|  |
| --- |
| **Page 1** |

User Manual

**User Manual**

**MM32F003**

**32 ARM Cortex M0**

**1.19\_q**

Reserves the right to change relevant information without notice

|  |
| --- |
| **Page 2** |

table of Contents

[**1** memory and bus architecture](https://translate.googleusercontent.com/translate_f#23)………………………..**1**

[1.1 System Architecture](https://translate.googleusercontent.com/translate_f#23) .................... ...1

[1.2 Memory Organization](https://translate.googleusercontent.com/translate_f#24) ................... ..2

[1.2.1 Introduction](https://translate.googleusercontent.com/translate_f#24) ................... ..2

[1.2.2 Memory Map and Register Addressing](https://translate.googleusercontent.com/translate_f#24) ..................2

[1.3 Built-in SRAM](https://translate.googleusercontent.com/translate_f#26) ......................4

[1.4 Overview of Flash Memory](https://translate.googleusercontent.com/translate_f#26) ....................4

[1.5 Boot configuration](https://translate.googleusercontent.com/translate_f#26) ....................4

[**2** Embedded Flash **(FLASH)**](https://translate.googleusercontent.com/translate_f#28)**……………..6**

[2.1 Main Features of Flash Memory](https://translate.googleusercontent.com/translate_f#28) ........................6

[2.2 Flash memory function description](https://translate.googleusercontent.com/translate_f#28) .................... ..6

[2.2.1 Flash memory structure](https://translate.googleusercontent.com/translate_f#28) ..................6

[2.2.2 FLASH read operation](https://translate.googleusercontent.com/translate_f#29) ..................7

[2.2.3 Flash write and erase operations](https://translate.googleusercontent.com/translate_f#29) .................7

[2.3 Storage Protection](https://translate.googleusercontent.com/translate_f#35) ................... 14

[2.3.1 Main space write protection](https://translate.googleusercontent.com/translate_f#36) ................... 14

[2.3.2 Option byte write protection](https://translate.googleusercontent.com/translate_f#36) .................. 14

[2.4 Flash interruption](https://translate.googleusercontent.com/translate_f#36) .................. 14

[2.5 Option byte descriptions](https://translate.googleusercontent.com/translate_f#36) .................. 14

[2.6 Flash Register Descriptions](https://translate.googleusercontent.com/translate_f#38) ....................................................16

[2.6.1 Flash access control register (FLASH\_ACR)](https://translate.googleusercontent.com/translate_f#38) ....................................16

[2.6.2 Flash access control register (FLASH\_KEYR)](https://translate.googleusercontent.com/translate_f#39) ....................................17

[2.6.3 Flash OPTKEY register (FLASH\_OPTKEYR)](https://translate.googleusercontent.com/translate_f#39) ............ 17

[2.6.4 Flash Status Register (FLASH\_SR)](https://translate.googleusercontent.com/translate_f#40) .........................................18 is

[2.6.5 Flash Memory Control Register (FLASH\_CR)](https://translate.googleusercontent.com/translate_f#41) .......................................19

[2.6.6 Flash Memory Address Register (FLASH\_AR)](https://translate.googleusercontent.com/translate_f#42) .........................................20 is

[2.6.7 Register option byte (FLASH\_OBR)](https://translate.googleusercontent.com/translate_f#42) ...................................20 is

[2.6.8 Write Protection Register (FLASH\_WRPR)](https://translate.googleusercontent.com/translate_f#43) ...................................21

[**3** Cyclic Redundancy Check Calculation Unit **(CRC)**](https://translate.googleusercontent.com/translate_f#45)

**twenty three**

[3.1 Introduction to CRC](https://translate.googleusercontent.com/translate_f#45) ..................... 23

[3.2 Main Features of CRC](https://translate.googleusercontent.com/translate_f#45) .................. 23

[3.3 CRC Function Introduction](https://translate.googleusercontent.com/translate_f#45) ................... 23

[3.4 CRC register](https://translate.googleusercontent.com/translate_f#46) ................... 24

[3.4.1 CRC data register (CRC\_DR)](https://translate.googleusercontent.com/translate_f#46) ......................................24

[3.4.2 CRC independent data register (CRC\_IDR)](https://translate.googleusercontent.com/translate_f#46) ...................................24

[3.4.3 CRC control register (CRC\_CTRL)](https://translate.googleusercontent.com/translate_f#47) ...................................25

[**4** Power Control **(PWR)**](https://translate.googleusercontent.com/translate_f#48)**……………26**

[4.1 Power supply](https://translate.googleusercontent.com/translate_f#48) ....................26

[4.1.1 Independent A/D converter power supply and reference voltage](https://translate.googleusercontent.com/translate_f#48) ............. 26

[4.1.2 Voltage Regulator](https://translate.googleusercontent.com/translate_f#48) ................... 26

[4.2 Power Manager](https://translate.googleusercontent.com/translate_f#49) ..................27

1

|  |
| --- |
| **Page 3** |

[4.2.1 Reset (POR) and brown-out reset (the PDR)](https://translate.googleusercontent.com/translate_f#49) ......................................27

[4.2.2 programmable voltage monitor (the PVD)](https://translate.googleusercontent.com/translate_f#49) ............................................27

[4.3 Low power consumption mode](https://translate.googleusercontent.com/translate_f#50) .................. 28

[4.3.1 Reducing system clock](https://translate.googleusercontent.com/translate_f#51) ........................................29

[4.3.2 Control of external clock](https://translate.googleusercontent.com/translate_f#51) ................

