ST7MC1K2-Auto, ST7MC1K6-Auto ST7MC2S4-Auto, ST7MC2S6-Auto

8-bit MCU for automotive with nested interrupts, Flash, 10-bit ADC, brushless motor control, 5 timers, SPI, LINSCI™

Features

- Memories
  - 8 to 32 Kbyte dual voltage Flash program memory or ROM with read-out protection capability, in-application programming and in-circuit programming
  - 384 to 1 Kbyte 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.
  - 4 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 34 I/O ports
  - Up to 34 multifunctional bidirectional I/O lines
  - Up to 10 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

LQFP32 7 x 7

LQFP44 10 x 10

- 2 communication interfaces
  - SPI synchronous serial interface
  - LINSCI™ asynchronous serial interface
- Brushless motor control peripheral
  - 6 high sink PWM output channels for sinewave 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 up to 11 input pins
- In-circuit debug
- Instruction set
  - 8-bit data manipulation
  - 63 basic instructions with illegal opcode detection
  - 17 main addressing modes
  - 8x8 unsigned multiply instruction
  - True bit manipulation
- Development tools
  - Full hardware/software development package

Contents

List of tables

List of figures

1 Description

The ST7MC1K2-Auto, ST7MC1K6-Auto, ST7MC2S4-Auto, and ST7MC2S6-Auto devices are members of the ST7 microcontroller family designed for mid-range automotive 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.

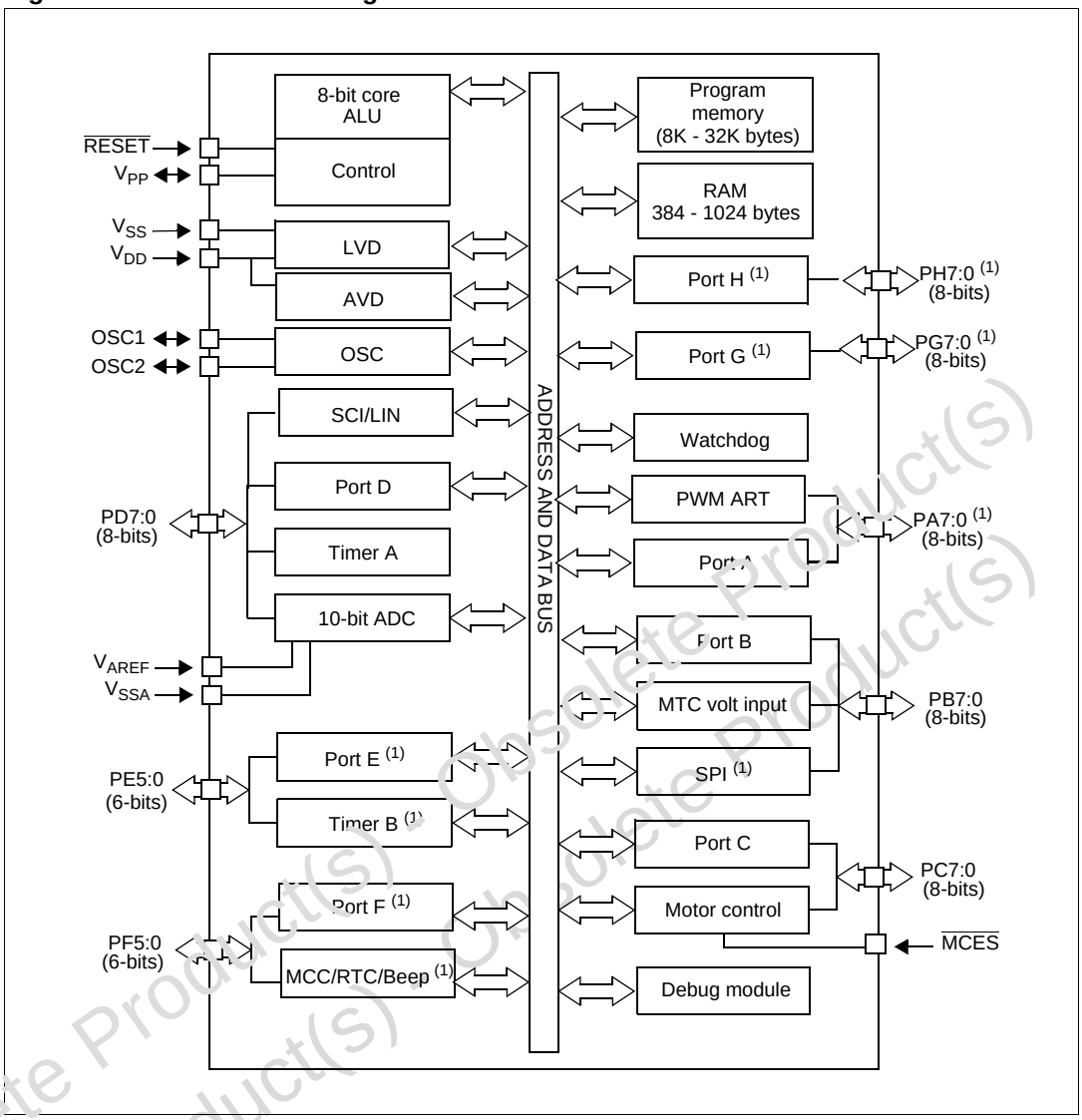
The enhanced instruction set and addressing modes of the ST7 offer both power and flexibility to software developers, enabling the design of highly efficient and compact application code. In addition to standard 8-bit data management, all ST7 microcontrollers feature true bit manipulation, 8x8 unsigned 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*.

Table 1. Device summary

Device	Program memory - bytes	RAM (stack) - bytes	Operating supply vs. frequency	Temp. range	Package	Peripherals	
ST7MC1K2-Auto	Flash/ROM 8 K	384 (256)	4.5 to 5.5 V with $f_{CPU} \leq$ 8 MHz	-40°C to 85°C/ -40 to 125°C	LQFP32	Watchdog, 16-bit timer A, LINSCI™, 10-bit ADC, MTC, 8-bit PWM ART, ICD	-
ST7MC1K6-Auto	Flash 32 K	1024 (256)		-40°C to 125°C			
ST7MC2S4-Auto	Flash/ROM 16 K	768 (256)		-40°C to 85°C/ -40°C to 125°C	LQFP44		SPI, 16-bit timer B
ST7MC2S6-Auto	Flash 32 K	1024 (256)					

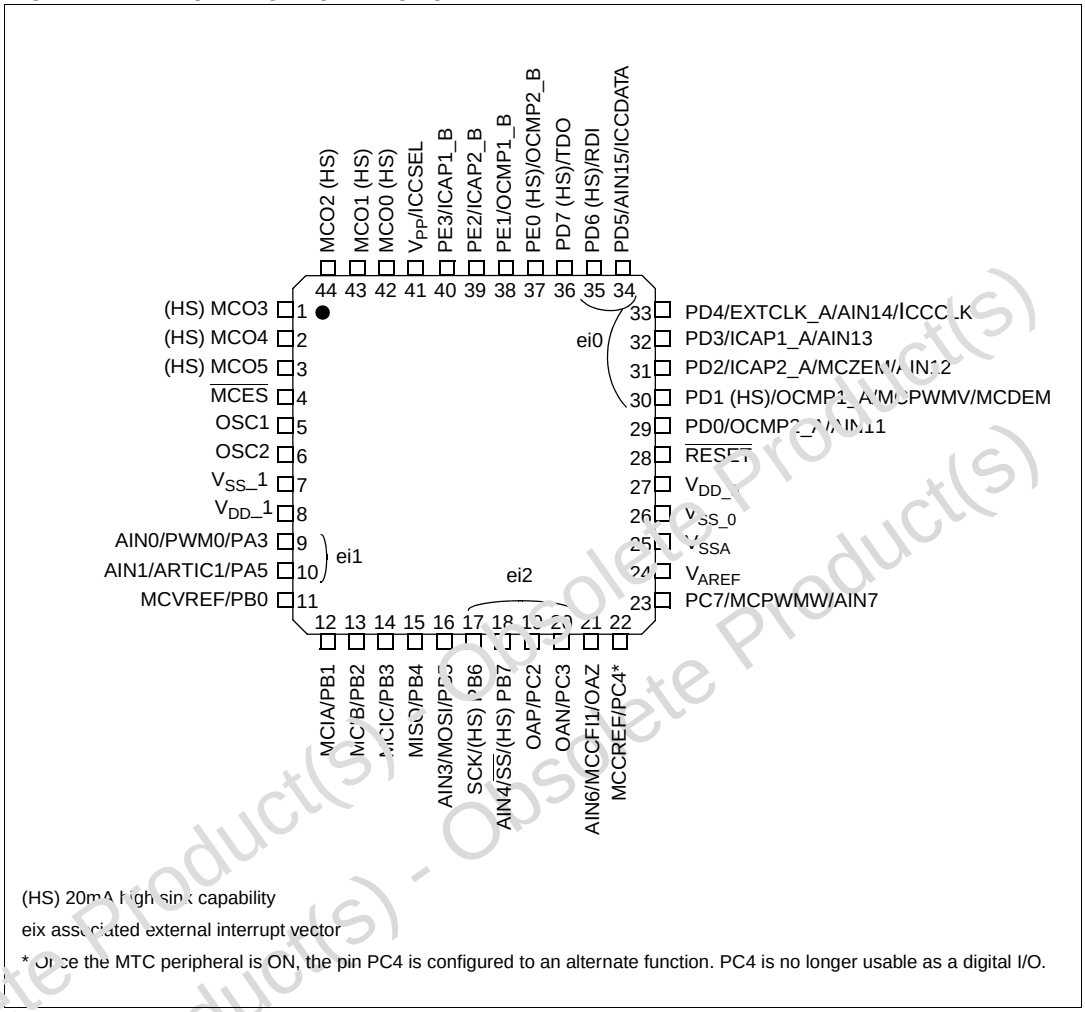
Figure 1. Device block diagram



1. On some devices only; see *Table 2: Device pin description on page 23*

2 Pin description

Figure 2. 44-pin LQFP package pinouts

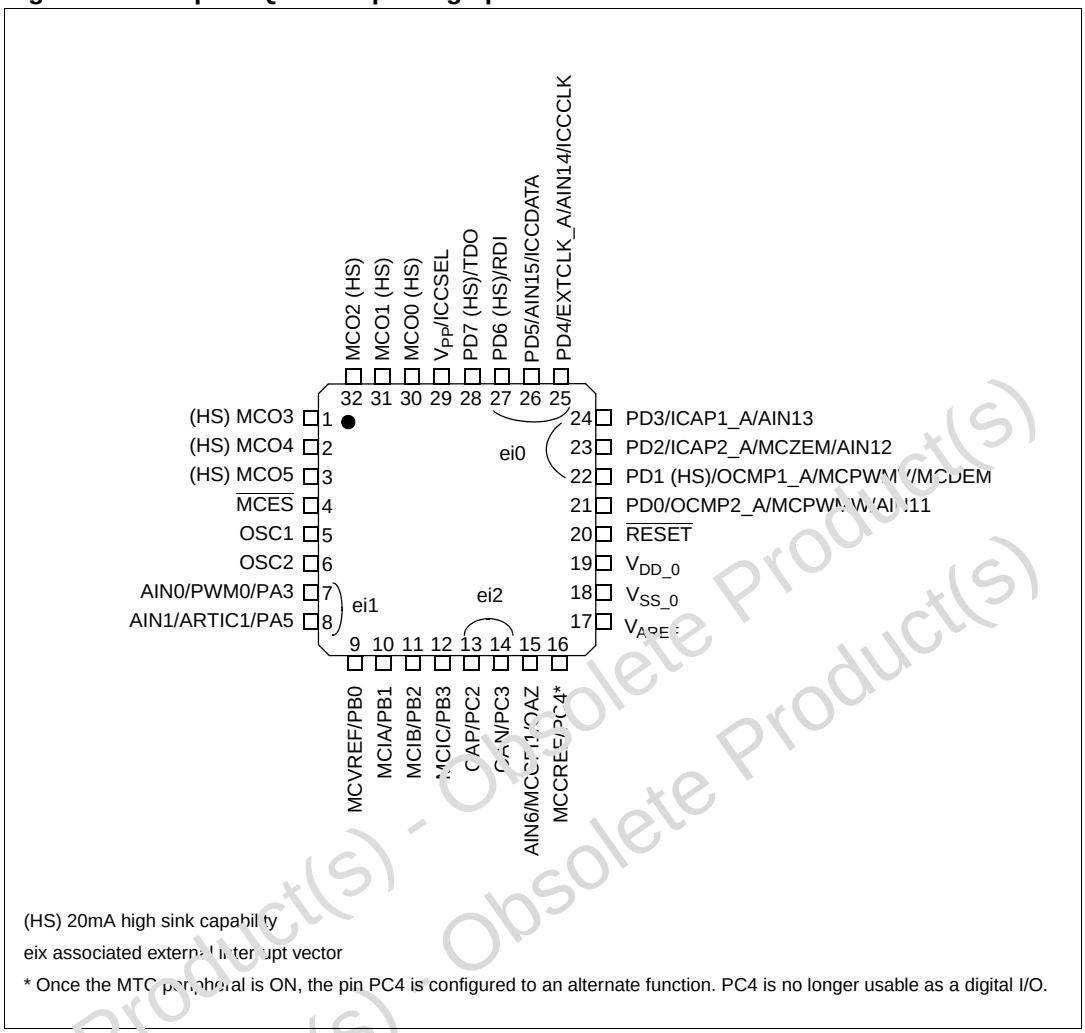


(HS) 20mA high sink capability

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.

Figure 3. 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.

For external pin connection guidelines, see *Section 12: Electrical characteristics on page 312*.

Legend/abbreviations for *Table 2*:

Type

I = input

O = output

S = supply

Input level

C_T = CMOS $0.3V_{DD}/0.7V_{DD}$ with Schmitt trigger

T_T = refer to the G and H ports characteristics in *Section 12.8.1 on page 329*

Output level

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

Port and control configuration

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

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

Refer to *Section 5: Central processing unit on page 37* 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 the reset state.

Table 2. Device pin description(1)

Pin number		Pin name	Type	Level		Port						Main function (after reset)	Alternate function(2)
						Input				Output			
LQFP44	LQFP32			Input	Output	float	wpu	int(3)	ana	OD	PP		
1	1	MCO3 (HS)	O		HS						X	Motor control output 3	
2	2	MCO4 (HS)	O		HS						X	Motor control output 4	
3	3	MCO5 (HS)	O		HS						X	Motor control output 5	
4	4	MCES(4)	I	C_T		**X**						MTC emergency stop	
(5)	(5)	PG0	I/O	T_T		**X**	X			X	X	Port G0	
		PG1	I/O	T_T		**X**	X			X	X	Port G1	
		PG2	I/O	T_T		**X**	X			X	X	Port G2	
		PG3	I/O	T_T		**X**	X			X	X	Port G3	
5	5	OSC1(6)	I									External clock input or resonator oscillator inverter input	
6	6	OSC2(6)	I/O									Resonator oscillator inverter output	
7	(5)	V_{SS_1}(7)	S									Digital ground voltage	
8	(5)	V_{DD_1}(7)	S									Digital main supply voltage	
(5)	(5)	PA0/PWM3	I/O	C_T		**X**	X			X	X	Port A0	PWM output 3
		PA1/PWM2	I/O	C_T	HS	**X**	X			X	X	Port A1	PWM output 2
		PA2PWM1	I/O	C_T		**X**	X			X	X	Port A2	PWM output 1

Table 2. Device pin description[1] (continued)

Pin number		Pin name	Type	Level		Port						Main function (after reset)	Alternate function[2]	
						Input				Output				
LQFP44	LQFP32			Input	Output	float	wpu	int[3]	ana	OD	PP			
9	7	PA3/PWM0/AIN0	I/O	C_T		X	ei1		X	X	X	Port A3	PWM output 0	ADC analog input 0
(5)	(5)	PA4 (HS)/ARTCLK	I/O	C_T	HS	X	X			X	X	Port A4	PWM-ART external clock	
10	8	PA5/ARTIC1/AIN1	I/O	C_T		X		ei1	X	X	X	Port A5	PWM-ART input capture 1	ADC analog input 1
(5)	(5)	PA6/ARTIC2	I/O	C_T		X	ei1			X	X	Port A6	PWM-ART input capture 2	
		PA7/AIN2	I/O	C_T		X		ei1	X	X	X	Port A7	ADC analog input 2	
11	9	PB0/MCVREF	I/O	C_T		X	X		X	X	X	Port B0	MTC voltage reference	
12	10	PB1/MCIA	I/O	C_T		X	X		X	X	X	Port B1	MTC input A	
13	11	PB2/MCIB	I/O	C_T		X	X		X	X	X	Port B2	MTC input B	
14	12	PB3/MCIC	I/O	C_T		X	X		X	X	X	Port B3	MTC input C	
15	(5)	PB4/MISO	I/O	C_T		X	X			X	X	Port B4	SPI master in/slave out data	
16		PB5/MOSI/AIN3	I/O	C_T		X	X			X	X	Port B5	SPI master out/slave in data	ADC analog input 3
17		PB6/SCK	I/O	C_T	HS	X	ei2			X	X	Port B6	SPI serial clock	
18		PB7/SS/AIN4	I/O	C_T	HS	X		ei2		X	X	Port B7	SPI slave select (active low)	ADC analog input 4
(5)		PG4	I/O	T_T		X	X			X	X	Port G4		
		PG5	I/O	T_T		X	X			X	X	Port G5		
		PG6	I/O	T_T		X	X			X	X	Port G6		
		PG7	I/O	T_T		X	X			X	X	Port G7		
		PC0	I/O	C_T	HS	X		ei2		X	X	Port C0		
(5)	(5)	PC1/MCCFI0[8]/AIN5	I/O	C_T		X	ei2		X	X	X	Port C1	MTC current feedback input 0[8]	ADC analog input 5
19	13	PC2/OAP	I/O	C_T		X		ei2	X	X	X	Port C2	Op-amp positive input	
20	14	PC3/OAN	I/O	C_T		X	X	ei2	X	X	X	Port C3	Op-amp negative input	

Table 2. Device pin description[1] (continued)

Pin number		Pin name	Type	Level		Port						Main function (after reset)	Alternate function[2]	
						Input				Output				
LQFP44	LQFP32			Input	Output	float	wpu	int[3]	ana	OD	PP			
21	15	OAZ/MCCFI1[8]/AIN6	I/O						X			Op-amp Output	MTC current feedback input 1[8]	ADC analog input 6
22	16	PC4/MCCREF	I/O	C_T		**X**	X		X	X	X	Port C4	MTC current feedback reference[9]	
(5)	(5)	PC5/MCPWMU	I/O	C_T		**X**	X			X	X	Port C5	MTC PWM output U	
		PC6/MCPWMV[10]	I/O	C_T		**X**	X			X	X	Port C6	MTC PWM output V[10]	
23	(5)	PC7/MCPWMW[10]/AIN7	I/O	C_T		**X**	X		X	X	X	Port C7	MTC PWM output W[10]	ADC analog input 7
24	17	V_{AREF}	I									Analog reference voltage for ADC		
25	(5)	V_{SSA}[7]	S									Analog ground voltage		
26	18	V_{SS_0}[7]	S									Digital ground voltage		
27	19	V_{DD_0}[7]	S									Digital main supply voltage		
28	20	$\overline{\text{RESET}}$	I/O	C_T								Top priority non maskable interrupt		
(5)	(5)	PF0/MCDEM[11]/AIN8	I/O	C_T		**X**	X		X	X	X	Port F0	MTC demagnetization output[11]	ADC analog input 8
		PF1/MCBEMF[11]/AIN9	I/O	CT		**X**	X		X	X	X	Port F1	MTC BEMF output[11]	ADC analog input 9
		PF2/MCO/AIN10	I/O	C_T		**X**	X		X	X	X	Port F2	Main clock out (f_{OSC}/2)	ADC analog input 10
		PF3/Beep	I/O	C_T	HS	**X**	X			X	X	Port F3	Beep signal output	
		PF4	I/O	C_T	HS	**X**	X			X	X	Port F4		
		PF5	I/O	C_T	HS	**X**	X			X	X	Port F5		
		PH0	I/O	T_T		**X**	X			X	X	Port H0		
(5)	(5)	PH1	I/O	T_T		**X**	X			X	X	Port H1		
		PH2	I/O	T_T		**X**	X			X	X	Port H2		
		PH3	I/O	T_T		**X**	X			X	X	Port H3		

Table 2. Device pin description[1] (continued)

Pin number		Pin name	Type	Level		Port						Main function (after reset)	Alternate function[2]	
						Input				Output				
LQFP44	LQFP32			Input	Output	float	wpu	int[3]	ana	OD	PP			
29	21	PD0/OCMP2_A/ MCPWMW[10]/AIN11	I/O	C_T		X			X	X	X	Port D0	Timer A output compare 2	
													MTC PWM output W[10]	
													ADC analog input 11	
30	22	PD1(HS)/OCMP1_A/ MCPWMV[10]/ MCDEM[11]	I/O	C_T	HS	X		ei0		X	X	Port D1	Timer A output compare 1	
													MTC PWM output V[10]	
													MTC demagnetization[11]	
31	23	PD2/ICAP2_A/ MCZEM[11]/AIN12	I/O	C_T		X		ei0		X	X	Port D2	Timer A input capture 2	
													MTC BEMF[11]	
													ADC analog input 12	
32	24	PD3/ICAP1_A/AIN13	I/O	C_T		X		ei0		X	X	Port D3	Timer A input capture 1	ADC analog input 13
33	25	PD4/EXTCLK_A/ICCCLK/ AIN14	I/O	C_T		X		ei0		X	X	Port D4	Timer A external clock source	
													ICC clock output	
													ADC analog input 14	
34	26	PD5/ICCDATA/AIN15	I/O	C_T		X		ei0		X	X	Port D5	ICC data input	
													ADC analog input 15	
35	27	PD6/RDI	I/O	C_T	HS	X		ei0		X	X	Port D6	SCI receive data in	
36	28	PD7/TDO	I/O	C_T	HS	X	X			X	X	Port D7	SCI transmit data output	
(5)	(5)	V_{SS_2}[7]	S									Digital ground voltage		
		V_{DD_2}[7]	S									Digital main supply voltage		
		PH4	I/O	T_T		X	X			X	X	Port H4		
		PH5	I/O	T_T		X	X			X	X	Port H5		
		PH6	I/O	T_T		X	X			X	X	Port H6		

Table 2. Device pin description[1] (continued)

Pin number		Pin name	Type	Level		Port						Main function (after reset)	Alternate function[2]
						Input				Output			
LQFP44	LQFP32			Input	Output	float	wpu	int[3]	ana	OD	PP		
(5)	(5)	PH7	I/O	T_T		X	X			X	X	Port H7	
37		PE0/OCMP2_B	I/O	C_T	HS	X	X			X	X	Port E0	Timer B output compare 2
38		PE1/OCMP1_B	I/O	C_T		X	X		X	X	X	Port E1	Timer B output compare 1
39		PE2/ICAP2_B	I/O	C_T		X	X			X	X	Port E2	Timer B input capture 2
40		PE3/ICAP1_B/	I/O	C_T		X	X		X	X	X	Port E3	Timer B input capture 1
(5)		PE4/EXTCLK_B	I/O	C_T		X	X			X	X	Port E4	Timer B external clock source
		PE5	I/O	C_T		X	X		X	X	X	Port E5	
41	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 334*	
42	30	MCO0 (HS)	O		HS						X	MTC output channel 0	
43	31	MCO1 (HS)	O		HS						X	MTC output channel 1	
44	32	MCO2 (HS)	O		HS						X	MTC output channel 2	

1. On the chip, each I/O port has eight 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 15.6 on page 366*
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 are ORed on the output pin.
3. In the interrupt column, 'eiX' defines the associated external interrupt vector. If 'wpu' is merged with 'int', then I/O configuration is pull-up interrupt input, otherwise the configuration is floating interrupt input.
4. MCES is a floating input. To disable this function, a pull-up resistor must be used.
5. Pin(s) not present on package configuration.
6. OSC1 and OSC2 pins connect a crystal/ceramic resonator or an external source to the on-chip oscillator; see *Section 1: Description* and *Section 12.5: Clock and timing characteristics* for more details.
7. 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.
8. MCCFI can be mapped on two different pins on 80-, 64- and 56-pin packages. This allows:
   - either using PC1 as a standard I/O and mapping MCCFI on OAZ (MCCFI1) with or without using the operational amplifier (selected case after reset),
   - or mapping MCCFI on PC1 (MCCFI0) and using the amplifier for another function. The mapping can be selected in the MREF register of Motor Control cell (see *Motor controller (MTC)*) for more details.
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.
10. 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.
11. 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.

3 Register and memory map

As shown in *Figure 4* and *Figure 5: Memory map and sector address on page 34*, the MCU is capable of addressing 64 Kbytes of memories and I/O registers.

The available memory locations consist of 128 bytes of register locations, up to 1 Kbytes of RAM and up to 32 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.

Caution: Memory locations marked as 'reserved' must never be accessed. Accessing a reserved area can have unpredictable effects on the device.

Figure 4. Memory map

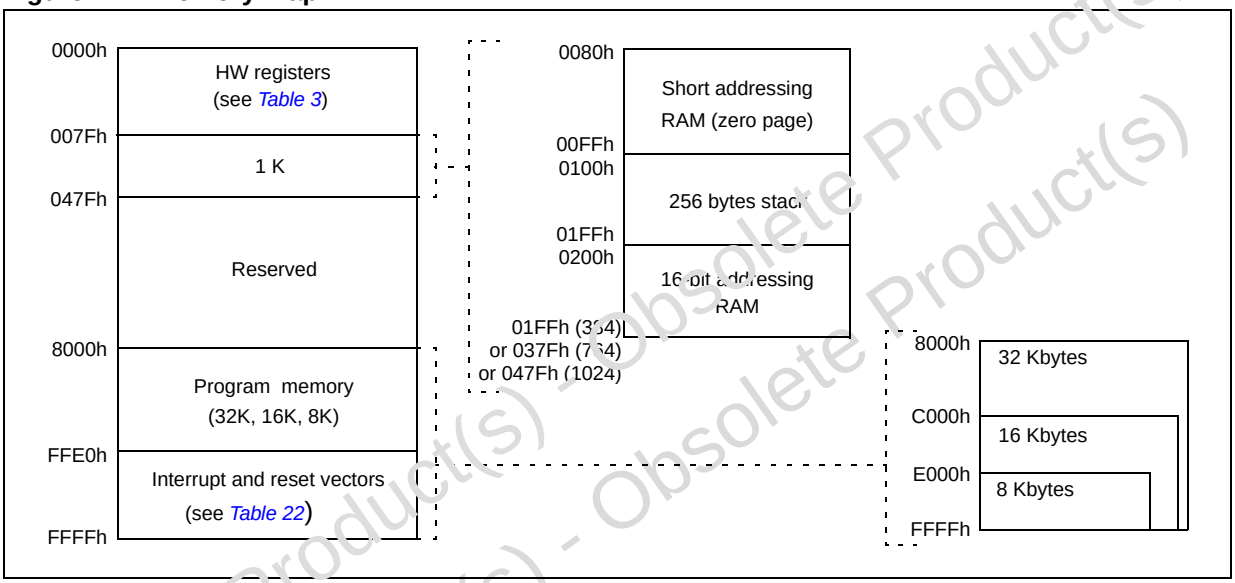


Table 3. Hardware register map

Address	Block	Register label	Register name	Reset status	Remarks
0000h	Port A	PADR	Port A data register	00h(1)	R/W(2)
0001h		PADDR	Port A data direction register	00h	R/W
0002h		PAOR	Port A option register	00h	R/W(3)
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(3)
000Eh		PEOR	Port E option register	00h	R/W(3)
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(4)	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
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

Table 3. Hardware register map (continued)

Address	Block	Register label	Register name	Reset status	Remarks
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 3. Hardware register map (continued)

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 and 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	deadtime 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 3. Hardware register map (continued)

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 register	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	Op-amp	OACSR	Op-amp control/status register	00h	R/W

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. R/W = read/write
3. The bits associated with unavailable pins must always keep their reset value.
4. x = undefined.

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 organization 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 organized in sectors and can be used for both code and data storage.

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

The first two sectors have a fixed size of 4 Kbytes (see *Figure 5*). They are mapped in the upper part of the ST7 addressing space so the reset and interrupt vectors are located in Sector 0 (F000h-FFFFh).

Table 4. Sectors available in Flash devices

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

4.3.1 Read-out protection

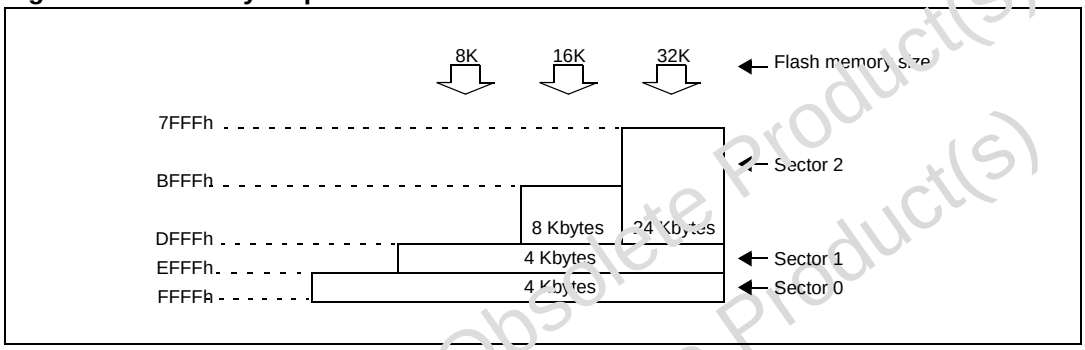
Read-out protection, when selected, provides a protection against Program Memory content extraction and against write access to Flash memory. Even if no protection can be considered as totally unbreakable, the feature provides a very high level of protection for a general purpose microcontroller.

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 5. Memory map and sector address

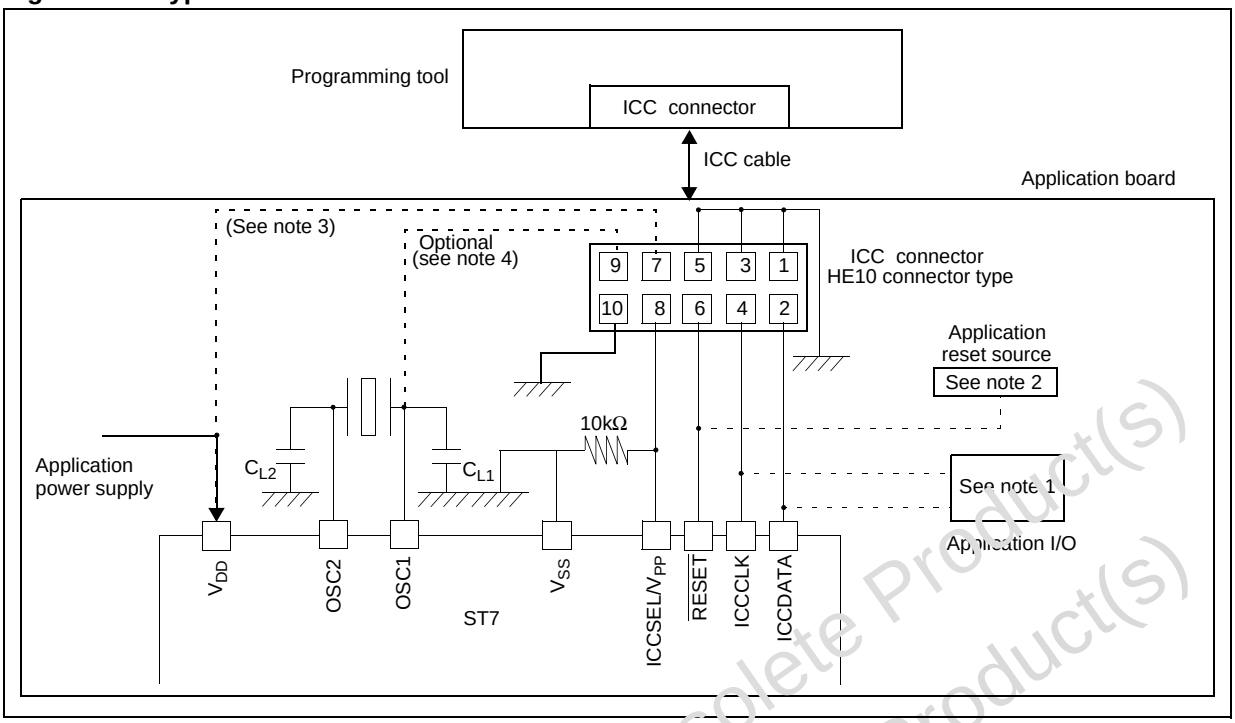


4.4 ICC interface

ICC (in-circuit communication) needs a minimum of 4 and up to 6 pins to be connected to the programming tool (see *Figure 6*). 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 6*, Note 3)

Figure 6. Typical ICC interface



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.

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 6*). 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 Flash control status register (FCSR)

FCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
0							
R/W							

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 7* are not present in the memory mapping and are accessed by specific instructions.

5.3.1 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.

5.3.2 Index registers (X and Y)

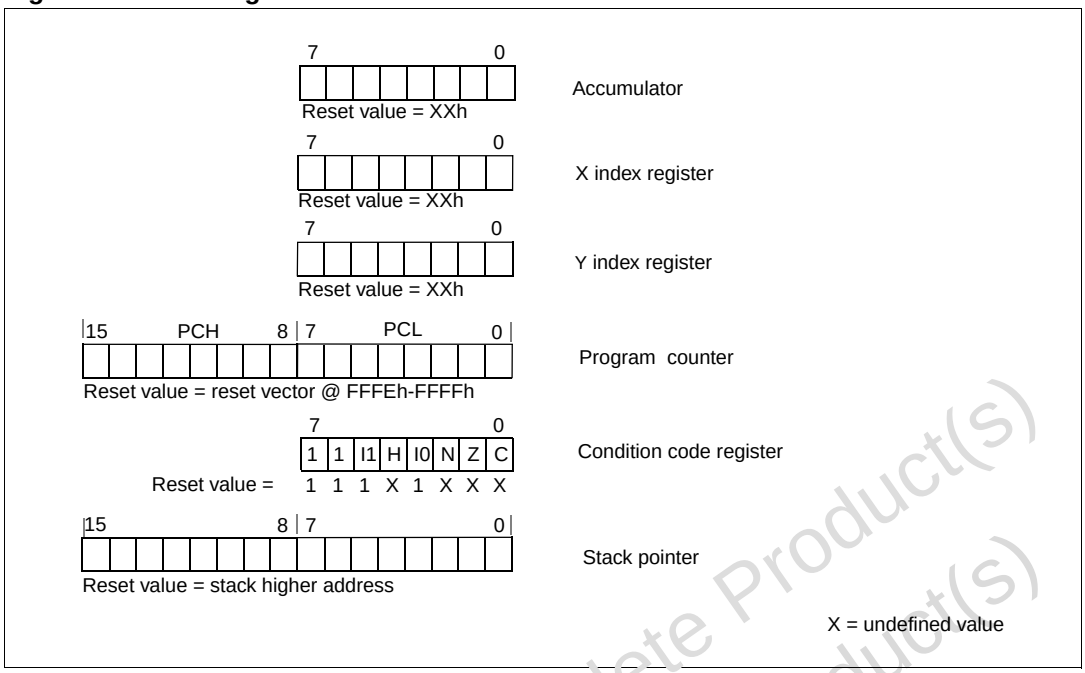
These 8-bit registers are used to create effective addresses or as temporary storage areas for data manipulation (the cross-assembler generates a precede instruction (PRE) to indicate that the following instruction refers to the Y register).

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

5.3.3 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 7. CPU registers



5.3.4 Condition code register (CC)

CC Reset value: 111x 1xxx

7	6	5	4	3	2	1	0
1		I1	H	I0	N	Z	C
R/W		R/W	R/W	R/W	R/W	R/W	R/W

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.

Table 5. CC register description

Bit	Name	Function
5, 3	I1, I0 (interrupt)	Interrupt management bits The combination of the I1 and I0 bits gives the current interrupt software priority: 10: Interrupt software priority = level 0 (main) 01: Interrupt software priority = level 1 00: Interrupt software priority = level 2 11: Interrupt software priority = level 3 (interrupt disable) 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 *Section 7: Interrupts on page 59* for more details.
4	H (half carry)	Arithmetic management bit 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.
2	N (negative)	Arithmetic management bit 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 7th 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.

Table 5. CC register description (continued)

Bit	Name	Function
1	Z (zero)	Arithmetic management bit 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.
0	C (carry/borrow)	Arithmetic management bit 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.

5.3.5 Stack pointer register (SP)

SP Reset value: 01 FFh

15	14	13	12	11	10	9	8
0							1
R/W							R/W

7	6	5	4	3	2	1	0
SP[7:0]							
R/W							R/W

The stack pointer is a 16-bit register which always points to the next free location in the stack. It is decremented after data has been pushed onto the stack and incremented before data is popped from the stack (see *Figure 8*).

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 LSB of the stack pointer (called S) can be directly accessed by an 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.

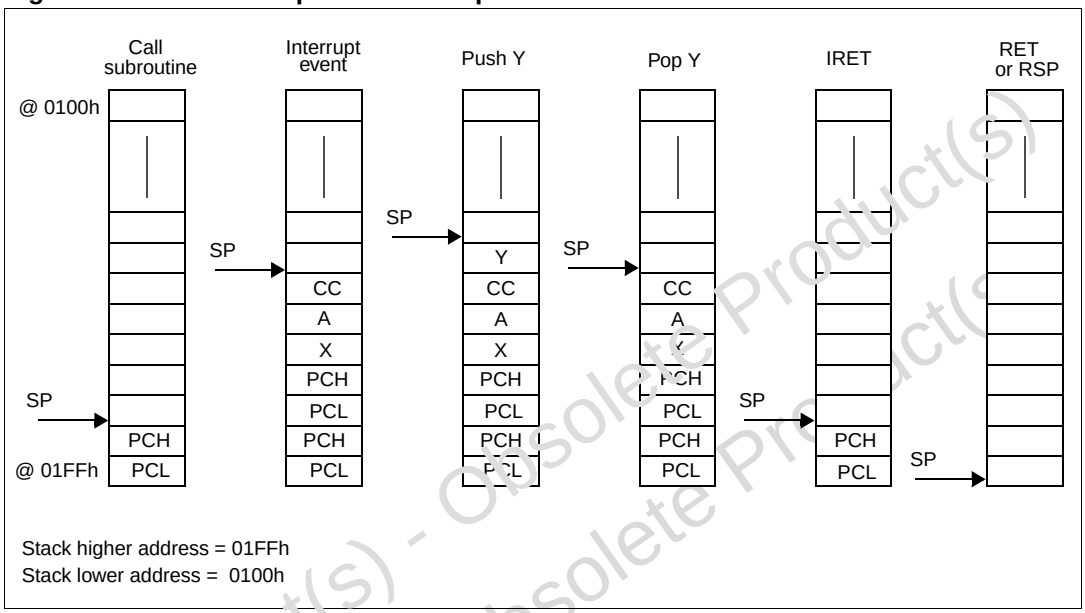
The stack is used to save the return address during a subroutine call and the CPU context during an interrupt. The user may also directly manipulate the stack by means of the push and pop instructions. In the case of an interrupt, the PCL is stored at the first location

pointed to by the SP. Then the other registers are stored in the next locations as shown in *Figure 8*.

- 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 8. Stack manipulation example



6 Supply, reset and clock management

6.1 Introduction

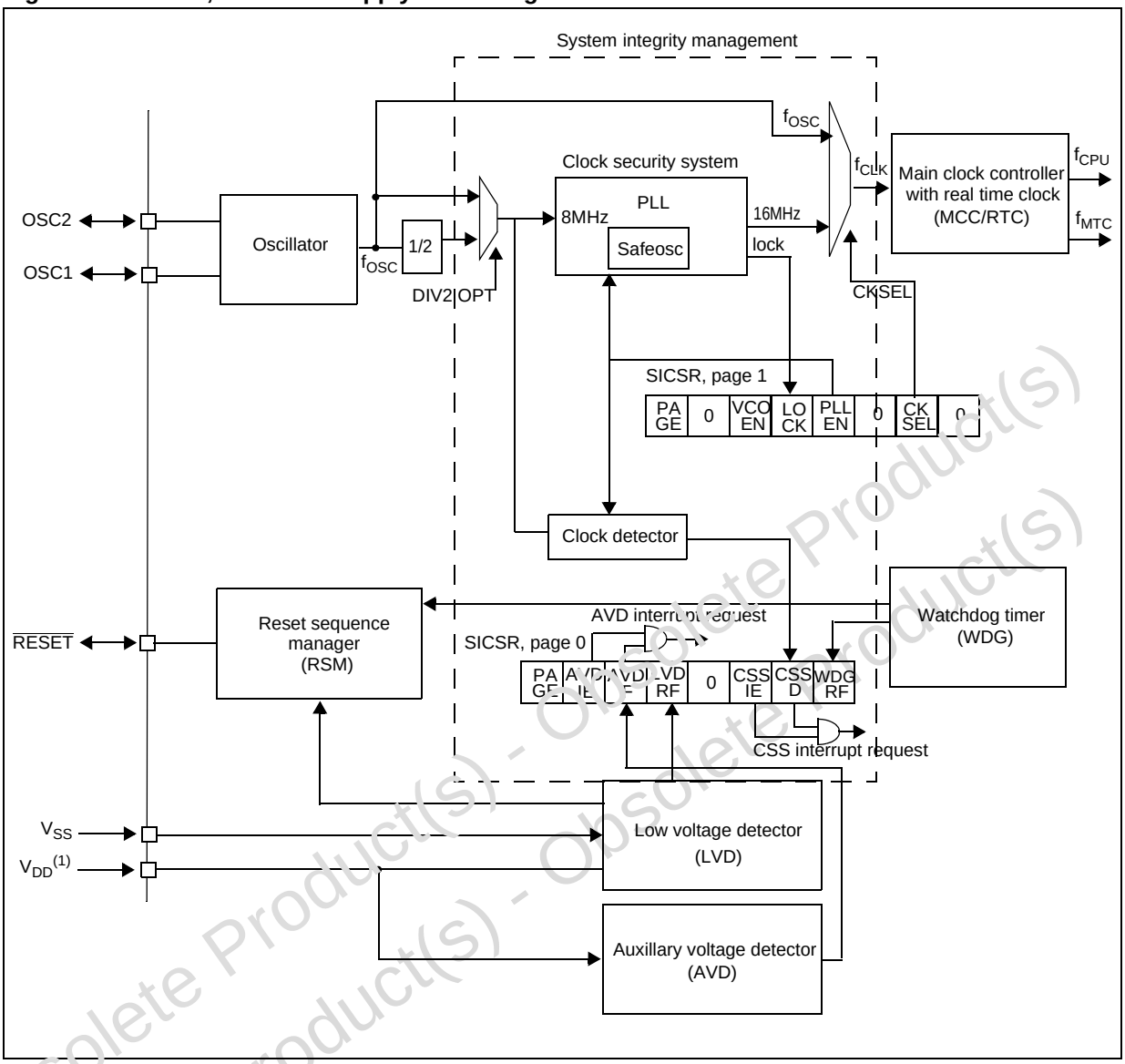
The device includes a range of utility features for securing the application in critical situations (for example in case of a power brown-out), and reducing the number of external components. An overview is shown in *Figure 9*.

For more details, refer to dedicated parametric section.

6.2 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 9. Clock, reset and supply block diagram



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

6.3 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 6*. Refer to the electrical characteristics section for more details.

6.3.1 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.

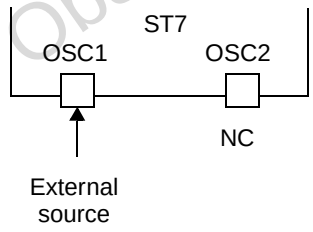
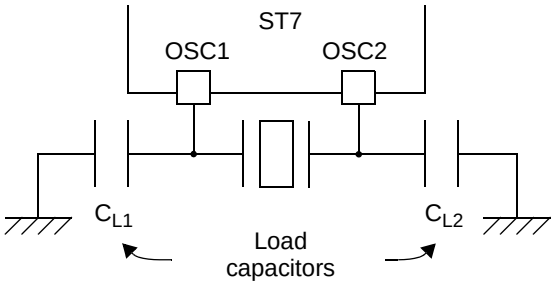
Crystal/ceramic oscillators

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 *Section 12: 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 6. ST7 clock sources

	Hardware configuration
External clock	
Crystal/ceramic resonators	

6.4 Reset sequence manager (RSM)

6.4.1 Introduction

The reset sequence manager includes three reset sources are shown in *Figure 11*.

- 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 11.2.2 on page 309 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 three phases as shown in *Figure 10*.

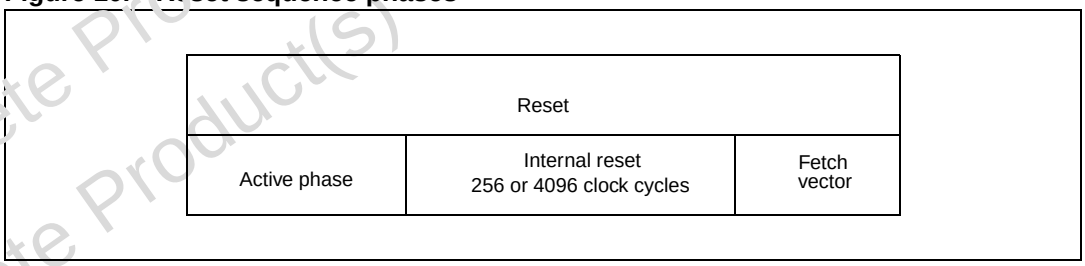
- Active phase depending on the reset source
- 256 or 4096 CPU clock cycle delay (selected by option byte)
- Reset vector fetch

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 $\overline{\text{RESET}}$ 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 stabilize 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 10. Reset sequence phases

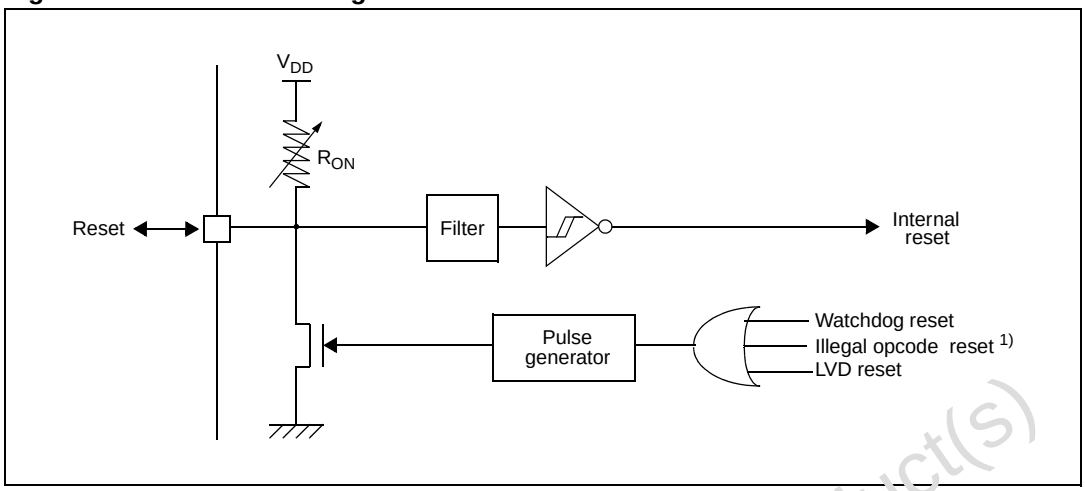


6.4.2 Asynchronous external $\overline{\text{RESET}}$ pin

The $\overline{\text{RESET}}$ pin is both an input and an open-drain output with integrated R_{ON} weak pull-up resistor. This pull-up has no fixed value but varies in accordance with the input voltage. It can be pulled low by external circuitry to reset the device. See *Section 12: Electrical characteristics* 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 12*). This detection is asynchronous and therefore the MCU can enter reset state even in Halt mode.

Figure 11. Reset block diagram



1. See *Section 11.2.2: Illegal opcode reset on page 309* for more details on illegal opcode reset conditions.

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.

6.4.3 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.

6.4.4 Internal low voltage detector (LVD) reset

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

- Power-on reset
- Voltage drop reset

The device $\overline{\text{RESET}}$ pin acts as an output that is pulled low when $V_{DD} < V_{IT+}$ (rising edge) or $V_{DD} < V_{IT-}$ (falling edge) as shown in *Figure 12*.

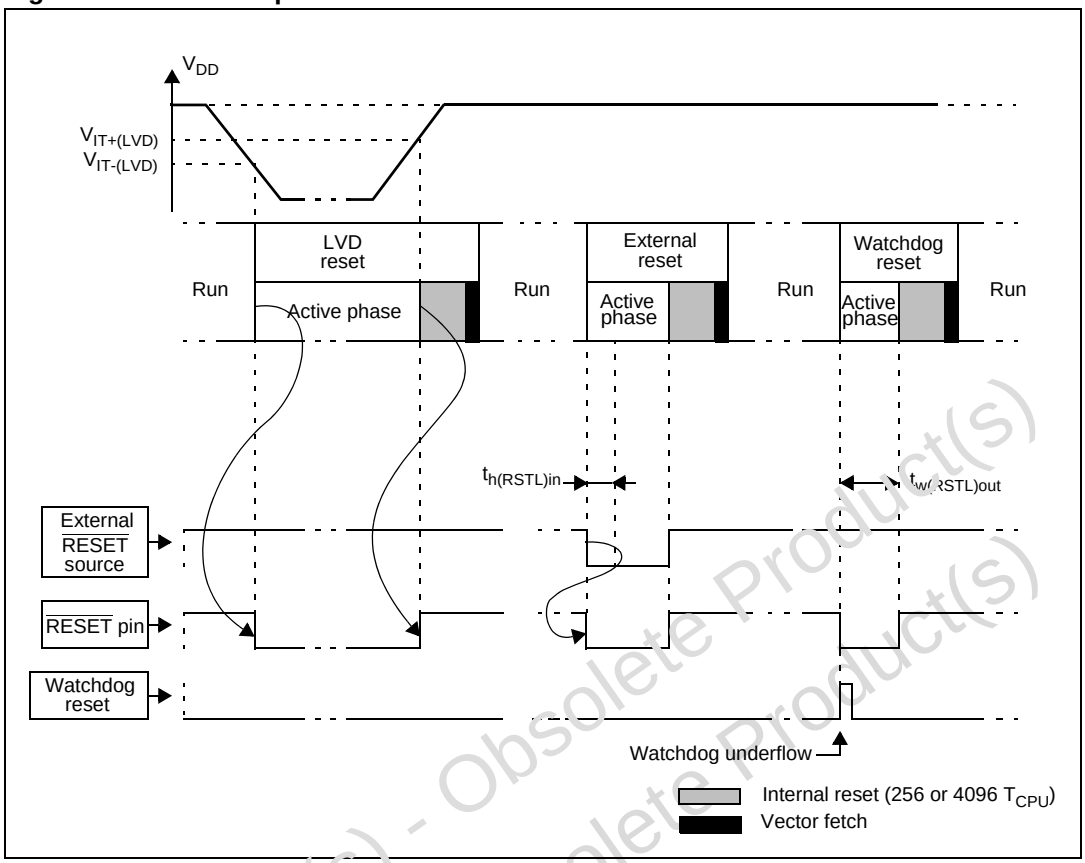
The LVD filters spikes on V_{DD} larger than $t_{g(VDD)}$ to avoid parasitic resets.

6.4.5 Internal watchdog reset

The RESET sequence generated by a internal watchdog counter overflow is shown in *Figure 12*.

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 12. Reset sequences



6.5 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 11.2.2 on page 309 for further details.

6.5.1 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 13*.

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.

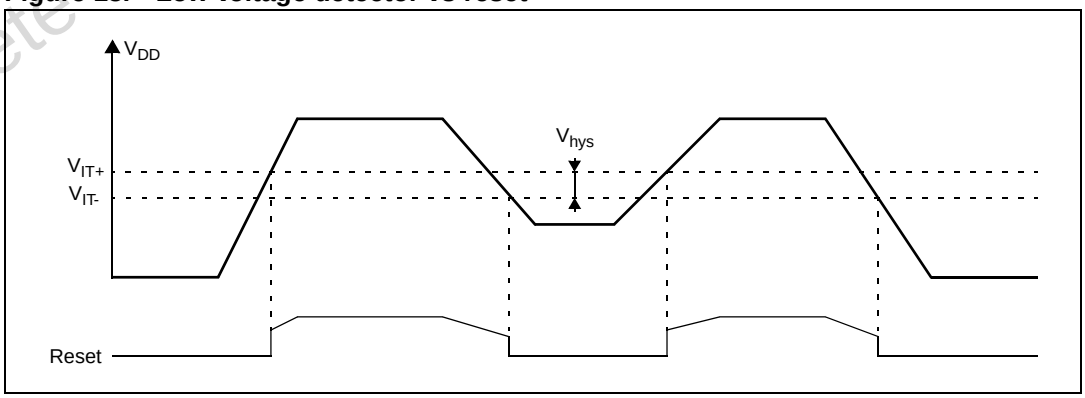
During a low voltage detector reset, the $\overline{\text{RESET}}$ pin is held low, thus permitting the MCU to reset other devices.

Note: 1 The LVD allows the device to be used without any external reset circuitry.

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

3 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 13. Low voltage detector vs reset



6.5.2 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 356*).

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 14*.

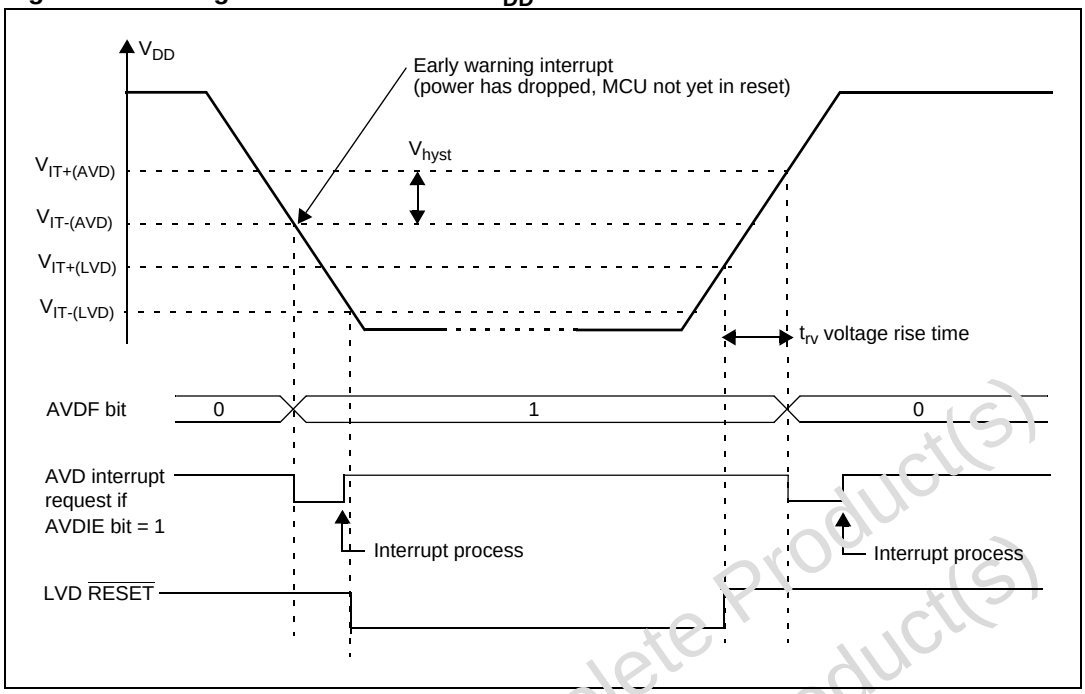
The interrupt on the rising edge is used to inform the application that the V_{DD} warning state is over.

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 is 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 two AVD interrupts are 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 occurs.

Figure 14. Using the AVD to monitor V_{DD}



6.5.3 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.

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 ST7MC1K2-Auto, ST7MC1K26Auto, ST7MC2S4-Auto, and ST7MC2S6-Auto do not start and are maintained in reset conditions.

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.

6.5.4 Low power modes

Table 7. Effect of low power modes on SI

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.

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).

Table 8. SI interrupt control/wake-up capability

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

1. This interrupt allows to exit from Active Halt mode.

6.5.5 System integrity control/status register (SICSR, page 0)

SICSR, page 0 Reset value: 000x 000x (00h)

7	6	5	4	3	2	1	0
PAGE	AVDIE	AVDF	LVDRF	Reserved	CSSIE	CSSD	WDGRF
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Table 9. SICSR (page 0) register description

Bit	Name	Function
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
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

Table 9. SICSR (page 0) register description (continued)

Bit	Name	Function
5	AVDF	Voltage detector flag This read-only bit is set and cleared by hardware. If the AVDIE 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
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.
3	-	Reserved, must be kept cleared
2	CSSIE	Clock security system 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.
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 must be kept cleared.
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 below: 00: Reset sources = external $\overline{\text{RESET}}$ pin 01: Reset sources = WDG 1X: Reset sources = LVD

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.

6.5.6 System integrity control/status register (SICSR, page 1)

SICSR, page 1 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
PAGE	Reserved	VCOEN	LOCK	PLLEN	Reserved	CKSEL	Reserved
R/W	-	R/W	RO	R/W	-	R/W	-

Table 10. SICSR (page 1) register description

Bit	Name	Function
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
6	-	Reserved, must be kept cleared
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}) *Note: 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.*
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
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:* *- During ICC session, this bit is set to 1.* *- PLL cannot be disabled if the PLL clock source is selected (CKSEL = 1).*
2	-	Reserved, must be kept cleared.
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:* *- During ICC session, this bit is set to 1. Then, CKSEL can be reset in order to run with f_{OSC}.* *- Clock from the PLL cannot be selected if the PLL is disabled (PLLEN = 0).* *- f the clock source is selected by PLL option bit, CKSEL bit selection has no effect.*
0	-	Reserved, must be kept cleared

6.6 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.

6.6.1 Programmable CPU clock prescaler

The programmable CPU clock prescaler supplies the clock for the ST7 CPU and its internal peripherals. It manages the 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.

6.6.2 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.

6.6.3 Real time clock timer (RTC)

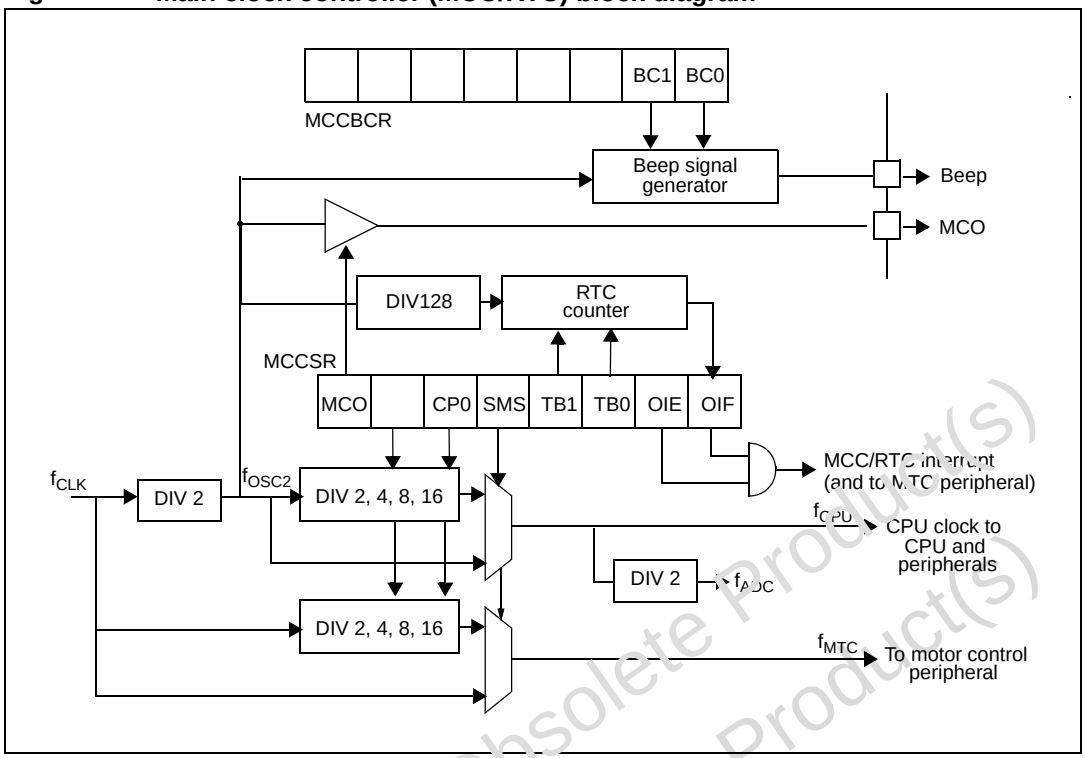
The counter of the real time clock timer allows an interrupt to be generated based on an accurate real time clock. Four different time bases depending directly on f_{OSC2} are available. The whole functionality is controlled by four bits of the MCCSR register: TB[1:0], OIE and OIF.

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.

6.6.4 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 15. Main clock controller (MCC/RTC) block diagram



6.6.5 Low power modes

Table 11. Effect of low power modes on MCC/RTC

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.

6.6.6 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).

Table 12. MCC/RTC interrupt control/wake-up capability

Interrupt event	Event flag	Enable control bit	Exit from WAIT	Exit from HALT
Time base overflow event	OIF	OIE	Yes	No(1)

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

6.6.7 MCC control status register (MCCSR)

MCCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
MCO	CP[1:0]		SMS	TB[1:0]		OIE	OIF
R/W	R/W		R/W	R/W		R/W	R/W

Table 13. MCCSR register description

Bit	Name	Function
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.*
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: 00: f_{CPU} in slow mode = $f_{OSC2}/2$ 01: f_{CPU} in slow mode = $f_{OSC2}/4$ 10: f_{CPU} in slow mode = $f_{OSC2}/8$ 11: f_{CPU} in slow mode = $f_{OSC2}/16$
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 6.6: Main clock controller with real time clock and beeper (MCC/RTC)* for more details.

Table 13. MCCSR register description (continued)

Bit	Name	Function
3:2	TB[1:0]	Time base control These bits select the programmable divider time base. They are set and cleared by software: 00: Time base (counter prescaler 16000) = 4 ms (f_{OSC2} = 4 MHz) and 2 ms (f_{OSC2} = 8 MHz) 01: Time base (counter prescaler 32000) = 8 ms (f_{OSC2} = 4 MHz) and 4 ms (f_{OSC2} = 8 MHz) 10: Time base (counter prescaler 80000) = 20 ms (f_{OSC2} = 4 MHz) and 10 ms (f_{OSC2} = 8 MHz) 11: Time base (counter prescaler 200000) = 50 ms (f_{OSC2} = 4 MHz) and 25 ms (f_{OSC2} = 8 MHz) 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 this time base to be used as a real time clock.
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.
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.

6.6.8 MCC beep control register (MCCBCR)

MCCBCR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved				ADSTS	ADC IE	BC[1:0]	
-				R/W	R/W	R/W	

Table 14. MCCBCR register description

Bit	Name	Function
7:4	-	Reserved, must be kept cleared
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)
2	ADC IE	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
1:0	BC[1:0]	Beep control These 2 bits select the PF1 pin beep capability: 00: Beep mode (with f_{OSC2} = 8 MHz) = off 01: Beep mode (with f_{OSC2} = 8 MHz) = ~2 kHz (output beep signal ~50% duty cycle) 10: Beep mode (with f_{OSC2} = 8 MHz) = ~1 kHz (output beep signal ~50% duty cycle) 11: Beep mode (with f_{OSC2} = 8 MHz) = ~500 Hz (output beep signal ~50% duty cycle) The beep output signal is available in Active Halt mode but has to be disabled to reduce the consumption.

6.6.9 Main clock controller register map and reset values

Table 15. 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 16*). The processing flow is shown in *Figure 16*.

When an interrupt request has to be serviced:

- Normal processing is suspended at the end of the current instruction execution.
- The PC, X, A and CC registers are saved onto the stack.
- I1 and I0 bits of CC register are set according to the corresponding values in the ISPRx registers of the serviced interrupt vector.
- The PC is then loaded with the interrupt vector of the interrupt to service and the first instruction of the interrupt service routine is fetched (refer to *Table 22: Interrupt mapping* for vector addresses).

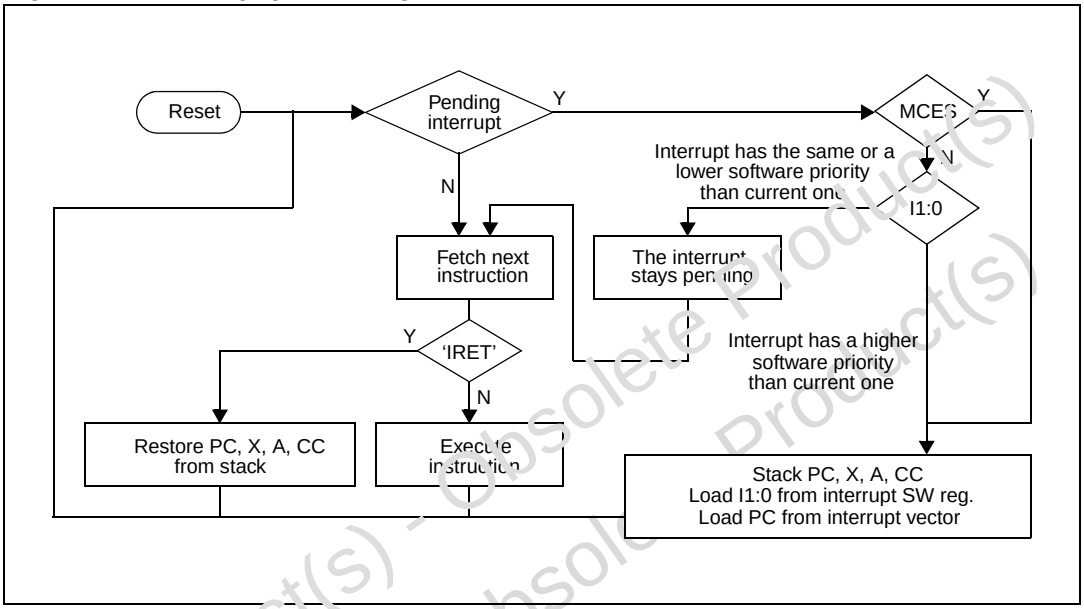
The interrupt service routine should end with the IRET instruction which causes the contents of the saved registers to be recovered from the stack.

Note: As a consequence of the IRET instruction, the I1 and I0 bits are restored from the stack and the program in the previous level is resumed.

Table 16. Interrupt software priority levels

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

Figure 16. Interrupt processing flowchart



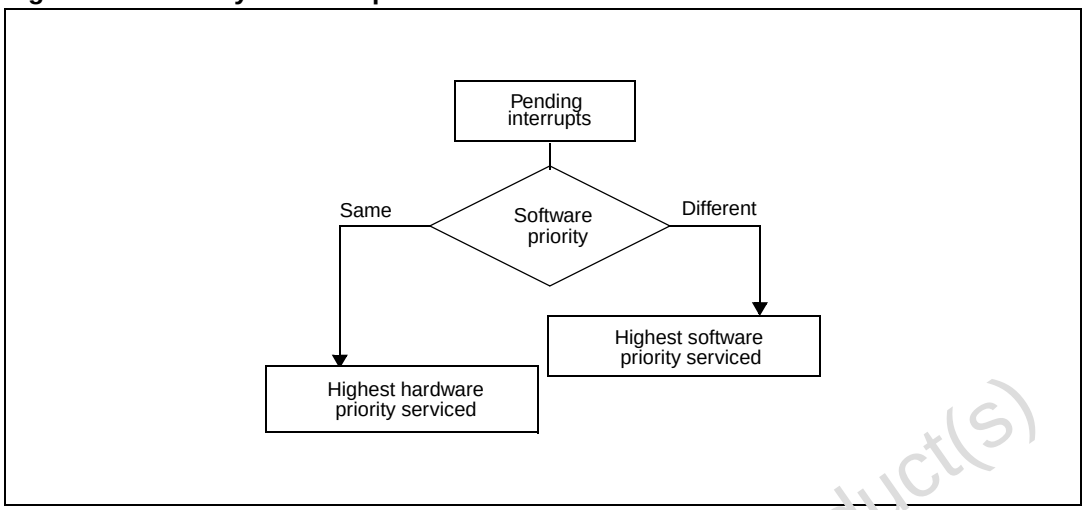
7.2.1 Servicing pending interrupts

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

- the highest software priority interrupt is serviced,
- if several interrupts have the same software priority then the interrupt with the highest hardware priority is serviced first.

Figure 17 describes this decision process.

Figure 17. Priority decision process



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

Note: 1 The hardware priority is exclusive while the software one is not. This allows the previous process to succeed with only one interrupt.

2 Reset, TRAP and MCES can be considered as having the highest software priority in the decision process.

7.2.2 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).

7.2.3 Non-maskable sources

These sources are processed regardless of the state of the I1 and I0 bits of the CC register (see *Figure 16*). After stacking the PC, X, A and CC registers (except for reset), the corresponding vector is loaded in the PC register and the I1 and I0 bits of the CC are set to disable interrupts (level 3). These sources allow the processor to exit Halt mode.

- TRAP (non maskable software interrupt)

  This software interrupt is serviced when the TRAP instruction is executed. It is serviced according to the flowchart in *Figure 16* 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 *Section 6.4: Reset sequence manager (RSM)* for more details.

7.2.4 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 is 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 are logically ORed.

- Peripheral interrupts:

  Usually the peripheral interrupts cause the MCU to exit from Halt mode except those mentioned in *Table 22: Interrupt mapping*.

  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. Therefore, a pending interrupt (that is, an interrupt waiting to be serviced) is lost if the clear sequence is executed.

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 *Table 22: Interrupt mapping*). When several pending interrupts are present while exiting Halt mode, the first one serviced can only be an interrupt with exit from Halt mode capability and it is selected through the same decision process shown in *Figure 17*.

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

7.4 Concurrent and nested management

Figure 18 and *Figure 19* show two different interrupt management modes. The first is called concurrent mode and does not allow an interrupt to be interrupted, unlike the nested mode in *Figure 19*. The interrupt hardware priority is given in this order from the lowest to the highest: MAIN, IT4, IT3, IT2, IT1, IT0, MCES. The software priority is given for each interrupt.

