset Value	Switched / Auto Switched Mode	Speed Measurement Mode
MTIM	00h	Timer Value	16-bit Timer MSB Value
MTIML	00h	N/A	16-bit Timer LSB Value
MZREG	00h	Capture/compare Zn	Capture of 16-bit Timer MSB
MZPRV	00h	Capture Zn-1	Capture of 16-bit Timer LSB
MCOMP	00h	Compare Cn+1	N/A
MDREG	00h	Demagnetization Dn	Compare for Speed Error interrupt generation

10.6.8 PWM Manager

The PWM manager controls the motor via the six output channels in voltage mode or current mode depending on the V0C1 bit in the MCRA register. A block diagram of this part is given in Figure 106.

10.6.8.1 Voltage Mode

In Voltage mode (V0C1 bit = "0"), the PWM signal which is applied to the switches is generated by the 12-bit PWM Generator compare U.

Its duty cycle is programmed by software (refer to the PWM Generator section) as required by the application (speed regulation for example).

The current comparator is used for safety purposes as a current limitation. For this feature, the detected current must be present on the MCCFI pin and the current limitation must be present on pin MCCREF. This current limitation is fixed by a voltage reference depending on the maximum current acceptable for the motor. This current limitation is generated with the V_{DD} voltage by means of an external resistor divider but can also be adjusted with an external reference voltage ($\leq$5 V). The external components are adjusted by the user depending on the application needs. In Voltage mode, it is mandatory to set a current limitation. As this limitation is set for safety purposes, an interrupt can be generated when the motor current feedback reaches the current limitation in voltage mode. This is the current limitation interrupt and it is enabled by setting the corresponding CLM bit in the MIMR register. This is useful in voltage mode for security purposes.

The PWM signal is directed to the channel manager that connects it to the programmed outputs (see Figure 106).

MOTOR CONTROLLER (Cont'd)

10.6.8.2 Over Current Handling in Voltage mode

When the current limitation interrupt is enabled by setting the CLIM bit in the MIMR register (available only in Voltage mode), the OCV bit in MCRB register will determine the effect of this interrupt on the MCOx outputs as shown in Table 43.

Table 43. OCV bit effect

CLIM bit	CLI bit	OCV bit	Output effect	Interrupt
0	0	x	Normal running mode	No
0	1	x	PWM is put OFF on Current loop effect	No
1	0	x	Normal running mode	No
1	1	0	PWM is put OFF on Current loop effect	Yes
1	1	1	All MCOx outputs are put in reset state (MOE reset)[1)]	Yes

For safety purposes, it can be necessary to put all MCOx outputs in reset state (high impedance, high state or low state depending on the setting made by the option byte) on a current limitation interrupt. This is the purpose of the OCV bit. When a current limitation interrupt occurs, if the OCV bit is reset, the effect on the MCOx outputs is only to put the PWM signal OFF on the concerned outputs. If the OCV bit is set, when the current limitation interrupt occurs, all the MCOx outputs are put in reset state.

Note 1: Only this functionality (CLIM = CLI = OCV = 1) is valid when the 3 PWM channels are enabled (PCN bit =1 in the MDTG register). It can also be used as an over-current protection for three-phase PWM application (only if voltage mode is selected)

10.6.8.3 Current Mode

In current mode, the PWM output signal is generated by a combination of the output of the measurement window generator (see Figure 107) and the output of the current comparator, and is directed to the output channel manager as well (Figure 108).

The current reference is provided to the comparator by Phase U, V or W of the PWM Generator (up to 12-bit accuracy) the signal from the three compare registers U, V or W can be output by setting the PWMU, PWMV or PWMW bits in the MPWME register. The PWM signal is filtered through an external RC filter on pin MCCREF.

The detected current input must be present on the MCCFI pin.

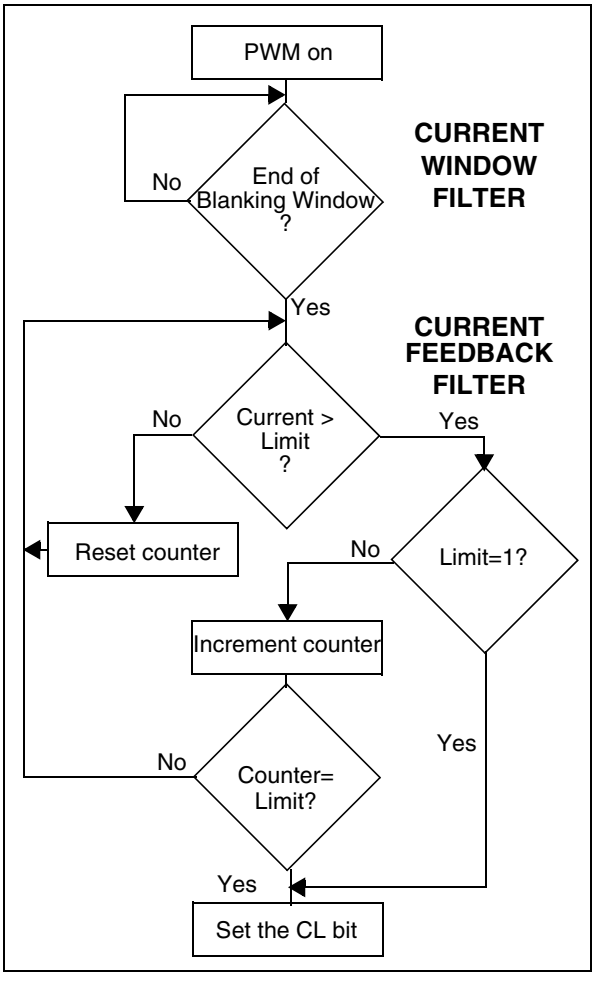
10.6.8.4 Current Feedback Comparator

Two programmable filters are implemented:

– A blanking window (Current Window Filter) after PWM has been switched ON to avoid spurious PWM OFF states caused by parasitic noise

– An event counter (Current Feedback Filter) to prevent PWM being turned OFF when the first comparator edge is detected.

Figure 105. Current Window and Feedback Filters



MOTOR CONTROLLER (Cont'd)

Table 44. Current Window filter Setting

CFW2	CFW1	CFW0	Blanking window length
0	0	0	Blanking window off
0	0	1	0.5 µs
0	1	0	1 µs
0	1	1	1.5 µs
1	0	0	2 µs
1	0	1	2.5 µs
1	1	0	3 µs
1	1	1	3.5 µs

Note: Times are indicated for 4 MHz f_{PERIPH}

The Current Window filter is activated each time the PWM is turned ON. It blanks the output of the current comparator during the time set by the CFW[2:0] bits in the MCFR register. The reset value is 000b (blanking window off).

The Current feedback filter sets the number of consecutive valid samples (when current is above the limit) needed to generate the active CL event used to turn OFF the PWM. The reset value is 1.

The sampling of the current comparator is done at $f_{PERIPH}/4$.

Table 45. Current Feedback Filter Setting

CFF2	CFF1	CFF0	Nb of Feedback Samples needed to turn OFF PWM
0	0	0	1
0	0	1	2
0	1	0	3
0	1	1	4
1	0	0	5
1	0	1	6
1	1	0	7
1	1	1	8

The ON time of the resulting PWM starts at the end of the measurement window (rising edge), and ends either at the beginning of the next measurement window (falling edge), or when the current level is reached.

Note: Be careful that the current comparator is OFF until the CKE and/or DAC bits are set in the MCRA register.

MOTOR CONTROLLER (Cont'd)

10.6.8.5 Current feedback amplifier

In both current and voltage mode, the current feedback from the motor can be amplified before entering the comparator. This is done by an integrated Op-amp that can be used when the OAON bit is set in the OACSR register and the CFAV bit in the MREF register is reset. This allows the three points of the Op-amp to be accessed for a programmable gain. The CFAV bit in the MREF register selects the MCCFI0 or OAZ(MCCFI1) pin as the comparator input as shown in the following table.

Table 46. Comparator input selection

CFAV bit	Meaning
0	Select OAZ(MCCFI1) as the current comparator input
1	Select MCCFI0 as the current comparator input

If the amplifier is not used for current feedback, it can be used for other purposes. In this case, the OAON bit in the OACSR register and the CFAV bit in the MREF register both have to be set. This means that the current feedback has to be on the MCCFI0 pin to be directly connected to the comparator and the OAP, OAN and OAZ (MCCFI1) pins can be used to amplify another signal. Both the OAZ(MCCFI1) and MCCFI0 pins can be connected to an ADC entry. See (Figure 106).

Note: The MCCFI0 pin is not available in LQFP32; SDIP32 and LQFP44 devices. In this case, the CFAV bit must be reset. The choice to use the Op-amp or not is made with the OAON bit.

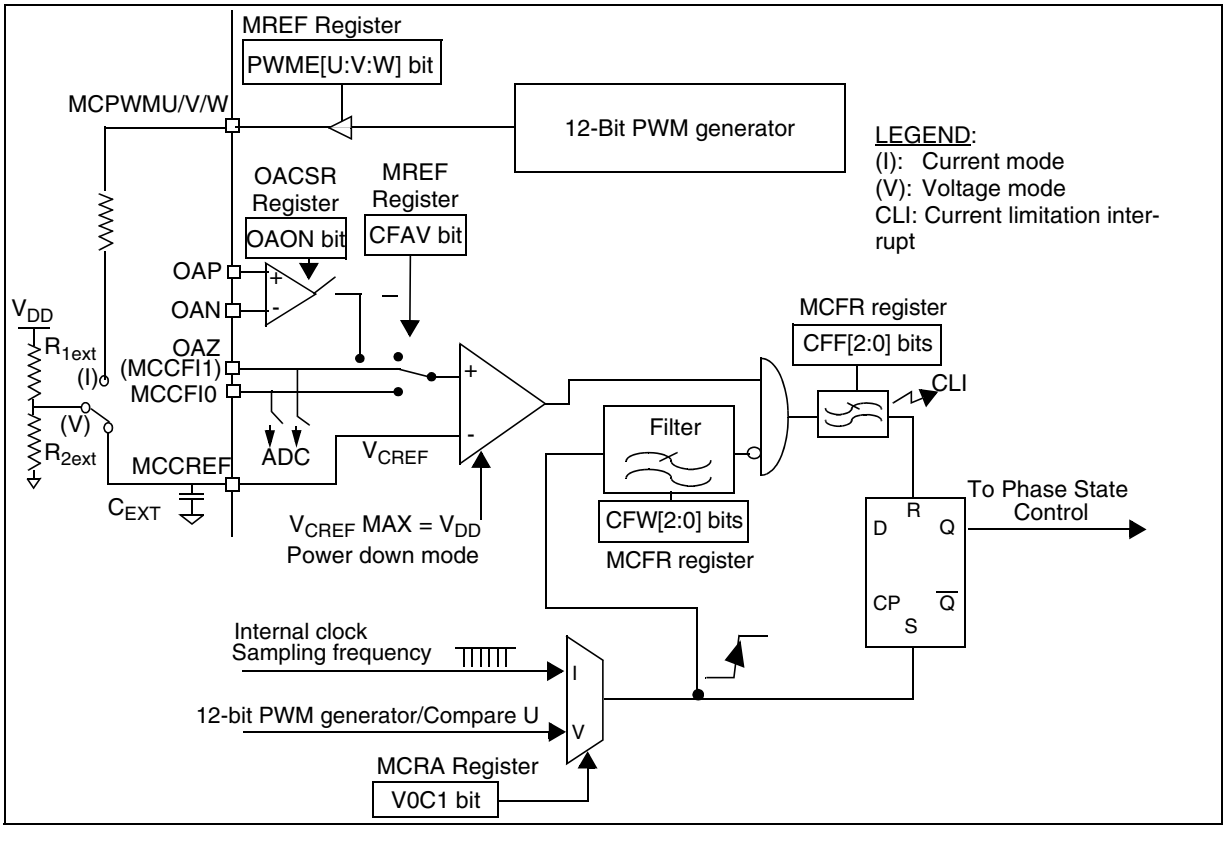
10.6.8.6 Measurement Window

In current mode, the measurement window frequency can be programmed between 390Hz and 50KHz by the means of the SA[3:0] bits in the MPRSR register.

Note: These frequencies are given for a 4 MHz peripheral input frequency for a BLDC drive (XT16, XT8 bits in MCONF register).

In sensorless mode this measurement window can be used to detect BEMF zero crossing events. Its width can be defined between 2.5µs and 40µs as a minimum in sensorless mode by the OT[3:0] bits in the MPWME register.

Figure 106. Current Feedback



MOTOR CONTROLLER (Cont'd)

This sets the minimum off time of the PWM signal generated by this internal clock. This off time can vary depending on the output of the current feedback comparator. In sensor mode (SR=1) and when the sampling for the Z event is done during the PWM ON time in sensorless mode (SPLG bit is set in MCRC register and /or DS[3:0] bits with a value other than 000 in MCONF register), there is no minimum OFF time required anymore, the minimum off time is set automatically to 0µs and the OFF time of the PWM signal is controlled only by the current regulation loop.

Table 47. Sampling Frequency Selection

SA3	SA2	SA1	SA0	Sampling Frequency
0	0	0	0	50.0 KHz
0	0	0	1	40.0 KHz
0	0	1	0	33.33 KHz
0	0	1	1	25.0 KHz
0	1	0	0	20.0 KHz
0	1	0	1	18.1 KHz
0	1	1	0	15.4 KHz
0	1	1	1	12.5 KHz
1	0	0	0	10 KHz
1	0	0	1	6.25 KHz
1	0	1	0	3.13 KHz
1	0	1	1	1.56 KHz
1	1	0	0	1.25 KHz
1	1	0	1	961 Hz
1	1	1	0	625 Hz
1	1	1	1	390 Hz

Note: Times are indicated for 4 MHz f_{PERIPH}

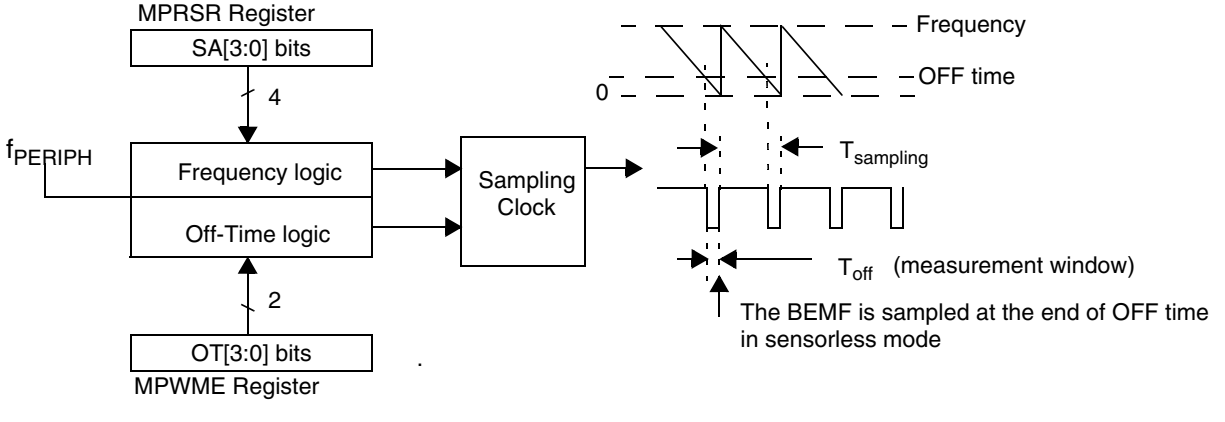
Warning: If the off time value set is superior than the period of the PWM signal (for example 40µs off time for a 50KHz (25µs period) PWM frequency), then the signal output on MCOx pins selected is a 100% duty cycle signal (always at 1).

Table 48. Off time table

OT3	OT2	OT1	OT0	Off Time sensorless mode (SR=0) (DS[3:0]=0)	Sensor Mode (SR=1) or sampling during ON time in sensorless (SPLG =1 and/or DS[3:0] bits)
0	0	0	0	2.5 µs	No minimum off time
0	0	0	1	5 µs	
0	0	1	0	7.5 µs	
0	0	1	1	10 µs	
0	1	0	0	12.5 µs	
0	1	0	1	15 µs	
0	1	1	0	17.5 µs	
0	1	1	1	20 µs	
1	0	0	0	22.5 µs	
1	0	0	1	25 µs	
1	0	1	0	27.5 µs	
1	0	1	1	30 µs	
1	1	0	0	32.5 µs	
1	1	0	1	35 µs	
1	1	1	0	37.5 µs	
1	1	1	1	40 µs	

Note: Times are indicated for 4 MHz f_{PERIPH}

Figure 107. Sampling clock generation block



Note: The MTC controller input frequency (f_{PERIPH}) is 4 MHz in this example, It can be configured to 8MHz with the XT16: XT8 bits in the MCONF register

MOTOR CONTROLLER (Cont'd)

10.6.9 Channel Manager

The channel manager consists of:

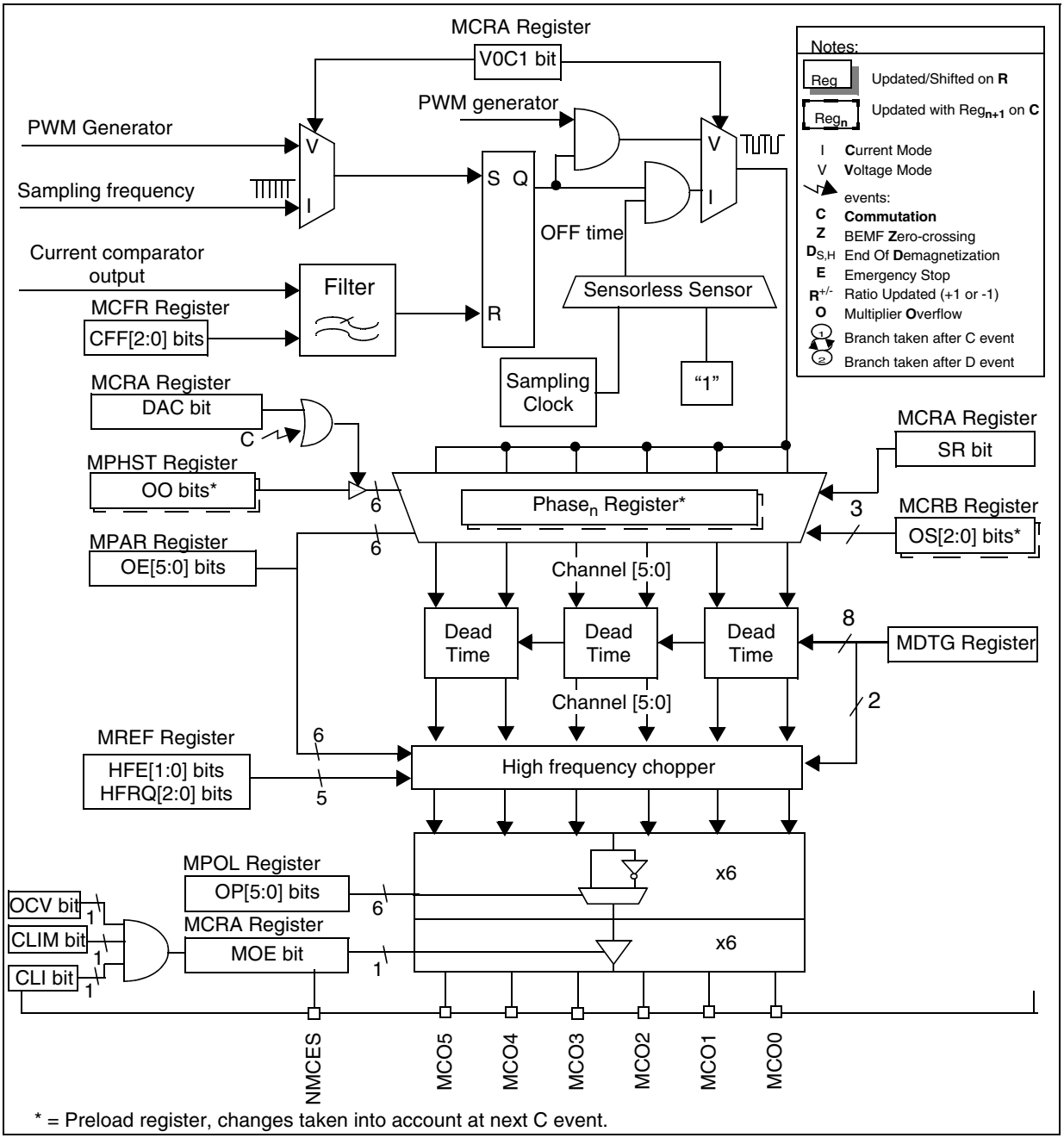
– A Phase State register with preload and polarity function

– A multiplexer to direct the PWM to the low and/ or high channel group

– A tristate buffer asynchronously driven by an emergency input

The block diagram is shown in Figure 108.

Figure 108. Channel Manager Block Diagram



MOTOR CONTROLLER (Cont'd)

10.6.9.1 MPHST Phase State Register

A preload register enables software to asynchronously update the channel configuration for the next step (during the previous commutation interrupt routine for example): the OO[5:0] bits in the MPHST register are copied to the Phase register on a C event.

Table 49. Output State

OP[5:0] bit	OO[5:0] bit	MCO[5:0] Pin
0	0	1 (OFF)
0	1	0-(PWM allowed)
1	0	0 (OFF)
1	1	1-(PWM allowed)

Direct access to the phase register is also possible when the DAC bit in the MCRA register is set.

Note: In Direct Access Mode (DAC bit is set in MCRA register):

1: A C event is generated as soon as there is a write access to OO[5:0] bits in MPHST register,
2: The PWM application is selected by the OS0 bit in the MCRB register,
3: Regardless of the value of the CKE bit in the MCRA register, the MTIM Clock is disabled and D and Z events are not detected.

Table 50. DAC and MOE Bit Meaning

MOE bit	DAC bit	Effect on Output
0	x	Reset state*
1	0	Standard running mode
1	1	MPHST register value (depending on MPOL, MPAR register values and PWM setting) see Table 74

***Note:** The reset state of the outputs can be either high impedance, low or high state depending on the corresponding option bit.

The polarity register is used to match the polarity of the power drivers keeping the same control logic and software. If one of the OPx bits in the MPOL register is set, this means the switch x is ON when MCOx is V_{DD}.

Each output status depends also on the momentary state of the PWM, its group (low or high), and the peripheral state.

PWM Features

The outputs can be split in two PWM groups in order to differentiate the high side and the low side switches. This output property can be programmed using the OE[5:0] bits in the MPAR register.

Table 51. Meaning of the OE[5:0] Bits

OE[5:0]	Channel group
0	High channel
1	Low channel

The multiplexer directs the PWM to the upper channel, the lower channel or both of them alternatively or simultaneously according to the peripheral state.

This means that the PWM can affect any of the upper or lower channels allowing the selection of the most appropriate reference potential when freewheeling the motor in order to:

- Improve system efficiency
- Speed up the demagnetization phase
- Enable Back EMF zero crossing detection.

The OS[2:0] bits in the MCRB register allow the PWM configuration to be configured for each case as shown in Figure 110 and Figure 109.

During demagnetization, the OS2 bit is used to control PWM mode, and it is latched in a preload register so it can be modified when a commutation event occurs and the configuration is active immediately.

The OS1 bit is used to control the PWM between the D and Z events to control back-emf detection.

OS0 bit will allow to control the PWM signal between Z event and next C event.

Note about demagnetization speed-up: during demagnetization the voltage on the winding has to be as high as possible in order to reduce the demagnetization time. Software can apply a different PWM configuration on the outputs between the C and D events, to force the free wheeling on the appropriate diodes to maximize the demagnetization voltage.

10.6.9.2 Emergency Feature

When the NMCES pin goes low

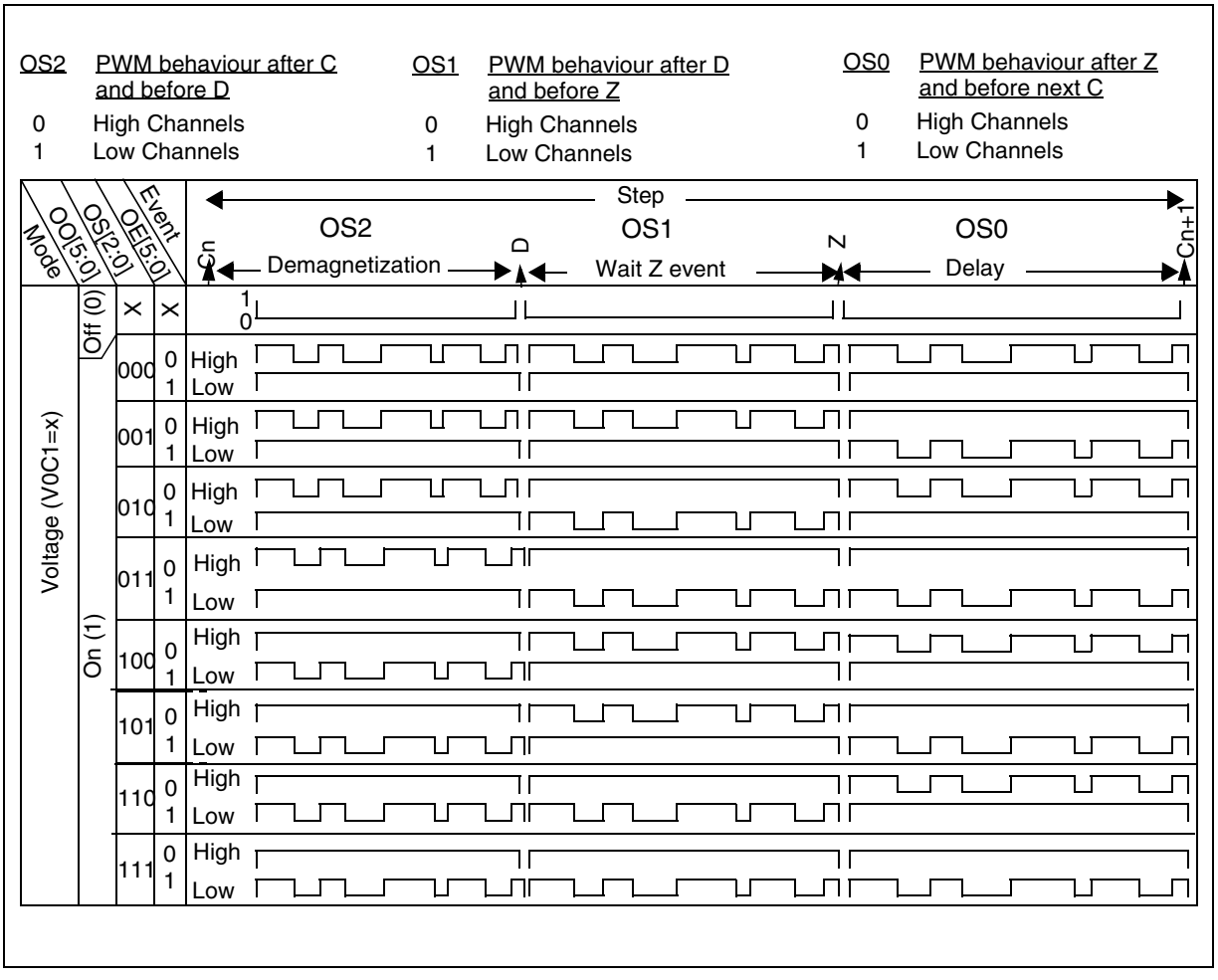
- The tristate output buffer is put in reset state asynchronously
- The MOE bit in the MCRA register is reset
- An interrupt request is sent to the CPU if the EIM bit in the MIMR register is set

This bit can be connected to an alarm signal from the drivers, thermal sensor or any other security component.

This feature functions even if the MCU oscillator is off.

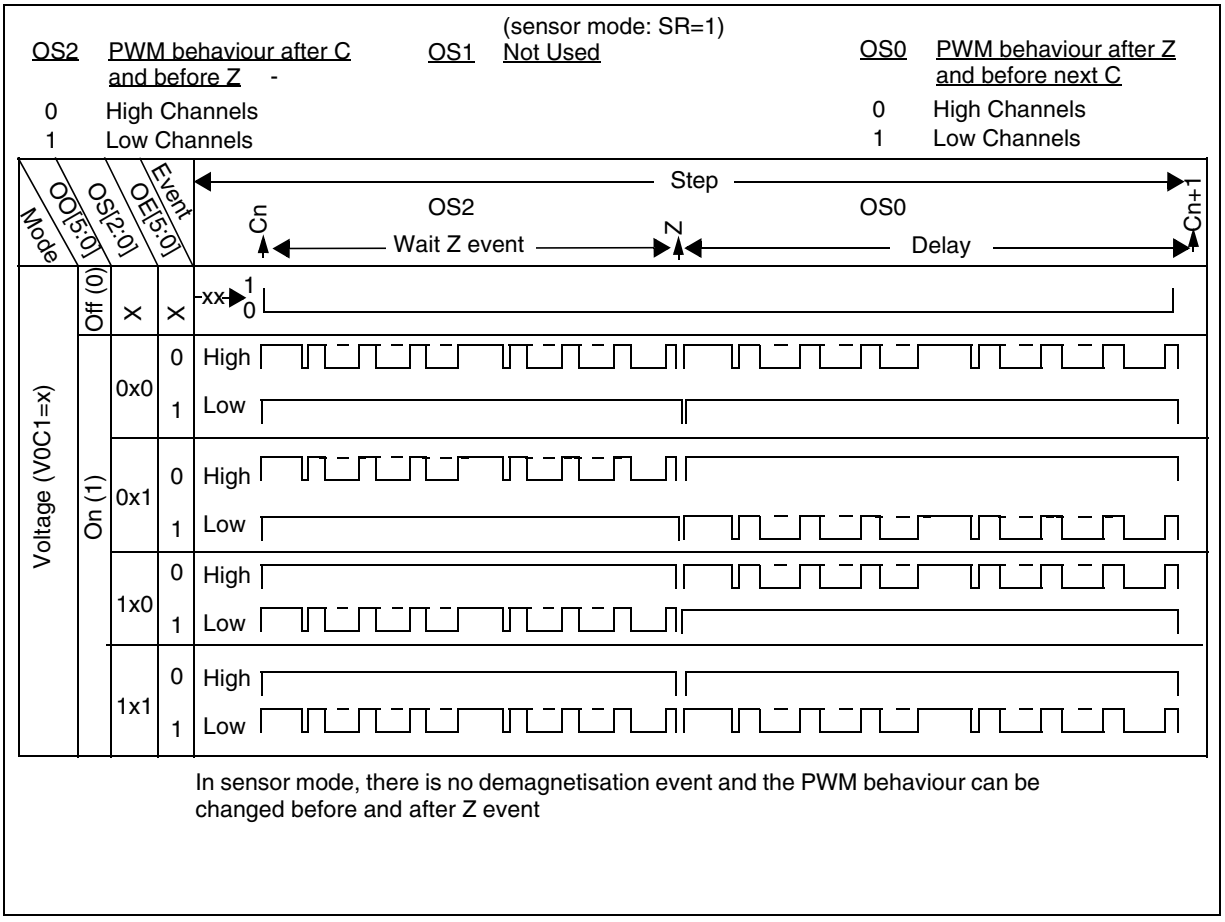
MOTOR CONTROLLER (Cont'd)

Figure 109. PWM application in Voltage or Current sensorless mode (see Table 61)



MOTOR CONTROLLER (Cont'd)

Figure 110. PWM application in Voltage or Current Sensor Mode (see Table 62)



In sensor mode, there is no demagnetisation event and the PWM behaviour can be changed before and after Z event

MOTOR CONTROLLER (Cont'd)

10.6.9.3 Dead Time Generator

When using typical triple half bridge topology for power converters, precautions must be taken to avoid short circuits in half bridges. This is ensured by driving high and low side switches with complementary signals and by managing the time between the switching-off and the switching-on instants of the adjacent switches.

This time is usually known as deadtime and has to be adjusted depending on the devices connected to the PWM outputs and their characteristics (intrinsic delays of level-shifters, delays due to power switches,...).

When driving motors in six-step mode, the deadtime generator function also allows synchronous rectification to be performed on the switch adjacent to the one where PWM is applied to reduce conduction losses.

For each of the three PWM channels, there is one 6-bit Dead Time generator available.

It generates two output signals: A and B.

The A output signal is the same as the input phase signal except for the rising edge, which is delayed relative to the input signal rising edge.

The B output signal is the opposite of the input phase signal except the rising edge which is delayed relative to the input signal falling edge.

Figure 111 shows the relationship between the output signals of the deadtime register and its inputs.

If the delay is greater than the width of the active phase (A or B) then the corresponding pulse is not generated (see Figure 112 and Figure 113).

Figure 111. Dead Time waveforms

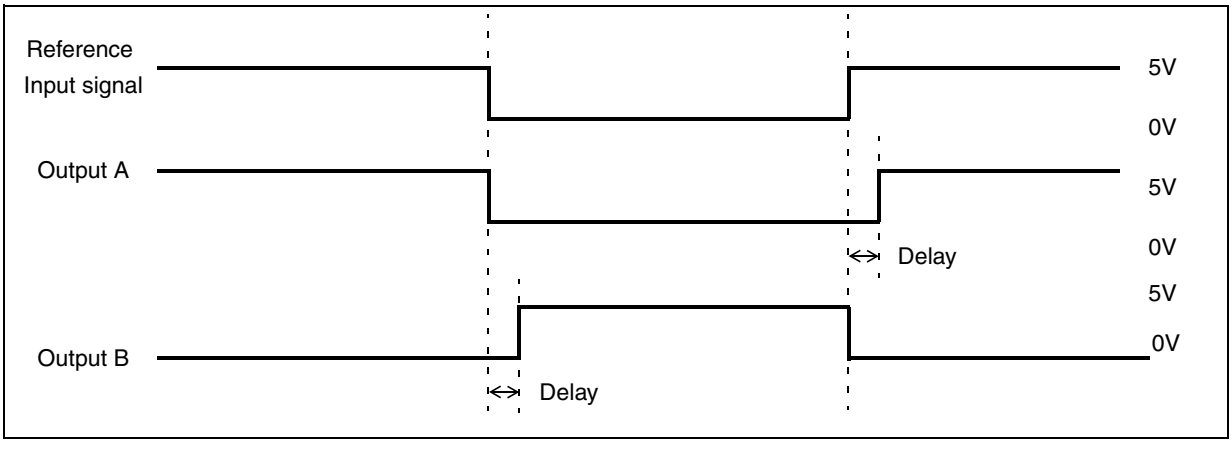
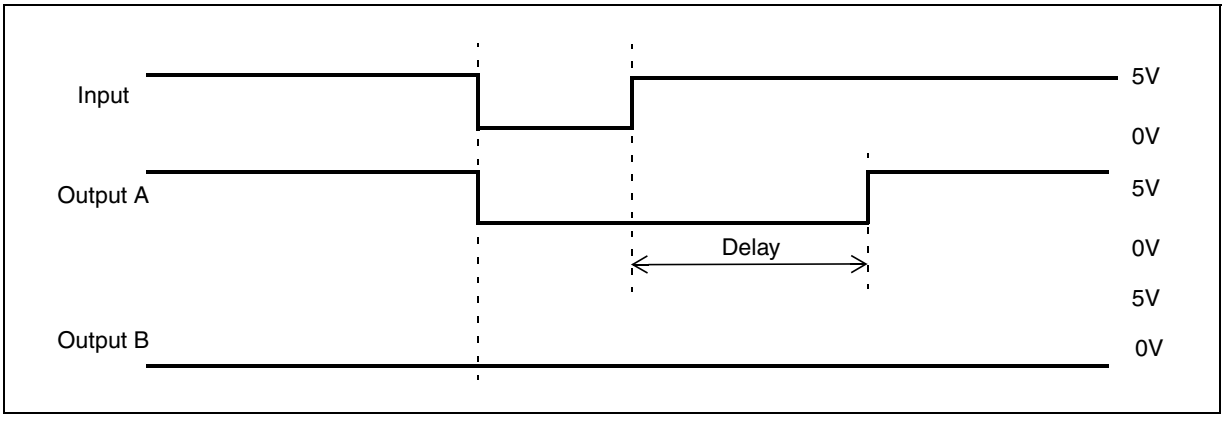


Figure 112. Dead time waveform with delay greater than the negative PWM pulse



MOTOR CONTROLLER (Cont'd)

Figure 113. Dead Time waveform with delay greater than the positive PWM pulse

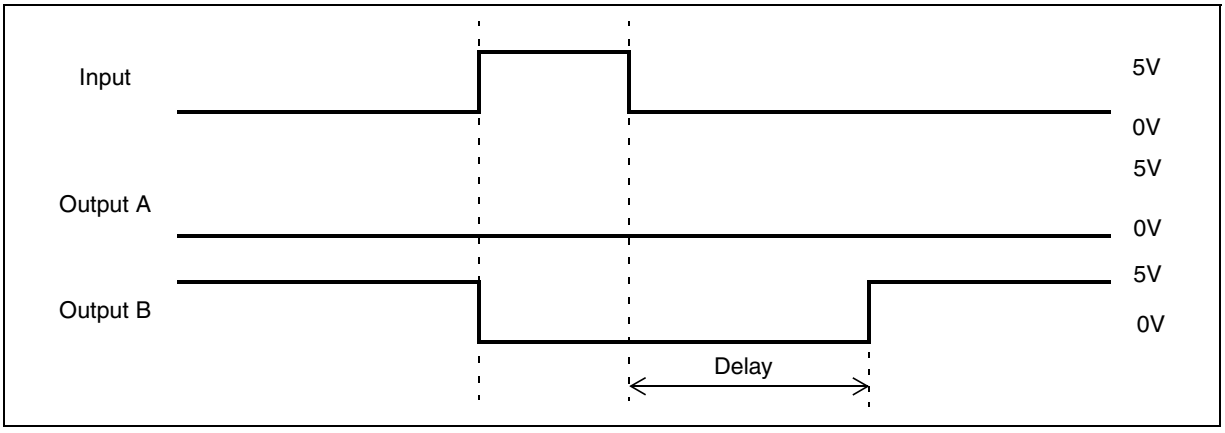


Table 52. Dead time programming and example

DTG5	DTG4	T_{dtg}	Deadtime expression	Deadtime value	T_{dtg} @16MHz F_{mtc}	Dead time range @ 16MHz F_{mtc}
0	X	$2xT_{mtc}$	(DTG[4..0]+1) x T_{dtg}	From 1 to 32 T_{dtg}	125ns	0.125µs to 4µs
1	0	$4xT_{mtc}$	(DTG[3..0]+17) x T_{dtg}	From 17 to 32 T_{dtg}	250ns	4.25µs to 8µs
1	1	$8xT_{mtc}$			500ns	8.5µs to 16µs

The deadtime delay is the same for each of the channels and is programmable with the DTG[5..0] bits in the MDTG register.