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

Figure 18. Concurrent interrupt management

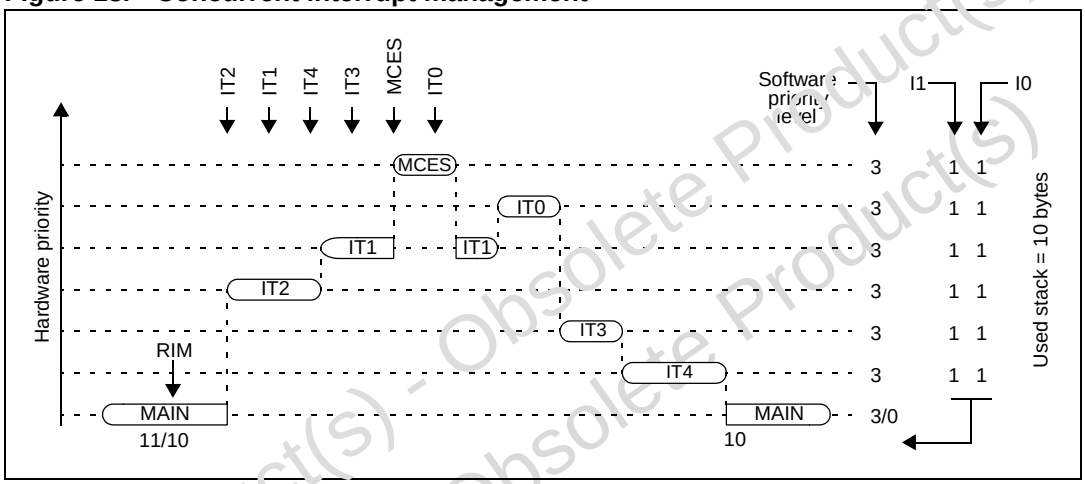
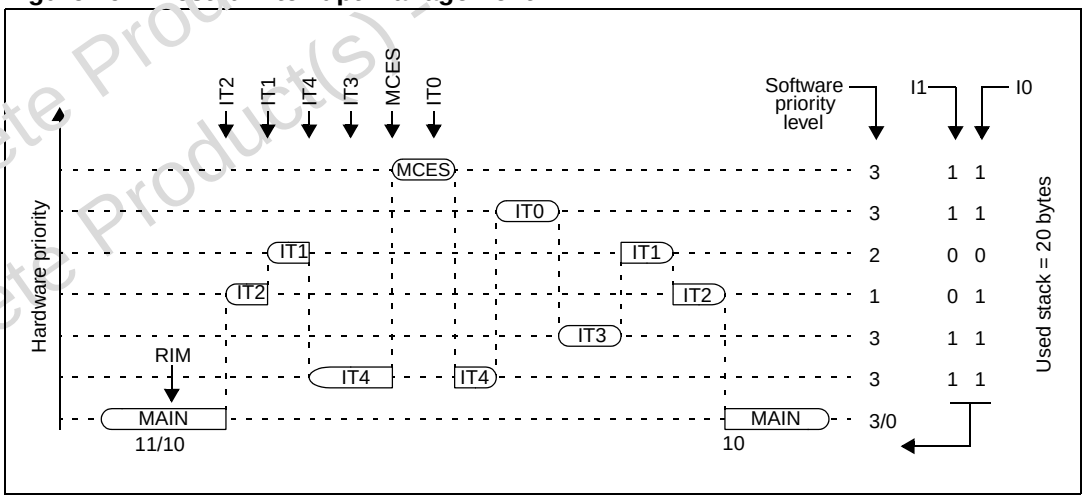


Figure 19. Nested interrupt management



7.5 Interrupt registers

7.5.1 CPU CC register interrupt bits

CPU CC register — Reset value: 111x 1010 (xAh)

7	6	5	4	3	2	1	0
1		I1	H	I0	N	Z	C
R/W		R/W	R/W	R/W	R/W	R/W	R/W

Table 17. CPU CC register interrupt bits description

Bit	Name	Function
5, 3	I1, I0	Software interrupt priority These two bits indicate the current interrupt software priority: 10: Interrupt software priority = level 0 (main) 01: Interrupt software priority = level 1 00: Interrupt software priority = level 2 11: Interrupt software priority = level 3 (interrupt disable)(1) These 2 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 *Table 19: Dedicated interrupt instruction set*).

1. MCES, TRAP and reset events can interrupt a level 3 program.

7.5.2 Interrupt software priority registers (ISPRX)

ISPR0 Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
I1_3	I0_3	I1_2	I0_2	I1_1	I0_1	I1_0	I0_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ISPR1 Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
I1_7	I0_7	I1_6	I0_6	I1_5	I0_5	I1_4	I0_4
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ISPR2 Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
I1_11	I0_11	I1_10	I0_10	I1_9	I0_9	I1_8	I0_8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ISPR3 Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
1	1	1	1	I1_13	I0_13	I1_12	I0_12
RO	RO	RO	RO	R/W	R/W	R/W	R/W

These four registers contain the interrupt software priority of each interrupt vector.

- Each interrupt vector (except reset and TRAP) has corresponding bits in those registers where its own software priority is stored. This correspondence is shown in *Table 18*.
- 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.

Table 18. ISPRx interrupt vector correspondence

Vector address	ISPRx bits
FFFBh-FFFAh	I1_0 and I0_0 bits(1)
FFF9h-FFF8h	I1_1 and I0_1 bits
FFE1h-FFE0h	I1_13 and I0_13 bits

1. 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 behavior 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).

7.6 Interrupt instructions

Table 19. Dedicated interrupt instruction set(1)

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 IMI	1		1			
WFI	Wait for interrupt		1		0			

1. 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.

7.7 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}.

7.7.1 I/O port interrupt sensitivity

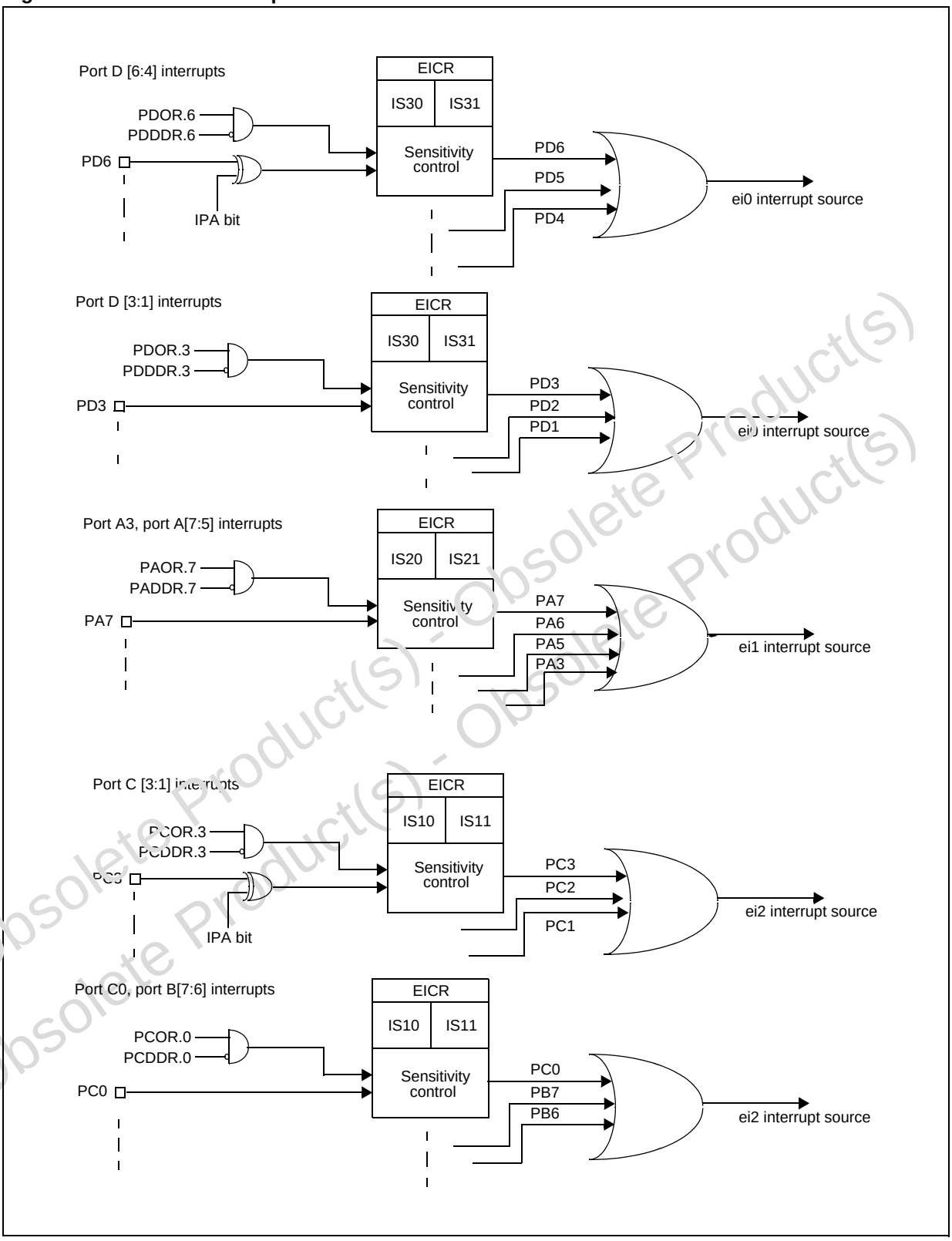
The external interrupt sensitivity is controlled by the IPA, IPB and ISxx bits of the EICR register (*Figure 20*). This control allows to have up to four fully independent external interrupt source sensitivities.

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).

Figure 20. External interrupt control bits



7.7.2 External interrupt control register (EICR)

EICR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
IS1[1:0]		IPB	IS2[1:0]		IS3[1:0]		IPA
R/W		R/W	R/W		R/W		R/W

Table 20. EICR register description

Bit	Name	Function
7:6	IS1[1:0]	Interrupt sensitivity (ei2 sensitivity) The interrupt sensitivity, defined using the IS1[1:0] bits, is applied to the following external interrupts: **External interrupt ei2 (port C[3:2]):** 00: External interrupt sensitivity = falling edge and low level (IPB bit = 0) and rising edge and high level (IPB bit = 1) 01: External interrupt sensitivity = rising edge only (IPB bit = 0) and falling edge only (IPB bit = 1) 10: External interrupt sensitivity = falling edge only (IPB bit = 0) and rising edge only (IPB bit = 1) 11: External interrupt sensitivity = rising and falling edge (IPB bit = 0 and IPB bit = 1) **External interrupt ei2 (port B[7:6]):** 00: External interrupt sensitivity = falling edge and low level 01: External interrupt sensitivity = rising edge only 10: External interrupt sensitivity = falling edge only 11: External interrupt sensitivity = 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).
5	IPB	Interrupt polarity (for port C) This bit is used to invert the sensitivity of port C[3:2] 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
4:3	IS2[1:0]	Interrupt sensitivity (ei1 sensitivity) The interrupt sensitivity, defined using the IS2[1:0] bits, is applied to the following external interrupts: **External interrupt ei1 (port A3 and A5):** 00: External interrupt sensitivity = falling edge and low level 01: External interrupt sensitivity = rising edge only 10: External interrupt sensitivity = falling edge only 11: External interrupt sensitivity = rising and falling edge

Table 20. EICR register description (continued)

Bit	Name	Function
2:1	IS3[1:0]	Interrupt sensitivity (ei0 sensitivity) The interrupt sensitivity, defined using the IS2[1:0] bits, is applied to the following external interrupts: **External interrupt ei0 (port D[6:4]):** 00: External interrupt sensitivity = falling edge and low level (IPA bit = 0) and rising edge and high level (IPA bit = 1) 01: External interrupt sensitivity = rising edge only (IPA bit = 0) and falling edge only (IPA bit = 1) 10: external interrupt sensitivity = falling edge only (IPA bit = 0) and rising edge only (IPA bit = 1) 11: external interrupt sensitivity = rising and falling edge (IPA bit = 0 and IPA bit = 1) **External interrupt ei0 (port D[3:1]):** 00: External interrupt sensitivity = falling edge and low level 01: External interrupt sensitivity = rising edge only 10: External interrupt sensitivity = falling edge only 11: External interrupt sensitivity = 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).
0	IPA	Interrupt polarity (for port D) This bit is used to invert the sensitivity of 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

7.8 Nested interrupts register map and reset values

Table 21. Nested interrupts register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0024h	ISPR0	ei1		ei0		MCC + SI		MCES	
	Reset value	I1_3 1	I0_3 1	I1_2 1	I0_2 1	I1_1 1	I0_1 1	1	1
0025h	ISPR1	MTC C/D		MTC R/Z		MTC U/CL		ei2	
	Reset value	I1_7 1	I0_7 1	I1_6 1	I0_6 1	I1_5 1	I0_5 1	I1_4 1	I0_4 1
0026h	ISPR2	SCI		Timer B		Timer A		SPI	
	Reset value	I1_11 1	I0_11 1	I1_10 1	I0_10 1	I1_9 1	I0_9 1	I1_8 1	I0_8 1
0027h	ISPR3					PWMART		AVD	
	Reset value	I1_15 1	I0_15 1	I1_14 1	I0_14 1	I1_13 1	I0_13 1	I1_12 1	I0_12 1
0028h	EICR	IS11 0	IS10 0	IPB 0	IS21 0	IS20 0	IPA 0	0	0
	Reset value								

7.9 Interrupt addresses

Table 22. Interrupt mapping

No.	Source block	Description	Register label	Priority order	Exit from HALT(1)	Address vector
	Reset	Reset	N/A	Highest 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, 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	Lowest priority	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

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

8 Power saving modes

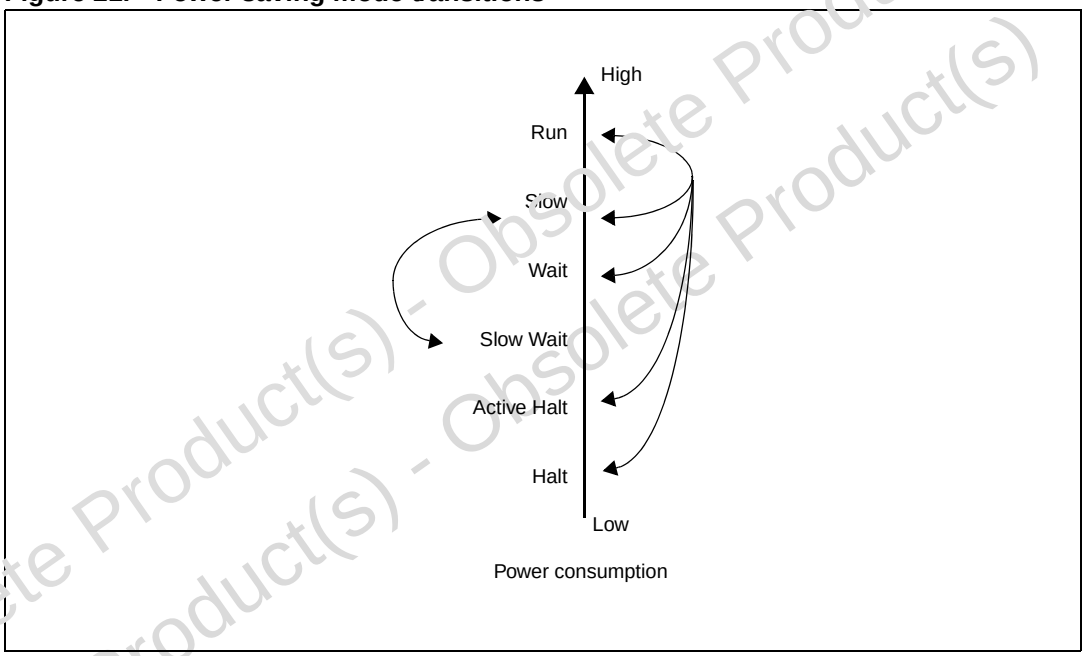
8.1 Introduction

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 21*): 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 21. Power saving mode transitions



8.2 Slow mode

This mode has two targets:

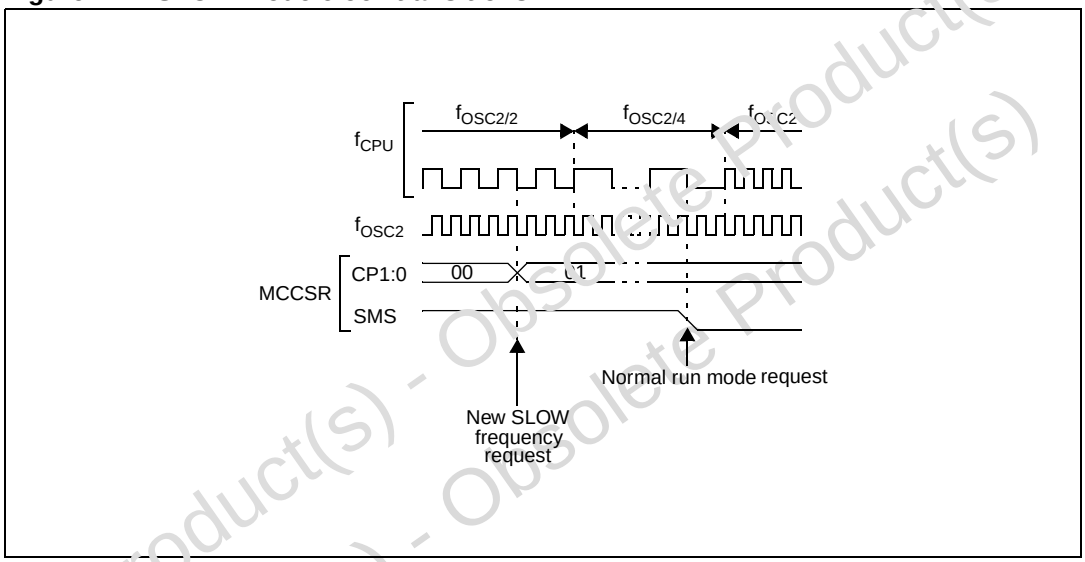
- To reduce power consumption by decreasing the internal clock in the device,
- To adapt the internal clock frequency (f_{CPU}) to the available supply voltage.

SLOW mode is controlled by three bits in the MCCSR register: the SMS bit enables or disables SLOW mode and two CPx bits 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 22. SLOW mode clock transitions



8.3 Wait mode

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

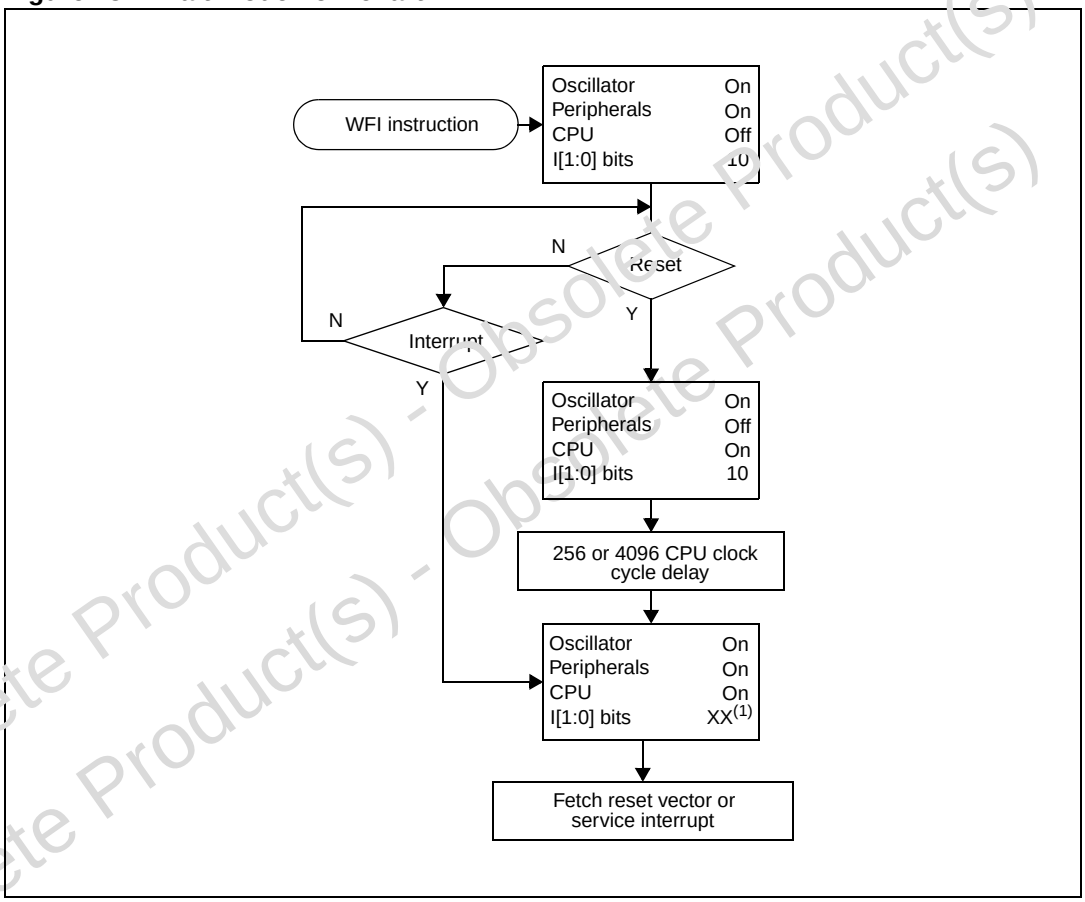
This power saving mode is selected by calling the 'WFI' instruction.

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 remains in Wait mode until a reset or an interrupt occurs, causing it to wake up.

Refer to *Figure 23*.

Figure 23. Wait mode flow-chart



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.

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).

Table 23. Active Halt and Halt power saving modes

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 6.6 on page 54* 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 22: Interrupt mapping on page 70*) 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 25*).

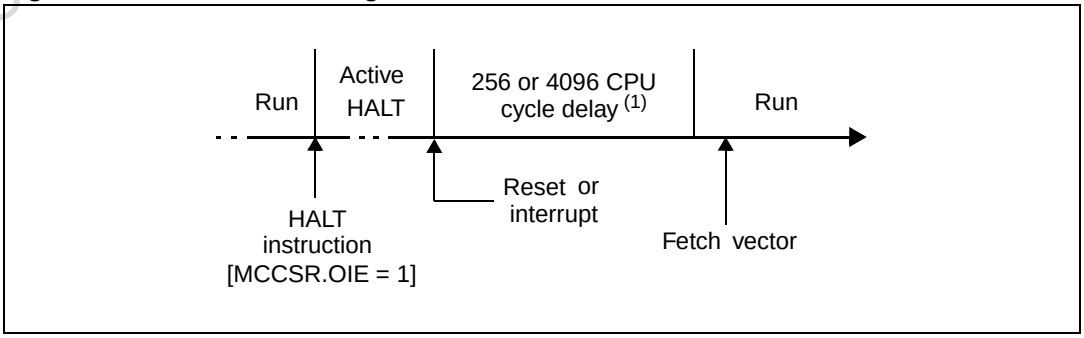
When entering Active Halt mode, the I[1:0] bits in the CC register are forced to '10b' to enable interrupts. Therefore, if an interrupt is pending, the MCU wakes up immediately.

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.

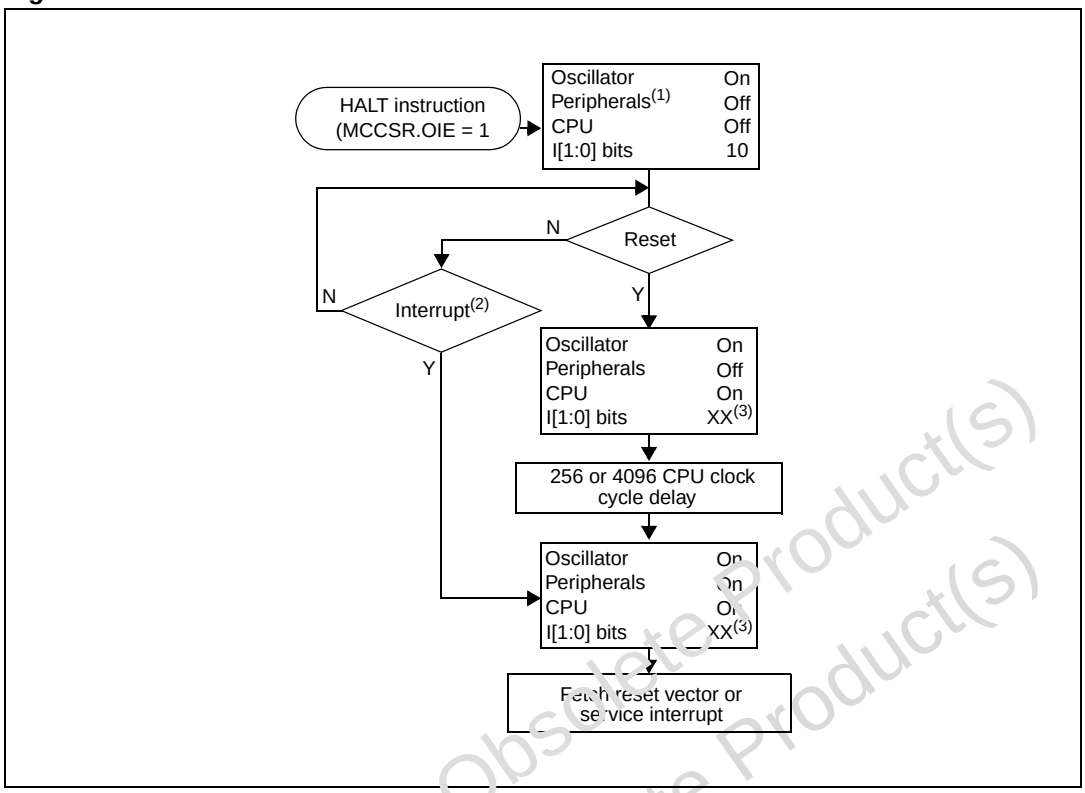
Note: As soon as the interrupt capability of one of the oscillators is selected (MCCSR.OIE bit set), entering Active Halt mode while the watchdog is active does not generate a reset. This means that the device cannot spend more than a defined delay in this power saving mode.

Figure 24. Active Halt timing overview



1. This delay occurs only if the MCU exits Active Halt mode by means of a reset.

Figure 25. TActive HALT mode flow-chart



1. Peripherals clocked with an external clock source can still be active.
2. Only the MCC/RTC interrupt and some specific interrupts can exit the MCU from Active Halt mode (such as external interrupt). Refer to Table 22, *Interrupt mapping on page 70* for more details.
3. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits of the CC register are set to the current software priority level of the interrupt routine and restored when the CC register is popped.

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 6.6 on page 54* for more details on the MCCSR register).

The MCU can exit Halt mode on reception of either a specific interrupt (see *Table 22: Interrupt mapping on page 70*) 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 27*).

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 356* for more details).

Figure 26. HALT timing overview

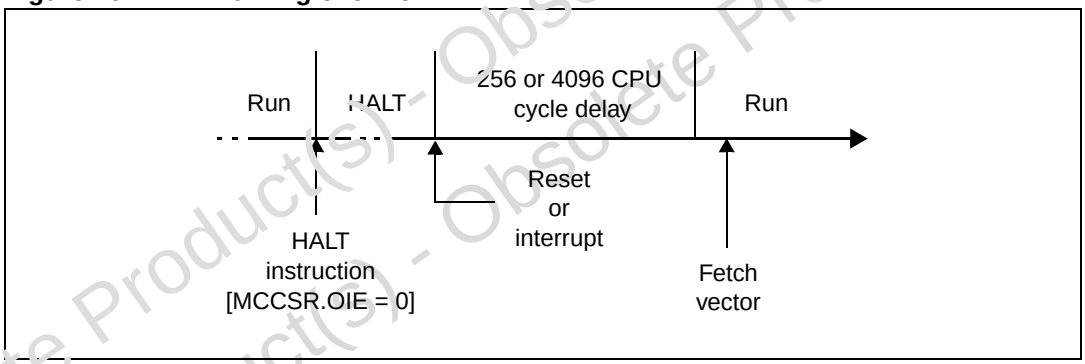
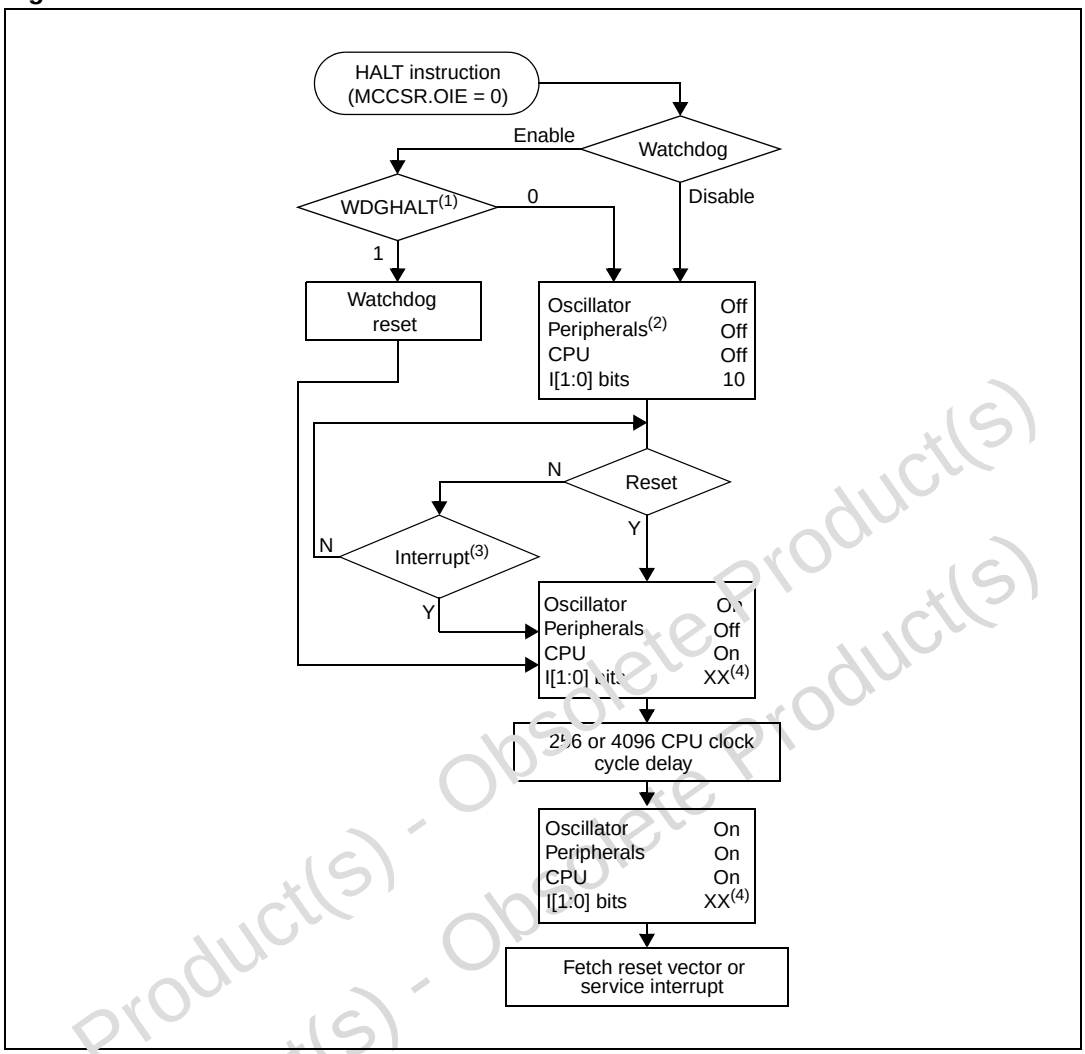


Figure 27. Halt mode flowchart



1. WDGHALT is an option bit. See option byte section for more details.
2. Peripherals 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 22: Interrupt mapping on page 70* 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 eight 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 *Section 9.3: I/O port implementation*). The generic I/O block diagram is shown in *Figure 28*.

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.

Note: 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 *Section 2: Pin description* and *Section 7: Interrupts*). 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.

Table 24. DR register value and output pin status

DR	Push-pull	Open-drain
0	V_{SS}	V_{SS}
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.

Figure 28. I/O port general block diagram

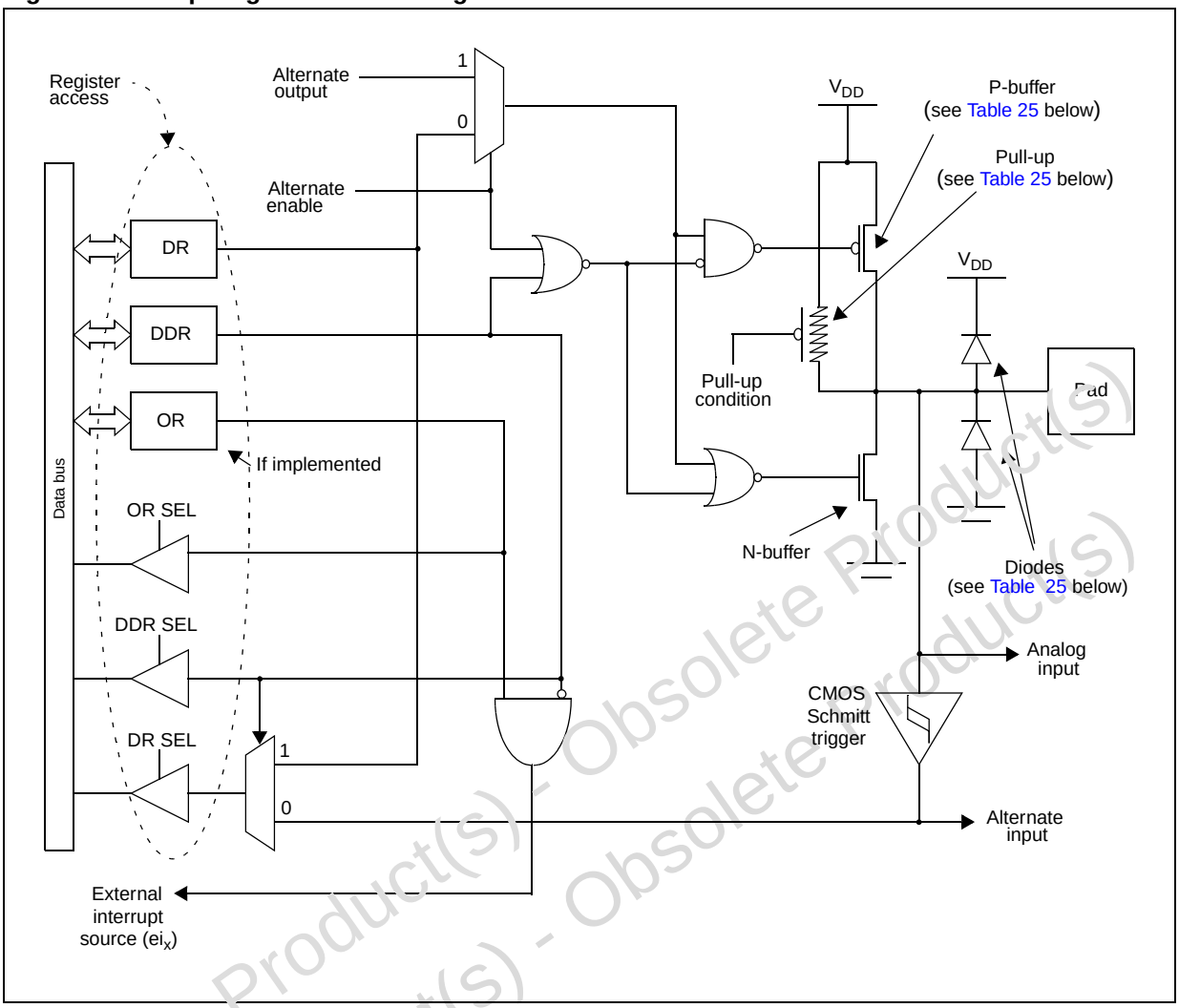
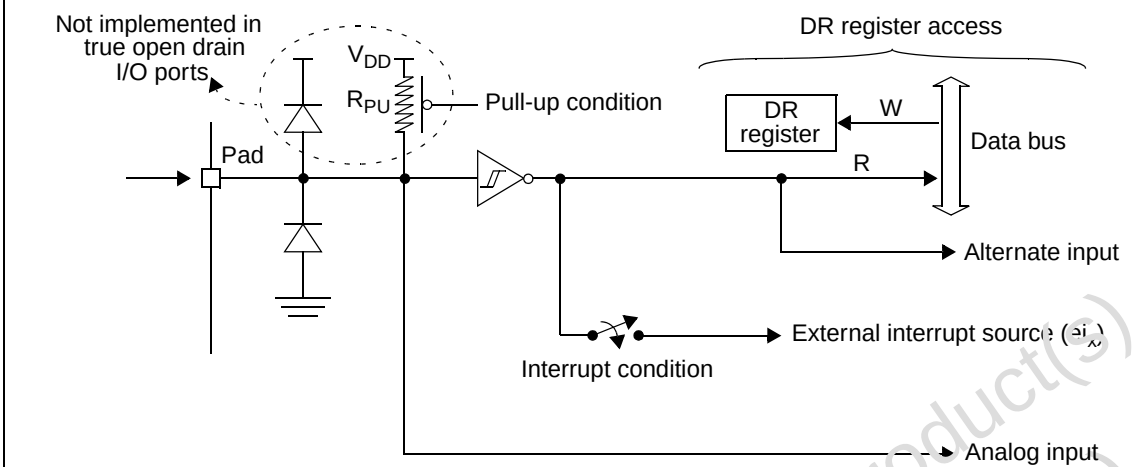
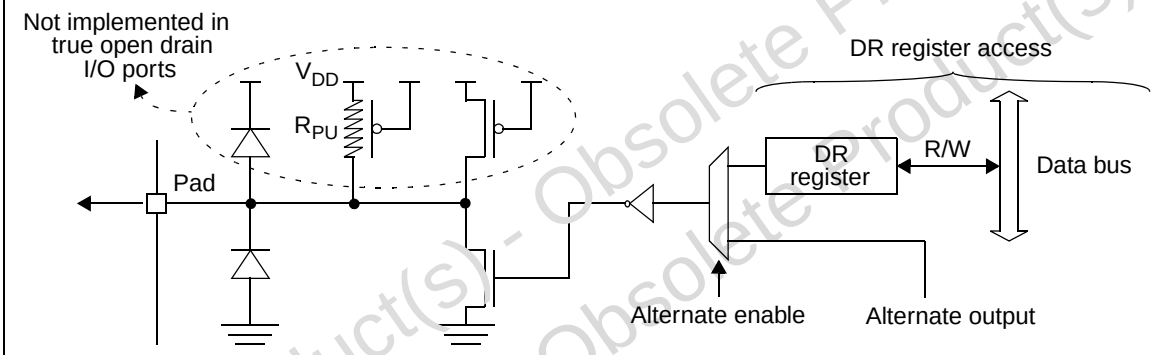
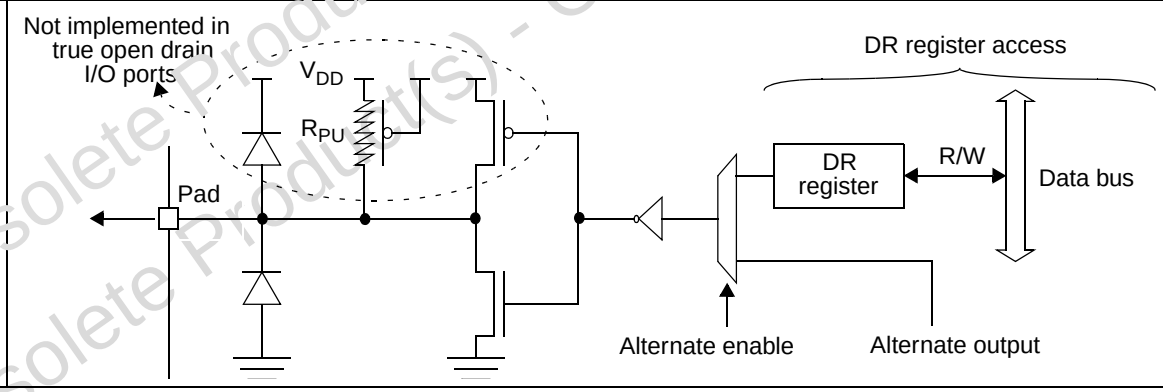


Table 25. I/O port mode options

Configuration mode		Pull-up	P-buffer	Diodes(1)	
				to V_{DD}	to V_{SS}
Input	Floating with/without interrupt	Off(2)	Off	On	On
	Pull-up with/without interrupt	On(3)			
Output	Push-pull	Off(2)	On		
	Open drain (logic level)		Off		
	True open drain	NI(4)	NI(4)	NI(4)	

1. 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.
2. Implemented not activated.
3. Implemented and activated.
4. Not implemented.

Table 26. I/O port configurations

	Hardware configuration
Input(1)	
Open drain output(2)	
Push-pull output(2)	

1. When the I/O port is in input configuration and the associated alternate function is enabled as an output, reading the DR register reads 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.

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.

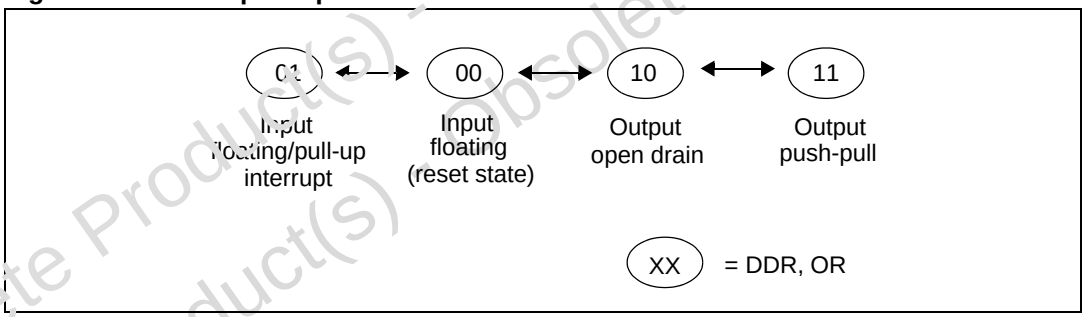
Warning: The analog input voltage level must be within the limits stated in the absolute maximum ratings.

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 29*. Other transitions are potentially risky and should be avoided, since they are likely to present unwanted side-effects such as spurious interrupt generation.

Figure 29. Interrupt I/O port state transitions



9.4 Low power modes

Table 27. Effect of low power modes on I/O ports

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).

Table 28. I/O port interrupt control/wake-up capability

Interrupt event	Event flag	Enable control bit	Exit from WAIT	Exit from HALT
External interrupt on selected external event	-	DDRx, ORx	Yes	

9.5.1 I/O port implementation

The I/O port register configurations are summarized below.

Standard ports

Table 29. 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

Table 30. Interrupt ports with pull-up: PA6, PA3, PB6, PC3, PC1, PD5, PD4, PD2

Mode	DDR	OR
Floating input	0	0
Pull-up interrupt input	0	1
Open drain output	1	0
Push-pull output	1	1

Table 31. Interrupt ports without pull-up: PA7, PA5, PB7, PC2, PC0, PD6, PD3, PD

Mode	DDR	OR
Floating input	0	0
Floating interrupt input	0	1
Open drain output	1	0
Push-pull output	1	1

Table 32. 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		Pull-up interrupt	Open drain	Push-pull
	PA2:0		Pull-up	Open drain	Push-pull
Port B	PB7		Floating interrupt	Open drain	Push-pull
	PB6		Pull-up interrupt	Open drain	Push-pull
	PB5:0		Pull-up	Open drain	Push-pull
Port C	PC7:4		Pull-up	Open drain	Push-pull
	PC3, PC1		Pull-up interrupt	Open drain	Push-pull
	PC2, PC0		Floating interrupt	Open drain	Push-pull
Port D	PD7, PD0		Pull-up	Open drain	Push-pull
	PD6, PD3, PD1		Floating interrupt	Open drain	Push-pull
	PD5, PD4, PD2		Pull-up interrupt	Open drain	Push-pull
Port E	PE5:0		Pull-up	Open drain	Push-pull
Port F	PF5:0		Pull-up	Open drain	Push-pull
Port G	PG7:0		Pull-up	Open drain	Push-pull
Port H	PH7:0		Pull-up	Open drain	Push-pull

9.6 I/O port register map and reset values

Table 33. 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

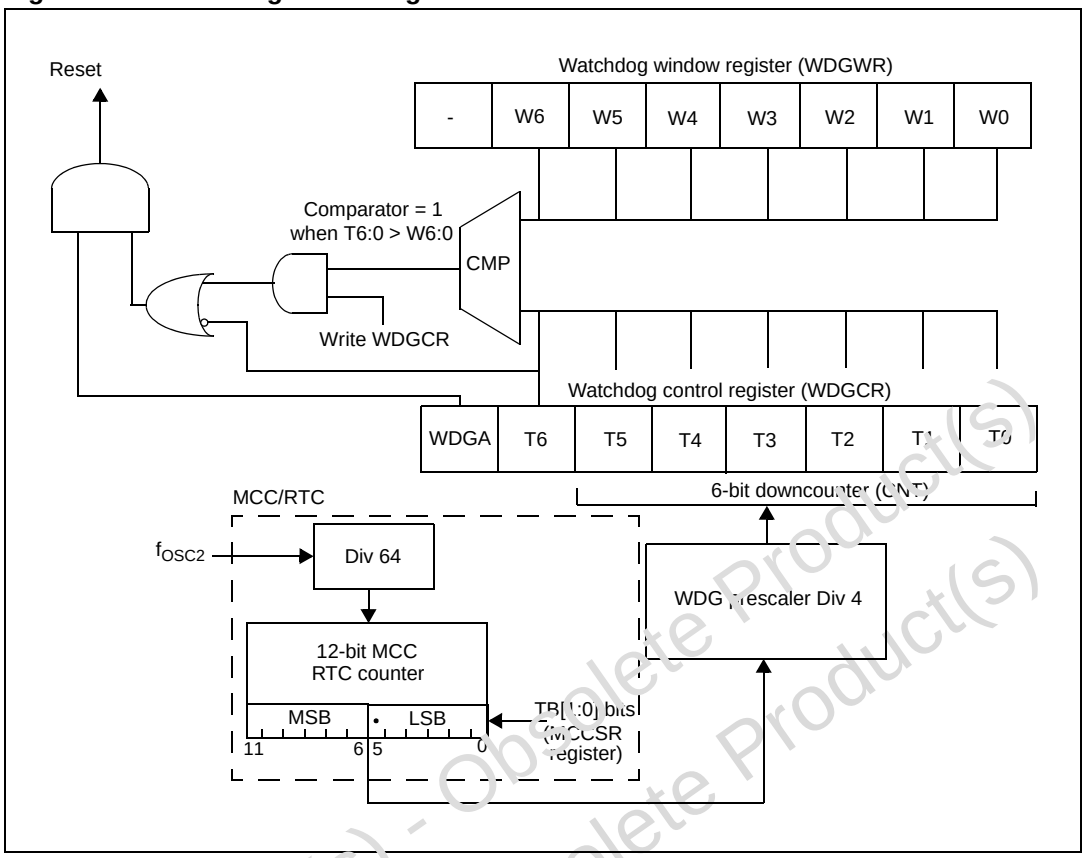
- Programmable free-running downcounter
- Conditional reset
  - reset (if watchdog activated) when the downcounter value becomes less than 40h
  - reset (if watchdog activated) if the downcounter is reloaded outside the window (see *Figure 33*)
- Hardware/software watchdog activation (selectable by option 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 30. Watchdog block diagram



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 3Fh and C0h (see *Figure 31*).

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 and 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 represent the time delay before the watchdog produces a reset (see *Figure 31: 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 32*).

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 33* 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 generates a reset.

10.1.4 Using Halt mode with the watchdog

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.

10.1.5 How to program the watchdog timeout

Figure 31 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 32*.

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

Figure 31. Approximate timeout duration

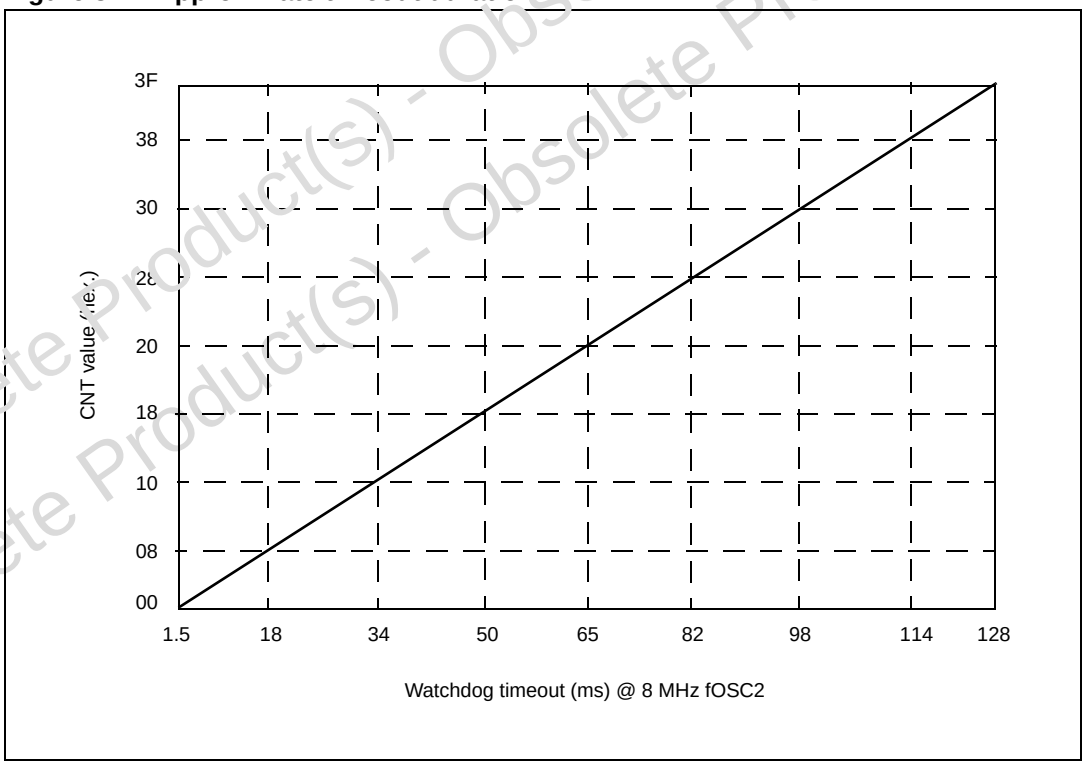


Figure 32. Exact timeout duration (t_{min} and t_{max})

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

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

To calculate the maximum watchdog timeout (t_{max}):

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

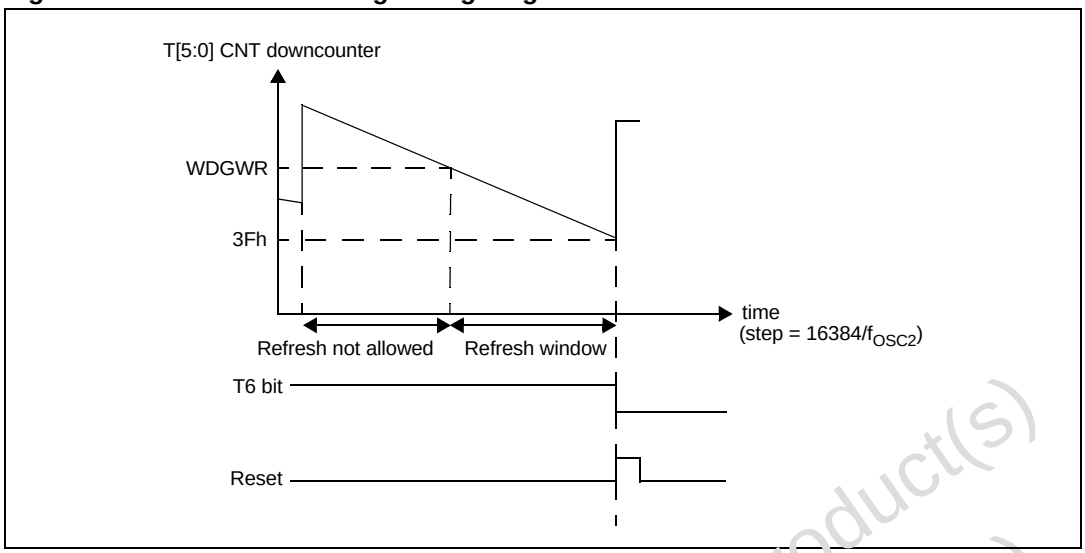
ELSE $t_{max} = t_{max0} + \left[16384 \times \left(CNT - \left[\frac{4CNT}{MSB}\right]\right) + (192 + LSB) \times 64 \times \left[\frac{4CNT}{MSB}\right]\right] \times t_{osc2}$

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

Figure 33. Window watchdog timing diagram



10.1.6 Low power modes

Table 34. Effect of low power modes on window watchdog

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 *Table 22: Interrupt 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 stops 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 *Section 14.1: Flash option bytes*.

10.1.8 Using Halt mode with the watchdog (WDGHALT option)

If Halt mode is used when the watchdog is enabled, refresh the WDG counter before executing the HALT instruction to avoid an unexpected WDG reset immediately after waking up the microcontroller.

10.1.9 Watchdog interrupts

None.

10.1.10 Watchdog control register (WDGCR)

WDGCR Reset value: 0111 1111 (7Fh)

7	6	5	4	3	2	1	0
WDGA	T[6:0]						
R/W	R/W						

Table 35. WDGCR register description

Bit	Name	Function
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.*
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).

10.1.11 Watchdog window register (WDGWR)

WDGWR Reset value: 0111 1111 (7Fh)

7	6	5	4	3	2	1	0
Reserved	W[6:0]						
-	R/W						

Table 36. WDGWR register description

Bit	Name	Function
7	-	Reserved, must be kept cleared
6:0	W[6:0]	7-bit window value These bits contain the window value to be compared to the downcounter.

10.1.12 Watchdog timer register map and reset values

Table 37. Watchdog timer register map and reset values

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

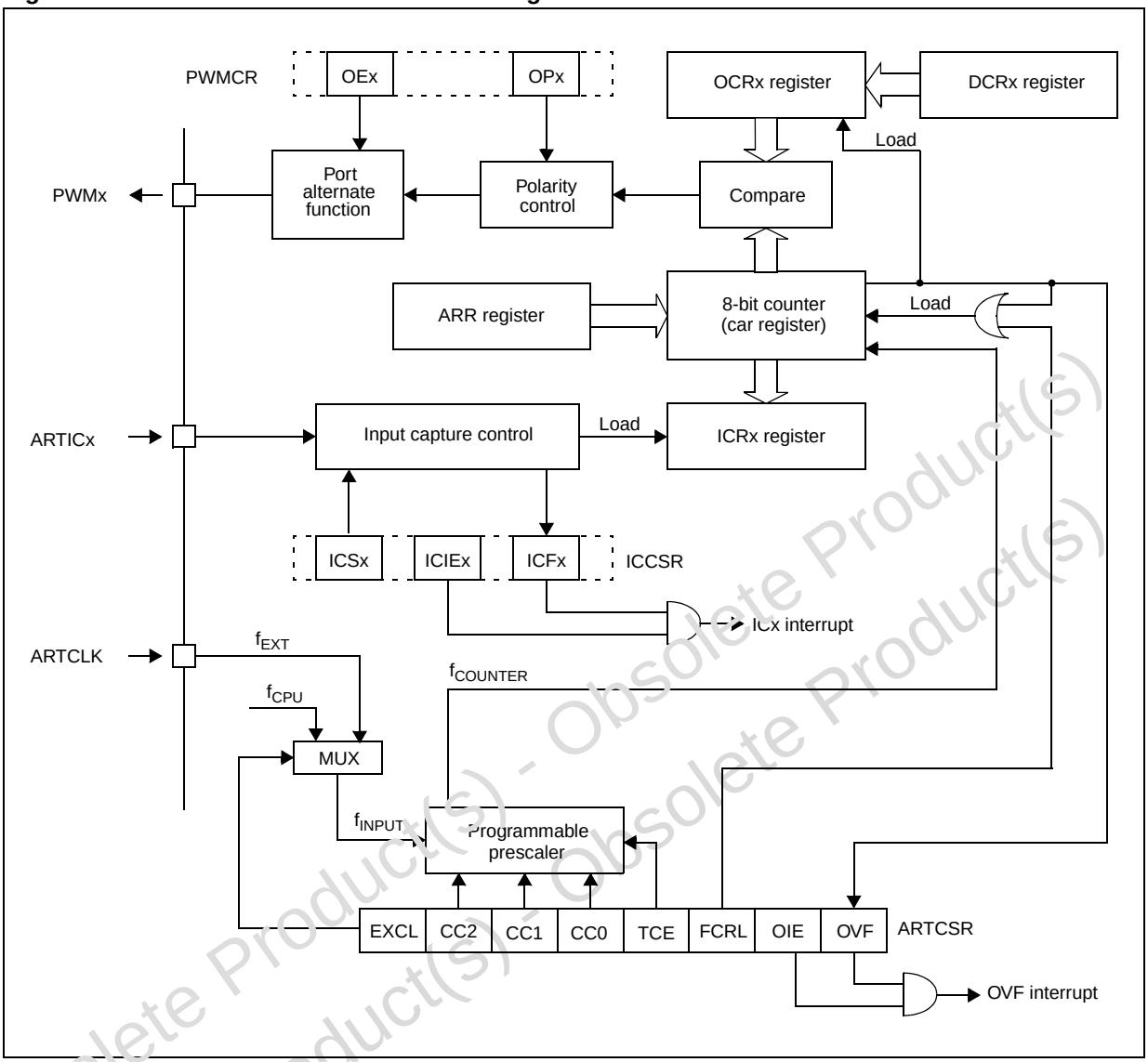
These resources allow five possible operating modes:

- Generation of up to 4 independent PWM signals
- Output compare and time base interrupt
- Up to 2 input capture functions
- External event detector
- Up to 2 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 34. PWM auto-reload timer block diagram



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 auto-reload register (ARTARR). 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 eight available taps of the prescaler, as defined by CC[2:0] bits in the 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 starts from a known value.

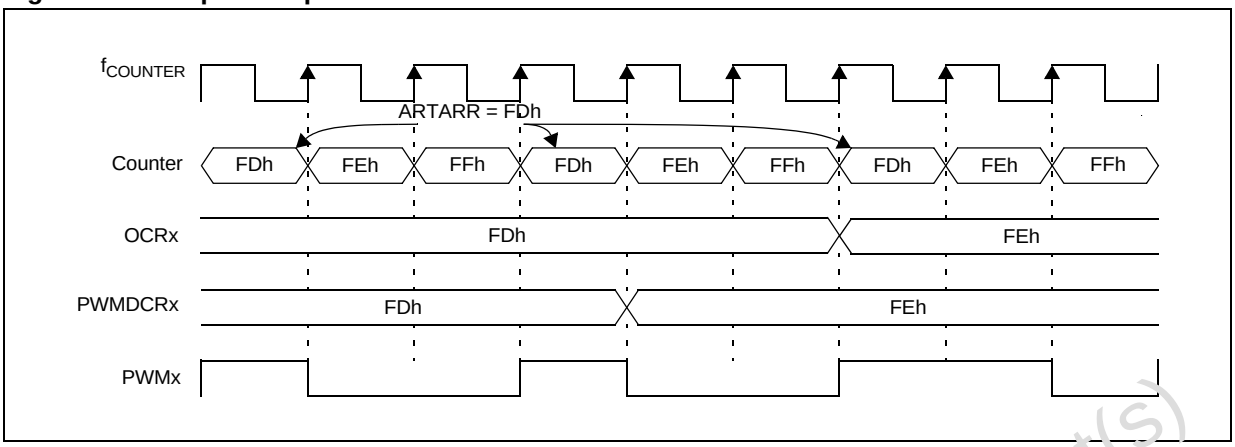
Direct access to the prescaler is not possible.

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 35. Output compare control



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 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 36. PWM auto-reload timer function

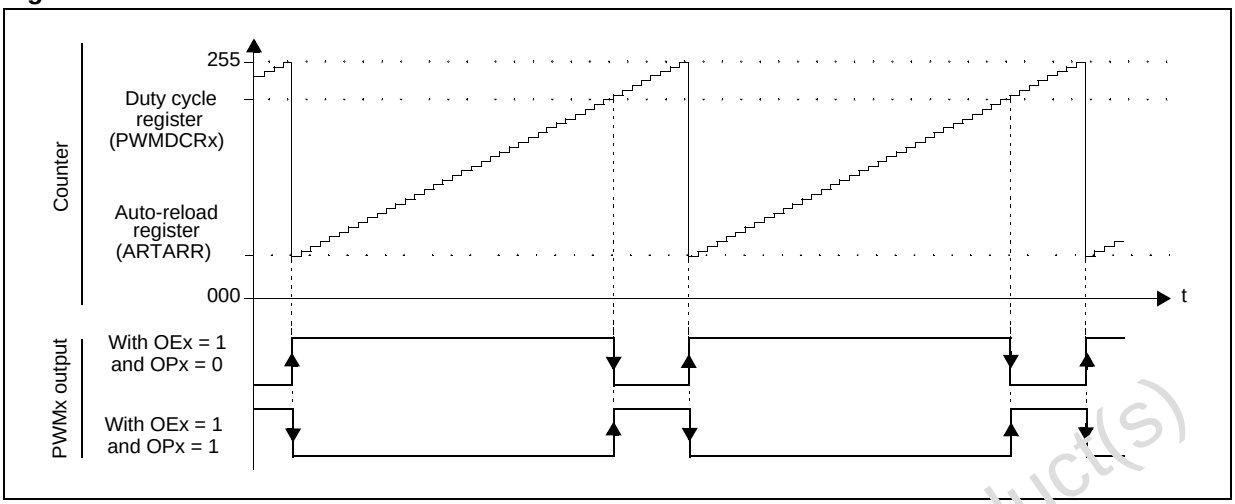
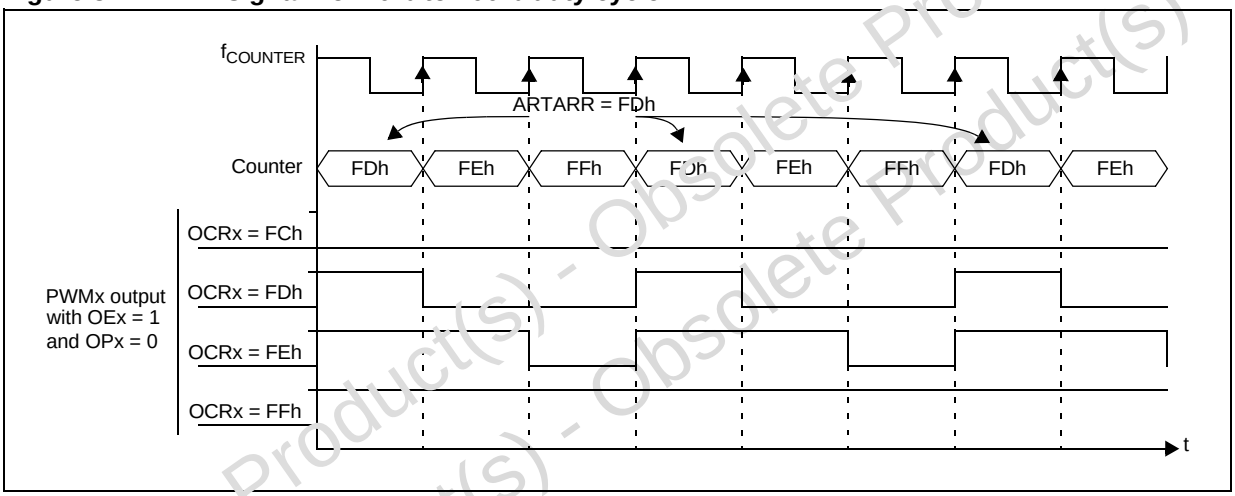


Figure 37. PWM signal from 0% to 100% duty cycle



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.

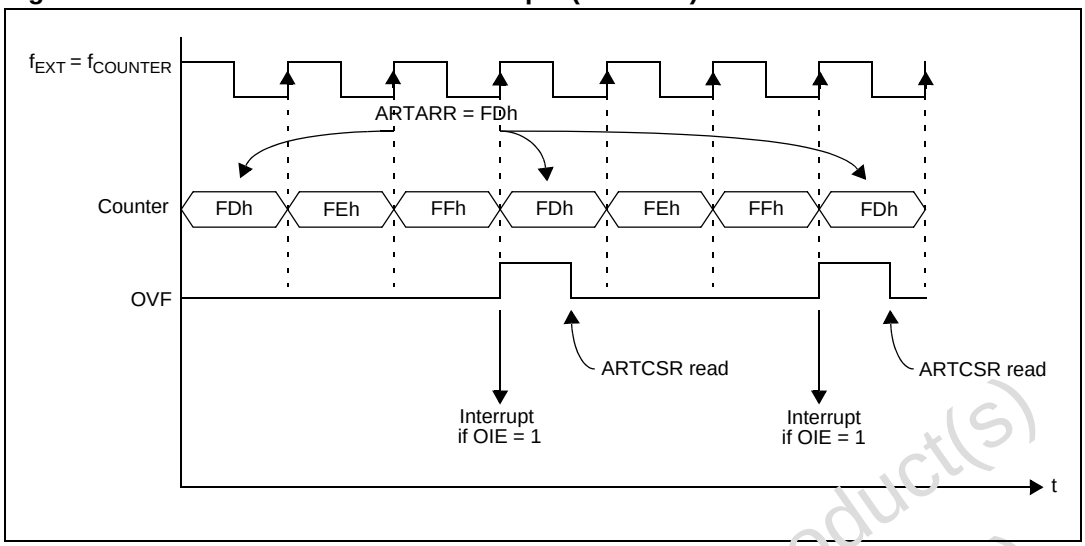
External clock and event detector mode

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 38. External event detector example (3 counts)



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.

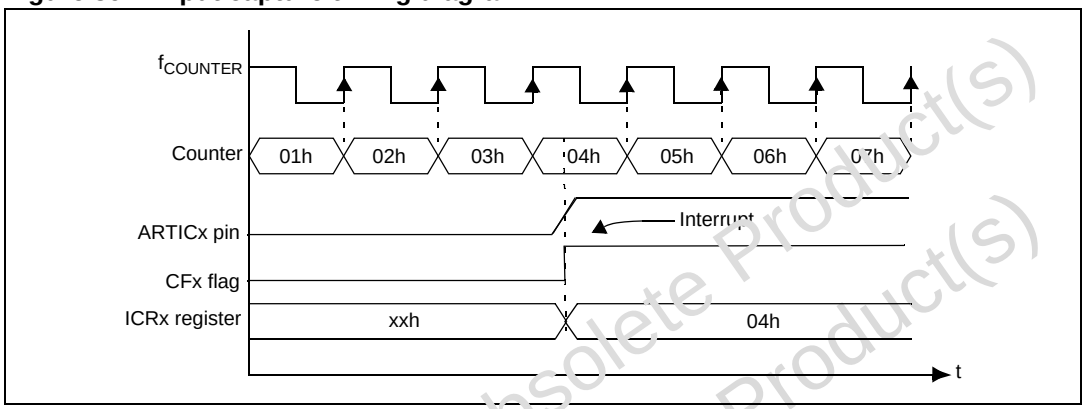
External interrupt capability

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 39. Input capture timing diagram



10.2.3 PWM ART registers

ART control/status register (ARTCSR)

ARTCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
EXCL	CC[2:0]			TCE	FCRL	OIE	OVF
R/W	R/W			R/W	RO	R/W	R/W

Table 38. ARTCSR register description

Bit	Name	Function
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
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}: 000: Prescaler division ratio from f_{INPUT} = f_{INPUT} ($f_{COUNTER}$) and 8 MHz (with f_{INPUT} = 8 MHz) 001: Prescaler division ratio from f_{INPUT} = f_{INPUT}/2 ($f_{COUNTER}$) and 4 MHz (with f_{INPUT} = 8 MHz) 010: Prescaler division ratio from f_{INPUT} = f_{INPUT}/4 ($f_{COUNTER}$) and 2 MHz (with f_{INPUT} = 8 MHz) 011: Prescaler division ratio from f_{INPUT} = f_{INPUT}/8 ($f_{COUNTER}$) and 1 MHz (with f_{INPUT} = 8 MHz) 100: Prescaler division ratio from f_{INPUT} = f_{INPUT}/16 ($f_{COUNTER}$) and 500 kHz (with f_{INPUT} = 8 MHz) 101: Prescaler division ratio from f_{INPUT} = f_{INPUT}/32 ($f_{COUNTER}$) and 250 kHz (with f_{INPUT} = 8 MHz) 110: Prescaler division ratio from f_{INPUT} = f_{INPUT}/64 ($f_{COUNTER}$) and 125 kHz (with f_{INPUT} = 8 MHz) 111: Prescaler division ratio from f_{INPUT} = f_{INPUT}/128 ($f_{COUNTER}$) and 62.5 kHz (with f_{INPUT} = 8 MHz)
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
2	FCRL	Force counter reload This bit is write-only and any attempt to read it yields 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.
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 (bit 0) is set. 0: Overflow interrupt disable 1: Overflow interrupt enable
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

ART counter access register (ARTCAR)

ARTCAR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CA[7:0]							
R/W							

Table 39. ARTCAR register description

Bit	Name	Function
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)

ARTARR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
AR[7:0]							
R/W							

Table 40. ARTARR register description

Bit	Name	Function
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 See below for PWM frequency versus resolution: f_{PWM} min/max (~0.244 kHz/31.25 kHz) = resolution (8-bit): ARTARR value 0 f_{PWM} min/max (~0.244 kHz/62.5 kHz) = resolution (> 7-bit): ARTARR value 0..127 f_{PWM} min/max (~0.488 kHz/125 kHz) = resolution (> 6-bit): ARTARR value 128..191 f_{PWM} min/max (~0.977 kHz/250 kHz) = resolution (> 5-bit): ARTARR value 192..223 f_{PWM} min/max (~1.953 kHz/500 kHz) = resolution (> 4-bit): ARTARR value 224..239

PWM control register (PWMCR)

PWMCR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
OE[3:0]				OP[3:0]			
R/W				R/W			

Table 41. PWMCR register description

Bit	Name	Function
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
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 (see *Table 42*).

Table 42. PWM output signal polarity selection

PWMx output level		OPx(1)
Counter ≤ OCRx	Counter > OCRx	
1	0	0
0	1	1

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

PWM duty cycle registers (PWMDCRx)

PWMDCRx Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DC[7:0]							
R/W							

Table 43. PWMDCRx register description

Bit	Name	Function
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.

ART input capture control/status register (ARTICCSR)

ARTICCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved		CS[2:1]		CIE[2:1]		CF[2:1]	
-		R/W		R/W		R/W	

Table 44. ARTICCSR register description

Bit	Name	Function
7:6	-	Reserved, must be kept cleared.
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
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
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

ART input capture registers (ARTICRx)

ARTICRx Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
IC[7:0]							
RO							

Table 45. ARTICRx register description

Bit	Name	Function
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 register map and reset values

Table 46. PWM auto-reload timer register map and reset values

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

Note: Some timer pins may not be available (not bonded) in some devices. Refer to Table 2: Device pin description on page 23.

The block diagram is shown in *Figure 40*.

When reading an input signal on a non-bonded pin, the value is always ‘1’.

10.3.3 Functional description

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 and low.

Counter register (CR)

- Counter high register (CHR) is the most significant byte (MSB).
- Counter low register (CLR) is the least significant byte (LSB).

Alternate counter register (ACR)

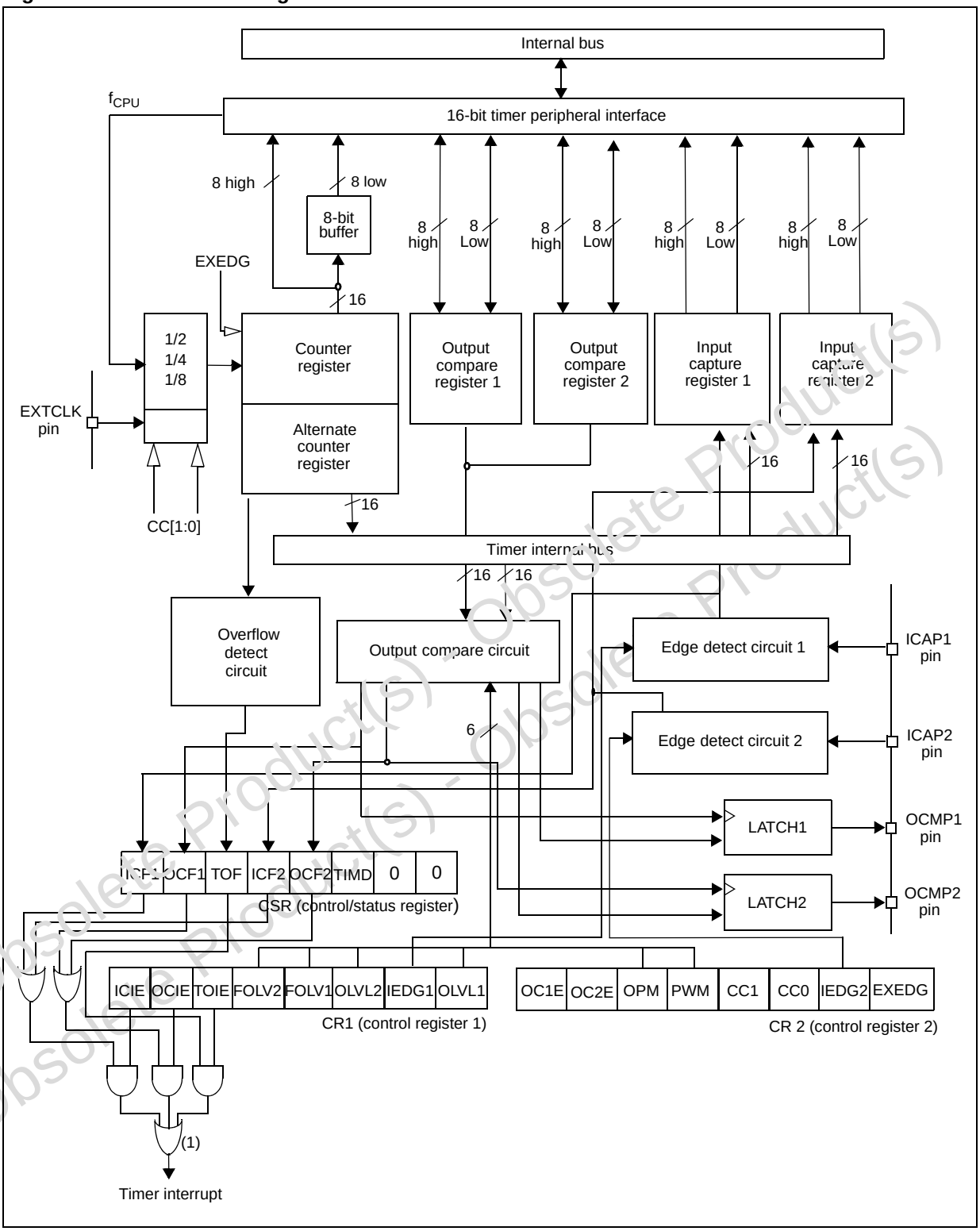
- Alternate counter high register (ACHR) is the MSB.
- Alternate counter low register (ACLR) is the LSB.