The resolution is variable and depends on the DTG5 and DTG4 bits. Table 52 summarizes the set-up of the deadtime generator.

IT_{mtc} is the period of the Dead Time Generator input clock (F_{mtc} = 16 MHz in most cases, not affected by the XT16:XT8 prescaler bits in the MCONF register).

For safety reasons and since the deadtime depends only on external component characteristics (level-shifter delay, power components switching duration,...) the register used to set-up deadtime duration can be written only once after the MCU reset. This prevents a corrupted program counter modifying this system critical set-up, which may cause excessive power dissipation or destructive shoot-through in the power stage half bridges.

When using the three independent U, V and W PWM signals (PCN bit set) (see Figure 114) to drive the MCOx outputs, deadtime is added as shown in Figure 111.

The dead time generator is enabled/disabled using the DTE bit.

The effect of the DTE bit depends on the PCN bit value.

If the PCN bit is set:

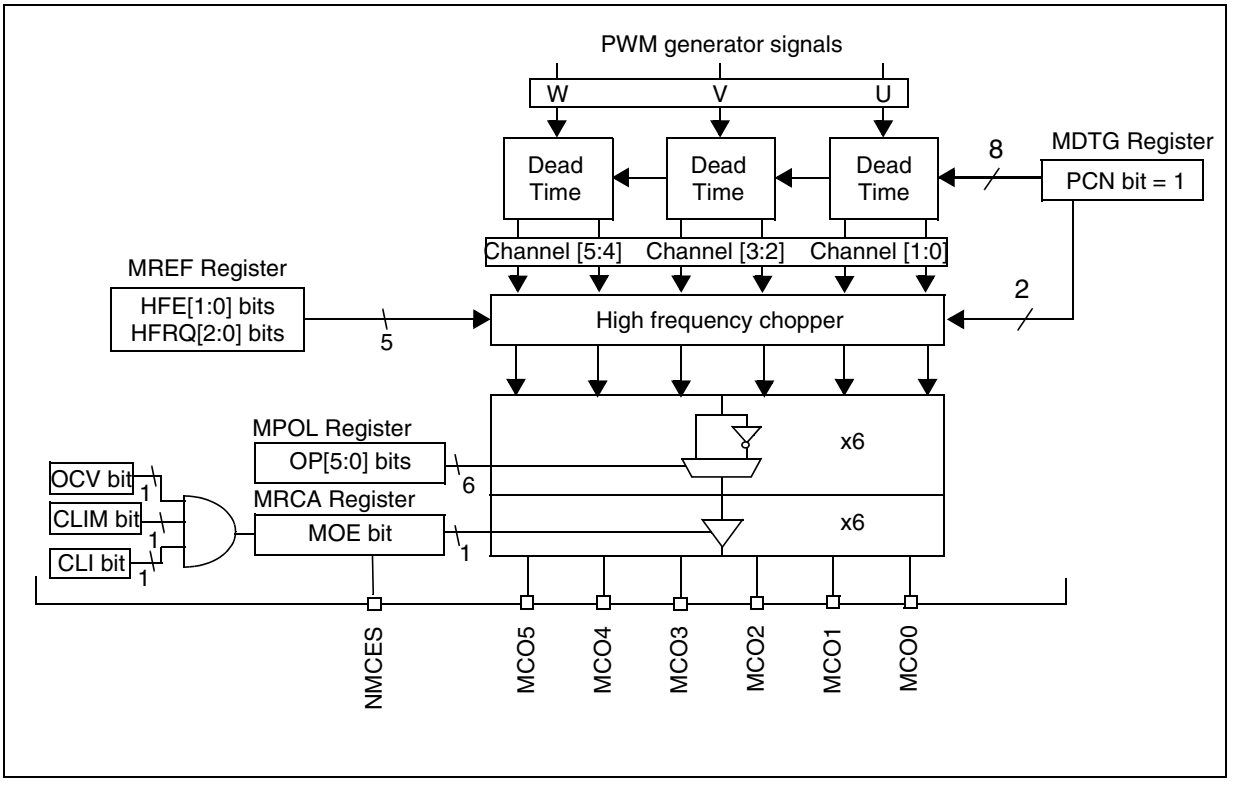
- DTE is read only. To reset it, first reset the PCN bit, then reset DTE and set PCN to 1 again.
- If DTE=0, the high and low side outputs are simply complemented (no deadtime insertion, DTG[5:0] bits are not significant); this is to allow the use of an external dead time generator.

Note: The reset value of the MDTG register is FFh so when configuring the dead time, it is mandatory to follow one the two following sequences:

- To use dead t imes while the PCN bit is set; from reset state write the MDTG value at once. The DTE bit will be read back as 1 whatever the programming value (read only if PCN=1)
- To use dead times while the PCN bit is reset, write first the dead time value in DTG[5:0], then reset the PCN bit, or do both actions at the same time.

MOTOR CONTROLLER (Cont'd)

Figure 114. Channel Manager Output Block Diagram with PWM generator delivering 3 PWM signals



Note: The output of the current limitation comparator can be used when 3 PWM signals are enabled if the VOC1 bit =0 in the MCRA register.

MOTOR CONTROLLER (Cont'd)

If the PCN bit is reset, one of the three PWM signals (the one set by the compare U register pair) or the output of the measurement window generator (depending on if the driving mode is voltage or current) is used to provide six-step signals through the PWM manager (to drive a PM BLDC motor for instance).

In that case, DTE behaves like a standard bit (with multiple write capability). When the deadtime generator is enabled (bit DTE=1), some restrictions are applied, summarized in Table 53:

- Channels are now grouped by pairs: Channel[0:1], Channel[2:3], Channel[4:5]; a deadtime generator is allocated to each of these pairs (see cautions below);
- The input signal of the deadtime generator is the active output of the PWM manager for the corresponding channel. For instance, if we consider the Channel[0:1] pair, it may be either Channel0 or Channel1.
- When both channels of a pair are inactive, the corresponding outputs will also stay inactive (this is mandatory to allow BEMF zero-crossing detection).

Table 53 summarizes the functionality of the deadtime generator when the PCN bit is reset. 1(pwm*) means that the corresponding channel is active (1 in the corresponding bit in the MPHST register), and a PWM signal is applied on it (using the MPAR register and the OS[2:0] bits in MCRB register). $\overline{PWM}$ represents the complementary signals (although the duty cycle is slightly different due to deadtime insertion). 0 means that the channel is inactive and 1 means that the channel is active and a logic level 1 is applied on it (no PWM signal).

Table 53. Dead Time generator outputs

PCN = 0; DTE =1; x= 0, 2, 4			
On/Off x (OOx bit)	**On/Off x+1 (OOx+1 bit)**	**MCOx output**	**MCOx+1 output**
0	1 (pwm*)	$\overline{PWM}$	PWM
1 (pwm*)	0	PWM	$\overline{PWM}$
1	1 (pwm*)	0	0
1 (pwm*)	1	0	0
1	0	1	0
0	1	0	1
0	0	0	0

* PWM generation enabled

Warning: Grouping channels by pairs imposes the external connections between the MCO outputs and power devices; the user must therefore pay attention to respect the "recommended schematics" described in Figure 123. on page 228 and Figure 124

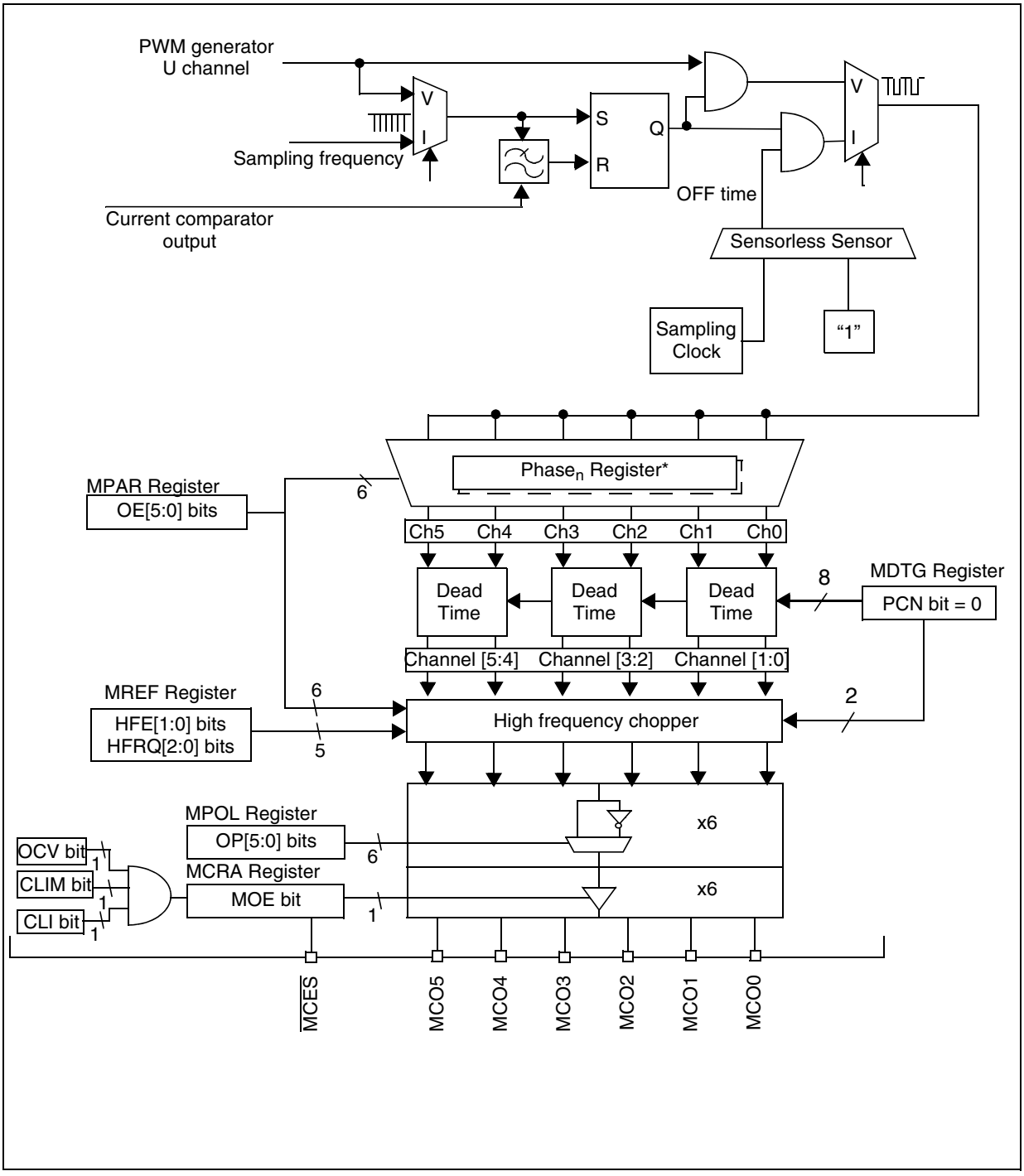
Note: As soon as the channels are grouped in pairs, special care has to be taken in configuring the MPAR register for a PM BLDC drive. If both channels of the same pair are both labelled "high" for example and if the PWM is applied on high channels, the active MCO output x (OOx=1 bit in the MPHST register) outputs PWM and the paired MCO output x+1 (OOx+1bit in the MPHST register) outputs $\overline{PWM}$ and vice versa.

Caution: When PCN=0 and a complementary PWM is applied (DTE=1) on one channel of a pair, if both channels are active, this corresponds in output to both channels OFF. This is for security purpose to avoid cross-conduction.

Caution: To clear the DTE bit from reset state of MDTG register (FFh), the PCN bit must be cleared before.

MOTOR CONTROLLER (Cont'd)

Figure 115. Channel Manager Output Block Diagram with PWM generator delivering 1 PWM signal



MOTOR CONTROLLER (Cont'd)

10.6.9.4 Programmable Chopper

Depending on the application hardware (use of a pulse transformer, for example), a chopper may be needed for the PWM signal. The MREF register allows the chopping frequency and mode to be programmed.

The HFE[1:0] bits program the channels on which chopping is to be applied. The chopped PWM signal may be needed for high side switches only, low side switches or both of them in the same time (see Table 54).

Table 54. Chopping mode

HFE[1:0] bits		Chopping mode	
HFE1	HFE0	PCN bit =0	PCN bit =1
0	0	OFF	OFF
0	1	Low channels only	Low side switches MCO1, 3, 5
1	0	High channels only	High side switches MCO0, 2, 4
1	1	Both Low and High channels	Both high and low sides

The chopping frequency can any of the 8 values from 100KHz to 2MHz selected by the HFRQ[2:0] bits in the MREF register (see Table 55).

Table 55. Chopping frequency

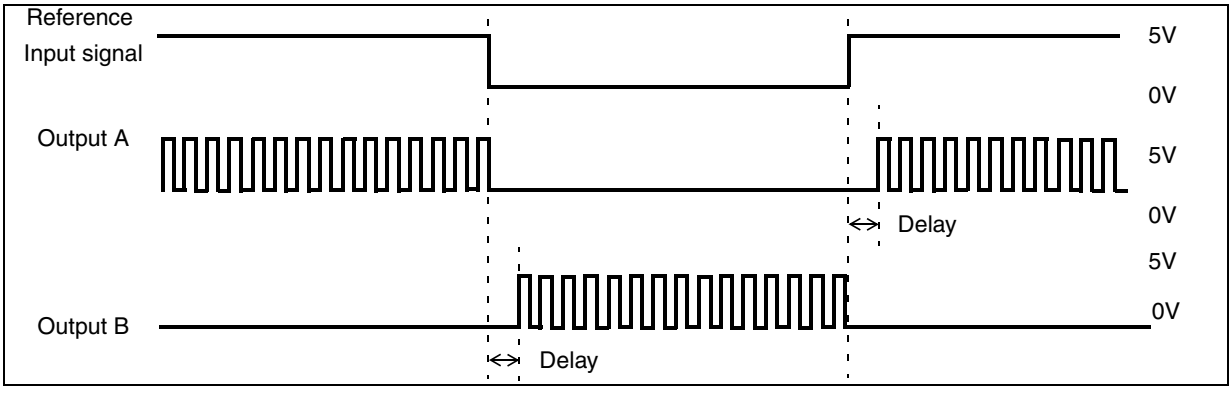
HFRQ2	HFRQ1	HFRQ0	Chopping frequency F_{mtc} = 16MHz F_{mtc} = 8MHz	Chopping frequency F_{mtc} = 4MHz
0	0	0	100 KHz	50 KHz
0	0	1	200 KHz	100 KHz
0	1	0	400 KHz	200 KHz
0	1	1	500 KHz	250 KHz
1	0	0	800 KHz	400 KHz
1	0	1	1 MHz	500 KHz
1	1	0	1.33 MHz	666.66 MHz
1	1	1	2 MHz	1 MHz

Note: When the PCN bit = 0:

– If complementary PWM signals are not applied (DTE bit = 0), the high and low drivers are fixed by the MPAR register. Figure 108, Figure 114 and Figure 115 indicate where the HFE[1:0] bits are taken into account depending on the PWM application.

– If complementary PWM signals are applied (DTE bit = 1), the channels are paired as explained in "Dead Time Generator" on page 194. This means that the high and low channels are fixed and the HFE[1:0] bits indicate where to apply the chopper. Figure 116 shows typical complementary PWM signals with high frequency chopping enabled on both high and low drivers.

Figure 116. Complementary PWM signals with chopping frequency on high and low side drivers



MOTOR CONTROLLER (Cont'd)

10.6.10 PWM Generator Block

The PWM generator block produces three independent PWM signals based on a single carrier frequency with individually adjustable duty cycles.

Depending on the motor driving method, one or three of these signals may be redirected to the other functional blocks of the motor control peripheral, using the PCN bit in the MDTG register.

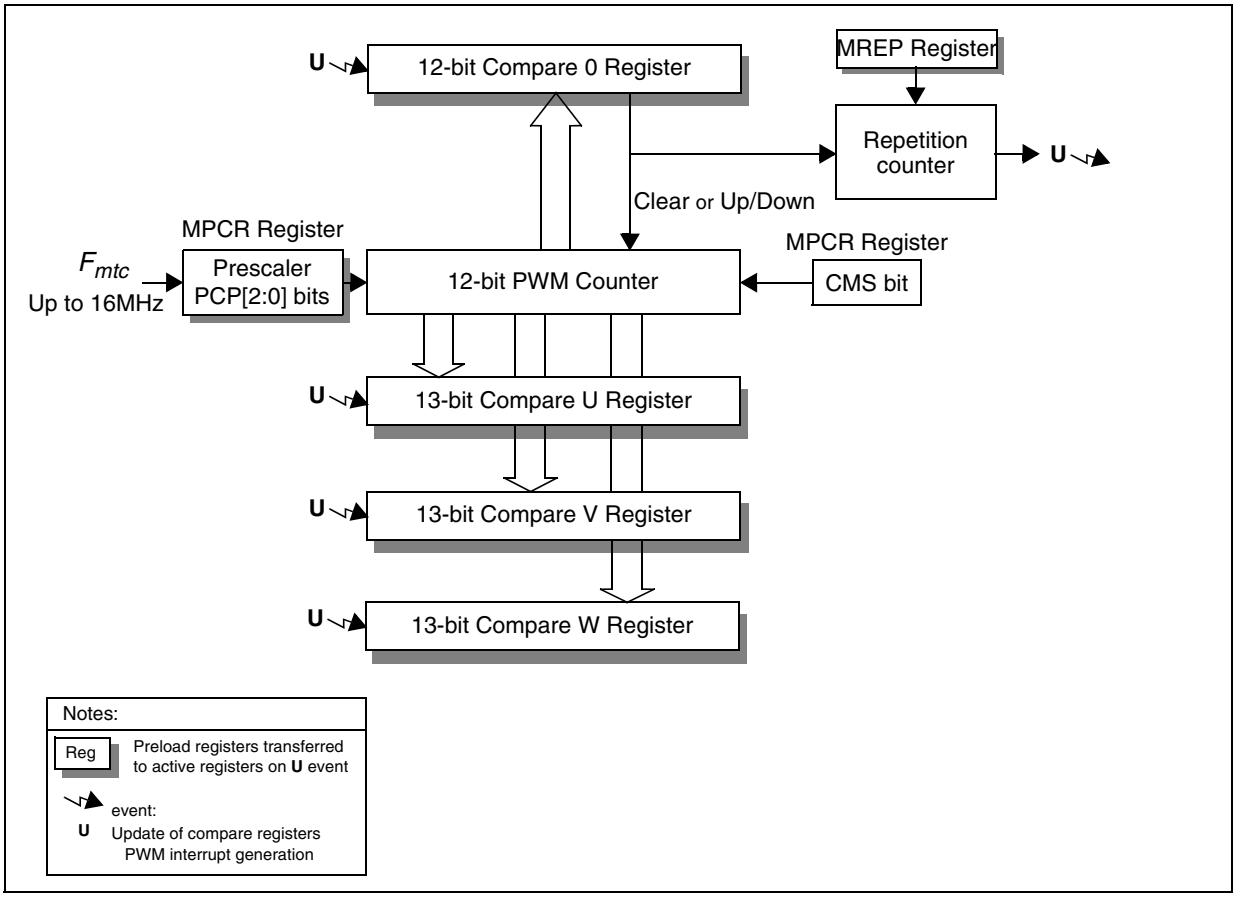
When driving PM BLDC motors in six-step mode (voltage mode only, either sensored or sensorless) a single PWM signal (Phase U) is used to supply the Input Stage, PWM and Channel Manager blocks according to the selected modes.

For other kind of motors requiring independent PWM control for each of the three phases, all PWM signals (Phases U, V and W) are directed to the channel manager, in which deadtime or a high frequency carrier may be added. This is the case of AC induction motors or PMAC motors for instance, supplied with 120° shifted sinewaves in voltage mode.

10.6.10.1 Main Features

- 12-bit PWM free-running Up/Down Counter with up to 16MHz input clock (F_{mtc}).
- Edge-aligned and center-aligned PWM operating modes
- Possibility to re-load compare registers twice per PWM period in center-aligned mode
- Full-scale PWM generation
- PWM update interrupt generation
- 8-bit repetition counter
- 8-bit PWM mode
- Timer re-synchronisation feature

Figure 117. PWM generator block diagram



MOTOR CONTROLLER (Cont'd)

10.6.10.2 Functional Description

The 3 PWM signals are generated using a free-running 12-bit PWM Counter and three 13-bit Compare registers for phase U, V and W: MCMPU, MCMPV and MCMPW registers.

A fourth 12-bit register is needed to set-up the PWM carrier frequency: MCMP0 register.

Each of these compare registers is buffered with a preload register. Transfer from preload to active registers is done synchronously with PWM counter underflow or overflow depending on configuration. This allows to write compare values without risks of spurious PWM transitions.

The block diagram of the PWM generator is shown on Figure 117.

10.6.10.3 Prescaler

The 12-bit PWM Counter clock is supplied through a 3-bit prescaler to allow the generation of lower PWM carrier frequencies. It divides F_{mtc} by 1, 2, 3, ..., 8 to get $F_{mtc\text{-}pwm}$.

This prescaler is accessed through three bits PCP[2:0] in MPCR register; this register is buffered: the new value is taken into account after a PWM update event.

10.6.10.4 PWM Operating mode

The PWM generator can work in center-aligned or edge-aligned mode depending on the CMS bit setting in the MPCR register.

Figure 118 shows the corresponding counting sequence .

It offers also an 8-bit mode to get a full 8-bit range with a single compare register write access by setting the PMS bit in MPCR register.

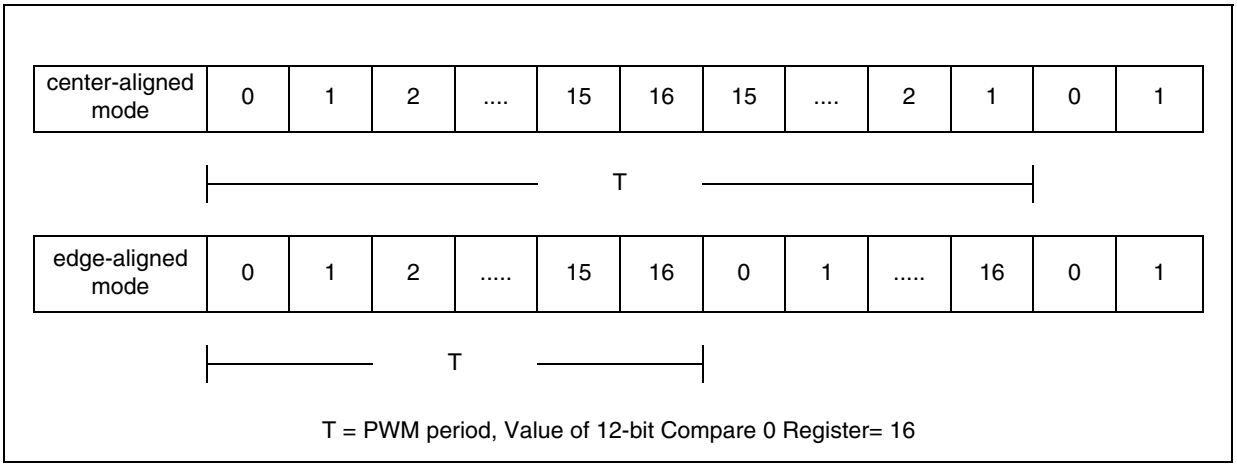
The comparisons described here are performed between the PWM Counter value extended to 13 bits and the 13-bit Compare register. Having a compare range greater than the counter range is mandatory to get a full PWM range (i.e. up to 100% modulation). This principle is maintained for 8-bit PWM operations.

- Center-aligned Mode (CMS bit = 1)

In this operating mode, the PWM Counter counts up to the value loaded in the 12-bit Compare 0 register then counts down until it reaches zero and restarts counting up.

The PWM signals are set to '0' when the PWM Counter reaches, in up-counting, the corresponding 13-bit Compare register value and they are set to '1' when the PWM Counter reaches the 13-bit Compare value again in down-counting.

Figure 118. Counting sequence in center-aligned and edge-aligned mode



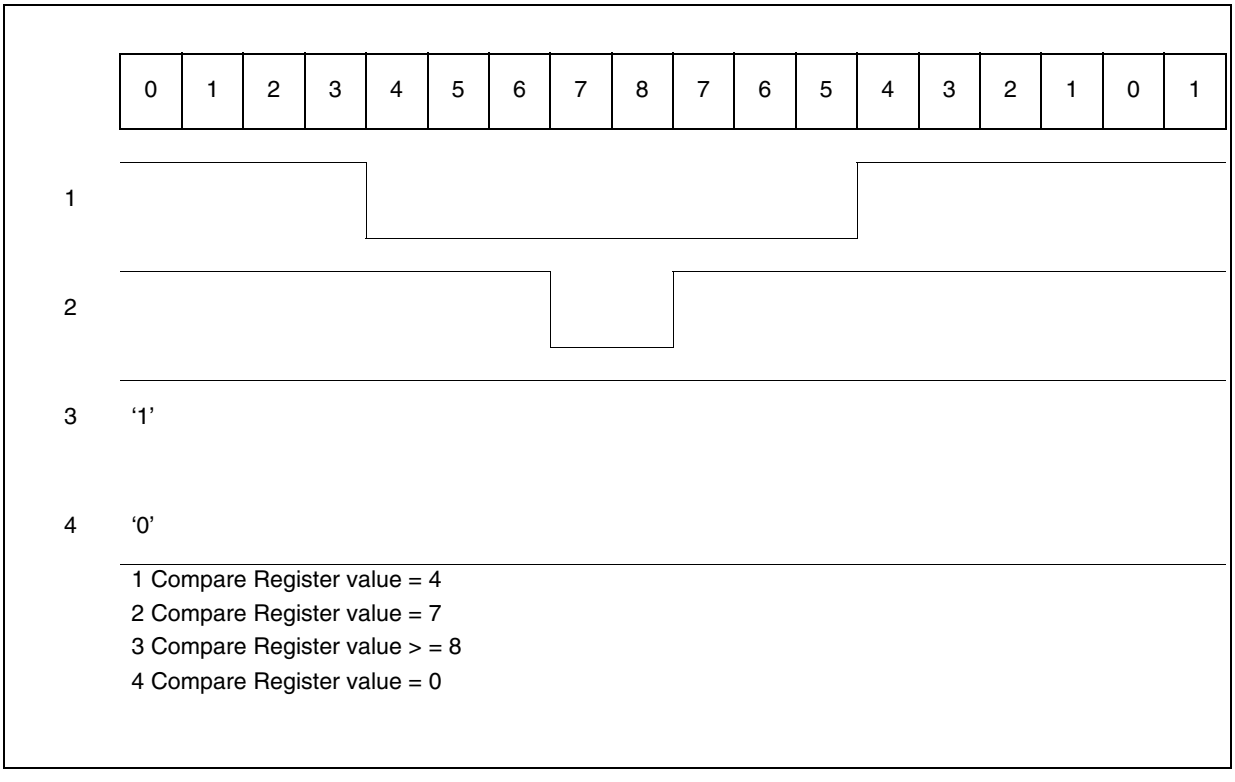
MOTOR CONTROLLER (Cont'd)

If the 13-bit Compare register value is greater than the extended Compare 0 Register (the 13th bit is set to '0'), the corresponding PWM output signal is held at '1'.

If the 13-bit Compare register value is 0, the corresponding PWM output signal is held at '0'.

Figure 119 shows some center-aligned PWM waveforms in an example where the Compare 0 register value = 8.

Figure 119. Center-aligned PWM Waveforms (Compare 0 Register = 8)



MOTOR CONTROLLER (Cont'd)

■ Edge-aligned Mode (CMS bit = 0)

In this operating mode, the PWM Counter counts up to the value loaded in the 12-bit Compare Register. Then the PWM Counter is cleared and it restarts counting up.

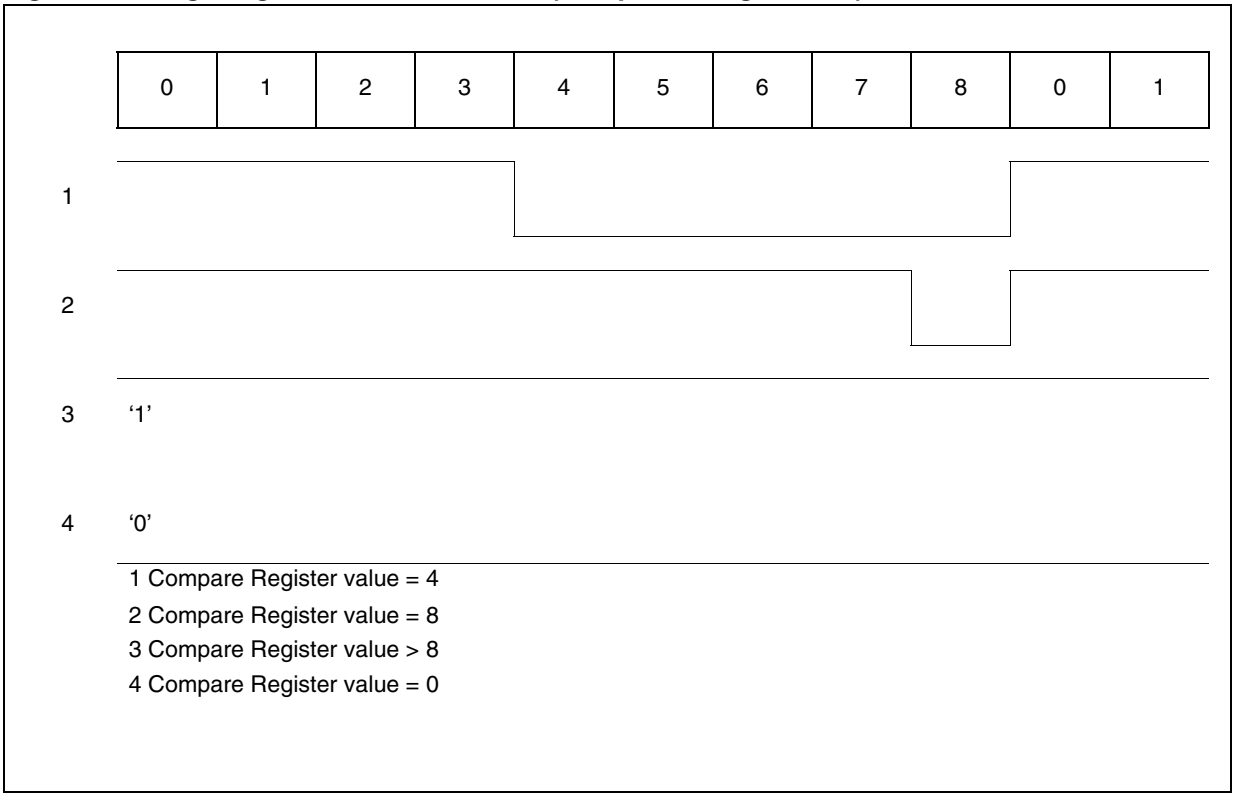
The PWM signals are set to '0' when the PWM Counter reaches, in up-counting, the corresponding 13-bit Compare register value and they are set to '1' when the PWM Counter is cleared.

If the 13-bit Compare register value is greater than the extended Compare 0 register (the 13th bit is set to '0'), the corresponding PWM output signal is held at '1'.

If the 13-bit Compare register value = 0, the corresponding PWM output signal is held at '0'.

Figure 120 shows some edge-aligned PWM waveforms in an example where the Compare 0 register value = 8.

Figure 120. Edge-aligned PWM Waveforms (Compare 0 Register = 8)



MOTOR CONTROLLER (Cont'd)

■ 12-bit Mode (PMS bit = 0 in the MPCR register)

This mode is useful for MCMP0 values ranging from 9 bits to 12 bits. Figure 121 presents the way Compare 0 and Compare U, V, W should be loaded). It requires loading two bytes in the MCMPxH and MCMPxL registers (i.e. MCMP0, MCMPU, MCMPV and MCMPW 16-bit registers) following the sequence described below:

– write to the MCMPxL register (LSB) first

– then write to the MCMPxH register (MSB).

The 16-bit value is then ready to be transferred in the active register as soon as an update event occurs. This sequence is necessary to avoid potential conflicts with update interrupts causing the hardware transfer from preload to active registers: if an update event occurs in the middle of the above sequence, the update is effective only when the MSB has been written.

■ 8-bit PWM mode (PMS bit = 1 in MPCR register)

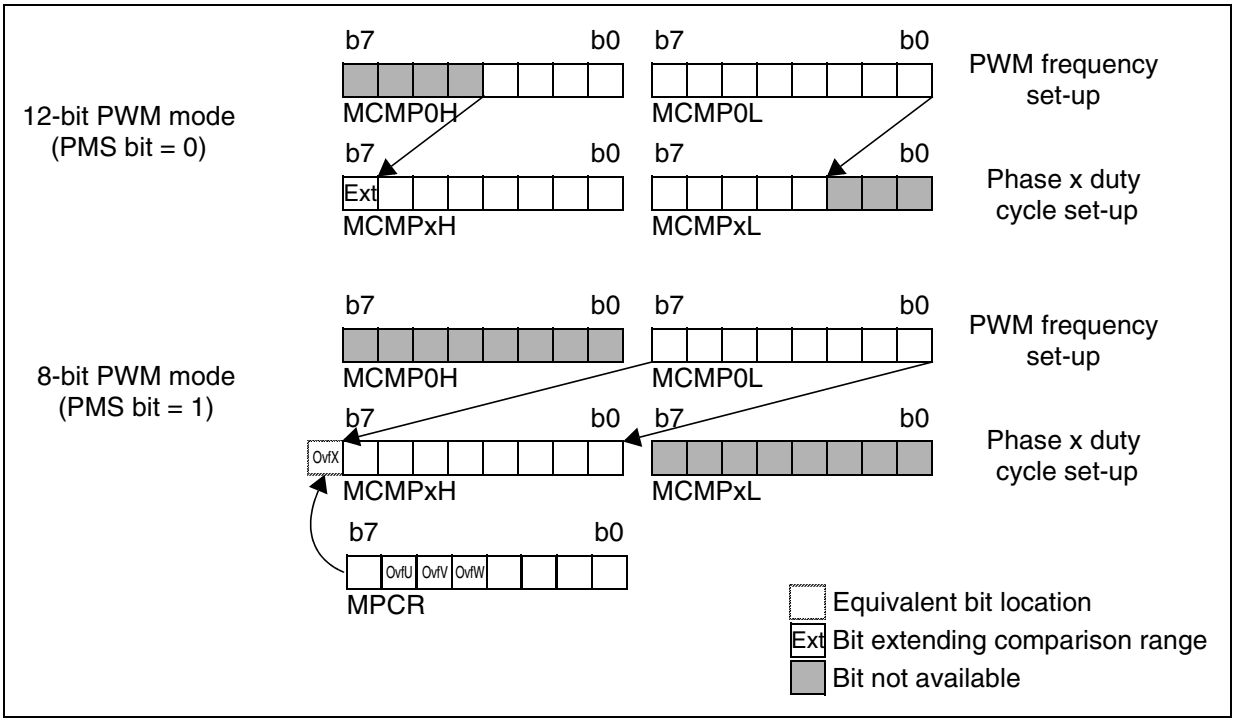
This mode is useful whenever the MCMP0 value is less or equal to 8-bits. It allows significant CPU resource savings when computing three-phase duty cycles during PWM interrupt routines. In this mode, the Compare 0 and Compare U, V, W registers have the same size (8 bits). The extension of the MCMPx registers is done in using the OVFx bits in the MPCR register (refer to Figure 121). These bits force the related duty-cycles to 100% and are reset by hardware on occurence of a PWM update event.

Note about read access to registers with preload: during read accesses, values read are the content of the preload registers, not the active registers.

Note about compare register active bit locations: the 13 active bits of the MCMPx registers are left-aligned. This allows temporary calculations to be done with 16-bit precision, round-up is done automatically to the 13-bit format when loading the values of the MCMPx registers.

Note about MCMP0x registers: the configuration MCMP0H=MCMP0L=0 is not allowed

Figure 121. Comparison between 12-bit and 8-bit PWM mode



MOTOR CONTROLLER (Cont'd)

10.6.10.5 Repetition Down-Counter

Both in center-aligned and edge-aligned modes, the four Compare registers (one Compare 0 and three for the U, V and W phases) are updated when the PWM counter underflow or overflow and the 8-bit Repetition down-counter has reached zero.

This means that data are transferred from the preload compare registers to the compare registers every N cycles of the PWM Counter, where N is the value of the 8-bit Repetition register in edge -aligned mode. When using center-aligned mode, the repetition down-counter is decremented every time the PWM counter overflows or underflows. Although this limits the maximum number of repetition to 128 PWM cycles, this makes it possible to update the duty cycle twice per PWM period. As a result, the effective PWM resolution in that case is equal to the resolution we can get using edge-aligned mode, i.e. one T_{mtc} period. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is $2x T_{mtc}$, due to the symmetry of the pattern.