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 *16-bit read sequence (from either the counter register or the alternate counter register) on page 107*).

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 51*. 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.

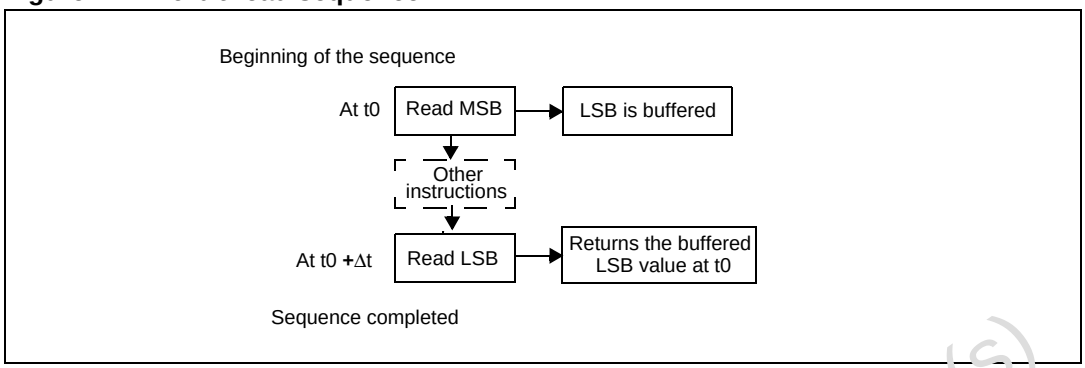
Figure 40. Timer block diagram



1. If IC, OC and TO interrupt requests have separate vectors then the last OR is not present (see *Table 22: Interrupt mapping*)

16-bit read sequence (from either the counter register or the alternate counter register)

Figure 41. 16-bit read sequence



The user must read the MSB first, then the LSB value is buffered automatically.

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

After a complete reading sequence, if only the CLR register or ACLR register are read, they return the LSB 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.

Note: 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).

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 triggers the free running counter.

The counter is synchronized 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.

Figure 42. Counter timing diagram, internal clock divided by 2

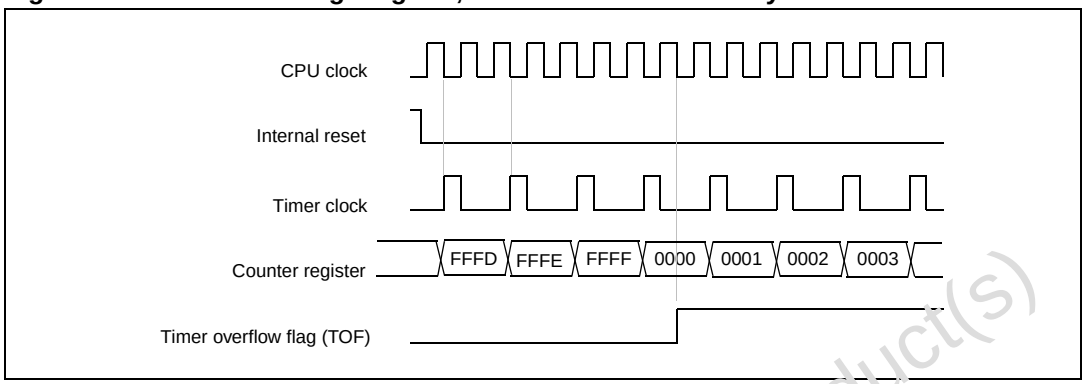


Figure 43. Counter timing diagram, internal clock divided by 4

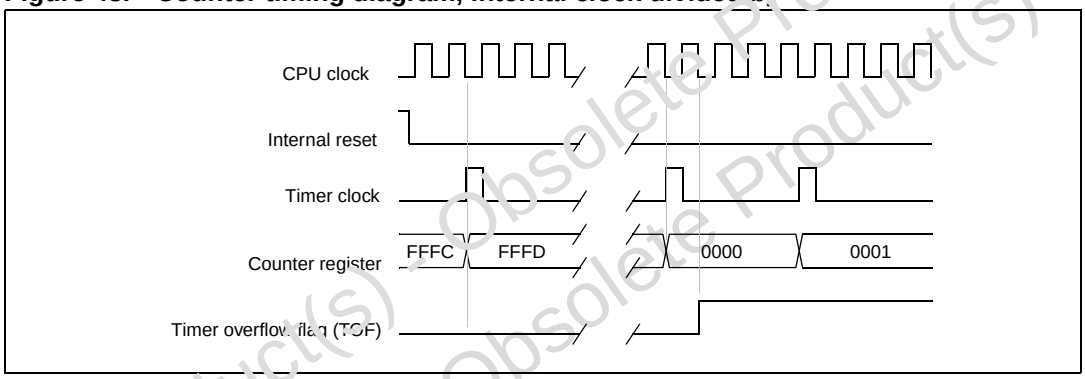
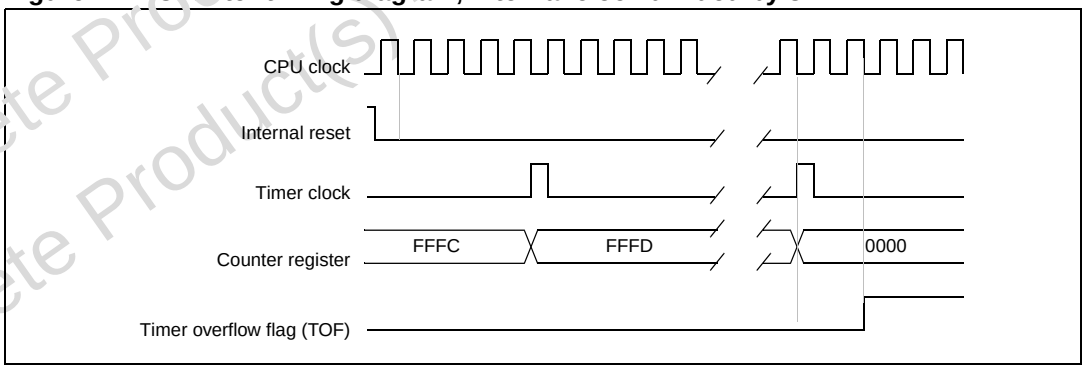


Figure 44. Counter timing diagram, internal clock divided by 8



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

Input capture

In this section, the index, *i*, may be 1 or 2 because there are two 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 below).

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

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:

- The 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 46*).
- 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 (that is, clearing the ICF*i* bit) is done in two steps:

1. By reading the SR register while the ICF*i* bit is set.
2. By accessing (reading or writing) the IC*i*LR register.

Note: 1 After reading the ICiHR register, transfer of input capture data is inhibited and ICFi is never set until the ICiLR register is also read.

2 The ICiR register contains the free running counter value which corresponds to the most recent input capture.

3 The two input capture functions can be used together even if the timer also uses the two output compare functions.

4 In One Pulse mode and PWM mode only the input capture 2 can be used.

5 The alternate inputs (ICAP1 and ICAP2) are always directly connected to the timer. So any transitions on these pins activate the input capture function. Moreover if one of the ICAPi 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 ICiHR (see note 1).

6 The TOF bit can be used with interrupt in order to measure event that go beyond the timer range (FFFFh).

Figure 45. Input capture block diagram

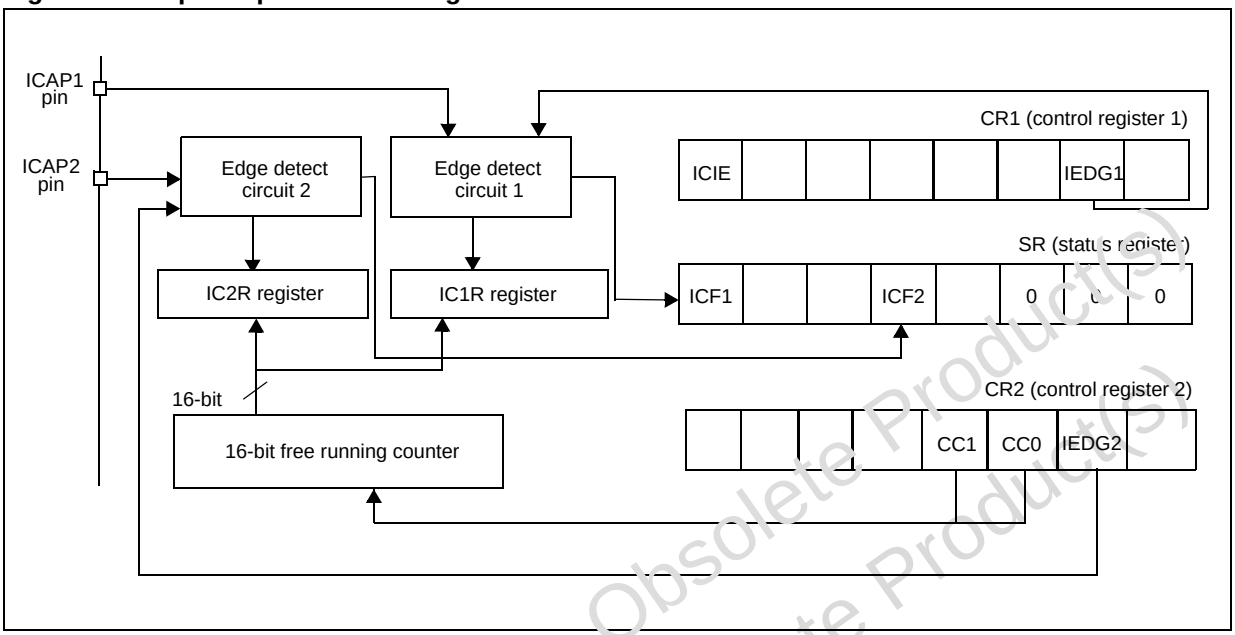
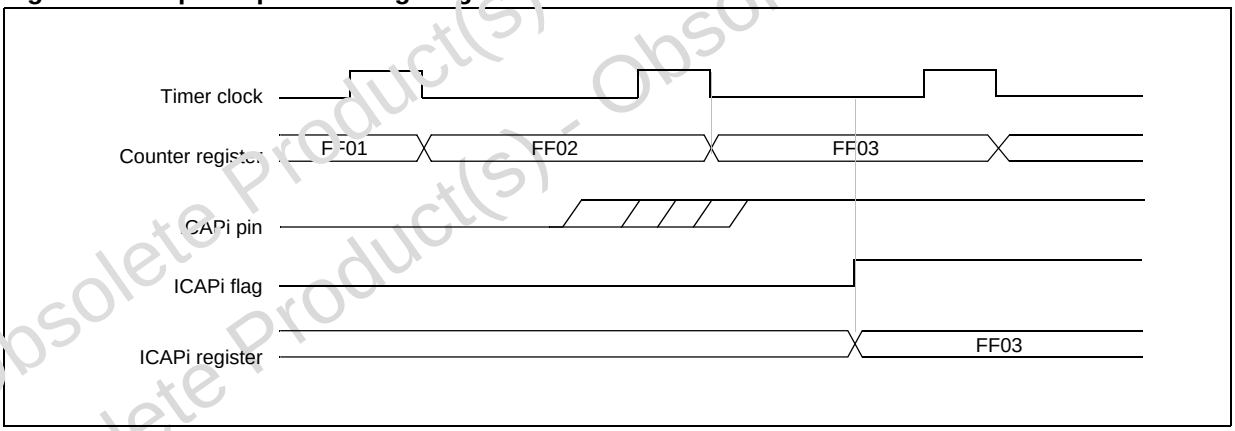


Figure 46. Input capture timing diagram



1. The active edge is the rising edge.
2. 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.

Output compare

In this section, the index, *i*, may be 1 or 2 because there are two 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.

	MSB	LSB
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 51*).

In the CR1 register select the following:

- Select the OLVL*i* bit to be 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:

- Set the OCF*i* bit.
- 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:

Equation 1

$$\Delta OCiR = \frac{\Delta t * f_{CPU}}{PRESC}$$

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 51*)

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

Equation 2

$$\Delta OCiR = \Delta t * f_{EXT}$$

Where:

Δt = output compare period (in seconds)

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

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

1. Reading the SR register while the OCF*i* bit is set.
2. Accessing (reading or writing) 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 time it is written to the OC*i*R register:

- Write to the OC*i*HR register (further compares are inhibited).
- Read the SR register (first step in 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).

Note: 1 After a processor write cycle to the OCiHR register, the output compare function is inhibited until the OCiLR register is also written.

2 If the OCiE bit is not set, the OCMPi pin is a general I/O port and the OLVLi bit does 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 48 for an example with $f_{CPU}/2$ and Figure 49 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 OCMPi pins even if the input capture mode is also used.

5 The value in the 16-bit OCiR register and the OLVi 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 47. Output compare block diagram

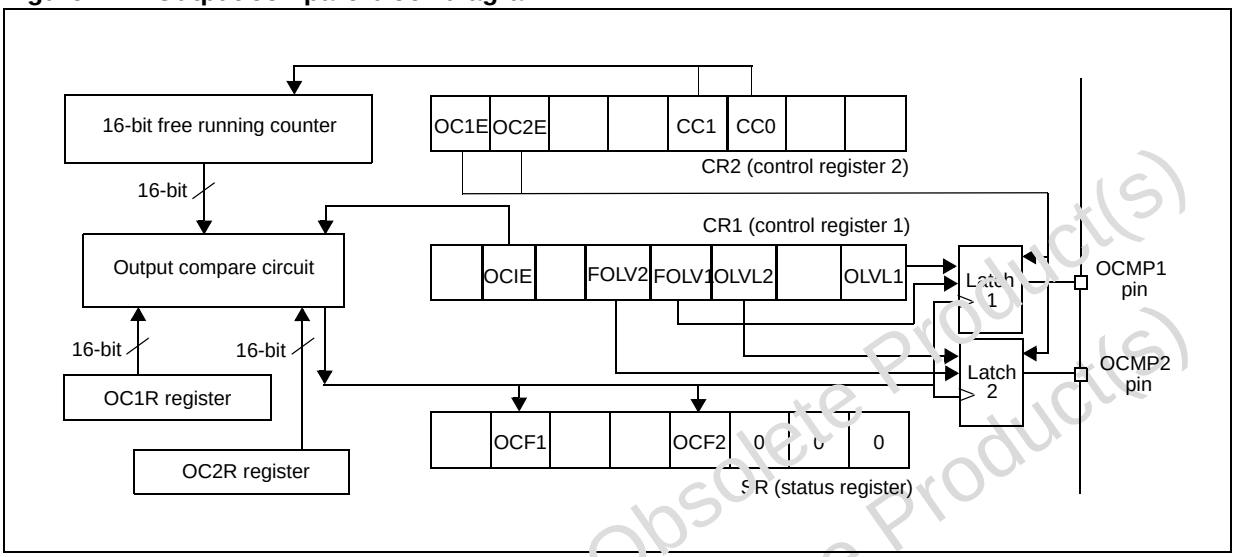


Figure 48. Output compare timing diagram, $f_{TIMER} = f_{CPU}/2$

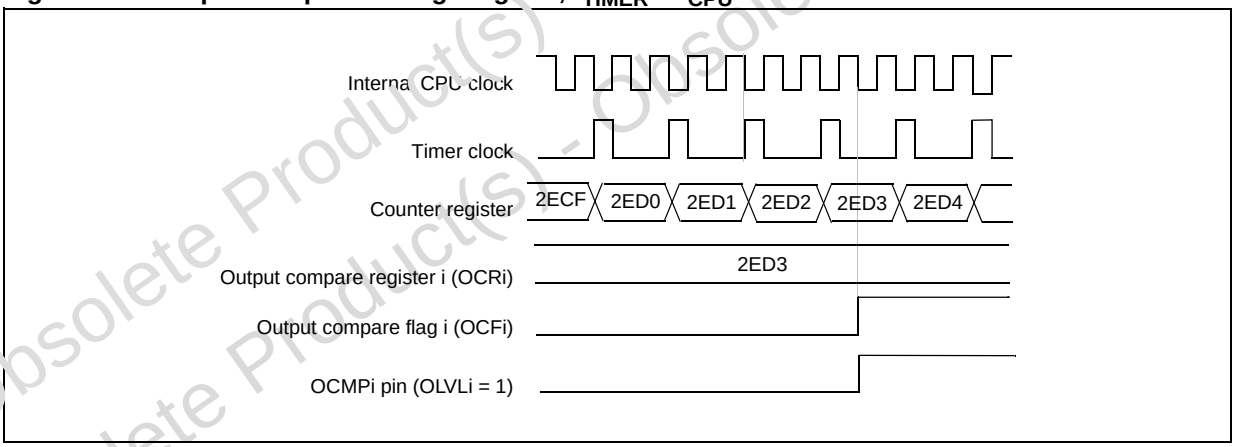
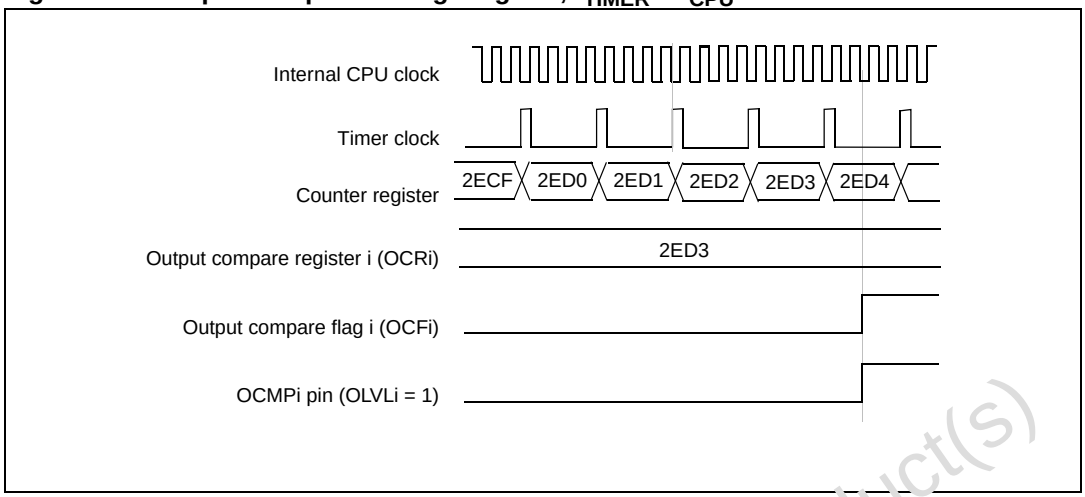


Figure 49. Output compare timing diagram, $f_{TIMER} = f_{CPU}/4$



One pulse mode

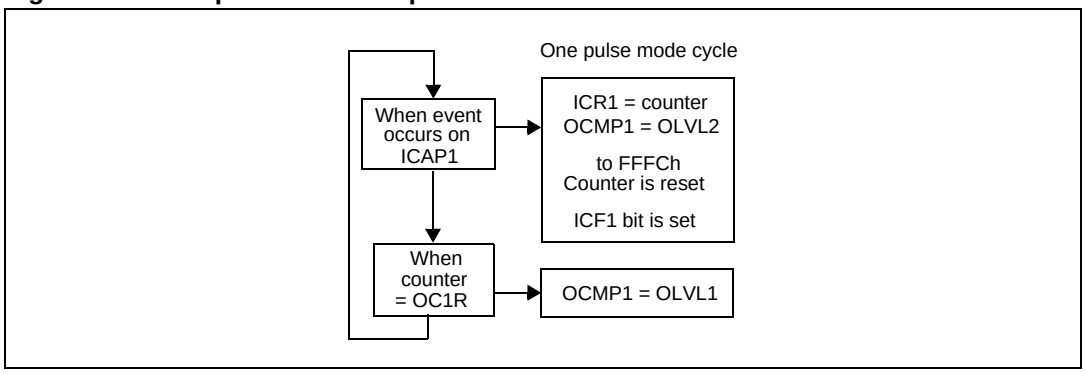
One pulse mode enables the generation of a pulse when an external event occurs. This mode is selected via the OPM bit in the CR2 register.

The one pulse mode uses the input capture1 function and the output compare1 function.

Procedure

1. Load the OC1R register with the value corresponding to the length of the pulse (see *Equation 3* below).
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 51*).

Figure 50. One pulse mode sequence



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 (that is, clearing the ICF*i* bit) is done in two steps:

1. Reading the SR register while the ICF*i* bit is set.
2. Accessing (reading or writing) the IC*i*LR register.

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

Equation 3

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

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 51*)

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

Equation 4

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

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 51*).

Note: 1 The OCF1 bit cannot be set by hardware in one pulse mode but the OCF2 bit can generate an output compare interrupt.

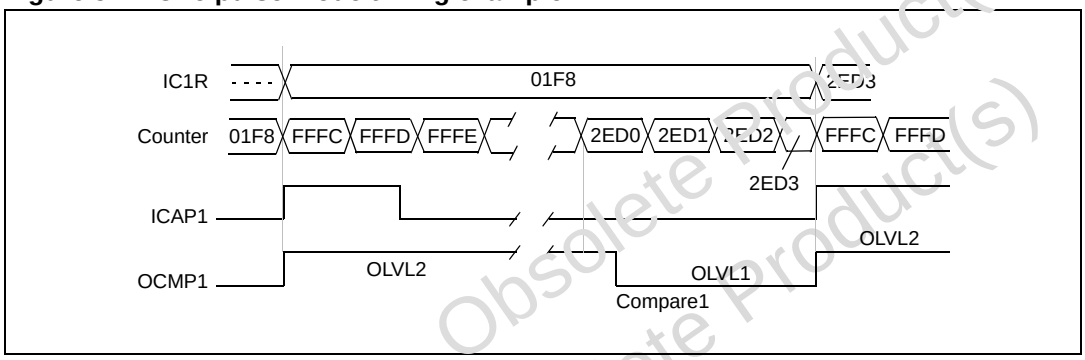
2 When the pulse width modulation (PWM) and one pulse mode (OPM) bits are both set, the PWM mode is the only active one.

3 If OLVL1 = OLVL2 a continuous signal is 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.

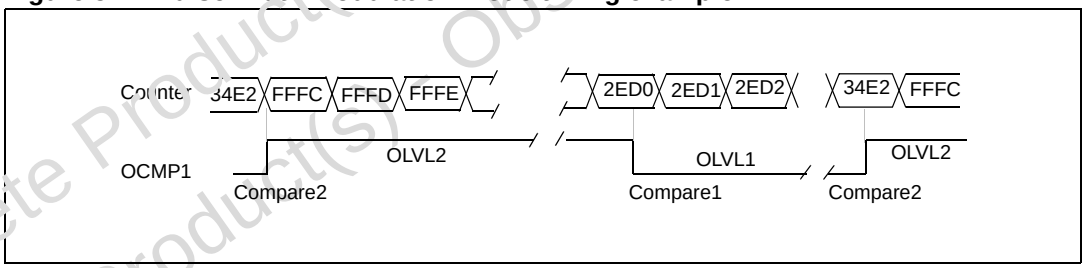
5 When one pulse mode is used OC1R is dedicated to this mode. Nevertheless OC2R and OCF2 can be used to indicate a period of time has been elapsed but cannot generate an output waveform because the level OLVL2 is dedicated to the one pulse mode.

Figure 51. One pulse mode timing example



1. IEDG1 = 1, OC1R = 2ED0h, OLVL1 = 0, OLVL2 = 1

Figure 52. Pulse width modulation mode timing example



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

Pulse width modulation mode

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

Pulse width modulation mode uses the complete output compare 1 function plus the OC2R register, and so this functionality can not be used when PWM mode is activated.

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

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

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

Procedure

To use pulse width modulation mode:

1. Load the OC2R register with the value corresponding to the period of the signal using the formula in the opposite column.
2. Load the OC1R register with the value corresponding to the period of the pulse if (OLVL1 = 0 and OLVL2 = 1) using *Equation 5*.
3. Select the following in the CR1 register:

   Using the OLVL1 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC1R register.

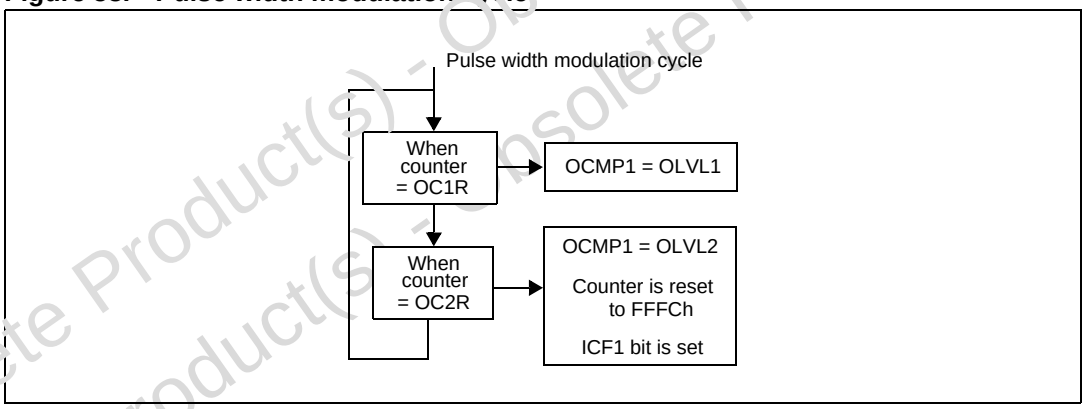
   Using the OLVL2 bit, select the level to be applied to the OCMP1 pin after a successful comparison with OC2R register.
4. Select the following in the CR2 register:

   Set OC1E bit: the OCMP1 pin is then dedicated to the output compare 1 function.

   Set the PWM bit.

   Select the timer clock (CC[1:0]) (see *Table 51*).

Figure 53. Pulse width modulation cycle



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

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

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

Equation 5

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

Where:

t = signal or pulse period (in seconds)

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

PRESC = timer prescaler factor (2, 4 or 8 depending on CC[1:0] bits, see *Table 51*)

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

Equation 6

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

Where:

t = signal or pulse period (in seconds)

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

The output compare 2 event causes the counter to be initialized to FFFCh (see *Figure 52*)

Note: 1 The OCF1 and OCF2 bits cannot be set by hardware in PWM mode therefore the output compare interrupt is inhibited.

2 The ICF1 bit is set by hardware when the counter reaches the OC2R value and can produce a timer interrupt if the ICIE bit is set and the I bit is cleared.

3 In PWM mode the ICAP1 pin can not be used to perform input capture because it is disconnected to the timer. The ICAP2 pin can be used to perform input capture (ICF2 can be set and IC2R can be loaded) but the user must take care that the counter is reset each period and ICF1 can also generates interrupt if ICIE is set.

4 When the pulse width modulation (PWM) and one pulse mode (OPM) bits are both set, the PWM mode is the only active one.

10.3.4 Low power modes

Table 47. Effect of low power modes on 16-bit timer

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

10.3.5 Interrupts

Table 48. 16-bit timer interrupt control/wake-up capability(1)

Interrupt event	Event flag	Enable control bit	Exit from WAIT	Exit from HALT
Input capture 1 event/counter reset in PWM mode	ICF1	ICIE	Yes	No
Input capture 2 event	ICF2		Yes	No
Output compare 1 event (not available in PWM mode)	OCF1	OCIE	Yes	No
Output compare 2 event (not available in PWM mode)	OCF2		Yes	No
Timer overflow event	TOF	TOIE	Yes	No

1. The 16-bit timer interrupt events are connected to the same interrupt vector (see *Section 7: Interrupts*). 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).

10.3.6 Summary of 16-bit timer modes

Table 49. Summary of 16-bit timer modes

Modes	Available resources			
	Input capture 1	Input capture 2	Output compare 1	Output compare 2
Input capture(1) and/or (2)	Yes	Yes	Yes	Yes
Output compare (1) and/or (2)	Yes	Yes	Yes	Yes
One pulse mode	No	Not recommended(1)	No	Partially(2)
PWM mode	No	Not recommended(3)	No	No

1. See note 4 in *One pulse mode on page 114*.
2. See note 5 in *One pulse mode on page 114*.
3. See note 4 in *Pulse width modulation mode on page 116*.

10.3.7 16-bit timer registers

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

Control register 1 (CR1)

CR1 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ICIE	OCIE	TOIE	FOLV2	FOLV1	OLVL2	IEDG1	OLVL1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 50. CR1 register description

Bit	Name	Function
7	ICIE	Input capture interrupt enable 0: Interrupt is inhibited 1: A timer interrupt is generated whenever the ICF1 or ICF2 bit of the SR register is set
6	OCIE	Output compare interrupt enable 0: Interrupt is inhibited 1: A timer interrupt is generated whenever the OCF1 or OCF2 bit of the SR register is set
5	TOIE	Timer overflow interrupt enable 0: Interrupt is inhibited 1: A timer interrupt is enabled whenever the TOF bit of the SR register is set
4	FOLV2	Forced output compare 2 This bit is set and cleared by software. 0: No effect on the OCMP2 pin 1: Forces the OLVL2 bit to be copied to the OCMP2 pin, if the OC2E bit is set and even if there is no successful comparison
3	FOLV1	Forced output compare 1 This bit is set and cleared by software. 0: No effect on the OCMP1 pin 1: Forces OLVL1 to be copied to the OCMP1 pin, if the OC1E bit is set and even if there is no successful comparison
2	OLVL2	Output level 2 This bit is copied to the OCMP2 pin whenever a successful comparison occurs with the OC2R register and OCxE is set in the CR2 register. This value is copied to the OCMP1 pin in one pulse mode and pulse width modulation mode.
1	IEDG1	Input edge 1 This bit determines which type of level transition on the ICAP1 pin triggers the capture. 0: A falling edge triggers the capture 1: A rising edge triggers the capture
0	OLVL1	Output level 1 The OLVL1 bit is copied to the OCMP1 pin whenever a successful comparison occurs with the OC1R register and the OC1E bit is set in the CR2 register.

Control register 2 (CR2)

CR2 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
OC1E	OC2E	OPM	PWM	CC[1:0]		IEDG2	EXEDG
R/W	R/W	R/W	R/W	R/W		R/W	R/W

Table 51. CR2 register description

Bit	Name	Function
7	OC1E	Output compare 1 pin enable This bit is used only to output the signal from the timer on the OCMP1 pin (OLV1 in output compare mode, both OLV1 and OLV2 in PWM and one-pulse mode). Whatever the value of the OC1E bit, the output compare 1 function of the timer remains active. 0: OCMP1 pin alternate function disabled (I/O pin free for general-purpose I/O) 1: OCMP1 pin alternate function enabled
6	0C2E	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
5	OPM	One pulse mode 0: One pulse mode is not active 1: One pulse mode is active, the ICAP1 pin can be used to trigger one pulse on the OCMP1 pin; the active transition is given by the IEDG1 bit. The length of the generated pulse depends on the contents of the OC1R register.
4	PWM	Pulse width modulation 0: PWM mode is not active 1: PWM mode is active, the OCMP1 pin outputs a programmable cyclic signal; the length of the pulse depends on the value of OC1R register; the period depends on the value of OC2R register.
3:2	CC[1:0]	Clock control The timer clock mode depends on the following bits: 00: Timer clock = $f_{CPU}/4$ 01: Timer clock = $f_{CPU}/2$ 10: Timer clock = $f_{CPU}/8$ 11: Timer clock = external clock (where available) *Note: If the external clock pin is not available, programming the external clock configuration stops the counter.*

Table 51. CR2 register description (continued)

Bit	Name	Function
1	IEDG2	Input edge 2 This bit determines which type of level transition on the ICAP2 pin triggers the capture. 0: A falling edge triggers the capture 1: A rising edge triggers the capture
0	EXEDG	External clock edge This bit determines which type of level transition on the external clock pin EXTCLK triggers the counter register. 0: A falling edge triggers the counter register 1: A rising edge triggers the counter register

Control/status register (CSR)

The 3 least significant bits are not used.

CSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ICF1	OCF1	TOF	ICF2	OCF2	TIMD	Reserved	
RO	RO	RO	RO	RO	RO	-	

Table 52. CSR register description

Bit	Name	Function
7	ICF1	Input capture flag 1 0: No input capture (reset value) 1: An input capture has occurred on the ICAP1 pin or the counter has reached the OC2R value in PWM mode. To clear this bit, first read the SR register, then read or write the low byte of the IC1R (IC1LR) register.
6	OCF1	Output compare flag 1 0: No match (reset value) 1: The content of the free running counter has matched the content of the OC1R register. To clear this bit, first read the SR register, then read or write the low byte of the OC1R (OC1LR) register.
5	TOF	Timer overflow flag 0: No timer overflow (reset value) 1: The free running counter rolled over from FFFFh to 0000h. To clear this bit, first read the SR register, then read or write the low byte of the CR (CLR) register. *Note: reading or writing the ACLR register does not clear TOF.*
4	ICF2	Input capture flag 2 0: No input capture (reset value) 1: An input capture has occurred on the ICAP2 pin. To clear this bit, first read the SR register, then read or write the low byte of the IC2R (IC2LR) register.

Table 52. CSR register description (continued)

Bit	Name	Function
3	OCF2	Output compare flag 2 0: No match (reset value) 1: The content of the free running counter has matched the content of the OC2R register. To clear this bit, first read the SR register, then read or write the low byte of the OC2R (OC2LR) register.
2	TIMD	Timer disable This bit is set and cleared by software. When set, it freezes the timer prescaler and counter and disabled the output functions (OCMP1 and OCMP2 pins) to reduce power consumption. Access to the timer registers is still available, allowing the timer configuration to be changed while it is disabled. 0: Timer enabled 1: Timer prescaler, counter and outputs disabled
1:0	-	Reserved, must be kept cleared

Input capture 1 high register (IC1HR)

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

IC1HR — Reset value: undefined

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Input capture 1 low register (IC1LR)

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

IC1LR — Reset value: undefined

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Output compare 1 high register (OC1HR)

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

OC1HR — Reset value: 1000 0000 (80h)

7	6	5	4	3	2	1	0
MSB							LSB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Output compare 1 low register (OC1LR)

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

OC1LR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
MSB							LSB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Output compare 2 high register (OC2HR)

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

OC2HR — Reset value: 1000 0000 (80h)

7	6	5	4	3	2	1	0
MSB							LSB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Output compare 2 low register (OC2LR)

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

OC2LR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
MSB							LSB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Counter high register (CHR)

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

CHR — Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Counter low register (CLR)

This is an 8-bit read-only 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.

CLR — Reset value: 1111 1100 (FCh)

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Alternate counter high register (ACHR)

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

ACHR — Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Alternate counter low register (ACLR)

This is an 8-bit read-only 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.

ACLR — Reset value: 1111 1100 (FCh)

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Input capture 2 high register (IC2HR)

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

IC2HR — Reset value: undefined

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

Input capture 2 low register (IC2LR)

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

IC2LR — Reset value: undefined

7	6	5	4	3	2	1	0
MSB							LSB
RO	RO	RO	RO	RO	RO	RO	RO

16-bit timer register map and reset values

Table 53. 16-bit timer register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
Timer A: 32 Timer B: 42	CR1 Reset value	ICIE 0	OCIE 0	TOIE 0	FOLV2 0	FOLV1 0	OLVL2 0	IEDG1 0	OLVL1 0
Timer A: 31 Timer B: 41	CR2 Reset value	OC1E 0	OC2E 0	OPM 0	PWM 0	CC1 0	CC0 0	IEDG2 0	EXEDG 0
Timer A: 33 Timer B: 43	CSR Reset value	ICF1 0	OCF1 0	TOF 0	ICF2 0	OCF2 0	TIMD 0	- 0	- 0
Timer A: 34 Timer B: 44	ICHR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 35 Timer B: 45	ICLR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 36 Timer B: 46	OCHR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 37 Timer B: 47	OCLR1 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3E Timer B: 4E	OCHR2 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3F Timer B: 4F	OCLR2 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 38 Timer B: 48	CHR Reset value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 39 Timer B: 49	CLR Reset value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3A Timer B: 4A	ACHR Reset value	MSB 1	1	1	1	1	1	1	LSB 1
Timer A: 3B Timer B: 4B	ACLR Reset value	MSB 1	1	1	1	1	1	0	LSB 0
Timer A: 3C Timer B: 4C	ICHR2 Reset value	MSB -	-	-	-	-	-	-	LSB -
Timer A: 3D Timer B: 4D	ICLR2 Reset value	MSB -	-	-	-	-	-	-	LSB -

10.4 Serial peripheral interface (SPI)

10.4.1 Introduction

The serial peripheral interface (SPI) allows full-duplex, synchronous, serial communication with external devices. An SPI system may consist of a master and one or more slaves or a system in which devices may be either masters or slaves.

10.4.2 Main features

- Full duplex synchronous transfers (on three lines)
- Simplex synchronous transfers (on two lines)
- Master or slave operation
- 6 master mode frequencies (f_{CPU}/4 max.)
- f_{CPU}/2 max; slave mode frequency (see note below)
- $\overline{SS}$ management by software or hardware
- Programmable clock polarity and phase
- End of transfer interrupt flag
- Write collision, master mode fault and overrun flags

Note: In slave mode, continuous transmission is not possible at maximum frequency due to the software overhead for clearing status flags and to initiate the next transmission sequence.

10.4.3 General description

Figure 54 on page 128 shows the serial peripheral interface (SPI) block diagram. There are three registers:

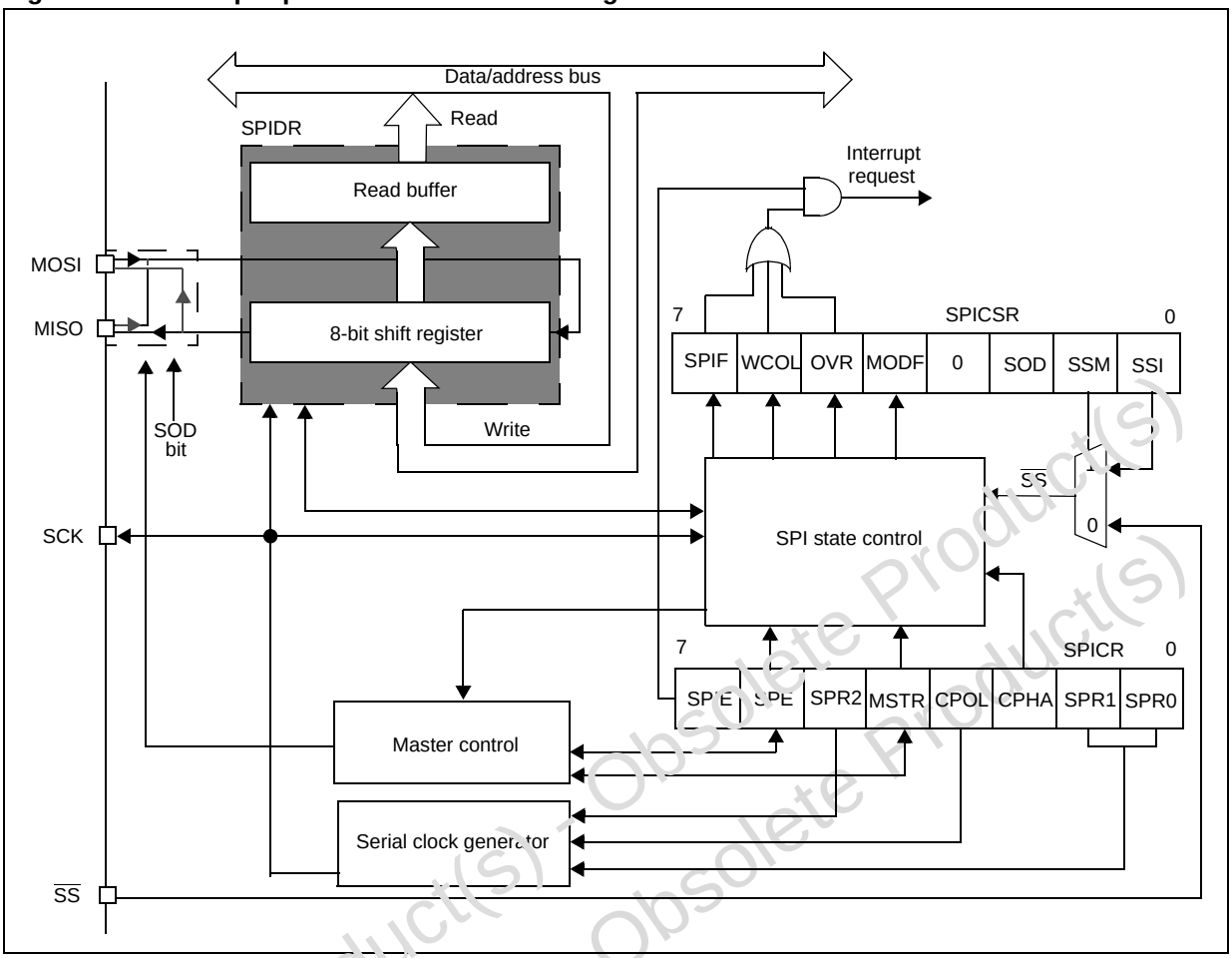
- SPI control register (SPICR)
- SPI control/status register (SPICSR)
- SPI data register (SPIDR)

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.

Figure 54. Serial peripheral interface block diagram



Functional description

A basic example of interconnections between a single master and a single slave is illustrated in *Figure 55*.

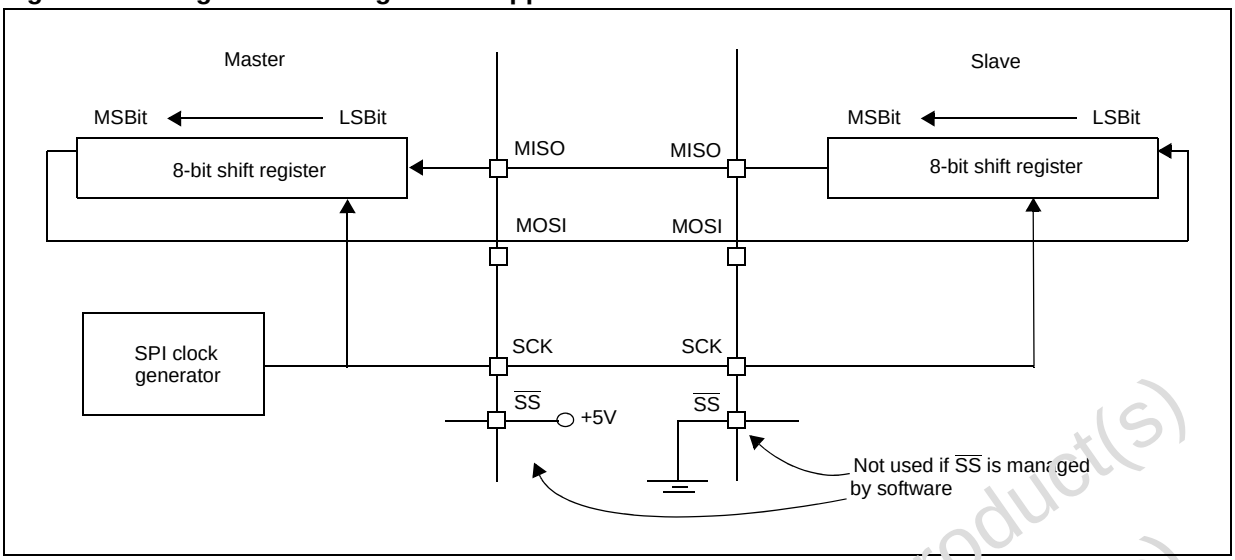
The MOSI pins are connected together and the MISO pins are connected together. In this way data is transferred serially between master and slave (most significant bit first).

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 58 on page 132*) but master and slave must be programmed with the same timing mode.

Figure 55. Single master/single slave application



Slave select management

As an alternative to using the $\overline{SS}$ pin to control the slave select signal, the application can choose to manage the slave select signal by software. This is configured by the SSM bit in the SPICSR register (see *Figure 57*).

In software management, the external $\overline{SS}$ pin is free for other application uses and the internal $\overline{SS}$ signal level is driven by writing to the SSI bit in the SPICSR register.

In master mode $\overline{SS}$ internal must be held high continuously.

In slave mode, there are two cases depending on the data/clock timing relationship (see *Figure 56*):

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 occurs when the slave writes to the shift register (see *Write collision error (WCOL) on page 133*).

Figure 56. Generic $\overline{SS}$ timing diagram

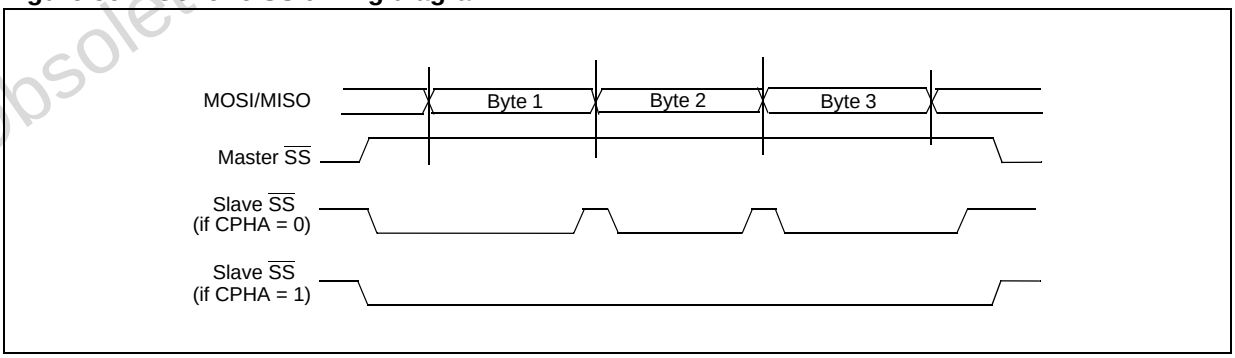
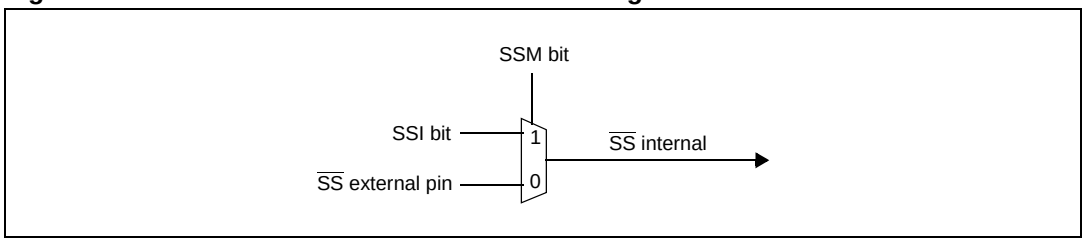


Figure 57. Hardware/software slave select management



Master mode operation

In master mode, the serial clock is output on the SCK pin. The clock frequency, polarity and phase are configured by software (refer to *SPI control/status register (SPICSR) on page 137*).

Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL = 1 or pulling down SCK if CPOL = 0).

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 58* shows the four possible configurations.

Note: The slave must have the same CPOL and CPHA settings as the master.

2. Write to the SPICSR register:
   - Either set the SSM bit and set the SSI bit or clear the SSM bit and tie the $\overline{SS}$ pin high for the complete byte transmit sequence.
3. Write to the SPICR register:
   - Set the MSTR and SPE bits

Note: MSTR and SPE bits remain set only if $\overline{SS}$ is high).

Caution: 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.

Master mode transmit sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MOSI pin most significant bit first.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if the SPIE bit is set and the interrupt mask in the CCR register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPICSR register while the SPIF bit is set
2. A read to the SPIDR register

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

Slave mode operation

In slave mode, the serial clock is received on the SCK pin from the master device.

To operate the SPI in slave mode:

1. Write to the SPICSR register to perform the following actions:
   - Select the clock polarity and clock phase by configuring the CPOL and CPHA bits (see *Figure 58*).

Note: The slave must have the same CPOL and CPHA settings as the master.

   - Manage the $\overline{SS}$ pin as described in *Slave select management on page 129* and *Figure 56*. If CPHA = 1 $\overline{SS}$ must be held low continuously. If CPHA = 0 $\overline{SS}$ must be held low during byte transmission and pulled up between each byte to let the slave write in the shift register.
2. Write to the SPICR register to clear the MSTR bit and set the SPE bit to enable the SPI I/O functions.

Slave mode transmit sequence

When software writes to the SPIDR register, the data byte is loaded into the 8-bit shift register and then shifted out serially to the MISO pin most significant bit first.

The transmit sequence begins when the slave device receives the clock signal and the most significant bit of the data on its MOSI pin.

When data transfer is complete:

- The SPIF bit is set by hardware.
- An interrupt request is generated if SPIE bit is set and interrupt mask in the CCR register is cleared.

Clearing the SPIF bit is performed by the following software sequence:

1. An access to the SPICSR register while the SPIF bit is set
2. A write or a read to the SPIDR register

Note: 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 *Overrun condition (OVR) on page 133*).

10.4.4 Clock phase and clock polarity

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits (see *Figure 58*).

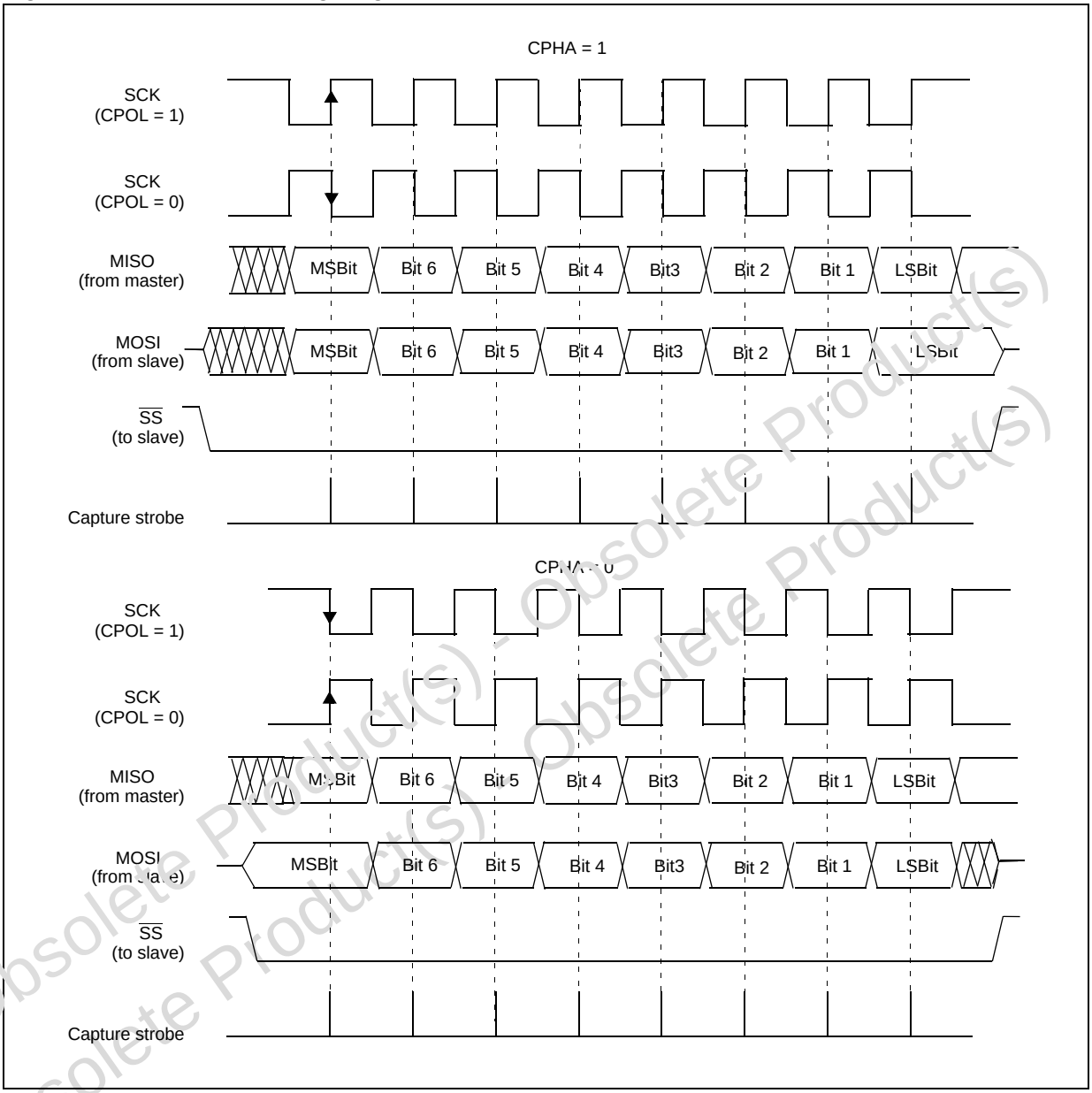
Note: The idle state of SCK must correspond to the polarity selected in the SPICSR register (by pulling up SCK if CPOL = 1 or pulling down SCK if CPOL = 0).

The combination of the CPOL clock polarity and CPHA (clock phase) bits selects the data capture clock edge.

Figure 58 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.

Figure 58. Data clock timing diagram



1. This figure should not be used as a replacement for parametric information. Refer to *Section 12: Electrical characteristics*.

10.4.5 Error flags

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.

Note: 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.

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.

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 is unsuccessful.

Write collisions can occur both in master and slave mode. See also *Slave select management on page 129*.

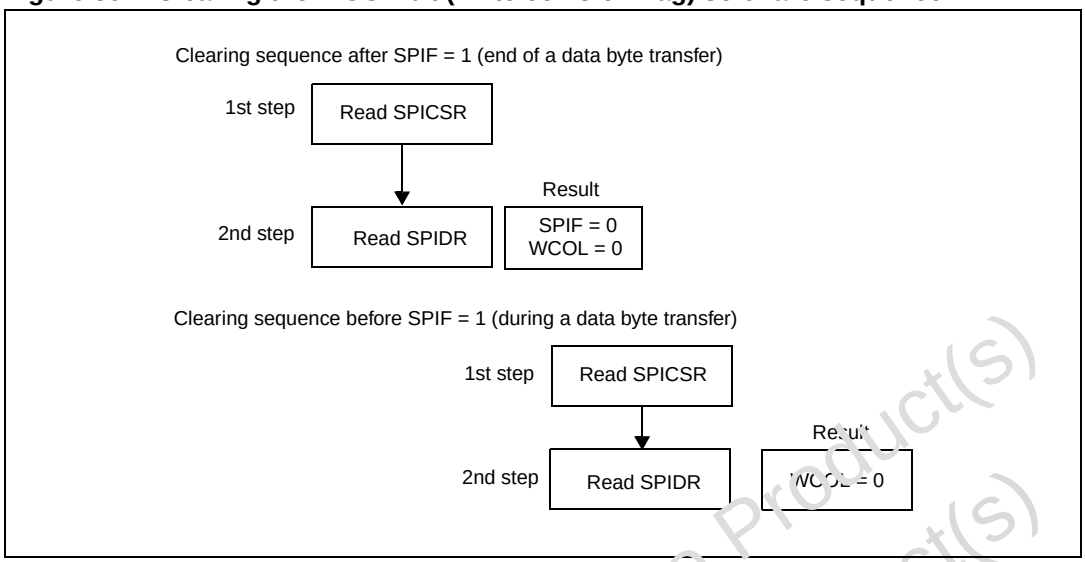
Note: A 'read collision' never occurs since the received data byte is placed in a buffer in which access is always synchronous with the CPU operation.

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 59*).

Figure 59. Clearing the WCOL bit (write collision flag) software sequence



1. Writing to the SPIDR register instead of reading it does not reset the WCOL bit.

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 60*).

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 are 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 receives 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.

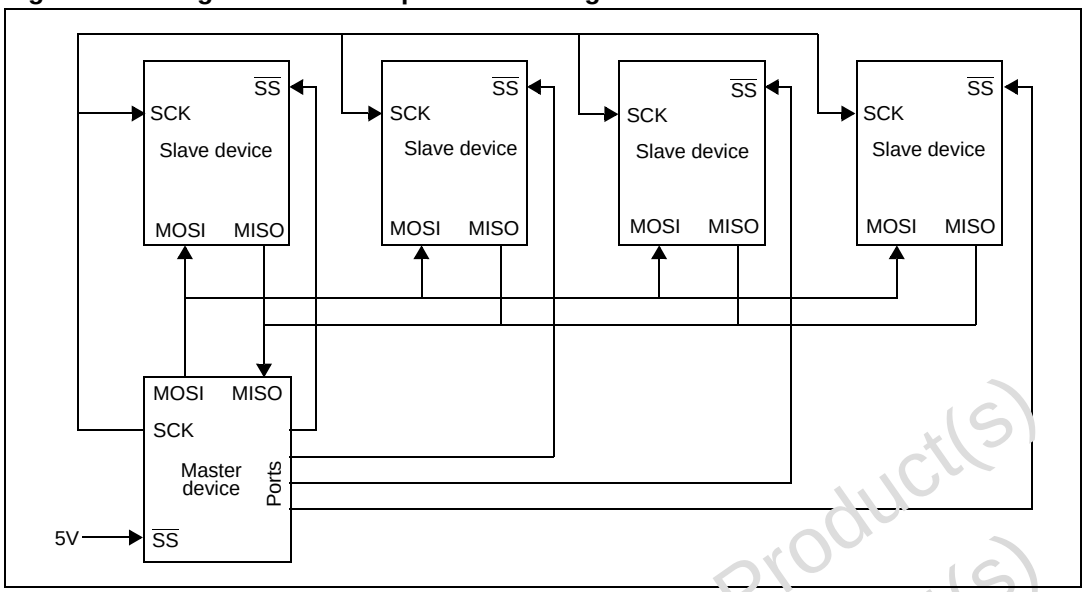
Other transmission security methods can use ports for handshake lines or data bytes with command fields.

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 60. Single master/multiple slave configuration



10.4.6 Low power modes

Table 54. Effect of low power modes on SPI

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.

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 *Slave select management on page 129*), make sure the master drives a low level on the $\overline{SS}$ pin when the slave enters Halt mode.

10.4.7 Interrupts

Table 55. SPI interrupt control/wake-up capability

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 SPI registers

SPI control register (SPICR)

SPICR Reset value: 0000 xxxx (0xh)

7	6	5	4	3	2	1	0
SPIE	SPE	SPR2	MSTR	CPOL	CPHA	SPR[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Table 56. SPICR register description

Bit	Name	Function
7	SPIE	Serial peripheral interrupt enable This bit is set and cleared by software. 0: Interrupt is inhibited 1: An SPI interrupt is generated whenever an end of transfer event, master mode fault or overrun error occurs (SPIF = 1, MODF = 1 or OVR = 1 in the SPICSR register).
6	SPE	Serial peripheral output enable This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see *Master mode fault (MODF) on page 133*). 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
5	SPR2	Divider enable This bit is set and cleared by software and is cleared by reset. 0: Divider by 2 enabled 1: Divider by 2 disabled This bit is used with the SPR[1:0] bits to set the baud rate: 100: Serial clock = $f_{CPU}/4$ 000: Serial clock = $f_{CPU}/8$ 001: Serial clock = $f_{CPU}/16$ 110: Serial clock = $f_{CPU}/32$ 010: Serial clock = $f_{CPU}/64$ 011: Serial clock = $f_{CPU}/128$ This bit has no effect in slave mode.

Table 56. SPICR register description (continued)

Bit	Name	Function
4	MSTR	Master mode This bit is set and cleared by software. It is also cleared by hardware when, in master mode, $\overline{SS}$ = 0 (see *Master mode fault (MODF) on page 133*). 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.
3	CPOL	Clock polarity This bit is set and cleared by software. This bit determines the idle state of the serial clock. The CPOL bit affects both the master and slave modes. 0: SCK pin has a low level idle state 1: SCK pin has a high level idle state *Note: If CPOL is changed at the communication byte boundaries, the SPI must be disabled by resetting the SPE bit.*
2	CPHA	Clock phase This bit is set and cleared by software. 0: The first clock transition is the first data capture edge 1: The second clock transition is the first capture edge *Note: The slave must have the same CPOL and CPHA settings as the master.*
1:0	SPR[1:0]	Serial clock frequency These bits are set and cleared by software. Used with the SPR2 bit, they select the baud rate of the SPI serial clock SCK output by the SPI in master mode. *Note: These 2 bits have no effect in slave mode.*

SPI control/status register (SPICSR)

SPICSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SPIF	WCOL	OVR	MODF	Reserved	SOD	SSM	SSI
RO	RO	RO	RO	-	R/W	R/W	R/W

Table 57. SPICSR register description

Bit	Name	Function
7	SPIF	Serial peripheral data transfer flag 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.*
6	WCOL	Write collision status 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 59*). 0: No write collision occurred 1: A write collision has been detected

Table 57. SPICSR register description (continued)

Bit	Name	Function
5	OVR	SPI overrun error 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 *Overrun condition (OVR) on page 133*). 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
4	MODF	Mode fault flag This bit is set by hardware when the $\overline{SS}$ pin is pulled low in master mode (see *Master mode fault (MODF) on page 133*). 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
3	-	Reserved, must be kept cleared
2	SOD	SPI output disable This bit is set and cleared by software. When set, it disables the alternate function of the SPI output (MOSI in master mode/MISO in slave mode). 0: SPI output enabled (if SPE = 1) 1: SPI output disabled
1	SSM	$\overline{SS}$ management This bit is set and cleared by software. When set, it disables the alternate function of the SPI $\overline{SS}$ pin and uses the SSI bit value instead (see *Slave select management on page 129*). 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)
0	SSI	$\overline{SS}$ internal mode This bit is set and cleared by software. It acts as a 'chip select' by controlling the level of the $\overline{SS}$ slave select signal when the SSM bit is set. 0: Slave selected 1: Slave deselected

SPI data I/O register (SPIDR)

SPIDR Reset value: undefined

7	6	5	4	3	2	1	0
D[7:0]							
R/W							

The SPIDR register is used to transmit and receive data on the serial bus. In a master device, a write to this register initiates the transmission/reception of another byte.

Note: 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 54*).

SPI register map and reset values

Table 58. SPI register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
0021h	SPIDR Reset value	MSB x	x	x	x	x	x	x	LSB x
0022h	SPICR Reset value	SPIE 0	SPE 0	SPR2 0	MSTR 0	CPOL x	CPHA x	SPR1 x	SPR0 x
0023h	SPICSR Reset value	SPIF 0	WCOL 0	OVR 0	MODF 0	0	SOD 0	SSM 0	SSI 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 *Section 10.5.5: SCI mode - functional description*. For LIN applications, refer to both *Section 10.5.5: SCI mode - functional description* and *Section 10.5.9: LIN mode - functional description*.

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

10.5.4 LINSCI serial communication interface - 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.

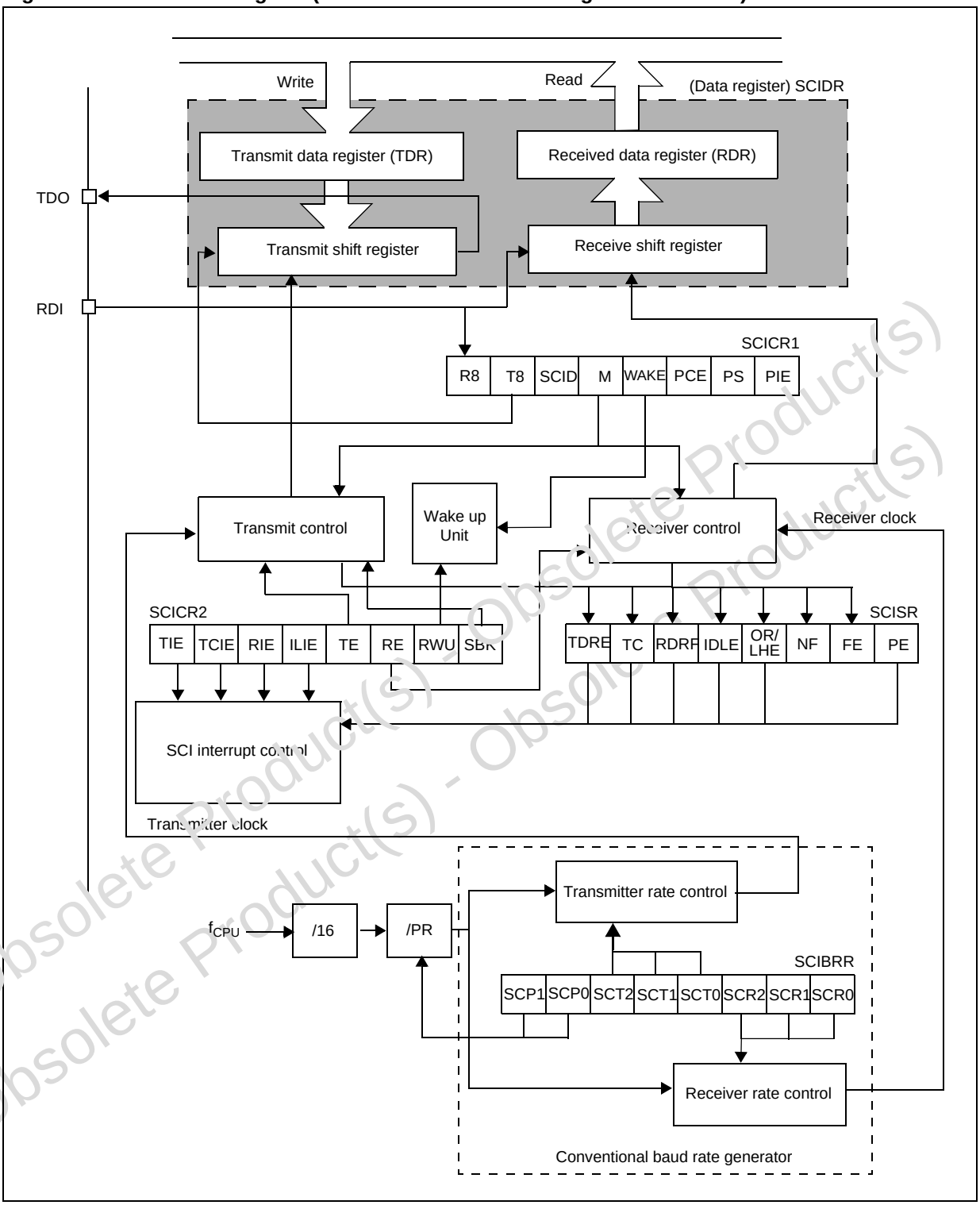
Through these pins, serial data is transmitted and received as characters comprising:

- 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

Figure 61. SCI block diagram (in conventional baud rate generator mode)



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 61*.

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 63*.

- an extended prescaler receiver register (SCIERPR)
- an extended prescaler transmitter register (SCIETPR)

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 62*).

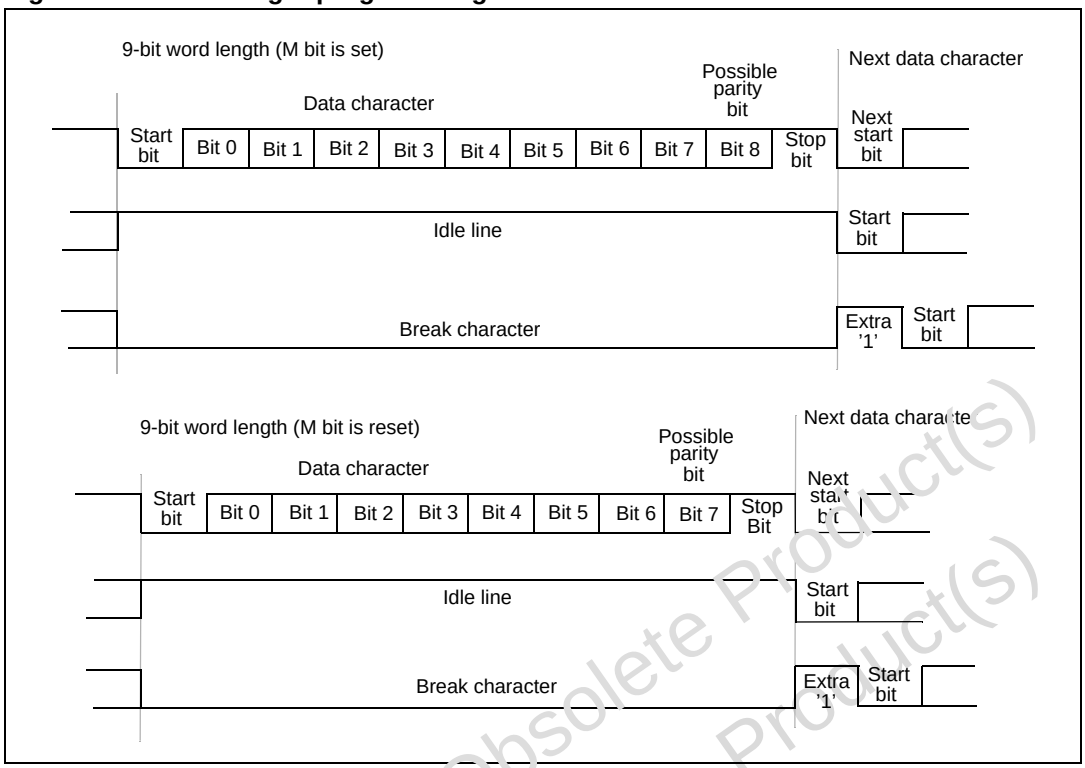
The TDO pin is in low state during the start bit.

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 62. Word length programming



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 61*).

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. Accessing the SCISR register.
2. Writing to the SCIDR register.

The TDRE bit is set by hardware and it indicates that:

- 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[|1:0] 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. Accessing the SCISR register.
2. Writing 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 62*).

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.

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 61*).

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. Accessing the SCISR register
2. Reading 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 is not lost.
- The shift register is 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.

Conventional baud rate generation

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

Equation 7

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

where

PR = 1, 3, 4 or 13 (see *SCI baud rate register (SCIBRR) on page 156*, SCP[1:0] bits)

TR = 1, 2, 4, 8, 16, 32, 64,128 (see *SCI baud rate register (SCIBRR) on page 156*, SCT[2:0] bits)

RR = 1, 2, 4, 8, 16, 32, 64,128 (see *SCI baud rate register (SCIBRR) on page 156*, SCR[2:0] bits)

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.

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 63*.

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

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:

Equation 8

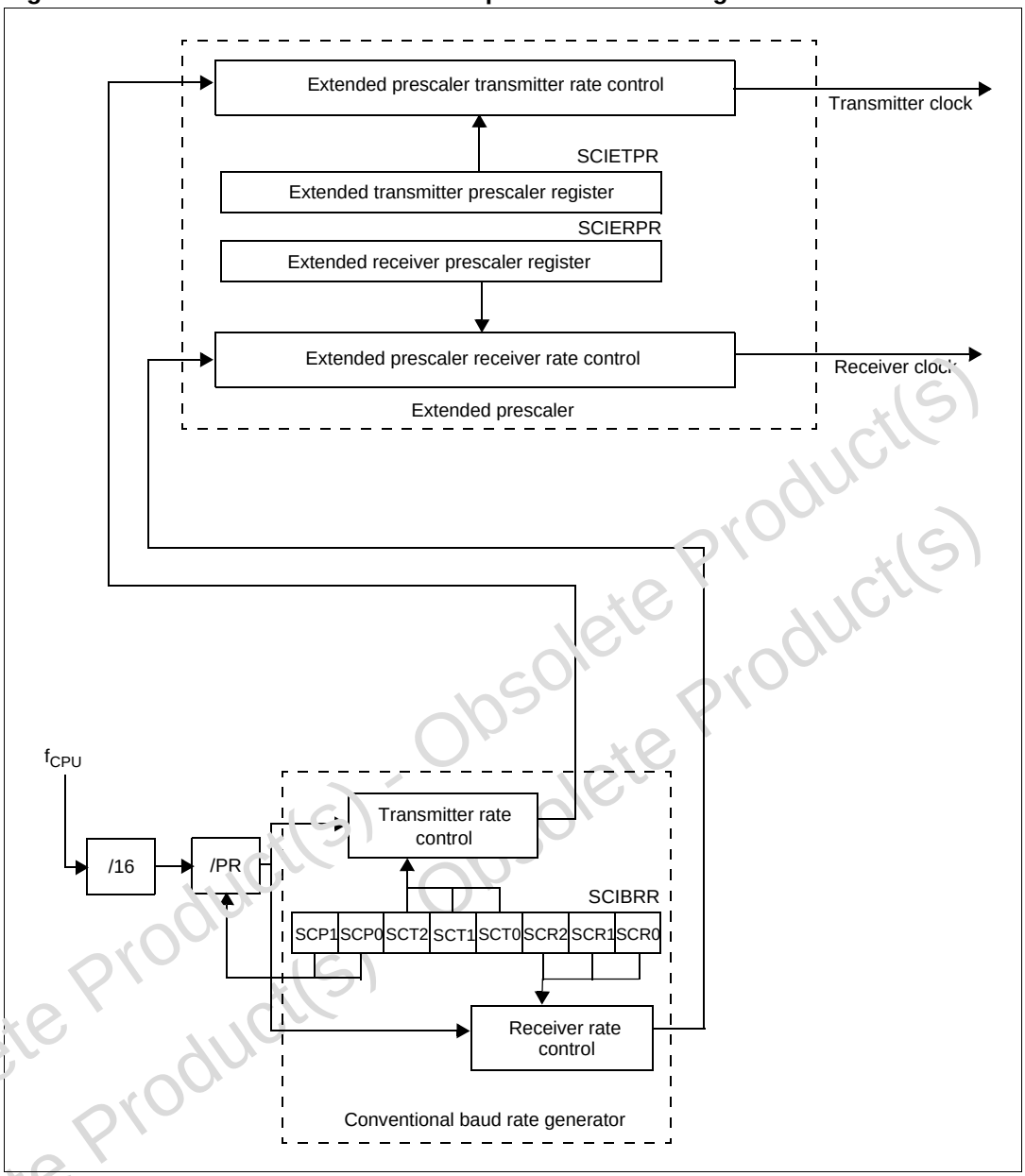
$$Tx = \frac{f_{CPU}}{16*ETPR*(PR*TR)} \quad Rx = \frac{f_{CPU}}{16*ERPR*(PR*RR)}$$

where

ETPR = 1, ..., 255 (see *SCI extended transmit prescaler division register (SCIETPR) on page 158*)

ERPR = 1, ..., 255 (see *SCI extended receive prescaler division register (SCIERPR) on page 157*)

Figure 63. SCI baud rate and extended prescaler block diagram



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 woken up and the first characters of the message which indicates the addressed receiver are analyzed. The receivers which are not addressed set RWU bit to enter in mute mode. Consequently, they do 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 receiver 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 receives 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 receive an address character (most significant bit = ‘1’), the receivers are woken up. The receivers which are not addressed set the RWU bit to enter in mute mode. Consequently, they do not treat the next characters constituting the next part of the message.

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 59*.

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 59. Character formats

M bit	PCE bit	Character format
0	0	| SB(1) | 8 bit data | STB(2) |
	1	| SB | 7-bit data | PB(3) | STB |
1	0	| SB | 9-bit data | STB |
	1	| SB | 8-bit data | PB | STB |

1. SB = start bit
2. STB = stop bit
3. 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 is 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 is 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, 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.

10.5.6 Low power modes

Table 60. Effect of low power modes on SCI

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

Table 61. SCI interrupt control/wake-up capability

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 *Section 7: Interrupts*).

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).

10.5.8 SCI mode registers

SCI status register (SCISR)

SCISR Reset value: 1100 0000 (C0h)

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	OR(1)	NF(1)	FE(1)	PE(1)
RO	RO	RO	RO	RO	RO	RO	RO

1. This bit has a different function in LIN mode, please refer to *Section 10.5.10: LIN mode registers*.

Table 62. SCISR register description

Bit	Name	Function
7	TDRE	Transmit data register empty 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
6	TC	Transmission complete 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.*
5	RDRF	Received data ready flag 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
4	IDLE	Idle line detected 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 is not set again until the RDRF bit has been set itself (that is, a new idle line occurs).*

Table 62. SCISR register description (continued)

Bit	Name	Function
3	OR	Overrun error 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 are not lost but the shift register is overwritten.*
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.*
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 is transferred and only the OR bit is set.*
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

SCI control register 1 (SCICR1)

SCICR1 Reset value: x000 0000 (x0h)

7	6	5	4	3	2	1	0
R8	T8	SCID	M	WAKE	PCE(1)	PS	PIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

1. This bit has a different function in LIN mode; please refer to *Section 10.5.10: LIN mode registers*

Table 63. SCICR1 register description

Bit	Name	Function
7	R8	Receive data bit 8 This bit is used to store the 9th bit of the received word when M = 1.
6	T8	Transmit data bit 8 This bit is used to store the 9th bit of the transmitted word when M = 1.
5	SCID	Disabled for low power consumption When this bit is set the SCI prescalers and outputs are stopped at 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
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).*
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.*
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
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 is selected after the current byte. 0: Even parity 1: Odd parity
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

SCI control register 2 (SCICR2)

SCICR2 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU(1)	SBK(1)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

1. This bit has a different function in LIN mode; please refer to *Section 10.5.10: LIN mode registers*.

Table 64. SCICR2 register description

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

Table 64. SCICR2 register description (continued)

Bit	Name	Function
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 wakeup 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.*
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 sends a BREAK word at the end of the current word.*

SCI data register (SCIDR)

SCIDR — Reset value: undefined

7	6	5	4	3	2	1	0
DR[7:0]							
R/W							

The data register contains the received or transmitted data character, depending on whether it is read from or written to.

This 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 61*).

The RDR register provides the parallel interface between the input shift register and the internal bus (see *Figure 61*).

SCI baud rate register (SCIBRR)

SCIBRR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SCP[1:0]		SCT[2:0]			SCR[2:0]		
R/W		R/W			R/W		

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

Table 65. SCIBRR register description

Bit	Name	Function
7:6	SCP[1:0]	First SCI prescaler These 2 prescaling bits allow several standard clock division ranges: 00: PR prescaling factor = 1 01: PR prescaling factor = 3 10: PR prescaling factor = 4 11: PR prescaling factor = 13
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: 000: TR dividing factor = 1 001: TR dividing factor = 2 010: TR dividing factor = 4 011: TR dividing factor = 8 100: TR dividing factor = 16 101: TR dividing factor = 32 110: TR dividing factor = 64 111: TR dividing factor = 128
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: 000: RR dividing factor = 1 001: RR dividing factor = 2 010: RR dividing factor = 4 011: RR dividing factor = 8 100: RR dividing factor = 16 101: RR dividing factor = 32 110: RR dividing factor = 64 111: RR dividing factor = 128

SCI extended receive prescaler division register (SCIERPR)

SCIERPR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ERPR[7:0]							
R/W							

Table 66. SCIERPR register description

Bit	Name	Function
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 63*) 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.

SCI extended transmit prescaler division register (SCIETPR)

SCIETPR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ETPR[7:0]							
R/W							

Table 67. SCIETPR register description

Bit	Name	Function
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 63*) 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.*

10.5.9 LIN mode - functional description

The block diagram of the serial control interface in LIN slave mode is shown in *Figure 65*.

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 10.5.10: LIN mode registers* for the definitions of each bit.

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 sends 13 low bits.

The baud rate can then be programmed using the SCIBRR, SCIERPR and SCIETPR registers.

In LIN master mode, the conventional prescaler 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 has no effect.

In LIN slave mode the LIN baud rate generator is selected instead of the conventional prescaler or extended prescaler. The LIN baud rate generator is common to the transmitter and the receiver.

The baud rate can then 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.

LIN transmission

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

The procedure to transmit the LIN header is as follows:

1. First set the SBK bit in the SCICR2 register to start transmitting a 13-bit LIN synch break.
2. Reset the SBK bit.
3. Load the LIN synch field (0x55) in the SCIDR register to request synch field transmission.
4. Wait until the SCIDR is empty (TDRE bit set in the SCISR register).
5. Load the LIN message identifier in the SCIDR register to request identifier transmission.

Figure 64. LIN characters

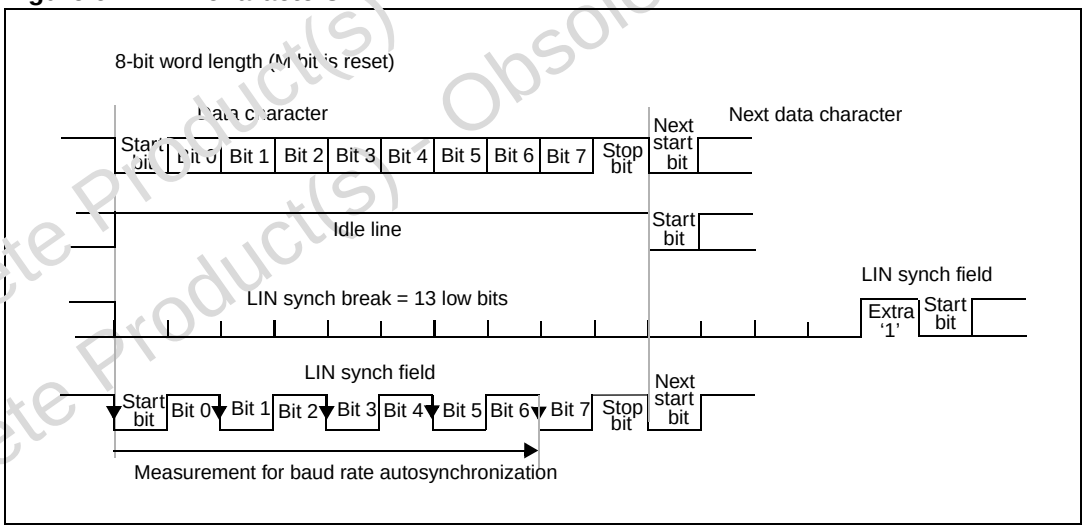
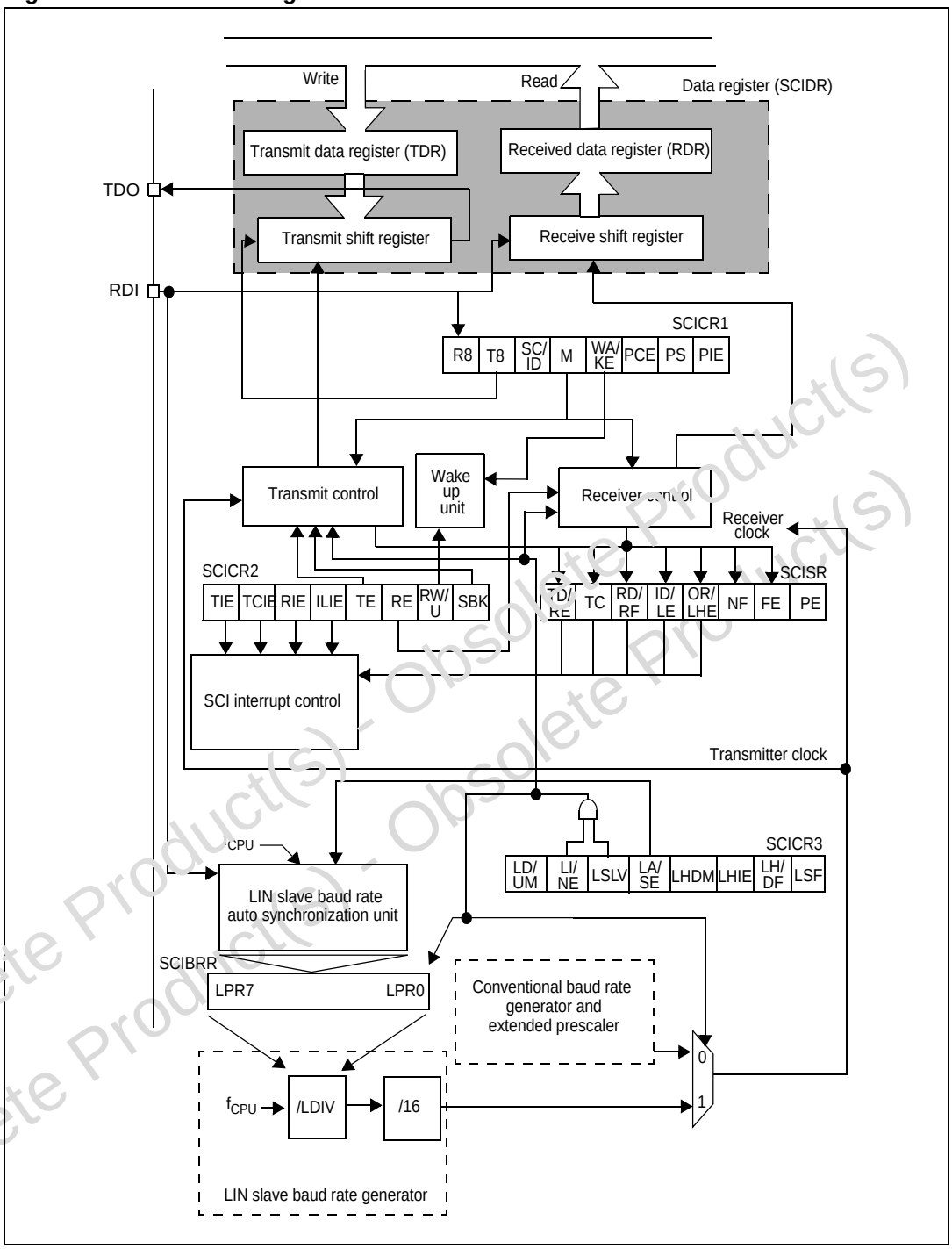


Figure 65. SCI block diagram in LIN slave mode



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 *LIN baud rate on page 165*.

LIN header handling by a slave

Depending on the LIN header detection method the LINSCI signals 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.

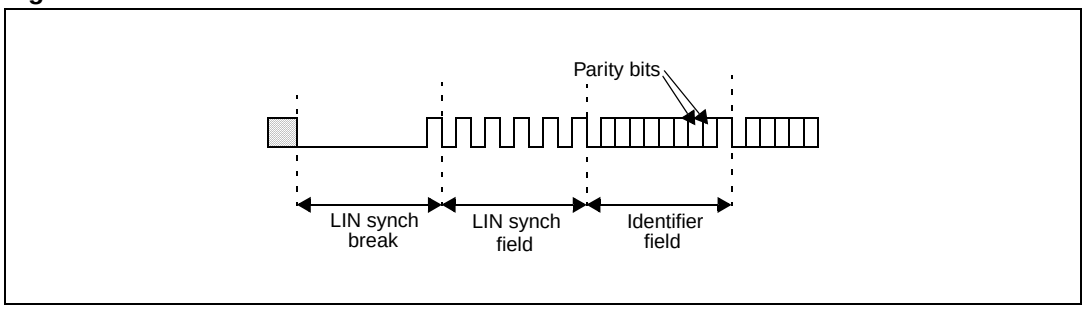
Note: During LIN synch field measurement, the SCI state machine is switched off: No characters are transferred to the data register.

LIN slave parity

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

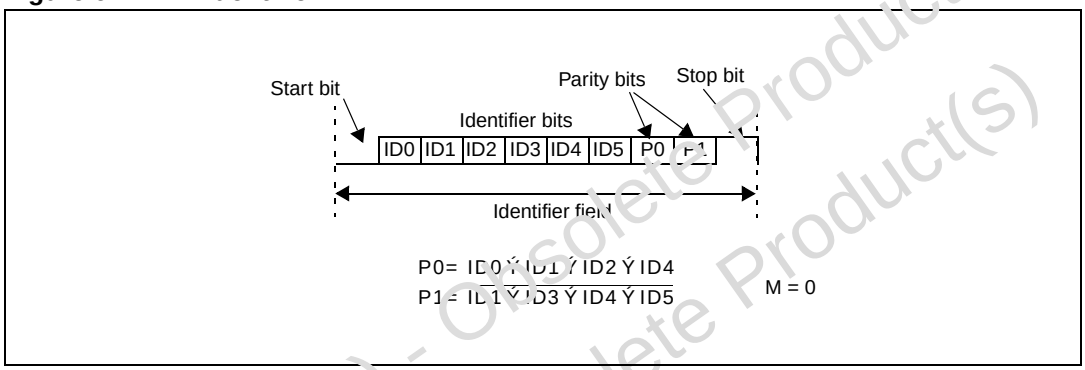
In this case, the parity bits of the LIN identifier field are checked. The identifier character is recognized as the third received character after a break character (included) (see *Figure 66*).

Figure 66. LIN header



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 in *Figure 67* by the LIN specification.

Figure 67. LIN identifier



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 the following:

- The received LIN synch field is not equal to 55h.
- An overrun has 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 the following:

- 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 has occurred. If $T_{HEADER} > T_{HEADER\_MAX}$ then the LHE flag is set (only if LHDM is set to 1), see *Figure 68*.
- 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).
- An overrun has occurred on fields other than the synch field (as in standard SCI mode).

Deviation error on the synch field

The deviation error is checked 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.

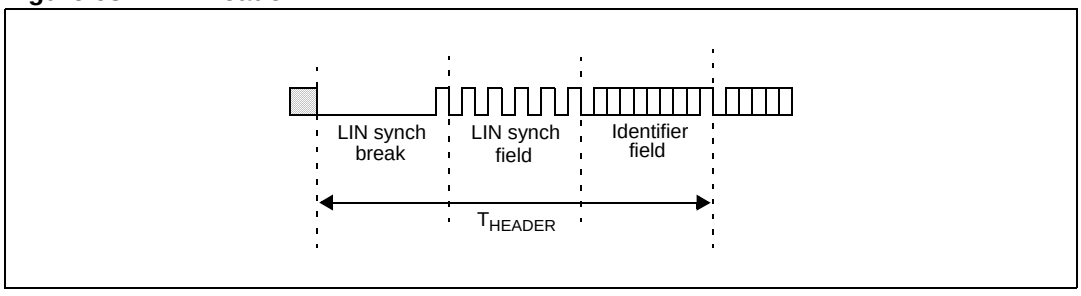
When LHE is set due to this error then the SCI goes into a blocked state (LSF bit is set).

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 68. LIN header



The time-out counter is enabled at each break detection. It is stopped in the following situations:

- 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
- The 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)

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 = 39 T_{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} + 10 T_{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 the T_{FRAME_MAX} condition given by the LIN protocol, at software level.

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

Mute mode and errors

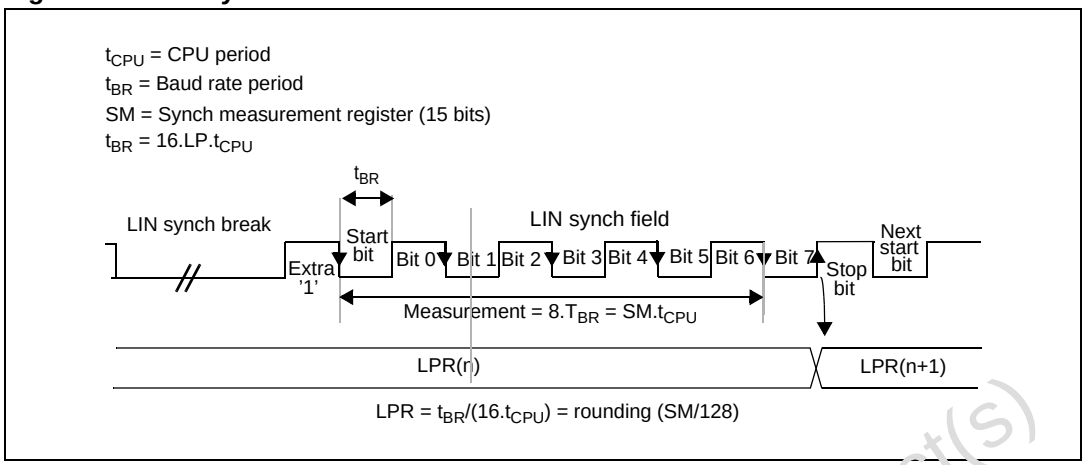
In mute mode when LHDM bit = 1, if an LHE error occurs during the analysis of the LIN synch field or if a LIN header time-out occurs then the LHE bit is set but it does not wake up from mute mode. In this case, the current header analysis is discarded. If needed, the software has to reset LSF bit. Then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

In mute mode, if a framing error occurs on 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 69. LIN synch field measurement



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 69*). 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.

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. Thus, the baud rate for the receiver and transmitter are both set to the same value, which depends on the LIN Slave baud rate generator:

Equation 9

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

where

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 70* and *Figure 71*. It depends on the LDUM bit setting (LIN divider update method)

Note: As explained in Figure 70 and Figure 71, 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.

Figure 70. LDIV read/write operations when LDUM = 0

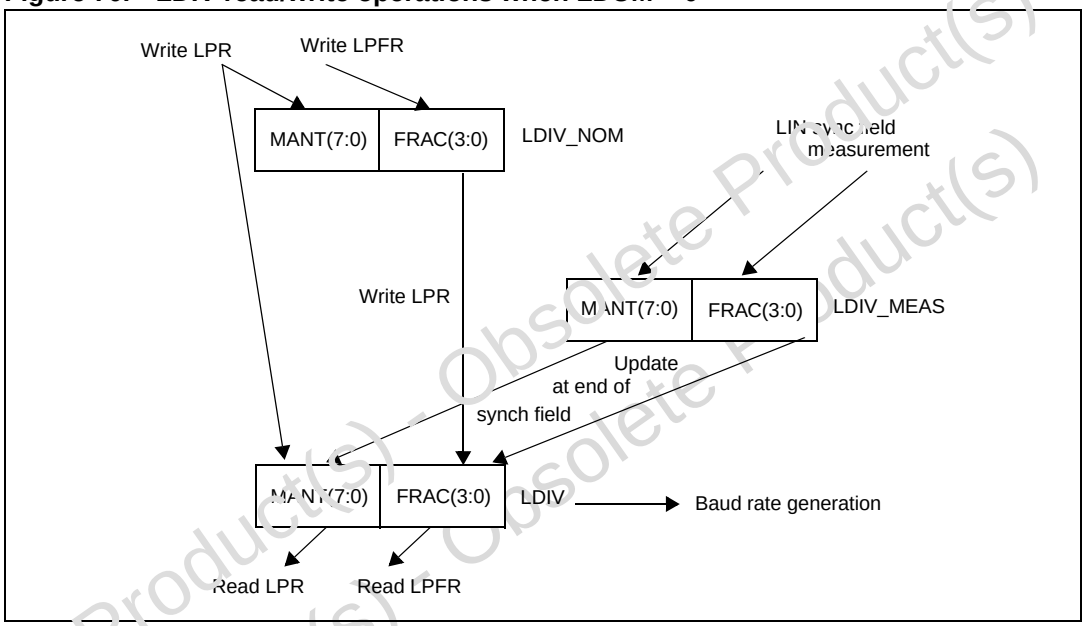
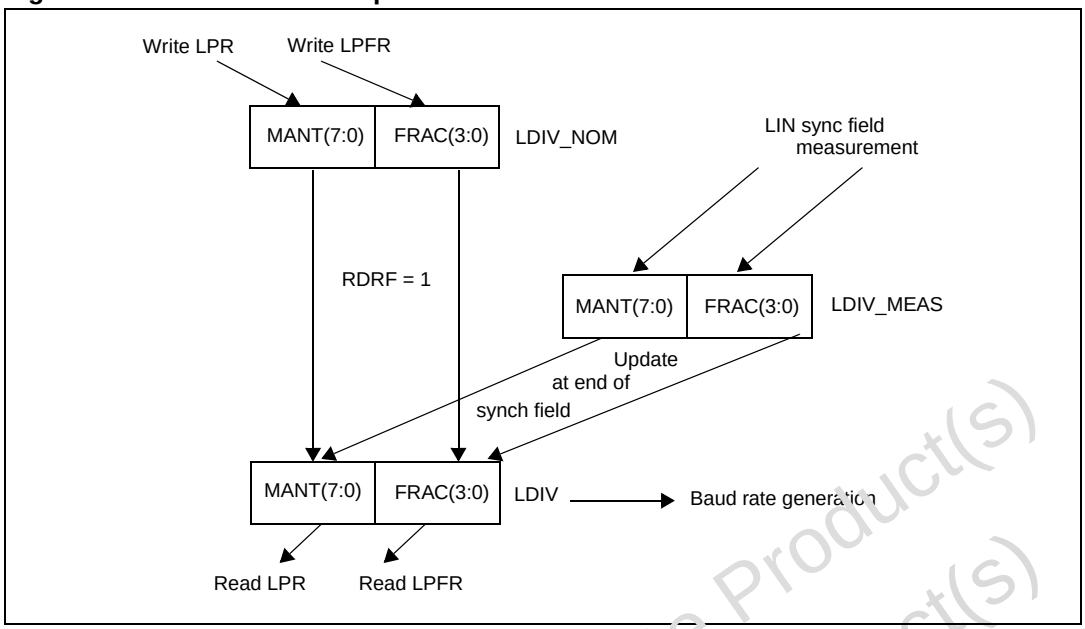


Figure 71. LDIV read/write operations when LDUM = 1



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%.

Clock deviation causes

The causes which contribute to the total deviation are:

- D_{TRA}: deviation due to transmitter error.

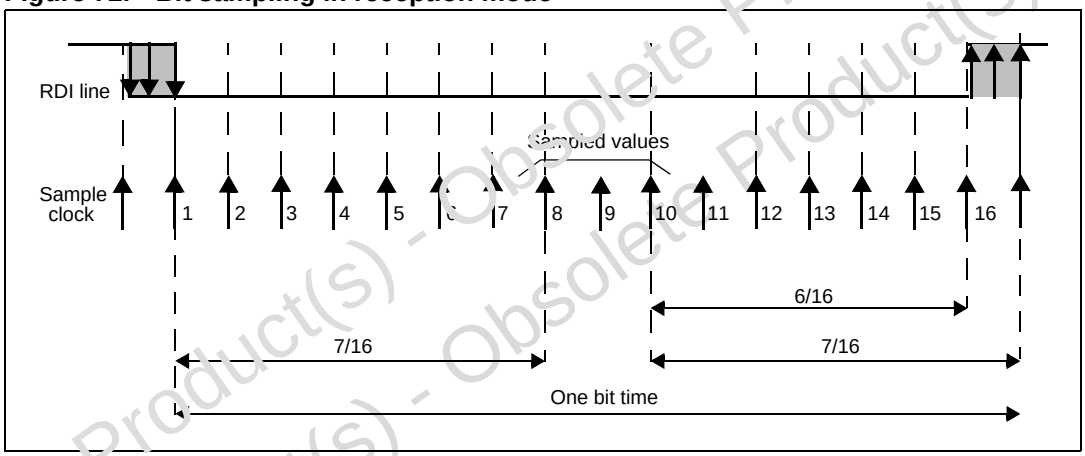
Note: The transmitter can be either a master or a slave (in case of a slave listening to the response of another slave).

- 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 72. Bit sampling in reception mode



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:

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

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

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 a step.

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

Impact of clock deviation on maximum baud rate

The choice of the nominal baud rate ($LDIV_{NOM}$) influences 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.

10.5.10 LIN mode registers

SCI status register (SCISR)

SCISR Reset value: 1100 0000 (C0h)

7	6	5	4	3	2	1	0
TDRE	TC	RDRF	IDLE	LHE	NF	FE	PE
RO	RO	RO	RO	RO	RO	RO	RO

Table 68. SCISR register description(1)

Bit	Name	Function
7	TDRE	Transmit data register empty 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
6	TC	Transmission complete 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.*
5	RDRF	Received data ready flag 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
4	IDLE	Idle line detected 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 is not set again until the RDRF bit has been set itself (that is, a new idle line occurs).*
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 *SCI status register (SCISR) on page 152*). 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 SCI status register (SCISR) on page 152).*

Table 68. SCISR register description(1) (continued)

Bit	Name	Function
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 *SCI status register (SCISR) on page 152*. In LIN slave mode (LINE bit = 1 and LSLV bit = 1) this bit has no meaning.
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
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

1. Bits 7:4 have the same function as in SCI mode, please refer to *SCI status register (SCISR) on page 152*.

SCI control register 1 (SCICR1)

SCICR1 Reset value: x000 0000 (x0h)

7	6	5	4	3	2	1	0
R8	T8	SCID	M	WAKE	PCE	Reserved	PIE
R/W	R/W	R/W	R/W	R/W	R/W	-	R/W

Table 69. SCICR1 register description(1)

Bit	Name	Function
7	R8	Receive data bit 8 This bit is used to store the 9th bit of the received word when M = 1.
6	T8	Transmit data bit 8 This bit is used to store the 9th bit of the transmitted word when M = 1.
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

Table 69. SCICR1 register description(1) (continued)

Bit	Name	Function
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).*
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.*
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.
1	-	Reserved, must be kept cleared
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

1. Bits 7:3 and bit 0 have the same function as in SCI mode; please refer to *SCI control register 1 (SCICR1) on page 153*.

SCI control register 2 (SCICR2)

SCICR2 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 70. SCICR2 register description(1)

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

Table 70. SCICR2 register description(1) (continued)

Bit	Name	Function
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
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
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.*
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
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 LIN reception on page 161.* *- In LIN slave mode, when RDRF is set, the software can not set or clear the RWU 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 sends a BREAK word at the end of the current word.*

1. Bits 7:2 have the same function as in SCI mode; please refer to *SCI control register 2 (SCICR2) on page 155*.

SCI control register 3 (SCICR3)

SCICR3 Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
LDUM	LINE, LSLV		LASE	LHDM	LHIE	LHDF	LSF
R/W	R/W		R/W	R/W	R/W	R/W	R/W

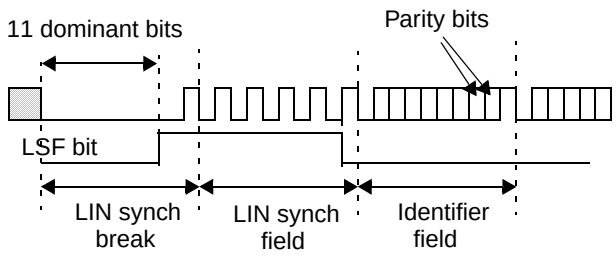
Table 71. SCICR3 register description

Bit	Name	Function
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 is 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.*
6:5	LINE, LSLV	LIN mode enable bits These bits configure the LIN mode: 0x: LIN mode disabled 10: LIN master mode 11: LIN slave mode **The LIN master configuration** enables the cabability 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)
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

Table 71. SCICR3 register description (continued)

Bit	Name	Function
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 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
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
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 *Note: 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.*
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 73*). 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 73. LSF bit set and clear



10.5.11 LIN divider (LDIV) 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)

LPR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
LPR[7:0]							
R/W							

Table 72. LPR register description

Bit	Name	Function
7:0	LPR[7:0]	LIN prescaler (mantissa of LDIV) These 8 bits define the value of the mantissa of the LDIV (see *Table 73*).

Table 73. LIN mantissa rounded values

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.

LIN prescaler fraction register (LPFR)

LPFR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved				LPFR[3:0]			
-				R/W			

Table 74. LPFR register description

Bit	Name	Function
7:4	-	Reserved
3:0	LPFR[3:0]	Fraction of LDIV These 4 bits define the fraction of the LDIV (see *Table 75*).

Table 75. LDIV fractions

LPFR[3:0]	Fraction (LDIV)
0h	0
1h	1/16
...	...
Eh	14/16
Fh	15/16

Note: 1 When initializing LDIV, the LPFR register must be written first. Then, the write to the LPR register effectively updates 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 (25.62d) = 25d = 1Bh

Example 3: LDIV = 25.99d

This leads to:

LPFR = rounded(16*0.99d) = rounded(15.84d) = 16d

The carry must be propagated to the mantissa: LPR = mantissa (25.99) + 1 = 26d = 1Ch.

LIN header length register (LHLR)

LHLR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
LHLR[7:0]							
RO							

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.

Table 76. LHLR register description

Bit	Name	Function
7:0	LHLR[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 as 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 register coding:

T_{HEADER_MAX} = 57

LHL (7:2) represents the mantissa of (57 - T_{HEADER}) (see *Table 77*)

LHL (1:0) represents the fraction (57 - T_{HEADER}) (see *Table 78*)

Table 77. LIN header mantissa values

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

Table 78. LIN header fractions

LHL[1:0]	Fraction (57 - T_{HEADER})
0h	0
1h	1/4
2h	1/2
3h	3/4

Examples 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

SCI register map and reset values

Table 79. SCI register map and reset values

Addr. (Hex.)	Register name	7	6	5	4	3	2	1	0
0018h	SCI1SR Reset value	TDRE 1	TC 1	RDRF 0	IDLE 0	OR/LHE 0	NF 0	FE 0	PE 0
0019h	SCI1DR Reset value	DR7 -	DR6 -	DR5 -	DR4 -	DR3 -	DR2 -	DR1 -	DR0 -
001Ah	SCI1BRR LPR (LIN slave mode) Reset value	SCP1 LPR7 0	SCP0 LPR6 0	SCT2 LPR5 0	SCT1 LPR4 0	SCT0 LPR3 0	SCR2 LPR2 0	SCR1 LPR1 0	SCR0 LPR0 0
001Bh	SCI1CR1 Reset value	R8 x	T8 0	SCID 0	M 0	WAKE 0	PCE 0	PS 0	PIE 0
001Ch	SCI1CR2 Reset value	TIE 0	TCIE 0	RIE 0	ILIE 0	TE 0	RE 0	RWU 0	SBK 0
001Dh	SCI1CR3 Reset Value	LDUM 0	LINE 0	LSLV 0	LASE 0	LHDM 0	LHIE 0	LHDF 0	LSF 0
001Eh	SCI1ERPR LHLR (LIN slave mode) Reset value	ERPR LHL7 0	ERPR LHL6 0	ERPR LHL5 0	ERPR LHL4 0	ERPR3 LHL3 0	ERPR LHL2 0	ERPR LHL1 0	ERPR LHL0 0
001Fh	SCI1TPR LPRF (LIN slave mode) Reset value	ETPR7 0 0	ETPR6 0 0	ETPR5 0 0	ETPR4 0 0	ETPR3 LPRF3 0	ETPR2 LPRF2 0	ETPR1 LPRF1 0	ETPR0 LPRF0 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, that is, 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 200mV 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 80. MTC functional blocks

Table 80. MTC functional blocks (continued)

Table 81. MTC registers

Register	Description	Register page (RPGS bit)	Page
MTIM	Timer counter register	0	260
MTIML	Timer counter register LSB (mode dependent)	0	260
MZPRV	Capture Z_{n-1} register	0	260
MZREG	Capture Z_n register	0	261
MCOMP	Compare C_{n+1} register	0	261
MDREG	Demagnetization register	0	261
MWGHT	A_n weight register	0	262
MPRSR	Prescaler and sampling register	0	262
MIMR	Interrupt mask register	0	263
MISR	Interrupt status register	0	264
MCRA	Control register A	0	265
MCRB	Control register B	0	268
MCRC	Control register C	0	270
MPHST	Phase state register	0	271
MCFR	Motor current feedback register	0	272
MDFR	Motor D event filter register	0	273
MREF	Reference register	0	275
MPCR	PWM control register	0	276
MREP	Repetition counter register	0	277
MCPWH	Compare phase W preload register high	0	277
MCPWL	Compare phase W preload register low	0	277
MCPVH	Compare phase V preload register high	0	278
MCPVL	Compare phase V preload register low	0	278
MCPUH	Compare phase U preload register high	0	278
MCPUL	Compare phase U preload register low	0	279
MCP0H	Compare 0 preload register high	0	279
MCP0L	Compare 0 preload register low	0	279
MDTG	Deadtime generator register	1	280
MPOL	Polarity register	1	282
MPWME	PWM register	1	283
MCONF	Configuration register	1	284
MPAR	Parity register	1	285
MZFR	Motor Z event filter register	1	286
MSCR	Motor sampling clock register	1	288

10.6.2 Main features

- 2 on-chip analog comparators: one for BEMF zero-crossing detection, the other for current regulation or limitation
- 7 selectable reference voltages for the hysteresis comparator (0.2V, 0.6V, 1V, 1.5V, 2V, 2.5V, 3.5V) 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 and 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 *Section 14: ST7MCxxx-Auto device configuration and ordering information*).
  - An 'emergency stop' input pin (active low) asynchronously forces the outputs in HiZ, Low or High state, depending on option byte 1 (refer to *Section 14: ST7MCxxx-Auto device configuration and ordering information*).

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 75* shows the phase steps and voltage, while *Table 82* shows the relevant phase configurations.

To run this kind of motor efficiently, an autoswitching mode has to be used, that is, 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.

Figure 74. Chronogram of events (in autoswitched mode)

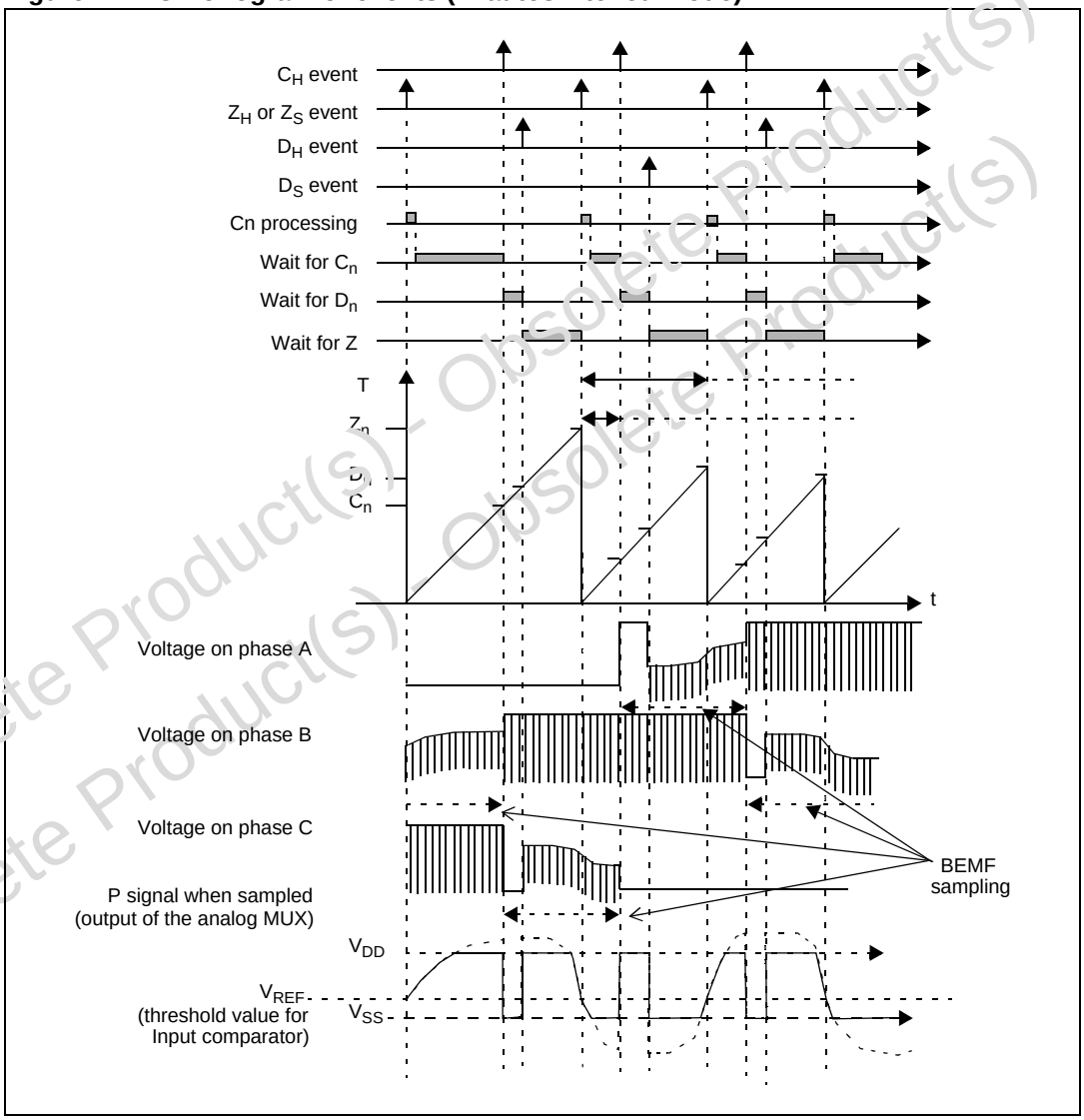
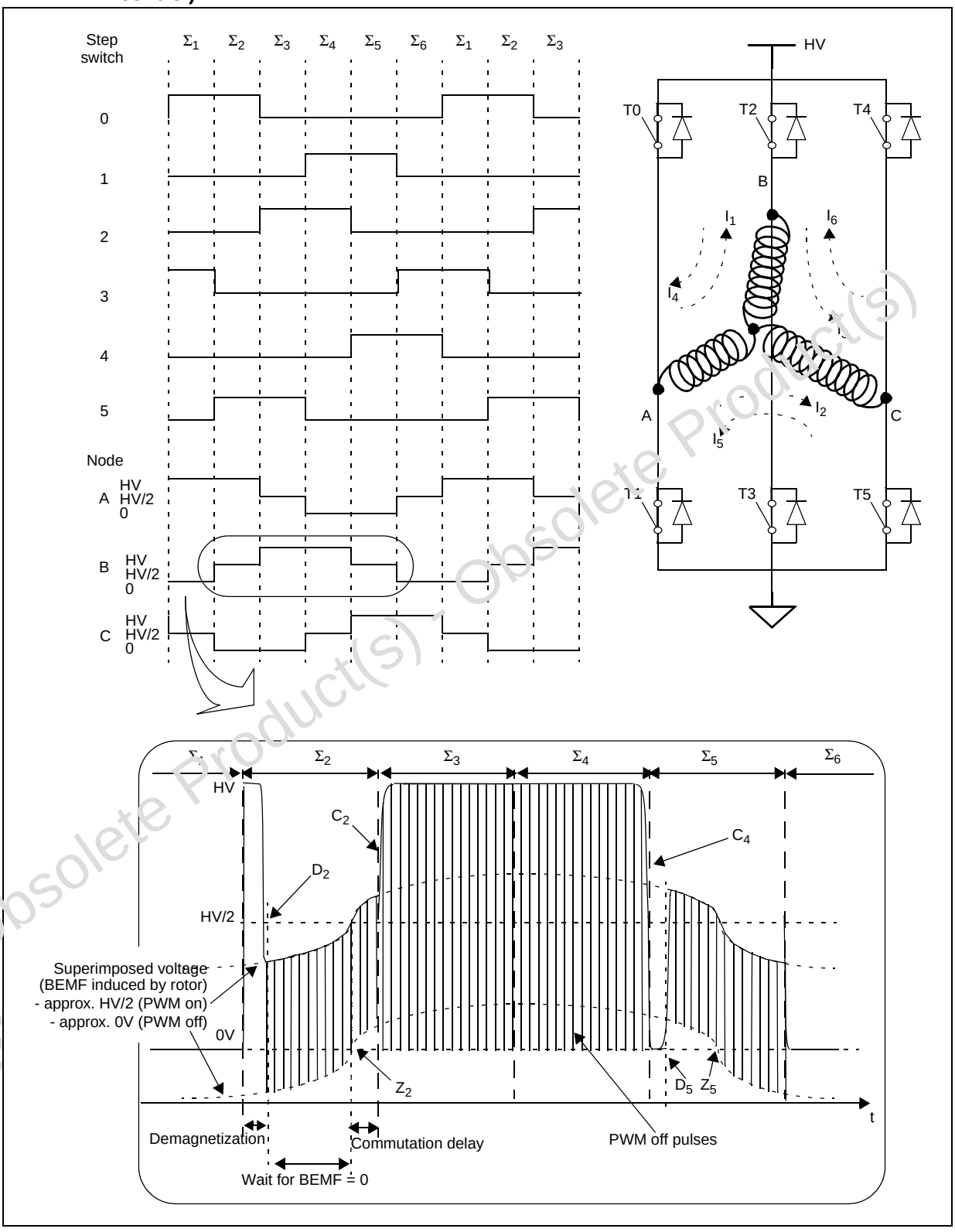


Figure 75. Example of command sequence for 6-step mode (typical 3-phase PM BLDC motor control)



1. Control and sampling PWM influence is not represented on these simplified chronograms.

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 82. 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: MTC registers*.

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 76 on page 187*). 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 application software, in charge of computing the compare values to be loaded for a given PWM duty-cycle (refer to *Figure 77*).

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 76. Complementary PWM generation for three-phase induction motor (1 phase represented)

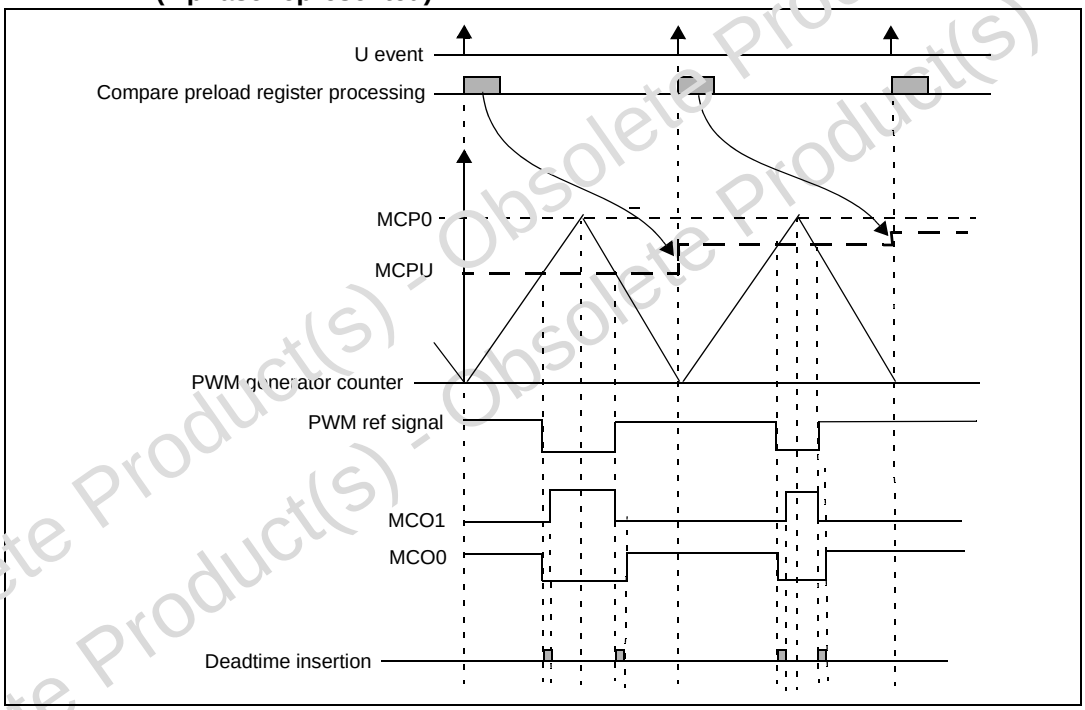
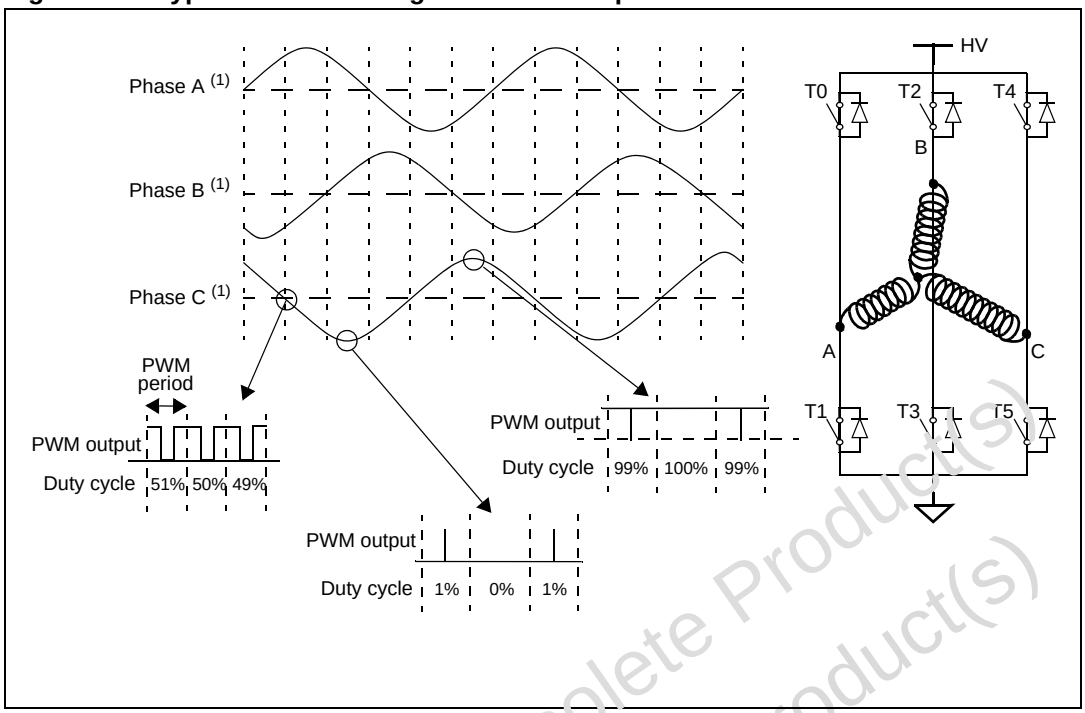


Figure 77. Typical command signals of a three-phase induction motor



1. These simplified chronograms represent the phase voltages after low-pass filtering of the PWM outputs reference signals.

10.6.5 Functional description

The MTC can be split into five main parts as shown in the simplified block diagram in *Figure 78*. 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:
  - an 8-bit delay manager
  - a 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 93* for set-up information). The block diagram is shown in *Figure 79* for the position sensor/sensorless modes (TES[1:0 = 00) and in *Figure 89* for the speed sensor mode (TES[1:0] = 01, 10, 11).

Input pins

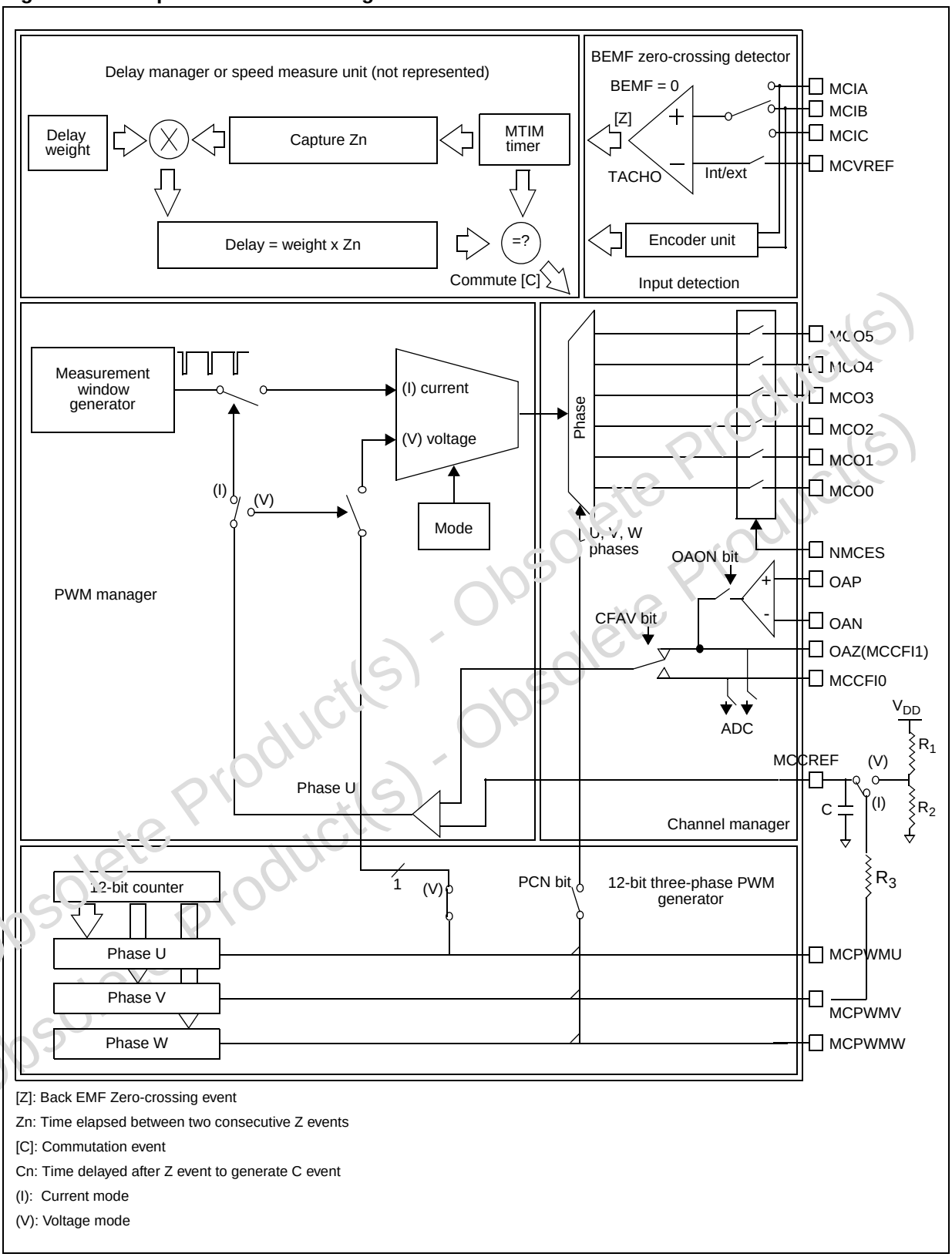
The MCIA, MCIB and MCIC input pins can be used as analog or as digital pins.

- 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 (example, 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.

Figure 78. Simplified MTC block diagram



[Z]: Back EMF Zero-crossing event

Zn: Time elapsed between two consecutive Z events

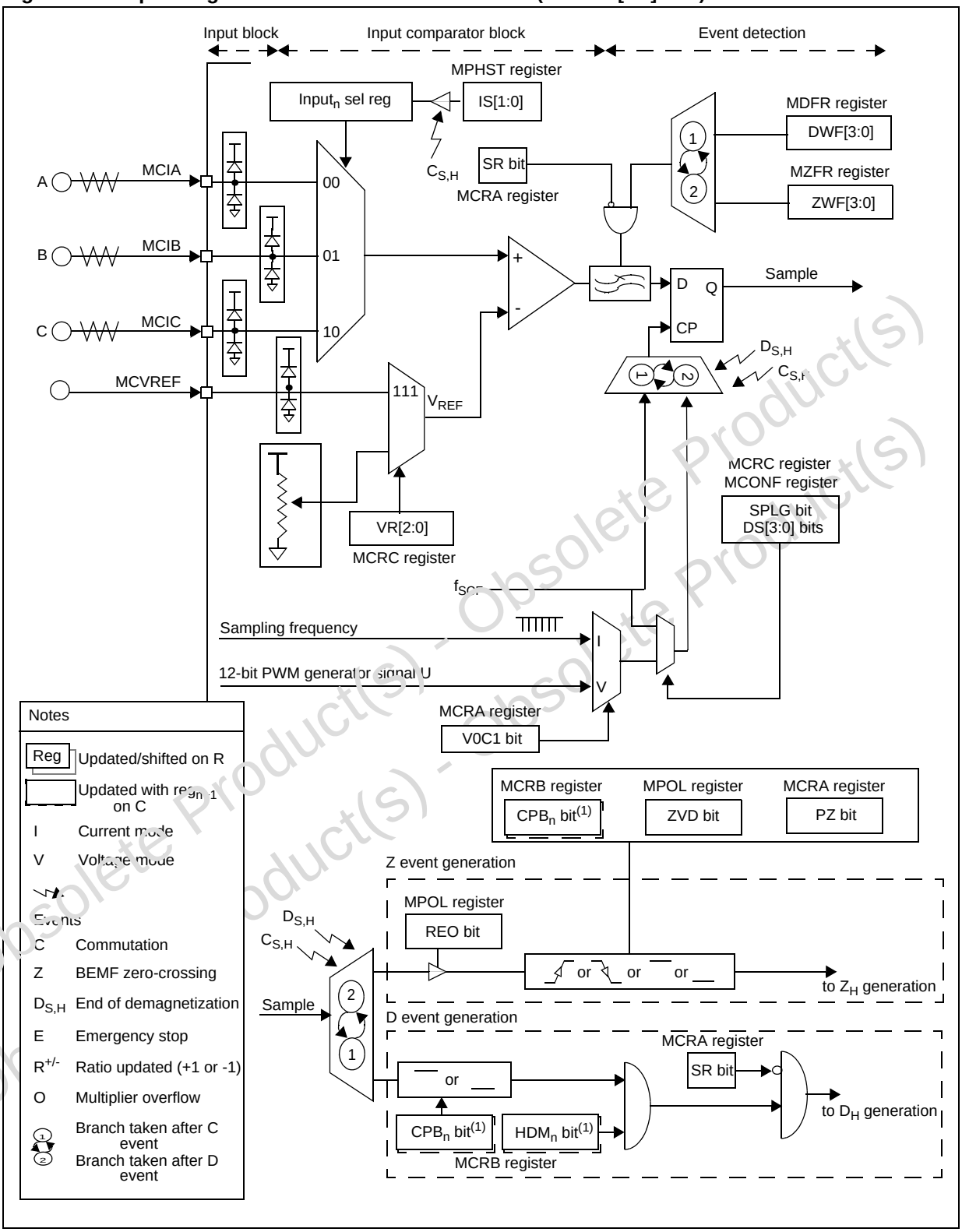
[C]: Commutation event

Cn: Time delayed after Z event to generate C event

(I): Current mode

(V): Voltage mode

Figure 79. Input stage in sensorless or sensor mode (bits TES[1:0] = 00)



1. Preload register; changes taken into account at next C event

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 the input to be selected which drives to the comparator (either MCIA, B or C). Be careful that the comparator is OFF until CKE and/or DAC bits 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 83. Threshold voltage setting

VR2	VR1	VR0	V_{REF} voltage threshold
1	**1**	**1**	**Threshold voltage set by external MCVREF pin typical value for V_{DD} = 5V**
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

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 is read is no longer excited but needs a demagnetization 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 166*).

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 88*.

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 80* and *Figure 81*.

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 84*). 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 163*.

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 80. D window and event filter flowchart

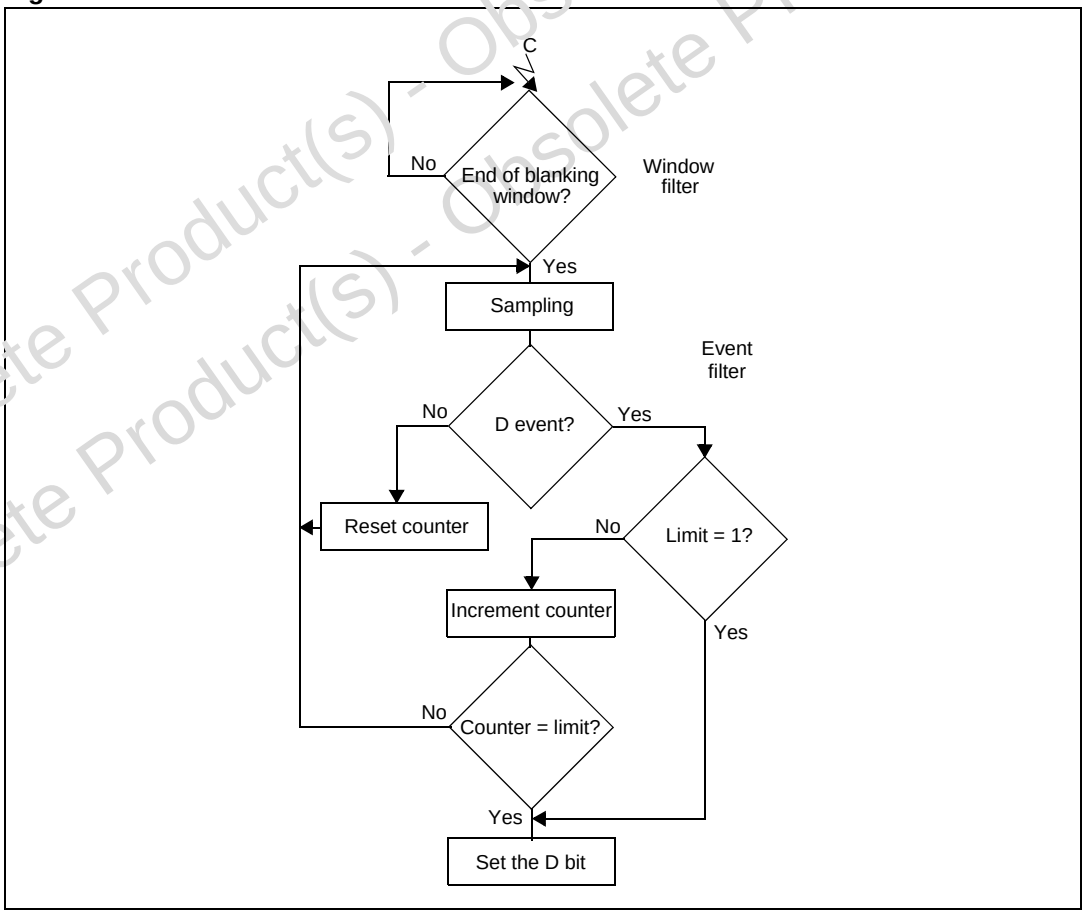


Table 84. D window filter setting(1)

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	

1. Times are indicated for 4 MHz f_{ERIH}.

Table 85. 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	

Table 85. D event filter setting (continued)

DEF3	DEF2	DEF1	DEF0	D event limit	SR = 1
1	1	1	0	15	No D event filter
1	1	1	1	16	

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 86*). 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 166*.

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 81. Z window and event filter flowchart

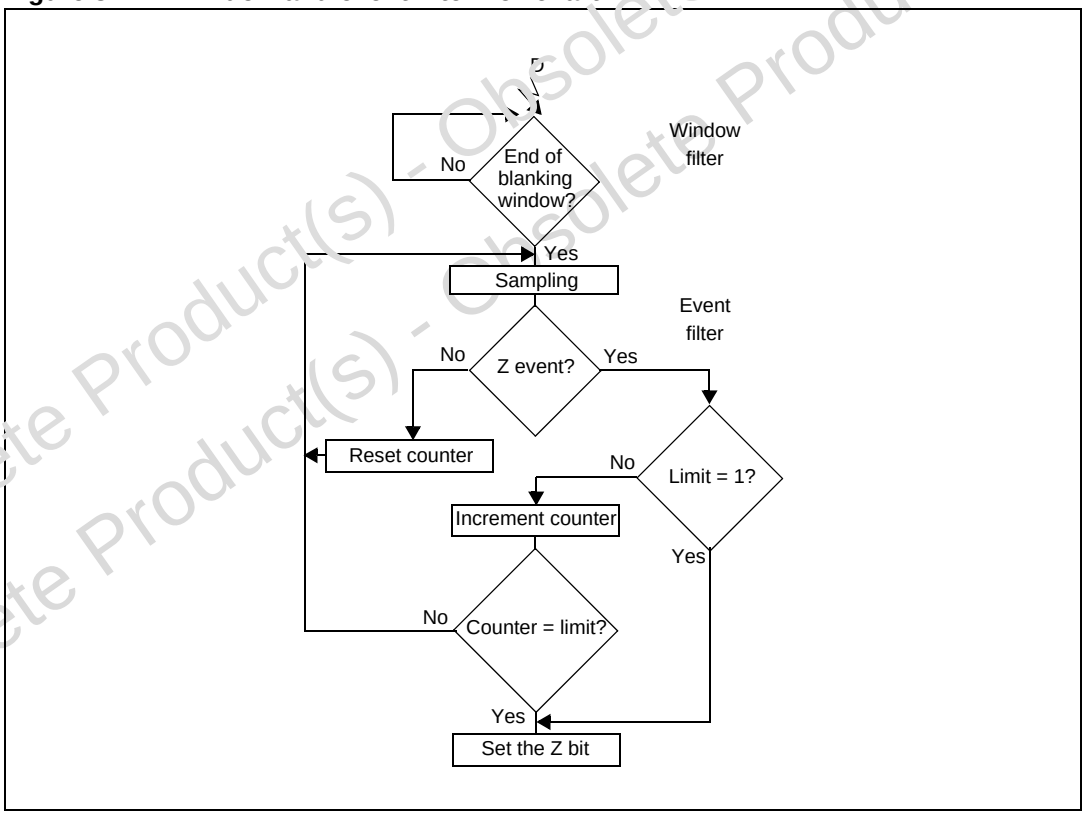


Table 86. Z window filter setting(1)

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	

1. Times are indicated for 4 MHz f_{ERCH}.

Table 87. Z event filter setting

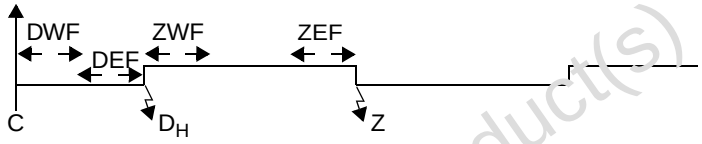
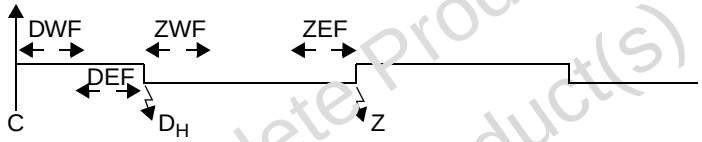
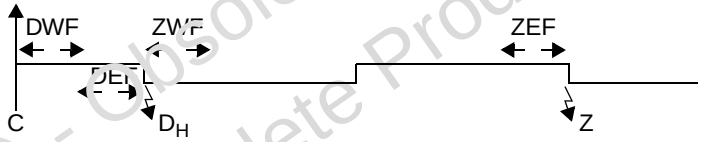
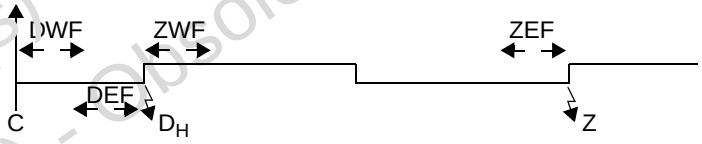
ZEF3	ZEF2	ZEF1	ZEF0	Z event limit
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14

Table 87. Z event filter setting (continued)

ZEF3	ZEF2	ZEF1	ZEF0	Z event limit
1	1	1	0	15
1	1	1	1	16

Table 88 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 88. ZVD and CPB edge selection bits(1)

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.		

1. Legend:
DW = D window filter
DEF = D event filter
ZWF = Z window filter
ZEF = Z event filter. Refer also to *Table 92 on page 207*.

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 88*.

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.

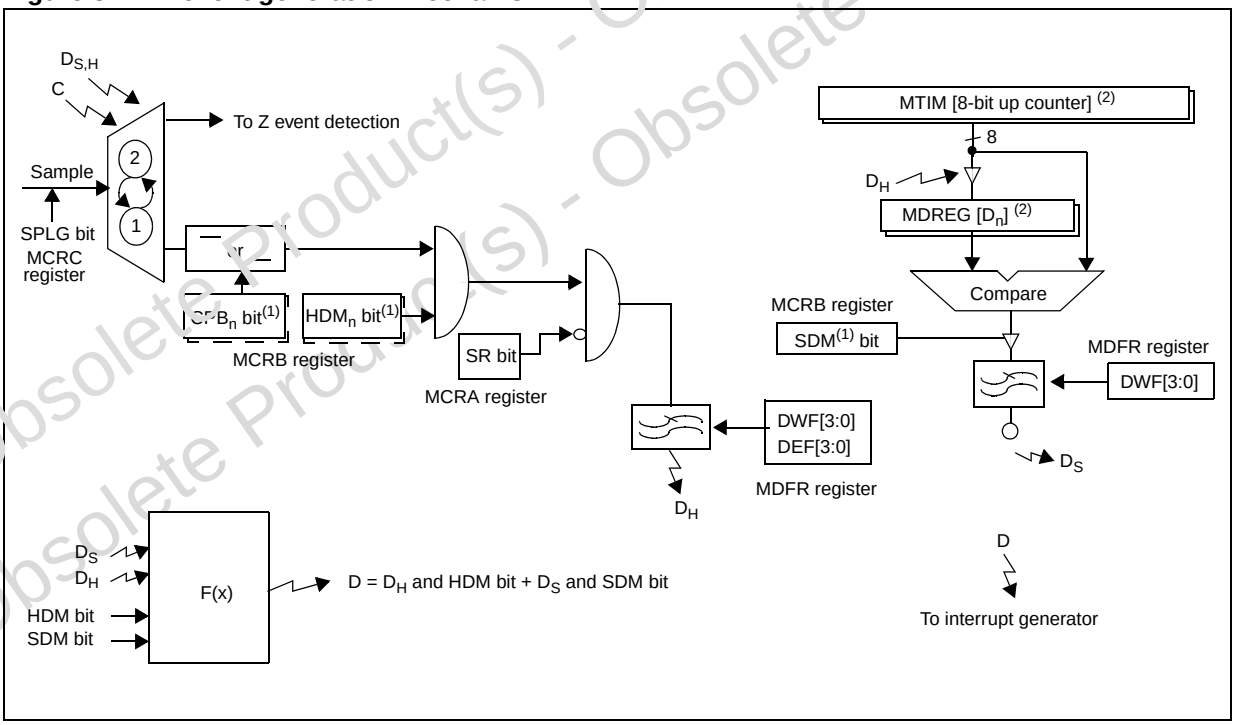
Simulated demagnetization can also be always used if the HDM bit is reset and the SDM bit is set. This mode works as a programmable masking time between the C_H and Z events. To drive the motor securely, the masking time must be always greater than the real demagnetization time in order to avoid a spurious 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.

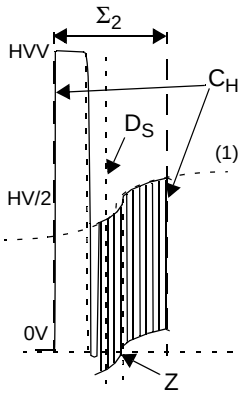
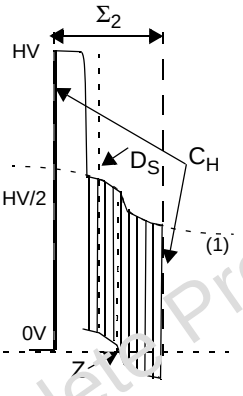
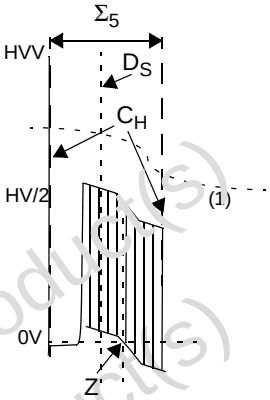
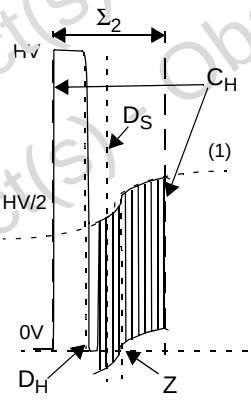
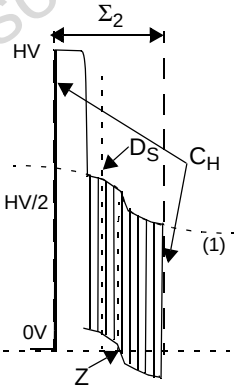
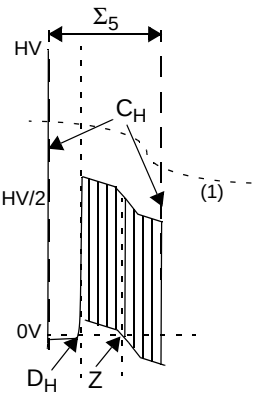
Caution: **2:** Due to the event generation protection in the MZREG, MCOMP and MDREG registers for Soft Event generation (see *Built-in checks and controls for simulated events on page 221*), the value written in the MDREG register in soft demagnetization mode (SDM = 1) is checked by hardware after the C event. If this value is less than or equal to the MTIM counter value at this moment, the Software demagnetization event is generated immediately and the MTIM current value overwrites the value in the MDREG register to be able to re-use the right demagnetization time for another simulated event generation.