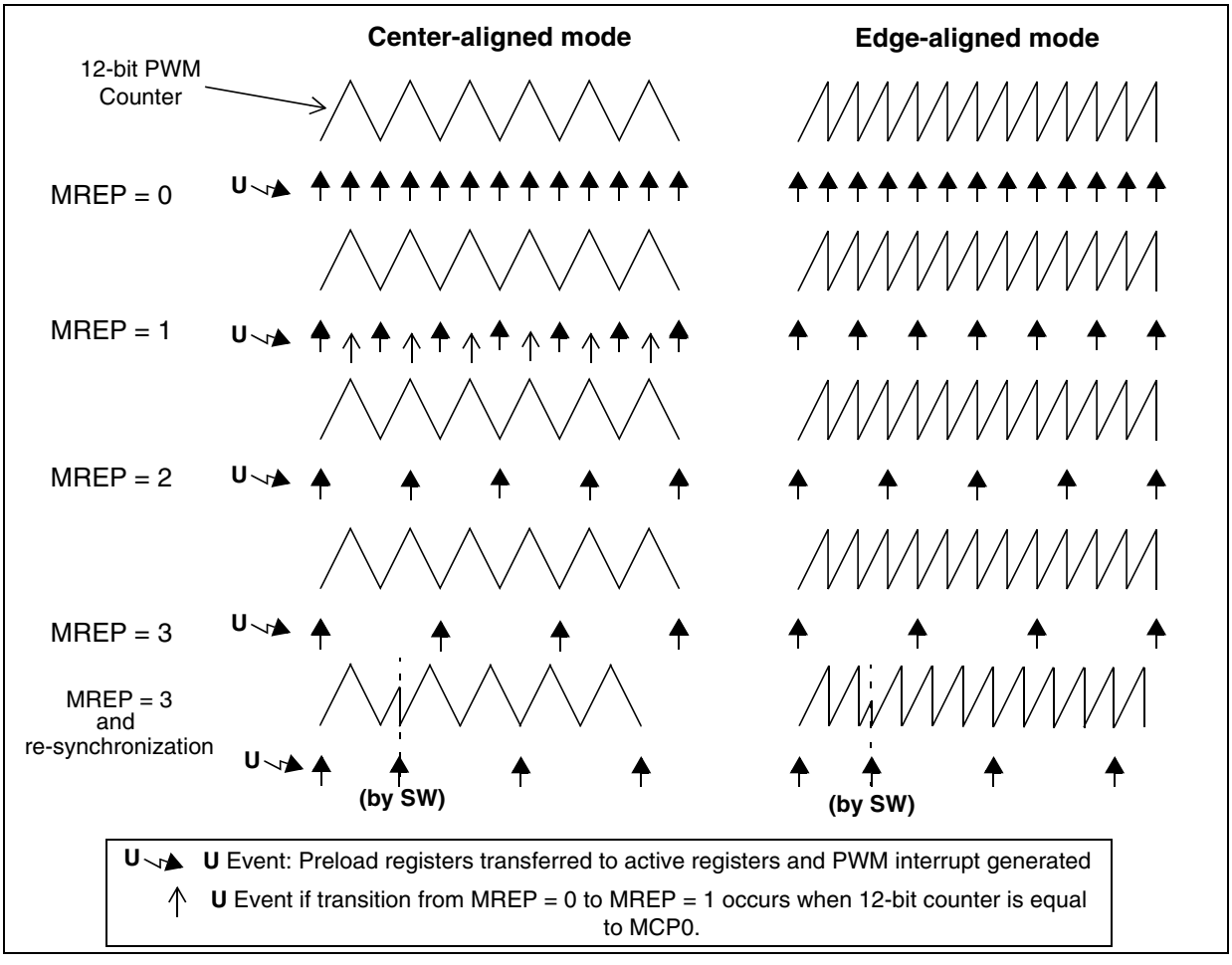
The repetition down counter is an auto-reload type; the repetition rate will be maintained as defined by the MREP register value (refer to Figure 122).

10.6.10.6 PWM interrupt generation

A PWM interrupt is generated synchronously with the "U" update event, which allows to refresh compare values by software before the next update event. As a result, the refresh rate for phases duty cycles is directly linked to MREP register setting.

A signal reflecting the update events may be output on a standard I/O port for debugging purposes. Refer to section10.6.7.3 on page 172 for more details.

Figure 122. Update rate examples depending on mode and MREP register settings



MOTOR CONTROLLER (Cont'd)

10.6.10.7 Timer Re-synchronisation

The 12-bit timer can be re-synchronized by a simple write access with FFh value in the MISR register. Re-synchronization means that the 12-bit counter is reset and all the compare preload registers MCP0, MCPU, MCPV, MCPW are transferred to the active registers.

To re-synchronize the 12-bit timer properly , the following procedure must be applied:

– 1. Load the new values in the preload compare registers

– 2. Load FFh value in the MISR register (this will reset the counter and transfer the compare preload registers in the active registers: U event)

– 3. Reset the PUI flag by loading 7Fh in the MISR register. Refer to Note 2 on page 209

Note: Loading FFh value in the MISR register will have no effect on any other flag than the PUI flag and will generate a PWM update interrupt if the PUM bit is set.

Warning: In switched mode (SWA bit is reset), the procedure is the same and loading FFh in the MISR register will have no effect on flags except on the PUI flag. As a consequence, it is recommended to avoid setting RMI and RPI flags at the same time in switched mode because none of them will be taken into account.

10.6.10.8 PWM generator initialization and start-up

The three-phase generator counter stays in reset state (i.e. stopped and equal to 0), as long as MTC peripheral clock is disabled (CKE = 0).

Setting the CKE bit has two actions on the PWM generator:

- It starts the PWM counter
- It forces the update of all registers with preload registers transferred on U update event, i.e. MREP, MPCR, MCMP0, MCMPU, MCMPV, MCMPW (in 12-bit mode, both MCMPxL and MCMPxH must have been written, following the mandatory LSB/MSB sequence, before setting CKE bit). It consequently generates a U interrupt.

10.6.11 Low Power Modes

Before executing a HALT or WFI instruction, software must stop the motor, and may choose to put the outputs in high impedance.

Mode	Description
Wait	No effect on MTC interface. MTC interrupts exit from Wait mode.
Halt	MTC registers are frozen. In Halt mode, the MTC interface is inactive. The MTC interface becomes operational again when the MCU is woken up by an interrupt with "exit from Halt mode" capability.

10.6.12 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
Ratio increment	RPI	RIM	Yes	No
Ratio decrement	RMI		Yes	No
Speed Error	SEI	SEM	Yes	No
Emergency Stop	EI	EIM	Yes	No
Current Limitation	CLI	CLIM	Yes	No
BEMF Zero-Crossing	ZI	ZIM	Yes	No
End of Demagnetization	DI	DIM	Yes	No
Commutation or Capture	CI	CIM	Yes	No
PWM Update	PUI	PUM	Yes	No
Sampling Out	SOI	SOM	Yes	Not

The MTC interrupt events are connected to the three interrupt vectors (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).

MOTOR CONTROLLER (Cont'd)

10.6.13 Register Description

TIMER COUNTER REGISTER (MTIM)

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

7	6	5	4	3	2	1	0
T7	T6	T5	T4	T3	T2	T1	T0

Bits 7:0 = **T[7:0]**: *MTIM Counter Value.*
These bits contain the current value of the 8-bit up counter. In Speed Measurement Mode, when using Encoder sensor and MTIM captures triggered by SW (refer to Figure 102) a read access to MTIM register causes a capture of the [MTIM:MTIML] register pair to the [MZREG: MZPRV] registers.

TIMER COUNTER REGISTER LSB (MTIML)

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

7	6	5	4	3	2	1	0
TL7	TL6	TL5	TL4	TL3	TL2	TL1	TL0

Bits 7:0 = **TL[7:0]**: *MTIM Counter Value LSB.*
These bits contain the current value of the least significant byte of the MTIM up counter, when used in Speed Measurement Mode (i.e. as a 16-bit timer)

CAPTURE Z_{n-1} REGISTER (MZPRV)

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

7	6	5	4	3	2	1	0
ZP7	ZP6	ZP5	ZP4	ZP3	ZP2	ZP1	ZP0

Bits 7:0 = **ZP[7:0]**: *Previous Z Value or Speed capture LSB.*
These bits contain the previous captured BEMF value (Z_{N-1}) in Switched and Autoswitched mode or the LSB of the captured value of the [MTIM:MTIML] registers in Speed Sensor Mode.

CAPTURE Z_n REGISTER (MZREG)

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

7	6	5	4	3	2	1	0
ZC7	ZC6	ZC5	ZC4	ZC3	ZC2	ZC1	ZC0

Bits 7:0 = **ZC[7:0]**: *Current Z Value or Speed capture MSB.*
These bits contain the current captured BEMF value (Z_N) in Switched and Autoswitched mode or the MSB of the captured value of the [MTIM:MTIML] registers in Speed Sensor Mode. A read access to MZREG in this case disable the Speed captures up to MZPRV reading (refer to Section 10.6.7.5 Speed Measurement Mode on page 180).

COMPARE C_{n+1} REGISTER (MCOMP)

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

7	6	5	4	3	2	1	0
DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0

Bits 7:0 = **DC[7:0]**: *Next Compare Value.*
These bits contain the compare value for the next commutation (C_{N+1}).

DEMAGNETIZATION REGISTER (MDREG)

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

7	6	5	4	3	2	1	0
DN7	DN6	DN5	DN4	DN3	DN2	DN1	DN0

Bits 7:0 = **DN[7:0]**: *D Value.*
These bits contain the compare value for simulated demagnetization (D_N) and the captured value for hardware demagnetization (D_H) in Switched and Autoswitched mode.

In Speed Sensor Mode, the register contains the value used for comparison with MTIM registers to generate a Speed Error event.

MOTOR CONTROLLER (Cont'd)

A_N WEIGHT REGISTER (MWGHT)

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

7	6	5	4	3	2	1	0
AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

Bits 7:0 = **AN[7:0]**: *A Weight Value.*
These bits contain the A_N weight value for the multiplier. In autoswitched mode the MCOMP register is automatically loaded with:

$$\frac{Z_n \times MWGHT}{256(d)} \quad \text{or} \quad \frac{Z_{N-1} \times MWGHT}{256(d)} \quad (*)$$

when a Z event occurs.
(*) depending on the DCB bit in the MCRA register.

PRESCALER & SAMPLING REGISTER (MPRSR)

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

7	6	5	4	3	2	1	0
SA3	SA2	SA1	SA0	ST3	ST2	ST1	ST0

Bits 7:4 = **SA[3:0]**: *Sampling Ratio.*
These bits contain the sampling ratio value for current mode. Refer to Table 47, "Sampling Frequency Selection," on page 189.

Bits 3:0 = **ST[3:0]**: *Step Ratio.*
These bits contain the step ratio value. It acts as a prescaler for the MTIM timer and is auto incremented/decremented with each R+ or R- event. Refer to Table 40, "Step Frequency/Period Range (4MHz)," on page 179 and Table 41, "Modes of Accessing MTIM Timer-Related Registers," on page 179.

INTERRUPT MASK REGISTER (MIMR)

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

7	6	5	4	3	2	1	0
PUM	SEM	RIM	CLIM	EIM	ZIM	DIM	CIM

Bit 7 = **PUM:** *PWM Update Mask bit.*
0: PWM Update interrupt disabled
1: PWM Update interrupt enabled

Bit 6 = **SEM:** Speed Error Mask bit.
0: Speed Error interrupt disabled
1: Speed Error interrupt enabled

Bit 5 = **RIM**: *Ratio update Interrupt Mask bit.*
0: Ratio update interrupts (R+ and R-) disabled
1: Ratio update interrupts (R+ and R-) enabled

Bit 4 = **CLIM**: *Current Limitation Interrupt Mask bit.*
0: Current Limitation interrupt disabled
1: Current Limitation interrupt enabled

This interrupt is available only in Voltage Mode (VOC1 bit=0 in MCRA register) and occurs when the Motor current feedback reaches the external current limitation value.

Bit 3 = **EIM**: *Emergency stop Interrupt Mask bit.*
0: Emergency stop interrupt disabled
1: Emergency stop interrupt enabled

Bit 2 = **ZIM**: *Back EMF Zero-crossing Interrupt Mask bit.*
0: BEMF Zero-crossing Interrupt disabled
1: BEMF Zero-crossing Interrupt enabled

Bit 1 = **DIM**: *End of Demagnetization Interrupt Mask bit.*
0: End of Demagnetization interrupt disabled
1: End of Demagnetization interrupt enabled if the HDM or SDM bit in the MCRB register is set

Bit 0 = **CIM**: *Commutation / Capture Interrupt Mask bit*
0: Commutation / Capture Interrupt disabled
1: Commutation / Capture Interrupt enabled

MOTOR CONTROLLER (Cont'd)

INTERRUPT STATUS REGISTER (MISR)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
PUI	RPI	RMI	CLI	EI	ZI	DI	CI

Bit 7 = **PUI:** *PWM Update Interrupt flag.*

This bit is set by hardware when all the PWM Compare register are transferred from the preload to the active registers. The corresponding interrupt allows the user to refresh the preload registers before the next PWM update event defined with MREP register.
0: No PWM Update interrupt pending
1: PWM Update Interrupt pending

Bit 6 = **RPI**: *Ratio Increment interrupt flag.*
Autoswitched mode (SWA bit =1)**:**
0: No R+ interrupt pending
1: R+ Interrupt pending

Switched mode (SWA bit =0)**:**
0: No R+ action
1: The hardware will increment the ST[3:0] bits when the next commutation occurs and shift all timer registers right.

Speed Sensor mode (SWA bit =x, TES[1:0] bits =01, 10, 11)**:**
0: No R+ interrupt pending
1: R+ Interrupt pending

Bit 5 = **RMI**: *Ratio Decrement interrupt flag.*
Autoswitched mode (SWA bit =1)**:**
0: No R- interrupt pending
1: R- Interrupt pending
Switched mode (SWA bit =0)**:**
0: No R- action
1: The hardware will decrement the ST[3:0] bits when the next commutation occurs and shift all timer registers left.

Speed Sensor mode (SWA bit =x, TES[1:0] bits =01, 10, 11)**:**
0: No R- interrupt pending
1: R- Interrupt pending

Bit 4 = **CLI**: *Current Limitation interrupt flag.*
0: No Current Limitation interrupt pending
1: Current Limitation interrupt pending

Bit 3 = **EI**: *Emergency stop Interrupt flag.*
0: No Emergency stop interrupt pending
1: Emergency stop interrupt pending

Bit 2 = **ZI**: *BEMF Zero-crossing interrupt flag.*
0: No BEMF Zero-crossing Interrupt pending
1: BEMF Zero-crossing Interrupt pending

Bit 1 = **DI**: *End of Demagnetization interrupt flag.*
0: No End of Demagnetization interrupt pending
1: End of Demagnetization interrupt pending

Bit 0 = **CI**: *Commutation / Capture interrupt flag*
0: No Commutation / Capture Interrupt pending
1: Commutation / Capture Interrupt pending

Note 1: Loading value FFh in the MISR register will reset the PWM generator counter and transfer the compare preload registers in the active registers by generating a U event (PUI bit set to 1). Refer to "Timer Re-synchronisation" on page 206.

Note 2: When several MTC interrupts are enabled at the same time the BRES instruction must not be used to avoid unwanted clearing of status flags: if a second interrupt occurs while BRES is executed (which performs a read-modify-write sequence) to clear the flag of a first interrupt, the flag of the second interrupt may also be cleared and the corresponding interrupt routine will not be serviced. It is thus recommended to use a load instruction to clear the flag, with a value equal to the logical complement of the bit. For instance, to clear the PUI flag:

ld MISR, # 0x7F.

Note 3: **In Autoswitched mode** (SWA=1 in the MRCA register): As all bits in the MISR register are status flags, they are set by internal hardware signals and must be cleared by software. Any attempt to write them to 1 will have no effect (they will be read as 0) without interrupt generation.

In Switched mode (SWA=0 in the MRCA register):

To avoid any losing any interrupts when modifying the RMI and RPI bits the following instruction sequence is recommended:

ld MISR, # 0x9F ; reset both RMI & RPI bits

ld MISR, # 0xBF ; set RMI bit

ld MISR, # 0xDF ; set RPI bit

MOTOR CONTROLLER (Cont'd)

CONTROL REGISTER A (MCRA)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
MOE	CKE	SR	DAC	V0C1	SWA	PZ	DCB

Bit 7 = **MOE:** *Output Enable bit.*
0: Outputs disabled
1: Outputs enabled

MOE bit	MCO[5:0] Output pin State
0	Reset state
1	Output enabled

Notes:

– The reset state is either high impedance, high or low state depending on the corresponding option bit.

– When the MOE bit in the MCRA register is reset (MCOx outputs in reset state), and the SR bit in the MCRA register is reset (sensorless mode) and the SPLG bit in the MCRC register is reset (sampling at PWM frequency) then, depending on the state of the ZSV bit in the MSCR register, Z event sampling can run or be stopped (and D event is sampled).

Bit 6 = **CKE**: *Clock Enable Bit.*
0: Motor Control peripheral Clocks disabled
1: Motor Control peripheral Clocks enabled

Note: Clocks disabled means that all peripheral internal clocks (Delay manager, internal sampling clock, PWM generator) are disabled. Therefore, the peripheral can no longer detect events and the preload registers do not operate.

When Clocks are disabled, write accesses are allowed, so for example, MTIM counter register can be reset by software.

Table 56. Output configuration summary

CKE bit	MOE bit	DAC bit	Peripheral Clock	Effect on MCOx Output
0	0	x	Disabled	Reset state
0	1	0	Disabled	Peripheral frozen (see note 1 below)
0	1	1	Disabled	Direct access via MPHST (only logical level)
1	0	x	Enabled	Reset state
1	1	0	Enabled	Standard running mode.
1	1	1	Enabled	Direct access via MPHST (PWM can be applied)

Note 1: "Peripheral frozen" configuration is not recommended, as the peripheral may be stopped in a unknown state (depending on PWM generator outputs,etc.). It is better practice to exit from run mode by first setting output state (by toggling either MOE or DAC bits) and then to disabling the clock if needed.

Note 2: In Direct Access Mode (DAC=1), when CKE=0 (Peripheral Clock disabled) only logical level can be applied on the MCOx outputs when they are enabled whereas when CKE=1 (Peripheral Clock enabled), a PWM signal can be applied on them. Refer to Table 74, "DeadTime generator set-up," on page 221

Note 3: When clocks are disabled (CKE bit reset) while outputs are enabled (MOE bit set), the effects on the MCOx outputs where PWM signal is applied depend on the running mode selected:

– in voltage mode (VOC1 bit=0), the MCOx outputs where PWM signal is applied stay at level 1.

– in current mode (VOC1 bit=1), the MCOx outputs where PWM signal is applied are put to level 0.

In all cases, MCOx outputs where a level 1 was applied before disabling the clocks stay at level 1. That is why it is recommended to disable the MCOx outputs (reset MOE bit) before disabling the clocks. This will put all the MCOx outputs under reset state defined by the corresponding option bit.

MOTOR CONTROLLER (Cont'd)

Effect on PWM generator: the PWM generator 12-bit counter is reset as soon as CKE = 0; this ensures that the PWM signals start properly in all cases. When these bits are set, all registers with preload on Update event are transferred to active registers.

Bit 5 = **SR**: *Sensor ON/OFF.*
0: Sensorless mode
1: Position Sensor mode

Table 57. Sensor Mode Selection

SR bit	Mode	OS[2:0] bits	Behaviour of the output PWM
0	Sensors not used	OS[2:0] bits enabled	"Between C_n&D" behaviour, "between D&Z" behaviour and "between Z&C_{n+1}" behaviour
1	Sensors used	OS1 disabled	"Between C_n&Z" behaviour and "between Z&C_{n+1}" behaviour

See also Table 61 and Table 62

Bit 4 = **DAC**: *Direct Access to phase state register.*
0: No Direct Access (reset value). In this mode the preload value of the MPHST and MCRB registers is taken into account at the C event.
1: Direct Access enabled. In this mode, write a value in the MPHST register to access the outputs directly.

Note: In Direct Access Mode (DAC bit is set in MCRA register), a C event is generated as soon as there is a write access to the OO[5:0] bits in MPHST register. In this case, the PWM low/high selection is done by the OS0 bit in the MCRB register.

Table 58. DAC Bit Meaning

MOE bit	DAC bit	Effect on Output
0	x	Reset state depending on the option bit
1	0	Standard running mode.
1	1	MPHST register value (depending on MPOL, MPAR register values and PWM setting) see Table 74

Bit 3 = **V0C1**: *Voltage/Current Mode*
0: Voltage Mode
1: Current Mode

Bit 2 = **SWA**: *Switched/Autoswitched Mode*
0: Switched Mode
1: Autoswitched Mode

Note 1 : after reset, in autoswitched mode (SWA =1) , the motor control peripheral is waiting for a C commutation event.

Note 2: After reset, a C event is immediately generated when CKE and SWA are simultaneaously set due to a nil value of MCOMP.

Bit 1 = **PZ**: *Protection from parasitic Zero-crossing event detection*
0: Protection disabled
1: Protection enabled

Note: If the PZ bit is set, the Z event filter (ZEF[3:0] in the MZFR register is ignored.

Bit 0 = **DCB**: *Data Capture bit*
0: Use MZPRV (Z_N-1) for multiplication
1: Use MZREG (Z_N) for multiplication

Table 59. Multiplier Result

DCB bit	Commutation Delay
0	MCOMP = MWGHT x MZPRV / 256
1	MCOMP = MWGHT x MZREG / 256

MOTOR CONTROLLER (Cont'd)

CONTROL REGISTER B (MCRB)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
0	CPB*	HDM*	SDM*	OCV	OS2*	OS1	OS0

Bit 7= Reserved, must be kept at reset value.

Bit 6= **CPB***: *Compare Bit for Zero-crossing detection.*
0: Zero crossing detection on falling edge
1: Zero crossing detection on rising edge

Bit 5= **HDM***: *Hardware Demagnetization event Mask bit*
0: Hardware Demagnetization disabled
1: Hardware Demagnetization enabled

Bit 4= **SDM***: *Simulated Demagnetization event Mask bit*
0: Simulated Demagnetization disabled
1: Simulated Demagnetization enabled

Bit 3 = **OCV**: Over Current Handling in Voltage mode
0: Over Current protection is OFF
1:Over current protection is ON

This bit acts as follows

Table 60. Over current handling

CLIM bit	CLI bit	OCV bit	Output effect	Interrupt
0	0	x	Normal running mode	No
0	1	x	PWM is put off as Current loop effect	No
1	0	x	Normal running mode	No
1	1	0	PWM is put off as Current loop effect	Yes
1	1	1	All MCOx outputs are put in reset state (MOE reset) [1)]	Yes

Note 1: This feature is also available when using the three PWM outputs (PCN bit=1 in the MDTG register), providing that the VOC1bit = 0 (MCRA register). See section 10.6.8.2 on page 186

Bits 2:0 = **OS2*, OS[1:0]**: *Operating output mode Selection bits*

Refer to the Step behaviour diagrams (Figure 109, Figure 110) and Table 61, "Step Behaviour/ sensorless mode," on page 212.

These bits are used to define the various PWM output configurations.

Note: OS2 is the only preload bit.

Table 61. Step Behaviour/ sensorless mode

OS2 bit	PWM after C and before D	OS1 bit	PWM after D and before Z	OS0	PWM after Z and before next C
0	On High Channels	0	On High Channels	0	On high channels
				1	On low channels
		1	On Low Channels	0	On high channels
				1	On low channels
1	On Low Channels	0	On High Channels	0	On high channels
				1	On low channels
		1	On Low Channels	0	On high channels
				1	On low channels

Note: For more details, see Step behaviour diagrams (Figure 109 and Figure 110).

* Preload bits, new value taken into account at the next C event. A C event is generated at each write to MPHST in Direct Access mode.

MOTOR CONTROLLER (Cont'd)

Table 62. PWM mode when SR=1

OS2 bit	PWM after C and before Z	OS1 bit	Unused	OS0	PWM after Z and before next C
0	On High Channels	x	x	0	On high channels
				1	On low channels
1	On Low Channels	x	x	0	On high channels
				1	On low channels

Table 63. PWM mode when DAC=1

OS2 bit	Unused	OS1 bit	Unused	OS0	PWM on outputs
x	x	x	x	0	On high channels
				1	On low channels

Warning: As the MCRB register contains preload bits with, it has to be written as a complete byte. A Bit Set or Bit Reset instruction on a non-preload bit will have the effect of resetting all the preload bits.

CONTROL REGISTER C (MCRC)
Read/Write (except EDIR bit)
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SEI / OI	EDIR/ HZ	SZ	SC	SPLG	VR2	VR1	VR0

Bit 7= **SEI/OI**: *Speed Error interrupt flag / MTIM Overflow flag*

Position Sensor or Sensorless mode (TES[1:0] bits =00)**:**

OI: *MTIM Overflow flag*
This flag signals an overflow of the MTIM timer. It has to be cleared by software.
0: No MTIM timer overflow
1: MTIM timer overflow

Note: No interrupt is associated with this flag

Speed Sensor mode (TES[1:0] bits =01, 10, 11)**:**

SEI: *Speed error interrupt flag*
0: No Tacho Error interrupt pending
1: Tacho Error interrupt pending

Bit 6= **EDIR/HZ** : *Encoder Direction bit/ Hardware zero-crossing event bit*

Position Sensor or Sensorless mode (TES[1:0] bits =00)**:**

HZ: *Hardware zero-crossing event bit*
This Read/Write bit selects if the Z event is hardware or not.
0: No hardware zero-crossing event
1: Hardware zero-crossing event

Speed Sensor mode (TES[1:0] bits =01, 10, 11)**:**

EDIR:*Encoder Direction bit*
This bit is Read only. As the rotation direction depends on encoder outputs and motor phase connections, this bit cannot indicate absolute direction. It therefore gives the relative phase-shift (i.e. advance/delay) between the two signals in quadrature output by the encoder (see Figure 90).
0: MCIA input delayed compared to MCIB input.
1: MCIA input in advance compared to MCIB input

Bit 5 = **SZ**: *Simulated zero-crossing event bit*
0: No simulated zero-crossing event
1: Simulated zero-crossing event

Bit 4 = **SC**: *Simulated commutation event bit*
0: Hardware commutation event in auto-switched mode (SWA = 1 in MCRA register)
1: Simulated commutation event in auto-switched mode (SWA = 1 in MCRA register).

Bit 3 = **SPLG**: *Sampling Z event at high frequency in sensorless mode (SR=0)*

This bit enables sampling at high frequency in sensorless mode independently of the PWM signal or only during ON time if the DS[3:0] bits in the MCONF register contain a value. Refer to Table 77, "Sampling Delay," on page 224
0: Normal mode (Z sampling at PWM frequency at the end of the off time)
1: Z event sampled at f_{SCF} (see Table 82)

Note: When the SPLG bit is set, there is no minimum OFF time programmed by the OT [3:0] bits, the OFF time is forced to 0µs. This means that in current mode, the OFF time of the PWM signal will come only from the current loop.

MOTOR CONTROLLER (Cont'd)

Bits 2:0 = **VR[2:0]**: *BEMF/demagnetisation Reference threshold*

These bits select the Vref value as shown in the Table 64. The Vref value is used for BEMF and Demagnetisation detection.

Table 64. Threshold voltage setting

VR2	VR1	VR0	Vref voltage threshold
1	1	1	Threshold voltage set by external MCVREF pin
1	1	0	3.5V*
1	0	1	2.5V*
1	0	0	2V*
0	1	1	1.5V*
0	1	0	1V*
0	0	1	0.6V*
0	0	0	0.2V*

*Typical values for V_{DD}=5V

PHASE STATE REGISTER (MPHST)

Read/Write

Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
IS1*	IS0*	OO5*	OO4*	OO3*	OO2*	OO1*	OO0*

Bit 7:6 = **IS[1:0]***: *Input Selection bits*

These bits mainly select the input to connect to comparator as shown in Table 65. The fourth configuration (IS[1:0] = 11) specifies that an incremental encoder is used (in that case MCIA and MCIB digital signals are directly connected to the incremental encoder interface and the analog multiplexer is bypassed.

Table 65. Input Channel Selection

IS1	IS0	Channel selected
0	0	MCIA
0	1	MCIB
1	0	MCIC
1	1	Both MCIA and MCIB: Encoder Mode

Bits 5:0 =**OO[5:0]***: *Channel On/Off bits*

These bits are used to switch channels on/off at the next C event if the DAC bit =0 or directly if DAC=1

0: Channel Off, the relevant switch is OFF, no PWM possible

1: Channel On the relevant switch is ON, PWM is possible (not signifiant when PCN or DTE bit is set).

Table 66. OO[5:0] Bit Meaning

OO[5:0]	Output Channel State
0	Inactive
1	Active

* Preload bits, new value taken into account at next C event.

Caution: As the MPHST register contains bits with preload, the whole register has to be written at once. This means that a Bit Set or Bit Reset instruction on only one bit without preload will have the effect of resetting all the bits with preload.

MOTOR CONTROLLER (Cont'd)

MOTOR CURRENT FEEDBACK REGISTER (MCFR)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
RPGS	RST	CFF2	CFF1	CFF0	CFW2	CFW1	CFW0

Bit 7= **RPGS:** *Register Page Selection:*
0: Access to registers mapped in page 0
1: Access to registers mapped in page 1

Bit 6= **RST**: *Reset MTC registers.*
Software can set this bit to reset all MTC registers without resetting the ST7.
0: No MTC register reset
1: Reset all MTC registers

Bits 5:3 = **CFF[2:0]**: *Current Feedback Filter bits*
These bits select the number of consecutive valid samples (when the current is above the limit) needed to generate the active event. Sampling is done at $f_{PERIPH}/4$.

Table 67. Current Feedback Filter Setting

CFF2	CFF1	CFF0	Current Feedback Samples
0	0	0	1
0	0	1	2
0	1	0	3
0	1	1	4
1	0	0	5
1	0	1	6
1	1	0	7
1	1	1	8

Bits 2:0 = **CFW[2:0]**: *Current Window Filter bits:*

These bits select the length of the blanking window activated each time PWM is turned ON. The filter blanks the output of the current comparator.

Table 68. Current Feedback Window Setting

CFW2	CFW1	CFW0	Blanking Window
0	0	0	Blanking window off
0	0	1	0.5µs
0	1	0	1µs
0	1	1	1.5µs
1	0	0	2µs
1	0	1	2.5µs
1	1	0	3µs
1	1	1	3.5µs

Note: Times are indicated for 4 MHz f_{PERIPH}

MOTOR CONTROLLER (Cont'd)

MOTOR D EVENT FILTER REGISTER (MDFR)
Read/Write
Reset Value: 0000 1111 (0Fh)

7	6	5	4	3	2	1	0
DEF3	DEF2	DEF1	DEF0	DWF3	DWF2	DWF1	DWF0

Bits 7:4 = **DEF[3:0]**: *D Event Filter bits*
These bits select the number of valid consecutive D events (when the D event is detected) needed to generate the active event. Sampling is done at the selected f_{SCF} frequency, see Table 82.

Table 69. D Event filter Setting

DEF3	DEF2	DEF1	DEF0	D event Samples	SR=1
0	0	0	0	1	No D Event Filter
0	0	0	1	2	
0	0	1	0	3	
0	0	1	1	4	
0	1	0	0	5	
0	1	0	1	6	
0	1	1	0	7	
0	1	1	1	8	
1	0	0	0	9	
1	0	0	1	10	
1	0	1	0	11	
1	0	1	1	12	
1	1	0	0	13	
1	1	0	1	14	
1	1	1	0	15	
1	1	1	1	16	

Bit 3:0 = **DWF[3:0]**: *D Window Filter bits*
These bits select the length of the blanking window activated at each C event. The filter blanks the D event detection.

Table 70. D Window Filter setting

DWF3	DWF2	DWF1	DWF0	C to D Window Filter in Sensorless mode (SR=0)	SR=1
0	0	0	0	5µs	No window filter after C event
0	0	0	1	10µs	
0	0	1	0	15µs	
0	0	1	1	20µs	
0	1	0	0	25µs	
0	1	0	1	30µs	
0	1	1	0	35µs	
0	1	1	1	40µs	
1	0	0	0	60µs	
1	0	0	1	80µs	
1	0	1	0	100µs	
1	0	1	1	120µs	
1	1	0	0	140µs	
1	1	0	1	160µs	
1	1	1	0	180µs	
1	1	1	1	200µs	

Note: Times are indicated for 4 MHz f_{PERIPH}

MOTOR CONTROLLER (Cont'd)

REFERENCE REGISTER (MREF)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
HST	CL	CFAV	HFE1	HFE0	HFRQ2	HFRQ1	HFRQ0

Bit 7 = **HST**: *Hysteresis Comparator Value.*
This read only bit contains the hysteresis comparator output.
0: Demagnetisation/BEMF comparator is under V_{REF}
1: Demagnetisation/BEMF comparator is above V_{REF}

Bit 6 = **CL**: *Current Loop Comparator Value.*
This read only bit contains the current loop comparator output value.
0: Current detect voltage is under V_{CREF}
1: Current detect voltage is above V_{CREF}

Bit 5= **CFAV***: Current Feedback Amplifier entry Validation*
0: OAZ(MCCFI1) is the current comparator entry
1: MCCFI0 is the current comparator entry

Bits 4:3 = **HFE[1:0]**: *Chopping mode selection*

These bits select the chopping mode as shown in the following table.

Table 71. Chopping mode

HFE1	HFE0	Chopping mode
0	0	OFF
0	1	On Low channels only
1	0	On High channels only
1	1	Both High and Low channels

Bits 2:0 = **HFRQ[2:0]** : *Chopper frequency selection*

These bits select the chopping frequency.

Table 72. Chopping frequency selection

HFRQ2	HFRQ1	HFRQ0	Chopping frequency F_{mtc} = 16MHz F_{mtc} = 8MHz	Chopping frequency F_{mtc} = 4MHz
0	0	0	100 KHz	50 KHz
0	0	1	200 KHz	100 KHz
0	1	0	400 KHz	200 KHz
0	1	1	500 KHz	250 KHz
1	0	0	800 KHz	400 KHz
1	0	1	1 MHz	500 KHz
1	1	0	1.33 MHz	666.66 MHz
1	1	1	2 MHz	1 MHz

Note: The chopper signal has a 50% duty cycle.

MOTOR CONTROLLER (Cont'd)

PWM CONTROL REGISTER (MPCR)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
PMS	OVFU	OVFV	OVFW	CMS	PCP2	PCP1	PCP0

Bit 7 = **PMS:** *PWM Mode Selection*.
0: Standard mode: bit b7 in the MCPxH register represents the extension bit.
1: "8-bit" mode: bit b7 (extension bit) in the MCPxH register is located in the MPCR register (OVFx bits); the number of active bits in MCPxH and MCPxL is decreased to b15:b8 instead of b15:b3.

Bit 6 = **OVFU:** *Phase U 100% duty cycle Selection.*
0: Duty cycle defined by MCPUH:MCPUL register.
1: Duty cycle set at 100% on phase U at next update event and maintained till the next one. This bit is reset once transferred to the active register on update event.

Bit 5 = **OVFV:** *Phase V 100% duty cycle Selection.*
0: Duty cycle defined by MCPVH:MCPVL register.
1: Duty cycle set at 100% on phase V at next update event and maintained till the next one. This bit is reset once transferred to the active register on update event.

Bit 4 = **OVFW:** *Phase W 100% duty cycle Selection.*
0: Duty cycle defined by MCPWH:MCPWL register.
1: Duty cycle set at 100% on phase W at next update event and maintained till the next one. This bit is reset once transferred to the active register on update event.

Bit 3 = **CMS:** *PWM Counter Mode Selection.*
0: Edge-aligned mode
1: Center-aligned mode

Bits 2:0 = **PCP[2:0]** *PWM counter prescaler value*.

This value divides the F_{mtc} frequency by N, where N is PCP[2:0] value. Table 73 shows the resulting frequency of the PWM counter input clock.

Table 73. PWM clock prescaler

PCP2	PCP1	PCP0	PWM counter input clock
0	0	0	F_{mtc}
0	0	1	$F_{mtc}/2$
0	1	0	$F_{mtc}/3$
0	1	1	$F_{mtc}/4$
1	0	0	$F_{mtc}/5$
1	0	1	$F_{mtc}/6$
1	1	0	$F_{mtc}/7$
1	1	1	$F_{mtc}/8$

MOTOR CONTROLLER (Cont'd)

REPETITION COUNTER REGISTER (MREP)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
REP7	REP6	REP5	REP4	REP3	REP2	REP1	REP0

Bits 7:0 = **REP[7:0]** *Repetition counter value (N).*

This register allows the user to set-up the update rate of the PWM counter compare register (i.e. periodic transfers from preload to active registers), as well as the PWM Update interrupt generation rate, if these interrupts are enabled.

Each time the MREP related Down-Counter reaches zero, the Compare registers are updated, a U interrupt is generated and it re-starts counting from the MREP value.

After a microcontroller reset, setting the CKE bit in the MCRA register (i.e. enabling the clock for the MTC peripheral) forces the transfer from the MREP preload register to its active register and generates a U interrupt. During run-time (while CKE bit = 1) a new value entered in the MREP preload register is taken into account after a U event.

As shown in Figure 122, (N+1) value corresponds to:

– The number of PWM periods in edge-aligned mode

– The number of half PWM periods in center-aligned mode.

–

COMPARE PHASE W PRELOAD REGISTER HIGH (MCPWH)
Read/Write
Reset Value: 0000 0000 (00h)

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

Bits 7:0 = **CPWH[7:0]** *Most Significant Byte of phase W preload value*

COMPARE PHASE W PRELOAD REGISTER LOW (MCPWL)
Read/Write (except bits 2:0)
Reset Value: 0000 0000 (00h)

7							0
CPWL 7	CPWL 6	CPWL 5	CPWL 4	CPWL 3	-	-	-

Bits 7:5 = **CPWL[7:3]** *Low bits of phase W preload value*.

Bits 2:0 = Reserved.

COMPARE PHASE V PRELOAD REGISTER HIGH (MCPVH)
Read/Write
Reset Value: 0000 0000 (00h)

7							0
CPVH7	CPVH6	CPVH5	CPVH4	CPVH3	CPVH2	CPVH1	CPVH0

Bit 7:0 = **CPVH[7:0]** *Most Significant Byte of phase V preload value*

COMPARE PHASE V PRELOAD REGISTER LOW (MCPVL)
Read/Write (except bits 2:0)
Reset Value: 0000 0000 (00h)

7							0
CPVL7	CPVL6	CPVL5	CPVL4	CPVL3	-	-	-

Bits 7:5 = **CPVL[7:3]** *Low bits of phase V preload value*.

Bits 2:0 = Reserved.

MOTOR CONTROLLER (Cont'd)

COMPARE PHASE U PRELOAD REGISTER HIGH (MCPUH)
Read/Write
Reset Value: 0000 0000 (00h)

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

Bits 7:0 = **CPUH[7:0]** *Most Significant Byte of phase U preload value*

COMPARE PHASE U PRELOAD REGISTER LOW (MCPUL)
Read/Write Read/Write (except bits 2:0)
Reset Value: 0000 0000 (00h)

7							0
CPUL7	CPUL6	CPUL5	CPUL4	CPUL3	-	-	-

Bits 7:5 = **CPUL[7:3]** *Low bits of phase U preload value*.

Bits 2:0 = Reserved.

COMPARE 0 PRELOAD REGISTER HIGH (MCP0H)
Read/Write (except bits 7:4)
Reset Value: 0000 1111 (0Fh)

7							0
-	-	-	-	CP0H3	CP0H2	CP0H1	CP0H0

Bits 7:4 = Reserved.

Bits 3:0 = **CP0H[3:0]** *Most Significant Bits of Compare 0 preload value.*

COMPARE 0 PRELOAD REGISTER LOW (MCP0L)
Read/Write
Reset Value: 1111 1111 (FFh)

7							0
CP0L7	CP0L6	CP0L5	CP0L4	CP0L3	CP0L2	CP0L1	CP0L0

Bits 7:0 = **CP0L[7:0]** *Low byte of Compare 0 preload value*.