Figure 82. D event generation mechanism



1. Preload bit, new value taken into account at the next C 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 *Control register A (MCRA) on page 265*. The use of a preload register allows all the registers to be updated at the same time
2. Register updated on R event

Table 89. Demagnetization (D) event generation (example for ZVD = 0)

HDM bit	Meaning	CPB bit = 1	CPB bit = 0
0	Simulated mode (SDM bit = 1 and HDM bit = 0)	D = D_S = output compare [MDREG, MTIM registers] Undershoot due to motor parasite or first sampling  Weak/null undershoot and BEMF positive  	
1	Hardware/simulated mode (SDM bit = 1 and HDM bit = 1)	D = D_H + D_S (hardware detection or output compare true) Undershoot due to motor parasite or first sampling  Weak/null undershoot and BEMF positive  	D = D_H (hardware detection only) 

1. This is a zoom to the additional voltage induced by the rotor (back EMF)

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 88*). 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 is done through the low side diode and the terminal is 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 (that is, 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 84*).

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 84 on page 202*

For this reason the MTC outputs can be split in two groups called LOW and HIGH and the BEMF reading is made 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.

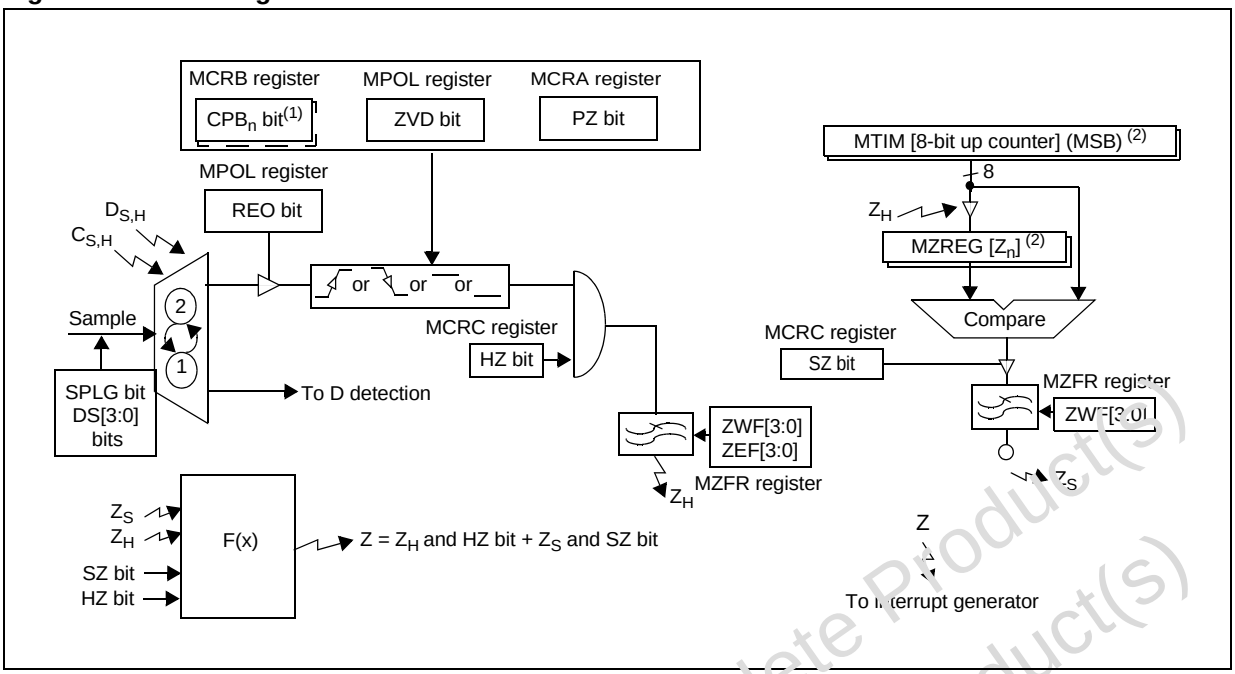
Depending on the edge and level selection (ZVD and CPB) bits and when PWM is applied on the correct group, a BEMF zero crossing detection (either Z_H or Z_S) sets the ZI bit in the MISR register and generates an interrupt if the ZIM bit is set in the MIMR register.

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 simulated 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 221*.

The Z event also triggers some timer/multiplier operations, for more details see *Section 10.6.7: Delay manager*.

Figure 83. Z event generation



1. Preload register; changes taken into account at next C event.
2. Register updated on R event.

Protection for Z_H event detection

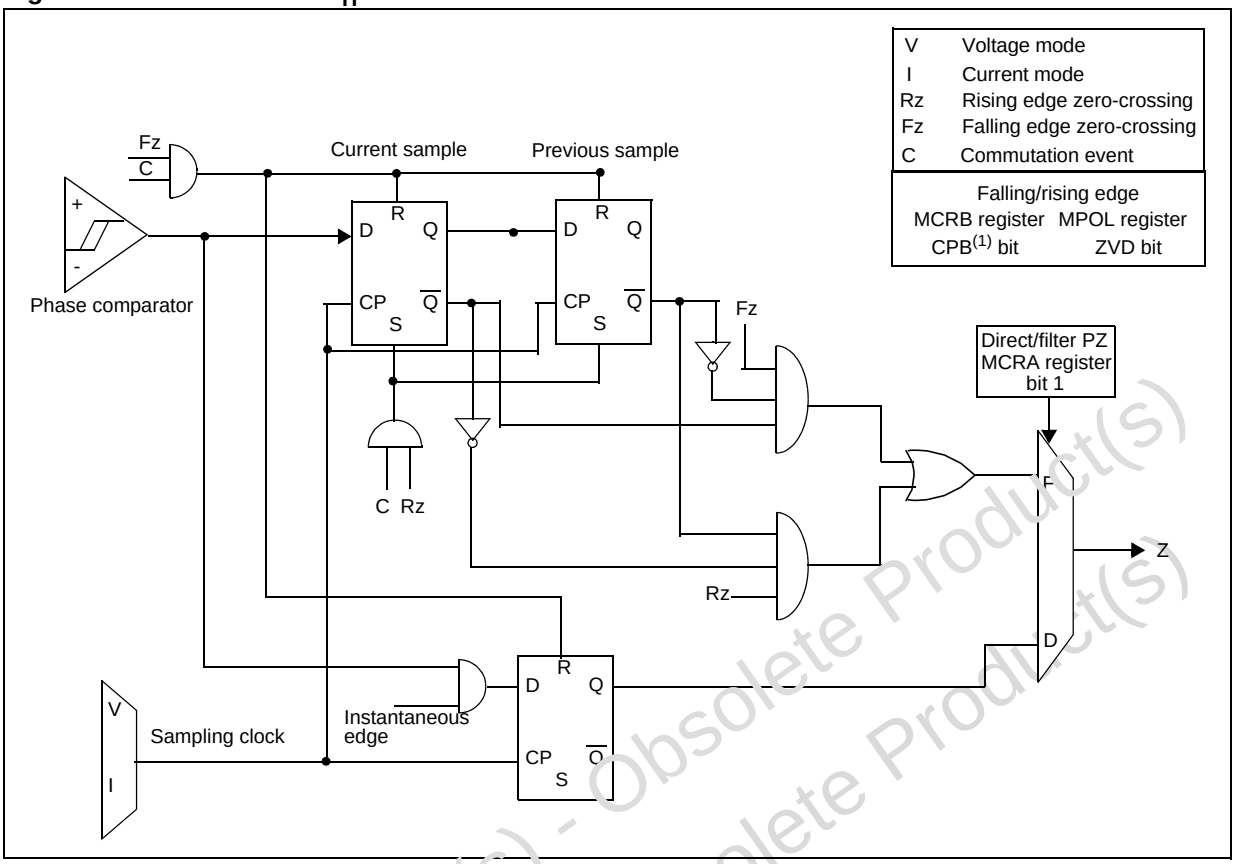
To avoid an erroneous detection of a hardware zero-crossing event, a filter can be enabled by setting the PZ bit in the MCRA register. This filter ensures the detection of a Z_H event on an edge transition between D event and Z_H event.

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 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 84. Protection of Z_H event detection



1. Preload bit, new value taken into account at the next C 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 *Control register A (MCRA) on page 265*. The use of a preload register allows all the registers to be updated at the same time

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 90*).

Table 90. Demagnetization access

SR bit MCRA register	Demagnetization feature availability
1	No
0	Yes

In sensor mode configuration the rotor detection does not 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 166*). 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 is enabled or disabled (depending on the CKE bit in the 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)

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 the 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 stabilization.

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 stabilization. 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 166*) 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 demagnetization) 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 91. Delay length before sampling(1)

DS3	DS2	DS1	DS0	Delay added to sample at T_{ON}
0	0	0	0	No delay added. Sample during T_{OFF}
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

1. 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.

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 is not taken into account. 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 85* and *Figure 86* show 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 is 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 87* shows in detail the sampling at f_{SCF} high frequency during ON time.

Figure 85. Adding the delay to sample during ON time for Z detection

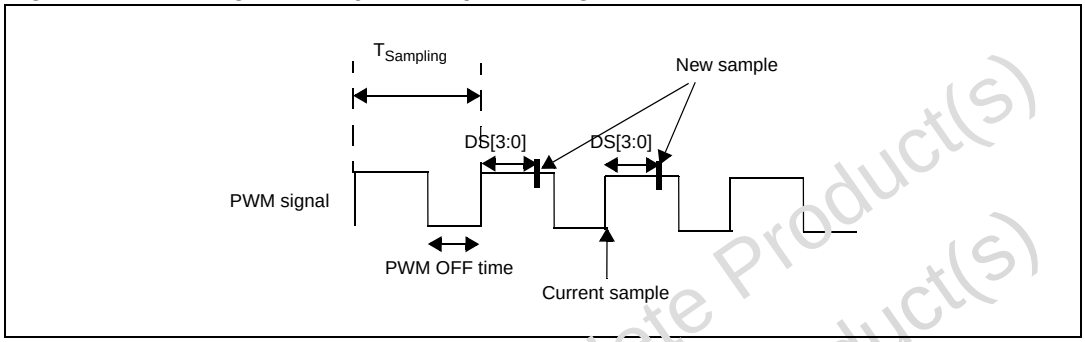
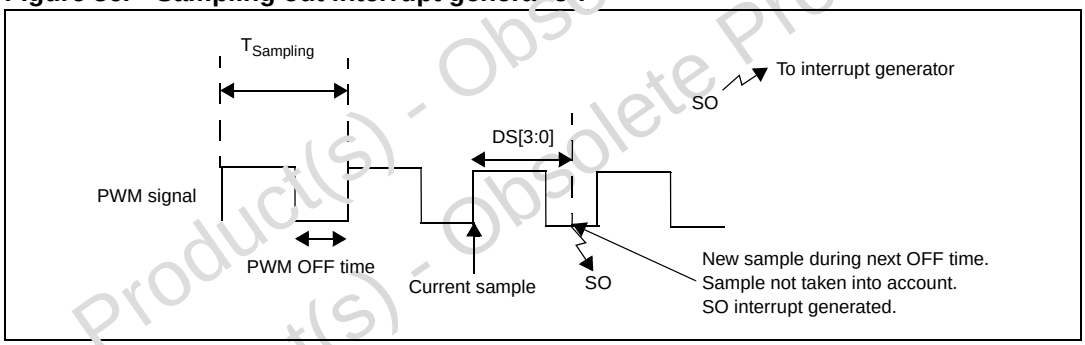


Figure 86. Sampling out interrupt generation



In conclusion, there are four 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 166*).
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 166*).

Note: 1 The sampling type is applied only for Z event detection after the D event has occurred. 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 166), independently of the PWM signal (either during ON or OFF time). Table 92 explains the different sampling types in sensorless and in sensor mode.

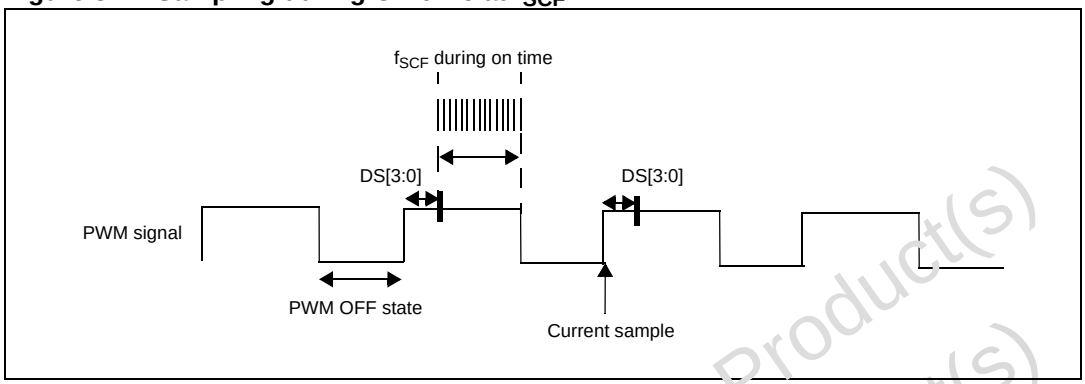
2 When the MOE bit in the MCRA register is reset (MCOx outputs in reset state), and the SR bit in the MCRA register is reset (sensorless mode) and the SPLG bit in the MCRC register

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).

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.*

4 *The ZEF[3:0] event counter in the MZFR register is active in all configurations.*

Figure 87. Sampling during ON time at f_{SCF}



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 203*). To avoid any erroneous detection due to PWM commutation noise, an hardware filter of 1µs (for f_{PERIPH} = 4 MHz) 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.

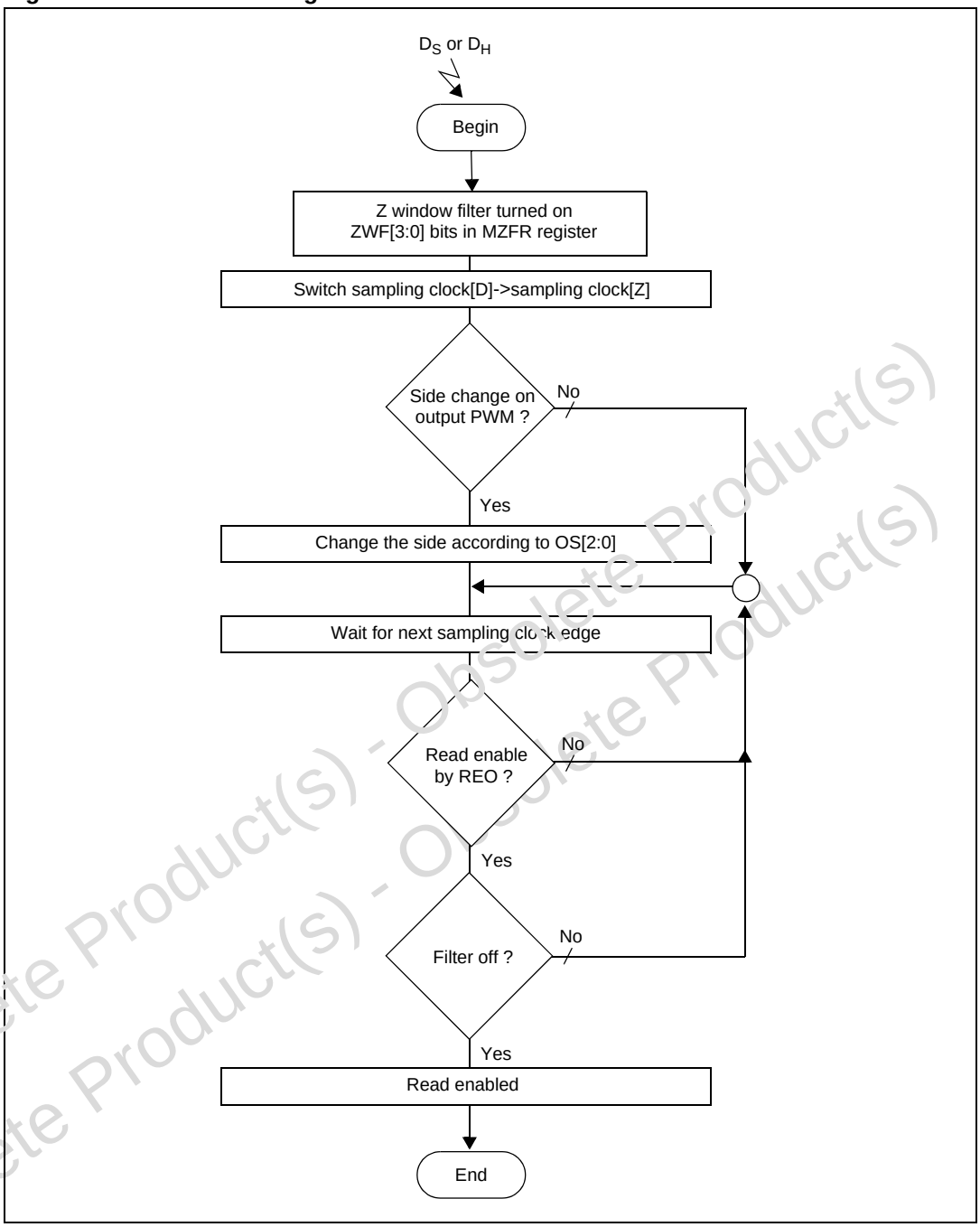
This filter is active all the time for the D event and it is active for the Z event when the SPLG bit is set and DS[3:0] bits are cleared (meaning that the Z event is sampled at high frequency during the PWM ON or OFF time).

Table 92. Sensor/sensorless mode and D and Z event selection(1)

SR bit	SPLG bit	DS[3:0] bits	Mode	OS[2:0 bits use	Event detection sampling clock	Sampling behavior for Z event detection	Window and event filters	Behavior of the output PWM
0	0	000	Sensors not used	Enabled	D: f_{SCF} Z: SA and 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 88 on page 197*	'Before D' behavior, 'between D and Z' behavior and 'after Z' behavior
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' behavior, 'between D and Z' behavior and 'after Z' behavior
0	0	Not equal to 000	Sensors not used	Enabled	D: f_{SCF} Z: SA and OT config. PWM frequency	During ON time of the PWM signal		'Before D' behavior, 'between D and Z' behavior and 'after Z' behavior
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' behavior, 'between D and Z' behavior and 'after Z' behavior
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' behavior and 'after Z' behavior

1. For f_{SCF} selection, see *Table 166*

Figure 88. Functional diagram of Z detection after D event



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 89*.

Either incremental encoder or tachogenerator-type speed sensor can be selected with the IS[1:0] bits in the MPHST register.

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 91 presents the signals generated internally with different tacho input and TES bit settings.

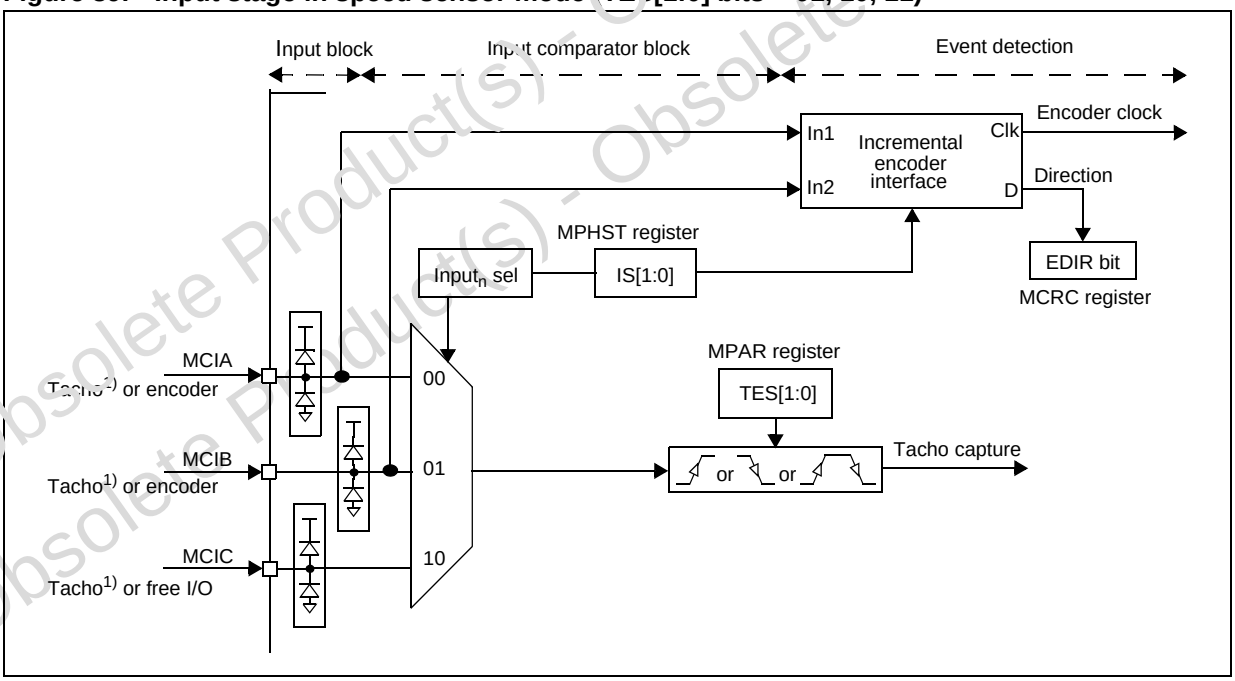
Hall sensors

This configuration is also suitable for motors using three hall sensors for position detection and not driven in six-step mode (refer to *Speed measurement mode on page 227*).

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 89. Input stage in speed sensor mode (TES[1:0] bits = 01, 10, 11)



1. According to IS[1:0] bits setting.

Encoder mode (IS[1:0] = 11)

Figure 91 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 (+/-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 not affect counting direction of clocked timer (see *Section 10.6.7: Delay manager*). As a result, one cannot extract position information from encoder inputs during speed reversal.

Figure 90. Tacho capture events configured by the TES[1:0] bits

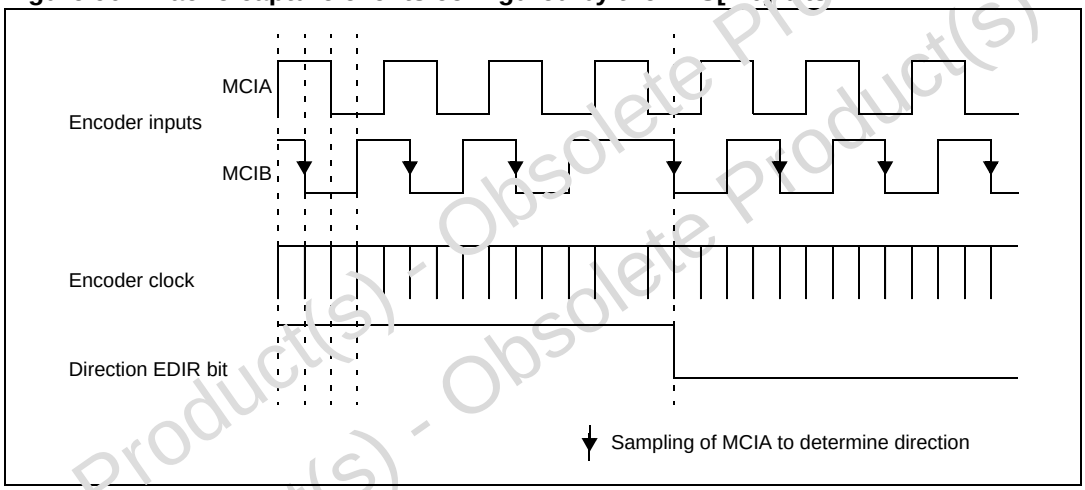
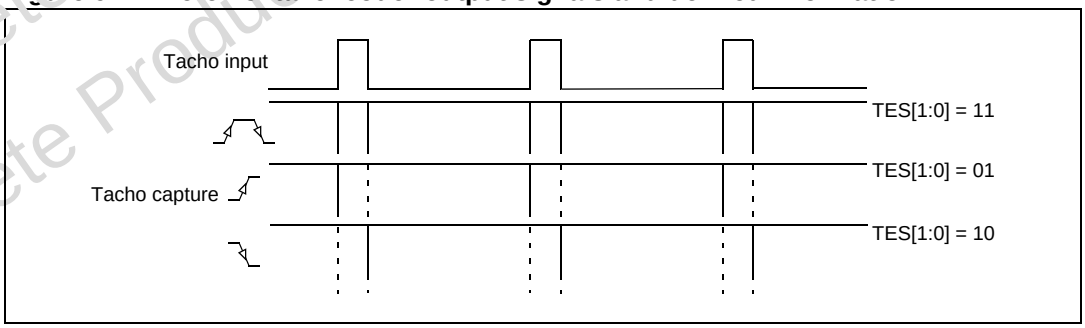


Figure 91. Incremental encoder output signals and derived information



Note: If only one encoder output is available, it may be input either on MCIA or MCIB and an encoder clock signal is still generated (in this case the frequency is 50% less than with two inputs.

The state of EDIR bit depends on signals present on MCIA and MCIB pins, the result is given by sampling the falling edges of MCIA with MCIB.

Summary

The input detection block set-up for the different available modes is summarized in *Table 93*.

Table 93. 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	-	-	0	00	00 01 10
Speed sensor	Incremental encoder	Both rising and falling edges (imposed)	^	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

Note on using the three 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 94 on page 212*.

Table 94. MCIx pin configuration summary(1)

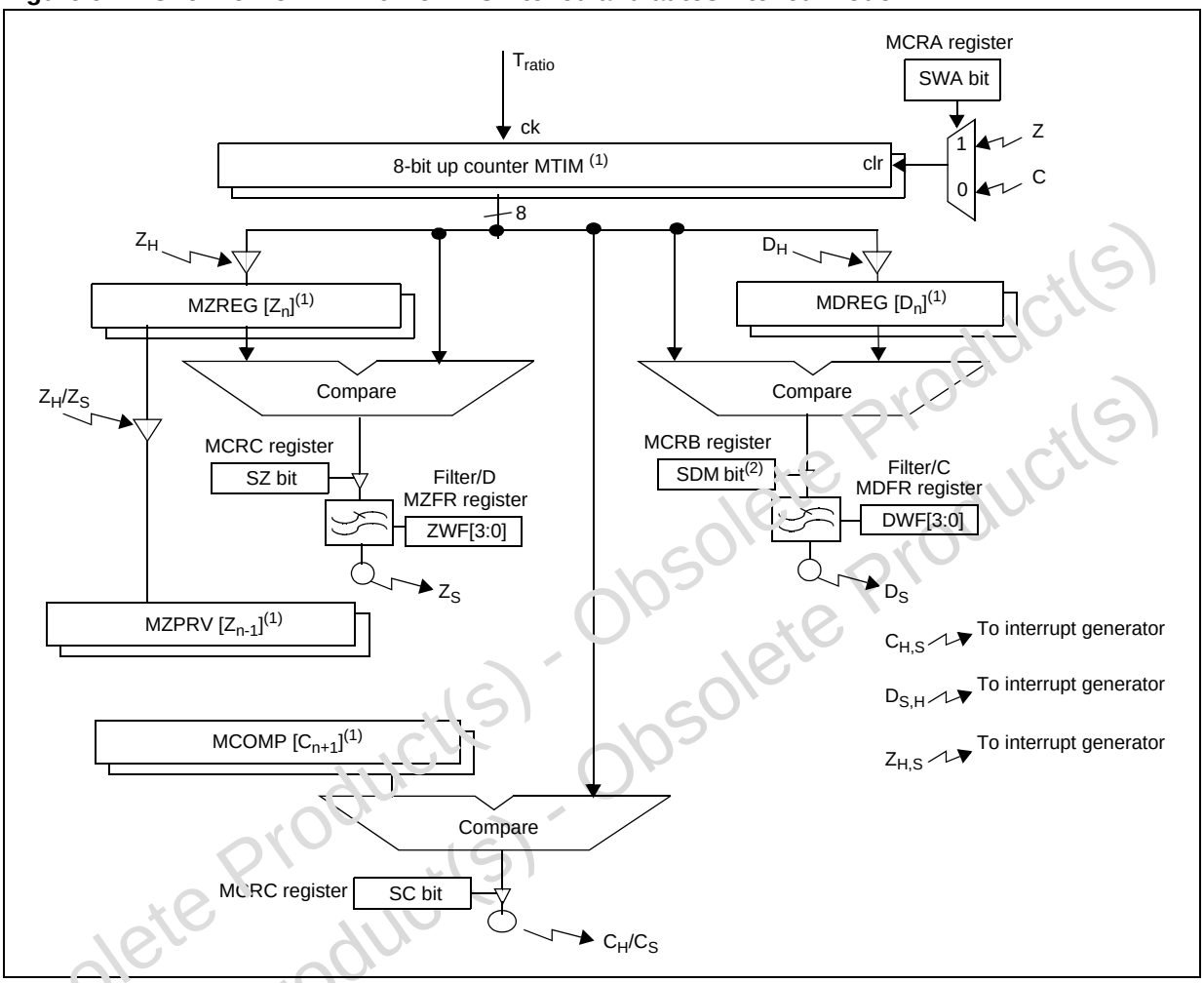
PCN	TES	SR	IS[1:0]	MCIA	MCIB	MCIC	Input detection block mode	Comments
0	00	0	00	Analog input(2)	Hi-Z or GND	Hi-Z or GND	Sensorless	All MCIx pins are reserved for the MTC peripheral.
			01	Hi-Z or GND	Analog input(2)	Hi-Z or GND		
			10	Hi-Z or GND	Hi-Z or GND	Analog input(2)		
			11	NA	NA	NA	NA	
		1	00	Analog input(2)	Standard I/O	Standard I/O	Position sensor	From 1 to 3 MCIx pins reserved depending on sensor
			01	Standard I/O	Analog input(2)	Standard I/O		
			10	Standard I/O	Standard I/O	Analog input(2)		
			11	Standard I/O	Standard I/O	Standard I/O	-	All MCIx pins are standard I/Os. Phase comparator is OFF
	≠ 0	x	xx	NA	NA	NA	NA	
1	00	x	00	Analog input(2)	Standard I/O	Standard I/O	-	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(2)	Standard I/O		
			10	Standard I/O	Standard I/O	Analog input(2)		
			11	Standard I/O	Standard I/O	Standard I/O	-	All MCIx pins are standard I/Os. Recommended configuration: phase comparator OFF
	≠ 00	x	00	Digital input(3)	Standard I/O	Standard I/O	Speed sensor tachogenerator	Phase comparator is OFF
			01	Standard I/O	Digital input(3)	Standard I/O		
			10	Standard I/O	Standard I/O	Digital input(3)		
			11	Digital input(3)	Digital input(3)	Standard I/O	Speed sensor encoder	

1. 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.
2. 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.
3. Digital input: use of standard V_{IL}, V_{IH} I/O level. This input can also be read via the associated I/O port.

10.6.7 Delay manager

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 92*.

Figure 92. Overview of MTIM timer in switched and autoswitched mode



1. Register updated on R event.
2. Preload bits, new value taken into account at the next C 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 *Control register A (MCRA) on page 265*. The use of a preload register allows all the registers to be updated at the same time.

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
- 1 compare register for phase commutation generation (MCOMP)

The MTIM timer module can work in two main modes when driving synchronous motors in six-step 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 (*Built-in checks and controls for simulated events on page 221*).

Both in switched mode and autoswitched mode, if the SC bit in the MCRC register is set (software commutation enabled), no comparison between

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 occurs.

In speed measurement mode, when using encoder or tachogenerator speed sensors (that is, 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 227*.

Table 95. Switched and autoswitched modes

SWA bit	Commutation type	MCOMP user access
0	Switched mode	Read/Write
1	Autoswitched mode	

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 (see *Section 10.6.13: MTC registers on page 260*). 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 is updated but also all the MTIM timer-related registers are 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 96*.

Only one update per step is allowed, so if both RPI and RMI bits are set simultaneously by software, there is no effect on the MISR register: the write access to these two bits together is not taken into account and the previous state is upheld. This means that if either RPI or RMI bits were set before the write access of both bits at the same time, this bit (RPI or RMI) is kept at 1. If neither bit was set before the simultaneous write access, neither of them are 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.

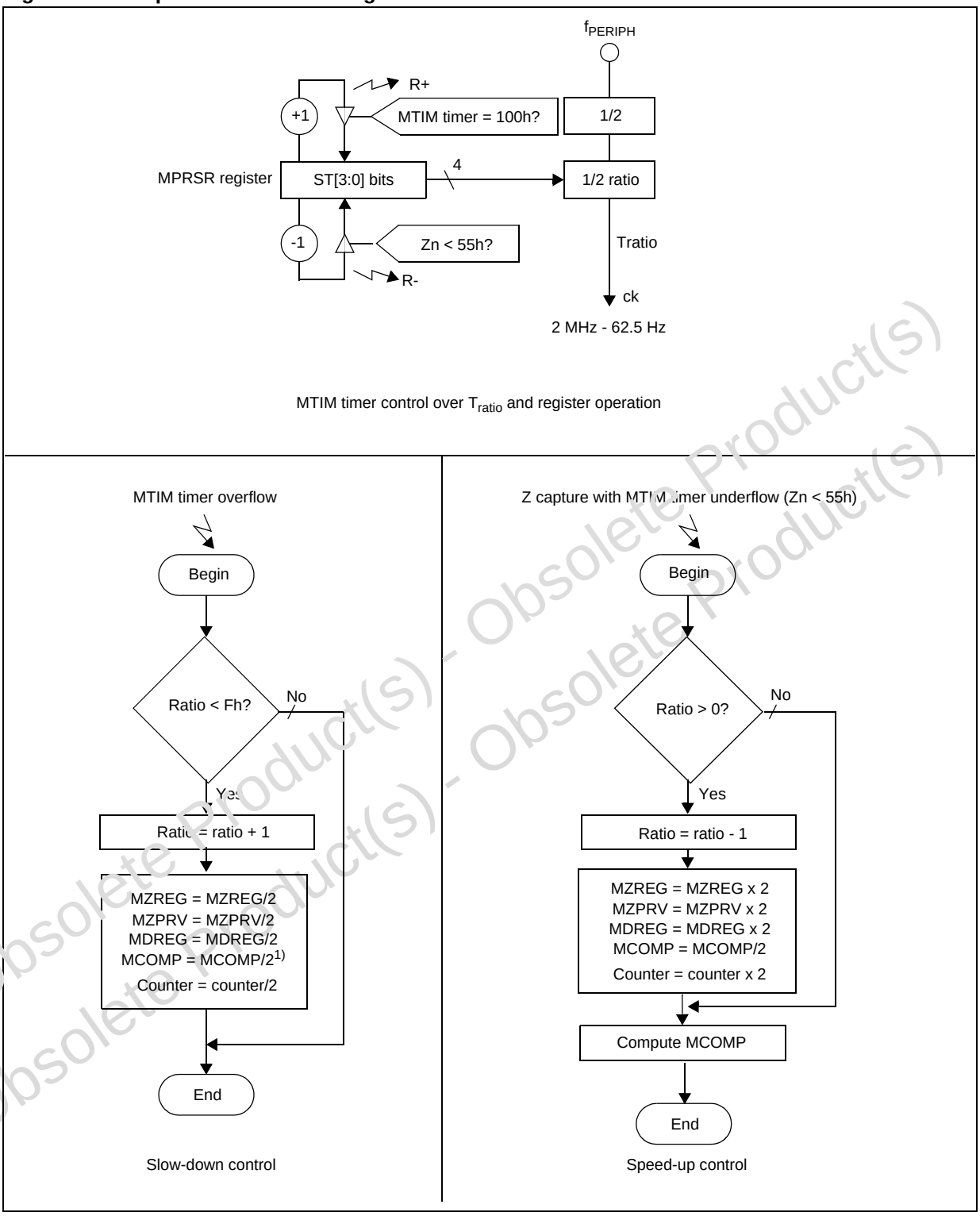
In this mode, if an MTIM overflow occurs, it restarts counting from 0x00h and the OI overflow flag in the MCRC register is set if the TES[1:0] bits = 00.

Caution: In this mode, MCOMP must never be written to 0.

Table 96. 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 until next commutation	Set RMI bit in the MISR register until next commutation
Autoswitched	00	1	1	Enabled		Automatically updated according to MZREG value	
Speed measure	01 10 11	1	x	Enabled			

Figure 93. Step ratio functional diagram



1. Only in auto-switched mode (SWA = 1 in MCRA register).

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 94*.

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 74*.

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 97. Multiplier result

DCB bit	Commutation delay
0	MCOMP = MWGHT x MZPRV/256
1	MCOMP = MWGHT x MZREG/256

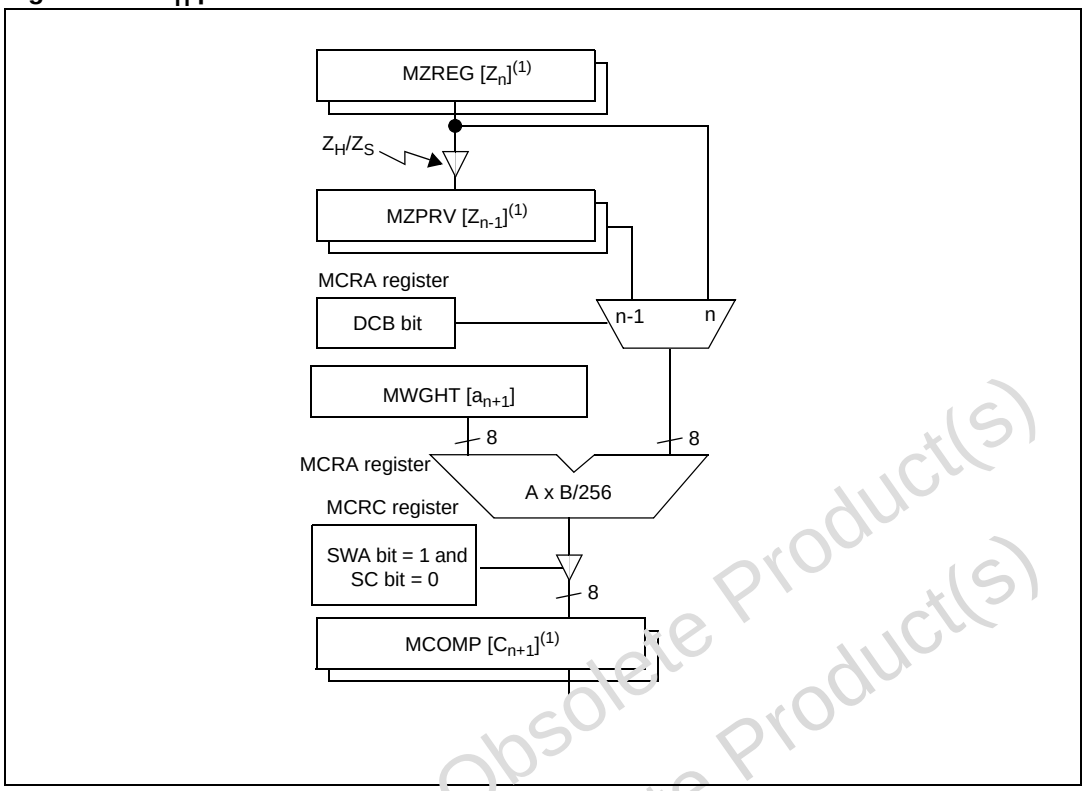
After each shift operation the multiply is recomputed for greater precision.

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 is 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 never occurs because no comparison is made 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 accepts consecutive commutation events and does 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 94. C_H processor block



1. Register updated on R event.

Note: 1 An overflow of the MTIM timer generates an RPI interrupt if the RIM bit is set.

2 When simulated commutation mode is enabled, the D and Z events are not ignored by the peripheral; this means that if a Z event happens, the MTIM 8-bit internal counter is reset.

3 To generate consecutive simulated commutations (C_S), the successive value has to be written in the MCOMP register only after a C event generation. Otherwise, the C event never occurs.

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

- In switched mode: the MTIM timer is driven by software only and any prescaler change has to be done by software (see *Switched mode on page 214* for more details).
- In autoswitched mode: an auto-updated prescaler always configures the MTIM timer for best accuracy. *Figure 93* illustrates 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 0 x 80h 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, that is, with a lower accuracy. No RMI interrupt 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 0 x 00h 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 99*.

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 95* 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.

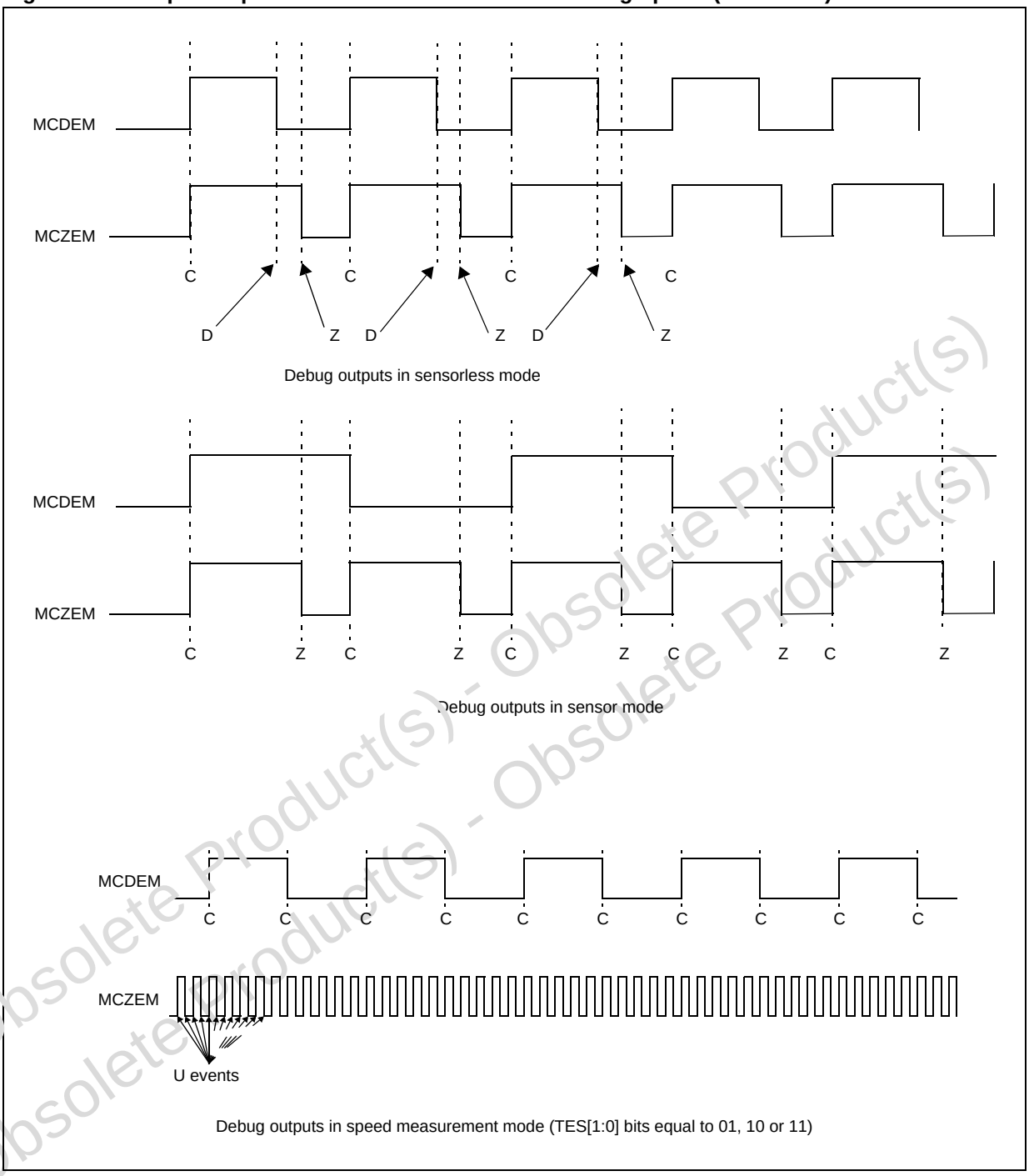
This option is also available in speed measurement mode with different signal outputs (see Figure 95):

- 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.

2 In sensor mode, the MCDEM output pin toggles at each C event. The MCZEM pin outputs the Z event.

Figure 95. Output on pins MCDEM and MCZEM with debug option (DG bit = 1)



Note: ***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 + demagnetization_time > FFh), the value that is stored in MDREG is: 80h + (MCOMP + demagnetization_time-FFh)/2.

Note: ***Commutation interrupts****: It is good practice to modify the configuration for the next step as soon as possible, that is, 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: ***Initializing the MTC****: As shown in Table 99 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.

Built-in checks and controls for simulated events

As described in Figure 92 on page 213, 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 demagnetization
  - SDM bit for simulated demagnetization.
- In the MCRC register for simulated zero-crossing and commutation
  - SC bit for simulated commutation.
  - SZ bit for simulated zero-crossing event.

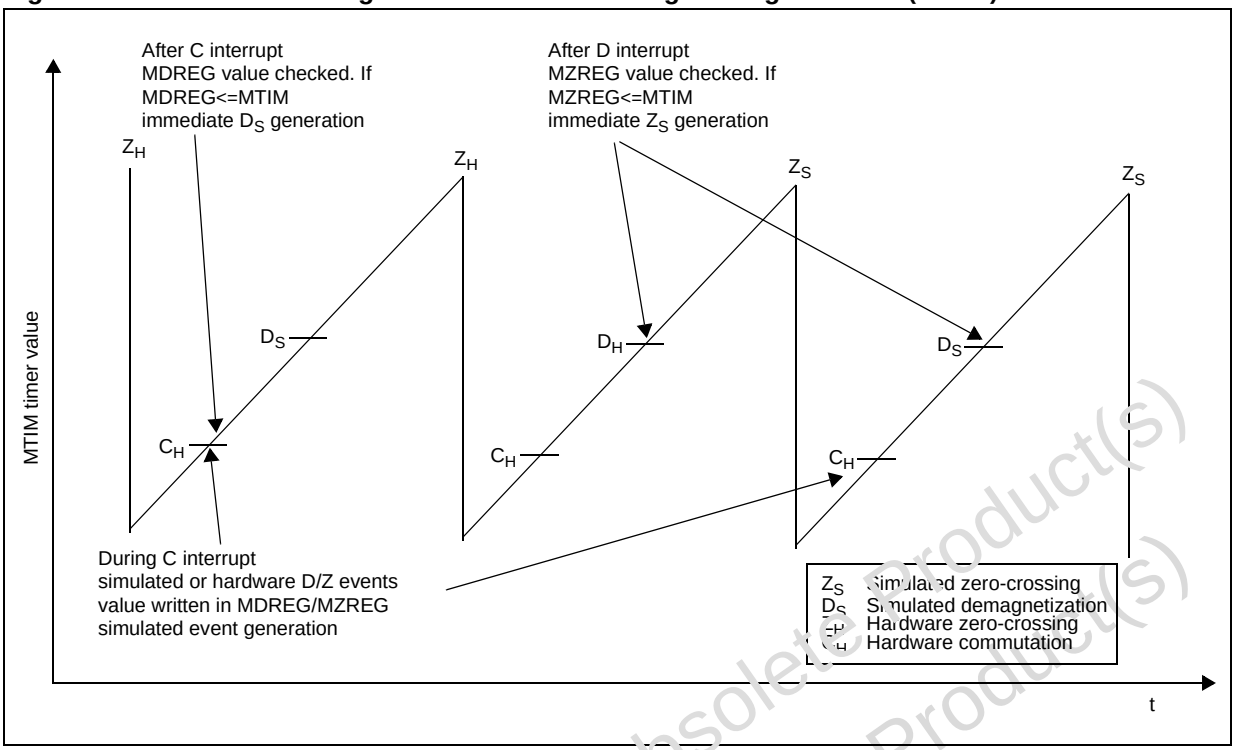
To avoid a system stop, special attention is needed when writing in the register to generate the corresponding simulated event. The value written in the register has to be greater than the current value of the MTIM timer when writing in the registers. If the value written in the registers (MDREG, MZREG or MCOMP) is already less than the current value of MTIM, the simulated event is never generated and the system is 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 96. Simulated demagnetization/zero-crossing event generation (SC = 0)



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 demagnetization, a value must be written in the MDREG register to generate the simulated demagnetization. This value must be written after the C (either C_S or C_H) event preceding the simulated demagnetization.

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 respected.

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 is 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

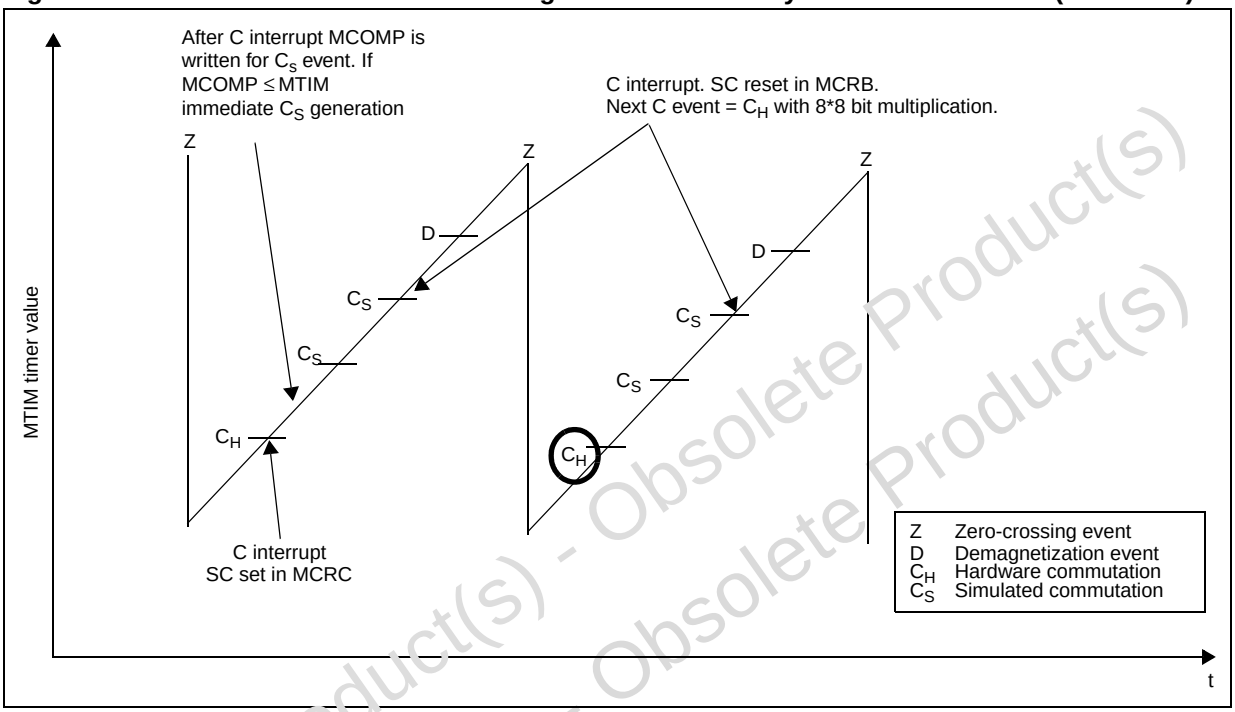
recommended to write the MCOMP register during the Z interrupt routine to avoid any spurious comparison as several consecutive C_S events can be generated.

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 96, *Figure 97* and *Figure 98* shows details of simulated event generation.

Figure 97. Simulated commutation event generation with only 1 hall effect sensor (SC bit = 1)



1. 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 98. Simulated commutation and Z event

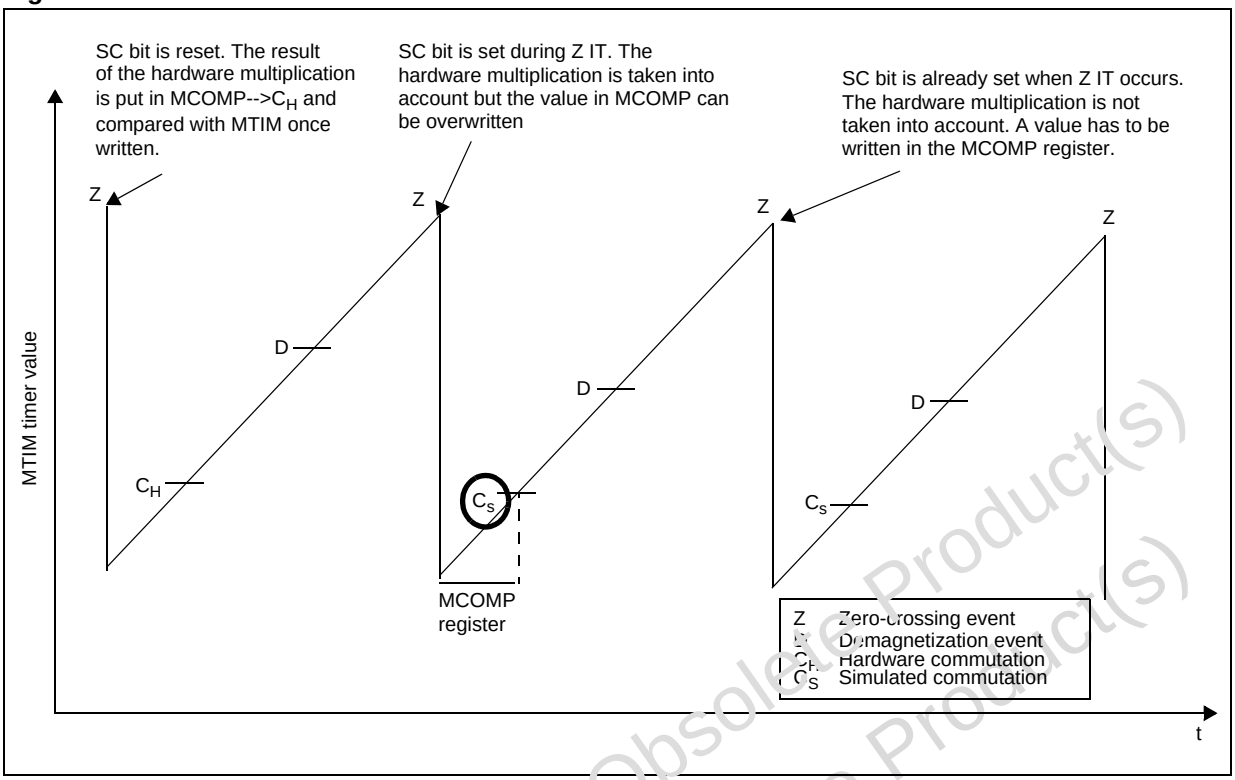


Figure 99 gives the step ratio register value (left axis) and the number of BEMF sampling during one electrical step with the corresponding accuracy on the measure (right axis) as a function of the mechanical frequency.

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} = 4 MHz, 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 = 15000 and a sampling frequency of 15 kHz, there are approximately 10 samples in one step and there is a 10% error rate on the measurement.

Figure 99. Step ratio bits decoding and accuracy results and BEMF sampling rate

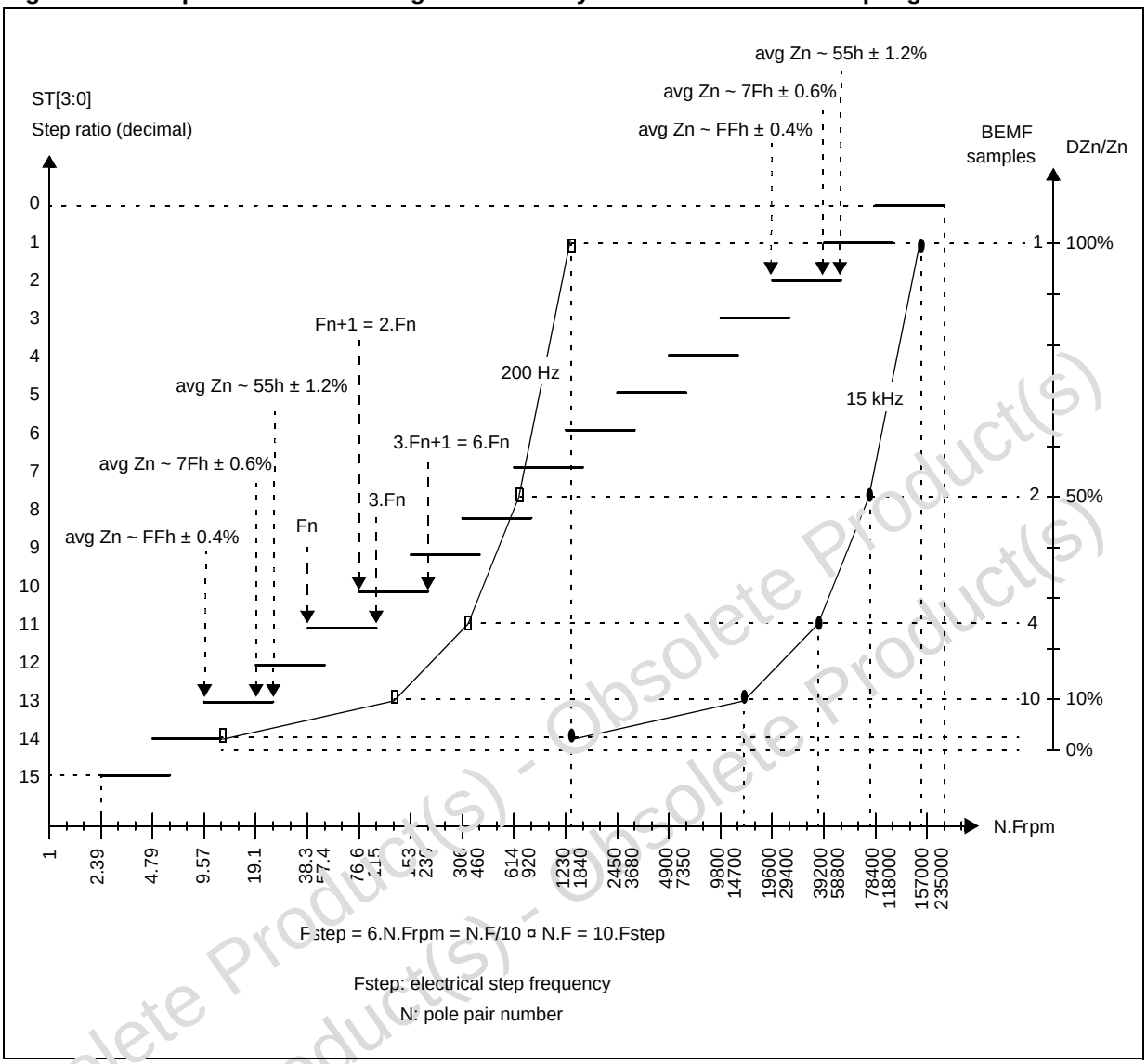


Table 98. Step frequency/period range (4 MHz)

Step ratio bits ST[3:0] in MPRSR register	Step frequency		Step period	
	Maximum	Minimum	Minimum	Maximum
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 99. modes of accessing main 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.

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 101*).

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 (that is, relevant transitions of the incoming signals). Accuracy is imposed by the reference clock, that is, the CPU clock (refer to *Figure 100*).

Figure 100. Tachogenerator period acquisition using MTIM timer

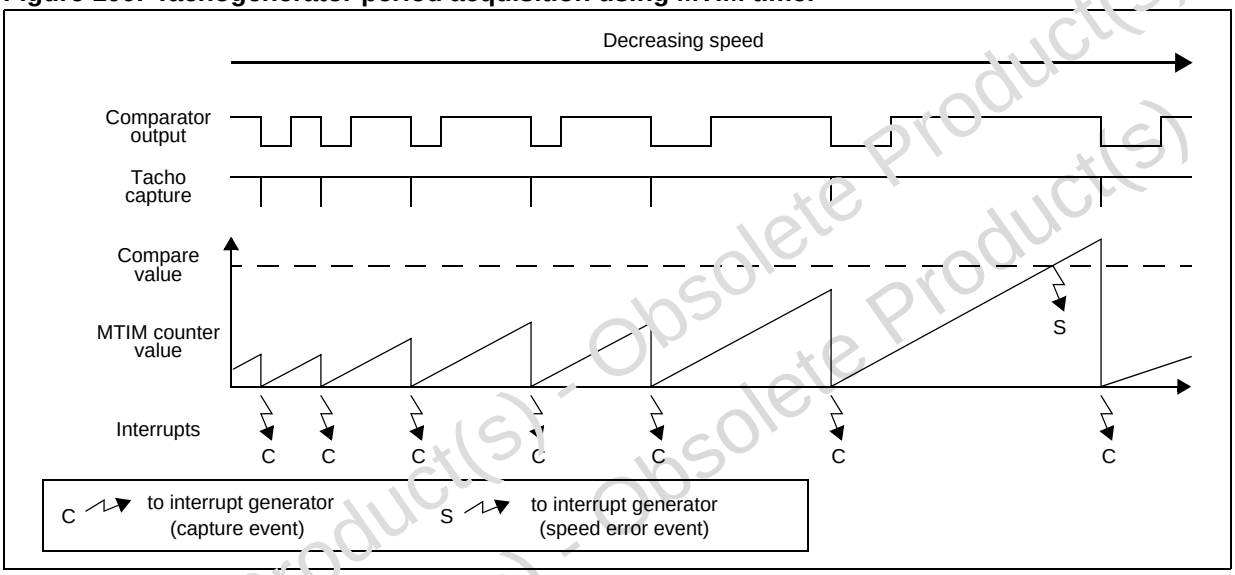
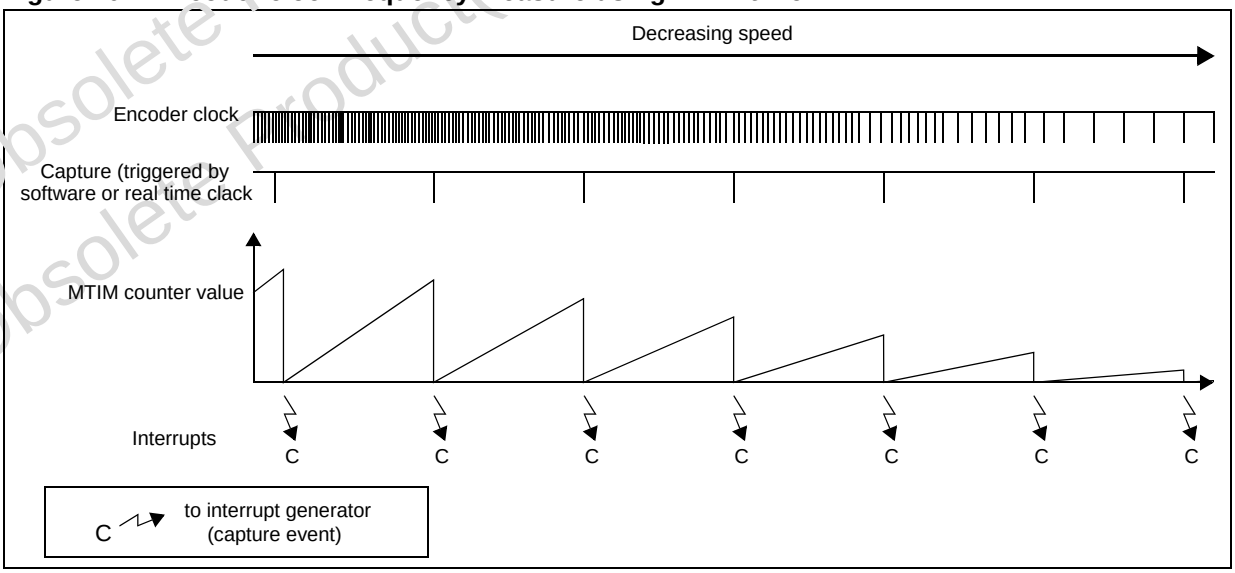


Figure 101. Encoder clock frequency measure using MTIM timer



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 three sensors toggled; an interrupt is generated just after the expected transition (refer to *Figure 102*).

As described in *Figure 103*, the MTIM timer is re-configured 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 is 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 (that is, 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 (that is, 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 102. Hall sensor period acquisition using MTIM timer

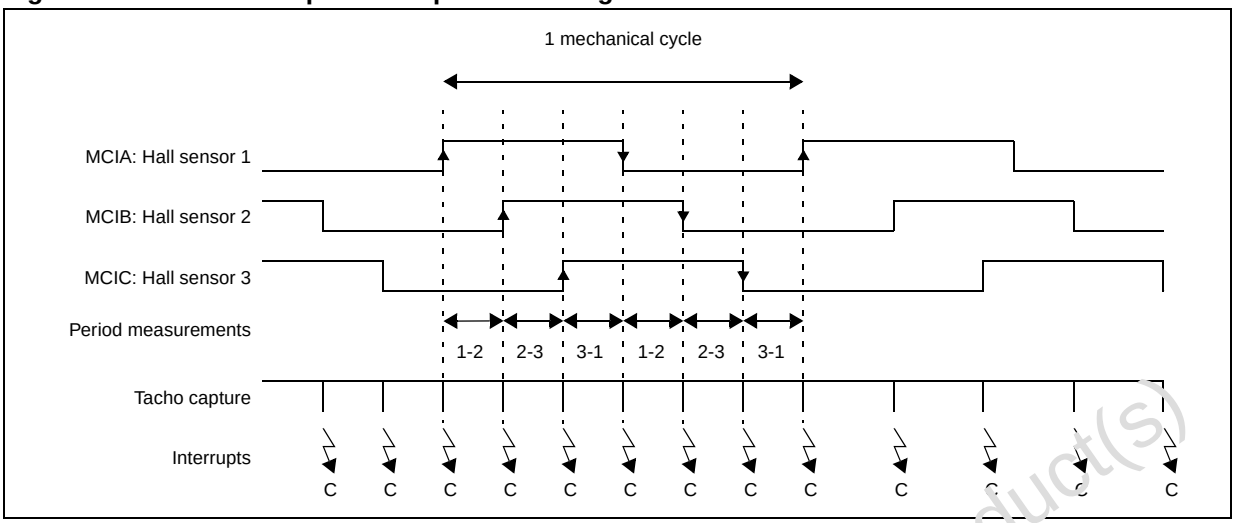
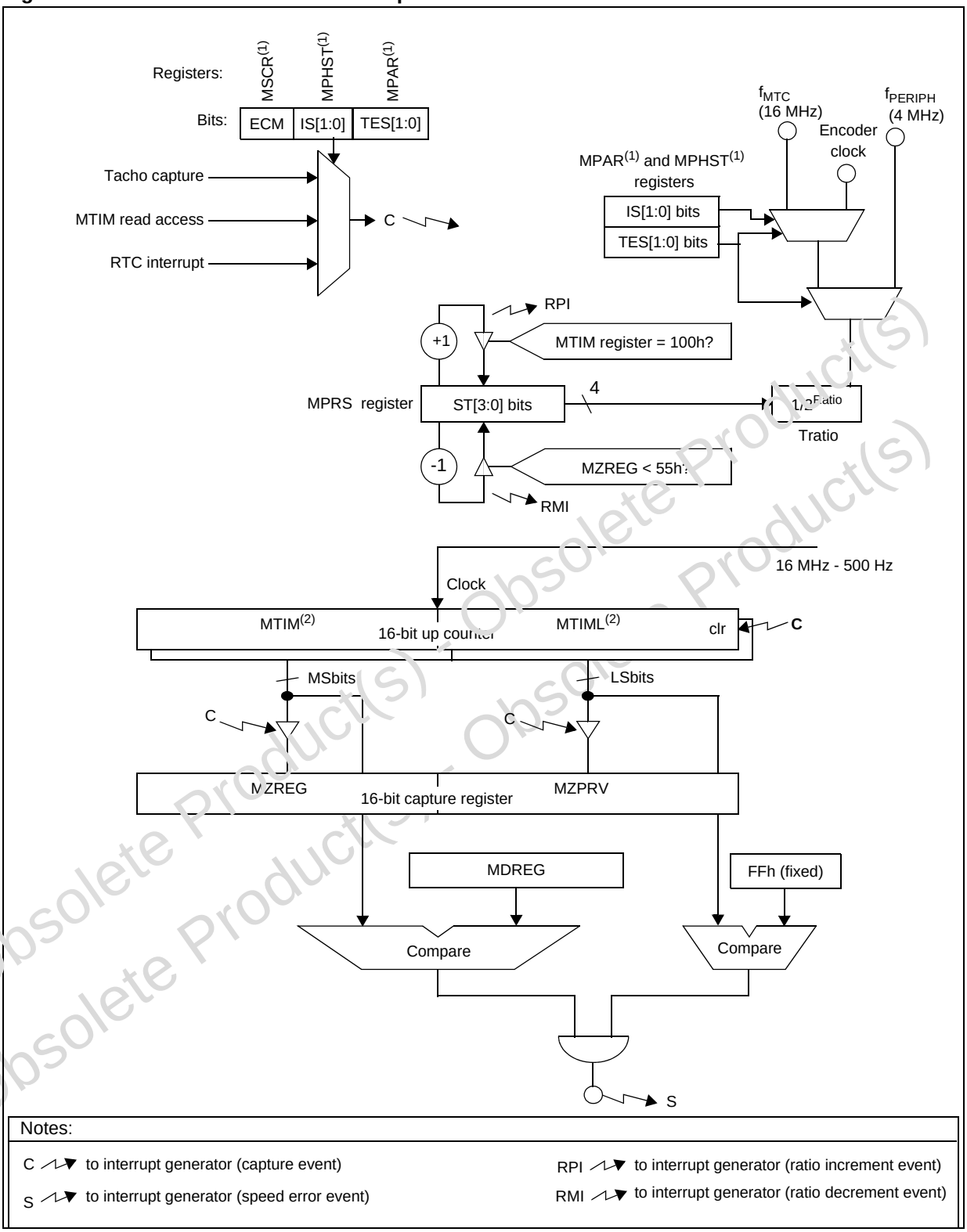


Figure 103. Overview of MTIM timer in speed measurement mode



Notes:

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)

1. Register set-up described in *Speed sensor mode on page 208*.
2. Register updated on R event.

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:

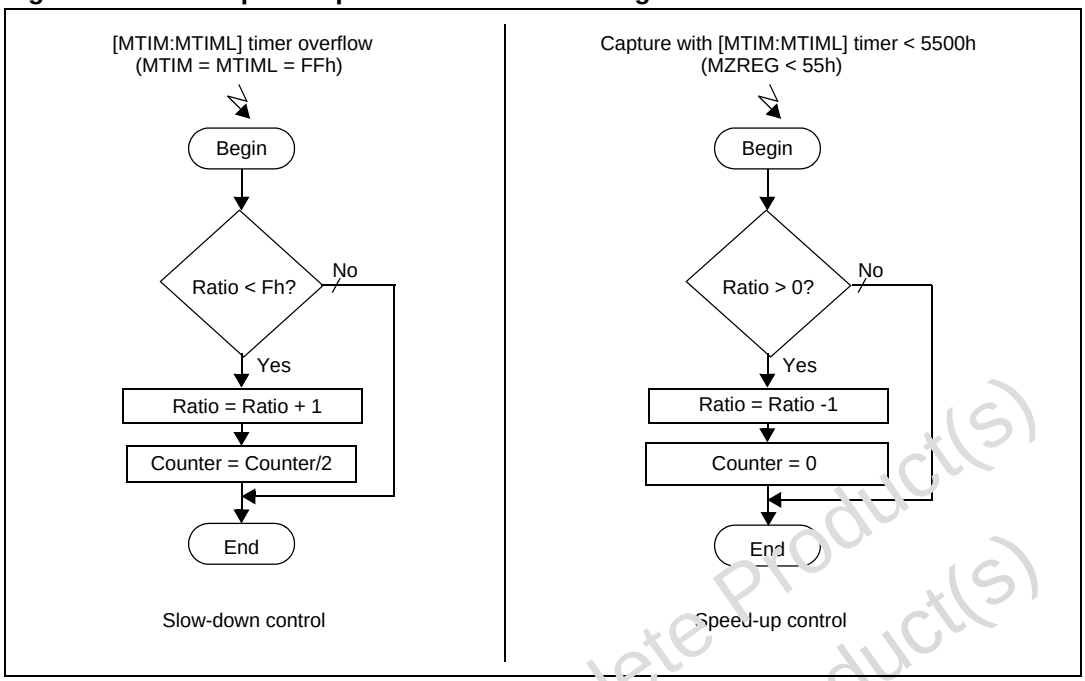
- Motor control peripheral clock (16 MHz) with tachogenerator or hall sensors
- Encoder clock

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 104* and *Table 96*).

- 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, that is, with a lower accuracy. No RMI interrupt 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 summarized in *Table 99*.

Figure 104. Auto-updated prescaler functional diagram



Three kinds of interrupt can be generated in Speed sensor mode, as summarized in *Figure 105*:

- 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 (that is, 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: ***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:*

1. 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 99)

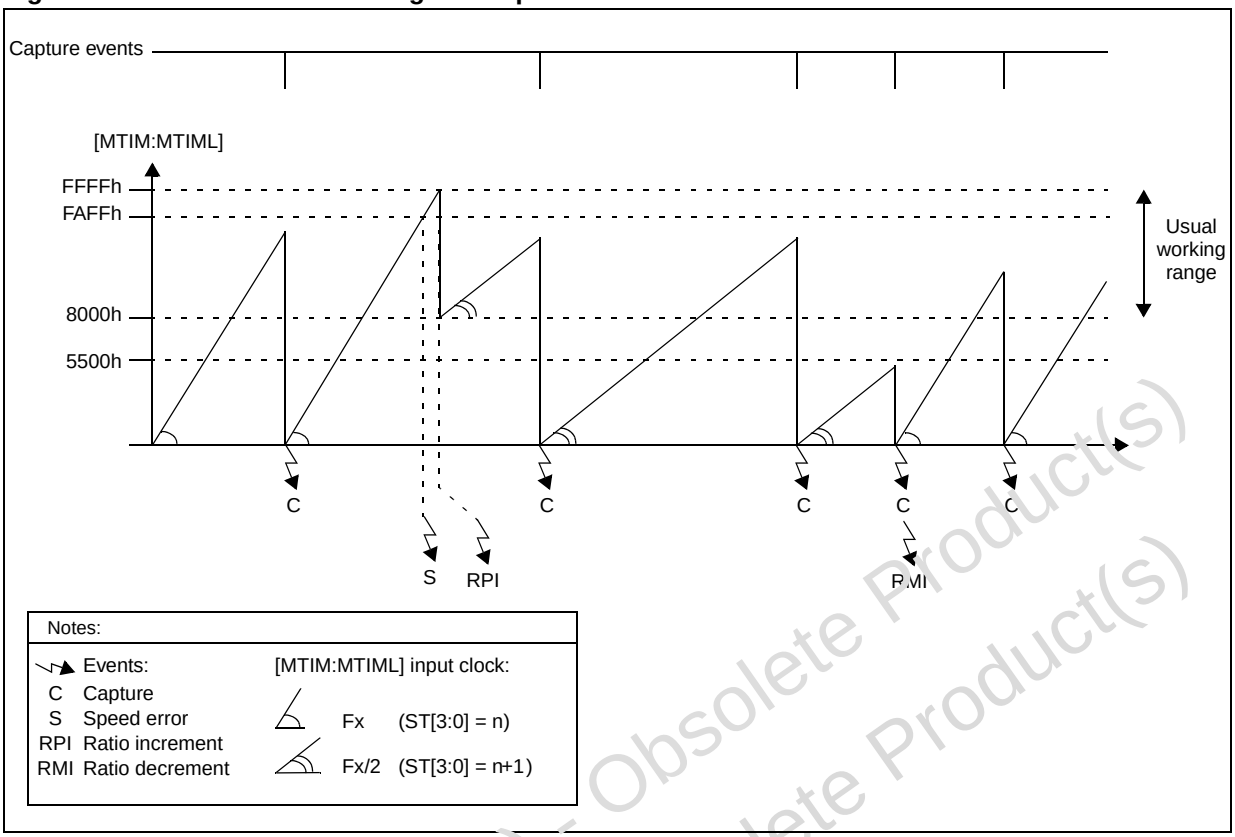
2. MTIM, MTIML must be reset and appropriate values must be written in the ST[3:0] prescaler adapt to the frequency of the signal being measured and to allow speed measurement with sufficient resolution.

Note: ***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 *Debug option on page 219* for more details.

Figure 105. Prescaler auto-change example



Summary

The use of the delay manager registers for the various available modes is summarized in *Table 100*.

Table 100. 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 107*.

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 (≤5V). 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 107*).

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 determines the effect of this interrupt on the MCOx outputs as shown in *Table 101*.

Table 101. 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

1. Only this functionality (CLIM = CLI = OCV = 1) is valid when the three 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).

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.

Current mode

In current mode, the PWM output signal is generated by a combination of the output of the measurement window generator (see *Figure 108*) and the output of the current comparator, and is directed to the output channel manager as well (*Figure 109*).

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.

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 106. Current window and feedback filters

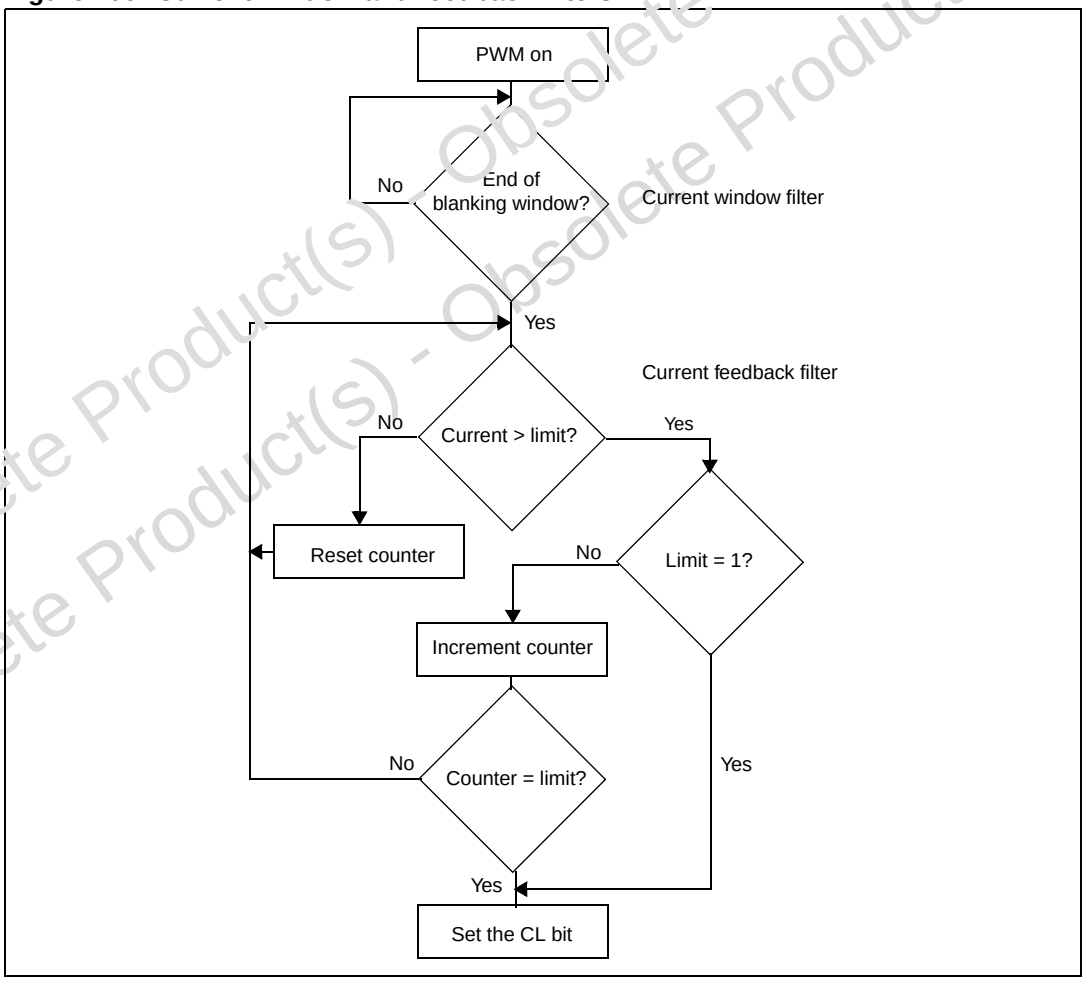


Table 102. Current window filter setting(1)

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

1. 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 103. Current feedback filter setting

CFF2	CFF1	CFF0	No. 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.

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 104. 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 107*).

Note: The MCCFI0 pin is not available in LQFP32 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.

Measurement window

In current mode, the measurement window frequency can be programmed between 390Hz and 50 kHz 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.

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.

Figure 107. Current feedback

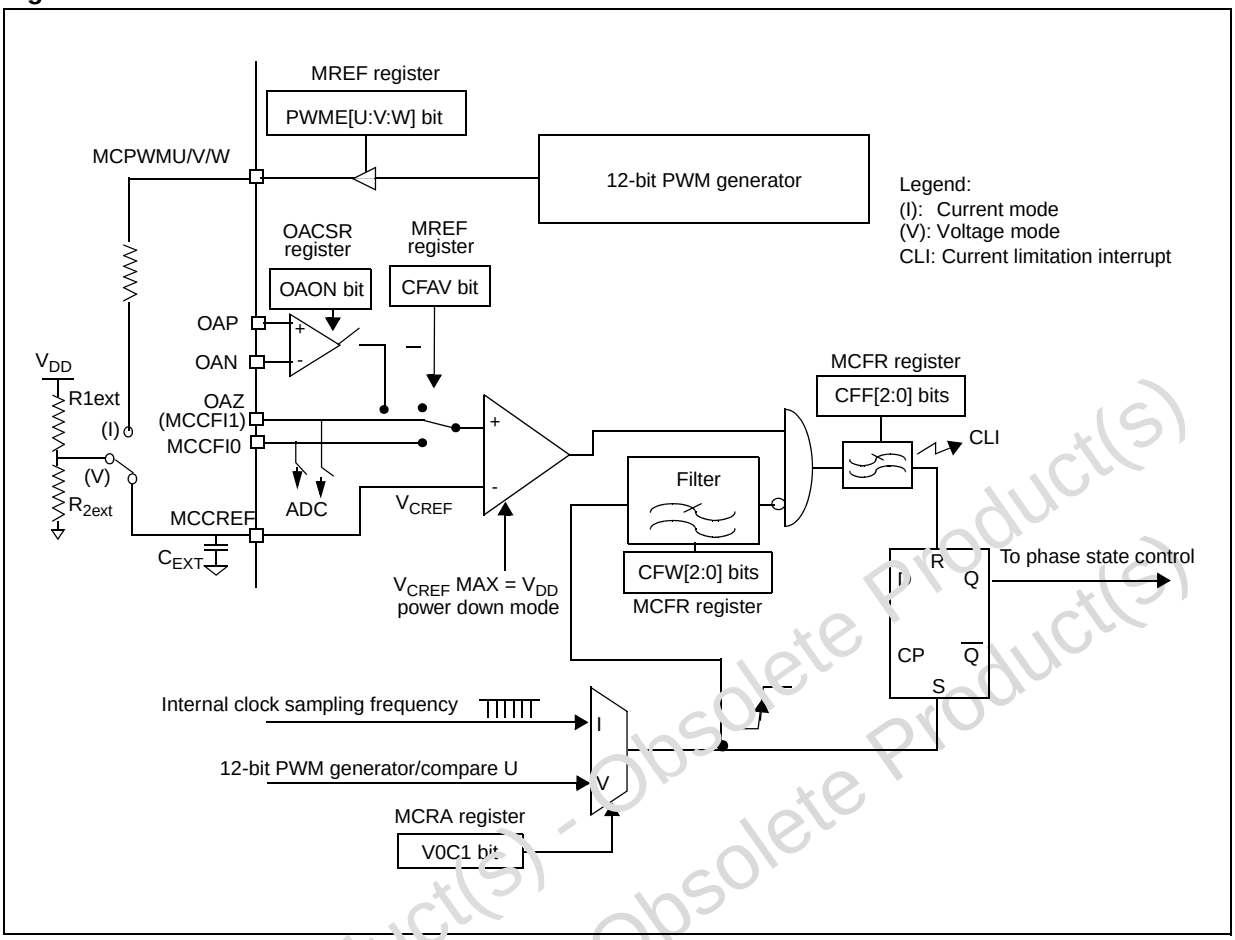


Table 105. Sampling frequency selection(1)

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

Table 105. Sampling frequency selection(1) (continued)

SA3	SA2	SA1	SA0	Sampling frequency
1	1	0	1	961 Hz
1	1	1	0	625 Hz
1	1	1	1	390 Hz

1. *Times are indicated for 4 MHz f_{PERIPH}.*

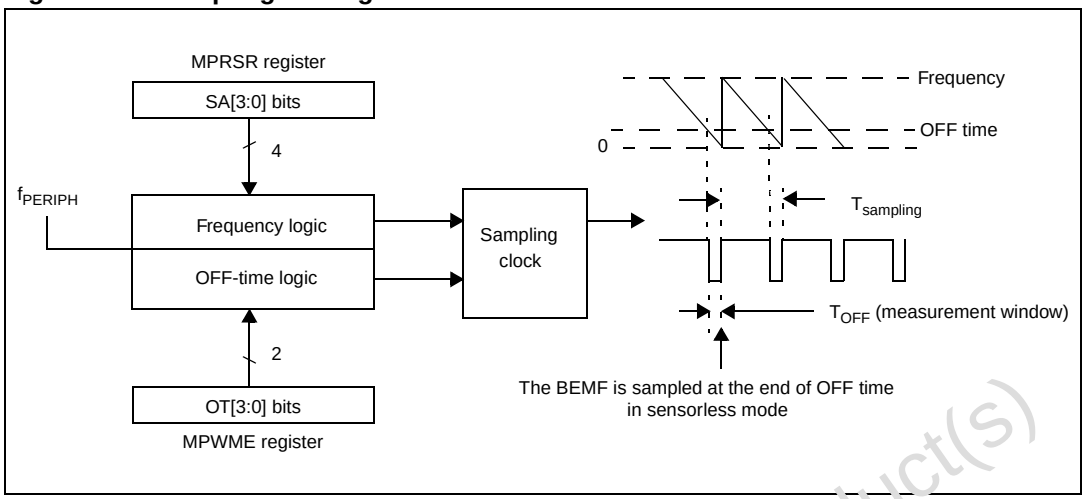
Warning: If the off time value set is superior than the period of the PWM signal (for example 40µs off time for a 50 kHz (25µs period) PWM frequency), then the signal output on MCOx pins selected is a 100% duty cycle signal (always at 1).

Table 106. Off time table(1)

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 mode (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	

1. Times are indicated for 4 MHz f_{PERIPH}.

Figure 108. Sampling clock generation block(1)



1. The MTC controller input frequency (f_{PERIPH}) is 4 MHz in this example, It can be configured to 8 MHz with the XT16: XT8 bits in the MCONF register

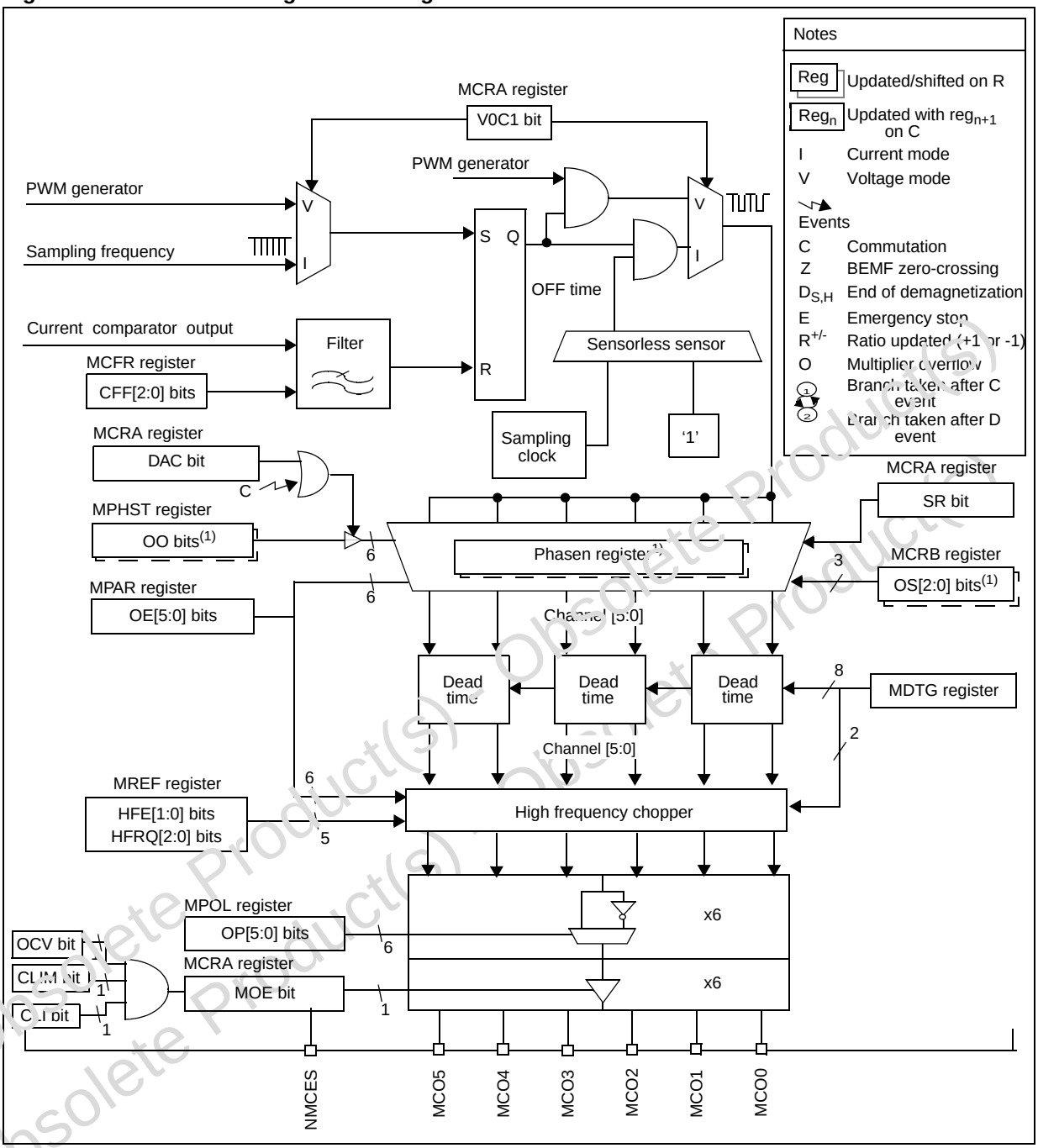
10.6.9 Channel manager

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 109*.

Figure 109. Channel manager block diagram



1. Preload register, changes taken into account at next C event.

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 107. 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 108. DAC and MOE bit meaning

MOE bit	DAC bit	Effect on output
0	x	Reset state(1)
1	0	Standard running mode
1	1	MPHST register value (depending on MPOL, MPAR register values and PWM setting). See *Table 155*.

1. 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 109. 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 free-wheeling 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 111* and *Figure 110*.

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 allows control of the PWM signal between the Z event and the next C event.

Note: ***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.*

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.

Figure 110. PWM application in voltage or current sensorless mode (see *Table 133*)

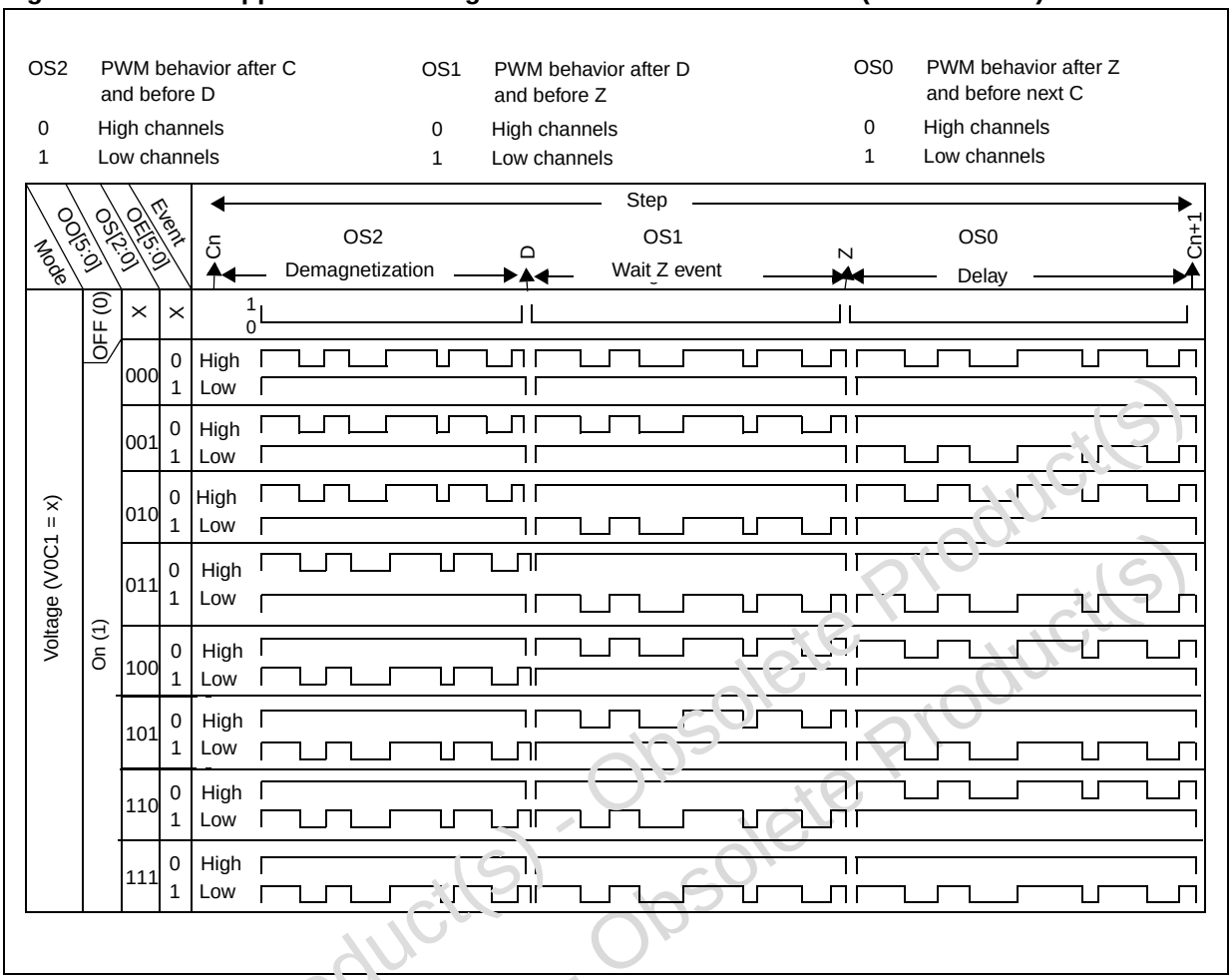
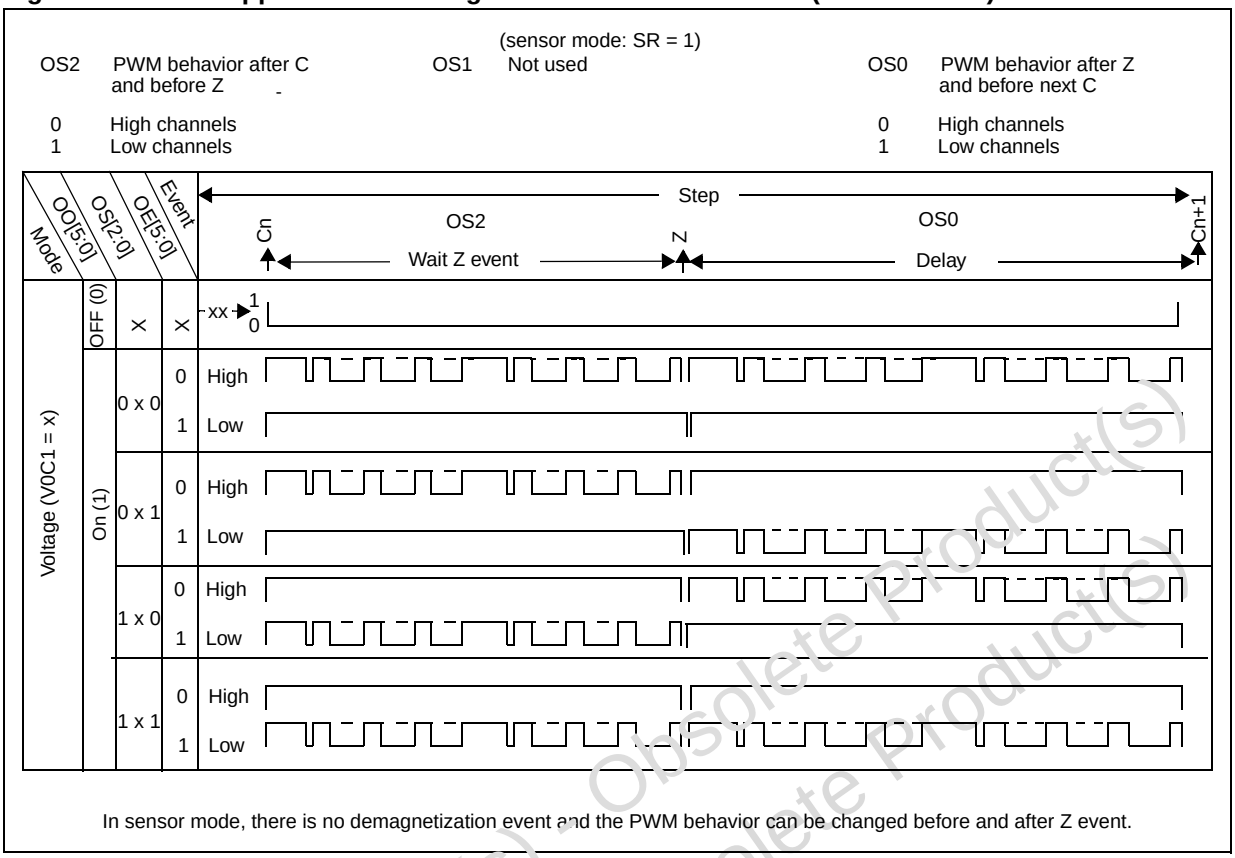


Figure 111. PWM application in voltage or current sensor mode (see *Table 134*)



In sensor mode, there is no demagnetization event and the PWM behavior can be changed before and after Z event.

Deadtime 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, etc.)

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 deadtime 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 112 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 113* and *Figure 114*).

Figure 112. Deadtime waveforms

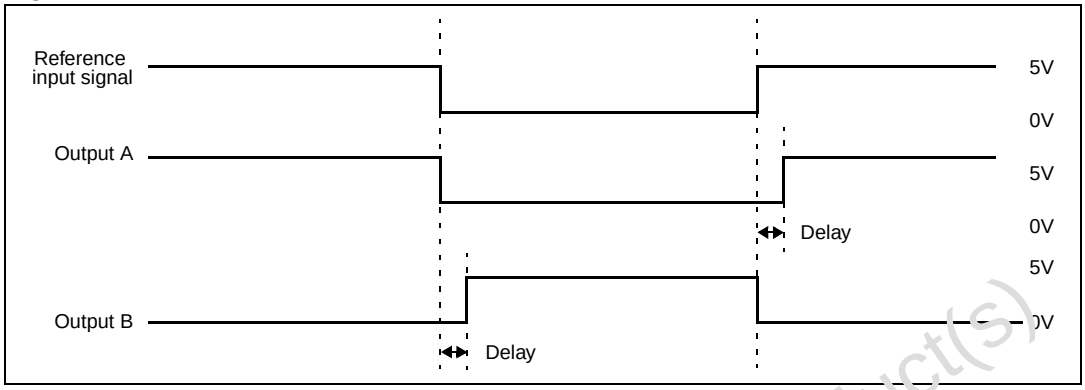


Figure 113. Deadtime waveform with delay greater than the negative PWM pulse

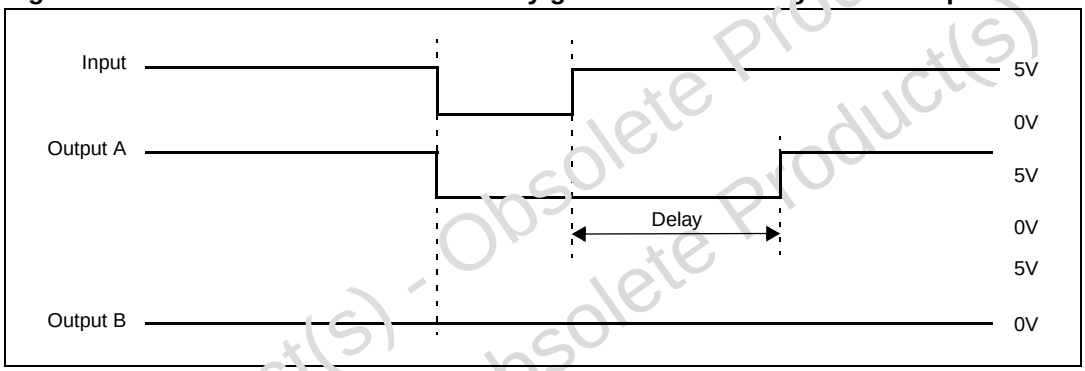


Figure 114. Deadtime waveform with delay greater than the positive PWM pulse

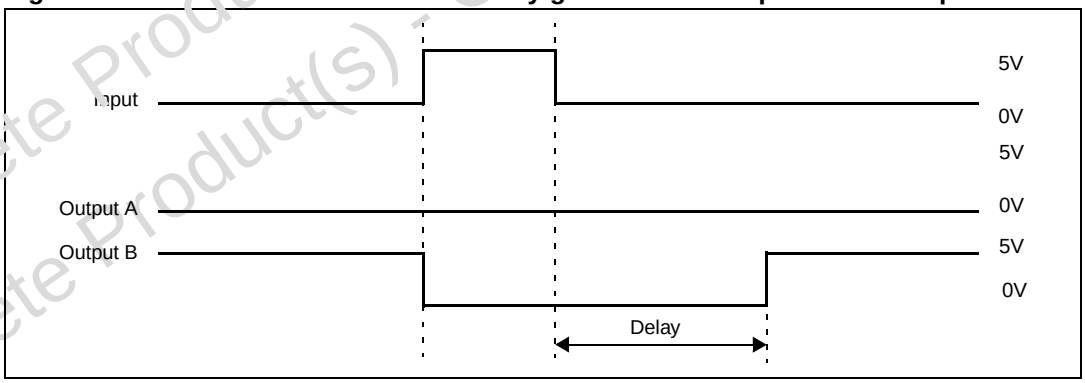


Table 110. Deadtime programming and example

DTG5	DTG4	T_{dtg}	Deadtime expression	Deadtime value	T_{dtg} @ 16 MHz f_{mtc}	Deadtime range @ 16 MHz 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 110* summarizes the set-up of the deadtime generator.

IT_{mtc} is the period of the deadtime 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, etc.), 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 115*) to drive the MCOx outputs, deadtime is added as shown in *Figure 112*.

The deadtime generator is enabled/disabled using the DTE bit.

The effect of the DTE bit depends on the PCN bit value.

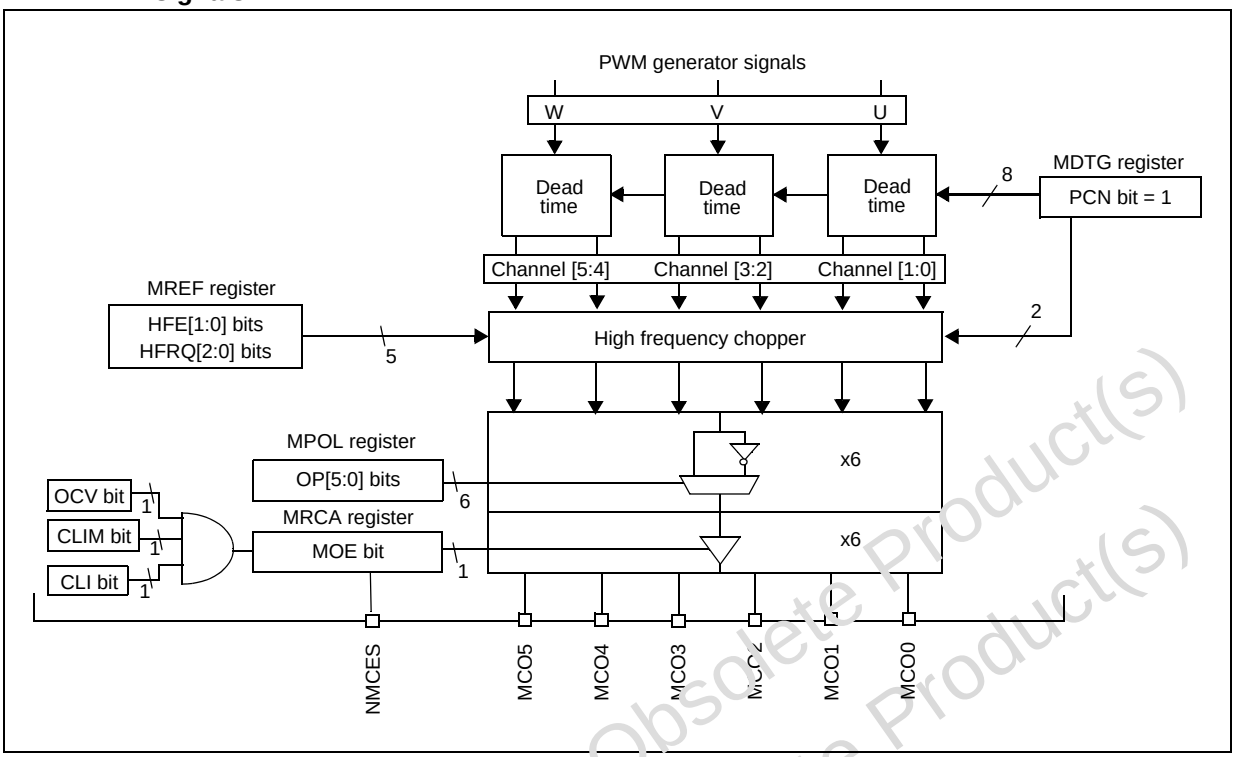
If the PCN bit is set:

- 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 deadtime generator.

Note: The reset value of the MDTG register is FFh so when configuring the deadtime, it is mandatory to follow one the two following sequences:

1. *To use deadtimes while the PCN bit is set; from reset state write the MDTG value at once. The DTE bit is read back as 1 whatever the programming value (read only if PCN = 1)*
2. *To use deadtimes while the PCN bit is reset, write first the deadtime value in DTG[5:0], then reset the PCN bit, or do both actions at the same time.*

Figure 115. Channel manager output block diagram with PWM generator delivering three PWM signals



1. The output of the current limitation comparator can be used when three PWM signals are enabled if the VOC1 bit = 0 in the MCRA register.

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), the following restrictions are applied:

- 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 channel 0 or channel 1.
- When both channels of a pair are inactive, the corresponding outputs also stay inactive (this is mandatory to allow BEMF zero-crossing detection).

These restrictions are summarized in *Table 111*, which also 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 111. Deadtime 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[1])	PWM	PWM
1 (PWM[1])	0	PWM	PWM
1	1 (PWM[1])	0	0
1 (PWM[1])	1	0	0
1	0	1	0
0	1	0	1
0	0	0	0

1. PWM generation enabled.

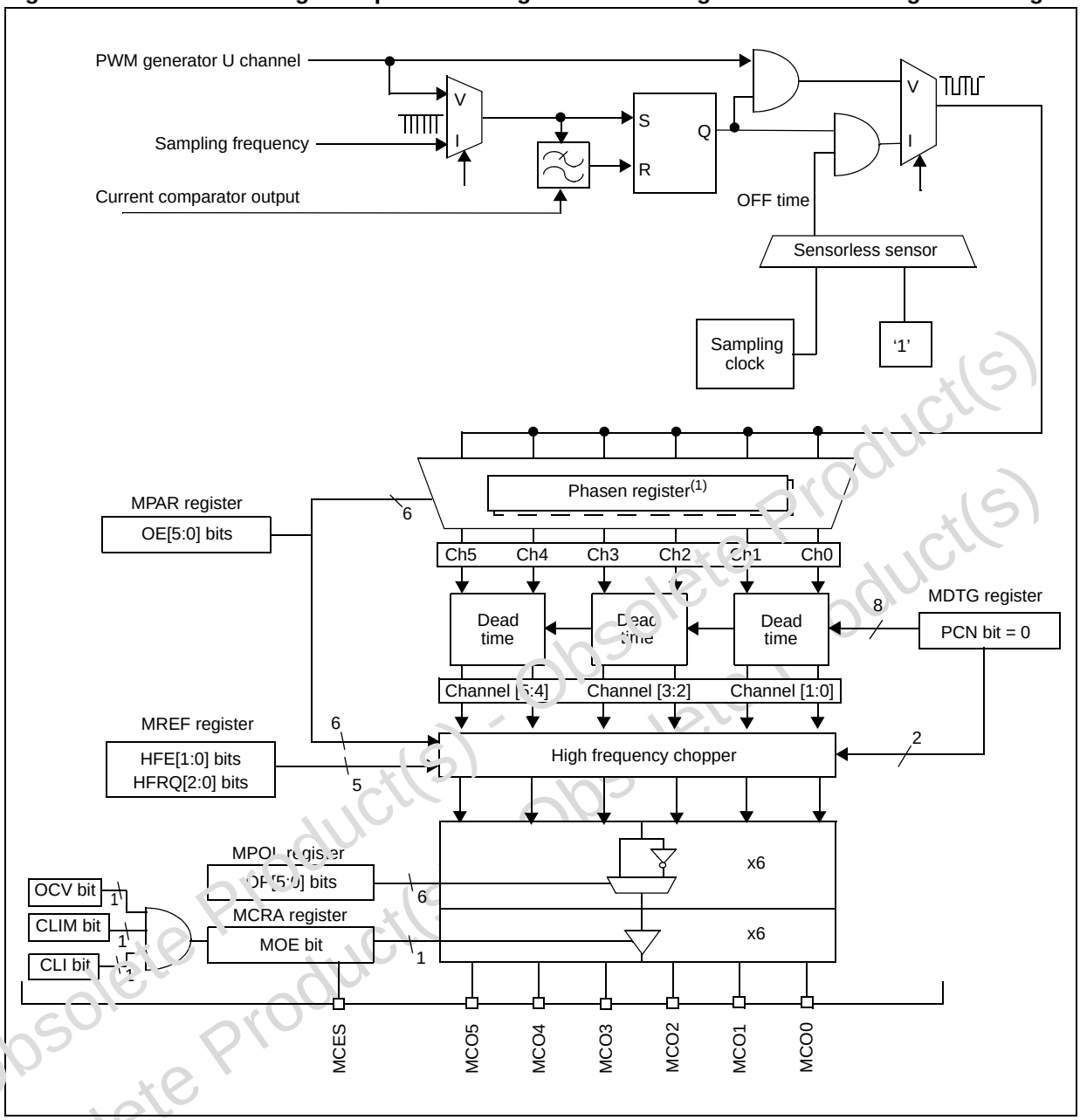
Warning: Grouping channels by pairs imposes the external connections between the MCO outputs and power devices; the user must therefore pay attention to respect the 'recommended schematics' described in *Figure 124 on page 289* and *Figure 125 on page 290*.

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 purposes to avoid cross-conduction.

Caution: To clear the DTE bit from reset state of MDTG register (FFh), the PCN bit must be cleared before.

Figure 116. Channel manager output block diagram with PWM generator delivering 1 PWM signal



1. Preload register, changes taken into account at next C event.

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 112*).

Table 112. 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 be any of the eight values from 100 kHz to 2 MHz selected by the HFRQ[2:0] bits in the MREF register (see *Table 113*).

Table 113. Chopping frequency

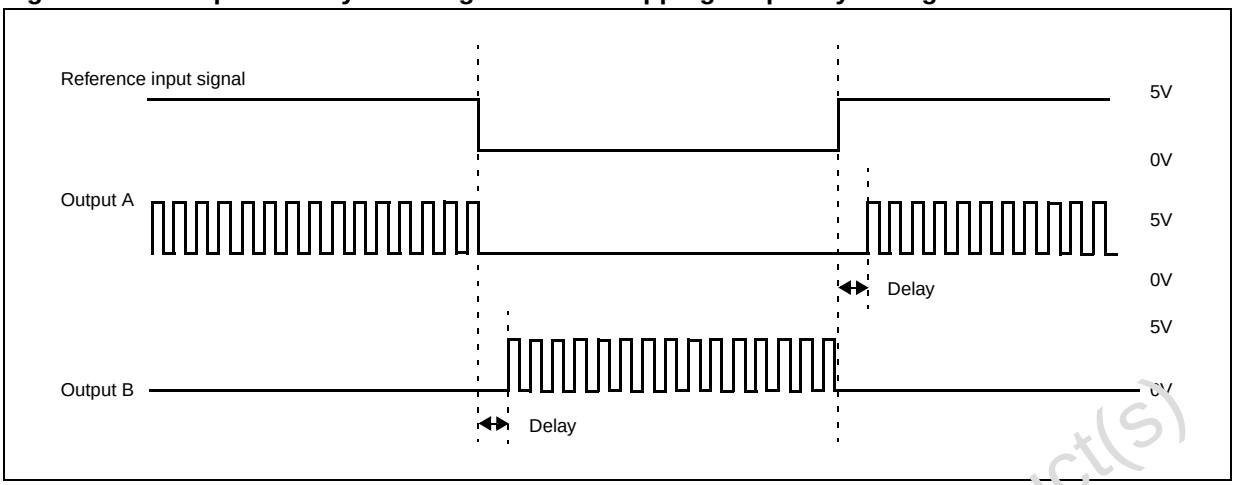
HFRQ2	HFRQ1	HFRQ0	Chopping frequency	
			F_{mtc} = 16 MHz; F_{mtc} = 8 MHz	F_{mtc} = 4 MHz
0	0	0	100 kHz	50 kHz
0	0	1	200 kHz	100 kHz
0	1	0	400 kHz	200 kHz
0	1	1	500 kHz	250 kHz
1	0	0	800 kHz	400 kHz
1	0	1	1 MHz	500 kHz
1	1	0	1.33 MHz	666.66 MHz
1	1	1	2 MHz	1 MHz

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 109, Figure 115 and Figure 116 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 Deadtime generator on page 245. This means that the high and low channels are fixed and the HFE[1:0] bits indicate where to apply the chopper. Figure 117 shows typical complementary PWM signals with high frequency chopping enabled on both high and low drivers.

Figure 117. Complementary PWM signals with chopping frequency on high and low side drivers



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.

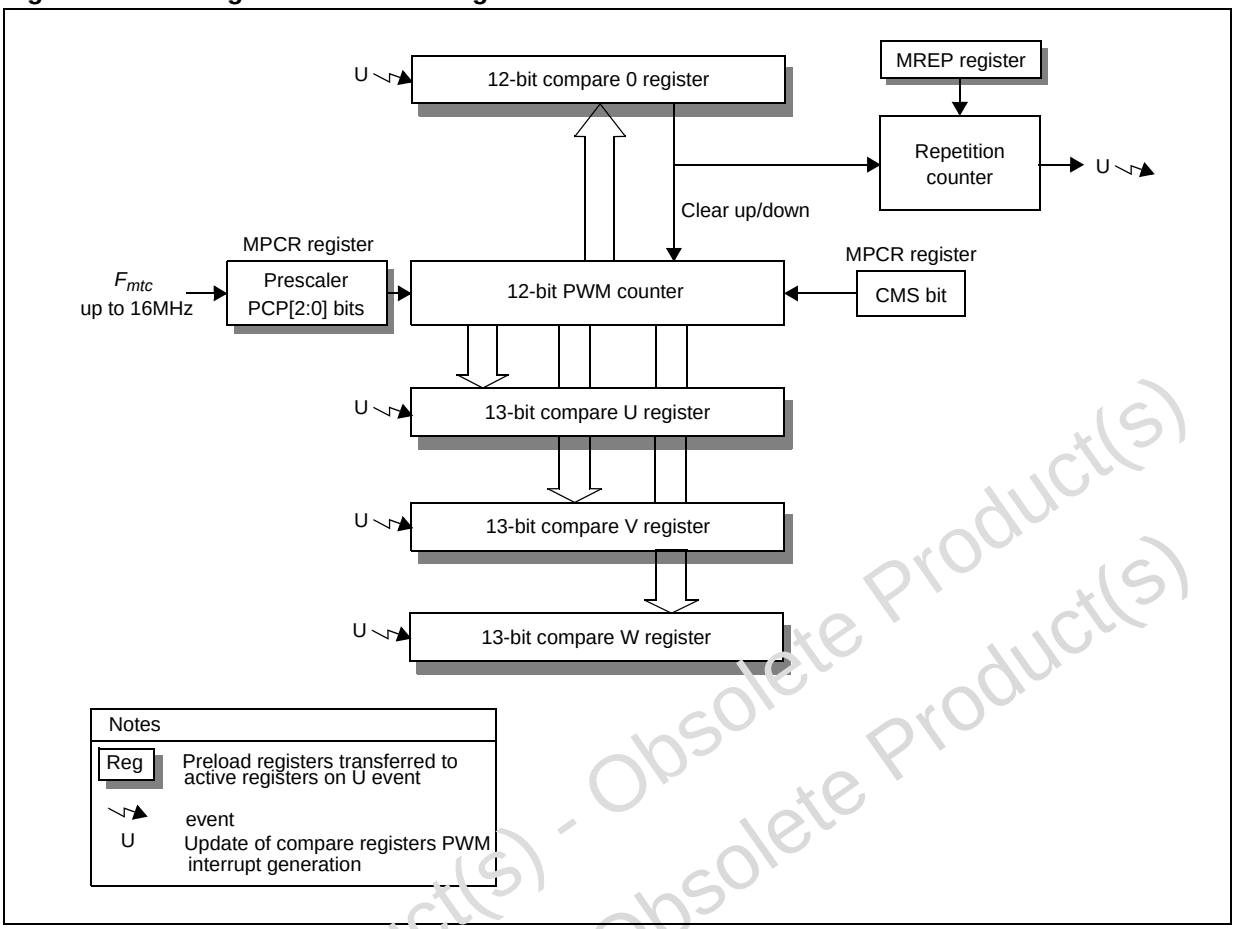
When driving PM BLDC motors in six-step mode (voltage mode only, either sensored or sensorless) a single PWM signal (phase U) is used to supply the input stage, PWM and channel manager blocks according to the selected modes.

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.

Main features

- 12-bit PWM free-running up/down counter with up to 16 MHz input clock (F_{mtc})
- Edge-aligned and center-aligned PWM operating modes
- Possibility to reload 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 resynchronization feature

Figure 118. PWM generator block diagram



Functional description

The three 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 respectively.

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 compare values to be written without risks of spurious PWM transitions.

The block diagram of the PWM generator is shown in *Figure 118*.

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.

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 119 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.

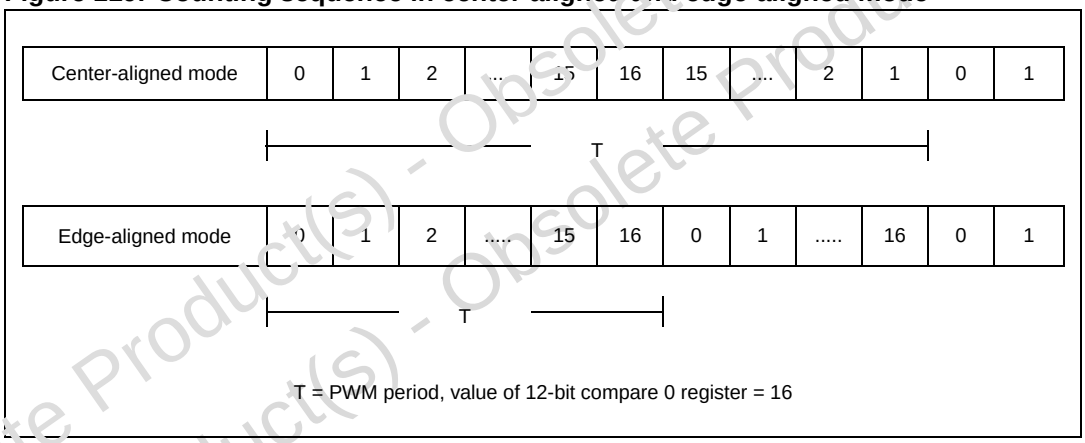
The comparisons described here are performed between the PWM counter value extended to 13 bits and the 13-bit compare register. Having a compare range greater than the counter range is mandatory to get a full PWM range (that is, up to 100% modulation). This principle is maintained for 8-bit PWM operations.

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 119. Counting sequence in center-aligned and edge-aligned mode

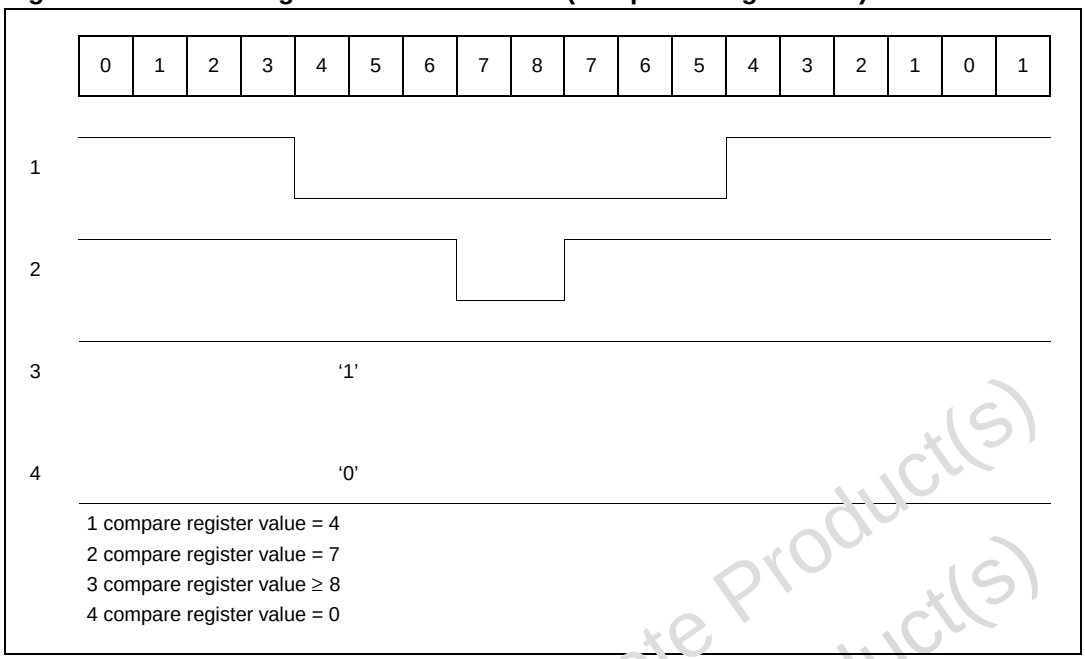


If the 13-bit compare register value is greater than the extended compare 0 register (the 13th bit is set to '0'), the corresponding PWM output signal is held at '1'.

If the 13-bit compare register value is 0, the corresponding PWM output signal is held at '0'.

Figure 120 shows some center-aligned PWM waveforms in an example where the compare 0 register value = 8.

Figure 120. Center-aligned PWM waveforms (compare 0 register = 8)



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.

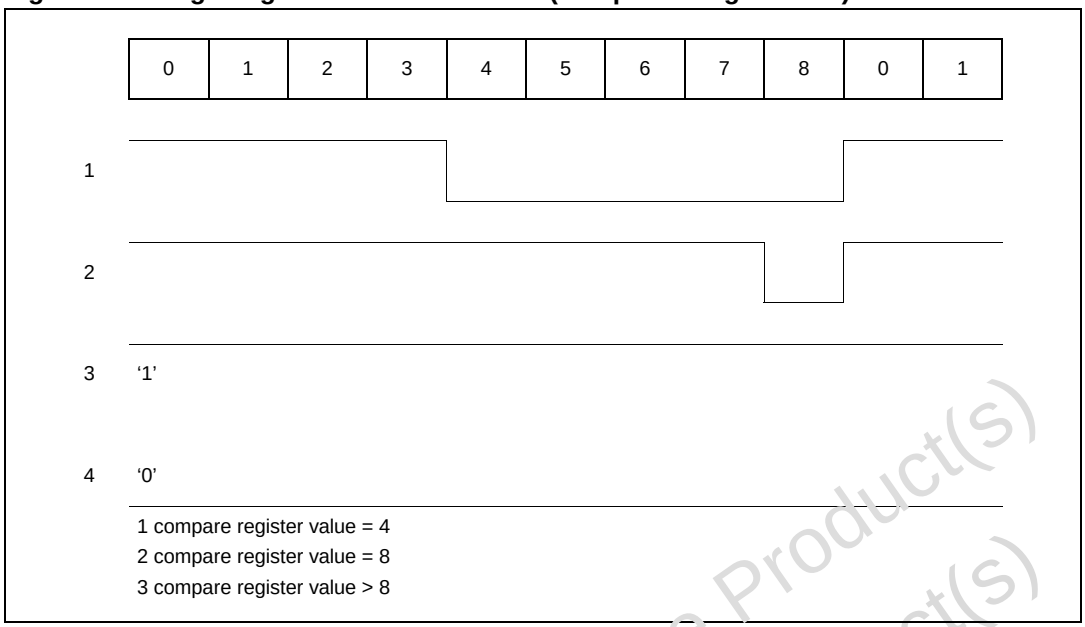
The PWM signals are set to ‘0’ when the PWM counter reaches, in up-counting, the corresponding 13-bit Compare register value and they are set to ‘1’ when the PWM counter is cleared.

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 121 shows some edge-aligned PWM waveforms in an example where the compare 0 register value = 8.

Figure 121. Edge-aligned PWM waveforms (compare 0 register = 8)



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 122* presents the way compare 0 and compare U, V, W should be loaded. It requires loading two bytes in the MCMPxH and MCMPxL registers (that is, 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)

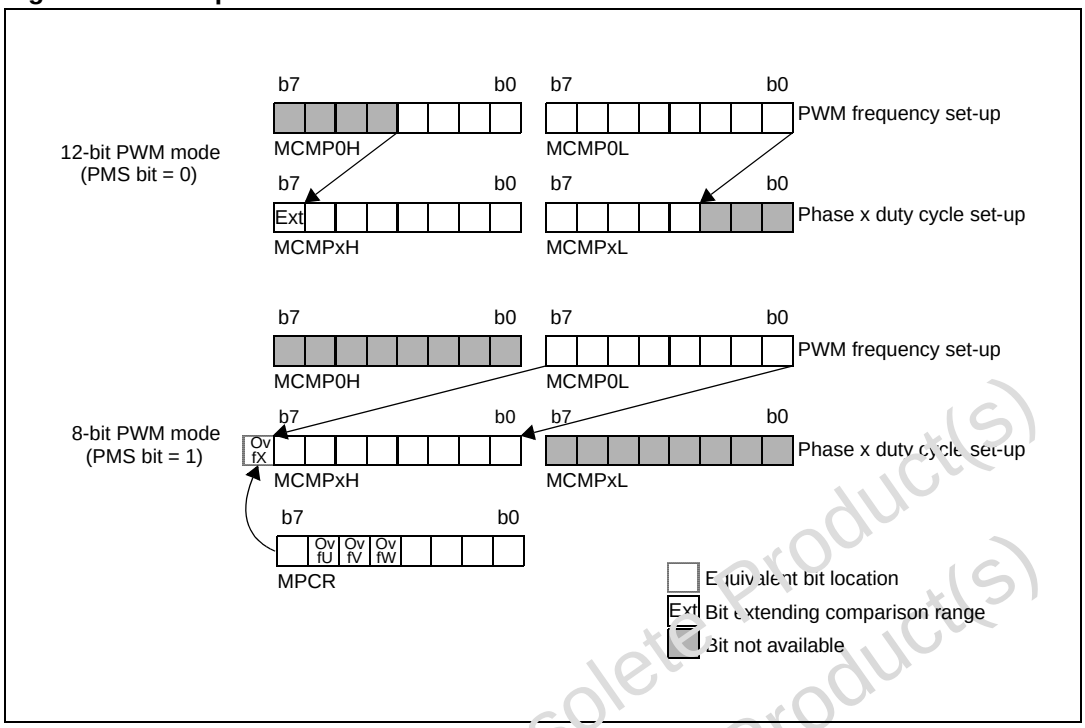
This mode is useful whenever the MCMP0 value is less than or equal to 8-bits. It allows significant CPU resource savings when computing three-phase duty cycles during PWM interrupt routines. In this mode, the compare 0 and compare U, V, W registers have the same size (8 bits). The extension of the MCMPx registers is done in using the OVFx bits in the MPCR register (refer to *Figure 122*). These bits force the related duty-cycles to 100% and are reset by hardware on occurence of a PWM update event.

Note: ***Read access to registers with preload****: During read accesses, values read are the content of the preload registers, not the active registers.*

Note: ***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: ***MCMP0x registers****: The configuration MCMP0H = MCMP0L = 0 is not allowed.*

Figure 122. Comparison between 12-bit and 8-bit PWM mode



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, that is, one T_{mtc} period. When refreshing compare registers only once per PWM period in center-aligned mode, maximum resolution is $2xT_{mtc}$, due to the symmetry of the pattern.

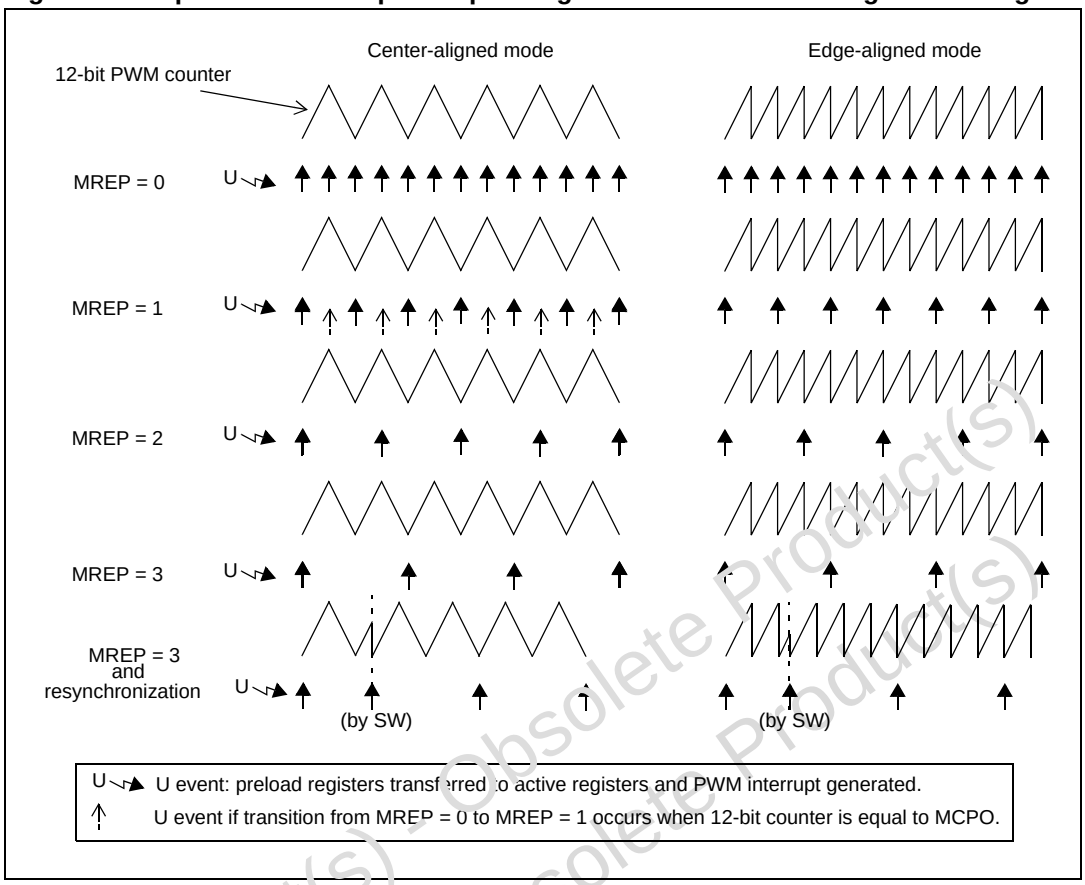
The repetition down counter is an auto-reload type; the repetition rate is maintained as defined by the MREP register value (refer to *Figure 123*).

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 *Debug option on page 219* for more details.

Figure 123. Update rate examples depending on mode and MREP register setting



Timer resynchronization

The 12-bit timer can be resynchronized by a simple write access with FFh value in the MISR register. Resynchronization 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 resynchronize 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 resets the counter and transfers 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 291.

Note: Loading FFh value in the MISR register has no effect on any flag other than the PUI flag and generates 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 has no effect on any flags except those 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 are taken into account.

PWM generator initialization and start-up

The three-phase generator counter stays in reset state (that is, 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, that is, 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.

Table 114. Effect of low power modes on MTC

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

Table 115. MTC interrupt control/wake-up capability

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	No

The MTC interrupt events are connected to the three interrupt vectors (see *Section 7: Interrupts*).

They generate an interrupt if the corresponding enable control bit is set and the interrupt mask in the CC register is reset (RIM instruction).

10.6.13 MTC registers

Timer counter register (MTIM)

MTIM Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
T[7:0]							
R/W							

Table 116. MTIM register description

Bit	Name	Function
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 103*) 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)

MTIML Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TL[7:0]							
R/W							

Table 117. MTIML register description

Bit	Name	Function
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 (that is, as a 16-bit timer).

Capture Z_{n-1} register (MZPRV)

MZPRV Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ZP[7:0]							
R/W							

Table 118. MZPRV register description

Bit	Name	Function
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)

MZREG Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ZC[7:0]							
R/W							

Table 119. MZREG register description

Bit	Name	Function
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:MTIMR] registers in speed sensor mode. A read access to MZREG in this case disable the speed captures up to MZPRV reading (refer to *Speed measurement mode on page 227*).

Compare C_{n+1} register (MCOMP)

MCOMP Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DC[7:0]							
R/W							

Table 120. MCOMP register description

Bit	Name	Function
7:0	DC[7:0]	Next compare value These bits contain the compare value for the next commutation (C_{N+1}).

Demagnetization register (MDREG)

MDREG Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DN[7:0]							
R/W							

Table 121. MDREG register description

Bit	Name	Function
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.

A_N weight register (MWGHT)

MWGHT Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
AN[7:0]							
R/W							

Table 122. MWGHT register description

Bit	Name	Function
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 when a Z event occurs (see *Equation 10*).

Equation 10

$$\frac{Zn \times MWGHT}{256(d)} \text{ or } \frac{Zn\text{-}1 \times MWGHT}{256(d)} \quad (*)$$

where (*) depends on the DCB bit in the MCRA register

Prescaler and sampling register (MPRSR)

MPRSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SA[3:0]				ST[3:0]			
R/W				R/W			

Table 123. MPRSR register description

Bit	Name	Function
7:4	SA[3:0]	Sampling ratio These bits contain the sampling ratio value for current mode. Refer to *Table 105: Sampling frequency selection on page 238*.
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 98: Step frequency/period range (4 MHz) on page 226* and *Table 99: modes of accessing mtim timer-related registers on page 226*.

Interrupt mask register (MIMR)

MIMR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
PUM	SEM	RIM	CLIM	EIM	ZIM	DIM	CIM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 124. MIMR register description

Bit	Name	Function
7	PUM	PWM update mask bit 0: PWM update interrupt disabled 1: PWM update interrupt enabled
6	SEM	Speed error mask bit 0: Speed error interrupt disabled 1: Speed error interrupt enabled
5	RIM	Ratio update interrupt mask bit 0: Ratio update interrupts (R+ and R-) disabled 1: Ratio update interrupts (R+ and R-) enabled
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.
3	EIM	Emergency stop interrupt mask bit 0: Emergency stop interrupt disabled 1: Emergency stop interrupt enabled
2	ZIM	Back EMF zero-crossing interrupt mask bit 0: BEMF Zero-crossing Interrupt disabled 1: BEMF Zero-crossing Interrupt enabled
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
0	CIM	Commutation/capture interrupt mask bit 0: Commutation/capture interrupt disabled 1: Commutation/capture interrupt enabled

Interrupt status register (MISR)

MISR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
PUI	RPI	RMI	CLI	EI	ZI	DI	CI
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 125. MISR register description

Bit	Name	Function
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
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 increments the ST[3:0] bits when the next commutation occurs and shifts 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
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 decrements the ST[3:0] bits when the next commutation occurs and shifts 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
4	CLI	Current limitation interrupt flag 0: No current limitation interrupt pending 1: Current limitation interrupt pending
3	EI	Emergency stop interrupt flag 0: No emergency stop interrupt pending 1: Emergency stop interrupt pending
2	ZI	BEMF zero-crossing interrupt flag 0: No BEMF zero-crossing interrupt pending 1: BEMF zero-crossing interrupt pending

Table 125. MISR register description (continued)

Bit	Name	Function
1	DI	End of demagnetization interrupt flag 0: No end of demagnetization interrupt pending 1: End of demagnetization interrupt pending
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 resets the PWM generator counter and transfers the compare preload registers in the active registers by generating a U event (PUI bit set to 1). Refer to Timer resynchronization on page 258.

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 is not 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.

*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 has no effect (they are read as 0) without interrupt generation.*

***In switched mode** (SWA = 0 in the MRCA register): To avoid losing any interrupts when modifying the RMI and RPI bits the following instruction sequence is recommended:*

ld MISR, # 0x9F; reset both RMI and RPI bits.

ld MISR, # 0xBF; set RMI bit.

ld MISR, # 0xDF; set RPI bit.

Control register A (MCRA)

MCRA Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
MOE	CKE	SR	DAC	V0C1	SWA	PZ	DCB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 126. MCRA register description

Bit	Name	Function
7	MOE	Output enable bit 0: Outputs disabled; MC0[5:0] outputs are put in reset state(1)(2) 1: Outputs enabled; MC0[5:0] outputs enabled

Table 126. MCRA register description (continued)

Bit	Name	Function
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.* See *Table 127*.
5	SR	Sensor ON/OFF 0: Sensorless mode 1: Position sensor mode See *Table 128*, *Table 133* and *Table 134*.
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 See *Table 129*. *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.*
3	V0C1	Voltage/current mode 0: Voltage mode 1: Current mode
2	SWA	Switched/autoswitched mode 0: Switched mode 1: Autoswitched mode *Notes:* *1. After reset, in autoswitched mode (SWA = 1), the motor control peripheral is waiting for a C commutation event.* *2. After reset, a C event is immediately generated when CKE and SWA are simultaneously set due to a nil value of MCOMP.*
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.*
0	DCB	Data capture bit 0: Use MZPRV (Z_N-1) for multiplication 1: Use MZREG (Z_N) for multiplication See *Table 130*.

1. The reset state is either high impedance, high or low state depending on the corresponding option bit.
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).

Table 127. Output configuration summary(1)

CKE bit	MOE bit	DAC bit	Peripheral clock	Effect on MCOx output
0	0	x	Disabled	Reset state
0	1	0	Disabled	Peripheral frozen(2)
0	1	1	Disabled	Direct access via MPHST (only logical level)(3)
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)(3)

1. 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 puts all the MCOx outputs under reset state defined by the corresponding option bit.
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.
2. "Peripheral frozen" configuration is not recommended, as the peripheral may be stopped in an 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.
3. 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 155: Deadtime generator set-up on page 281*.

Table 128. Sensor mode selection

SR bit	Mode	OS[2:0] bits	Behavior of the output PWM
0	Sensors not used	OS[2:0] bits enabled	'Between C_n and D' behavior, 'Between D and Z' behavior and 'Between Z and C_{n+1}' behavior
1	Sensors used	OS1 disabled	'Between C_n and Z' behavior and 'Between Z and C_{n+1}' behavior

Table 129. 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 155*)

Table 130. Multiplier result

DCB bit	Commutation delay
0	MCOMP = MWGHT x MZPRV/256
1	MCOMP = MWGHT x MZREG/256

Control register B (MCRB)

MCRB Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved	CPB(1)	HDM(1)	SDM(1)	OCV	OS2(1)	OS[1:0]	
-	R/W	R/W	R/W	R/W	R/W	R/W	

1. Preload bits, new value taken into account at the next C 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 *Control register A (MCRA) on page 265*. The use of a preload register allows all the registers to be updated at the same time.

Table 131. MCRB register description

Bit	Name	Function
7	-	Reserved, must be kept at reset value.
6	CPB	Compare bit for zero-crossing detection 0: Zero crossing detection on falling edge 1: Zero crossing detection on rising edge
5	HDM	Hardware demagnetization event mask bit 0: Hardware demagnetization disabled 1: Hardware demagnetization enabled
4	SDM	Simulated demagnetization event mask bit 0: Simulated demagnetization disabled 1: Simulated demagnetization enabled
3	OCV	Over current handling in voltage mode 0: Overcurrent protection is OFF 1: Overcurrent protection is ON This bit acts as described in *Table 132*.
2:0	OS2, OS[1:0]	Operating output mode selection bits These bits are used to define the various PWM output configurations. Refer to the step behavior diagrams (*Figure 110* and *Figure 111*), *Table 133: Step behavior/sensorless mode*, *Table 134: PWM mode when SR = 1*, and *Table 135: PWM mode when DAC = 1*.

Table 132. 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

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 *Over current handling in voltage mode on page 234*.

Table 133. Step behavior/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 behavior diagrams (Figure 110 and Figure 111).

Table 134. 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 135. 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 resets ~~has the effect of resetting~~ all the preload bits.**

Control register C (MCRC)

MCRB Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
SEI/OI	EDIR/HZ	SZ	SC	SPLG	VR[2:0]		
R/W	RO	R/W	R/W	R/W	R/W		

Table 136. MCRC register description

Bit	Name	Function
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
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 (that is, advance/delay) between the two signals in quadrature output by the encoder (see *Figure 91*). 0: MCIA input delayed compared to MCIB input 1: MCIA input in advance compared to MCIB input
5	SZ	Simulated zero-crossing event bit 0: No simulated zero-crossing event 1: Simulated zero-crossing event

Table 136. MCRC register description (continued)

Bit	Name	Function
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)
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 160: MCONF register description on page 284*. 0: Normal mode (Z sampling at PWM frequency at the end of the OFF time) 1: Z event sampled at f_{SCF} (see *Table 166*) *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 comes only from the current loop.*
2:0	VR[2:0]	BEMF/demagnetization reference threshold These bits select the V_{REF} value as shown below. 111: V_{REF} voltage threshold = threshold voltage set by external MCVREF pin 110: 3.5V(1) 101: 2.5V(1) 100: 2V(1) 011: 1.5V(1) 010: 1V(1) 001: 0.6V(1) 000: 0.2V(1) The V_{REF} value is used for BEMF and demagnetization detection.

1. Typical values for V_{DD} = 5V

Phase state register (MPHST)

MPHST Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
IS[1:0](1)		OO[5:0](1)					
R/W		R/W					

1. Preload bits, new value taken into account at the next C event.

Table 137. MPHST register description

Bit	Name	Function
7:6	IS[1:0]	Input selection bits These bits mainly select the input to connect to the comparator: 00: channel selected = MCIA. 01: channel selected = MCIB. 10: channel selected = MCIC. 11: channel selected = Both MCIA and MCIB: encoder mode. The fourth configuration (IS[1:0] = 11) specifies that an incremental encoder is used (in this case MCIA and MCIB digital signals are directly connected to the incremental encoder interface and the analog multiplexer is bypassed).

Table 137. MPHST register description (continued)

Bit	Name	Function
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 if DAC directly = 1. 0: Channel OFF (output channel state inactive), the relevant switch is OFF, no PWM possible 1: Channel ON (output channel state active), the relevant switch is ON, PWM is possible (not significant when PCN or DTE bit is set)

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 resets all the bits with preload.

Motor current feedback register (MCFR)

MCFR — Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
RPGS	RST	CFF[2:0]			CFW[2:0]		
R/W	R/W	R/W			R/W		

Table 138. MCFR register description

Bit	Name	Function
7	RPGS	Register page selection 0: Access to registers mapped in page 0 1: Access to registers mapped in page 1
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
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(1): 000: current feedback samples = 1 001: current feedback samples = 2 010: current feedback samples = 3 011: current feedback samples = 4 100: current feedback samples = 5 101: current feedback samples = 6 110: current feedback samples = 7 111: current feedback samples = 8

Table 138. MCFR register description (continued)

Bit	Name	Function
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(2): 000: blanking window = off 001: blanking window = 0.5µs 010: blanking window = 1µs 011: blanking window = 1.5µs 100: blanking window = 2µs 101: blanking window = 2.5µs 110: blanking window = 3µs 111: blanking window = 3.5µs The filter blanks the output of the current comparator.