Note 1: The 16-bit Compare registers MCMPOx, MCMPUx, MCMPVx, MCMPWx MSB and LSB parts have to be written sequentially before being taken into account when an update event occurs; refer to section 10.6.10.4 on page 201 for details.

Note 2: The CPB, HDM, SDM, OS2 bits in the MCRB and the bits OE[5:0] are marked with *. It means that these bits are taken into account at the following commutation event (in normal mode) or when a value is written in the MPHST register when in direct access mode. For more details, refer to the description of the DAC bit in the MCRA register. The use of a Preload register allows all the registers to be updated at the same time.

Warning: Access to Preload registers

Special care has to be taken with Preload registers, especially when using the ST7 BSET and BRES instructions on MTC registers.

For instance, while writing to the MPHST register, you will write the value in the preload register. However, while reading at the same address, you will get the current value in the register and not the value of the preload register.

Excepted for three-phase PWM generator's registers, all preload registers are loaded in the active registers at the same time. In normal mode this is done automatically when a C event occurs, however in direct access mode (DAC bit=1) the preload registers are loaded as soon as a value is written in the MPHST register.

Caution: Access to write-once bits

Special care has to be taken with write-once bits in MPOL and MDTG registers; these bits have to be accessed first during the set-up. Any access to the other bits (not write-once) through a BRES or a BSET instruction will lock the content of write-once bits (no possibility for the core do distinguish individual bit access: Read/write internal signal acts on a whole register only). This protection is then only unlocked after a processor hardware reset.

MOTOR CONTROLLER (Cont'd)

DEAD TIME GENERATOR REGISTER (MDTG)
Read/Write (except bits 5:0 write once-only)
Reset Value: 1111 1111 (FFh)

7							0
PCN	DTE	DTG5	DTG4	DTG3	DTG2	DTG1	DTG0

Bit 7 = **PCN:** *Number* of *PWM Channels* .
0: Only PWM U signal is output to the PWM manager for six-step mode motor control (e.g. PM BLDC motors)
1: The three PWM signals U, V and W are output to the channel manager (e.g. for three-phase sinewave generation)

Bit 6 = **DTE***: Dead Time Generator Enable
0: Disable the Dead Time generator
1: Enable the Dead Time generator and apply complementary PWM signal to the adjacent switch
* write once-only bit if PCN bit is set, read/write if PCN bit is reset. To clear the DTE bit if PCN=1, it is mandatory to clear the PCN bit first.

Table 74. DeadTime generator set-up

DAC	PCN bit in MDTG register	DTE bit in MDTG register	Complementary PWM applied to adjacent switch
0	0	0	NO
0	0	1	YES
0	1	1	YES
0	1	0	YES, but WITHOUT deadtime
1	0	0	NO Complementary PWM
1	0	1	YES
1	1	1	YES
1	1	0	YES, but WITHOUT deadtime

Note 1: This table is true on condition that the CKE bit is set (Peripheral clock enabled) and the MOE bit is set (MCOx outputs enabled). See Table 56, "Output configuration summary," on page 210

When the PCN bit is reset (e.g. for PM BLDC motors), in Direct Access mode (DAC=1), if the DTE bit is reset, PWM signals can be applied on the MCOx outputs but not complementary PWM. Of course, logical levels can be also applied on the outputs.

If the DTE bit is set (PCN=0 and DAC=1), channels are paired and complementary PWM signals can be output on the MCOx pins. This will follow the rules detailed in Table 53, "Dead Time generator outputs," on page 197 as the channels are grouped in pairs.

In this case, the PWM application is selected by the OS0 bit in the MCRB register.

It is also possible to add a chopper on the PWM signal output using bits HFE[1:0] and HFRQ[2:0] in the MREF register.

Caution 1: The PWM mode will be selected via the 00[5:0] bits in the MPHST register, the OE[5:0] bits in the MPAR register and the OS2 and OS0 bits in the MCRB register as shown in Table 62, "PWM mode when SR=1," on page 213.

Caution 2: When driving motors with three independent pairs of complementary PWM signals (PCN=1), disabling the deadtime generator (DTE=0) causes the deadtime to be null: high and low side signals are exactly complemented.

It is therefore recommended not to disable the deadtime generator (it may damage the power stage), unless deadtimes are inserted externally.

Bits 5:0 = **DTG[5:0]*** *Dead time generator set-up*.

These bits set-up the deadtime duration and resolution. Refer to Table 52, "Dead time programming and example," on page 195 for details.

With F_{mtc} = 16MHz dead time values range from 125ns to 16µs with steps of 125ns, 250ns and 500ns.

* Write-once bits; once write-accessed these bits cannot be re-written unless the processor is reset (See "Caution: Access to write-once bits" on page 220.).

MOTOR CONTROLLER (Cont'd)

POLARITY REGISTER (MPOL)
Read/Write (some bits write-once)
Reset Value: 0011 1111 (3Fh)

7	6	5	4	3	2	1	0
ZVD	REO	OP5	OP4	OP3	OP2	OP1	OP0

Bit 7 = **ZVD**: *Z vs D edge polarity.*
0: Zero-crossing and End of Demagnetisation have opposite edges
1: Zero-crossing and End of Demagnetisation have same edge

Bit 6 = **REO**: *Read on High or* Low *channel bit*
0: Read the BEMF signal on High channels
1: Read on Low channels

Note: This bit always has to be configured whatever the sampling method.

Bits 5:0 = **OP[5:0]***: *Output channel polarity.*
These bits are used together with the OO[5:0] bits in the MPHST register to control the output channels.
0: Output channel is Active Low
1: Output channel is Active High.

* Write-once bits; once write-accessed these bits cannot be re-written unless the processor is reset (See "Caution: Access to write-once bits" on page 220.).

Table 75. Output Channel State Control

OP[5:0] bit	OO[5:0] bit	MCO[5:0] pin
0	0	1 (Off)
0	1	0 (PWM possible)
1	0	0 (Off)
1	1	1 (PWM possible)

Warning: OP[5:0] bits in the MPOL register must be configured as required by the application before enabling the MCO[5:0] outputs with the MOE bit in the MCRA register.

MOTOR CONTROLLER (Cont'd)

PWM REGISTER (MPWME)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DG	PWMW	PWMV	PWMU	OT3	OT2	OT1	OT0

Bit 7 = **DG**:*Debug Option.*
This bit is used to enter debug mode. As a result, C, D and Z events are output on 2 pins MCDEM and MCZEM in Switched and Autoswitched mode, C and U events are output in Speed Measurement mode. Refer to section10.6.7.3 on page 172 for more details
0: Normal mode
1: Debug mode

Bit 6 = **PWMW**: *PWM W output control*
0: PWM on Compare Register W is not output on MCPWMW pin
1: PWM on Compare Register W is output on MCPWMW pin

Bit 5 = **PWMV**: *PWM V output control*
0: PWM on Compare Register V is not output on MCPWMV pin
1: PWM on Compare Register V is output on MCPWMV pin

Bit 4 = **PWMU**: *PWM U output control*
0: PWM on Compare Register U is not output on MCPWMU pin
1: PWM on Compare Register U is output on MCPWMU pin

Bits 3:0 = **OT[3:0]**: *Off Time selection*
These bits are used to select the OFF time in sensorless current mode as shown in the following table.

Table 76. OFF time bits

OT3	OT2	OT1	OT0	Off Time sensorless mode (SR=0) (DS[3:0]=0)	Sensor Mode (SR=1) or sampling during ON ime in sensorless (SPLG =1 and/or DS [3:0] bits)
0	0	0	0	2.5 µs	No minimum off - time
0	0	0	1	5 µs	
0	0	1	0	7.5 µs	
0	0	1	1	10 µs	
0	1	0	0	12.5 µs	
0	1	0	1	15 µs	
0	1	1	0	17.5 µs	
0	1	1	1	20 µs	
1	0	0	0	22.5 µs	
1	0	0	1	25 µs	
1	0	1	0	27.5 µs	
1	0	1	1	30 µs	
1	1	0	0	32.5 µs	
1	1	0	1	35 µs	
1	1	1	0	37.5 µs	
1	1	1	1	40 µs	

Note: Times are indicated for 4 MHz f_{PERIPH}

MOTOR CONTROLLER (Cont'd)

CONFIGURATION REGISTER (MCONF)
Read/Write
Reset Value: 0000 0010 (02h)

7	6	5	4	3	2	1	0
DS3	DS2	DS1	DS0	SOI	SOM	XT16	XT8

Bits 7:4 = **DS[3:0]**: *Delay for sampling at Ton*
These bits are used to define the delay inserted before sampling in order to sample during PWM ON time.

Table 77. Sampling Delay

DS3	DS2	DS1	DS0	Delay added to sample at Ton
0	0	0	0	No delay added. Sample during Toff
0	0	0	1	2.5 µs
0	0	1	0	5 µs
0	0	1	1	7.5 µs
0	1	0	0	10 µs
0	1	0	1	12.5 µs
0	1	1	0	15 µs
0	1	1	1	17.5 µs
1	0	0	0	20 µs
1	0	0	1	22.5 µs
1	0	1	0	25 µs
1	0	1	1	27.5 µs
1	1	0	0	30 µs
1	1	0	1	32.5 µs
1	1	1	0	35 µs
1	1	1	1	37.5 µs

Note: Times are indicated for 4 MHz f_{PERIPH}

Bit 3 = **SOI** *Sampling Out Interrupt flag.*
This interrupt indicates that the sampling that should have been done during Ton has occured during the next Toff. In this case, the sample is discarded.
0: No Sampling Out Interrupt Pending
1: Sampling Out Interrupt Pending

Bit 2 = **SOM:** *Sampling Out Mask bit.*

This interrupt is available only for Z event sampling as D event sampling is always done at f_{SCF} high frequency.
0: Sampling Out interrupt disabled
1: Sampling Out interrupt enabled

This interrupt is available only when a delay has been set in the DS[3:0] bits in the MCONF register.

Note: It is recommended to disable the sampling out interrupt when software Z event is enabled (SZ bit in MCRC register is set) and if the value in the DS[3:0] bits is modified to change the sampling method during the application.

Bits [1:0] = **XT16:XT8** *BLDC drive Motor Control Peripheral input frequency selection:*

Table 78. Peripheral frequency

XT16	XT8	Peripheral frequency
0	0	$f_{PERIPH}=f_{MTC}$
0	1	$f_{PERIPH}=f_{MTC}/2$
1	0	$f_{PERIPH}=f_{MTC}/4$
1	1	$f_{PERIPH}=f_{MTC}/4$ (same as XT16=1,XT8=0)

Caution: It is recommended to set the peripheral frequency to 4MHz. Setting $f_{PERIPH}=f_{MTC}$ is used mainly when f_{clk} = 4MHz (for low power consumption).

MOTOR CONTROLLER (Cont'd)

PARITY REGISTER (MPAR)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TES1	TES0	OE5	OE4	OE3	OE2	OE1	OE0

Bits 7:6 = **TES[1:0]** *: Tacho Edge Selection bits*
The primary function of these bits is to select the edge sensitivity of the tachogenerator capture logic; clearing both TES[1:0] bits specifies that the Input Detection block does not operate in Speed Sensor Mode but either in Position Sensor or Sensorless Mode for a six-step motor drive).

Table 79. Tacho edges and input mode selection

TES 1	TES 0	Edge sensitivity	Operating Mode
0	0	Not applicable	Position Sensor or Sensorless
0	1	Rising edge	Speed Sensor
1	0	Falling edge	Speed Sensor
1	1	Rising and falling edges	Speed Sensor

Bits 5:0 = **OE[5:0]**: *Output Parity Mode.*
0: Output channel is High
1: Output channel Low

Note: These bits are not significant when PCN=1 (configuration with three independent phases).

MOTOR CONTROLLER (Cont'd)

MOTOR Z EVENT FILTER REGISTER (MZFR)
Read/Write
Reset Value: 0000 1111 (0Fh)

7	6	5	4	3	2	1	0
ZEF3	ZEF2	ZEF1	ZEF0	ZWF3	ZWF2	ZWF1	ZWF0

Bits 7:4 = **ZEF[3:0]**: *Z Event Filter bits*
These bits select the number of valid consecutive Z events (when the Z event is detected) needed to generate the active event. Sampling is done at the selected f_{SCF} frequency (see Table 82.) or at PWM frequency.

Table 80. Z Event filter Setting

ZEF3	ZEF2	ZEF1	ZEF0	Z event Samples
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14
1	1	1	0	15
1	1	1	1	16

Bits 3:0 = **ZWF[3:0]**: *Z Window Filter bits*
These bits select the length of the blanking window activated at each D event. The filter blanks the Z event detection until the end of the time window.

Table 81. Z Window filter Setting

ZWF3	ZWF2	ZWF1	ZWF0	D to Z window filter in Sensorless Mode (SR=0)	SR=1
0	0	0	0	5 µs	No Window Filter after D event
0	0	0	1	10 µs	
0	0	1	0	15 µs	
0	0	1	1	20 µs	
0	1	0	0	25 µs	
0	1	0	1	30 µs	
0	1	1	0	35 µs	
0	1	1	1	40 µs	
1	0	0	0	60 µs	
1	0	0	1	80 µs	
1	0	1	0	100 µs	
1	0	1	1	120 µs	
1	1	0	0	140 µs	
1	1	0	1	160 µs	
1	1	1	0	180 µs	
1	1	1	1	200 µs	

Note: Times are indicated for 4 MHz f_{PERIPH}

MOTOR CONTROLLER (Cont'd)

MOTOR SAMPLING CLOCK REGISTER (MSCR)
Read/Write
Reset Value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ZSV	0	0	0	SCF1	SCF0	ECM	DISS

Bit 7 = **ZSV** *Z Event Sampling Validation when MOE bit is reset*
This bit enables/disables Z event sampling in either mode (sampling at PWM frequency or at f_{SCF} frequency selected by SCF[1:0] bits)
0: Z event sampling disabled
1: Z event sampling enabled

Bits 6:4 = Reserved, must be kept cleared.

Bits 3:2 = **SCF[1:0]** *Sampling Clock Frequency*
These bits select the sampling clock frequency (f_{SCF}) used to count D & Z events.

Table 82. Sampling Clock Frequency

SCF1	SCF0	f_{SCF}
0	0	1 MHz (every 1µs)
0	1	500 kHz (every 2µs)
1	0	250 kHz (every 4µs)
1	1	125 kHz (every 8µs)

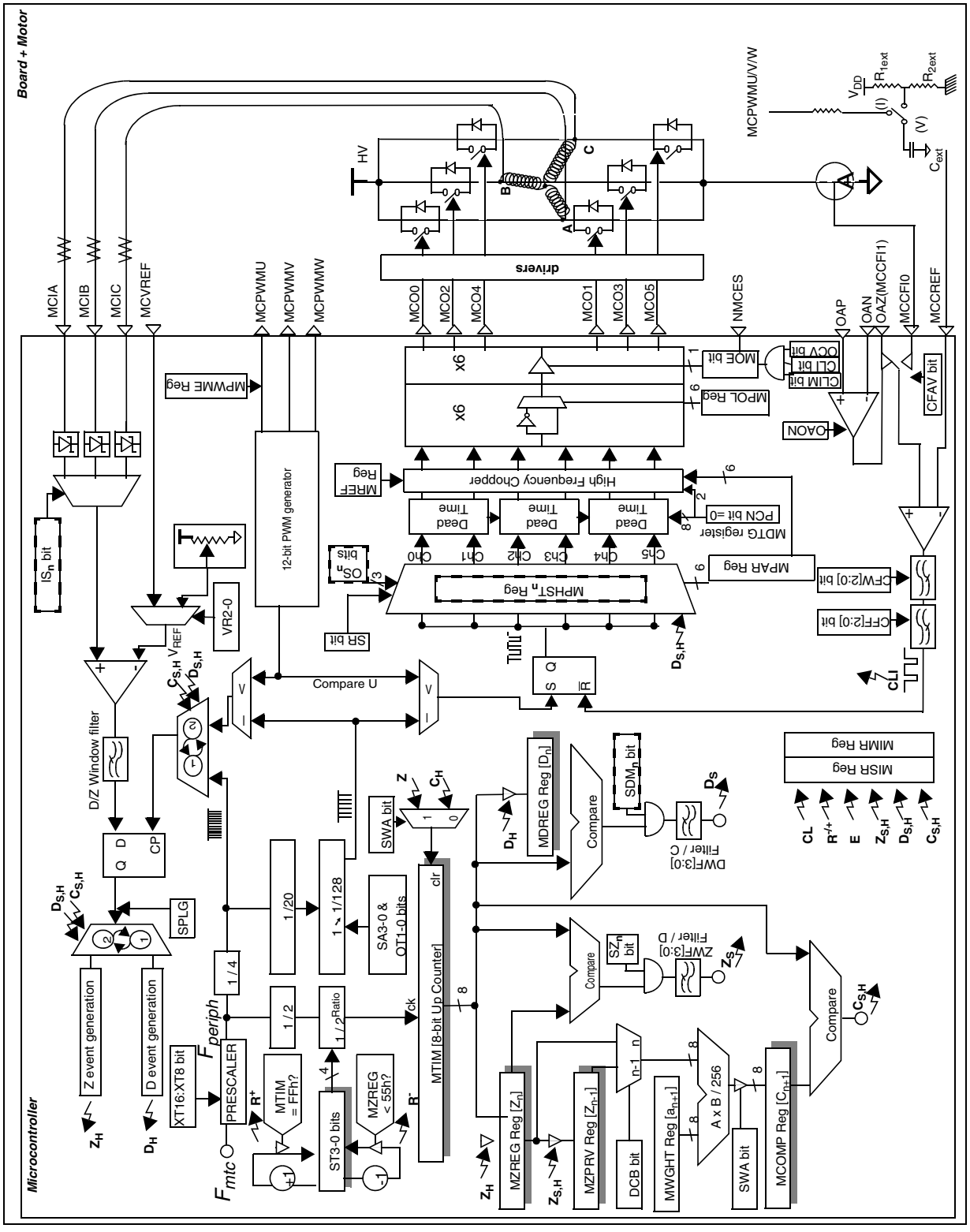
Note: Times are indicated for 4 MHz f_{PERIPH}

Bit 1 = **ECM**: *Encoder Capture Mode*
This bit is used to select the source of events which trigger the capture of the [MTIM:MTIML] counter when using Encoder speed sensor (see Figure 90).
0: Real-time Clock interrupts
1: Read access on MTIM register

Bit 0 = **DISS** *Data Input Selection*
This setting is effective only if PCN=0, TES=00 and SR=0.
0: Unused MCIx inputs are grounded
1: Unused MCIx inputs are put in HiZ

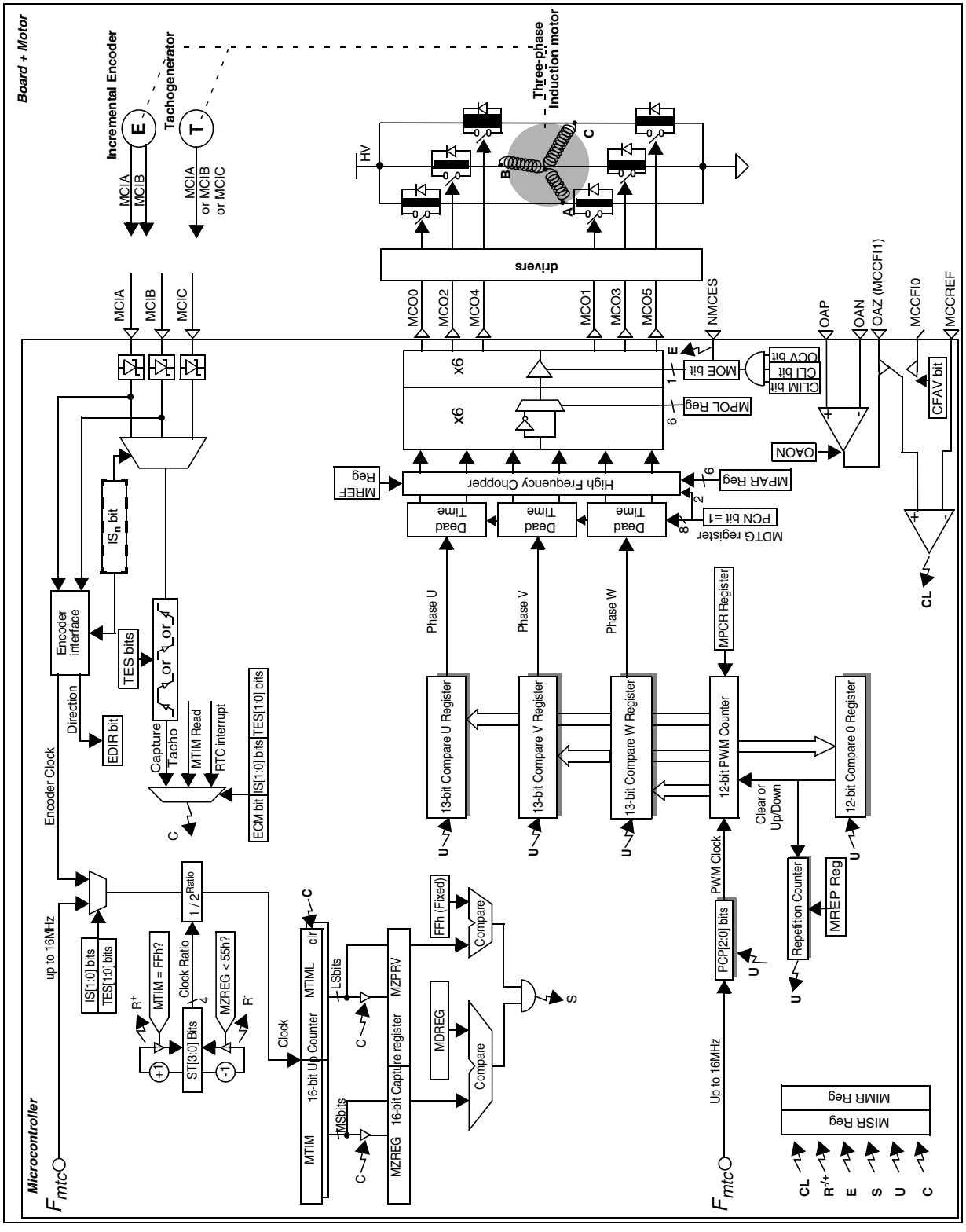
MOTOR CONTROLLER (Cont'd)

Figure 123. General view of the MTC for PM BLDC motor control



MOTOR CONTROLLER (Cont'd)

Figure 124. General view of the MTC configured for Induction motor control (proposal)



MOTOR CONTROLLER (Cont'd)

Table 83. MTC Page 0 Register Map and Reset Values

Register Name	7	6	5	4	3	2	1	0
MTIM Reset Value	T7 0	T6 0	T5 0	T4 0	T3 0	T2 0	T1 0	T0 0
MTIML Reset Value	TL7 0	TL6 0	TL5 0	TL4 0	TL3 0	TL2 0	TL1 0	TL0 0
MZPRV Reset Value	ZP7 0	ZP6 0	ZP5 0	ZP4 0	ZP3 0	ZP2 0	ZP1 0	ZP0 0
MZREG Reset Value	ZC7 0	ZC6 0	ZC5 0	ZC4 0	ZC3 0	ZC2 0	ZC1 0	ZC0 0
MCOMP Reset Value	DC7 0	DC6 0	DC5 0	DC4 0	DC3 0	DC2 0	DC1 0	DC0 0
MDREG Reset Value	DN7 0	DN6 0	DN5 0	DN4 0	DN3 0	DN2 0	DN1 0	DN0 0
MWGHT Reset Value	AN7 0	AN6 0	AN5 0	AN4 0	AN3 0	AN2 0	AN1 0	AN0 0
MPRSR Reset Value	SA3 0	SA2 0	SA1 0	SA0 0	ST3 0	ST2 0	ST1 0	ST0 0
MIMR Reset Value	PUM 0	SEM 0	RIM 0	CLIM 0	EIM 0	ZIM 0	DIM 0	CIM 0
MISR Reset Value	PUI 0	RPI 0	RMI 0	CLI 0	EI 0	ZI 0	DI 0	CI 0
MCRA Reset Value	MOE 0	CKE 0	SR 0	DAC 0	V0C1 0	SWA 0	PZ 0	DCB 0
MCRB Reset Value	0	CPB 0	HDM 0	SDM 0	OCV 0	OS2 0	OS1 0	OS0 0
MCRC Reset Value	SEI / OI 0	EDIR / HZ 0	SZ 0	SC 0	SPLG 0	VR2 0	VR1 0	VR0 0
MPHST Reset Value	IS1 0	IS0 0	OO5 0	OO4 0	OO3 0	OO2 0	OO1 0	OO0 0
MDFR Reset Value	DEF3 0	DEF2 0	DEF1 0	DEF0 0	DWF3 1	DWF2 1	DWF1 1	DWF0 1
MCFR Reset Value	RPGS 0	RST 0	CFF2 0	CFF1 0	CFF0 0	CFW2 0	CFW1 0	CFW0 0
MREF Reset Value	HST 0	CL 0	CFAV 0	HFE1 0	HFE0 0	HFRQ2 0	HFRQ1 0	HFRQ0 0
MPCR Reset Value	PMS 0	OVFU 0	OVFV 0	OVFW 0	CMS 0	PCP2 0	PCP1 0	PCP0 0
MREP Reset Value	REP7 0	REP6 0	REP5 0	REP4 0	REP3 0	REP2 0	REP1 0	REP0 0
MCPWH Reset Value	CPWH7 0	CPWH6 0	CPWH5 0	CPWH4 0	CPWH3 0	CPWH2 0	CPWH1 0	CPWH0 0

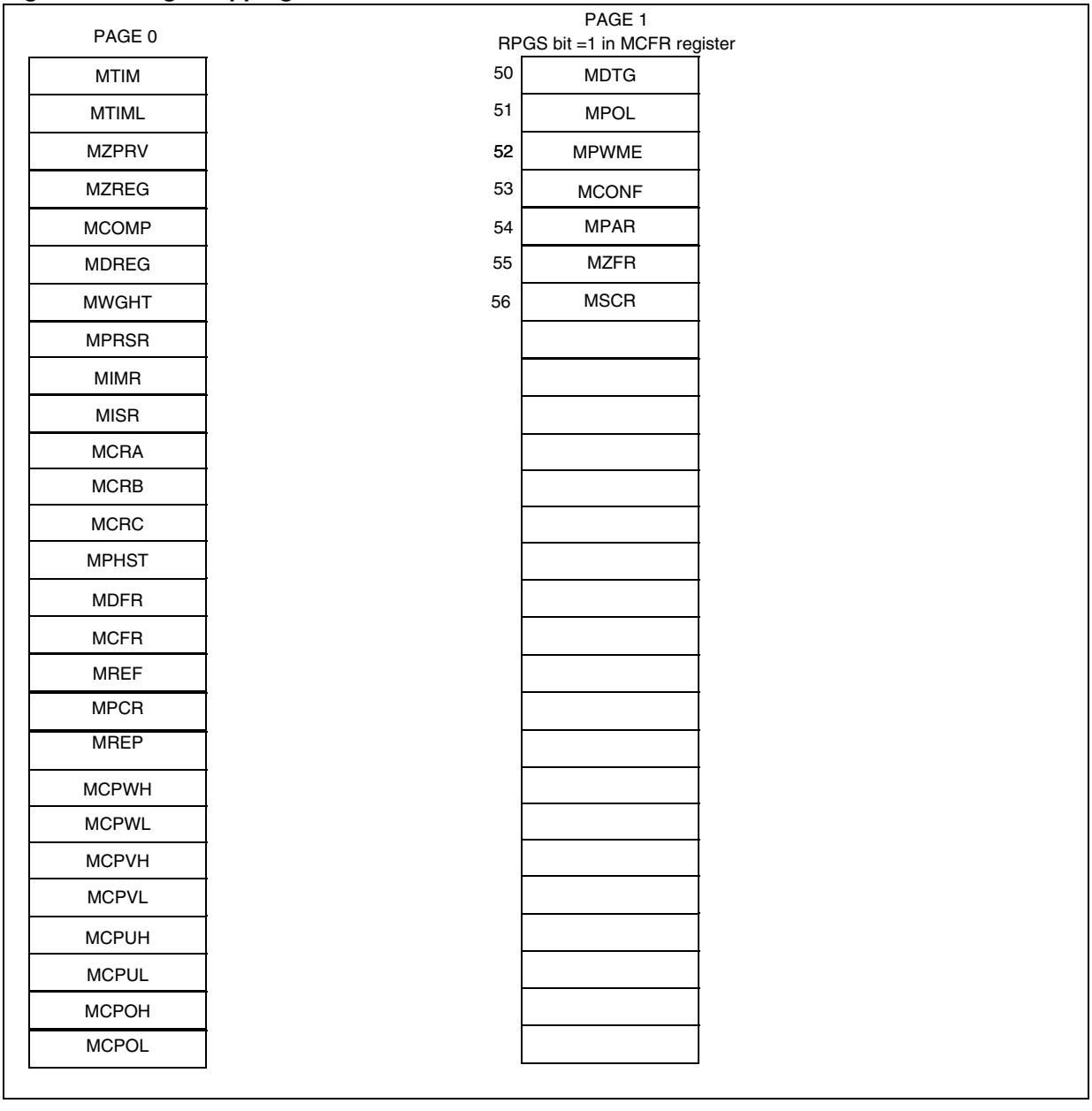
Register Name	7	6	5	4	3	2	1	0
MCPWL Reset Value	CPWL7 0	CPWL6 0	CPWL5 0	CPWL4 0	CPWL3 0	0	0	0
MCPVH Reset Value	CPVH7 0	CPVH6 0	CPVH5 0	CPVH4 0	CPVH3 0	CPVH2 0	CPVH1 0	CPVH0 0
MCPVL Reset Value	CPVL7 0	CPVL6 0	CPVL5 0	CPVL4 0	CPVL3 0	0	0	0
MCPUH Reset Value	CPUH7 0	CPUH6 0	CPUH5 0	CPUH4 0	CPUH3 0	CPUH2 0	CPUH1 0	CPUH0 0
MCPUL Reset Value	CPUL7 0	CPUL6 0	CPUL5 0	CPUL4 0	CPUL3 0	0	0	0
MCP0H Reset Value	0	0	0	0	CP0H3 1	CP0H2 1	CP0H1 1	CP0H0 1
MCP0L Reset Value	CP0L7 1	CP0L6 1	CP0L5 1	CP0L4 1	CP0L3 1	CP0L2 1	CP0L1 1	CP0L0 1

Table 84. MTC Page 1 Register Map and Reset Values

Register Name	7	6	5	4	3	2	1	0
MDTG Reset Value	PCN 1	DTE 1	DTG5 1	DTG4 1	DTG3 1	DTG2 1	DTG1 1	DTG0 1
MPOL Reset Value	ZVD 0	REO 0	OP5 1	OP4 1	OP3 1	OP2 1	OP1 1	OP0 1
MPWME Reset Value	DG 0	PWMW 0	PWMV 0	PWMU 0	OT3 0	OT2 0	OT1 0	OT0 0
MCONF Reset Value	DS3 0	DS2 0	DS1 0	DS0 0	SOI 0	SOM 0	XT16 1	XT8 0
MPAR Reset Value	TES1 0	TES0 0	OE5 0	OE4 0	OE3 0	OE2 0	OE1 0	OE0 0
MZFR Reset Value	ZEF3 0	ZEF2 0	ZEF1 0	ZEF0 0	ZWF3 1	ZWF2 1	ZWF1 1	ZWF0 1
MSCR Reset Value	ZSV 0	0	0	0	SCF1 0	SCF0 0	ECM 0	DISS 0

MOTOR CONTROLLER (Cont'd)

Figure 125. Page Mapping for Motor Control



10.6.14 Related Documentation

AN1904: ST7MC Three-phase AC Induction Motor Control Software Library

AN1905: ST7MC Three-Phase BLDC Motor Control Software Library

AN1946: Sensorless BLDC Motor Control And BEMF Sampling Methods With ST7MC

AN1947: ST7MC PMAC Sine Wave Motor Control Software Library

AN2009: PWM Management For 3-phase BLDC Motor Drives Using The ST7FMC

AN2030: Back EMF Detection During PWM On Time By ST7MC

10.7 OPERATIONAL AMPLIFIER (OA)

10.7.1 Introduction

The ST7 Op-Amp module is designed to cover various types of microcontroller applications where analog signals amplifiers are used.

It may be used to perform a variety of functions such as: differential voltage amplifier, comparator/ threshold detector, ADC zooming, impedance adaptor, general purpose operational amplifier.

10.7.2 Main Features

This module includes:

- 1 stand alone Op-Amp that may be externally connected using I/O pins
- Op-Amp output can be internally connected to the ADC inputs as well as to the motor control current feedback comparator input
- Input offset compensation with optional average
- On/Off bit to reduce power consumption and to enable the input/output connections with external pins

10.7.3 General Description

This Op-Amp can be used with 3 external pins (see device pinout description) and can be internally connected to the ADC and the Motor Control cells. The gain must be fixed with external components.

The input/output pins are connected to the Op-Amp as soon as it is switched ON (through the OACSR register).

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the “I/O ports” chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input.

The output is not connected (HiZ) when the Op-Amp is OFF. However the pin can still be used as an ADC or MTC input in this case.

When the Op-Amp is ON the output is connected to a dedicated pin which is not a standard I/O port. The output can be also be connected to the ADC or the MTC. The switches are controlled software (refer to the MTC and ADC chapters).

10.7.4 Input Offset Compensation

The Op-Amp incorporates a method to minimize the input offset which is dependant on process lot. It is useable by setting the OFFCMP bit of the control register, which launch the compensation cycle. The CMPVR bit is set by hardware as soon as this cycle is completed. The compensation is valid as long as the OFFCMP bit is high. It can be re-performed by cycling OFFCMP ‘0’ then ‘1’.

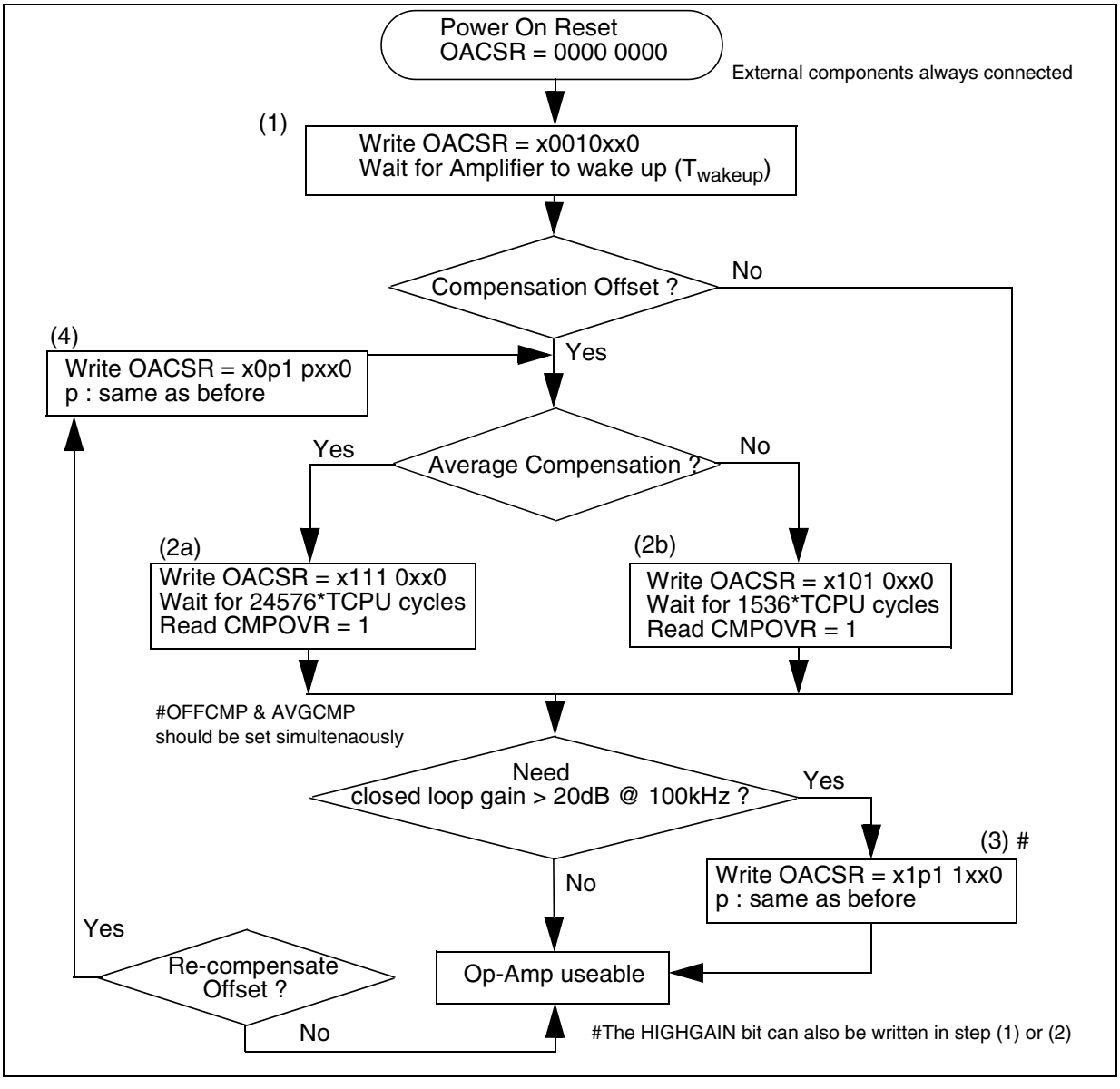
The compensation can be improved by averaging the calculation (over 16 times) setting the AVGCMP bit.

OP-AMP MODULE (Cont'd)

10.7.5 Op-Amp Programming

The flowchart for Op-Amp operation is shown in Figure 126

Figure 126. Normal Op-Amp Operation



OP-AMP MODULE (Cont'd)

10.7.6 Low power modes

Note: The Op-Amp can be disabled by resetting the OAON bit. This feature allows reduced power consumption when the amplifier is not used.

Mode	Description
Wait	No effect on Op-Amp
Halt	Op-Amp disabled After wake-up from Halt mode, the Op-Amp requires a stabilization time (see Electrical characteristics) (to be defined)

10.7.7 Interrupts

None.

10.7.8 Register Description

CONTROL/STATUS REGISTER (OACSR)

Read/Write (except bit 7 read only)

Reset Value: 0000 0000(00h)

7	6	5	4	3	2	1	0
CMP OVR	OFF CMP	AVG CMP	OAO N	HIGH GAIN	0	0	0

Bit 7 = **CMPOVR** *Compensation Completed*
This read-only bit contains the offset compensation status.
0: No offset compensation if OFFCMP = 0, or Offset compensation cycle not completed if OFFCMP = 1
1: Offset compensation completed if OFFCMP = 1

Bit 6 = **OFFCMP** *Offset Compensation*
0: Reset offset compensation values
1: Request to start offset compensation

Bit 5 = **AVGCMP** *Average Compensation*
0: One-shot offset compensation
1: Average offset compensation over 16 times

Bit 4 = **OAON** *Amplifier On*
0: Op-Amp powered off
1: Op-Amp on

Bit 3 = **HIGHGAIN** *Gain range selection*
This bit must be programmed depending on the application. It can be used to ensure 35dB open loop gain when high, it must be low when the closed loop gain is below 20dB for stability reasons.
0: Closed loop gain up to 20dB
1: Closed loop gain more than 20dB

Bits 2:0 = Reserved, must be kept cleared.

10.8 10-BIT A/D CONVERTER (ADC)

10.8.1 Introduction

The on-chip Analog to Digital Converter (ADC) peripheral is a 10-bit, successive approximation converter with internal sample and hold circuitry. This peripheral has up to 16 multiplexed analog input channels (refer to device pin out description) that allow the peripheral to convert the analog voltage levels from up to 16 different sources.

The result of the conversion is stored in 2 8-bit Data Registers. The A/D converter is controlled through a Control/Status Register.

10.8.2 Main Features

- 10-bit conversion
- Up to 16 channels with multiplexed input
- 2 software-selectable sample times
- External positive reference voltage V_{REF+} can be independent from supply
- Linear successive approximation
- Data registers (DR) which contain the results
- Conversion complete status flag
- Maskable interrupt
- On/off bit (to reduce consumption)

The block diagram is shown in Figure 127.

10.8.3 Functional Description

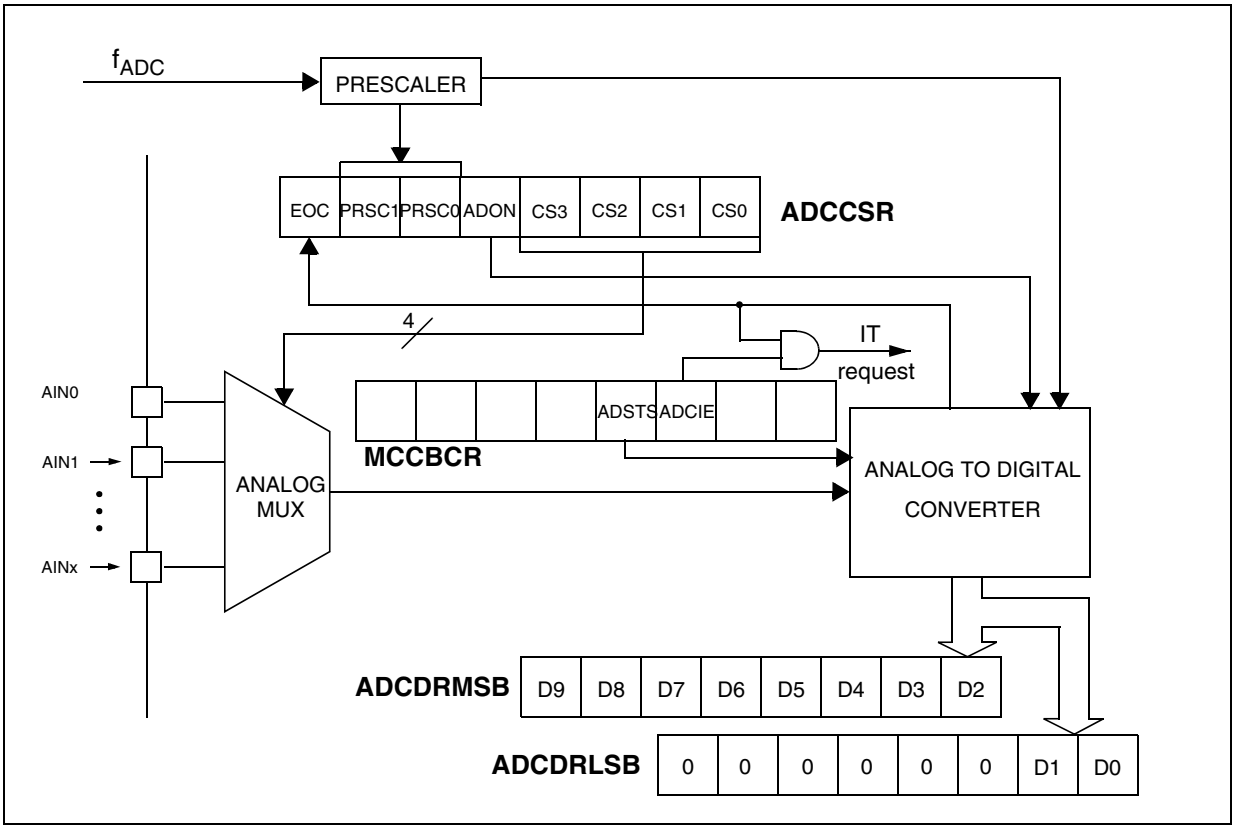
10.8.3.1 Analog References

V_{REF+} and V_{REF-} are the high and low level reference voltage pins. Conversion accuracy may therefore be impacted by voltage drops and noise on these lines. V_{REF+} can be supplied by an intermediate supply between V_{DDA} and V_{SSA} to change the conversion voltage range. V_{REF-} must be tied to V_{SSA}. An internal resistor bridge is implemented between V_{REF+} and V_{REF-} pins, with a typical value of 15kΩ

10.8.3.2 Analog Power Supply

V_{DDA} and V_{SSA} are the supply and ground pins providing power to the converter part. They must be tied to V_{DD} and V_{SS} respectively.

Figure 127. ADC Block Diagram



10-BIT A/D CONVERTER (ADC) (Cont'd)

10.8.3.3 Digital A/D Conversion Result

The conversion is monotonic, meaning that the result never decreases if the analog input does not and never increases if the analog input does not.

If the input voltage (V_{AIN}) is greater than V_{REF+} (high-level voltage reference) then the conversion result is FFh in the ADCDRMSB register and 03h in the ADCDRLSB register (without overflow indication).

If the input voltage (V_{AIN}) is lower than V_{REF-} (low-level voltage reference) then the conversion result in the ADCDRMSB and ADCDRLSB registers is 00 00h.

The A/D converter is linear and the digital result of the conversion is stored in the ADCDRMSB and ADCDRLSB registers. The accuracy of the conversion is described in the Electrical Characteristics Section.

R_{AIN} is the maximum recommended impedance for an analog input signal. If the impedance is too high, this will result in a loss of accuracy due to leakage and sampling not being completed in the alloted time.

R_{REF} is the value of the resistive bridge implemented in the device between V_{REF+} and V_{REF-}.

10.8.3.4 A/D Conversion

The analog input ports must be configured as input, no pull-up, no interrupt. Refer to the «I/O ports» chapter. Using these pins as analog inputs does not affect the ability of the port to be read as a logic input. If the application used the high-impedance analog inputs, then the sample time should be stretched by setting the ADSTS bit in the MCCBCR register.

In the ADCCSR register:

– Select the CS[3:0] bits to assign the analog channel to convert.

ADC Conversion mode

In the ADCCSR register:

– Set the ADON bit to enable the A/D converter and to start the conversion. From this time on, the ADC performs a continuous conversion of the selected channel.

– The EOC bit is kept low by hardware during the conversion.

Note: Changing the A/D channel during conversion will stop the current conversion and start conversion of the newly selected channel.

10-BIT A/D CONVERTER (ADC) (Cont'd)

When a conversion is complete:

- The EOC bit is set by hardware
- An interrupt request is generated if the ADCIE bit in the MCCBCR register is set (see section 6.4.7 on page 38).
- The result is in the ADCDR registers and remains valid until the next conversion has ended.

To read the 10 bits, perform the following steps:

1. Poll the EOC bit or wait for EOC interrupt
2. Read ADCDRLSB
3. Read ADCDRMSB

The EOC bit is reset by hardware once the ADCDRMSB is read.

To read only 8 bits, perform the following steps:

1. Poll the EOC bit or wait for EOC interrupt
2. Read ADCDRMSB

The EOC bit is reset by hardware once the ADCDRMSB is read.

Changing the conversion channel

The application can change channels during conversion. In this case the current conversion is stopped and the A/D converter starts converting the newly selected channel.

ADCCR consistency

If an End Of Conversion event occurs after software has read the ADCDRLSB but before it has read the ADCDRMSB, there would be a risk that the two values read would belong to different samples.

To guarantee consistency:

- The ADCDRMSB and the ADCDRLSB are locked when the ADCCRLSB is read
- The ADCDRMSB and the ADCDRLSB are unlocked when the MSB is read or when ADON is reset.

Thus, it is mandatory to read the ADCDRMSB just after reading the ADCDRLSB. Otherwise the ADCDR register will not be updated until the ADCDRMSB is read.

10.8.4 Low Power Modes

Note: The A/D converter may be disabled by resetting the ADON bit. This feature allows reduced power consumption when no conversion is needed.

Mode	Description
Wait	No effect on A/D Converter
Halt	A/D Converter disabled. After wake up from Halt mode, the A/D Converter requires a stabilization time t_{STAB} (see Electrical Characteristics) before accurate conversions can be performed.

10.8.5 Interrupts

Interrupt Event	Event Flag	Enable Control Bit	Exit from Wait	Exit from Halt
End of Conversion	EOC	ADCIE[1)]	Yes	No

[1)]The ADCIE bit is in the MCCBCR register (see section 6.4.7 on page 38)

10-BIT A/D CONVERTER (ADC) (Cont'd)

10.8.6 Register Description

CONTROL/STATUS REGISTER (ADCCSR)
Read/Write (Except bit 7 read only)
Reset Value: 0000 0000 (00h)

7							0
EOC	PRSC1	PRSC0	ADON	CS3	CS2	CS1	CS0

Bit 7 = **EOC** *End of Conversion*
This bit is set by hardware. It is cleared by software reading the ADCDRMSB register.
0: Conversion is not complete
1: Conversion complete

Bit 6:5 = **PRSC[1:0]** *ADC clock prescaler selection*
These bits are set and cleared by software.

f_{ADC}	PRSC1	PRSC0
4MHz	0	0
2MHz	0	1
1MHz	1	0

Bit 4 = **ADON** *A/D Converter on*
This bit is set and cleared by software.
0: Disable ADC and stop conversion
1: Enable ADC and start conversion

Bit 3:0 = **CS[3:0]** *Channel Selection*
These bits are set and cleared by software. They select the analog input to convert.

Channel Pin*	CH3	CH2	CH1	CH0
AIN0	0	0	0	0
AIN1	0	0	0	1
AIN2	0	0	1	0
AIN3	0	0	1	1
AIN4	0	1	0	0
AIN5	0	1	0	1
AIN6	0	1	1	0
AIN7	0	1	1	1
AIN8	1	0	0	0
AIN9	1	0	0	1
AIN10	1	0	1	0
AIN11	1	0	1	1
AIN12	1	1	0	0
AIN13	1	1	0	1
AIN14	1	1	1	0
AIN15	1	1	1	1

*The number of channels is device dependent. Refer to the device pinout description.

DATA REGISTER (ADCDRMSB)
Read Only
Reset Value: 0000 0000 (00h)

7							0
D9	D8	D7	D6	D5	D4	D3	D2

Bit 7:0 = **D[9:2]** *MSB of Analog Converted Value*
This register contains the MSB of the converted analog value.

DATA REGISTER (ADCDRLSB)
Read Only
Reset Value: 0000 0000 (00h)

7							0
0	0	0	0	0	0	D1	D0

Bit 7:2 = Reserved. Forced by hardware to 0.

Bit 1:0 = **D[1:0]** *LSB of Analog Converted Value*
This register contains the LSB of the converted analog value.

10-BIT A/D CONVERTER (ADC) (Cont'd)

Table 85. ADC Register Map and Reset Values

Address (Hex.)	Register Label	7	6	5	4	3	2	1	0
2E	**ADCCSR**	EOC	PRSC1	PRSC0	ADON	CS3	CS2	CS1	CS0
	Reset Value	0	0	0	0	0	0	0	0
2F	**ADCDRMSB**	D9	D8	D7	D6	D5	D4	D3	D2
	Reset Value	0	0	0	0	0	0	0	0
30	**ADCDRLSB**	0	0	0	0	0	0	D1	D0
	Reset Value	0	0	0	0	0	0	0	0

11 INSTRUCTION SET

11.1 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 86. 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)

11.1.1 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

11.1.2 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

11.1.3 Direct

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

The direct addressing mode consists of two submodes:

Direct (short)

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

Direct (long)

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

11.1.4 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.

11.1.5 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)

11.1.6 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 87. 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

11.1.7 Relative mode (Direct, Indirect)

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

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

The relative addressing mode consists of two submodes:

Relative (Direct)

The offset is following the opcode.

Relative (Indirect)

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

INSTRUCTION SET OVERVIEW (Cont'd)

11.2 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.

11.2.1 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	

12 ELECTRICAL CHARACTERISTICS

12.1 PARAMETER CONDITIONS

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

12.1.1 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).

12.1.2 Typical values

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

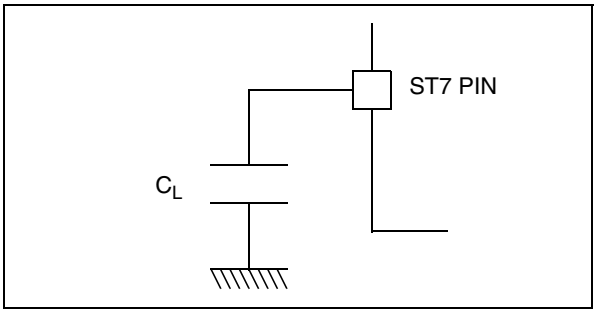
12.1.3 Typical curves

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

12.1.4 Loading capacitor

The loading conditions used for pin parameter measurement are shown in Figure 128.

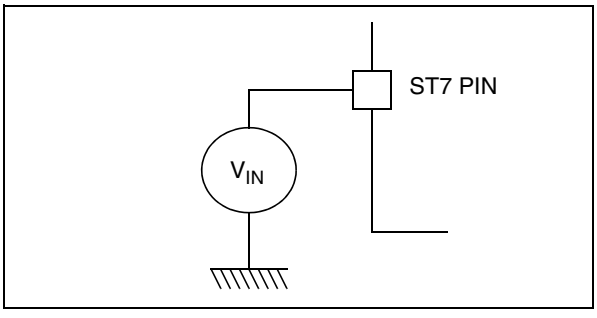
Figure 128. Pin loading conditions



12.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in Figure 129.

Figure 129. Pin input voltage



12.2 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.

12.2.1 Voltage Characteristics

Symbol	Ratings	Maximum value	Unit
V_{DD} - V_{SS}	Supply voltage	6.5	V
V_{PP} - V_{SS}	Programming Voltage	13	
V_{IN}	Input voltage on any pin [1) & 2)]	V_{SS}-0.3 to V_{DD}+0.3	
$\lvert\Delta V_{DDx}\rvert$ and $\lvert\Delta V_{SSx}\rvert$	Variations between different digital power pins	50	mV
$\lvert V_{SSA} - V_{SSx}\rvert$	Variations between digital and analog ground pins	50	
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human Body Model)	see section 12.7.3 on page 263	
$V_{ESD(CDM)}$	Electro-static discharge voltage (Charged Device Model)		

12.2.2 Current Characteristics

Symbol	Ratings		Maximum value	Unit
I_{VDD}	Total current into V_{DD} power lines (source) [3)]	32-pin devices	75	mA
		44-pin devices	125	
		56, 64, 80-pin devices	175	
I_{VSS}	Total current out of V_{SS} ground lines (sink) [3)]	32-pin devices	75	
		44-pin devices	125	
		56, 64, 80-pin devices	175	
I_{IO}	Output current sunk by any standard I/O and control pin		25	
	Output current sunk by any high sink I/O pin		50	
	Output current source by any I/Os and control pin		- 25	
$I_{INJ(PIN)}$ [2) & 4)]	Injected current on V_{PP} pin		± 5	
	Injected current on $\overline{\text{RESET}}$ pin		± 5	
	Injected current on OSC1 and OSC2 pins		± 5	
	Injected current on any other pin [5)]		± 5	
$\Sigma I_{INJ(PIN)}$ [2)]	Total injected current (sum of all I/O and control pins) [5)]		± 20	

Notes:

1. Directly connecting the $\overline{\text{RESET}}$ and I/O pins to V_{DD} or V_{SS} could damage the device if an unintentional internal reset is generated or an unexpected change of the I/O configuration occurs (for example, due to a corrupted program counter). To guarantee safe operation, this connection has to be done through a pull-up or pull-down resistor (typical: 4.7kΩ for $\overline{\text{RESET}}$, 10kΩ for I/Os). For the same reason, unused I/O pins must not be directly tied to V_{DD} 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 $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$.
3. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
4. Negative injection disturbs the analog performance of the device. See note in “ADC Accuracy with VDD=5.0V” on page 284.
For best reliability, it is recommended to avoid negative injection of more than 1.6mA
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.

ABSOLUTE MAXIMUM RATINGS (Cont'd)

12.2.3 Thermal Characteristics

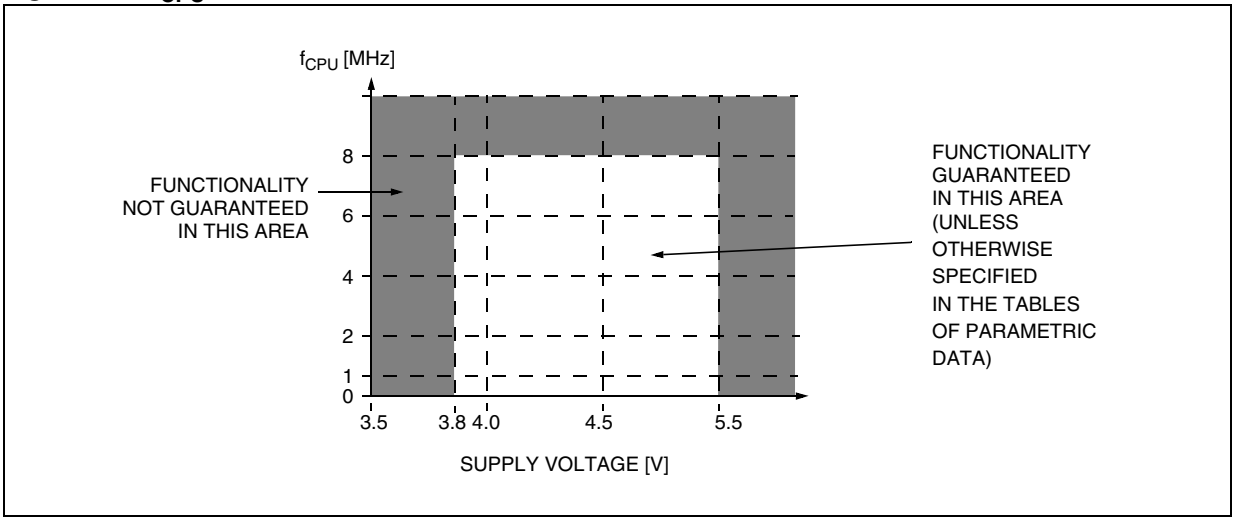
Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature (see Section 13.2 THERMAL CHARACTERISTICS)		

12.3 OPERATING CONDITIONS

12.3.1 General Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
f_{CPU}	Internal clock frequency versus V_{DD}		0	8	MHz
V_{DD}	Extended operating voltage	No Flash Write/Erase. Analog parameters not guaranteed [1]	3.8	5.5	V
	Standard operating voltage		4.5	5.5	
	Operating voltage for flash Write/Erase	V_{PP} = 11.4 to 12.6V	4.5	5.5	
T_A	Ambient temperature range	6 Suffix Version	-40	85	°C
		C Suffix Version	-40	125	

Figure 130. f_{CPU} Max Versus V_{DD}



Note 1: Clock Detector, ADC, comparator and OPAMP functionalities guaranteed only within 4.5-5.5V voltage range.

Note:

Some temperature ranges are only available with a specific package and memory size. Refer to Ordering Information.

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

OPERATING CONDITIONS (Cont'd)

12.3.2 Operating Conditions with Low Voltage Detector (LVD)

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

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$V_{IT+(LVD)}$	Reset release threshold (V_{DD} rise)		3.90	4.20	4.50	V
$V_{IT-(LVD)}$	Reset generation threshold (V_{DD} fall)		3.80	4.00	4.35	
$V_{hys(LVD)}$	LVD voltage threshold hysteresis	$V_{IT+(LVD)}$-$V_{IT-(LVD)}$		200		mV
Vt_{POR}	V_{DD} rise time rate [1]		20			µs/V
					100	ms/V
$t_{g(VDD)}$	Width of filtered glitches on V_{DD} [1] (which are not detected by the LVD)				40	ns

Notes:

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

12.3.3 Auxiliary Voltage Detector (AVD) Thresholds

Subject to general operating condition for V_{DD}, f_{OSC}, and T_A.

Symbol	Parameter	Conditions	Min	Typ	Max[2]	Unit
$V_{IT+(AVD)}$	1⇒0 AVDF flag toggle threshold (V_{DD} rise)		4.35	4.70	4.90	V
$V_{IT-(AVD)}$	0⇒1 AVDF flag toggle threshold (V_{DD} fall)		4.20	4.50	4.70	
$V_{hyst(AVD)}$	AVD voltage threshold hysteresis[1]	$V_{IT+(AVD)}$-$V_{IT-(AVD)}$		200		mV
ΔV_{IT-}	Voltage drop between AVD flag set and LVD reset activated	$V_{IT-(AVD)}$-$V_{IT-(LVD)}$		450		mV

Notes:

1. Data based on characterization results, not tested in production.
2. See "MAXIMUM VALUES OF AVD THRESHOLDS" on page 304..

12.4 SUPPLY CURRENT CHARACTERISTICS

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

12.4.1 Run and Slow modes (Flash devices)

Symbol	Parameter	Conditions		Typ	Max [1]	Unit
I_{DD}	Supply current in Run mode [2] (see Figure 131)	$4.5V \leq V_{DD} \leq 5.5V$	f_{OSC}=16MHz, f_{CPU}=8MHz	12	18	mA
	Supply current in Slow mode [2] (see Figure 132)		f_{OSC}=16MHz, f_{CPU}=500kHz	5	8	mA

Figure 131. Typical I_{DD} in Run vs. f_{CPU}

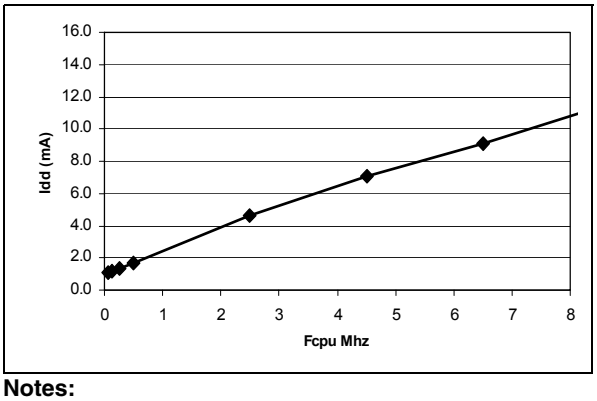
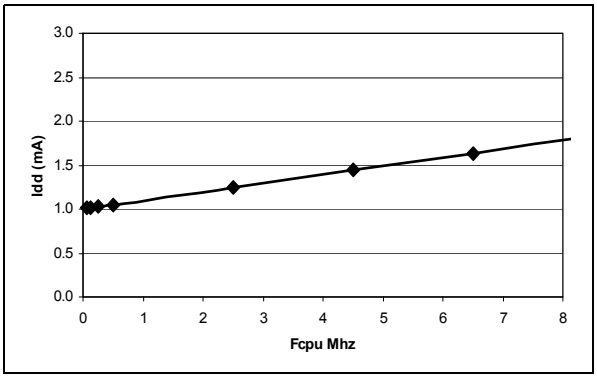


Figure 132. Typical I_{DD} in Slow vs. f_{CPU}



Notes:

1. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
2. Measurements are done in the following conditions:

- Program executed from RAM, CPU running with RAM access. The increase in consumption when executing from Flash is 50%.
- All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load)
- All peripherals in reset state.
- LVD disabled.
- Clock input (OSC1) driven by external square wave.
- In Slow and Slow-wait mode, f_{CPU} is based on f_{OSC} divided by 32.

To obtain the total current consumption of the device, add the clock source (Section 12.5.3) and the peripheral power consumption.

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

12.4.2 Wait and Slow-wait Modes

Symbol	Parameter	Conditions		Typ	Max [1]	Unit
I_{DD}	Supply current in Wait mode [2] (see Figure 133)	4.5V≤V_{DD}≤5.5V	f_{OSC}=16MHz, f_{CPU}=8MHz	8	12	mA
	Supply current in Slow-wait mode [2] (see Figure 134)		f_{OSC}=16MHz, f_{CPU}=500kHz	3.5	5	

Figure 133. Typical I_{DD} in Wait vs. f_{CPU}

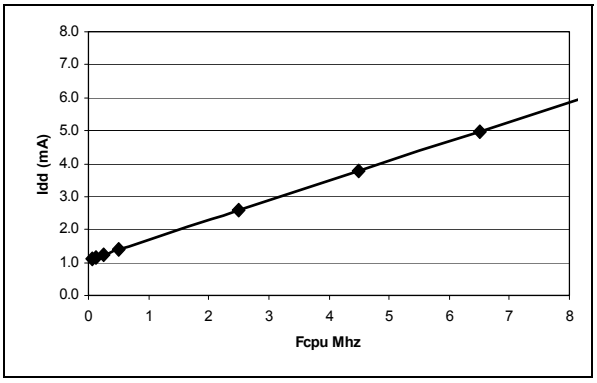
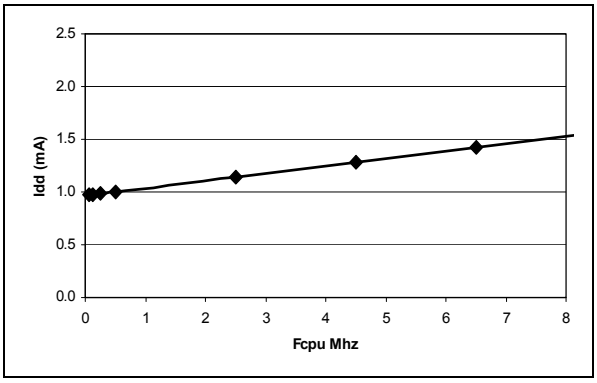


Figure 134. Typical I_{DD} in Slow-wait vs. f_{CPU}



Notes:

1. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.
2. Measurements are done in the following conditions:

- Program executed from RAM, CPU running with RAM access. The increase in consumption when executing from Flash is 50%.
- All I/O pins in input mode with a static value at V_{DD} or V_{SS} (no load)
- All peripherals in reset state.
- LVD disabled.
- Clock input (OSC1) driven by external square wave.
- In Slow and Slow-wait mode, f_{CPU} is based on f_{OSC} divided by 32.

To obtain the total current consumption of the device, add the clock source (Section 12.5.3) and the peripheral power consumption.

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

12.4.3 Halt and Active-halt modes

Symbol	Parameter	Conditions		Typ	Max	Unit
I_{DD}	Supply current in Halt mode [1]	V_{DD}=5.5V	-40°C≤T_A≤85°C	1	10	µA
			-40°C≤T_A≤125°C		50	
	Supply current in Active-halt mode [2]	16Mhz external clock		1	1.5	mA

1. All I/O pins in push-pull output mode (when applicable) with a static value at V_{DD} or V_{SS} (no load), PLL and LVD disabled. Data based on characterization results, tested in production at V_{DD} max. and f_{CPU} max.

2. All I/O pins in input mode with a static value at V_{DD} or V_{SS}. Tested in production at V_{DD} max and f_{cpu} max with clock input OSC1 driven by an external square wave; V_{DD} applied on OSC2 to reduce oscillator consumption. Consumption may be slightly different with a quartz or resonator.

12.4.4 Supply and Clock Managers

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

Symbol	Parameter	Conditions	Typ	Max	Unit
$I_{DD(LVD)}$	LVD supply current	Halt mode	180	280	µA
$I_{DD(PLL)}$	PLL supply current	V_{DD} = 5V	700		

SUPPLY CURRENT CHARACTERISTICS (Cont'd)

12.4.5 On-Chip Peripherals

Symbol	Parameter	Conditions		Typ	Unit
$I_{DD(TIM)}$	16-bit Timer supply current [1]	f_{CPU}=8MHz	V_{DD}=5.0V	50	µA
$I_{DD(ART)}$	ART PWM supply current [2]	f_{CPU}=8MHz	V_{DD}=5.0V	75	
$I_{DD(SPI)}$	SPI supply current [3]	f_{CPU}=8MHz	V_{DD}=5.0V	400	
$I_{DD(SCI)}$	SCI supply current [4]	f_{CPU}=8MHz	V_{DD}=5.0V	400	
$I_{DD(MTC)}$	MTC supply current [5]	f_{CPU}=8MHz	V_{DD}=5.0V	500	
$I_{DD(ADC)}$	ADC supply current when converting [6]	f_{ADC}=4MHz	V_{DD}=5.0V	400	
$I_{DD(OPAMP)}$	OPAMP supply current [7]	f_{CPU}=8MHz	V_{DD}=5.0V	1500	

Notes:

1. Data based on a differential I_{DD} measurement between reset configuration (timer counter running at $f_{CPU}/4$) and timer counter stopped (only TIMD bit set). Data valid for one timer.

2. Data based on a differential I_{DD} measurement betwwen reset configuration (timer stopped) and timer counter enable (only TCE bit set)

3. 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.

4. Data based on a differential I_{DD} measurement between SCI low power state (SCID=1) and a permanent SCI data transmit sequence.

5. Data based on a differential I_{DD} measurement between reset configuration (motor control disabled) and the whole motor control cell enable in speed measurement mode. MCO outputs are not validated.

6. Data based on a differential I_{DD} measurement between reset configuration and continuous A/D conversions.

7. Data based on a differential measurement between reset configuration (OPAMP disabled) and amplification of a sinewave (no load, A_{VCL}=1, V_{DD}=5V).

12.5 CLOCK AND TIMING CHARACTERISTICS

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

12.5.1 General Timings

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

Notes:

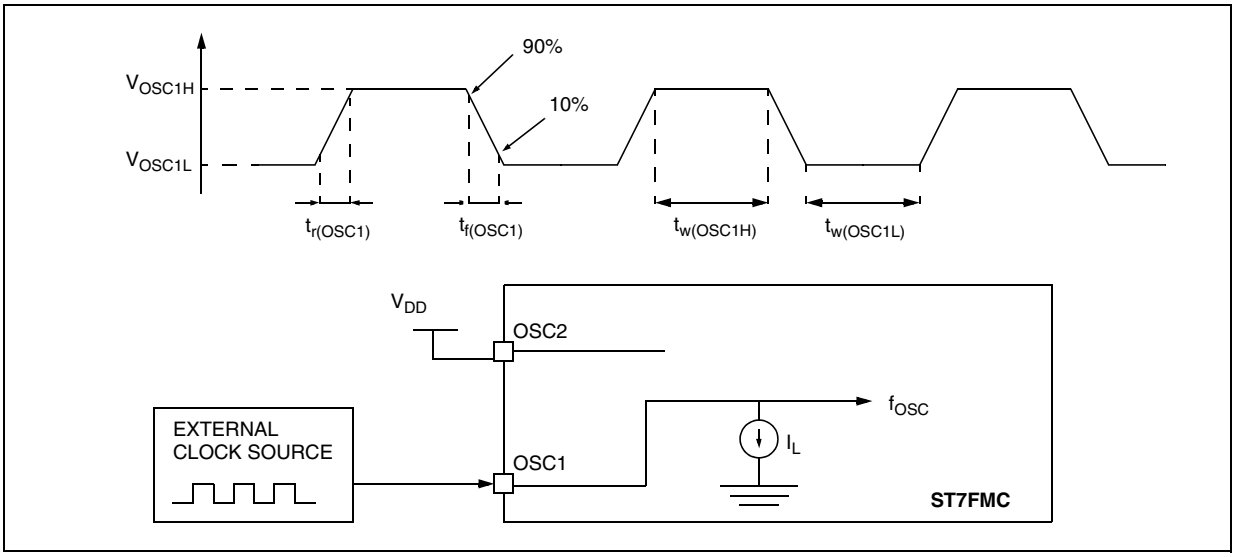
1. Data based on typical application software.

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.

12.5.2 External Clock Source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSC1H}	OSC1 input pin high level voltage	see Figure 135	0.7xV_{DD}		V_{DD}	V
V_{OSC1L}	OSC1 input pin low level voltage		V_{SS}		0.3xV_{DD}	
$t_{w(OSC1H)}$ $t_{w(OSC1L)}$	OSC1 high or low time [1]		25			ns
$t_{r(OSC1)}$ $t_{f(OSC1)}$	OSC1 rise or fall time [1]				5	
I_L	OSCx Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			± 1	µA

Figure 135. Typical Application with an External Clock Source



Notes:

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

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

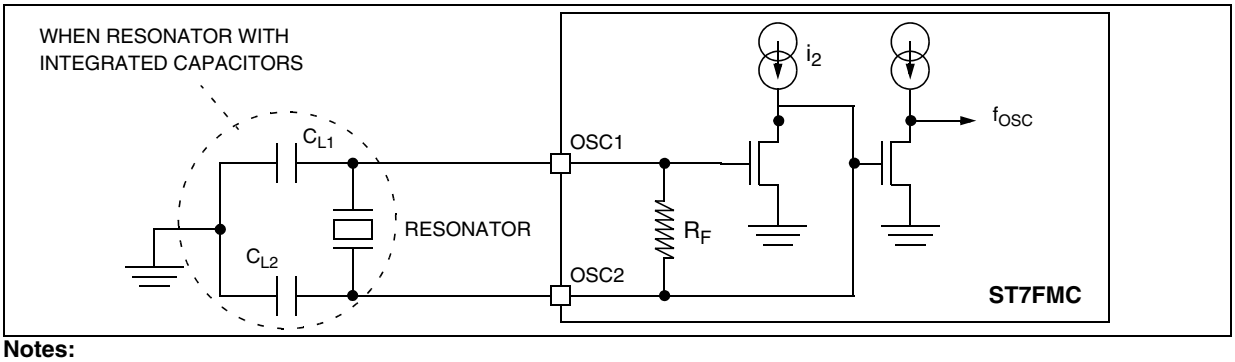
12.5.3 Crystal and Ceramic Resonator Oscillators

The ST7 internal clock can be supplied with four different Crystal/Ceramic resonator oscillators. All the information given in this paragraph are based on characterization results with specified typical external components. In the application, the resonator and the load capacitors have to be placed as close as possible to the oscillator pins in order to minimize output distortion and start-up stabilization time. Refer to the crystal/ceramic resonator manufacturer for more details (frequency, package, accuracy...).