1. Sampling is done at $f_{PERIPH}/4$.
2. Times are indicated for 4 MHz f_{PERIPH}.

Motor D event filter register (MDFR)

MDFR Reset value: 0000 1111 (0Fh)

7	6	5	4	3	2	1	0
DEF[3:0]				DWF[3:0]			
R/W				R/W			

Table 139. MDFR register description

Bit	Name	Function
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. See *Table 140*.
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. See *Table 141*.

Table 140. D event filter setting(1)

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	

Table 140. D event filter setting(1) (continued)

DEF3	DEF2	DEF1	DEF0	D event samples	SR = 1
1	0	0	1	10	No D event filter
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	

1. Sampling is done at the selected f_{SCF} frequency.

Table 141. D window filter setting(1)

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	

1. Times are indicated for 4 MHz f_{PERIPH}.

Reference register (MREF)

MREF Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
HST	CL	CFAV	HFE[1:0]			HFRQ[2:0]	
R/W	R/W	R/W	R/W			R/W	

Table 142. MREF register description

Bit	Name	Function
7	HST	Hysteresis comparator value This read only bit contains the hysteresis comparator output. 0: Demagnetization/BEMF comparator is under V_{REF} 1: Demagnetization/BEMF comparator is above V_{REF}
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}
5	CFAV	Current feedback amplifier entry validation 0: OAZ(MCCFI1) is the current comparator entry 1: MCCFI0 is the current comparator entry
4:3	HFE[1:0]	Chopping mode selection These bits select the chopping mode: 00: Chopping mode = off 01: Chopping mode = on low channels only 10: On high channels only 11: Both high and low channels
2:0	HFRQ[2:0]	Chopper frequency selection These bits select the chopping frequency (see *Table 143*).

Table 143. Chopping frequency selection(1)

HFRQ2	HFRQ1	HFRQ0	Chopping frequency	
			F_{mtc} = 16MHz F_{mtc} = 8MHz	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

1. The chopper signal has a 50% duty cycle.

PWM control register (MPCR)

MPCR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
PMS	OVFU	OVFV	OVFW	CMS	PCP[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W		

Table 144. MPCR register description

Bit	Name	Function
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.
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 until the next one. This bit is reset once transferred to the active register on update event.
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 until the next one. This bit is reset once transferred to the active register on update event
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 until the next one. This bit is reset once transferred to the active register on update event.
3	CMS	PWM counter mode selection 0: Edge-aligned mode 1: Center-aligned mode
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. The resulting frequency of the PWM counter input clock is shown below: 000: PWM counter input clock = F_{mtc} 001: PWM counter input clock = $F_{mtc}/2$ 010: PWM counter input clock = $F_{mtc}/3$ 011: PWM counter input clock = $F_{mtc}/4$ 100: PWM counter input clock = $F_{mtc}/5$ 101: PWM counter input clock = $F_{mtc}/6$ 110: PWM counter input clock = $F_{mtc}/7$ 111: PWM counter input clock = $F_{mtc}/8$

Repetition counter register (MREP)

MREP Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
REP[7:0]							
R/W							

Table 145. MREP register description

Bit	Name	Function
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 (that is, 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 (that is, 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 123*, (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)

MCPWH Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CPWH[7:0							
R/W							

Table 146. MCPWH register description

Bit	Name	Function
7:0	CPWH[7:0]	Most significant byte of phase W preload value

Compare phase W preload register low (MCPWL)

MCPWL Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CPWL[7:3]					Reserved		
R/W					-		

Table 147. MCPWL register description

Bit	Name	Function
7:5	CPWL[7:3]	Low bits of phase W preload value
2:0	-	Reserved

Compare phase V preload register high (MCPVH)

MCPVH Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CPVH[7:0]							
R/W							

Table 148. MCPVH register description

Bit	Name	Function
7:0	CPVH[7:0]	Most significant byte of phase V preload value

Compare phase V preload register low (MCPVL)

MCPVL Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CPVL[7:3]					Reserved		
R/W					-		

Table 149. MCPVL register description

Bit	Name	Function
7:5	CPVL[7:3]	Low bits of phase V preload value
2:0	-	Reserved

Compare phase U preload register high (MCPUH)

MCPUH Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CPVu[7:0]							
R/W							

Table 150. MCPUH register description

Bit	Name	Function
7:0	CPVu[7:0]	Most significant byte of phase U preload value

Compare phase U preload register low (MCPUL)

MCPUL Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CPUL[7:3]					Reserved		
R/W					-		

Table 151. MCPUL register description

Bit	Name	Function
7:5	CPUL[7:3]	Low bits of phase U preload value
2:0	-	Reserved

Compare 0 preload register high (MCP0H)

MCP0H Reset value: 0000 1111 (0Fh)

7	6	5	4	3	2	1	0
Reserved					CP0H[3:0]		
-					R/W		

Table 152. MCP0H register description

Bit	Name	Function
7:4	-	Reserved
3:0	CP0H[3:0]	Most significant bits of compare 0 preload value

Compare 0 preload register low (MCP0L)

MCP0L Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
CP0L[7:0]							
R/W							

Table 153. MCP0L register description

Bit	Name	Function
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 PWM operating mode on page 254 for details.

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, the value in the preload register is written. However, while reading at the same address, the current value in the register and not the value of the preload register is obtained.
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 first accessed during the set-up. Any access to the other bits (not write-once) through a BRES or a BSET instruction locks the content of write-once bits (no possibility for the core to 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.

Deadtime generator register (MDTG)

MDTG — Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
PCN	DTE	DTG[5:0]					
R/W	(1)	Write once only					

1. Write once-only bit if PCN bit is set, read/write if PCN bit is reset.

Table 154. MDTG register description

Bit	Name	Function
7	PCN	Number of PWM channels 0: Only PWM U signal is output to the PWM manager for six-step mode motor control (example, PM BLDC motors) 1: The three PWM signals U, V and W are output to the channel manager (example, for three-phase sinewave generation)
6	DTE	Deadtime generator enable(1) 0: Disable the deadtime generator 1: Enable the deadtime generator and apply complementary PWM signal to the adjacent switch
5:0	DTG[5:0]	Deadtime generator set-up(2) These bits set-up the deadtime duration and resolution. Refer to *Table 110 on page 247* for details. With F_{mtc} = 16 MHz, deadtime values range from 125ns to 16µs with steps of 125ns, 250ns and 500ns.

1. 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.
2. Write-once bits; once write-accessed these bits cannot be rewritten unless the processor is reset (see "Caution: Access to write-once bits" on *page 280*).

Table 155. Deadtime generator set-up(1)

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

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 127: Output configuration summary on page 267*.

When the PCN bit is reset (example, 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 follows the instructions detailed in *Table 111: Deadtime generator outputs on page 249* 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 is 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 134: PWM mode when SR = 1 on page 269*.

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.

Polarity register (MPOL)

MPOL — Reset value: 0011 1111 (3Fh)

7	6	5	4	3	2	1	0
ZVD	REO	OP[5:0]					
R/W	R/W	Write once only					

Table 156. MPOL register description

Bit	Name	Function
7	ZVD	Z vs D edge polarity 0: Zero-crossing and End of Demagnetization have opposite edges 1: Zero-crossing and End of Demagnetization have same edge
6	REO	Read on High or Low channel bit(1) 0: Read the BEMF signal on High channels 1: Read on Low channels
5:0	OP[5:0]	Output channel polarity(2) These bits are used together with the OO[5:0] bits in the MPHST register to control the output channels (see *Table 157*). 0: Output channel is Active Low 1: Output channel is Active High

1. *This bit always has to be configured whatever the sampling method.*
2. Write-once bits; once write-accessed these bits cannot be rewritten unless the processor is reset (see 'Caution: Access to write-once bits' on *page 280*).

Table 157. 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.**

PWM register (MPWME)

MPWME Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
DG	PWMW	PWMV	PWMU	OT[3:0]			
R/W	R/W	R/W	R/W	R/W			

Table 158. MPWME register description

Bit	Name	Function
7	DG	Debug option This bit is used to enter debug mode. As a result, C, D and Z events are output on two pins MCDEM and MCZEM in Switched and Autoswitched mode, C and D events are output in Speed Measurement mode. Refer to *Debug option on page 219* for more details. 0: Normal mode 1: Debug mode
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
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
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
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 159*.

Table 159. 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 time in sensorless mode (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	

Table 159. OFF time bits (continued)

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 mode (SPLG = 1 and/or DS[3:0] bits)
1	0	0	1	25 µs	No minimum off -time
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}.

Configuration register (MCONF)

MCONF Reset value: 0000 0010 (02h)

7	6	5	4	3	2	1	0
DS[3:0]				SOI	SOM	XT16:XT8	
R/W				R/W	R/W	R/W	

Table 160. MCONF register description

Bit	Name	Function
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(1): 0000: Delay added to sample at T_{ON} = no delay added, sample during T_{OFF} 0001: Delay added to sample at T_{ON} = 2.5µs 0010: Delay added to sample at T_{ON} = 5µs 0011: Delay added to sample at T_{ON} = 7.5µs 0100: Delay added to sample at T_{ON} = 10µs 0101: Delay added to sample at T_{ON} = 12.5µs 0110: Delay added to sample at T_{ON} = 15µs 0111: Delay added to sample at T_{ON} = 17.5µs 1000: Delay added to sample at T_{ON} = 20µs 1001: Delay added to sample at T_{ON} = 22.5µs 1010: Delay added to sample at T_{ON} = 25µs 1011: Delay added to sample at T_{ON} = 27.5µs 1100: Delay added to sample at T_{ON} = 30µs 1101: Delay added to sample at T_{ON} = 32.5µs 1110: Delay added to sample at T_{ON} = 35µs 1111: Delay added to sample at T_{ON} = 37.5µs

Table 160. MCONF register description (continued)

Bit	Name	Function
3	SOI	Sampling out interrupt flag This interrupt indicates that the sampling that should have been done during Ton has occurred during the next Toff. In this case, the sample is discarded. 0: No sampling out interrupt pending 1: Sampling out interrupt pending
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.*
1:0	XT16:XT8	BLDC drive motor control peripheral input frequency selection 00: f_{PERIPH} (peripheral frequency) = f_{MTC} 01: f_{PERIPH} (peripheral frequency) = $f_{MTC}/2$ 10: f_{PERIPH} (peripheral frequency) = $f_{MTC}/4$ 11: f_{PERIPH} (peripheral frequency) = $f_{MTC}/4$ (same as XT16 = 1, XT8 = 0) **Caution**: It is recommended to set the peripheral frequency to 4 MHz. Setting $f_{PERIPH} = f_{MTC}$ is used mainly when f_{clk} = 4 MHz (for low power consumption).

1. Times are indicated for 4 MHz f_{PERIPH}.

Parity register (MPAR)

MPAR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
TES[1:0]		OE[5:0](1)					
R/W		R/W					

1. Preload bits, new value taken into account at the next C 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 *Control register A (MCRA) on page 265*. The use of a preload register allows all the registers to be updated at the same time.

Table 161. MPAR register description

Bit	Name	Function
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). See *Table 162*.
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).*

Table 162. Tacho edges and input mode selection

TES 1	TES 0	Edge sensitivity	Operating mode
0	0	-	Position sensor or sensorless
0	1	Rising edge	Speed sensor
1	0	Falling edge	
1	1	Rising and falling edges	

Motor Z event filter register (MZFR)

MZFR Reset value: 0000 1111 (0Fh)

7	6	5	4	3	2	1	0
ZEF[3:0]				ZWF[3:0]			
R/W				R/W			

Table 163. MZFR register description

Bit	Name	Function
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 164*) or at PWM frequency.
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 (see *Table 165*).

Table 164. 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

Table 164. Z event filter setting (continued)

ZEF3	ZEF2	ZEF1	ZEF0	Z event samples
1	1	1	0	15
1	1	1	1	16

Table 165. Z window filter setting(1)

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	

1. Times are indicated for 4 MHz f_{PERIPH}

Motor sampling clock register (MSCR)

MSCR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
ZSV	Reserved			SCF[1:0]		ECM	DISS
R/W	-			R/W		R/W	R/W

Table 166. MSCR register description

Bit	Name	Function
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
6:4	-	Reserved, must be kept cleared
3:2	SCF[1:0]	Sampling clock frequency These bits select the sampling clock frequency (f_{SCF}) used to count D and Z events(1): 00: f_{SCF} = 1 MHz (every 1µs) 01: f_{SCF} = 500 kHz (every 2µs) 10: f_{SCF} = 250 kHz (every 4µs) 11: f_{SCF} = 125 kHz (every 8µs)
1	ECM	Encoder capture mode This bit is used to select the source of events which trigger the capture of the [MTIM:MTMLR] counter when using Encoder speed sensor (see *Figure 91*). 0: Real Time Clock interrupts 1: Read access on MTIM register
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

1. Times are indicated for 4 MHz f_{PERIPH}.

Figure 124. General view of the MTC for PM BLDC motor control

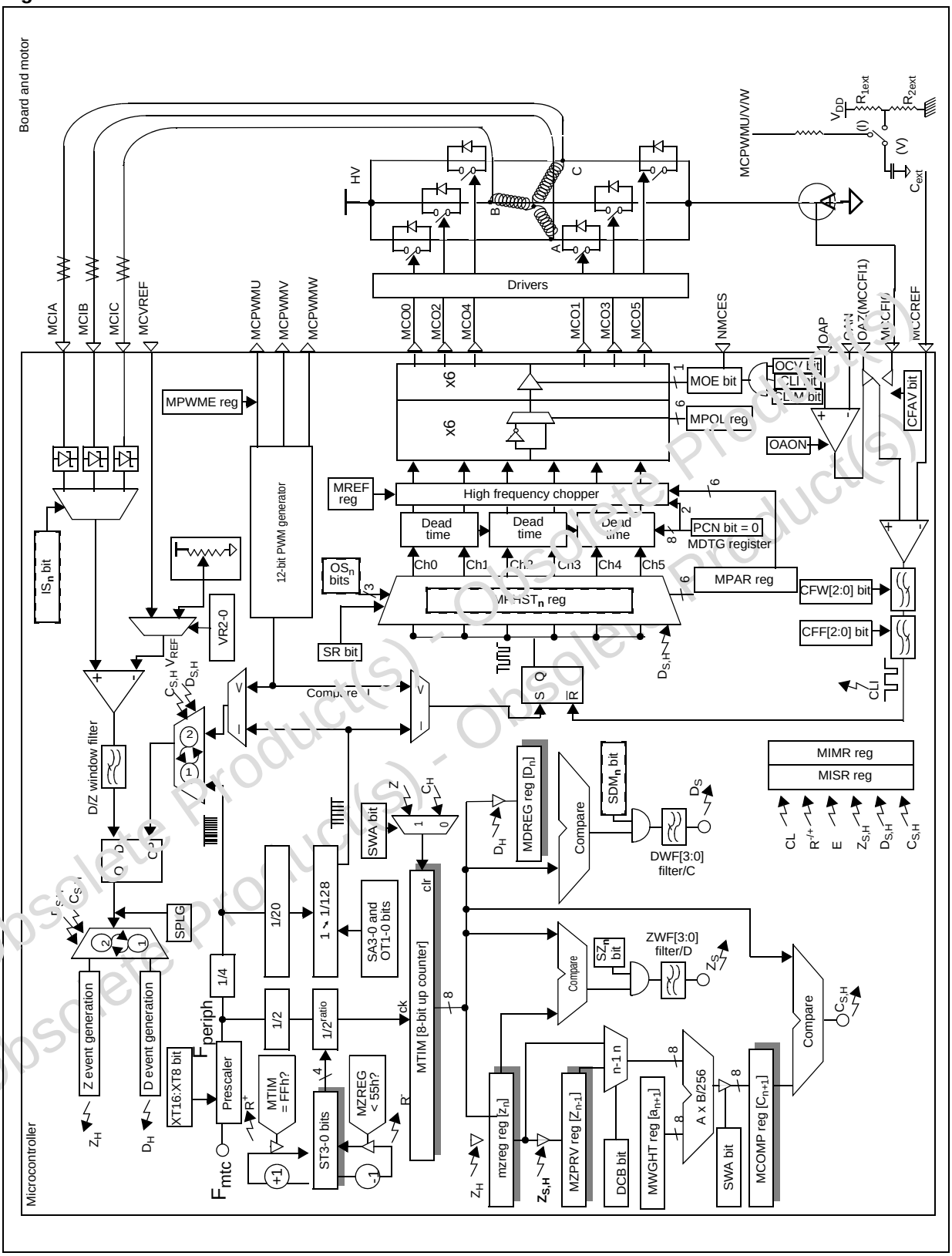


Figure 125. General view of the MTC configured for Induction motor control (proposal)

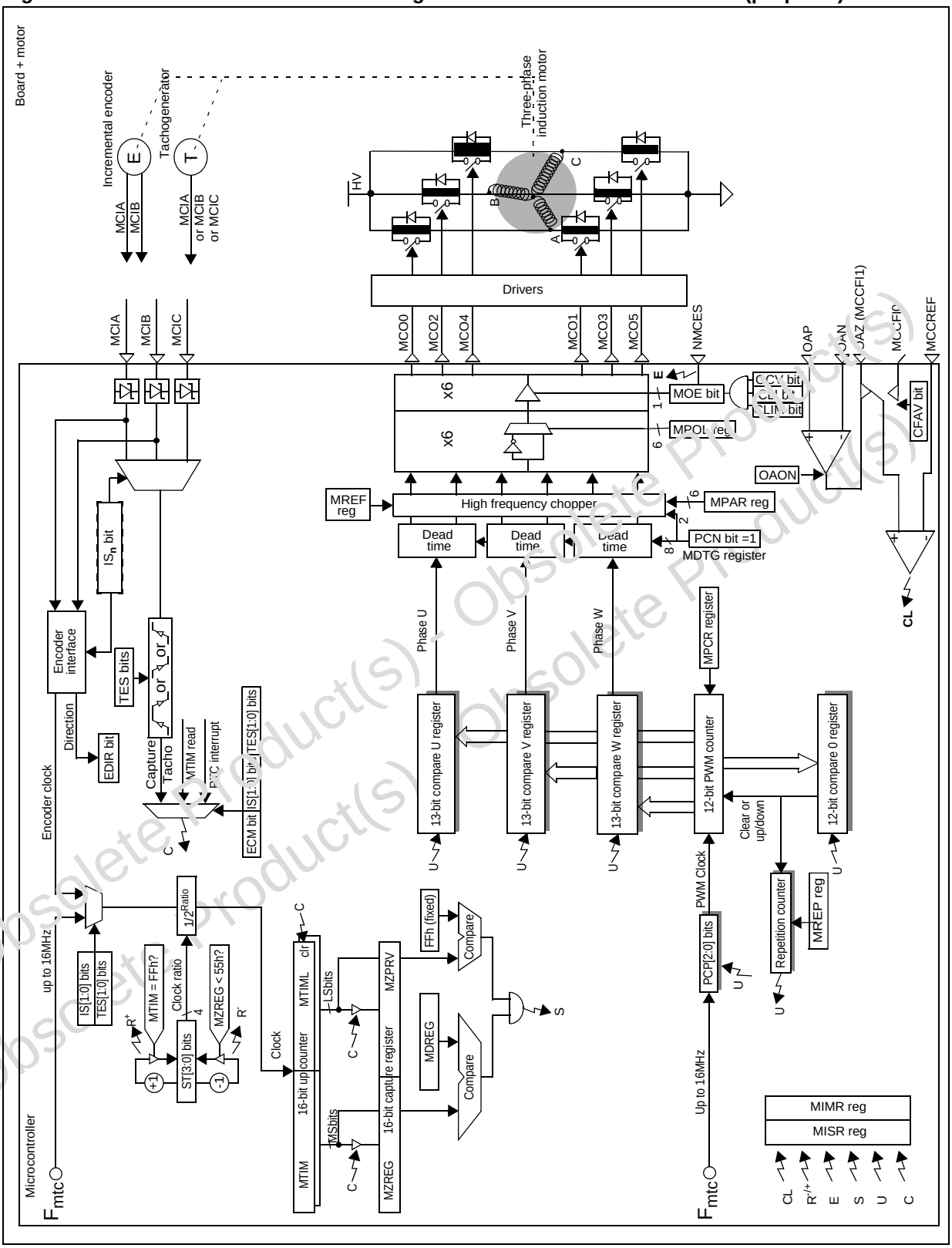


Table 167. MTC page 0 register map and reset values

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
MCPWL Reset value	CPWL7 0	CPWL6 0	CPWL5 0	CPWL4 0	CPWL3 0	0	0	0

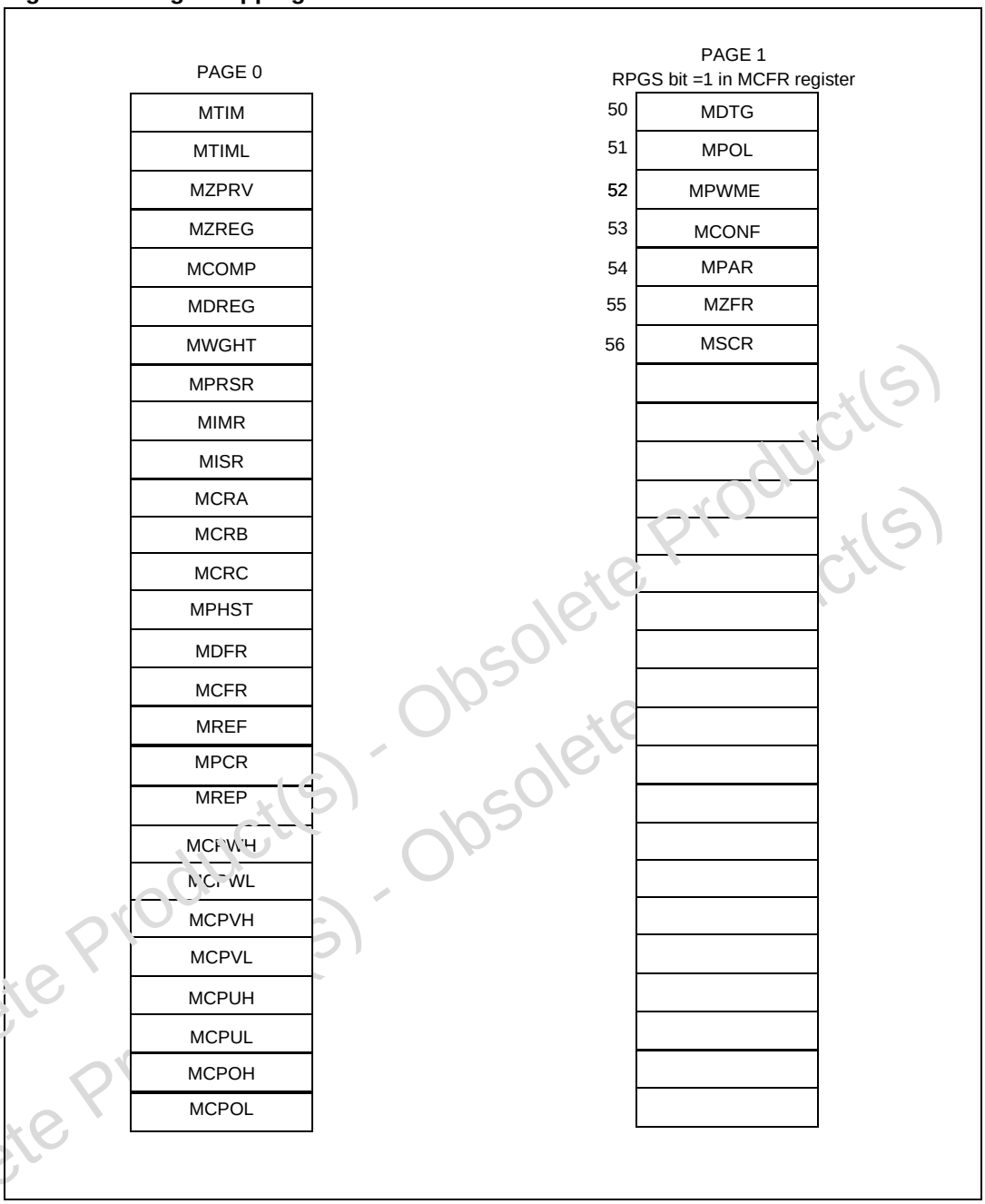
Table 167. MTC page 0 register map and reset values (continued)

Register name	7	6	5	4	3	2	1	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 168. 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

Figure 126. Page mapping for motor control



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 three 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

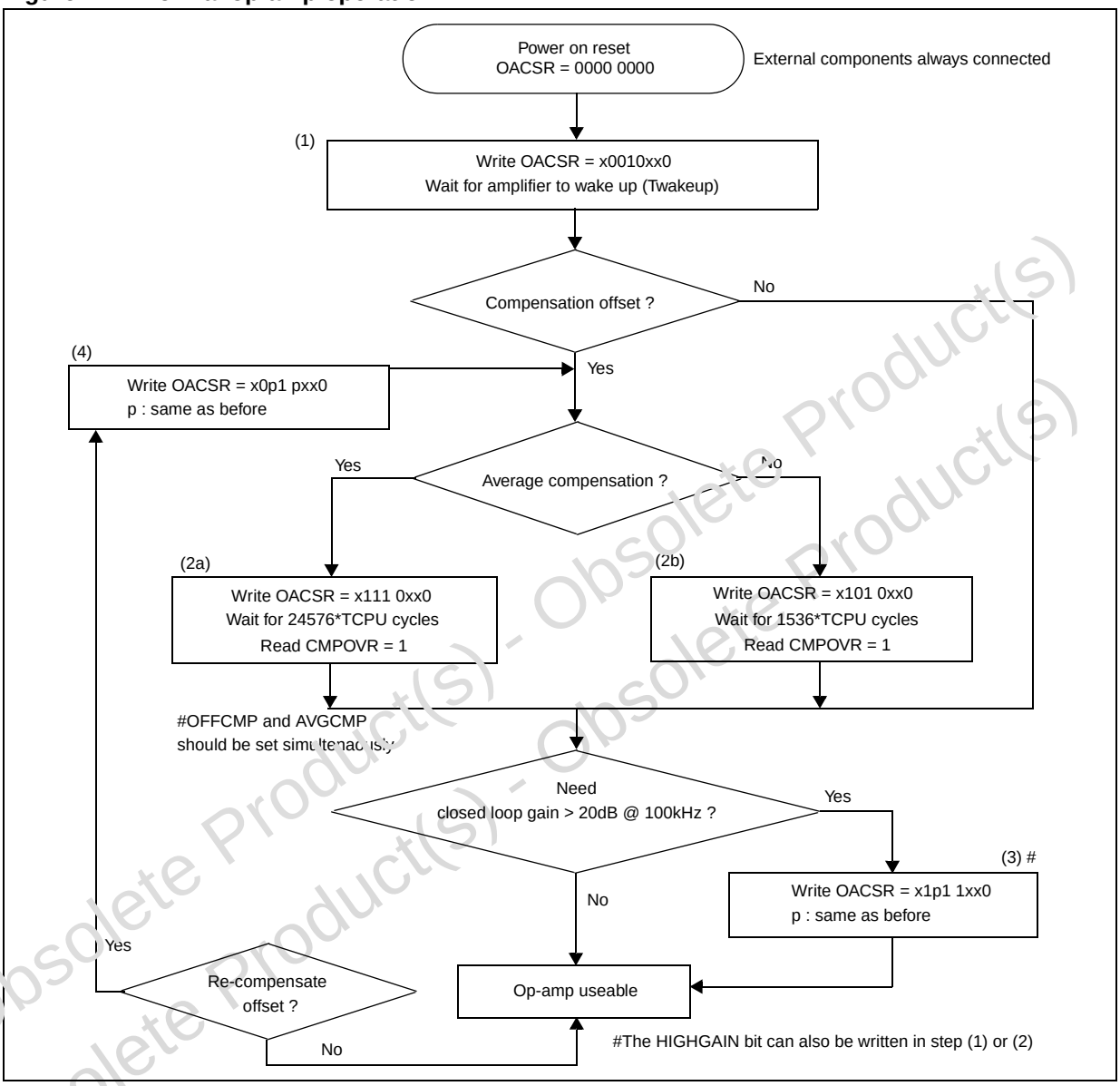
The op-amp incorporates a method to minimize the input offset which is dependant on process lot. It is useable by setting the OFFCMP bit of the control register, which launch the compensation cycle. The CMPVR bit is set by hardware as soon as this cycle is completed. The compensation is valid as long 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.

10.7.5 Op-amp programming

The flowchart for op-amp operation is shown in *Figure 127*.

Figure 127. Normal op-amp operation



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.

Table 169. Effect of low power modes on op-amp

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 *Section 12: Electrical characteristics*)

10.7.7 Interrupts

None.

10.7.8 Register description

Control/status register (OACSR)

OASCR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
CMPOVR	OFFCMP	AVGCMP	OAON	HIGH GAIN	Reserved		
RO	R/W	R/W	R/W	R/W	-		

Table 170. OACSR register description

Bit	Name	Function
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
6	OFFCMP	Offset compensation 0: Reset offset compensation values 1: Request to start offset compensation
5	AVGCMP	Average compensation 0: One-shot offset compensation 1: Average offset compensation over 16 times
4	OAON	Amplifier on 0: Op-amp powered off 1: Op-amp on

Table 170. OACSR register description (continued)

Bit	Name	Function
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
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 two 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 128*.

10.8.3 Functional description

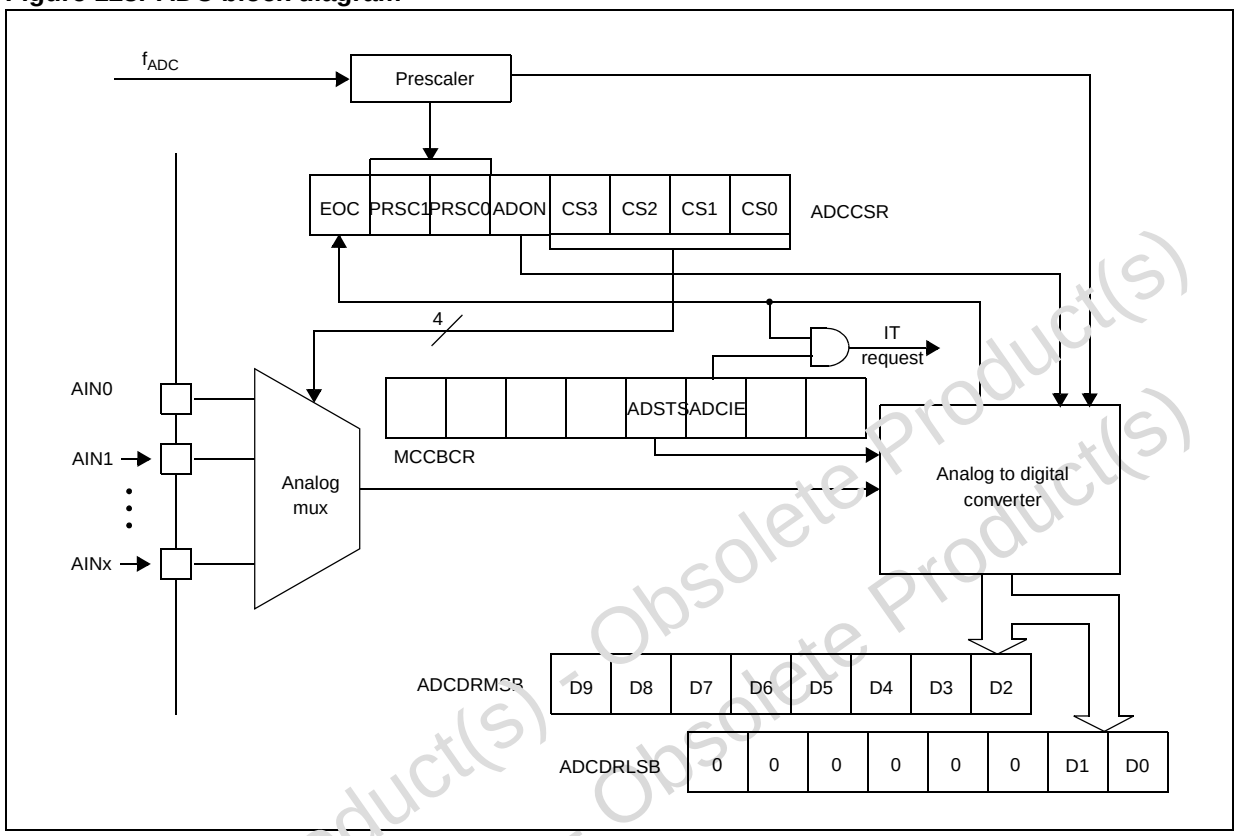
Analog references

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

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 128. ADC block diagram



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 *Section 12: Electrical characteristics*.

R_{AIN} is the maximum recommended impedance for an analog input signal. If the impedance is too high, there is a loss of accuracy due to leakage and sampling not being completed in the allotted time.

R_{REF} is the value of the resistive bridge implemented in the device between V_{REF+} and V_{REF-}.

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 stops the current conversion and starts conversion of the newly selected channel.

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 6.6.7: MCC control status register (MCCSR) on page 56*).
- 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 is not 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.

Table 171. Effect of low power modes on A/D converter

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 *Section 12: Electrical characteristics*) before accurate conversions can be performed.

10.8.5 Interrupts

Table 172. A/D converter interrupt control/wake-up capability

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 6.6.7: MCC control status register (MCCSR) on page 56*)

10.8.6 Register description

Control/status register (ADCCSR)

ADCCSR Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
EOC	PRSC[1:0]		ADON	CS[3:0]			
RO	R/W		R/W	R/W			

Table 173. ADCCSR register description

Bit	Name	Function
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
6:5	PRSC[1:0]	ADC clock prescaler selection These bits are set and cleared by software: 00: f_{ADC} = 4MHz 01:f_{ADC} = 2MHz 10: f_{ADC} = 1MHz

Table 173. ADCCSR register description (continued)

Bit	Name	Function
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
3:0	CS[3:0]	Channel selection(1) These bits are set and cleared by software. They select the analog input to convert: 0000: channel pin = AIN0 0001: channel pin = AIN1 0010: channel pin = AIN2 0011: channel pin = AIN3 0100: channel pin = AIN4 0101: channel pin = AIN5 0110: channel pin = AIN6 0111: channel pin = AIN7 1000: channel pin = AIN8 1001: channel pin = AIN9 1010: channel pin = AIN10 1011: channel pin = AIN11 1100: channel pin = AIN12 1101: channel pin = AIN13 1110: channel pin = AIN14 1111: channel pin = AIN15

1. The number of channels is device dependent. Refer to *Section 2: Pin description*.

Data register (ADCDRMSB)

ADCDRMSB Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
D[9:2]							
RO							

Table 174. ADCDRMSB register description

Bit	Name	Function
7:0	D[9:2]	MSB of analog converted value This register contains the MSB of the converted analog value.

Data register (ADCDRLSB)

ADCDRLSB Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved						D[1:0]	
-						RO	

Table 175. ADCDRLSB register description

Bit	Name	Function
7:2	-	Reserved. Forced by hardware to 0.
1:0	D[1:0]	LSB of analog converted value This register contains the LSB of the converted analog value.

Table 176. ADC register map and reset values

Address (Hex.)	Register label	7	6	5	4	3	2	1	0
2E	ADCCSR Reset Value	EOC 0	PRSC1 0	PRSC0 0	ADON 0	CS3 0	CS2 0	CS1 0	CS0 0
2F	ADCDRMSB Reset Value	D9 0	D8 0	D7 0	D6 0	D5 0	D4 0	D3 0	D2 0
30	ADCDRLSB Reset Value	0 0	0 0	0 0	0 0	0 0	0 0	D1 0	D0 0

11 Instruction set

11.1 CPU addressing modes

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

Table 177. CPU addressing mode 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 178. 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

Table 178. CPU addressing mode overview (continued)

Mode			Syntax	Destination	Pointer address (Hex.)	Pointer size (Hex.)	Length (bytes)
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

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.

Table 179. Inherent instructions

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.

Table 180. Immediate instructions

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 sub-modes:

- Direct (short)

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

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

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.

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 181. 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

Table 182. Short instructions and functions

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.

Table 183. Relative direct and indirect instructions

Available relative direct/indirect instructions	Function
JRxx	Conditional jump
CALLR	Call relative

The relative addressing mode consists of two sub-modes:

- Relative (direct)
  The offset is following the opcode.
- Relative (indirect)
  The offset is defined in memory, the address of which follows the opcode.

11.2 Instruction groups

11.2.1 Introduction

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:

Table 184. Instruction groups

Group	Instructions							
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 prebyte

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.

11.2.2 Illegal opcode reset

In order to provide enhanced robustness to the device against unexpected behavior, 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.

Table 185. Instruction set overview

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 >=								

Table 185. Instruction set overview (continued)

Mnemo	Description	Function/example	Dst	Src	I1	H	I0	N	Z	C
JRUGT	Jump if (C + Z = 0)	Unsigned >								
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 and 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±3Σσ).

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.

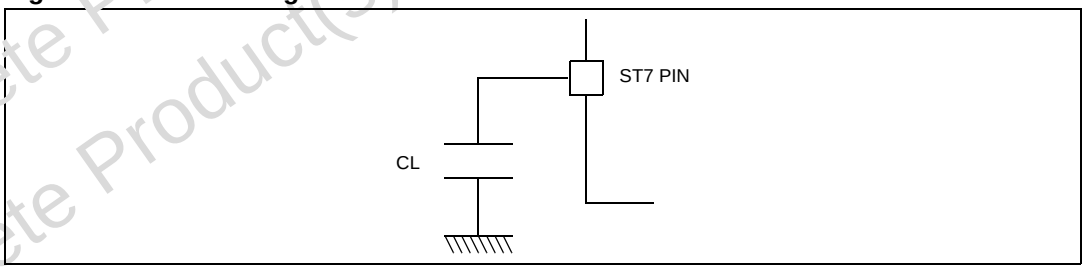
12.1.3 Typical curves

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 129*.

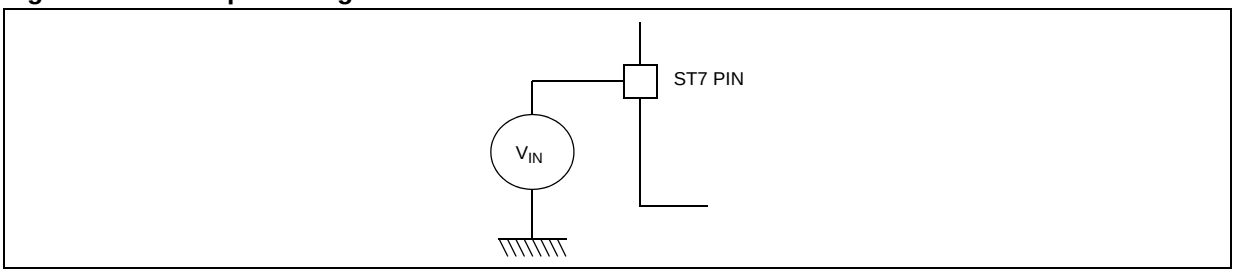
Figure 129. Pin loading conditions



12.1.5 Pin input voltage

The input voltage measurement on a pin of the device is described in *Figure 130*.

Figure 130. 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

Table 186. 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	
$\|\Delta V_{DDx}\|$ and $\|\Delta V_{SSx}\|$	Variations between different digital power pins	50	mV
$\|V_{SSA}$ - $V_{SSx}\|$	Variations between digital and analog ground pins	50	
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human body model)	See *Section 12.7.3: Absolute maximum ratings (electrical sensitivity) on page 327*	
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine model)		

1. Directly connecting the $\overline{RESET}$ 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 $\overline{RESET}$, 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 $V_{IN} < V_{SS}$.

12.2.2 Current characteristics

Table 187. Current characteristics

Symbol	Ratings		Max. value	Unit
I_{VDD}	Total current into V_{DD} power lines (source)(1)	32-pin devices	75	mA
		44-pin devices	125	
I_{VSS}	Total current out of V_{SS} ground lines (sink)(1)	32-pin devices	75	
		44-pin devices	125	
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)(3)	Injected current on V_{PP} pin		± 5	
	Injected current on $\overline{RESET}$ pin		± 5	
	Injected current on OSC1 and OSC2 pins		± 5	
	Injected current on any other pin(4)		± 5	
$\Sigma I_{INJ(PIN)}$(2)	Total injected current (sum of all I/O and control pins)(4)		± 20	

1. All power (V_{DD}) and ground (V_{SS}) lines must always be connected to the external supply.
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. Negative injection disturbs the analog performance of the device. See *Table 221: ADC accuracy with VDD = 5.0V on page 350* and *Note 1 on page 351*.
   For best reliability, it is recommended to avoid negative injection of more than 1.6mA
4. 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 characterization with $\Sigma I_{INJ(PIN)}$ maximum current injection on four I/O port pins of the device.

12.2.3 Thermal characteristics

Table 188. Thermal characteristics

Symbol	Ratings	Value	Unit
T_{STG}	Storage temperature range	-65 to +150	°C
T_J	Maximum junction temperature (see *Section 13.2.3: Thermal characteristics on page 355*)		

12.3 Operating conditions

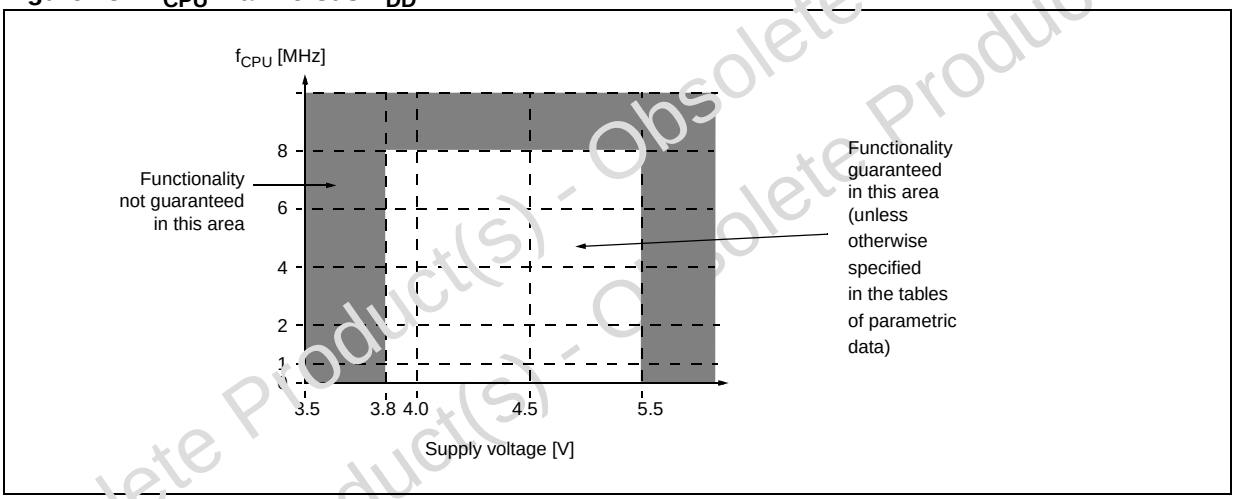
12.3.1 General operating conditions

Table 189. 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	A suffix version	-40	85	°C
		C suffix version	-40	125	

1. Clock detector, ADC, comparator and Op-amp functionalities guaranteed only within 4.5 to 5.5V voltage range.

Figure 131. f_{CPU} max versus V_{DD}



Note: Some temperature ranges are only available with a specific package and memory size. Refer to Section 14.2: Device ordering information and transfer of customer code on page 358.

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

12.3.2 Operating conditions with low voltage detector (LVD)

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

Table 190. Operating conditions with LVD

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

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.

Table 191. AVD thresholds

Symbol	Parameter	Conditions	Min	Typ	Max(1)	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(2)	$V_{IT+(AVD)}$-$V_{IT-(AVD)}$		200		mV
ΔV_{IT-}	Voltage drop between AVD flag set and LVD reset activated	$V_{IT-(AVD)}$-$V_{IT-(LVD)}$		450		mV

1. See *Section 15.7: Maximum values of AVD thresholds on page 366*
2. Data based on characterization results, not tested in production.

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)

Table 192. Run and slow modes (Flash devices)

Symbol	Parameter	Conditions		Typ	Max(1)	Unit
I_{DD}	Supply current in Run mode(2) (see *Figure 132*)	$4.5V \le V_{DD} \le 5.5V$	f_{OSC} = 16 MHz, f_{CPU} = 8 MHz	12	18	mA
	Supply current in Slow mode(2) (see *Figure 133*)		f_{OSC} = 16 MHz, f_{CPU} = 500 kHz	5	8	

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.

Figure 132. Typical I_{DD} in RUN vs f_{CPU}

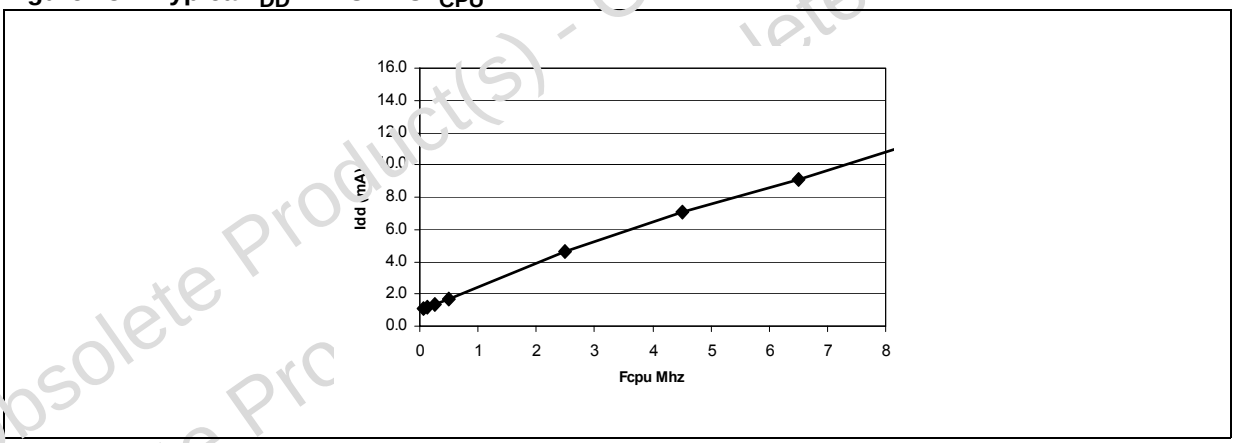
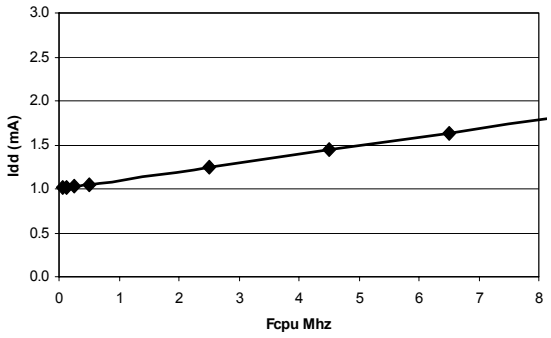


Figure 133. Typical I_{DD} in SLOW vs f_{CPU}



12.4.2 Wait and Slow Wait modes

Table 193. Wait and Slow Wait modes

Symbol	Parameter	Conditions		Typ	Max(1)	Unit
I_{DD}	Supply current in Wait mode(2) (see *Figure 134*)	4.5V ≤ V_{DD} ≤ 5.5 V	f_{OSC} = 16 MHz, f_{CPU} = 8 MHz	8	12	mA
	Supply current in Slow Wait mode(2) (see *Figure 135*)		f_{OSC} = 16 MHz, f_{CPU} = 500 kHz	3.5	5	

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.

Figure 134. Typical I_{DD} in Wait vs f_{CPU}

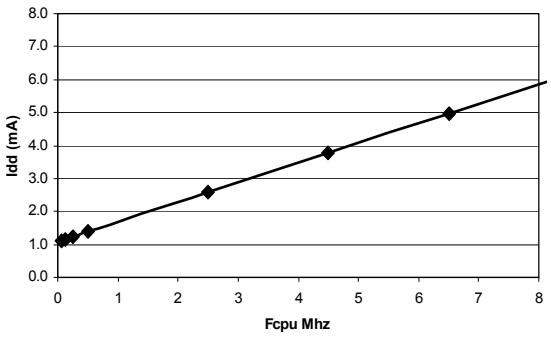
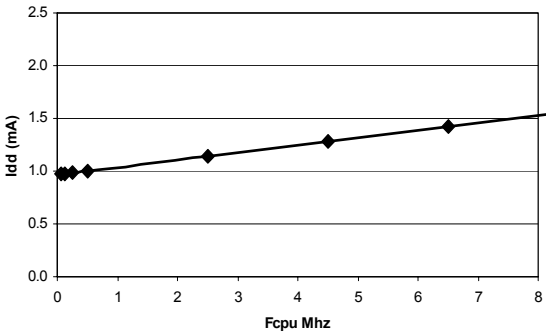


Figure 135. Typical I_{DD} in Slow Wait vs f_{CPU}



12.4.3 Halt and Active Halt modes

Table 194. 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)	16 MHz 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).

Table 195. Supply and clock managers

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		

12.4.5 On-chip peripherals

Table 196. On-chip peripherals

Symbol	Parameter	Conditions		Typ	Unit
$I_{DD(TIM)}$	16-bit Timer supply current(1)	f_{CPU} = 8 MHz	V_{DD} = 5.0V	50	µA
$I_{DD(ART)}$	ART PWM supply current(2)			75	
$I_{DD(SPI)}$	SPI supply current(3)			400	
$I_{DD(SCI)}$	SCI supply current(4)			400	
$I_{DD(MTC)}$	MTC supply current(5)			500	
$I_{DD(ADC)}$	ADC supply current when converting(6)	f_{ADC} = 4 MHz		400	
$I_{DD(op-amp)}$	Op-amp supply current(7)	f_{CPU} = 8 MHz		1500	

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 between 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 (op-amp 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

Table 197. General timings

Symbol	Parameter	Conditions	Min	Typ(1)	Max	Unit
$t_{c(INST)}$	Instruction cycle time		2	3	12	t_{CPU}
		f_{CPU} = 8 MHz	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} = 8 MHz	1.25		2.75	µs

1. Data based on typical application software
2. Time measured between interrupt event and interrupt vector fetch. $\Delta t_{c(INST)}$ is the number of t_{CPU} cycles needed to finish the current instruction execution.

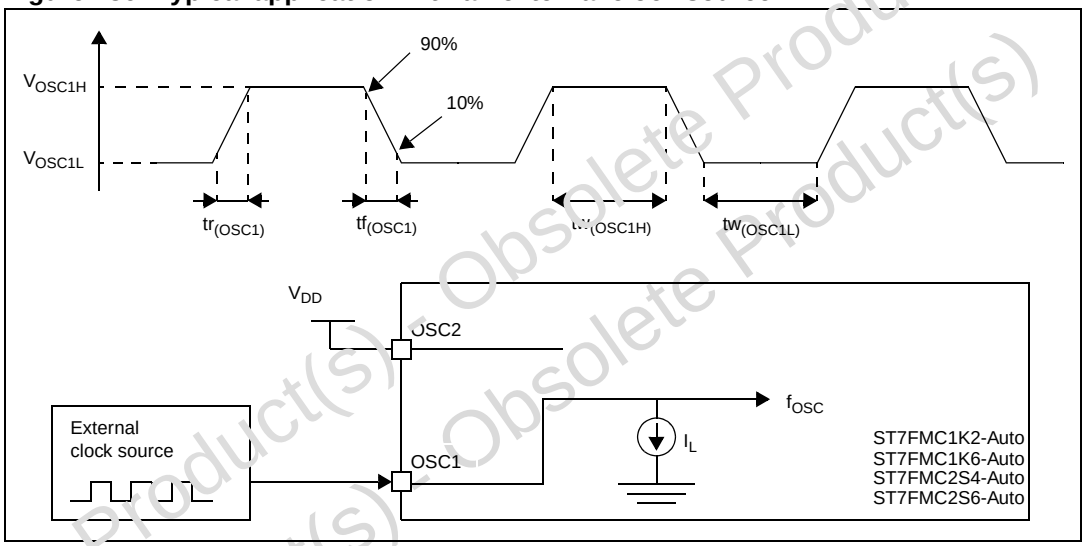
12.5.2 External clock source

Table 198. External clock source

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{OSC1H}	OSC1 input pin high level voltage	See *Figure 136*	0.7 x V_{DD}		V_{DD}	V
V_{OSC1L}	OSC1 input pin low level voltage		V_{SS}		0.3 x V_{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

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

Figure 136. Typical application with an external clock source



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...).

Table 199. Oscillator parameters

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 200* below			pF

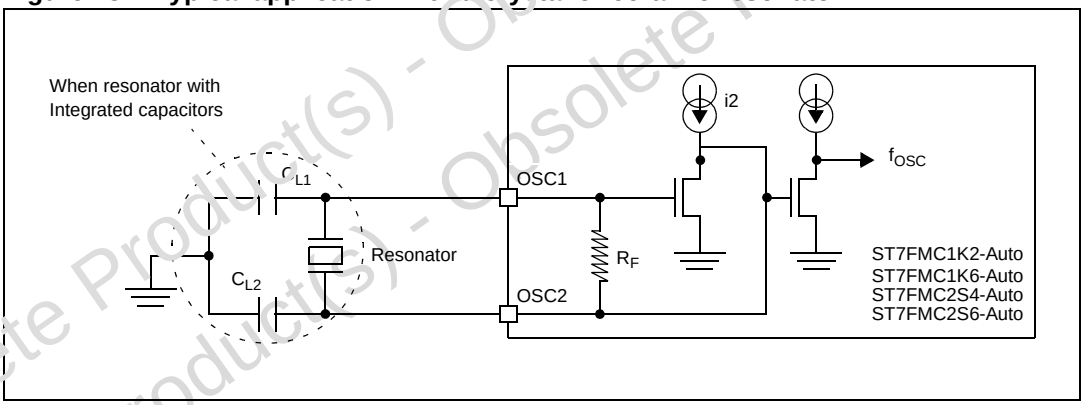
1. When PLL is used, please refer to *Table 201: PLL characteristics on page 323* and to *Section 6: Supply, reset and clock management on page 42* (f_{OSC} min. is 8 MHz with PLL).

Table 200. Examples of recommended references

Supplier	f_{OSC} (MHz)	Typical ceramic resonators(1)	CL1 [pF]	CL2 [pF]
		Reference		
Murata	4	CSTCR4M00G53-R0	(15)	(15)
	8	CSTCE8M00G52-R0	(10)	(10)
	16	CSTCE16M0V53-R0	(15)	(15)

1. Resonator characteristics given by the ceramic resonator manufacturer. For more information on these resonators, please consult www.murata.com

Figure 137. Typical application with a crystal or ceramic resonator



12.5.4 Clock security system with PLL

Table 201. PLL characteristics

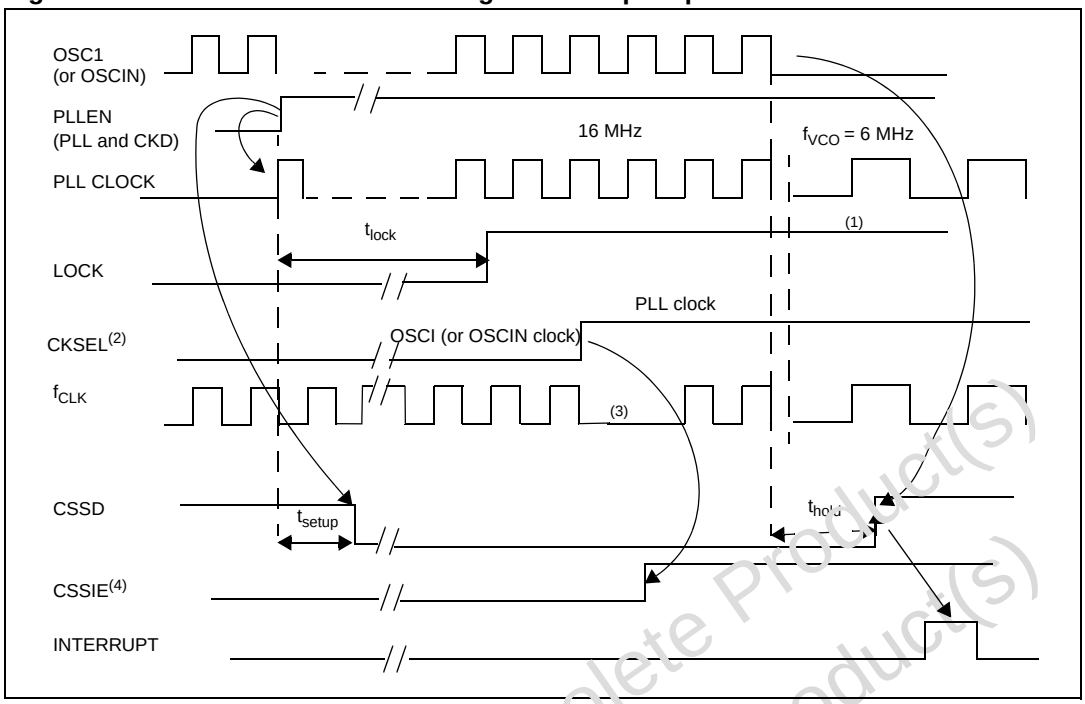
Symbol	Parameter	Min	Typ	Max	Unit
f_{OSC}	PLL input frequency range	7		8	MHz
Output frequency	Output frequency when the PLL attains lock		16		MHz
t_{Lock}	PLL lock time (Locked = 1)		50	100	µs
Jitter	Jitter in the output clock		2		%
f_{CPU}	CPU clock frequency when VCO is connected to ground (ICD internal clock or back up oscillator)		3		MHz

Table 202. Clock detector characteristics

Symbol	Parameter	Min	Typ	Max	Unit
f_{Detect}	Detected minimum input frequency			500(1)	kHz
t_{setup}	Time needed to detect OSC1 (or OSCIN) once CKD is enabled		3		µs
t_{hold}	Time needed to detect that OSC1 (or OSCIN) stops		3		µs

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

Figure 138. PLL and clock detector signal start up sequence



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

Table 203. RAM and hardware registers

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{RM}	Data retention mode(1)	Halt mode (or reset)	1.6			V

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.

12.6.2 Flash memory

Table 204. Dual voltage HDFlash memory

Symbol	Parameter	Conditions	Min(1)	Typ	Max(1)	Unit
f_{CPU}	Operating frequency	Read mode	0		8	MHz
		Write/Erase mode	1		8	
V_{PP}	Programming voltage(2)	$4.5V \leq V_{DD} \leq 5.5V$	11.4		12.6	V
I_{PP}	V_{PP} current(3)(4)	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	25			
		T_A = 125°C	10			
N_{RW}	Write erase cycles	T_A = 25°C	100			cycles
T_{PROG} T_{ERASE}	Programming or erasing temperature range		-40	25	85	°C

1. Data based on characterization results, not tested in production
2. V_{PP} must be applied only during the programming or erasing operation and not permanently for reliability reasons.
3. Data based on simulation results, not tested in production
4. In Write/Erase mode the I_{DD} supply current consumption is the same as in Run mode (*Section 12.4.1: Run and slow modes (Flash devices) on page 317*)

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 two 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.

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 behavior is detected, the software can be hardened to prevent unrecoverable errors occurring (see application note AN1015).

Table 205. EMS test results

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} = 8 MHz, LVD OFF, Conforms to IEC 1000-4-2	4A
			V_{DD} = 5V, T_A = +25°C, f_{OSC} = 8 MHz, 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} = 8 MHz, Conforms to IEC 1000-4-4	4A
		ROM devices	V_{DD} = 5V, T_A = +25°C, f_{OSC} = 8 MHz, Conforms to IEC 1000-4-4	3B

12.7.2 Electro magnetic interference (EMI)

Based on a simple application running on the product (toggling two 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.

Table 206. EMI emission(1)(2)

Symbol	Parameter	Conditions	Device/package	Monitored frequency band	Max vs. [f_{OSC}/f_{CPU}]		Unit
					8/4 MHz	16/8 MHz	
S_{EMI}	Peak level	V_{DD} - 5V, T_A = +25°C, conforming to SAE J 1752/3	Flash/LQFP64	0.1 MHz to 30 MHz	8	6	dBµV
				30 MHz to 130 MHz	8	12	
				130 MHz to 1 GHz	1	9	
				SAE EMI Level	1.5	2.5	-

1. *Data based on characterization results, not tested in production.*
2. *Refer to Application Note AN1709 for data on other package types.*

12.7.3 Absolute maximum ratings (electrical sensitivity)

Based on three different tests (ESD, LU and DLU) using specific measurement methods, the product is stressed in order to determine its performance in terms of electrical sensitivity. For more details, refer to the application note AN1181.

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.

Table 207. ESD absolute maximum ratings

Symbol	Ratings	Conditions	Maximum value(1)	Unit
$V_{ESD(HBM)}$	Electro-static discharge voltage (Human body model)	T_A = +25°C	2000	V
$V_{ESD(MM)}$	Electro-static discharge voltage (Machine model)	T_A = +25°C	200	
$V_{ESD(CDM)}$	Electro-static discharge voltage (Charged device model)	T_A = +25°C	250	

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

Static and dynamic latch-up

- **LU**: 3 complementary static tests are required on 10 parts to assess the latch-up performance. A supply overvoltage (applied to each power supply pin) and a current injection (applied to each input, output and configurable I/O pin) are performed on each sample. This test conforms to the EIA/JESD 78 IC latch-up standard. For more details, refer to the application note AN1181.
- **DLU**: Electro-static discharges (one positive then one negative test) are applied to each pin of three samples when the micro is running to assess the latch-up performance in dynamic mode. Power supplies are set to the typical values, the oscillator is connected as near as possible to the pins of the micro and the component is put in reset mode. This test conforms to the IEC1000-4-2 and SAEJ1752/3 standards. For more details, refer to the application note AN1181.

Table 208. Electrical sensitivities

Symbol	Parameter	Conditions	Class[1]
LU	Static latch-up class	T_A = +25°C T_A = +125°C	A A
DLU	Dynamic latch-up class	V_{DD} = 5.5V, f_{OSC} = 4 MHz, T_A = +25°C	A

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

12.8 I/O port pin characteristics

12.8.1 General characteristics

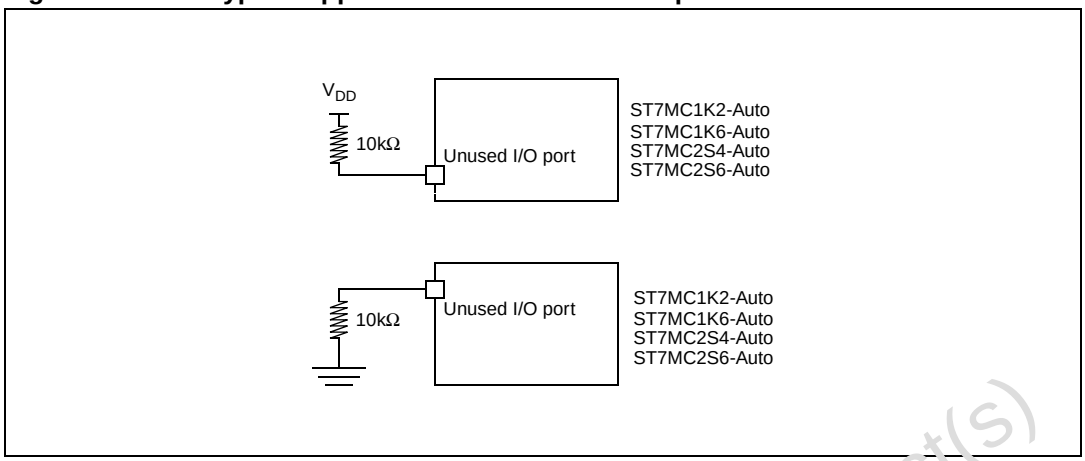
Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Table 209. General characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IL}	Input low level voltage	CMOS ports			0.3 x V_{DD}	V
V_{IH}	Input high level voltage		0.7 x V_{DD}			
V_{hys}	Schmitt trigger voltage hysteresis(1)			1		
V_{IL}	Input low level voltage	G and H ports			0.8	
V_{IH}	Input high level voltage		2.8			
V_{hys}	Schmitt trigger voltage hysteresis(1)			400		mV
$I_{INJ(PIN)}$(2)	Injected current on an I/O	V_{DD} = 5V			+5/-2	mA
$\Sigma I_{INJ(PIN)}$(2)	Total injected current (sum of all I/O and control pins)				± 25	
I_L	Input leakage current	$V_{SS} \le V_{IN} \le V_{DD}$			±1	µA
I_S	Static current consumption induced by each floating input pin(3)	Floating input mode		200		
R_{PU}	Weak pull-up equivalent resistor(4)	$V_{IN} = V_{SS}$	50	90	250	kΩ
C_{IO}	I/O pin capacitance			5		pF
$t_{f(IO)out}$	Output high to low level fall time(5)	C_L = 50pF Between 10% and 90%		25		ns
$t_{r(IO)out}$	Output low to high level rise time(5)			25		
$t_{w(IT)in}$	External interrupt pulse time(6)		1			t_{CPU}

1. Hysteresis voltage between Schmitt trigger switching levels. Based on characterization results, not tested.
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 $V_{IN}<V_{SS}$. Refer to *Section 12.2.2: Current characteristics on page 314* for more details.
3. 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 139*). 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.
4. The R_{PU} pull-up equivalent resistor is based on a resistive transistor (corresponding I_{PU} current characteristics described in *Figure 140*). This data is based on characterization results, tested in production at V_{DD} max.
5. Data based on characterization results, not tested in production.
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.

Figure 139. Two typical applications with unused I/O pin



1. 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 140. Typical I_{PU} vs. V_{DD} with $V_{IN} = V_{SS}$

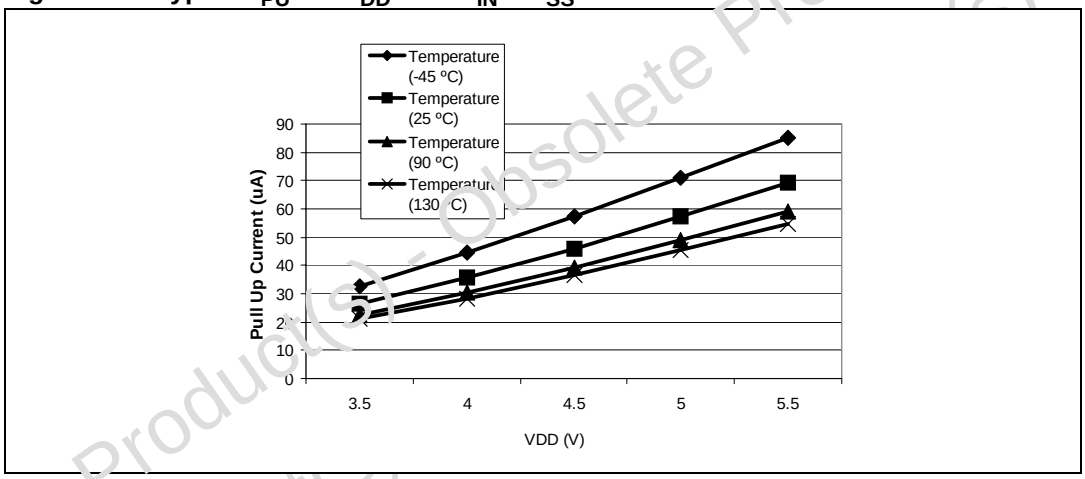
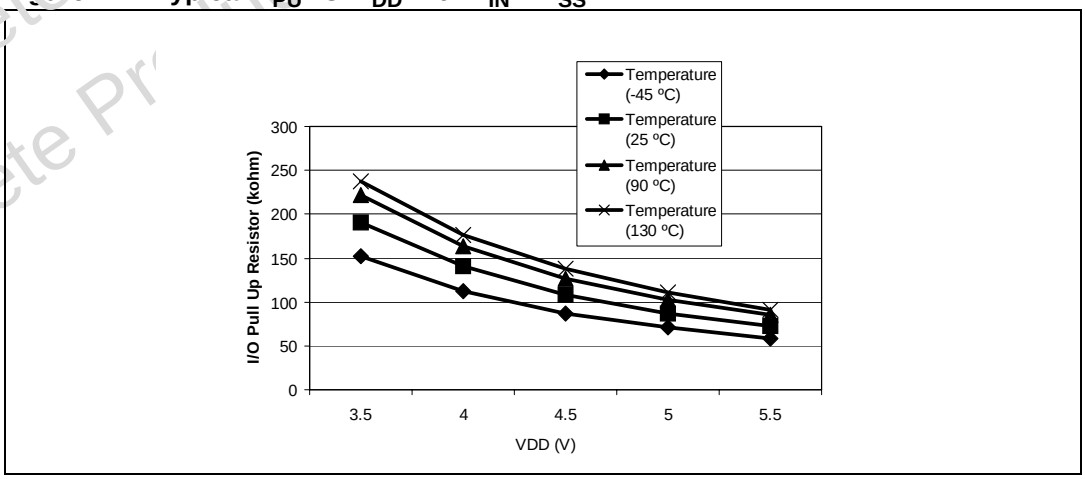


Figure 141. Typical R_{PU} vs. V_{DD} with $V_{IN} = V_{SS}$



12.8.2 Output driving current

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

Table 210. Output driving current

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 142*)	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 143*)		I_{IO} = +20mA, $T_A \le 85°C$ $T_A \ge 85°C$		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 144*)		I_{IO} = -5mA, $T_A \le 85°C$ $T_A \ge 85°C$	V_{DD}-1.4 V_{DD}-1.6		
			I_{IO} = -2mA	V_{DD}-0.7		

1. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section 12.2.2: Current characteristics on page 314* 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: Current characteristics on page 314* and the sum of I_{IO} (I/O ports and control pins) must not exceed I_{VDD}.