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	Oscillator Frequency [1]		4		16	MHz
R_F	Feedback resistor			92		kΩ
C_{L1} C_{L2}	Recommended load capacitance versus equivalent serial resistance of the crystal or ceramic resonator (R_S)		See table below			pF

Supplier	f_{OSC} (MHz)	Typical Ceramic Resonators [2] Reference [3]	CL1 [pF]	CL2 [pF]
Murata	4	CSTCR4M00G53-R0	(15)	(15)
	8	CSTCE8M00G52-R0	(10)	(10)
	16	CSTCE16M0V53-R0	(15)	(15)

Figure 136. Typical Application with a Crystal or Ceramic Resonator



Notes:

1. When PLL is used, please refer to the PLL characteristics chapter and to the "supply, reset and clock management" description chapter (f_{OSC} min. is 8 Mhz with PLL).
2. Resonator characteristics given by the ceramic resonator manufacturer. For more information on these resonators, please consult www.murata.com
3. SMD = [-R0: Plastic tape package (ý =180mm), -B0: Bulk]
   LEAD = [-A0: Flat pack package (Radial taping Ho= 18mm), -B0: Bulk]

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

12.5.4 Clock Security System with PLL

Table 88. PLL Characteristics

Symbol	Parameter	Min	Typ	Max	Unit
f_{OSC}	PLL input frequency range	7		8	MHz
Output Frequency	Output frequency when the PLL attain lock.		16		MHz
t_{Lock}	PLL Lock Time (LOCKED = 1)		50	100	µs
Jitter	Jitter in the output clock		2		%
f_{CPU}	CPU clock frequency when VCO is connected to ground (ICD internal clock or back up oscillator)		3		MHz

Table 89. Clock Detector Characteristics

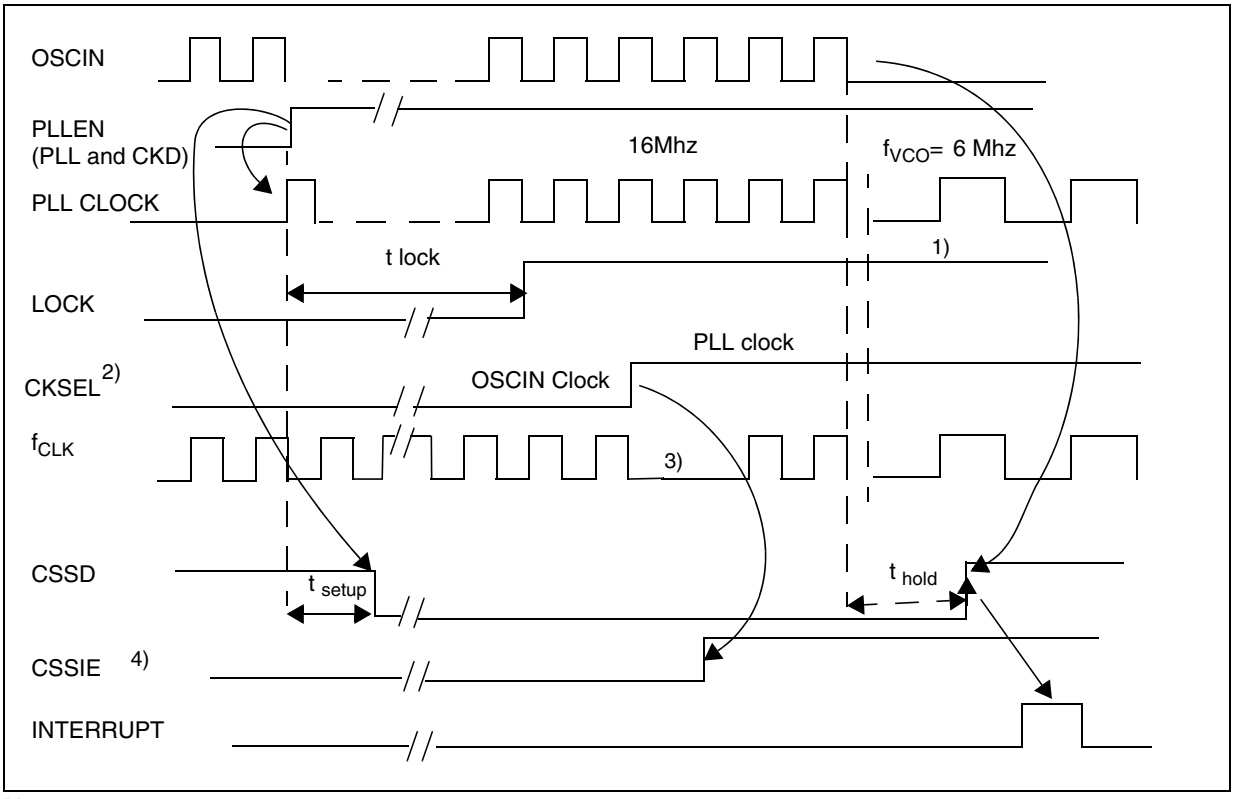
Symbol	Parameter	Min	Typ	Max	Unit
f_{Detect}	Detected Minimum Input Frequency			500 [1)]	KHz
t_{setup}	Time needed to detect OSCIN once CKD is enabled		3		µs
t_{hold}	Time needed to detect that OSCIN stops		3		µs

Notes:

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

CLOCK AND TIMING CHARACTERISTICS (Cont'd)

Figure 137. PLL And Clock Detector Signal Start Up Sequence



Notes:

1. Lock does not go low without resetting the PLLEN bit.

2. Before setting the CKSEL bit by software in order to switch to the PLL clock, a period of t_{lock} must have elapsed.

3. 2 clock cycles are missing after CKSEL = 1

4. CKSEL bit must be set before enabling the CSS interrupt (CSSIE=1).

12.6 MEMORY CHARACTERISTICS

12.6.1 RAM and Hardware Registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode [1]	Halt mode (or RESET)	1.6			V

12.6.2 FLASH Memory

DUAL VOLTAGE HDFLASH MEMORY						
Symbol	**Parameter**	**Conditions**	**Min [2]**	**Typ**	**Max [2]**	**Unit**
f_{CPU}	Operating frequency	Read mode	0		8	MHz
		Write / Erase mode	1		8	
V_{PP}	Programming voltage [3]	4.5V ≤V_{DD} ≤5.5V	11.4		12.6	V
I_{PP}	V_{PP} current[4] [5]	Read (V_{PP}=12V)			200	µA
		Write / Erase			30	mA
t_{VPP}	Internal V_{PP} stabilization time			10		µs
t_{RET}	Data retention	T_A=85°C	40			years
		T_A=105°C	15			
		T_A=125°C	7			
N_{RW}	Write erase cycles	T_A=25°C	100			cycles
T_{PROG} T_{ERASE}	Programming or erasing temperature range		-40	25	85	°C

Notes:

1. Minimum V_{DD} supply voltage without losing data stored in RAM (in Halt mode or under RESET) or in hardware registers (only in Halt mode). Not tested in production.
2. Data based on characterization results, not tested in production.
3. V_{PP} must be applied only during the programming or erasing operation and not permanently for reliability reasons.
4. Data based on simulation results, not tested in production
5. In Write/Erase mode the I_{DD} supply current consumption is the same as in Run mode (section 12.4.1 on page 252)

12.7 EMC CHARACTERISTICS

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

12.7.1 Functional EMS (Electro Magnetic Susceptibility)

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

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

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

12.7.1.1 Designing hardened software to avoid noise problems

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

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

Software recommendations:

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

– Corrupted program counter

– Unexpected reset

– Critical Data corruption (control registers...)

Prequalification trials:

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

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

Symbol	Parameter	Conditions		Level/ Class
V_{FESD}	Voltage limits to be applied on any I/O pin to induce a functional disturbance	Flash/ROM devices	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz, LVD OFF conforms to IEC 1000-4-2	4A
			V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz, LVD ON conforms to IEC 1000-4-2	2B
V_{FFTB}	Fast transient voltage burst limits to be applied through 100pF on V_{DD} and V_{DD} pins to induce a functional disturbance	Flash devices	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz, conforms to IEC 1000-4-4	4A
		ROM devices	V_{DD}=5V, T_A=+25°C, f_{OSC}=8MHz, conforms to IEC 1000-4-4	3B

EMC CHARACTERISTICS (Cont'd)

12.7.2 EMI (Electromagnetic interference)

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

Symbol	Parameter	Conditions	Device/ Package	Monitored Frequency Band	Max vs. [f_{OSC}/f_{CPU}]		Unit
					8/4MHz	16/8MHz	
S_{EMI}	Peak level	V_{DD}=5V, T_A=+25°C conforming to SAE J 1752/3	Flash/LQFP64	0.1MHz to 30MHz	8	6	dBµV
				30MHz to 130MHz	8	12	
				130MHz to 1GHz	1	9	
				SAE EMI Level	1.5	2.5	-

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)

12.7.3 Absolute Maximum Ratings (Electrical Sensitivity)

Based on two different tests (ESD and LU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. For more details, refer to the application note AN1181.

12.7.3.1 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). Three models can be simulated: Human Body Model, Machine Model and Charged Device Model. This test conforms to the JESD22-A114A/A115A/C101-A 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
$V_{ESD(CDM)}$	Electro-static discharge voltage (Charged Device Model)	T_A=+25°C	250	

Notes:

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

12.7.3.2 Static Latch-Up

- **LU**: two complementary static tests are required on 10 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard.

Electrical Sensitivities

Symbol	Parameter	Conditions	Class [1]
LU	Static latch-up class	T_A=+25°C T_A=+125°C	A 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).

12.8 I/O PORT PIN CHARACTERISTICS

12.8.1 General Characteristics

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

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IL}	Input low level voltage	CMOS ports			$0.3xV_{DD}$	V
V_{IH}	Input high level voltage		$0.7xV_{DD}$			
V_{hys}	Schmitt trigger voltage hysteresis [2]			1		V
V_{IL}	Input low level voltage	G & H ports			0.8	V
V_{IH}	Input high level voltage		2.8			
V_{hys}	Schmitt trigger voltage hysteresis [2]			400		mV
$I_{INJ(PIN)}$ [3]	Injected Current on an I/O	V_{DD}=5V			+5/-2	mA
$\Sigma I_{INJ(PIN)}$ [3]	Total injected current (sum of all I/O and control pins)				± 25	
I_L	Input leakage current	$V_{SS} \leq V_{IN} \leq V_{DD}$			±1	µA
I_S	Static current consumption induced by each floating input pin[4]	Floating input mode		200		
R_{PU}	Weak pull-up equivalent resistor [5]	$V_{IN}=V_{SS}$	50	90	250	kΩ
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time [1]	C_L=50pF Between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time [1]			25		
$t_{w(IT)in}$	External interrupt pulse time [6]		1			t_{CPU}

Notes:

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

2. Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.

3. $I_{INJ(PIN)}$ must never be exceeded. This is implicitly insured if V_{IN} maximum is respected. If V_{IN} maximum cannot be respected, the injection current must be limited externally to the $I_{INJ(PIN)}$ value. A positive injection is induced by $V_{IN}>V_{DD}$ while a negative injection is induced by $V_{IN}<V_{SS}$.
Refer to section 12.2.2 on page 248 for more details. For PD7, refer to 'INJECTED CURRENT ON PD7" on page 303.

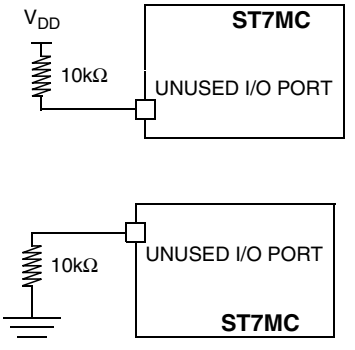
4. Configuration not recommended, all unused pins must be kept at a fixed voltage: using the output mode of the I/O for example or an external pull-up or pull-down resistor (see Figure 138). Static peak current value taken at a fixed V_{IN} value, based on design simulation and technology characteristics, 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 described in Figure 139). This data is based on characterization results, tested in production at V_{DD} max.

6. 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.

I/O PORT PIN CHARACTERISTICS (Cont'd)

Figure 138. 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 139. Typical I_{PU} vs. V_{DD} with $V_{IN}=V_{SS}$

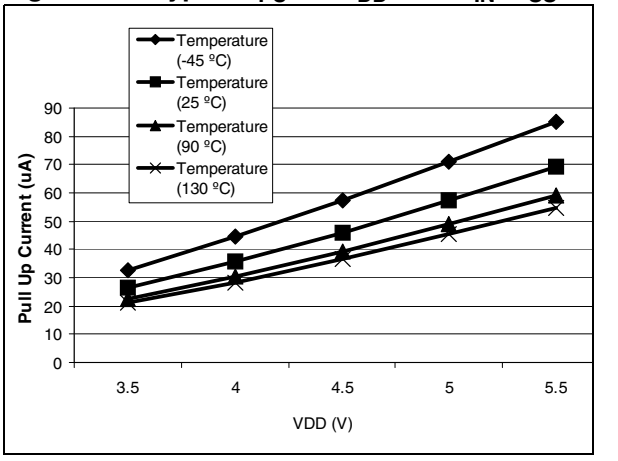
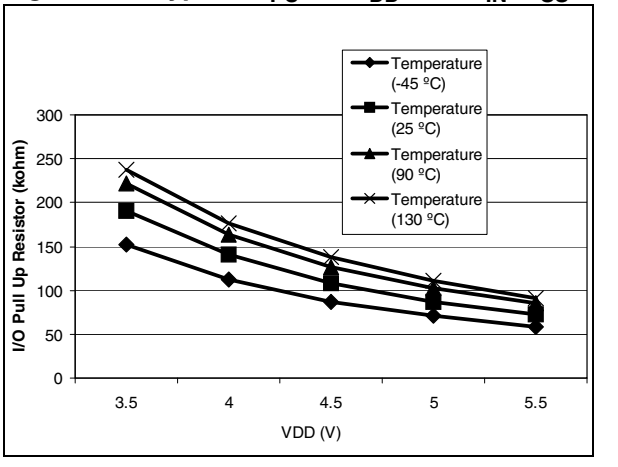


Figure 140. Typical R_{PU} vs. V_{DD} with $V_{IN}=V_{SS}$



I/O PORT PIN CHARACTERISTICS (Cont'd)

12.8.2 Output Driving Current

Subject to general operating conditions for V_{DD}, 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 141)	V_{DD}=5V	I_{IO}=+5mA		1.2	V
			I_{IO}=+2mA		0.5	
	Output low level voltage for a high sink I/O pin when 4 pins are sunk at same time (see Figure 142)		I_{IO}=+20mA, $T_A \leq 85°C$ $T_A \geq 85°C$		1.3 1.5	
			I_{IO}=+8mA		0.6	
V_{OH} 2)	Output high level voltage for an I/O pin when 4 pins are sourced at same time (see Figure 143)		I_{IO}=-5mA, $T_A \leq 85°C$ $T_A \geq 85°C$	V_{DD}-1.4 V_{DD}-1.6		
			I_{IO}=-2mA	V_{DD}-0.7		

Figure 141. Typical V_{OL} at V_{DD}=5V (standard)

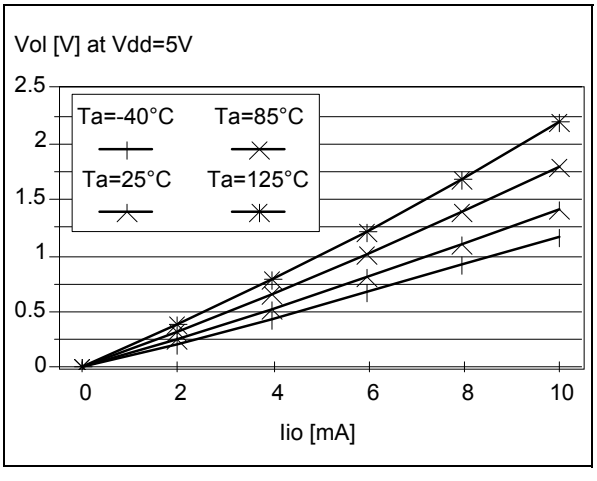


Figure 142. Typical V_{OL} at V_{DD}=5V (high-sink)

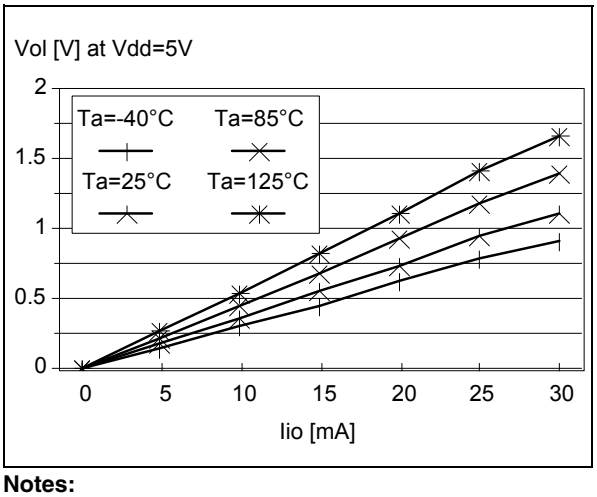
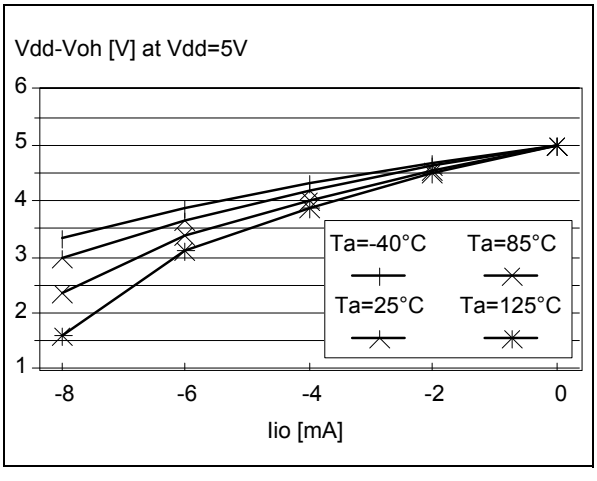


Figure 143. Typical V_{DD}-V_{OH} at V_{DD}=5V



Notes:

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 12.2.2 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 12.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}.

12.9 CONTROL PIN CHARACTERISTICS

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

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

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage					$0.3xV_{DD}$	V
V_{IH}	Input high level voltage			$0.7xV_{DD}$			
V_{hys}	Schmitt trigger voltage hysteresis [2]				1		V
V_{OL}	Output low level voltage [3]	V_{DD}=5V	I_{IO}=+5mA		0.5	1.2	V
			I_{IO}=+2mA		0.2	0.5	
I_{IO}	Driving current on $\overline{\text{RESET}}$ pin				2		mA
R_{ON}	Weak pull-up equivalent resistor	V_{IN}=V_{SS}, V_{DD}=5V		50	80	150	kΩ
$t_{w(RSTL)out}$	Generated reset pulse duration	Internal reset sources			30		µs
$t_{h(RSTL)in}$	External reset pulse hold time [4]			2.5			µs
$t_{g(RSTL)in}$	Filtered glitch duration [5]				450		ns

Notes:

1. Data based on characterization results, not tested in production.
2. Hysteresis voltage between Schmitt trigger switching levels.
3. The I_{IO} current sunk must always respect the absolute maximum rating specified in Section 12.2.2 and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VSS}.
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.

CONTROL PIN CHARACTERISTICS (Cont'd)

Figure 144. $\overline{\text{RESET}}$ pin protection when LVD is enabled.[1)2)3)4)]

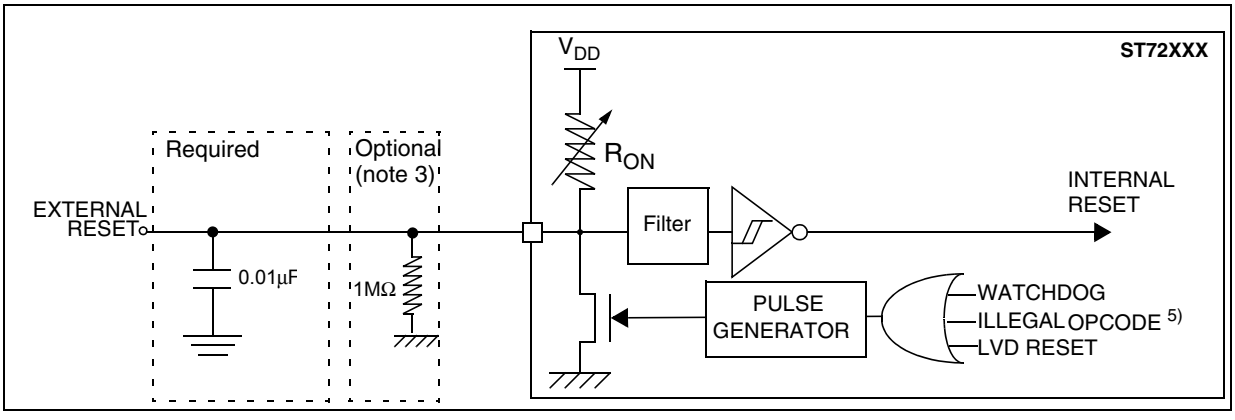
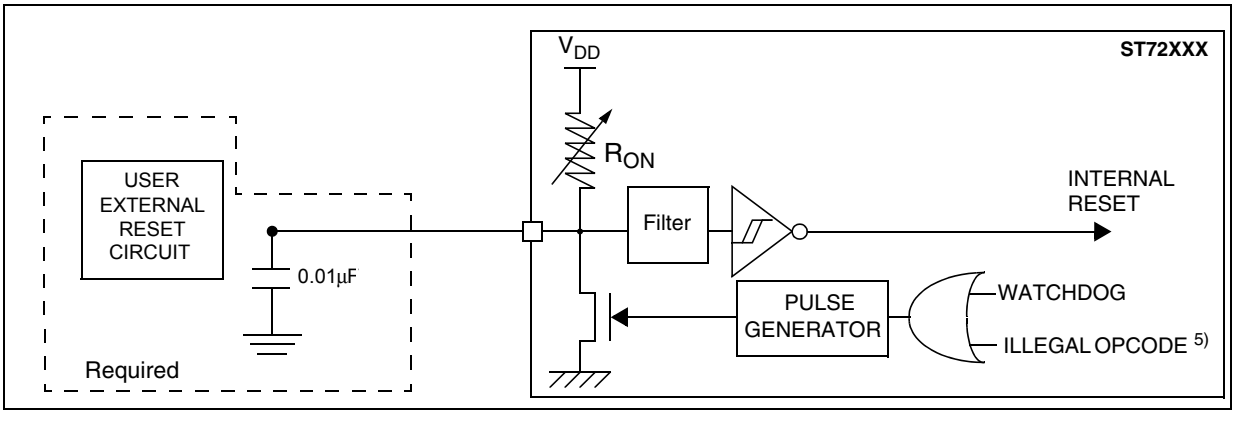


Figure 145. $\overline{\text{RESET}}$ pin protection when LVD is disabled.[1)]



Note 1:

- The reset network protects the device against parasitic resets.
- The output of the external reset circuit must have an open-drain output to drive the ST7 reset pad. Otherwise the device can be damaged when the ST7 generates an internal reset (LVD, illegal opcode or watchdog).
- 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 12.9.1 on page 267. Otherwise the reset will not be taken into account internally.
- 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 is less than the absolute maximum value specified for $I_{INJ(RESET)}$ in section 12.2.2 on page 248.

Note 2: When the LVD is enabled, it is recommended not to connect a pull-up resistor or capacitor. A 10nF pull-down capacitor is required to filter noise on the reset line.

Note 3: In case a capacitive power supply is used, it is recommended to connect a 1MΩ pull-down resistor to the $\overline{\text{RESET}}$ pin to discharge any residual voltage induced by the capacitive effect of the power supply (this will add 5µA to the power consumption of the MCU).

Note 4: Tips when using the LVD:

- 1. Check that all recommendations related to ICCCLK and reset circuit have been applied (see notes above)
- 2. Check that the power supply is properly decoupled (100nF + 10µF close to the MCU). Refer to AN1709 and AN2017. If this cannot be done, it is recommended to put a 100nF + 1MΩ pull-down on the $\overline{\text{RESET}}$ pin.
- 3. The capacitors connected on the RESET pin and also the power supply are key to avoid any start-up marginality. In most cases, steps 1 and 2 above are sufficient for a robust solution. Otherwise: replace 10nF pull-down on the $\overline{\text{RESET}}$ pin with a 5µF to 20µF capacitor."

Note 5: Please refer to "Illegal Opcode Reset" on page 244 for more details on illegal opcode reset conditions

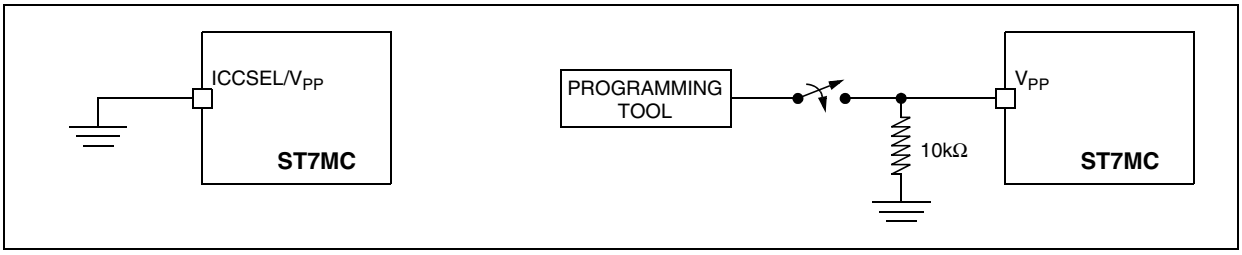
CONTROL PIN CHARACTERISTICS (Cont'd)

12.9.2 ICCSEL/V_{PP} Pin

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

Symbol	Parameter	Conditions	Min	Max	Unit
V_{IL}	Input low level voltage [1]		V_{SS}	0.2	V
V_{IH}	Input high level voltage [1] [2]	ICC mode entry	V_{DD}-0.1	12.6	
I_L	Input leakage current	$V_{IN}=V_{SS}$		±1	µA

Figure 146. Two typical Applications with V_{PP} Pin [3]



Notes:

1. Data based on design simulation and/or technology characteristics, not tested in production.
2. V_{PP} is also used to program the flash, refer to the Flash characteristics.
3. When the ICC mode is not required by the application ICCSEL/V_{PP} pin must be tied to V_{SS}.

12.10 TIMER PERIPHERAL CHARACTERISTICS

Subject to general operating conditions for V_{DD}, 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...).

12.10.1 8-Bit PWM-ART Auto-Reload Timer

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{res(PWM)}$	PWM resolution time		1			t_{CPU}
		f_{CPU}=8MHz	125			ns
f_{EXT}	ART external clock frequency		0		$f_{CPU}/2$	MHz
f_{PWM}	PWM repetition rate		0		$f_{CPU}/2$	
Res_{PWM}	PWM resolution				8	bit
V_{OS}	PWM/DAC output step voltage	V_{DD}=5V, Res=8-bits		20		mV

12.10.2 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}=8MHz	250			ns
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

12.11 COMMUNICATION INTERFACE CHARACTERISTICS

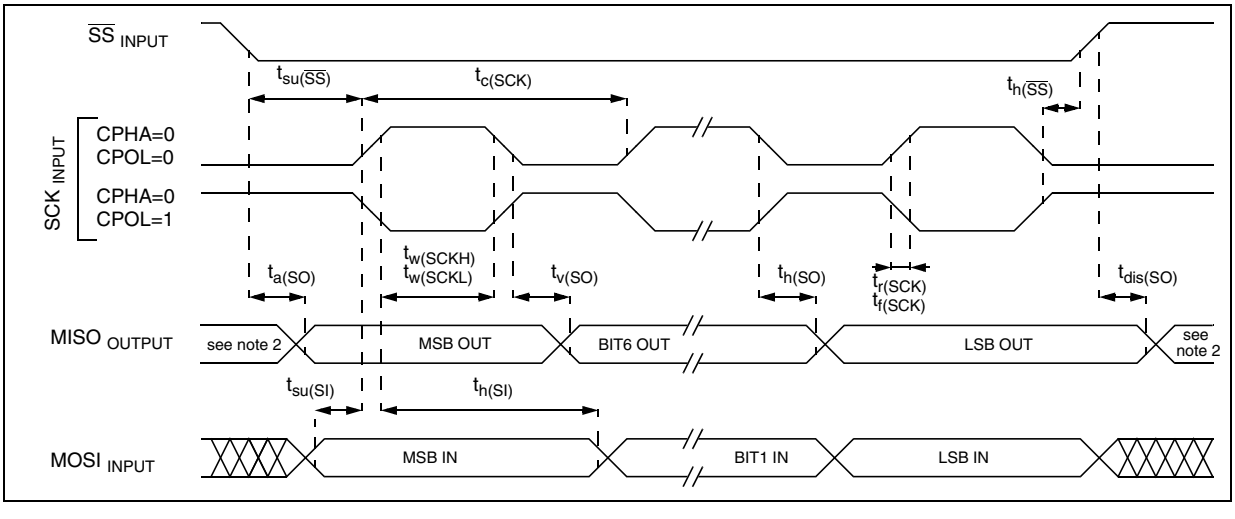
12.11.1 SPI - Serial Peripheral Interface

Subject to general operating conditions for V_{DD}, f_{OSC}, 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}=8MHz	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave f_{CPU}=8MHz	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		
$t_{w(SCKH)}$ $t_{w(SCKL)}$	SCK high and low time	Master Slave	100 90		
$t_{su(MI)}$ $t_{su(SI)}$	Data input setup time	Master Slave	100 100		
$t_{h(MI)}$ $t_{h(SI)}$	Data input hold time	Master Slave	100 100		
$t_{a(SO)}$	Data output access time	Slave	0	120	
$t_{dis(SO)}$	Data output disable time	Slave		240	
$t_{v(SO)}$	Data output valid time	Slave (after enable edge)		120	
$t_{h(SO)}$	Data output hold time		0		
$t_{v(MO)}$	Data output valid time	Master (after enable edge)		120	
$t_{h(MO)}$	Data output hold time		0		

Figure 147. 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: $0.3xV_{DD}$ and $0.7xV_{DD}$.
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 148. SPI Slave Timing Diagram with CPHA=1 [1)]

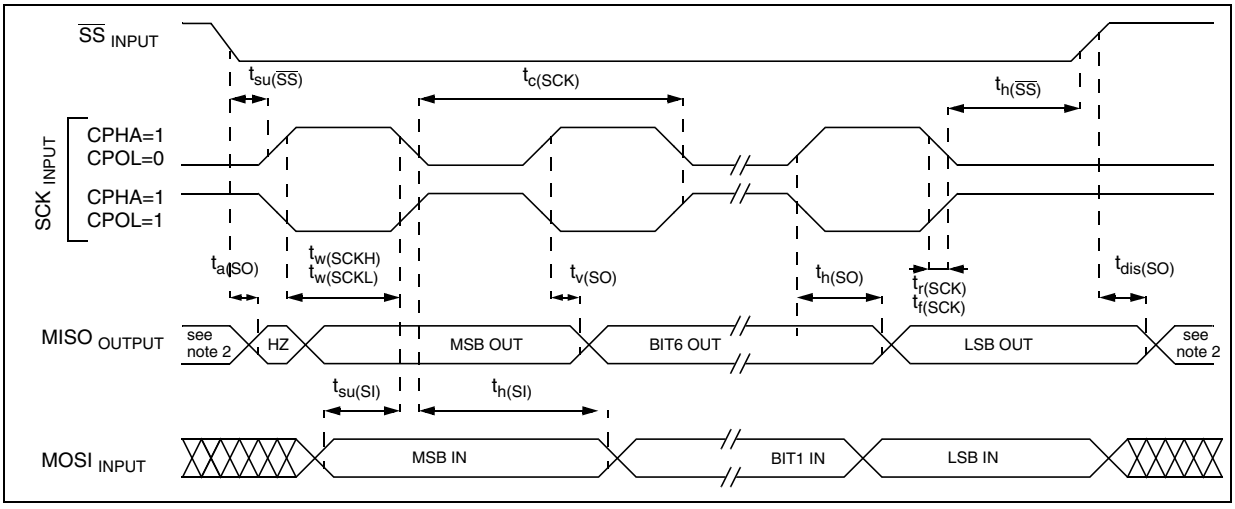
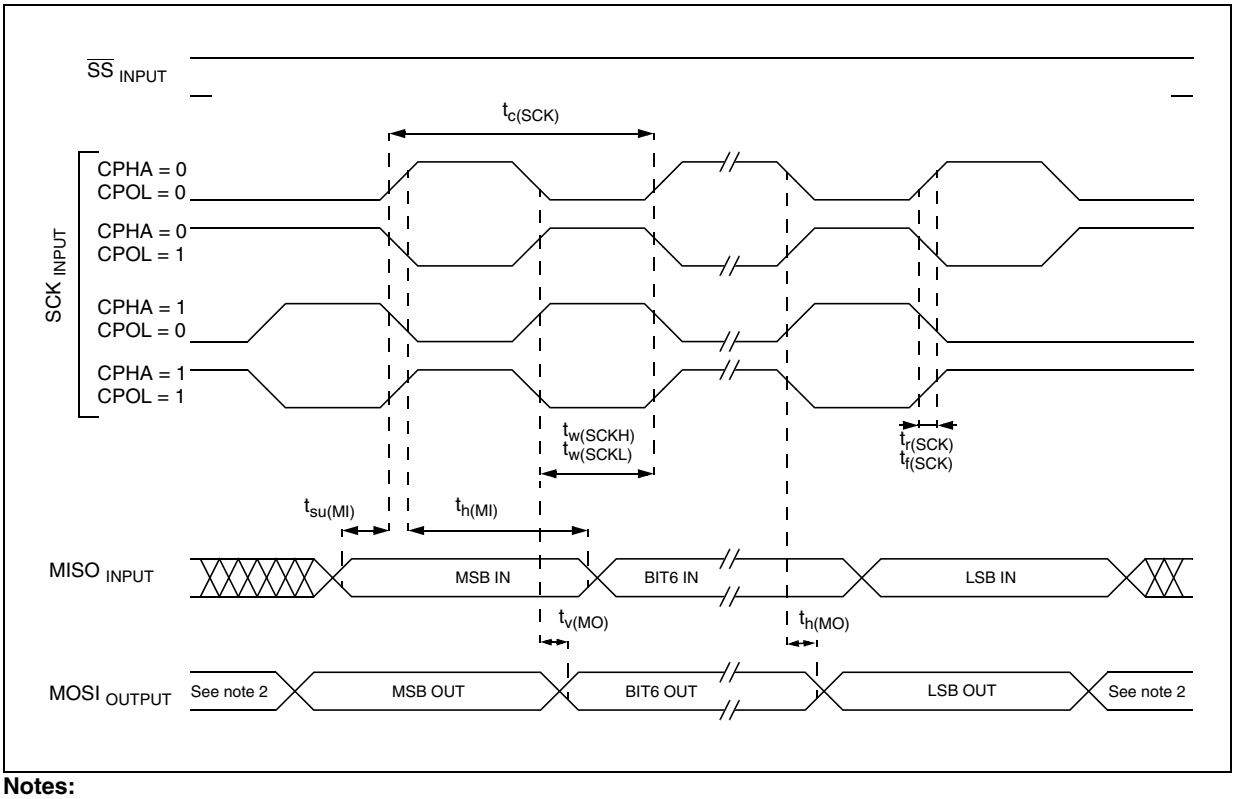


Figure 149. SPI Master Timing Diagram [1)]



Notes:

1. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.

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.

12.12 MOTOR CONTROL CHARACTERISTICS

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

12.12.1 Internal Reference Voltage

Symbol	Parameter	Conditions	Min	Typ [1]	Max	Unit
V_{REF}	Voltage threshold (VR [2:0] = 000)	VR [2:0] = 000		V_{DD}*0.04		V
		Example: V_{DD} -V_{SSA} = 5V		0.2		
	Voltage threshold (VR [2:0] = 001)	VR [2:0]= 001		V_{DD}*0.12		
		Example: V_{DD} -V_{SSA} = 5V		0.6		
	Voltage threshold (VR [2:0] = 010)	VR [2:0] = 010		V_{DD}*0.2		
		Example: V_{DD} -V_{SSA} = 5V		1.0		
	Voltage threshold (VR [2:0] = 011)	VR [2:0]= 011		V_{DD}*0.3		
		Example: V_{DD} -V_{SSA} = 5V		1.5		
	Voltage threshold (VR [2:0] = 100)	VR [2:0] = 100		V_{DD}*0.4		
		Example: V_{DD} -V_{SSA} = 5V		2.0		
	Voltage threshold (VR [2:0] = 101)	VR [2:0]= 101		V_{DD}*0.5		
		Example: V_{DD} -V_{SSA} = 5V		2.5		
	Voltage threshold (VR [2:0] = 110)	VR [2:0] = 110		V_{DD}*0.7		
		Example: V_{DD} -V_{SSA} = 5V		3.5		
$\Delta V_{REF}/V_{REF}$	Tolerance on V_{REF}			2.5	10	%

Note:

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

MOTOR CONTROL CHARACTERISTICS (Cont'd)

12.12.2 Input Stage (comparator + sampling)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IN}	Comparator input voltage range		V_{SSA} - 0.1		V_{DD} + 0.1	V
V_{offset}	Comparator offset error			5	40 [1]	mV
I_{offset}	Input offset current				1	µA
t_{propag}	Comparator propagation delay			35	100	ns
$t_{startup}$	Start-up filter duration[2]	Time waited before sampling when comparator is turned ON, i.e. CKE=1 or DAC=1 (with f_{PERIPH} = 4MHz)	3			µs
$t_{sampling}$	Digital sampling delay [3]	Time needed to generate a capture in tachogenerator mode as soon as the MCI input toggles	4 / f_{mtc}			
		Time needed to capture MTIM in MZREG (BEMF) when sampling during PWM signal OFF time as soon as MCO becomes ON	3 / f_{mtc} (see Figure 150)			
		Time needed to set/reset the HST bit when sampling during PWM signal OFF time as soon as MCO becomes ON (BEMF)	1 / f_{mtc} (see Figure 150)			
		Time needed to generate Z event (MTIM captured in MZREG) as soon as the comparator toggles (when sampling at f_{SCF})	1 / f_{SCF} + 3 / f_{mtc} (see Figure 151)			
		Time needed to generate D event (MTIM captured in MDREG) as soon as the comparator toggles	1 / f_{SCF} + 3 / f_{mtc} (see Figure 151)			
		Time needed to set/reset the HST bit when sampling during PWM signal ON time after a delay (DS>0) as soon as MCO becomes ON	Delay programmed in DS bits (MCONF) +1 / f_{mtc} (see Figure 152)			
		Time needed to generate Z event (MTIM in MZREG) when sampling during PWM signal ON time after a delay (DS>0) as soon as MCO becomes ON	Delay programmed in DS bits (MCONF) + 3 / f_{mtc} (see Figure 152)			
		Time needed to generate Z event (MTIM captured in MZREG) when sampling during PWM signal ON time at f_{SCF} after a delay (DS>0)	Delay programmed in DS bits (MCONF) + 1 / f_{SCF} + 3 / f_{mtc} (see Figure 152)			

Note:

1. The comparator accuracy is dependent of the environment. The offset value is given for a comparison done with all digital I/Os stable. Negative injection current on the I/Os close to the inputs may reduce the accuracy. In particular care must be taken to avoid switching on I/Os close to the inputs when the comparator is in use. This phenomenon is even more critical when a big external serial resistor is added on the inputs.
2. This filter is implemented to wait for comparator stabilization and avoid any wrong information during start-up.
3. This delay represents the number of clock cycles needed to generate an event as soon as the comparator output or MCO outputs change.

Example: In tachogenerator mode, this means that capture is performed on the 4th clock cycle after comparator commutation., i.e. there is a variation of ($1/f_{mtc}$) or ($1 / f_{SCF}$) depending on the case.

MOTOR CONTROL CHARACTERISTICS (Cont'd)

Figure 150. Example 1: Waveforms for Zero-crossing Detection with Sampling at the end of PWM off-time

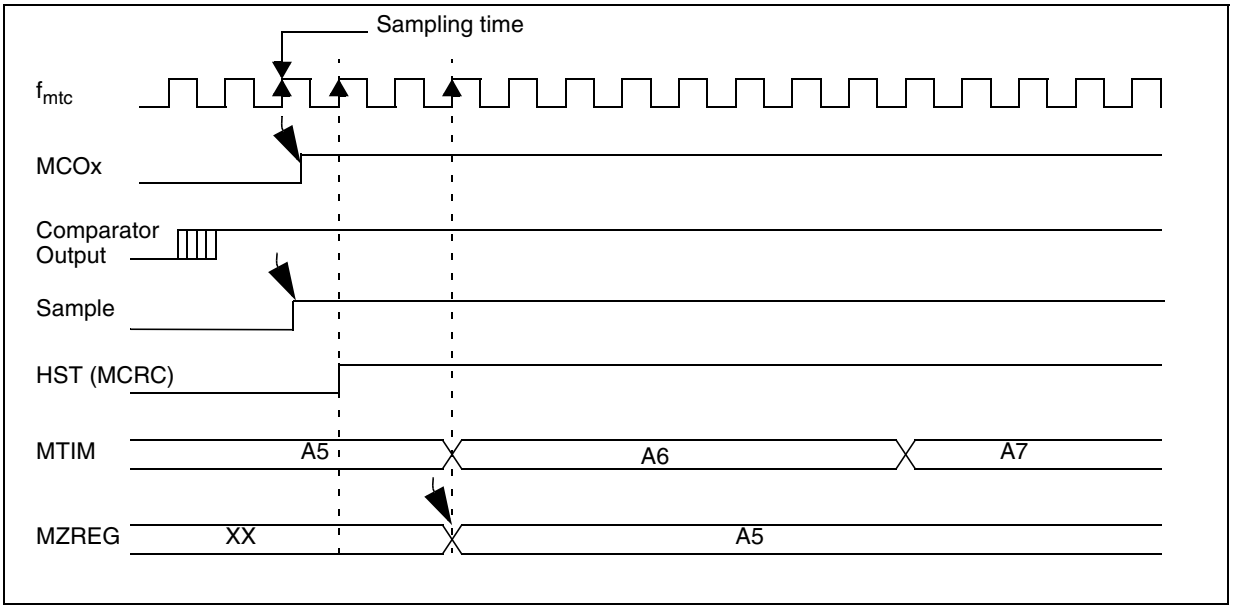
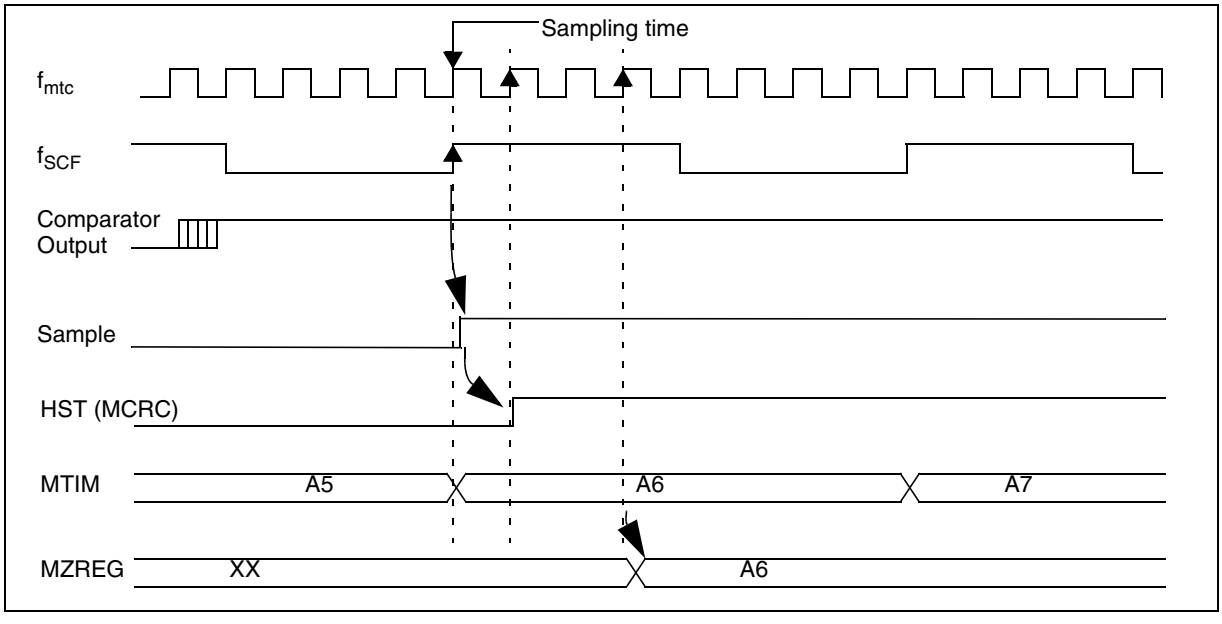


Figure 151. Example 2: Waveforms for Zero-crossing Detection with Sampling at f_{SCF}



MOTOR CONTROL CHARACTERISTICS (Cont'd)

Figure 152. Example 3: Waveforms for Zero-crossing Detection with Sampling after a Delay during PWM On-time

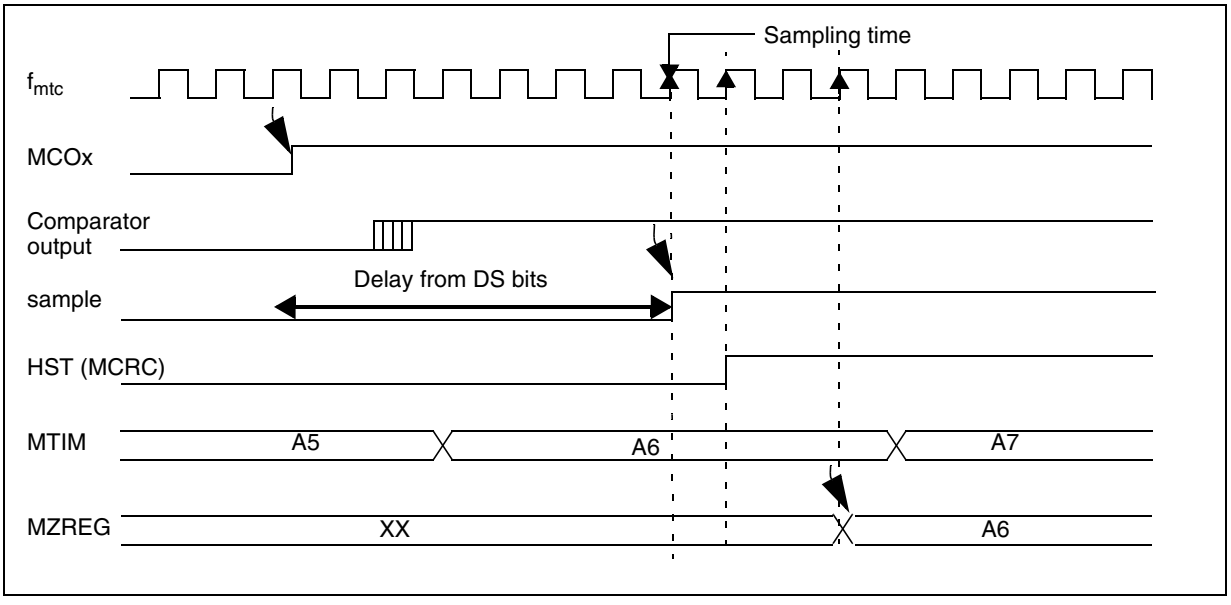
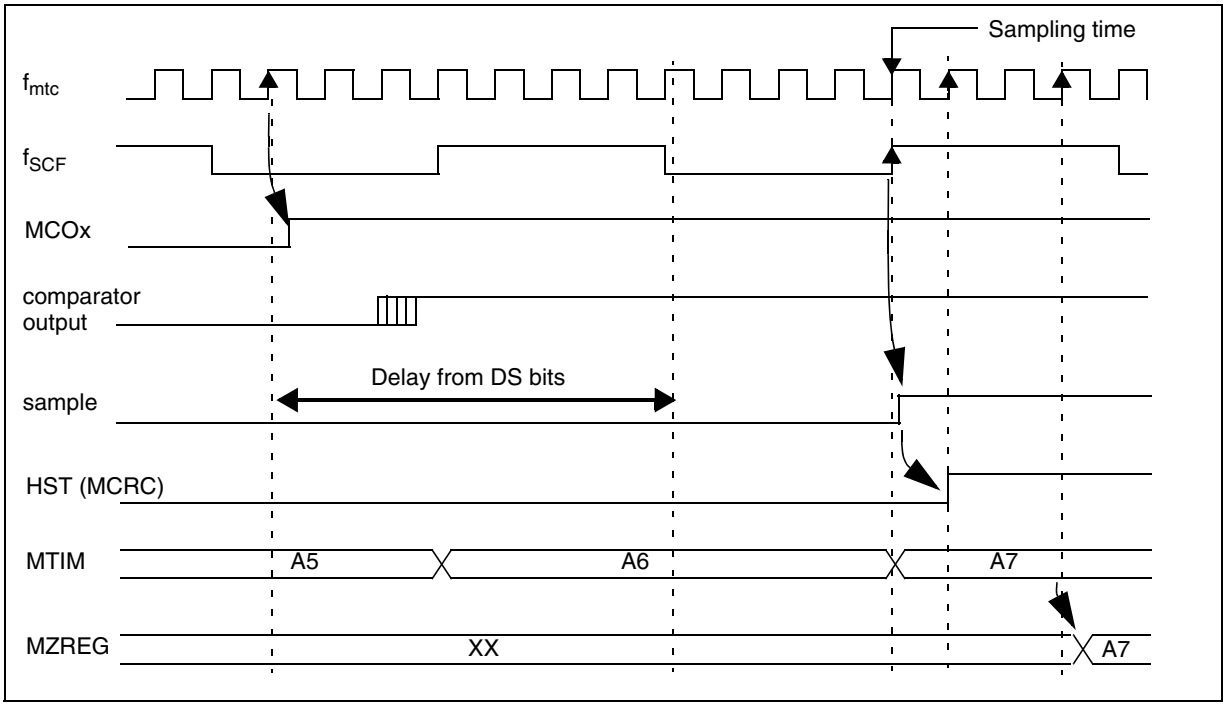
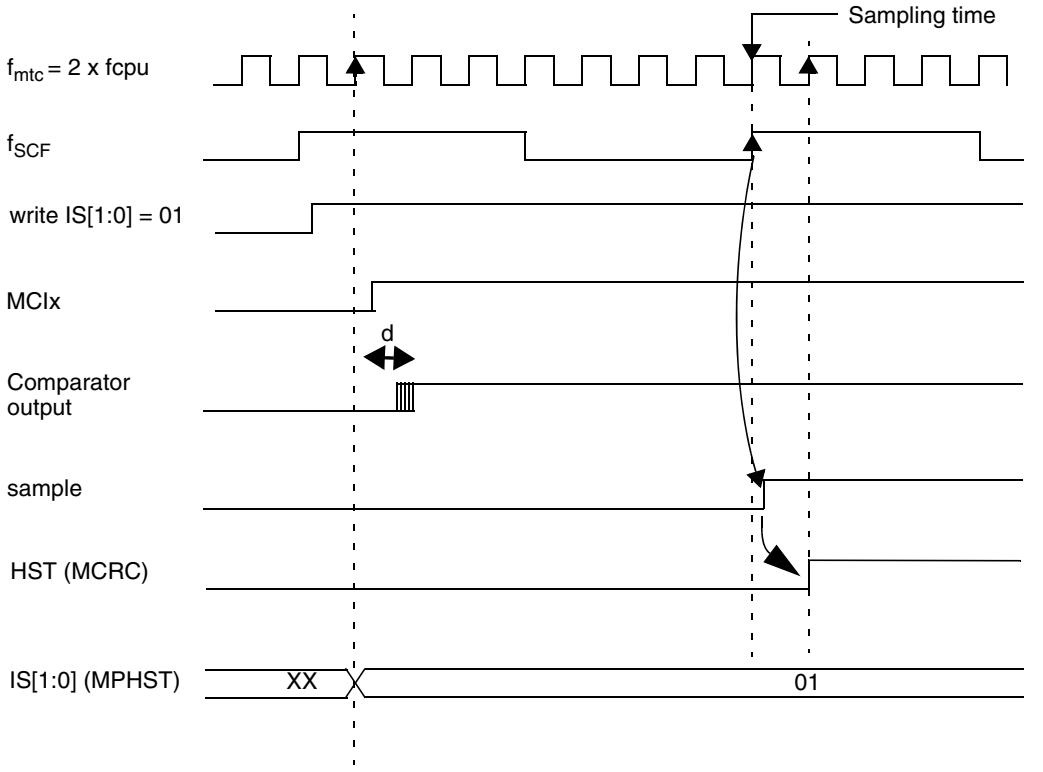


Figure 153. Example 4: Waveforms for zero-crossing detection with sampling after a delay at f_{SCF}



MOTOR CONTROL CHARACTERISTICS (Cont'd)

Figure 154. Example 5: Waveforms for sensor HST update timing diagram for a newly selected phase input



Sampling of sensor input is done at f_{SCF}, so after writing IS[1:0] in the MPHST register for input selection, you have to wait:

1) $1/f_{mtc}$ to resynchronize the data and set the MCIx bit
2) 100ns corresponding to the comparator & multiplexer propagation delay (d)
3) $1/f_{SCF}$ to sample the comparator output
4) 1/fmtc to resynchronize the data and set the HST bit

Example with $f_{PERIPH} = f_{mtc}$ = 4MHz :

SCF[1:0]	00	01	10	11
f_{SCF}	1MHz	500kHz	250kHz	125kHz
Secure waiting time to read HST	2µs	3µs	5µs	9µs

MOTOR CONTROL CHARACTERISTICS (Cont'd)

12.12.3 Input Stage (Current Feedback Comparator + Sampling)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IN}	Comparator input voltage range		V_{SSA} - 0.1		V_{DD} + 0.1	V
V_{offset}	Comparator offset error			5	40 [1]	mV
I_{offset}	Input offset current				1	µA
t_{propag}	Comparator propagation delay [1]			35	100	ns
$t_{startup}$	Start-up filter duration [2]	Time waited before sampling when comparator is turned ON, i.e. CKE=1 or DAC=1 (with f_{PERIPH} = 4MHz)	3			µs
$t_{sampling}$	Digital sampling delay [3]	Time needed to turn OFF the MCOs when comparator output rises (CFF=0)	4 / f_{MTC} (see Figure 155)			
		Time between a comparator toggle (current loop event) and bit CL becoming set (CFF=0)	2 / f_{MTC} (see Figure 155)			
		Time needed to turn OFF the MCOs when comparator output rises (CFF=x)	(1+x) * (4 / f_{PERIPH}) + (3 / f_{mtc}) (see Figure 156)			
		Time between a comparator toggle (current loop event) and bit CL becoming set (CFF=x)	(1+x) * (4 / f_{PERIPH}) + (1 / f_{mtc}) (see Figure 156)			

Notes:

1. The comparator accuracy depends on the environment. In particular, the following cases may reduce the accuracy of the comparator and must be avoided:
– Negative injection current on the I/Os close to the comparator inputs
– Switching on I/Os close to the comparator inputs
– Negative injection current on not used comparator input (MCCFI0 or MCCFI1)
– Switching with a high dV/dt on not used comparator input (MCCFI0 or MCCFI1)
These phenomena are even more critical when a big external serial resistor is added on the inputs.

2. This filter is implemented to wait for comparator stabilization and avoid any wrong information during start-up.

3.This delay represents the number of clock cycles needed to generate an event as soon as the comparator ouput changes.
Example: When CFF=0 (detection is based on a single detection), MCO outputs are turned OFF at the 4th clock cycle after comparator commutation, i.e. there is a variation of ($1/f_{mtc}$) or (4 / f_{PERIPH}) depending on the case.

MOTOR CONTROL CHARACTERISTICS (Cont'd)

Figure 155. Example 1: Waveforms For Overcurrent Detection with Current Feedback Filter OFF

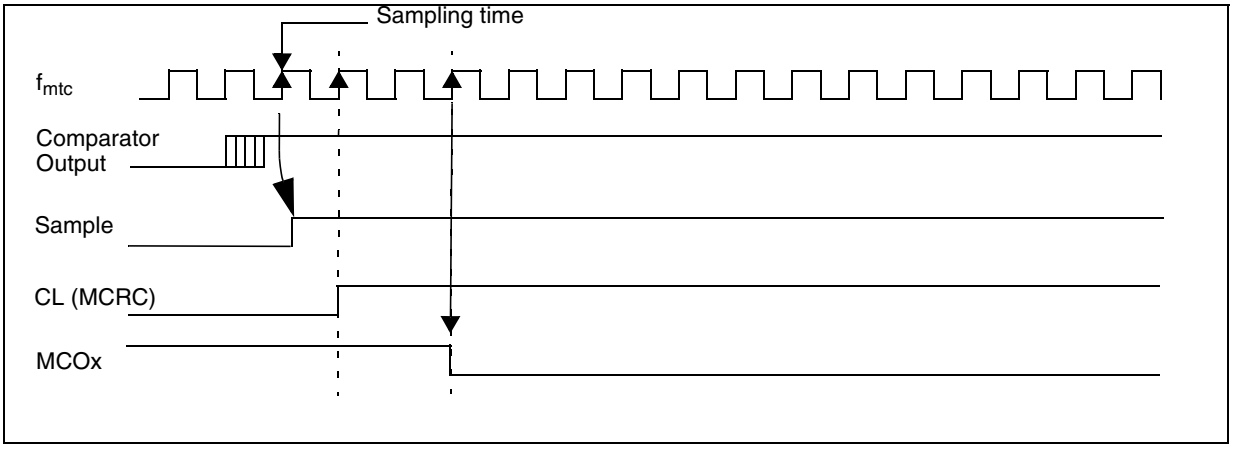
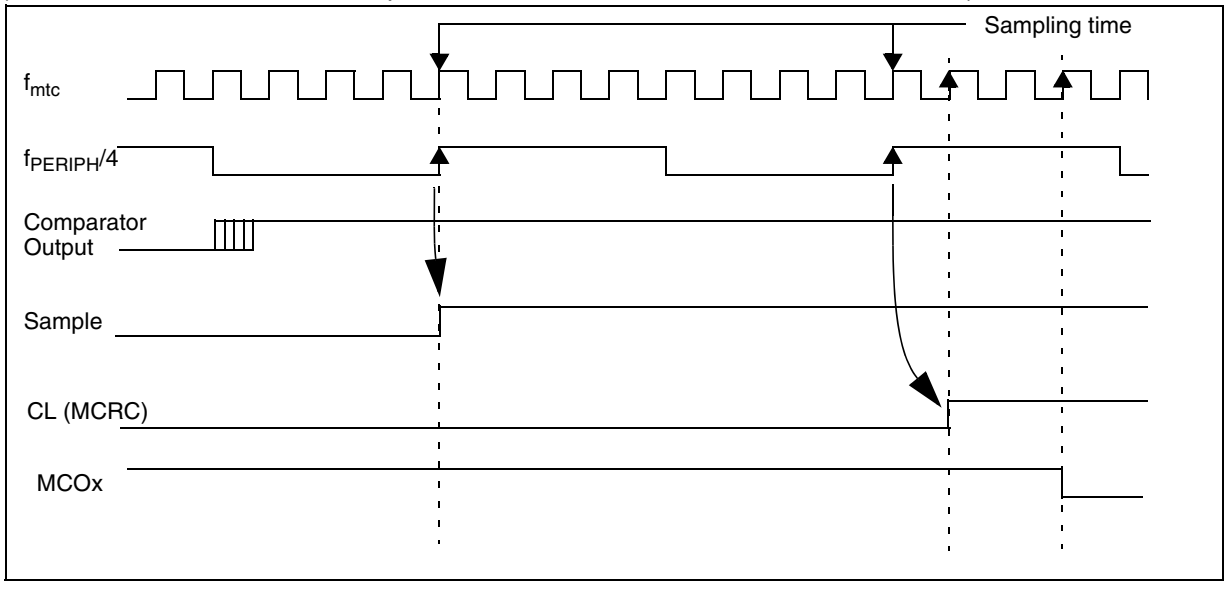


Figure 156. Example 2: waveforms for overcurrent detection with current feedback filter ON
(CFF=001 => 2 consecutive samples are needed to validate the overcurrent event)



12.13 OPERATIONAL AMPLIFIER CHARACTERISTICS

Subject to general operating conditions for f_{OSC}, and T_A unless otherwise specified.
(T_A = -40..+125°C, V_{DD}-V_{SSA} = 4.5..5.5V unless otherwise specified

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
R_L	Resistive Load (max 500µA @ 5V)			10			kΩ
C_L	Capacitive Load at V_{OUT} pin					150	pF
V_{CMIR}	Common Mode Input Range			V_{SSA}		V_{DD}/2	V
V_{io}	Input Offset Voltage (+ or -) 3)	After calibration, V_{IC} =1V			2.5	10 4)	mV
ΔV_{io}	Input Offset Voltage Drift from the calibrated Voltage, temperature conditions	with respect to temperature				8.5 5)	µV/°C
		with respect to common mode input				1 5)	mV/V
		with respect to supply				3.1 5)	mV/V
CMR	Common Mode Rejection Ratio	HIGHGAIN=0 @ 100kHz			74		dB
SVR	Supply Voltage Rejection Ratio	@ 100kHz		50 2)	65		dB
A_{vd}	Voltage Gain	R_L=10kΩ		(1.5) 2)	12		V/mV
V_{SAT_OH}	High Level Output Saturation Voltage (V_{DD}-V_{OUT})	R_L=10kΩ			60	90 2)	mV
V_{SAT_OL}	Low Level Output Saturation Voltage	R_L=10kΩ			30	90 2)	mV
GBP	Gain Bandwidth Product	V_{OAP} = V_{DD} / 2, measured at 0dB gain, feedback resistor ratio = 10 (20dB inverting configuration)	HIGHGAIN=0	1.3 2)	1.5	2 2)	MHz
			HIGHGAIN=1	2.6 2)	3	4 2)	
SR^+	Slew Rate while rising	HIGHGAIN=0 (A_{VCL}=1, R_L=10kΩ, C_L=150pF, V_i=1.75V to 2.75V) 1)		1 2)	2		V/µs
SR^-	Slew Rate while falling	HIGHGAIN=0 (A_{VCL}=1, R_L=10kΩ, C_L=150pF, V_i=1.75V to 2.75V) 1)		2.5 2)	7.5		V/µs
Φm	Phase Margin	HIGHGAIN=0			73		degrees
		HIGHGAIN=1			75		
T_{wakeup}	Wakeup time for the opamp from off state			0.8 6)		1.6 6)	µs

Note:

1. A_{VCL} = closed loop gain
2. Data based on characterization results, not tested in production.
3. after offset compensation has been performed.
4. The amplifier accuracy is dependent on the environment. The offset value is given for a measurement done with all digital I/Os stable. Negative injection current on the I/Os close to the inputs may reduce the accuracy. In particular care must be taken to avoid switching on I/Os close to the inputs when the opamp is in use. This phenomenon is even more critical when a big external serial resistor is added on the inputs.
5. The Data provided from simulations (not tested in production) to guide the user when re-calibration is needed.
6. The Data provided from simulations (not tested in production).

12.14 10-BIT ADC CHARACTERISTICS

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

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{AREF}	Analog Reference Voltage		3		V_{DD}	V
f_{ADC}	ADC clock frequency				4	MHz
V_{AIN}	Conversion voltage range [1]		V_{SSA}		V_{AREF}	V
I_{lkg}	Positive input leakage current for analog input				±1	µA
	Negative input leakage current on analog pins	$V_{IN}<V_{SS,}$ | I_{IN} |< 400µA on adjacent analog pin		5	6	µA
R_{AIN}	External input impedance				see Figure 157 and Figure 158 [2)3)4)]	kΩ
C_{AIN}	External capacitor on analog input					pF
f_{AIN}	Variation freq. of analog input signal					Hz
C_{ADC}	Internal sample and hold capacitor			6		pF
t_{ADC}	Conversion time (Sample+Hold)	f_{CPU}=8MHz, f_{ADC}=4MHz, ADSTS bit in MCCBCR register = 0		3.5		µs
	- Sample capacitor loading time - Hold conversion time			4 10		$1/f_{ADC}$
	Conversion time (Sample+Hold)	f_{CPU}=8MHz, f_{ADC}=4MHz, ADSTS bit in MCCBCR register = 1		6.5		µs
	- Sample capacitor loading time - Hold conversion time			16 10		$1/f_{ADC}$
R_{AREF}	Analog Reference Input Resistor			11		kΩ

Figure 157. R_{AIN} max. vs f_{ADC} with C_{AIN}=0pF[3)]

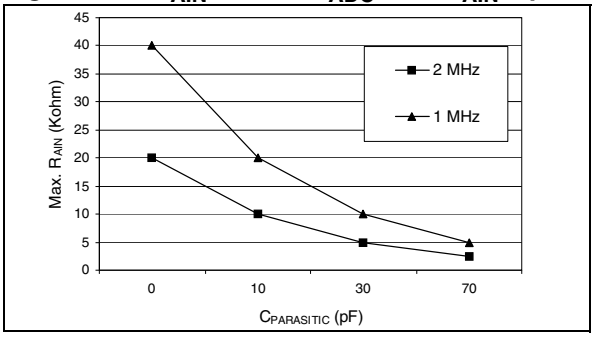
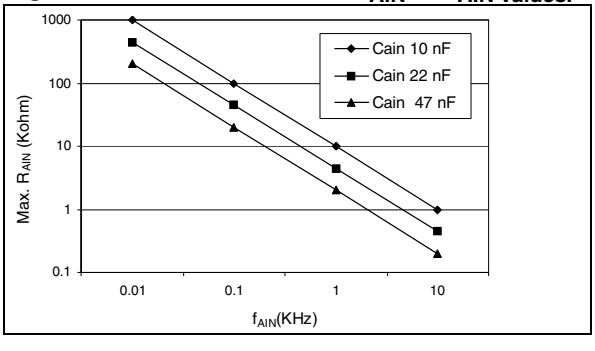
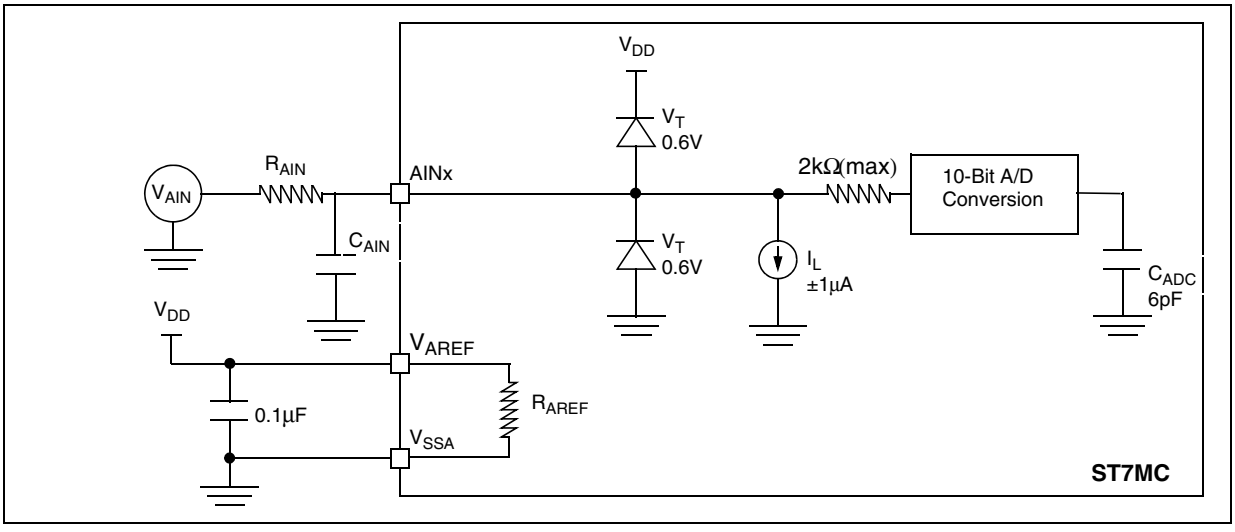


Figure 158. Recommended C_{AIN} & R_{AIN} values.[4)]



10-BIT ADC CHARACTERISTICS (Cont'd)

Figure 159. Typical Application with ADC



Notes:

1. When V_{SSA} pins are not available on the pinout, the ADC refer to V_{SS}.
2. Any added external serial resistor will downgrade the ADC accuracy (especially for resistance greater than 10kΩ). Data based on characterization results, not tested in production.
3. $C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high $C_{PARASITIC}$ value will downgrade conversion accuracy. To remedy this, f_{ADC} should be reduced.
4. This graph shows that depending on the input signal variation (f_{AIN}), C_{AIN} can be increased for stabilization time and decreased to allow the use of a larger serial resistor (R_{AIN}).

10-BIT ADC CHARACTERISTICS (Cont'd)

12.14.1 Analog Power Supply and Reference Pins

Depending on the MCU pin count, the package may feature separate V_{AREF} and V_{SSA} analog power supply pins. These pins supply power to the A/D converter cell and function as the high and low reference voltages for the conversion. In some packages, V_{AREF} and V_{SSA} pins are not available (refer to section 2 on page 6). In this case the analog supply and reference pads are internally bonded to the V_{DD} and V_{SS} pins.

Separation of the digital and analog power pins allow board designers to improve A/D performance. Conversion accuracy can be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines (see Section 12.14.2 General PCB Design Guidelines).

12.14.2 General PCB Design Guidelines

To obtain best results, some general design and layout rules should be followed when designing the application PCB to shield the noise-sensitive, analog physical interface from noise-generating CMOS logic signals.

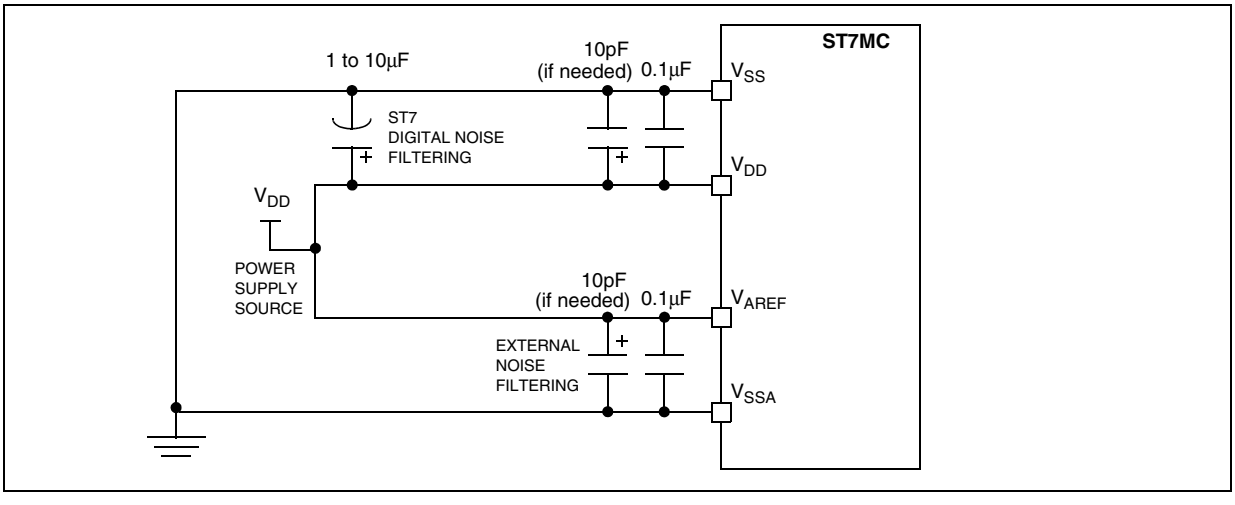
– Use separate digital and analog planes. The analog ground plane should be connected to the digital ground plane via a single point on the PCB.

– Filter power to the analog power planes. It is recommended to connect capacitors, with good high frequency characteristics, between the power and ground lines, placing 0.1µF and optionally, if needed 10pF capacitors as close as possible to the ST7 power supply pins and a 1 to 10µF capacitor close to the power source (see Figure 160).

– The analog and digital power supplies should be connected in a star network. Do not use a resistor, as V_{AREF} is used as a reference voltage by the A/D converter and any resistance would cause a voltage drop and a loss of accuracy.

– Properly place components and route the signal traces on the PCB to shield the analog inputs. Analog signals paths should run over the analog ground plane and be as short as possible. Isolate analog signals from digital signals that may switch while the analog inputs are being sampled by the A/D converter. Do not toggle digital outputs on the same I/O port as the A/D input being converted.

Figure 160. Power Supply Filtering



10-BIT ADC CHARACTERISTICS (Cont'd)

ADC Accuracy with V_{DD}=5.0V

Symbol	Parameter	Conditions	Typ	Max [2]	Unit
$\|E_T\|$	Total unadjusted error [1]	V_{AREF}=3.0V to 5.0V, f_{CPU}=8MHz, f_{ADC}=4MHz, R_{AIN}<10kΩ	4		LSB
$\|E_O\|$	Offset error [1]		2.5	4	
$\|E_G\|$	Gain Error [1]		2	4	
$\|E_D\|$	Differential linearity error [1]		2	4.5	
$\|E_L\|$	Integral linearity error [1]		2	4.5	

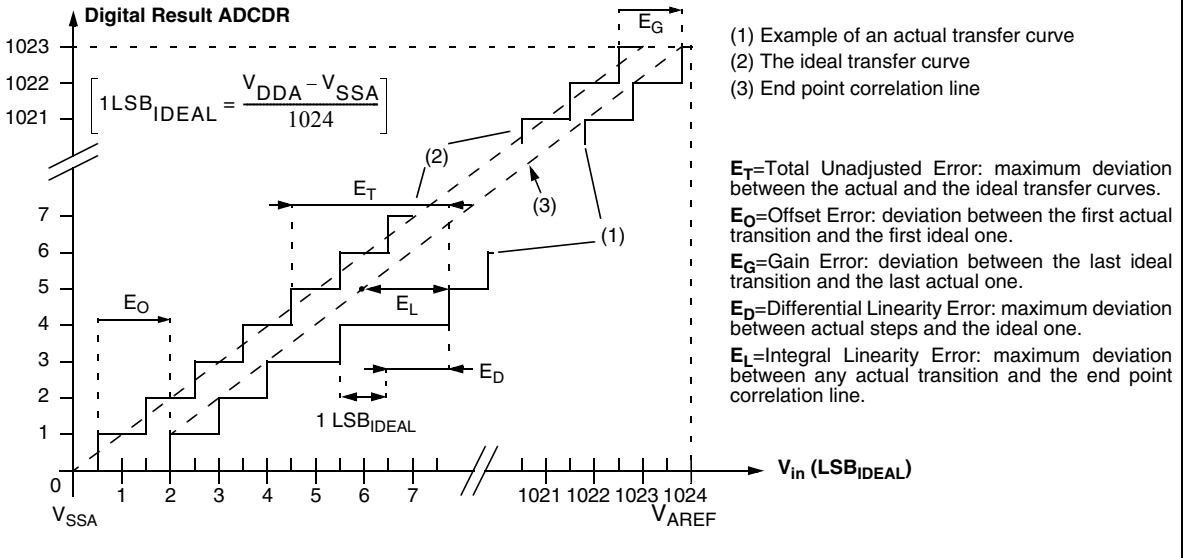
Notes:

1. ADC Accuracy vs. Negative Injection Current: Injecting negative current may reduce the accuracy of the conversion being performed on another analog input. The effect of negative injection current on analog pins is specified in Section 12.14.

Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in Section 12.8 does not affect the ADC accuracy.

2. Data based on characterization results, monitored in production to guarantee 99.73% within ± max value from -40°C to 125°C (± 3σ distribution limits).

Figure 161. ADC Accuracy Characteristics



Notes:

1. ADC Accuracy vs. Negative Injection Current:
For I_{INJ-}=0.8mA, the typical leakage induced inside the die is 1.6µA and the effect on the ADC accuracy is a loss of 4 LSB for each 10KΩ increase of the external analog source impedance. This effect on the ADC accuracy has been observed under worst-case conditions for injection:
- negative injection
- injection to an Input with analog capability, adjacent to the enabled Analog Input
- at 5V V_{DD} supply, and worst case temperature.

2. Data based on characterization results with T_A=25°C.

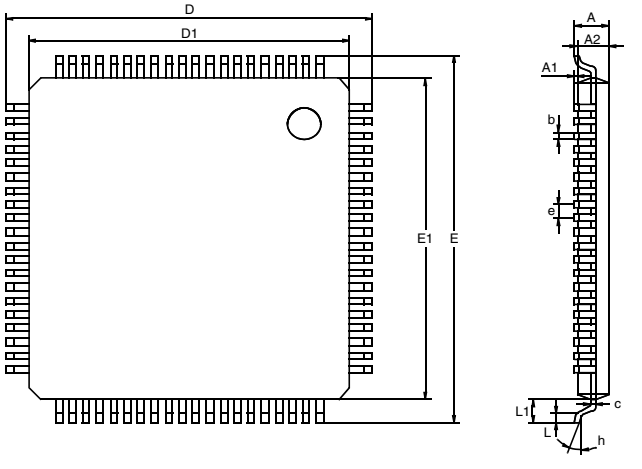
3. Data based on characterization results over the whole temperature range, monitored in production.

13 PACKAGE CHARACTERISTICS

In order to meet environmental requirements, ST offers these devices in different grades of ECOPACK® packages, depending on their level of environmental compliance. ECOPACK® specifications, grade definitions and product status are available at: www.st.com. ECOPACK® is an ST trademark.