Figure 142. Typical V_{OL} at V_{DD} = 5V (standard)

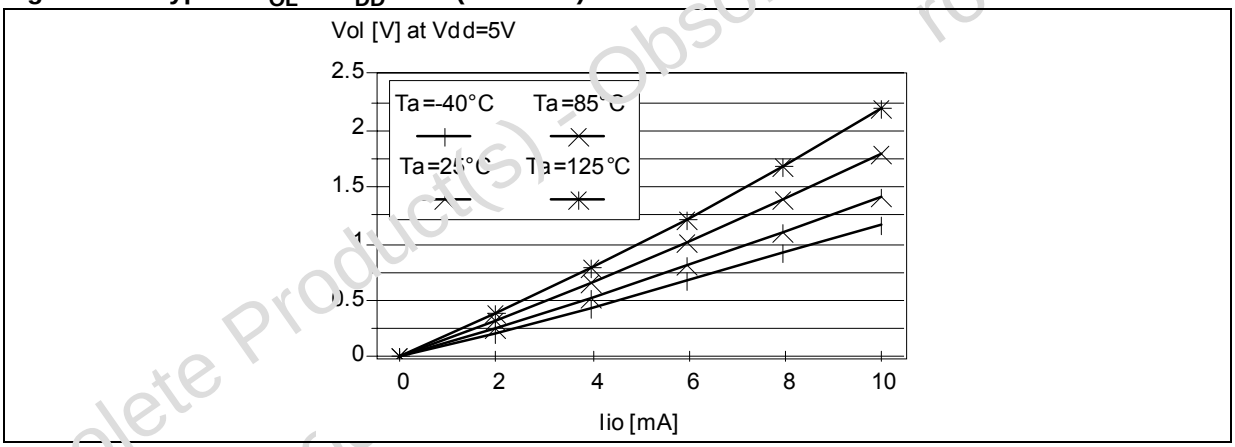


Figure 143. Typical V_{OL} at V_{DD} = 5V (high-sink)

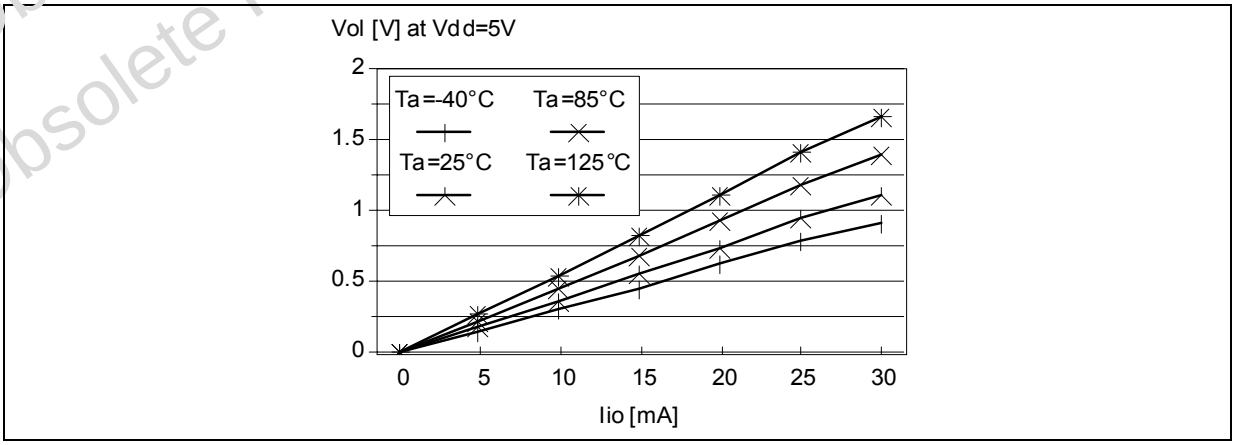
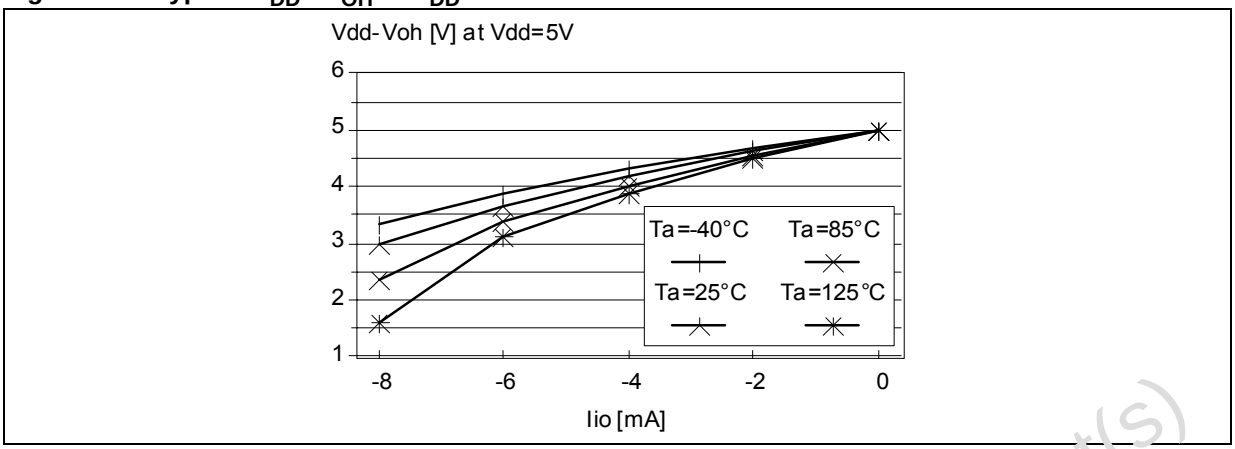


Figure 144. Typical V_{DD} - V_{OH} at V_{DD} = 5V



12.9 Control pin characteristics

12.9.1 Asynchronous $\overline{\text{RESET}}$ pin

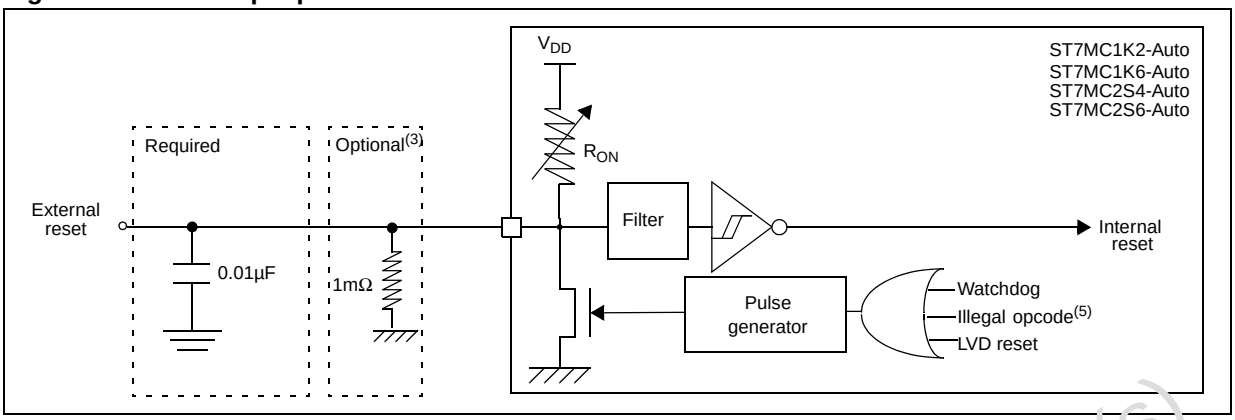
Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Table 211. Asynchronous $\overline{\text{RESET}}$ pin

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
V_{IL}	Input low level voltage(1)					0.3xV_{DD}	V
V_{IH}	Input high level voltage(1)			0.7xV_{DD}			
V_{hys}	Schmitt trigger voltage hysteresis(2)				1		
V_{OL}	Output low level voltage(3)	V_{DD} = 5V	I_{IO} = +5mA		0.5	1.2	
			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			
$t_{g(RSTL)in}$	Filtered glitch duration(5)				450		ns

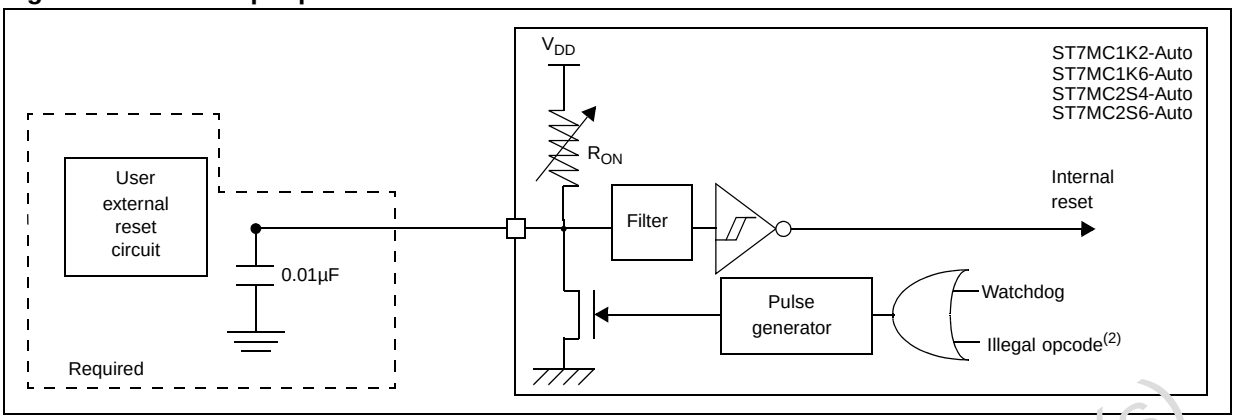
1. Data based on characterization results, not tested in production.
2. Hysteresis voltage between Schmitt trigger switching levels.
3. The I_{IO} current sunk must always respect the absolute maximum rating specified in *Section 12.2.2: Current characteristics on page 314* and the sum of I_{IO} (I/O 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.

Figure 145. $\overline{\text{RESET}}$ pin protection when LVD is enabled(1)(2)(4)



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: Asynchronous RESET pin on page 332*. Otherwise the reset is not 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: Current characteristics on page 314*.
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.
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 adds 5µA to the power consumption of the MCU).
4. Tips when using the LVD:
   A. Check that all recommendations related to ICCCLK and reset circuit have been applied (see notes above)
   B. 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.
   C. The capacitors connected on the $\overline{\text{RESET}}$ 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."
5. Please refer to *Section 11.2.2: Illegal opcode reset* for more details on illegal opcode reset conditions

Figure 146. $\overline{\text{RESET}}$ pin protection when LVD is disabled(1)



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: Asynchronous RESET pin on page 332*. Otherwise the reset is not 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: Current characteristics on page 314*.
2. Please refer to *Section 11.2.2: Illegal opcode reset on page 309* for more details on illegal opcode reset conditions

12.9.2 ICCSEL/V_{PP} pin

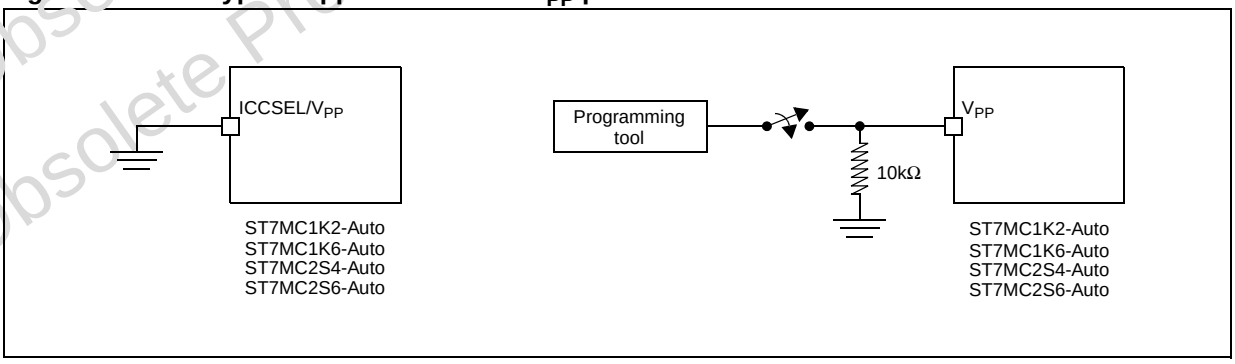
Subject to general operating conditions for V_{DD}, f_{CSC}, and T_A unless otherwise specified.

Table 212. ICCSEL/V_{PP} pin

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

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).

Figure 147. Two typical applications with V_{PP} pin(1)



1. When the ICC mode is not required by the application, the 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 *Section 9: I/O ports on page 78* 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

Table 213. 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} = 8 MHz	125			ns
f_{EXT}	ART external clock frequency		0		$f_{CPU}/2$	MHz
f_{PWM}	PWM repetition rate					
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

Table 214. 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} = 8 MHz	250			ns
f_{EXT}	Timer external clock frequency		0		$f_{CPU}/4$	MHz
f_{PWM}	PWM repetition rate					
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.

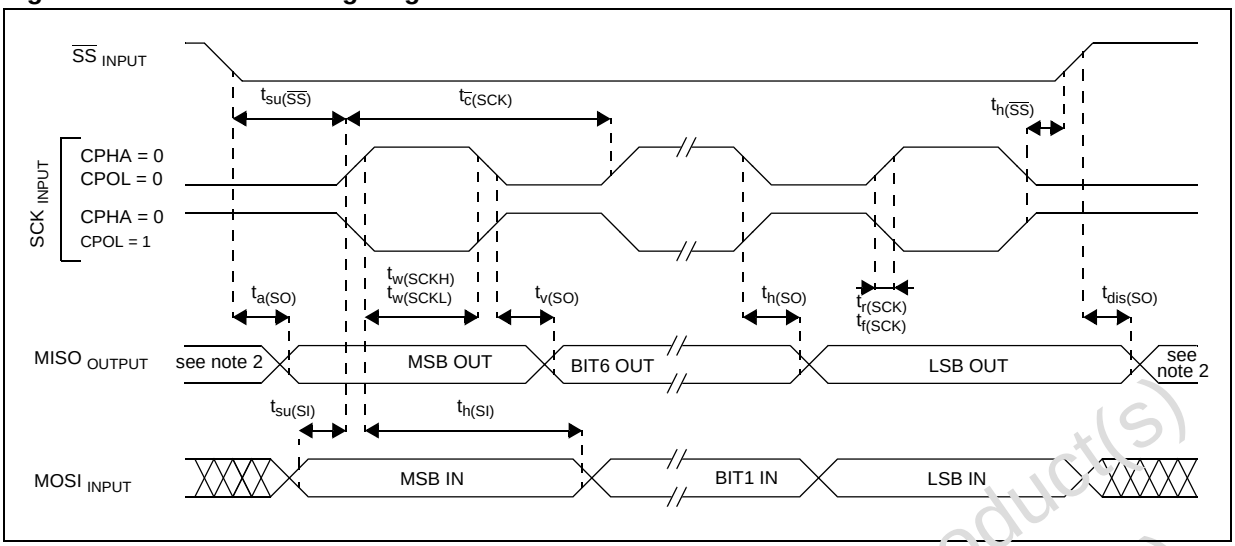
Refer to I/O port characteristics for more details on the input/output alternate function characteristics ($\overline{SS}$, SCK, MOSI, MISO).

Table 215. SPI characteristics

Symbol	Parameter	Conditions	Min	Max	Unit
f_{SCK} $1/t_{c(SCK)}$	SPI clock frequency	Master f_{CPU} = 8 MHz	$f_{CPU}/128$ 0.0625	$f_{CPU}/4$ 2	MHz
		Slave f_{CPU} = 8 MHz	0	$f_{CPU}/2$ 4	
$t_{r(SCK)}$ $t_{f(SCK)}$	SPI clock rise and fall time		see *Table 2 on page 23* for I/O port pin description		
$t_{su(\overline{SS})}$ (1)	$\overline{SS}$ setup time(2)	Slave	$(4 \times t_{CPU}) + 50$		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		

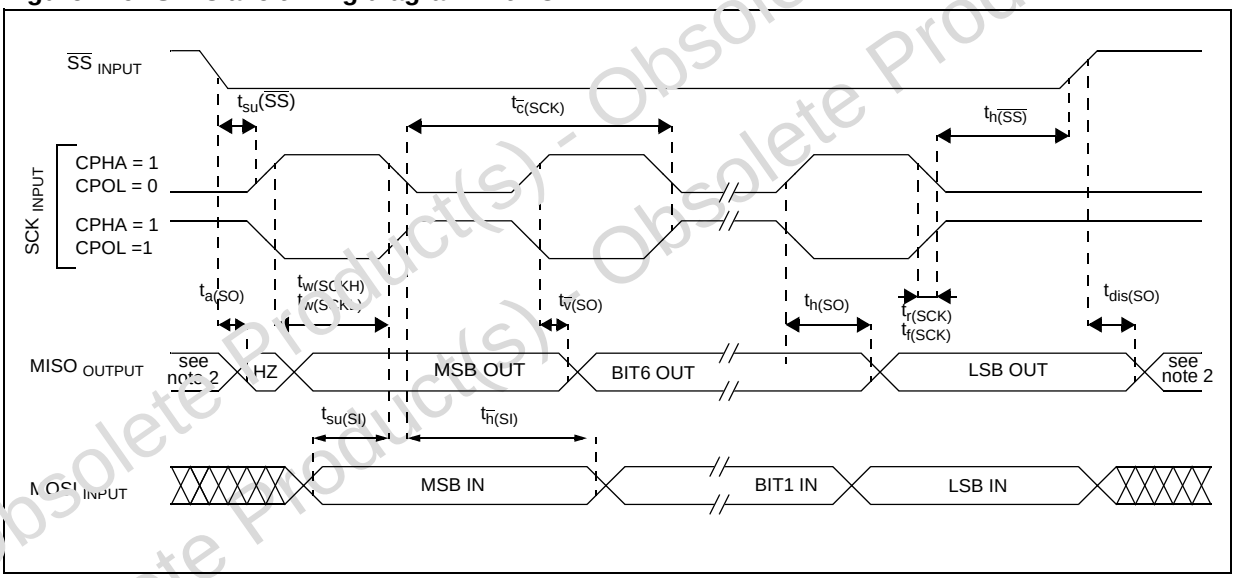
1. Data based on design simulation and/or characterization results, not tested in production.
2. Depends on f_{CPU}. For example, if f_{CPU} = 8 MHz, then $T_{CPU} = 1/f_{CPU}$ = 125ns and $t_{su(\overline{SS})}$ = 550ns.

Figure 148. SPI slave timing diagram with CPHA = 0(1)



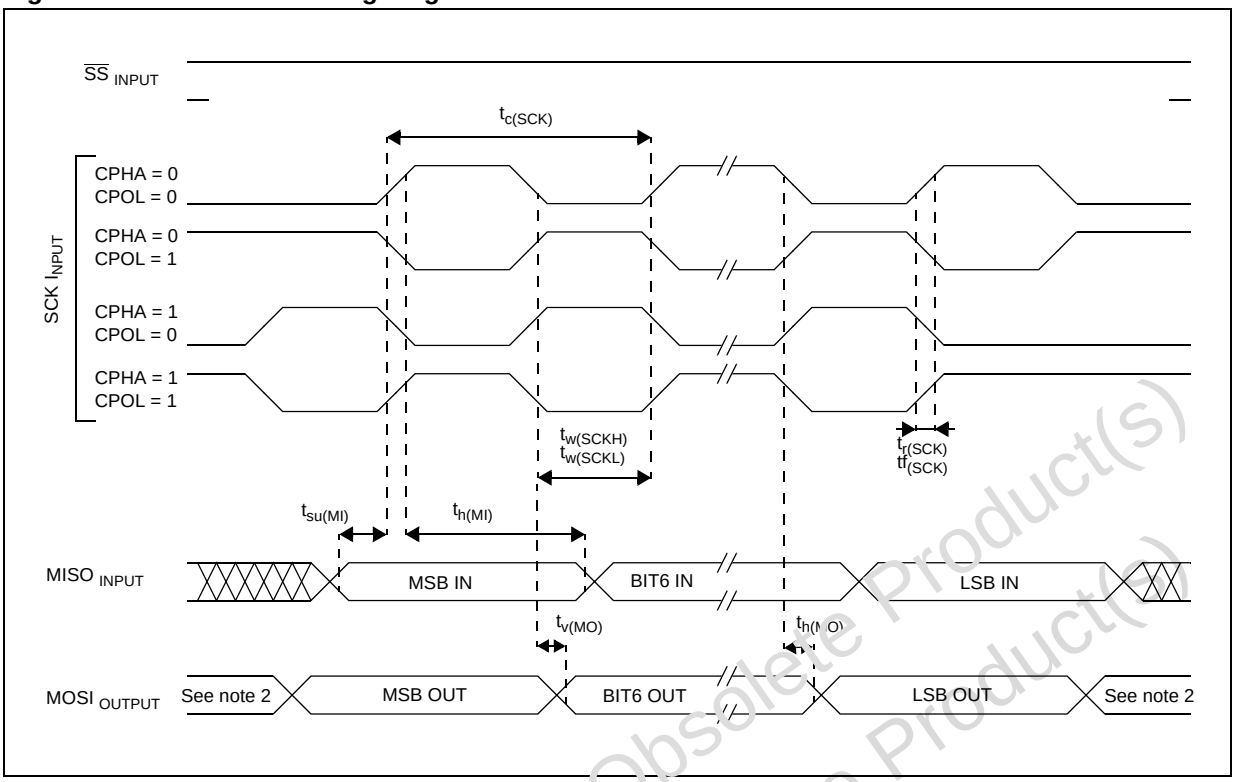
1. Measurement points are done at CMOS levels: $0.3xV_{DD}$ and $0.7xV_{DD}$.
2. When no communication is on-going the data output line of the SPI (MOSI in master mode, MISO in slave mode) has its alternate function capability released. In this case, the pin status depends on the I/O port configuration.

Figure 149. SPI slave timing diagram with CPHA = 1(1)



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 on the I/O port configuration.

Figure 150. SPI Master timing diagram(1)



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

Table 216. 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	%

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

12.12.2 Input stage (comparator + sampling)

Table 217. 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, that is, CKE = 1 or DAC = 1 (with f_{PERIPH} = 4 MHz)	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 151*)			
		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 151*)			
		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 152*)			
		Time needed to generate D event (MTIM captured in MDREG) as soon as the comparator toggles	$1/f_{SCF} + 3/f_{mtc}$ (see *Figure 152*)			
		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 153*)			
		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 153*)			
		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 153*)			

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, that is, there is a variation of ($1/f_{mtc}$) or ($1/f_{SCF}$) depending on the case.

Figure 151. Example 1: waveforms for zero-crossing detection with sampling at the end of PWM off-time

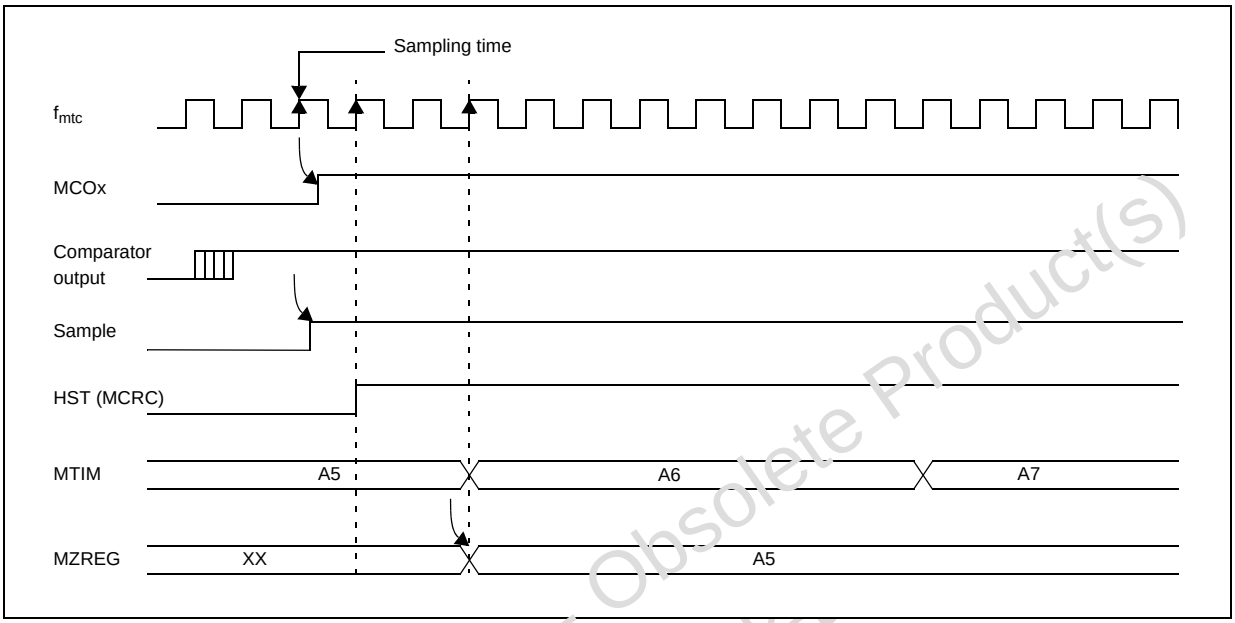


Figure 152. Example 2: waveforms for zero-crossing detection with sampling at f_{SCF}

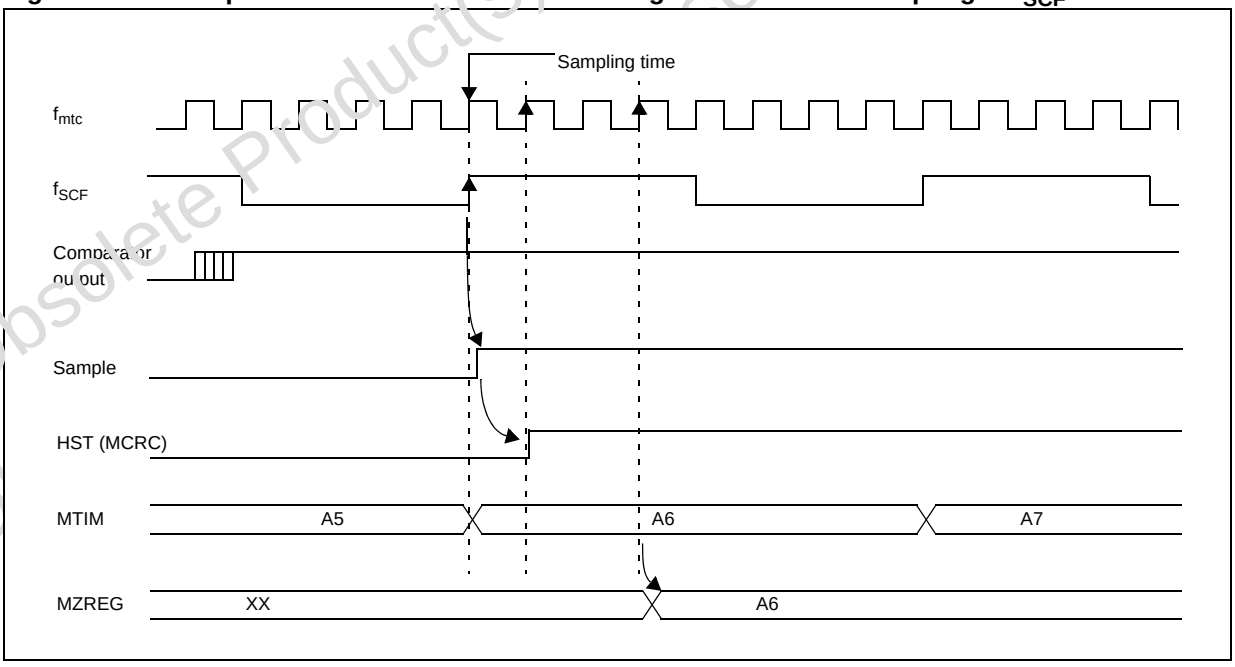


Figure 153. Example 3: Waveforms for zero-crossing detection with sampling after a delay during PWM on-time

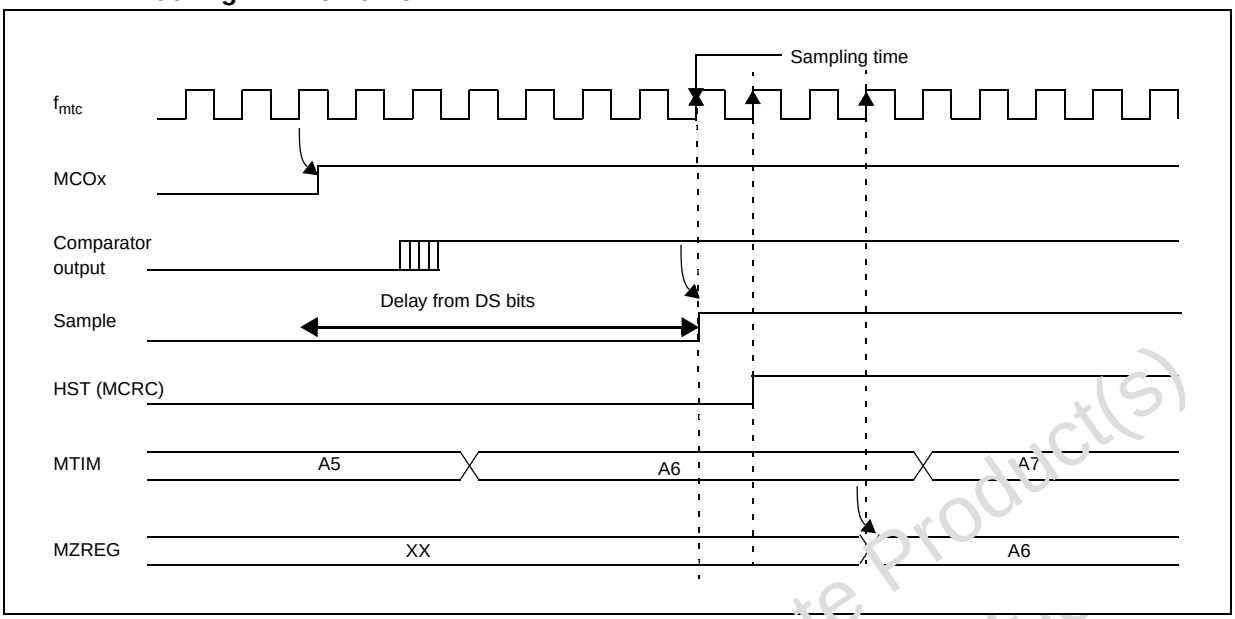


Figure 154. Example 4: Waveforms for zero-crossing detection with sampling after a delay at f_{SCF}

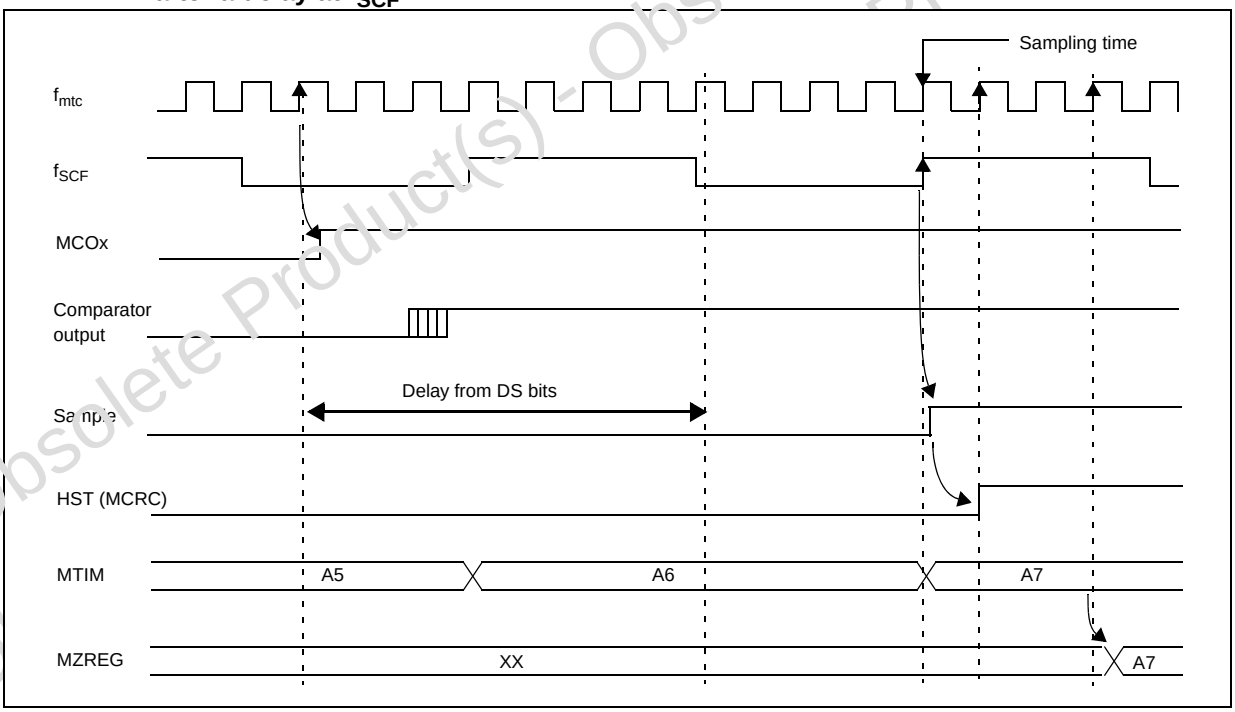
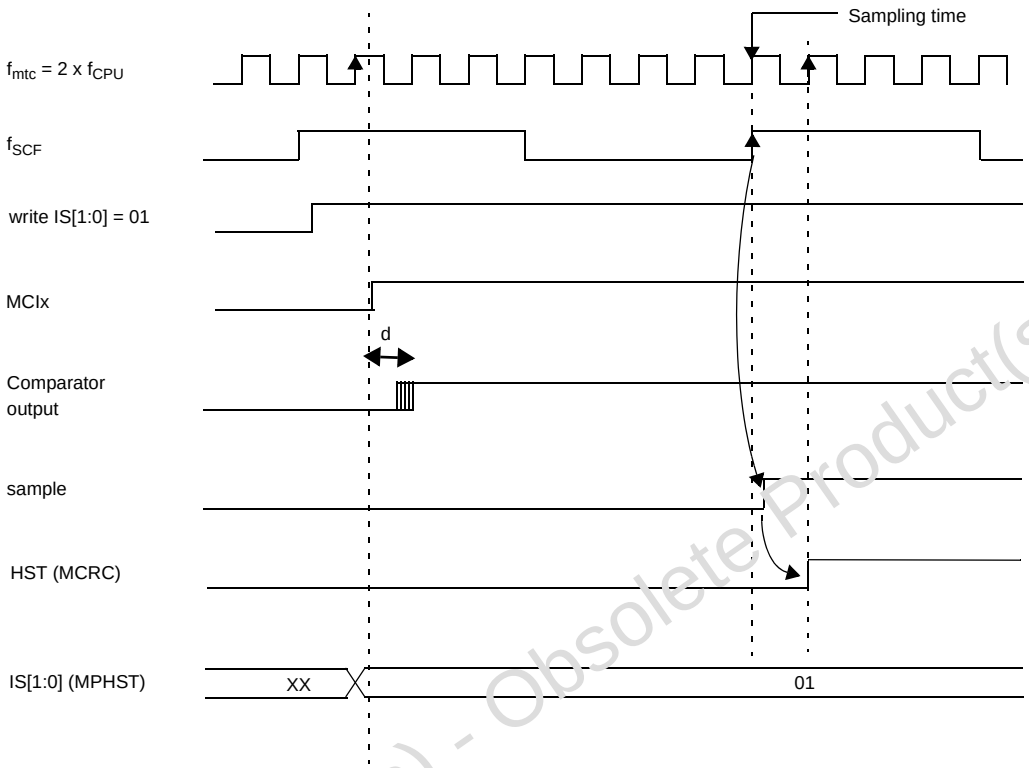


Figure 155. Example 5: Waveforms for sensor HST update timing diagram for a newly selected phase input



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 and multiplexer propagation delay (d)
3) $1/f_{SCF}$ to sample the comparator output
4) $1/f_{mtc}$ to resynchronize the data and set the HST bit

Example with $f_{PERIPH} = f_{mtc}$ = 4 MHz:

SCF[1:0]	00	01	10	11
f_{SCF}	1 MHz	500 kHz	250 kHz	125 kHz
Secure waiting time to read HST	2µs	3µs	5µs	9µs

12.12.3 Input stage (current feedback comparator and sampling)

Table 218. Input stage (current feedback comparator and 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, that is, CKE = 1 or DAC = 1 (with f_{PERIPH} = 4 MHz)	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 156*)			
		Time between a comparator toggle (current loop event) and bit CL becoming set (CFF = 0)	$2/f_{MTC}$ (see *Figure 156*)			
		Time needed to turn OFF the MCOs when comparator output rises (CFF = x)	$(1 + x) * (4/f_{PERIPH}) + (3/f_{mtc})$ (see *Figure 157*)			
		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 157*)			

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 output 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, that is, there is a variation of ($1 / f_{mtc}$) or ($4/f_{PERIPH}$) depending on the case.

Figure 156. Example 1: Waveforms for overcurrent detection with current feedback filter OFF

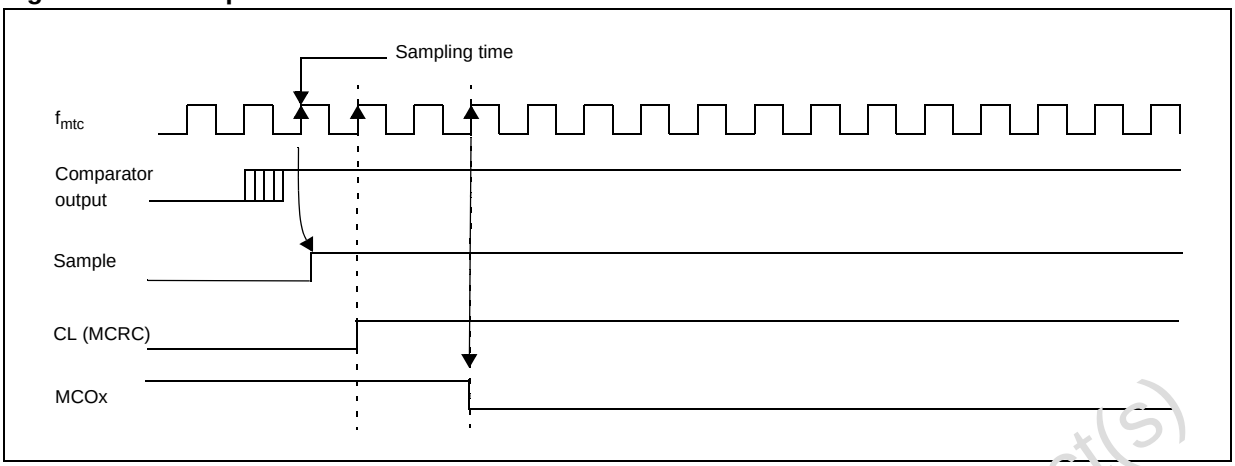
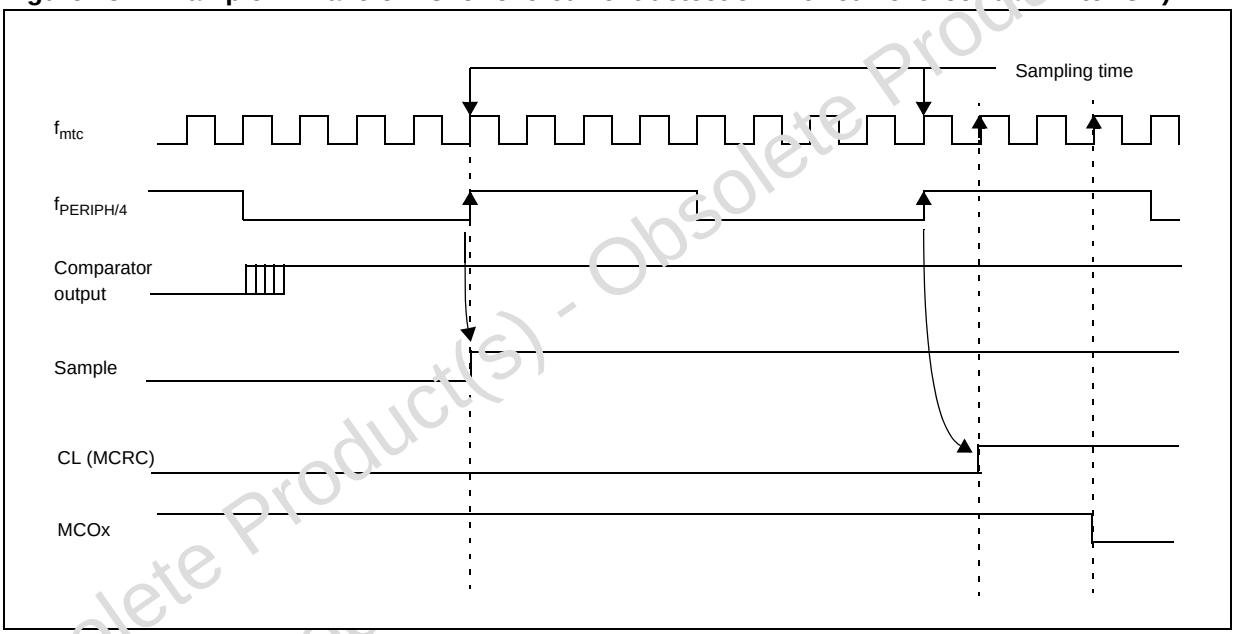


Figure 157. Example 2: Waveforms for overcurrent detection with current feedback filter ON)



1. 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 to +125°C, V_{DD} - V_{SSA} = 4.5 to 5.5V unless otherwise specified)

Table 219. Operational amplifier characteristics

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 -)(1)	After calibration, V_{IC} =1V		2.5	10(2)	mV
ΔV_{io}	Input offset voltage drift from the calibrated voltage, temperature conditions	With respect to temperature			8.5(3)	µV/°C
		With respect to common mode input			1(3)	mV/V
		With respect to supply			3.1(3)	
CMR	Common mode rejection ratio	HIGHGAIN = 0 @ 100 kHz		74		dB
SVR	Supply voltage rejection ratio	@ 100 kHz	50(4)	65		
A_{vd}	Voltage gain	R_L = 10kΩ	(1.5)(4)	12		V/mV
V_{SAT_OH}	High level output saturation voltage (V_{DD} - V_{OUT})	R_L = 10kΩ		60	90(4)	mV
V_{SAT_OL}	Low level output saturation voltage			30		
GBP	Gain bandwidth product	HIGHGAIN = 0	2(4)	4	6(4)	MHz
		HIGHGAIN = 1	7(4)	11	15(4)	
SR^+	Slew rate while rising	HIGHGAIN = 0 (A_{VCL} = 1, R_L = 10kΩ, C_L = 150pF, V_i = 1.75V to 2.75V)(5)	1(4)	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)(5)	2.5(4)	7.5		
Φm	Phase margin	HIGHGAIN = 0		73		degrees
		HIGHGAIN = 1		75		
T_{wakeup}	Wake-up time for the op-amp from off state		0.8(6)		1.6(6)	µs

1. After offset compensation has been performed.
2. 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 op-amp is in use. This phenomenon is even more critical when a big external serial resistor is added on the inputs.
3. The data are provided from simulations (not tested in production) to guide the user when re-calibration is needed.
4. Data based on characterization results, not tested in production.
5. A_{VCL} = closed loop gain.
6. The data are provided from simulations (not tested in production).

12.14 10-bit ADC characteristics

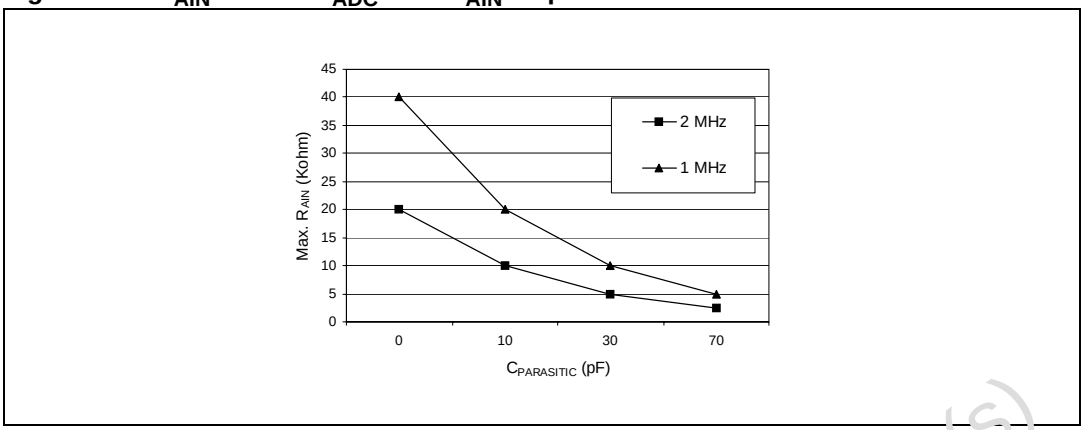
Subject to general operating conditions for V_{DD}, f_{OSC}, and T_A unless otherwise specified.

Table 220. 10-bit ADC characteristics

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}	Input leakage current for analog input				±1	µA
	Negative input leakage current on analog pins	$V_{IN} < V_{SS}$, $\lvert I_{IN} \rvert < 400\mu A$ on adjacent analog pin		5	0	
R_{AIN}	External input impedance				see *Figure 158* and *Figure 159* (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} = 8 MHz, f_{ADC} = 4 MHz, 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} = 8 MHz, f_{ADC} = 4 MHz, 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

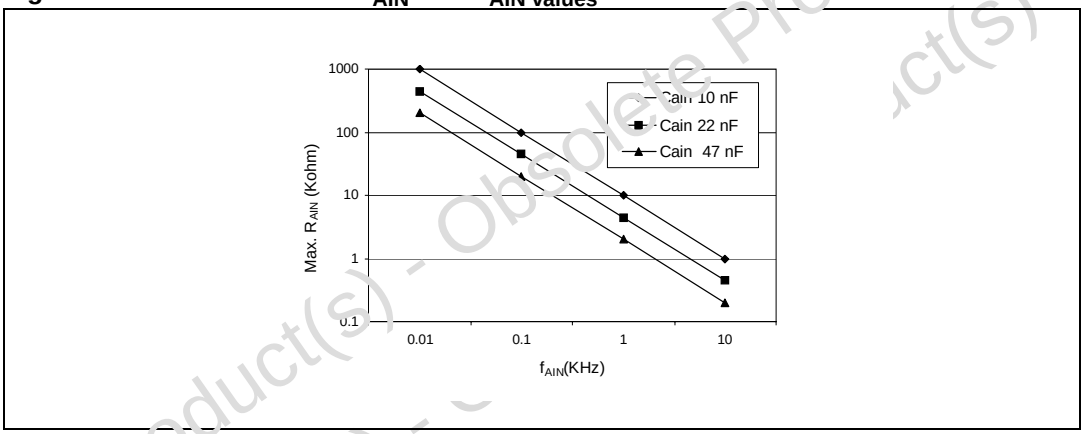
1. When V_{SSA} pins are not available on the pinout, the ADC refer to V_{SS}.
2. Any added external serial resistor downgrades the ADC accuracy (especially for resistance greater than 10kΩ). Data based on characterization results, not tested in production.
3. $C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high $C_{PARASITIC}$ value downgrades 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}).

Figure 158. R_{AIN} max. vs f_{ADC} with C_{AIN} = 0pF



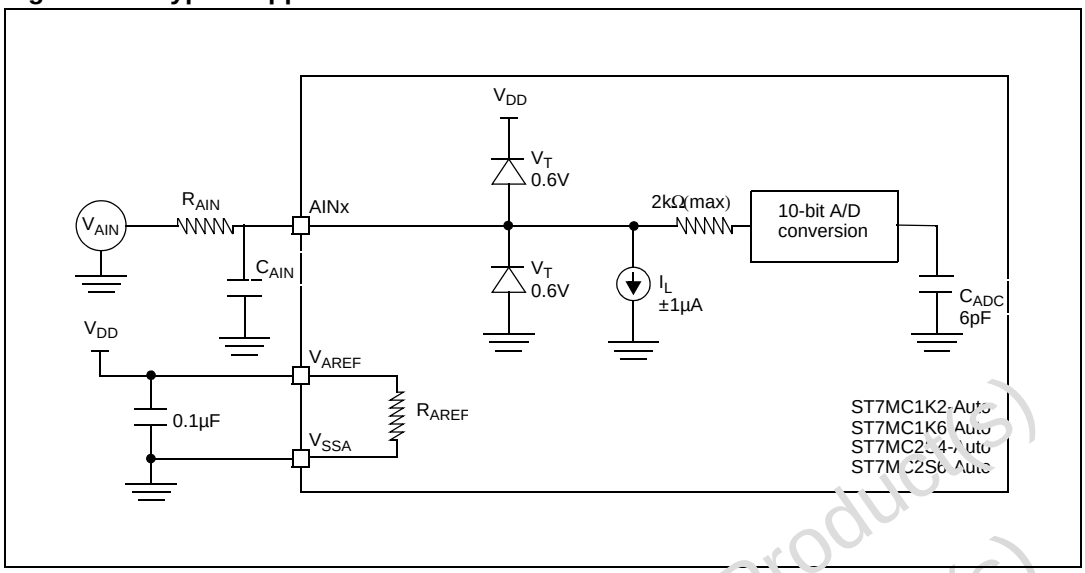
1. $C_{PARASITIC}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (3pF). A high $C_{PARASITIC}$ value downgrades conversion accuracy. To remedy this, f_{ADC} should be reduced.

Figure 159. Recommended C_{AIN} and R_{AIN} values



1. 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}).

Figure 160. Typical application with ADC



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: Pin description on page 21*). In this case the analog supply and reference pads are internally bonded to the V_{DD} and V_{SS} pins.

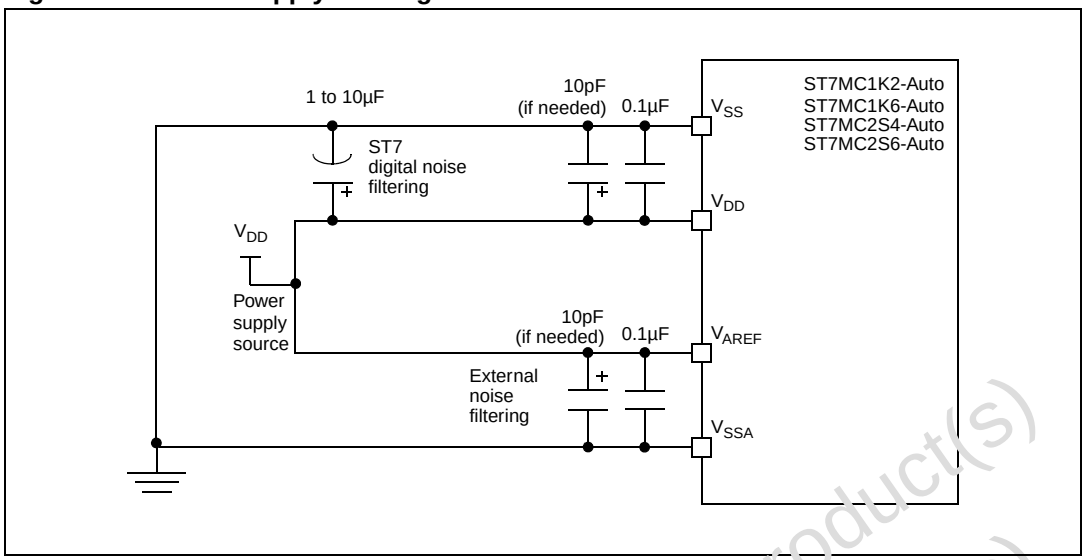
Separation of the digital and analog power pins allow board designers to improve A/D performance. Conversion accuracy can be impacted by voltage drops and noise in the event of heavily loaded or badly decoupled power supply lines (see *Section 12.14.2: General PCB design guidelines on page 349*).

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 161*).
- 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 161. Power supply filtering



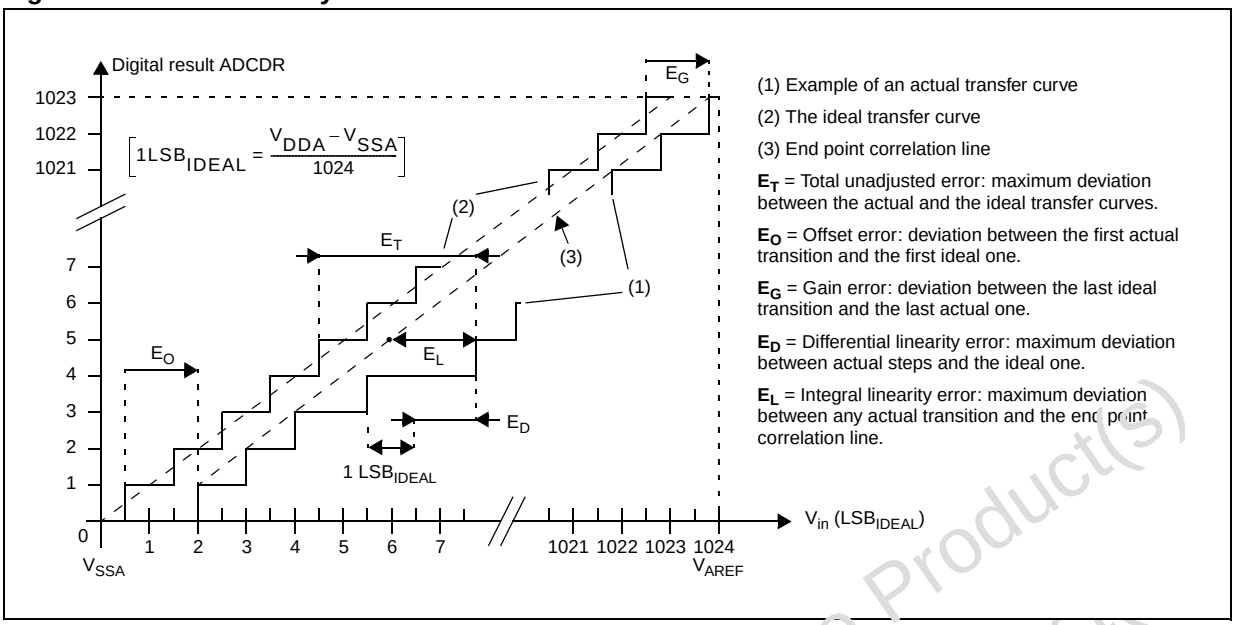
ADC accuracy with V_{DD} = 5.0V

Table 221. ADC accuracy with V_{DD} = 5.0V

Symbol	Parameter	Conditions	Typ	Max(1)	Unit
$\|E_T\|$	Total unadjusted error(2)	V_{AREF} = 3.0V to 5.0V, f_{CPU} = 8 MHz, f_{ADC} = 4 MHz, R_{AIN} < 10kΩ	4		LSB
$\|E_O\|$	Offset error(2)		2.5	4	
$\|E_G\|$	Gain error(2)		2		
$\|E_D\|$	Differential linearity error(2)		2	4.5	
$\|E_L\|$	Integral linearity error(2)				

1. Data based on characterization results, monitored in production to guarantee 99.73% within ± max value from -40°C to 125°C (± 3σ distribution limits).
2. 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: 10-bit ADC characteristics on page 347*. Any positive injection current within the limits specified for $I_{INJ(PIN)}$ and $\Sigma I_{INJ(PIN)}$ in *Section 12.8: I/O port pin characteristics on page 329* does not affect the ADC accuracy.

Figure 162. ADC accuracy characteristics



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

13.1 ECOPACK®

In order to meet environmental requirements, ST offers these devices in ECOPACK® packages. These packages have a Lead-free second level interconnect. The category of second Level Interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label.

ECOPACK is an ST trademark. ECOPACK specifications are available at: www.st.com.

13.2 LQFP packages

The following pages contain the package drawings and mechanical data as well as the thermal characteristics and soldering information for the 44- and 32-pin LQFP packages.

13.2.1 LQFP44 package

Figure 163. 44-pin low profile quad flat package outline

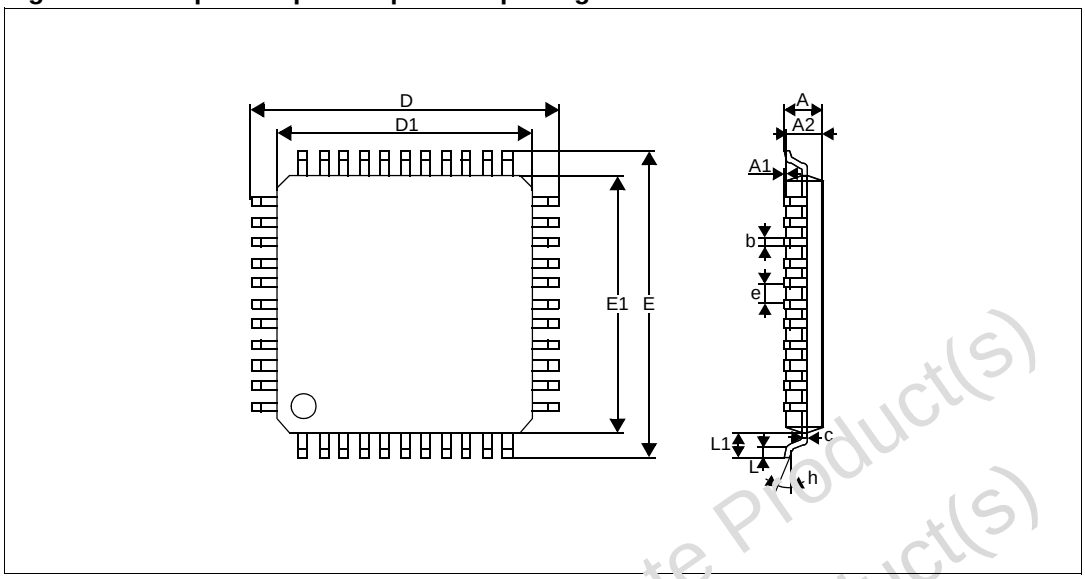


Table 222. 44-pin low profile quad flat package mechanical data

Dimension	mm			inches[1]		
	Minimum	Typical	Maximum	Minimum	Typical	Maximum
A			1.60			0.063
A1	0.05		0.15	0.002		0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.30	0.37	0.45	0.012	0.015	0.018
c	0.09		0.20	0.004	0.000	0.008
D		12.00			0.472	
D1		10.00			0.394	
E		12.00			0.472	
E1		10.00			0.394	
e		0.80			0.031	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
	Number of pins					
N	44					

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

13.2.2 LQFP32 package

Figure 164. 32-pin low profile quad flat packag outline

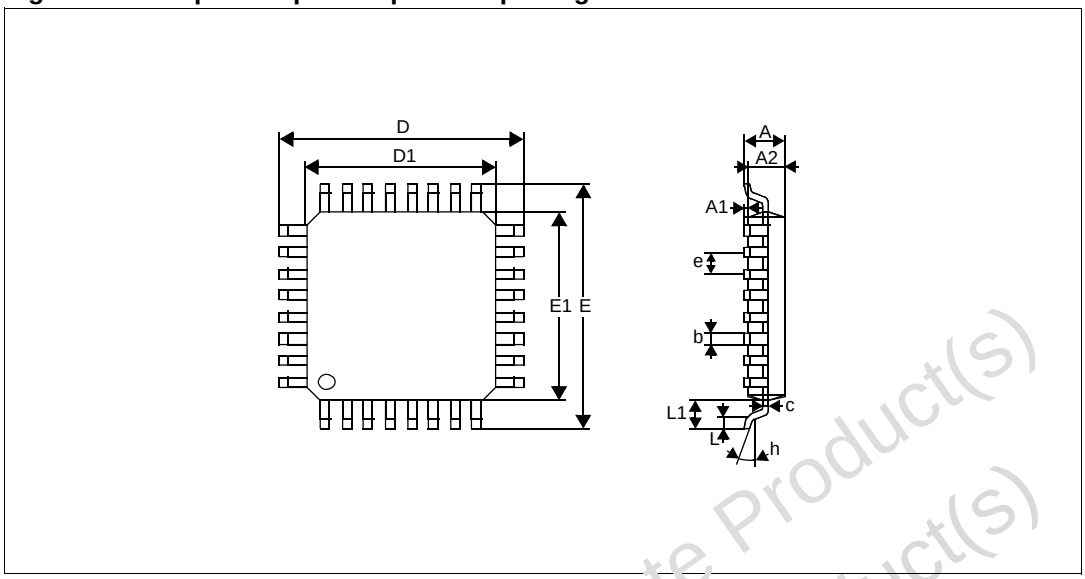


Table 223. 32-pin low profile quad flat package mechanical data

Dimension	mm			inches[1]		
	Minimum	Typical	Maximum	Minimum	Typical	Maximum
A			1.60			0.063
A1	0.05		0.15	0.002		0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
b	0.30	0.37	0.45	0.012	0.015	0.018
c	0.09		0.20	0.004		0.008
D		9.00			0.354	
D1		7.00			0.276	
E		9.00			0.354	
E1		7.00			0.276	
e		0.80			0.031	
θ	0°	3.5°	7°	0°	3.5°	7°
L	0.45	0.60	0.75	0.018	0.024	0.030
L1		1.00			0.039	
	Number of pins					
N	32					

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

13.2.3 Thermal characteristics

Table 224. Thermal characteristics

Symbol	Ratings	Value	Unit
R_{thJA}	Package thermal resistance (junction to ambient): – LQFP44 10x10 – LQFP32 7x7	 68 80	°C/W
T_{Jmax}	Maximum junction temperature(1)	150	°C
P_{Dmax}	Power dissipation(2)	500	mW

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.

13.2.4 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).

Forward compatibility

ECOPACK® LQFP packages are fully compatible with Lead (Pb) containing soldering process (see application note AN2034)

Table 225. Soldering compatibility (wave and reflow soldering process)

Package	Plating material devices	Pb solder paste	Pb-free solder paste
LQFP 32	NiPdAu (Nickel-Palladium-Gold)	Yes	Yes(1)
LQFP 44	Sn (pure Tin)		

1. Assemblers must verify that the Pb-package maximum temperature (mentioned on the Inner box label) is compatible with their Lead-free soldering process.

14 ST7MCxxx-Auto 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). ST7MC1K2-Auto and ST7MC2S4-Auto are ROM devices. ST7PMC1K2-Auto, ST7PMC2S4-Auto, ST7PMC1K6-Auto, and ST7PMC2S6-Auto devices are factory advanced service technique ROM (FASTROM) versions: They are programmed Flash devices.

ST7FMC1K2-Auto, ST7FMC1K6-Auto, ST7FMC2S4-Auto, and ST7FMC2S6-Auto 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

Table 226. Flash option bytes

	Static option byte 0								Static option byte 1							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	WDG		CKSEL	VD		RSTC	DIV2	FMP_R	PKG			Reserved			MCO	
	Halt	SW		1	0				2	1	0				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.

Table 227. Option byte 0

Bit	Name	Function
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

Table 227. Option byte 0 (continued)

Bit	Name	Function
OPT4:3	VD[1:0]	Voltage detection These option bits enable the voltage detection block (LVD, and AVD): 00: Selected low voltage detector = LVD and AVD on 01: Selected low voltage detector = LVD on and AVD off 10: Selected low voltage detector = LVD and AVD off 11: Selected low voltage detector = LVD and AVD off
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 OSC1 (or OSCIN) = 8 MHz 0: DIV2 divider enabled (in order to have 8 MHz required for the PLL with OSC1 (or 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: Read-out protection on page 34* for more details. 0: Read-out protection enabled 1: Read-out protection disabled

1. Even if PLL clock is selected, a clock signal must always be present (refer to *Figure 9: Clock, reset and supply block diagram on page 43*).

Table 228. Option byte 1

Bit	Name	Function
OPT7:5	PKG[2:0]	Package selection These option bits are used to select the device package: 000: Selected package = LQFP32 001: Selected package = LQFP44 011: Reserved 1xx: Reserved
OPT4:2	-	Reserved
OPT1:0	MCO	Motor control output options MCO port under reset: 00: Motor control output = HiZ 01: Motor control output = Low 10: Motor control output = High 11: Motor control output = HiZ

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 provides detailed information on contractual points.

Table 229. Supported part numbers

Part number	Program memory (bytes)	RAM (bytes)	Temp. range	Package
ST7FMC1K2TC	8K Flash	384	-40°C +125°C	LQFP32
ST7FMC1K6TC	32K Flash	1024		
ST7FMC2S4TC	16K Flash	768		LQFP44
ST7FMC2S6TC	32K Flash	1024		
ST7MC1K2TC/xxx(1)	8K ROM	384		LQFP32
ST7MC2S4TC/xxx(1)	16K ROM	768		LQFP44
ST7PMC1K2TC/xxx(1)	8K FASTROM	384		LQFP32
ST7PMC1K6TC/xxx(1)	32K FASTROM	1024		
ST7PMC2S4TC/xxx(1)	16K FASTROM	768		LQFP44
ST7PMC2S6TC/xxx(1)	32K FASTROM	1024		

1. /xxx stands for the ROM or FASTROM code assigned by STMicrolectronics.

ST7MC1K2-Auto, ST7MC1K6-Auto, ST7MC2S4-Auto, and ST7MC2S6-Auto microcontroller option list

(Last update: June 2006)

Customer: .....
Address: .....
.....
Contact: .....
Phone No: .....
Reference/ROM or FASTROM code: .....

The ROM or FASTROM/ROM code name is assigned by STMicroelectronics.
ROM or FASTROM/ROM 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
LQFP32:	[ ] ST7MC1K2		
LQFP44:		[ ] ST7MC2S4	

FASTROM	8K	16K	32K
LQFP32:	[ ] ST7PMC1K2		[ ] ST7PMC1K6
LQFP44:		[ ] ST7PMC2S4	[ ] ST7PMC2S6

Conditioning for LQFP package (check only one option):

[ ] Tape and Reel [ ] Tray

Temperature range: [ ] A (-40°C to +85°C)
[ ] C (-40°C to +125°C)

Special marking: [ ] No [ ] Yes "....." (10 char. max)

Authorized characters are letters, digits, '.', '-', '/' and spaces only.

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 .....

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 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 Development tool order codes for the ST7MCxxx-Auto family

Table 230. Development tool order codes for the ST7MCxxx-Auto family

MCU	Starter kit	Emulator	Programming tool
ST7MC1K2-Auto ST7MC1K6-Auto	ST7MC1K2-Auto, ST7MC1K6-Auto, ST7MC2S4-Auto, ST7MC2S6-Auto, -KIT/BLDC	ST7MDT50-EMU3	ST7-STICK(1)(2) STX-RLINK(3)
ST7MC2S4-Auto ST7MC2S6-Auto			

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

14.3.5 Package/socket footprint proposal

Table 231. Suggested list of socket types

Package/probe	Socket Reference		Emulator Adapter	
LQFP32 7x7	IRONWOOD	SF-QFE32SA-L-01	IRONWOOD	SK-UGA06/32A-01
LQFP44 10x10	YAMAICHI	IC149-044-*52-*5	YAMAICHI	ICP-044-5

15 Known limitations

15.1 Flash/FASTROM devices only

Two temperature versions are available with different limitations (see *Table 232*).

Table 232. Temperature version limitaions for Flash and FASTROM devices

Part number	Limitation
ST7FMC1K6TCE	Limitation corresponding to temperature version C
ST7FMC2S6TCE	
ST7FMC1K2TCE	Limitation corresponding to temperature version A
ST7FMC2S4TCE	

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, that is, 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

Example:

```
SIM
Reset flag or interrupt mask
RIM
```

Nested interrupt context

The symptom does not occur when the interrupts are handled normally, that is, 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 = 0x00h; // Disable the compare interrupt.
- TACSR | = 0x40; // Disable the timer.

Perform the following to enable the timer again

- TACSR & = ~0x40; // Enable the timer.
- TACR1 = 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 (f_{CPU} = 8 MHz and SCIBRR = 0xC9), the wrong break duration occurrence is around 1%.

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 (f_{CPU} = 8 MHz 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) x 1.4 = 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, that is, 208µs at 19200 baud.

Workaround

The workaround is the same as for SCI mode but considering the low probability of occurrence (1%), it may be preferable to keep the break generation sequence as it is.

15.4.2 Header time-out does not prevent wake-up from mute mode

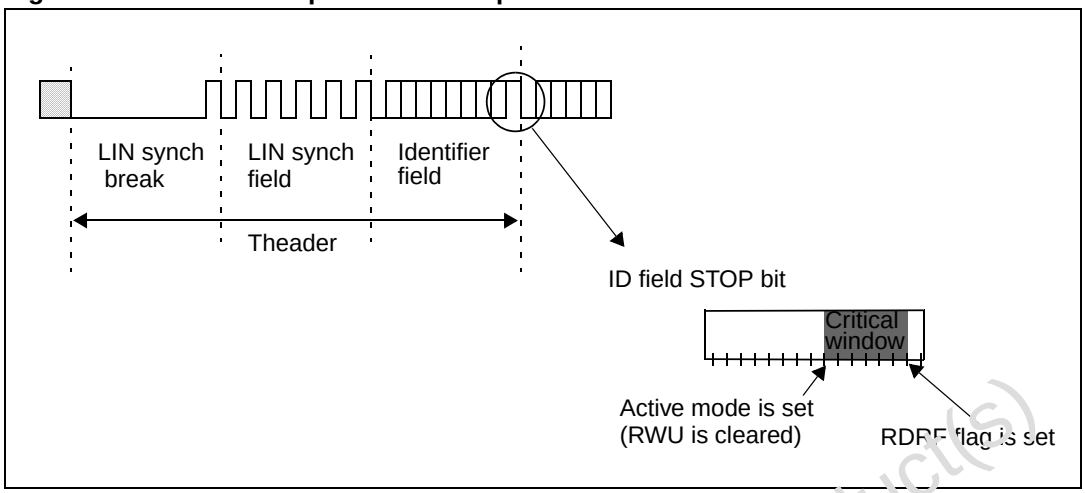
Normally, when LINSCI is configured in LIN slave mode, if a header time-out occurs during a LIN header reception (that is, header length > 57 bits), the LIN Header Error bit (LHE) is set, an interrupt occurs to inform the application but the LINSCI should stay in mute mode, waiting for the next header 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 165*), 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 generates another LINSCI interrupt (due to the RDRF flag setting).

Figure 165. Header reception event sequence



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 is 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 166*. Workaround is shown in bold characters.

Figure 166. 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 deadtime 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 is missing.

Workaround

The complementary PWM must be disabled by resetting the DTE bit in the MDTG register (see *Deadtime generator register (MDTG) on page 280*).

As the current in the motor is very low in this case, the MOSFET body diode can be used.

15.6 Reset value of unavailable pins

On rev. A silicon versions, some ports (ports A, C and E) have fewer than eight pins. The bits associated to the unavailable pins must always be kept at reset state.

15.7 Maximum values of AVD thresholds

On rev. A silicon versions, the maximum values of AVD thresholds are not tested in production.

15.8 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 is not detected and an interrupt is not generated.

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, that is, 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 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
```

16 Revision history

Table 233. Document revision history

Date	Revision	Changes
12-Jul-2007	Rev 1	Initial release Principal differences between initial release of ST7MCxxx-Auto datasheet and ST7MCx, revision 11 dated 8 December 2006: Changed document title on *page 1* Changed root part numbers to ST7MC1K2-Auto, ST7MC1K6-Auto and ST7MC2S4-Auto, ST7MC2S6-Auto throughout document Removed all references to the SDIP32, LQFP80, and LQFP64 packages thoughout document Updated all references to LQFP44 and LQFP32 packages Updated memory and RAM throughout document to be specific to the ST7MC1K2-Auto, ST7MC1K6-Auto and ST7MC2S4-Auto, ST7MC2S6-Auto devices. Updated temperatures ranges throughout document to include only versions A and C Removed all references relating to 'standard and industrial' from document *Features on page 1*: Changed number of I/0 ports from 60 to 34 and updated information on I/O ports and analog peripherals *Table 1: Device summary on page 19*: Updated Added footnote to *Table 2: Device pin description on page 23* 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. *Output compare on page 111*: Changed text of note 3 and removed compare register i latch signal from *Figure 49: Output compare timing diagram, $f_{TIMER} = f_{CPU}/4$ on page 114* *Section 13.2.4: Soldering information on page 355*: Replaced ECOPACK™ with ECOPACK® *Table 226 on page 356*: Modified option byte 2 *Table 229 on page 358*: Modified table for automotive versions only Added *Section 15.3: TIMD set simultaneously with OC interrupt on page 362* on the limitations of the 16-bit timer Minor content differences between initial release of ST7MCxxx-Auto datasheet and ST7MCx, revision 11 dated 8 December 2006: *Table 20: EICR register description on page 68*: Updated ports to include only those found on the LQFP32 and LQFP44 packages *SCI control register 1 (SCICR1) on page 171*: Changed description of bit 1 to reserved *Examples of LDIV coding on page 177*: Modified example 3 *Control register B (MCRB) on page 268* and *Parity register (MPAR) on page 285*: Added footnote pertaining to pre-load bits *Table 209: General characteristics on page 329*: Removed reference to PD7 *Table 220: 10-bit ADC characteristics on page 347*: Amended explanation of the parameter I_{lkg}

Table 233. Document revision history

Date	Revision	Changes
12-Jul-2007	Rev 1	Minor content differences between initial release of ST7MCxxx-Auto datasheet and ST7MCx, revision 11 dated 8 December 2006 (cont'd): Added footnote to *Table 222* and *Table 223 on page 354* *Section 13.2.4: Soldering information on page 355*: Updated for Lead-free soldering technology *Table 225: Soldering compatibility (wave and reflow soldering process) on page 355*: Added pure Tin for LQFP44 package Updated *ST7MC1K2-Auto, ST7MC1K6-Auto, ST7MC2S4-Auto, and ST7MC2S6-Auto microcontroller option list on page 359* Updated *Section 15.1: Flash/FASTROM devices only on page 361* Removed section '*Injected current on PD7 on page 371*' Removed figure '*Revision marking on box label and device marking on page 374*' Editing and formatting differences between initial release of ST7MCxxx-Auto datasheet and ST7MCx, revision 11 dated 8 December 2006: Small text changes throughout document Removed several tables that related to bit functioning and added information to the following register tables: *Table 5*, *Table 9*, *Table 13*, *Table 14*, *Table 17*, *Table 20*, *Table 38*, *Table 40*, *Table 51*, *Table 56*, *Table 65*, *Table 71*, *Table 126*, *Table 136*, *Table 137*, *Table 138*, *Table 142*, *Table 144*, *Table 160*, *Table 166*, *Table 173*, *Table 227*, *Table 228*. *Section 14.1: Flash option bytes on page 356*: Converted description of option bytes 1 and 2 into tables *Section 15: Known limitations on page 361*: Changed title