13.1 PACKAGE MECHANICAL DATA

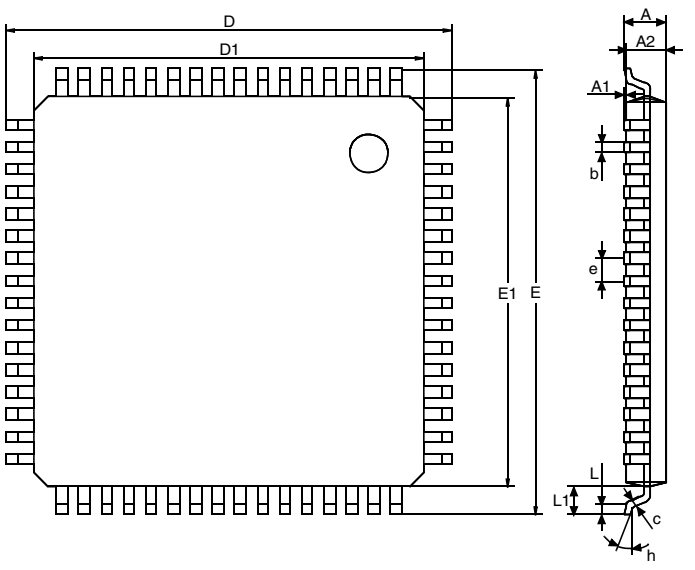
Figure 162. 80-Pin Low Profile Quad Flat Package



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.22	0.32	0.38	0.0087	0.0126	0.0150
C	0.09		0.20	0.0035		0.0079
D		16.00			0.6299	
D1		14.00			0.5512	
E		16.00			0.6299	
E1		14.00			0.5512	
e		0.65			0.0256	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	80					

PACKAGE CHARACTERISTICS (Cont'd)

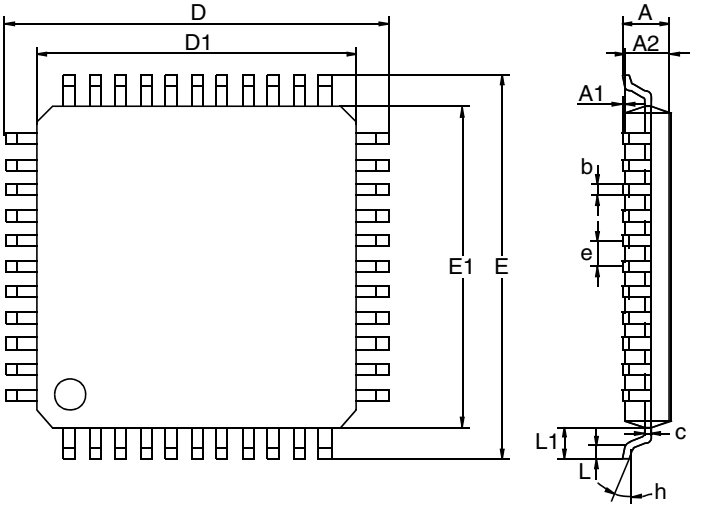
Figure 163. 64-Pin Low Profile Quad Flat Package (14x14)



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.30	0.37	0.45	0.0118	0.0146	0.0177
c	0.09		0.20	0.0035		0.0079
D		16.00			0.6299	
D1		14.00			0.5512	
E		16.00			0.6299	
E1		14.00			0.5512	
e		0.80			0.0315	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	64					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 164. 44-Pin Low Profile Quad Flat Package

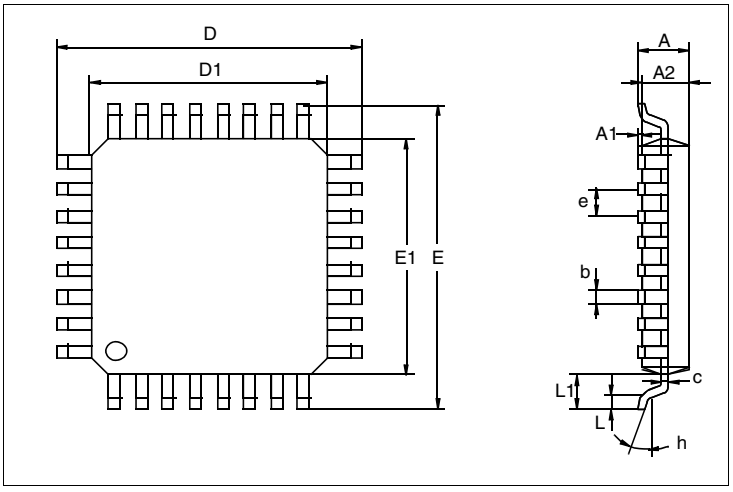


Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.30	0.37	0.45	0.0118	0.0146	0.0177
C	0.09		0.20	0.0035		0.0079
D		12.00			0.4724	
D1		10.00			0.3937	
E		12.00			0.4724	
E1		10.00			0.3937	
e		0.80			0.0315	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	44					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

PACKAGE CHARACTERISTICS (Cont'd)

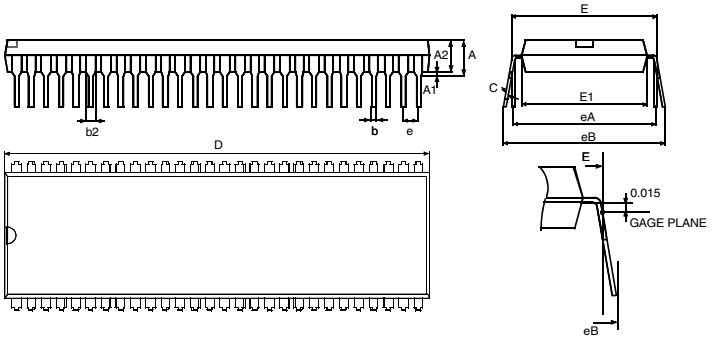
Figure 165. 32-Pin Low Profile Quad Flat Package



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			1.60			0.0630
A1	0.05		0.15	0.0020		0.0059
A2	1.35	1.40	1.45	0.0531	0.0551	0.0571
b	0.30	0.37	0.45	0.0118	0.0146	0.0177
C	0.09		0.20	0.0035		0.0079
D		9.00			0.3543	
D1		7.00			0.2756	
E		9.00			0.3543	
E1		7.00			0.2756	
e		0.80			0.0315	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.0177	0.0236	0.0295
L1		1.00			0.0394	
	Number of Pins					
N	32					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

Figure 166. 56-Pin Plastic Dual In-Line Package, Shrink 600-mil Width



Dim.	mm			inches[1]		
	Min	Typ	Max	Min	Typ	Max
A			6.35	0.2500		
A1	0.38			0.0150		
A2	3.18		4.95	0.1252		0.1949
b		0.41			0.0161	
b2		0.89			0.0350	
C	0.20		0.38	0.0079		0.0150
D	50.29		53.21	1.9799		2.0949
E		15.01			0.5909	
E1	12.32		14.73	0.4850		0.5799
e		1.78			0.0701	
eA		15.24			0.6000	
eB			17.78			0.7000
L	2.92		5.08	0.1150		0.2000
	Number of Pins					
N	56					

Note 1. Values in inches are converted from mm and rounded to 4 decimal digits.

PACKAGE CHARACTERISTICS (Cont'd)

13.2 THERMAL CHARACTERISTICS

Symbol	Ratings	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient) LQFP80 14x14 LQFP64 14x14 LQFP44 10x10 LQFP32 7x7 SDIP32 400mil SDIP56 600mil	 55 55 68 80 63 45	°C/W
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 ($I_{DD} \times V_{DD}$) and P_{PORT} is the port power dissipation depending on the ports used in the application.

13.3 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 and SO 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 90. Soldering Compatibility (wave and reflow soldering process)

Package	Plating material devices	Pb solder paste	Pb-free solder paste
SDIP & PDIP	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.

14 ST7MC DEVICE CONFIGURATION AND ORDERING INFORMATION

Each device is available for production in ROM versions and in user programmable versions (FLASH) as well as in factory coded versions (FASTROM). ST7MC are ROM devices. ST7PMC devices are Factory Advanced Service Technique ROM (FASTROM) versions: they are programmed Flash devices.

ST7FMC Flash devices are shipped to customers with a default content (FFh), while ROM/FASTROM factory coded parts contain the code supplied by the customer. This implies that Flash devices have to be configured by the customer using the Option Bytes while the ROM devices are factory-configured.

14.1 FLASH OPTION BYTES

	STATIC OPTION BYTE 0								STATIC OPTION BYTE 1							
	7							0	7							0
	WDG		CKSEL	VD		$\overline{\text{RSTC}}$	DIV2	FMP_R	PKG						MCO	
	HALT	SW		1	0				2	1	0					
Default value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

The option bytes allow the hardware configuration of the microcontroller to be selected. They have no address in the memory map and can be accessed only in programming mode (for example using a standard ST7 programming tool). The default content of the FLASH is fixed to FFh. This means that all the options have “1” as their default value.

OPTION BYTE 0

OPT7= **WDG HALT** *Watchdog and Halt mode*
This option bit determines if a RESET is generated when entering Halt mode while the Watchdog is active.
0: No Reset generation when entering Halt mode
1: Reset generation when entering Halt mode

OPT6= **WDG SW** *Hardware or software watchdog*
This option bit selects the watchdog type.
0: Hardware (watchdog always enabled)
1: Software (watchdog to be enabled by software)

OPT5 = **CKSEL** *Clock Source Selection.*
0: PLL clock selected[1)]
1: Oscillator clock selected

Note 1: Even if PLL clock is selected, a clock signal must always be present (refer to Figure 13. on page 28)

OPT4:3= **VD[1:0]** *Voltage detection*
These option bits enable the voltage detection block (LVD, and AVD).

Selected Low Voltage Detector	VD1	VD0
LVD and AVD On	0	0
LVD On and AVD Off	0	1
LVD and AVD Off	1	0
	1	1

OPT2 = **RSTC** *RESET clock cycle selection*
This option bit selects the number of CPU cycles applied during the RESET phase and when exiting Halt mode. For resonator oscillators, it is advised to select 4096 due to the long crystal stabilization time.
0: Reset phase with 4096 CPU cycles
1: Reset phase with 256 CPU cycles
Note: When the PLL clock is selected (CKSEL=0), the reset clock cycle selection is forced to 4096 CPU cycles.

OPT1= **DIV2** *Divider by 2*
1: DIV2 divider disabled with OSCIN = 8MHz
0: DIV2 divider enabled (in order to have 8 MHz required for the PLL with OSCIN =16 Mhz))

OPT0= **FMP_R** *Flash memory read-out protection*
Readout protection, when selected provides a protection against program memory content extraction and against write access to Flash memory. This protection is based on a read/write protection of the memory in test modes and ICP mode. Erasing the option bytes when the FMP_R option is selected causes the whole user memory to be erased first and the device can be reprogrammed. Refer to the ST7 Flash Programming Reference Manual and section 4.3.1 on page 22 for more details.
0: Read-out protection enabled
1: Read-out protection disabled

ST7MC DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

OPTION BYTE 1

OPT7:5= **PKG[2:0]** *package selection*
These option bits are used to select the device package.

Selected Package	PKG2	PKG1	PKG0
LQFP32 / SDIP32	0	0	0
LQFP44	0	0	1
SDIP 56	0	1	0
LQFP64	0	1	1
LQFP80	1	x	x

OPT4:2= Reserved

OPT1:0 = **MCO** *Motor Control Output Options*
MCO port under reset.

Motor Control Output	bit 1	bit 0
HiZ	0	0
Low	0	1
High	1	0
HiZ	1	1

14.2 DEVICE ORDERING INFORMATION AND TRANSFER OF CUSTOMER CODE

The FASTROM or ROM contents are to be sent on diskette, or by electronic means, with the hexadecimal file in .S19 format generated by the development tool. All unused bytes must be set to FFh.

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

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

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

Figure 167. Ordering information scheme

Example: ST7 F MC1 K 2 T 6

Family

ST7 microcontroller family

Memory type

F: Flash
P: FASTROM

Sub-family

MC1 or MC2

No. of pins

K = 32
S = 44
N = 56
R = 64
M = 80

Memory size

2 = 8K
4 = 16K
6 = 32K
7 = 48K
9 = 60K

Package

T = LQFP
B = Plastic DIP

Temperature range

6 = -40 °C to 85 °C
3 = -40 °C to 125 °C

For a list of available options (e.g. memory size, package) and orderable part numbers or for further information on any aspect of this device, please contact the ST Sales Office nearest to you.

ST7MC DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

ST7MC MICROCONTROLLER OPTION LIST
(Last update: September 2008)

Customer: . . . . . .
Address: . . . . . .
. . . . . .
Contact: . . . . . .
Phone No: . . . . . .
Reference/ROM or FASTROM Code* : . . . . . .
*The ROM or FASTROM code name is assigned by STMicroelectronics.
ROM or FASTROM code must be sent in .S19 format. .Hex extension cannot be processed.

Device type/memory size/package (check only one option):

ROM	8K	16K	32K	48K	60K
LQFP32:	[ ]				
LQFP44:		[ ]			

FASTROM	8K	16K	32K	48K	60K
LQFP32:	[ ]	[ ]			
LQFP44:		[ ]	[ ]		
LQFP64:			[ ]	[ ]	
LQFP80:					[ ]

Conditioning (check only one option):

[ ] Tape & Reel [ ] Tray

Special Marking [ ] No [ ] Yes "_ _ _ _ _ _ _ _ _ _ " (10 char. max)

Authorized characters are letters, digits, '.', '-', '/' and spaces only.

Temperature range [ ] - 40°C to + 85°C [ ] - 40°C to + 125°C

MCO (Motor Control Output state under reset)	[ ] Hiz	[ ] Low	[ ] High
DIV2	[ ] Disabled		[ ] Enabled
CKSEL	[ ] Oscillator clock		[ ] PLL clock
Watchdog Selection	[ ] Software Activation		[ ] Hardware Activation
Halt when Watchdog on	[ ] Reset		[ ] No reset
Readout Protection	[ ] Disabled		[ ] Enabled
LVD Reset	[ ] Disabled		[ ] Enabled
AVD Interrupt (if LVD enabled)	[ ] Disabled		[ ] Enabled
Reset Delay	[ ] 256 Cycles		[ ] 4096 Cycles

Supply Operating Range in the application: . . . . . .

Notes . . . . . .

Date . . . . . .

Signature . . . . . .

Please download the latest version of this option list from:
http://www.st.com/mcu > downloads > ST7 microcontrollers > Option list

ST7MC DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

14.3 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.

14.3.1 Starter kits

ST offers complete, affordable **starter kits** and full-featured that allow you to evaluate microcontroller features and quickly start developing ST7 applications. Starter kits are complete hardware/software tool packages that include features and samples to to help you quickly start developing your application.

14.3.2 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 16K of code.

The range of hardware tools includes full-featured **ST7-EMU2B series emulators** and the low-cost **RLink** in-circuit debugger/programmer. These tools are 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.

14.3.3 Programming tools

During the development cycle, the **ST7-EMU3 series emulators** and the **RLink** provide in-circuit programming capability for programming the Flash microcontroller on your application board.

ST also provides a low-cost dedicated in-circuit programmer, the **ST7-STICK**, as well as **ST7 Socket Boards** which provide all the sockets required for programming any of the devices in a specific ST7 sub-family on a platform that can be used with any tool with in-circuit programming capability for ST7.

For production programming of ST7 devices, ST's third-party tool partners also provide a complete range of gang and automated programming solutions, which are ready to integrate into your production environment.

14.3.4 Order codes for ST7MC development tools

Table 91. Development tool order codes for the ST7MC family

MCU	Starter kit	Emulator	Programming tool
ST7MC1 ST7MC2	ST7MC-KIT/BLDC	ST7MDT50-EMU3	ST7-STICK[1)2)] STX-RLINK[3)]

1. Add suffix /EU, /UK or /US for the power supply for your region

2. Parallel port connection to PC

3. RLink with ST7 tool set

For additional ordering codes for spare parts and accessories, refer to the online product selector at www.st.com/mcu.

ST7MC DEVICE CONFIGURATION AND ORDERING INFORMATION (Cont'd)

14.3.5 PACKAGE/SOCKET FOOTPRINT PROPOSAL

Table 92. Suggested List of Socket Types

Package / Probe	Socket Reference		Emulator Adapter	
LQFP64 14x14	CAB	3303262	CAB	3303351
LQFP80 14x14	YAMAICHI	IC149-080-*51-*5	YAMAICHI	ICP-080-7
LQFP32 7x7	IRONWOOD	SF-QFE32SA-L-01	IRONWOOD	SK-UGA06/32A-01
LQFP44 10x10	YAMAICHI	IC149-044-*52-*5	YAMAICHI	ICP-044-5
SDIP32	Standard		Standard	
SDIP56	Standard		Standard	

14.4 ST7 APPLICATION NOTES

Table 93. 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 93. 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 VS 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
AN2200	GUIDELINES FOR MIGRATING ST7LITE1X APPLICATIONS TO ST7FLITE1XB
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
AN1953	PFC FOR ST7MC STARTER KIT
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
AN1039	ST7 MATH UTILITY ROUTINES

Table 93. 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 AN ST72324 TARGET APPLICATION
AN1477	EMULATED DATA EEPROM WITH XFLASH MEMORY
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

15 IMPORTANT NOTES

15.1 FLASH/FASTROM DEVICES ONLY

The behaviors described in the following sections are present on Rev A, B or C ST7FMC and ST7PMC devices only.

They are identifiable:

- on the device package, by the last letter of the Trace Code marked on the device package.

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

See also Figure 170. on page 306

The following table gives the limitations and the impacted devices.

Table 94. Device Identification

Device Type	Trace Code marked on device / Internal Sales Type on box label	Limitations					
		Section 15.2 Clearing active interrupts	Section 15.4 LINSCI Limiations	Section 15.5 Missing detection of "Z event"	Section 15.6 Injected Curent on PD7	Section 15.7 Reset value of unavailable pins	Section 15.8 Max values of AVD thresholds
Flash Devices	"xxxxxxxxxB"/ ST7FMCxxxxx$XY	X	X	X	X	❍	❍
FASTROM Devices	"xxxxxxxxxB" / ST7PMCxxxxx$XY	X	X	X	X	❍	❍
Flash Devices	"xxxxxxxxxC"/ ST7FMCxxxxx$XY	X	X	X	❍	❍	❍
FASTROM Devices	"xxxxxxxxxC" / ST7PMCxxxxx$XY	X	X	X	❍	❍	❍
Flash Devices	"xxxxxxxxxA"/ ST7FMCxxxxx$XY	❍	❍	X	❍	X	X
FASTROM Devices	"xxxxxxxxxA" / ST7PMCxxxxx$XY	❍	❍	X	❍	X	X

Legend: Limitation present = X ; Limitation not present = ❍.

IMPORTANT NOTES (Cont'd)

15.2 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
```

Nested 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 with higher or identical priority level

- 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:

```
PUSH CC
SIM
reset flag or interrupt mask
POP CC
```

15.3 TIMD SET SIMULTANEOUSLY WITH OC INTERRUPT

If the 16-bit timer is disabled at the same time as the output compare event occurs then the output compare flag gets locked and cannot be cleared before the timer is enabled again.

15.3.1 Impact on the application

If the output compare interrupt is enabled, then the output compare flag cannot be cleared in the timer interrupt routine. Consequently the interrupt service routine is called repeatedly and the application gets stuck which causes the watchdog reset if enabled by the application.

15.3.2 Workaround

Disable the timer interrupt before disabling the timer. While enabling, first enable the timer, then enable the timer interrupts.

Perform the following to disable the timer

– TACR1 or TBCR1 = 0x00h; // Disable the compare interrupt.

– TACSR | or TBCSR |= 0x40; // Disable the timer.

– Perform the following to enable the timer again

– TACSR & or TBCSR & = ~0x40; // Enable the timer.

– TACR1 or TBCR1 = 0x40; // Enable the compare interrupt.

15.4 LINSCI LIMITATIONS

15.4.1 LINSCI wrong break duration

SCI Mode

A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 20 bits instead of 10 bits if M=0

- 22 bits instead of 11 bits if M=1.

In the same way, as long as the SBK bit is set, break characters are sent to the TDO pin. This may lead to generate one break more than expected.

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (fCPU=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

IMPORTANT NOTES (Cont'd)

Workaround

If this wrong duration is not compliant with the communication protocol in the application, software can request that an Idle line be generated before the break character. In this case, the break duration is always correct assuming the application is not doing anything between the idle and the

break. This can be ensured by temporarily disabling interrupts.

The exact sequence is:

- Disable interrupts
- Reset and Set TE (IDLE request)
- Set and Reset SBK (Break Request)
- Re-enable interrupts

LIN mode

If the LINE bit in the SCICR3 is set and the M bit in the SCICR1 register is reset, the LINSCI is in LIN master mode. A single break character is sent by setting and resetting the SBK bit in the SCICR2 register. In some cases, the break character may have a longer duration than expected:

- 24 bits instead of 13 bits

Occurrence

The occurrence of the problem is random and proportional to the baudrate. With a transmit frequency of 19200 baud (fCPU=8MHz and SCIBRR=0xC9), the wrong break duration occurrence is around 1%.

Analysis

The LIN protocol specifies a minimum of 13 bits for the break duration, but there is no maximum value. Nevertheless, the maximum length of the header is specified as (14+10+10+1)x1.4=49 bits. This is composed of:

- the synch break field (14 bits),
- the synch field (10 bits),
- the identifier field (10 bits).

Every LIN frame starts with a break character. Adding an idle character increases the length of each header by 10 bits. When the problem occurs, the header length is increased by 11 bits and becomes ((14+11)+10+10+1)=45 bits.

To conclude, the problem is not always critical for LIN communication if the software keeps the time between the sync field and the ID smaller than 4 bits, i.e. 208us at 19200 baud.

The workaround is the same as for SCI mode but considering the low probability of occurrence (1%), it may be baetter to keep the break generation sequence as it is.

15.4.2 Header Time-out does not prevent wake-up from mute Mode

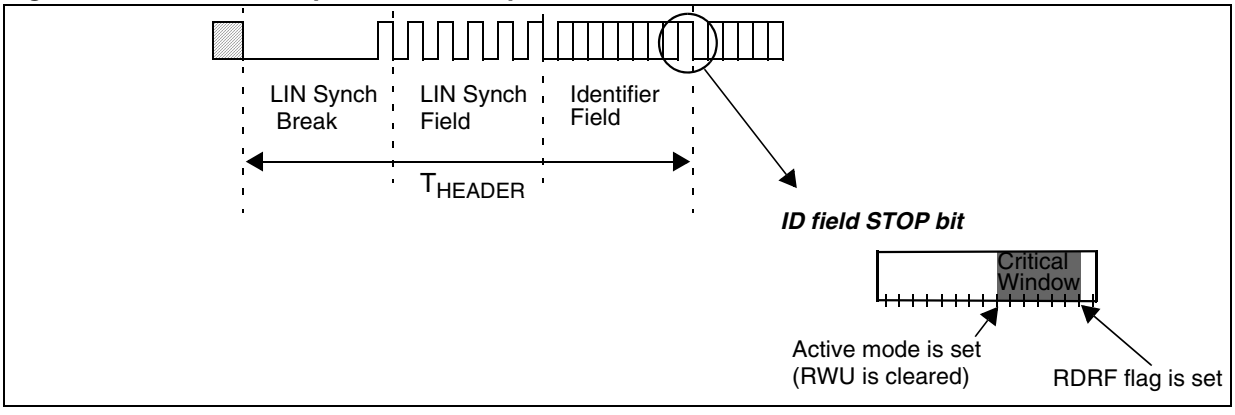
Normally, when LINSCI is configured in LIN slave mode, if a header time-out occurs during a LIN header reception (i.e. header length > 57 bits), the LIN Header Error bit (LHE) is set, an interrupt occurs to inform the application but the LINSCI should stay in mute mode, waiting for the next header reception.

Problem Description

The LINSCI sampling period is Tbit / 16. If a LIN Header time-out occurs between the 9th and the 15th sample of the Identifier Field Stop Bit (refer to Figure 168), the LINSCI wakes up from mute mode. Nevertheless, LHE is set and LIN Header Detection Flag (LHDF) is kept cleared.

In addition, if LHE is reset by software before this 15th sample (by accessing the SCISR register and reading the SCIDR register in the LINSCI interrupt routine), the LINSCI will generate another LINSCI interrupt (due to the RDRF flag setting).

Figure 168. Header Reception Event Sequence



IMPORTANT NOTES (Cont'd)

Impact on application

Software may execute the interrupt routine twice after header reception.

Moreover, in reception mode, as the receiver is no longer in mute mode, an interrupt will be generated on each data byte reception.

Workaround

The problem can be detected in the LINSCI interrupt routine. In case of timeout error (LHE is set and LHLR is loaded with 00h), the software can check the RWU bit in the SCICR2 register. If RWU is cleared, it can be set by software. Refer to Figure 169. Workaround is shown in bold characters.

Figure 169. LINSCI Interrupt routine

```
@interrupt void LINSCI_IT ( void ) /* LINSCI interrupt routine */
{
      /* clear flags */
      SCISR_buffer = SCISR;
      SCIDR_buffer = SCIDR;

      if ( SCISR_buffer & LHE )/* header error ? */
      {
            if (!LHLR)/* header time-out? */
            {
                  if ( !(SCICR2 & RWU) )/* active mode ? */
                  {
                        _asm("sim");/* disable interrupts */
                        SCISR;
                        SCIDR;/* Clear RDRF flag */
                        SCICR2 |= RWU;/* set mute mode */
                        SCISR;
                        SCIDR;/* Clear RDRF flag */
                        SCICR2 |= RWU;/* set mute mode */
                        _asm("rim");/* enable interrupts */
                  }
            }
      }
}
```

Example using Cosmic compiler syntax

15.5 MISSING DETECTION OF BLDC "Z EVENT"

For a BLDC drive, the Dead Time generator is enabled through the MDTG register (PCN=0 and DTE=1). If the duty cycle of the PWM signal generated to drive the motor is lower than the programmed deadtime, the Z event sampling will be missing.

Workaround

The complementary PWM must be disabled by resetting the DTE bit in the MDTG register (see page 221).

As the current in the motor is very low in this case, the MOSFET body diode can be used.

15.6 INJECTED CURRENT ON PD7

On rev.B silicon, the parameter Iinj(pin), injected current on I/O pins (see section 12.8.1 on page 264), is limited at 0 (instead of -2mA) for the pin PD7. This limitation is no longer present on rev.C and A silicon and all I/O pins have the max values +5/-2 mA.

15.7 RESET VALUE OF UNAVAILABLE PINS

On A silicon versions, some ports (Ports A, C and E) have less than 8 pins. The bits associated to the unavailable pins must always be kept at reset state.

15.8 MAXIMUM VALUES OF AVD THRESHOLDS

On rev. A silicon versions, the max. values of AVD thresholds are not tested in production.

15.9 EXTERNAL INTERRUPT MISSED

To avoid any risk if generating a parasitic interrupt, the edge detector is automatically disabled for one clock cycle during an access to either DDR and OR. Any input signal edge during this period will not be detected and will not generate an interrupt.

This case can typically occur if the application refreshes the port configuration registers at intervals during runtime.

Workaround

The workaround is based on software checking the level on the interrupt pin before and after writing to the PxOR or PxDDR registers. If there is a level change (depending on the sensitivity programmed for this pin) the interrupt routine is invoked using the call instruction with three extra PUSH instructions before executing the interrupt routine (this is to make the call compatible with the IRET instruction at the end of the interrupt service routine).

But detection of the level change does not make sure that edge occurs during the critical 1 cycle duration and the interrupt has been missed. This may lead to occurrence of same interrupt twice (one hardware and another with software call).

To avoid this, a semaphore is set to '1' before checking the level change. The semaphore is changed to level '0' inside the interrupt routine. When a level change is detected, the semaphore status is checked and if it is '1' this means that the last interrupt has been missed. In this case, the interrupt routine is invoked with the call instruction.

There is another possible case i.e. if writing to PxOR or PxDDR is done with global interrupts disabled (interrupt mask bit set). In this case, the semaphore is changed to '1' when the level change is detected. Detecting a missed interrupt is done after the global interrupts are enabled (interrupt mask bit reset) and by checking the status of the semaphore. If it is '1' this means that the last interrupt was missed and the interrupt routine is invoked with the call instruction.

To implement the workaround, the following software sequence is to be followed for writing into the PxOR/PxDDR registers. The example is for for Port PF1 with falling edge interrupt sensitivity. The software sequence is given for both cases (global interrupt disabled/enabled).

Case 1: Writing to PxOR or PxDDR with Global Interrupts Enabled:

```
LD A,#01
LD sema,A      ; set the semaphore to '1'
LD A,PFDR
AND A,#02
LD X,A         ; store the level before writing to PxOR/PxDDR
LD A,#$90
LD PFDDR,A     ; Write to PFDDR
LD A,#$ff
LD  PFOR,A     ; Write to PFOR
LD A,PFDR
AND A,#02
LD Y,A         ; store the level after writing to PxOR/PxDDR
LD A,X         ; check for falling edge
cp A,#02
jrne OUT
TNZ Y
jrne OUT
LD A,sema      ; check the semaphore status if edge is detected
CP A,#01
jrne OUT
call call_routine; call the interrupt routine
OUT:LD A,#00
LD sema,A
.call_routine  ; entry to call_routine
PUSH A
PUSH X
PUSH CC
.ext1_rt       ; entry to interrupt routine
LD A,#00
LD sema,A
IRET
```

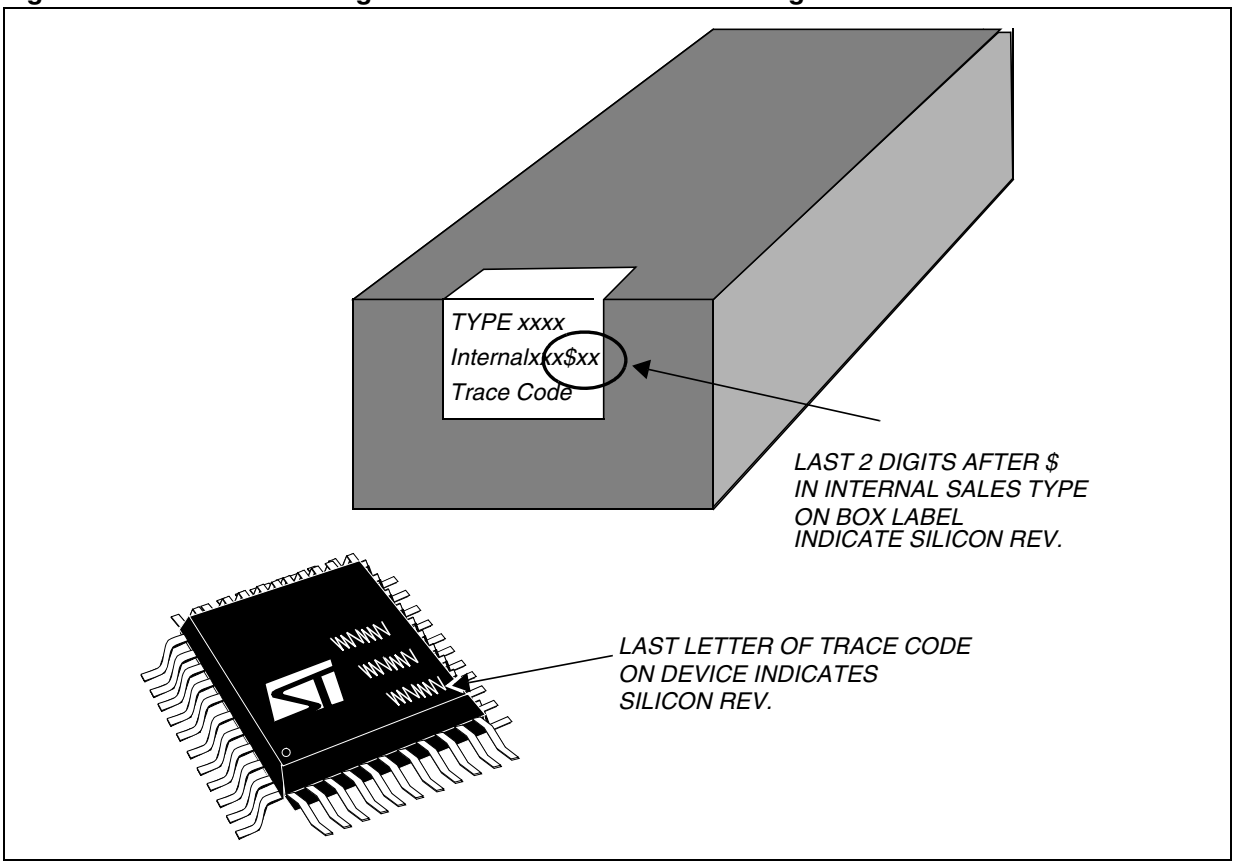
Case 2: Writing to PxOR or PxDDR with Global Interrupts Disabled:

```
SIM            ; set the interrupt mask
LD A,PFDR
AND A,#$02
LD X,A         ; store the level before writing to PxOR/PxDDR
LD A,#$90
```

```
LD PFDDR,A; Write into PFDDR
LD A,#$ff
LD PFOR,A          ; Write to PFOR
LD A,PFDR
AND A,#$02
LD Y,A        ; store the level after writing to PxOR/
PxDDR
LD A,X        ; check for falling edge
cp A,#$02
jrne OUT
TNZ Y
jrne OUT
LD A,#$01
LD sema,A  ; set the semaphore to '1' if edge is
detected
RIM             ; reset the interrupt mask
LD A,sema  ; check the semaphore status
CP A,#$01
jrne OUT
call call_routine; call the interrupt routine
RIM
OUT:          RIM
JP while_loop
.call_routine ; entry to call_routine
PUSH A
PUSH X
PUSH CC
.ext1_rt       ; entry to interrupt routine
LD A,#$00
LD sema,A
IRET
```

IMPORTANT NOTES (Cont'd)

Figure 170. Revision Marking on Box Label and Device Marking


TYPE xxxx
Internalxxx

16 REVISION HISTORY

Table 95. Revision History

Date	Revision	Description of Changes
April-2005	6	Added one sales type (ST7FMC1K4T6) Changed port description for $\overline{MCES}$ in Table 1, “ST7MC Device Pin Description,” on page 12 and added note 3 Changed register addresses in Table 15, “PWM Auto-Reload Timer Register Map and Reset Values,” on page 74 Changed title of Figure 123 on page 227 and Figure 124 Changed Table 36 on page 166 Changed MTIM timer value to 100h in section “b” on page 171 Changed text above Figure 98. on page 177 (15kHz instead of 20kHz and replaced f_{CPU} = 4MHz by f_{MTC} = 4MHz) and changed Figure 98 (15kHz instead of 20kHz) Changed Figure 107. on page 188 Changed Figure 108. on page 189 Changed Figure 115 on page 197 Changed description of OO[5:0] bits in the MPHST register (section 10.6.13 on page 206) Changed min and max values for R_{ON} in section 12.9.1 on page 266 Changed Figure 144 and Figure 145. on page 267 and notes
July-2005	7	Added ST7FMC1K2T3 and ST7FMC1K6TC sales types (updated device summary on 1st page and “Supported part numbers” on page 292) Changed V_{IT-} and V_{IT+} in section 12.3.2 on page 250 Removed note 1 in section 12.4.4 on page 253 Changed $I_{INJ(PIN)}$ in section 12.8.1 on page 263 and removed note 1 for V_{IL} and V_{IH} Changed R_{ON} max in section 12.9.1 on page 266 and removed note 1 for V_{IL}, V_{IH} and R_{ON} Removed subsection in Important Notes Section 14 : “Op-Amp input offset voltage” Notes for Table in Section 12.13 updated, removing link to ”Op-Amp input offset voltage”
Nov-2005	8	Flash memory data retention changed (first page and Section 12.6.2) Removed references to true open drain I/Os.Updated Table 1 on page 12: replaced MCZEM5) by MCZEM[6)], and replaced Port H0, Port H1, Port H2 and Port H3 respectively by Port H4, Port H5, Port H6 and Port H7 (in the rows corresponding to PH4, PH5, PH6 and PH7 in the “Pin name column” Updated note 8 in Table 1 on page 12 Added section 5 on page 24 Added note to section 6.1 on page 28 Added note to section 6.2.1 on page 29 and updated Figure 15. on page 29 Changed section 10.5.5.2 on page 110 (when a character transmission is complete, etc.) Modified Table 34 on page 161 (window and event filters column) Replaced MISR register by MIMR register in the 3rd paragraph of “Switched Mode” on page 168 Removed caution and added notes 3 and 4 below Figure 93. on page 170 in section 10.6.7.2 on page 170 In section 10.6.6.11 on page 163 and in Figure 94 on page 172, replace “TES[1:0] = 00,01 or 10” by “TES[1:0] = 00,10 or 11” Changed $V_{IT+(AVD)}$ min value in section 12.3.3 on page 250 Changed section 12.5.3 on page 256 Changed I_s parameter description and note 4 in section 12.8.1 on page 263 Updated note 3 in section 12.8.1 on page 263 Added note to Figure 138. on page 264 Removed 32K ROM versions: changed “Supported part numbers” on page 290 and “ST7MC MICROCONTROLLER OPTION LIST” on page 291 Changed “FLASH/FASTROM DEVICES ONLY” on page 297 (added table) and added “RESET VALUE OF UNAVAILABLE PINS” on page 300 Modified section 15.5 on page 300 Added Important Notes subsection section 15.5 on page 300: “Injected Current On PD7”

Table 95. Revision History

Date	Revision	Description of Changes
Mar-06	9	Added two sales types: ST7FMC2S4T3 and ST7FMC2S6T3 Changed QFP package name: TQFP replaced by LQFP Changed "Master Mode Operation" on page 98: added important note Changed "Interrupt Mapping" on page 43: modified interrupt n°13 Added note to V_{DD} in Figure 13. on page 27 Modified Figure 24. on page 45. Changed section 7.7 on page 46 (IPB, IS3[1:0] and IPA descriptions) Changed Table 36 on page 166 and added notes Changed SWA bit description after Table 58 on page 210 Added note to section 12.3.3 on page 250 Changed R_{ON} max values in section 12.9.1 on page 266 Added Figure 154. on page 276 Changed notes to "THERMAL CHARACTERISTICS" on page 288 Added text on ECOPACK packages in Section 13 PACKAGE CHARACTERISTICS and changed Section 13.3 SOLDERING INFORMATION Added "EXTERNAL INTERRUPT MISSED" on page 303 Added Section 15.7 MAXIMUM VALUES OF AVD THRESHOLDS
19-June-06	10	Modified MCES description in Section 7.2 on page 41 Modified name of bit 5 in the SPICSR register in Table 19 on page 106 Modifed section 10.4.3.3 on page 99 (added title "how to operate the SPI in Master mode) Modified table in section 12.7.1 on page 261 Modified section 12.11.1 on page 271 ($t_{su(\overline{SS})}$, $t_{v(MO)}$ and $t_{h(MO)}$) and added note1 to $t_{su(\overline{SS})}$ and $t_{h(\overline{SS})}$ Removed EMC protection circuitry in Figure 145 on page 268 (device works correctly without these components) Modified description of DIV2 bit in section 14.1 on page 290 Changed Table 91 on page 293: removed ST7MC1K2B6 and ST7PMC1K2B6 (SDIP32 package) Modified "ST7MC MICROCONTROLLER OPTION LIST" on page 293 Added revision history for revisions 6 and 7
08-Dec-06	11	Added caution to section 6.2.1 on page 30 Modified Figure 149 on page 272 ($t_{v(MO)}$, $t_{h(MO)}$) Replaced CPHA=0 with CPHA=1 in Figure 148. on page 272 Modified section 14.3 on page 294 Added one sales type (ST7FMC2S7T6) and modified Table 91 on page 293
25-Sep-08	12	Title of the document changed. Modified Table 1, "Device summary," on page 1 Removed SDIP32 package and part numbers for automotive products Removed reference to ST7MC1K6 Added footnote to Table 1, "ST7MC Device Pin Description," on page 12 indicating that it is mandatory to connect all available V_{DD} and V_{DDA} pins to the supply voltage and all V_{SS} and V_{SSA} pins to ground. Values in inches rounded to 4 decimal digits (instead of 3 decimal digits) in Section 13.1 PACKAGE MECHANICAL DATA. "Output Compare" on page 82: Changed text of note 3 and removed compare register i latch signal from Figure 52. "Output Compare Timing Diagram, fTIMER =fCPU/4" page 84. Modified t_{RET} values in Section 12.6.2 FLASH Memory Modified Section 12.7.3 Absolute Maximum Ratings (Electrical Sensitivity). Added "TIMD SET SIMULTANEOUSLY WITH OC INTERRUPT" on page 300. Modified section 14.2 on page 292. (Figure 167 on page 292 and "ST7MC MICROCONTROLLER OPTION LIST" on page 293)
02-Apr-09	13	$V_{ESD(MM)}$ removed in section 12.2.1 on page 248 and section 12.7.3 on page 263 Modified GPB values (and conditions) in section 12.13 on page 280 Modified ECOPACK text in section 13 on page 285

Notes:

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2009 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

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

www.st.com