[4.3.3 Sleep Mode](https://translate.googleusercontent.com/translate_f#51) .....................29

[4.3.4 Shutdown mode](https://translate.googleusercontent.com/translate_f#52) .................. 30

[4.3.5 Standby mode](https://translate.googleusercontent.com/translate_f#53) ....................31

[4.4 Power Control Register](https://translate.googleusercontent.com/translate_f#54) ..................32

[4.4.1 Power Control Register (PWR\_CR)](https://translate.googleusercontent.com/translate_f#54) .................................32

[4.4.2 Power Control / Status Register (PWR\_CSR)](https://translate.googleusercontent.com/translate_f#55) ..............................33 is

[**5** Reset and Clock Control **(RCC)**](https://translate.googleusercontent.com/translate_f#57)

**35**

[5.1 Reset](https://translate.googleusercontent.com/translate_f#57) ....................35

[5.1.1 System reset](https://translate.googleusercontent.com/translate_f#57) ................... 35

[5.1.2 Power reset](https://translate.googleusercontent.com/translate_f#57) ...................35

[5.2 Clock](https://translate.googleusercontent.com/translate_f#58) ..................... 36

[5.2.1](https://translate.googleusercontent.com/translate_f#60) [HSE clock](https://translate.googleusercontent.com/translate_f#60) .......................................38

[5.2.2 HSI clock](https://translate.googleusercontent.com/translate_f#61) ....................39

[5.2.3 LSI Clock](https://translate.googleusercontent.com/translate_f#61) ....................39

[5.2.4 System Clock (the SYSCLK) selection](https://translate.googleusercontent.com/translate_f#61) .......................................39

[5.2.5 Clock Security System (CSS)](https://translate.googleusercontent.com/translate_f#61) ..............................................39

[5.2.6 Watchdog clock](https://translate.googleusercontent.com/translate_f#62) ..................... 40

[5.2.7 Clock output](https://translate.googleusercontent.com/translate_f#62) ................... 40

[5.3 RCC register file and the memory map described](https://translate.googleusercontent.com/translate_f#62) .....................................40

[5.3.1 Clock Control Register (RCC\_CR)](https://translate.googleusercontent.com/translate_f#62) .......................................40

[5.3.2 Clock Configuration Register (RCC\_CFGR)](https://translate.googleusercontent.com/translate_f#64) ....................................42 is

[5.3.3 Clock Interrupt Register (RCC\_CIR)](https://translate.googleusercontent.com/translate_f#66) ................................44 is

[5.3.4 APB2 Peripheral Reset Register (RCC\_APB2RSTR)](https://translate.googleusercontent.com/translate_f#68) ............46

[5.3.5 APB1 Peripheral Reset Register (RCC\_APB1RSTR)](https://translate.googleusercontent.com/translate_f#69) ............47

[5.3.6 AHB Peripheral Clock Enable Register (RCC\_AHBENR)](https://translate.googleusercontent.com/translate_f#70) ............48

[5.3.7 APB2 Peripheral Clock Enable Register (RCC\_APB2ENR)](https://translate.googleusercontent.com/translate_f#71) ............49

[5.3.8 APB1 peripheral clock enable register (RCC\_APB1ENR)](https://translate.googleusercontent.com/translate_f#72) ............50

[5.3.9 Control Status Register (RCC\_CSR)](https://translate.googleusercontent.com/translate_f#73) ..............................51 is

[5.3.10 System Configuration Register (RCC\_SYSCFG)](https://translate.googleusercontent.com/translate_f#75) ...................................53 is

[**6** General function **I/O (GPIO)**](https://translate.googleusercontent.com/translate_f#77)**………….55**

[6.1 GPIO function description](https://translate.googleusercontent.com/translate_f#77) ....................55

[6.1.1 General I/O (GPIO)](https://translate.googleusercontent.com/translate_f#78) ...........................56

[6.1.2 single bit set or clear the bit](https://translate.googleusercontent.com/translate_f#79) ..........................................57

[6.1.3 External interrupt/wake-up line](https://translate.googleusercontent.com/translate_f#79) ..................57

[6.1.4 Multiplexing function](https://translate.googleusercontent.com/translate_f#79) ................... 57

[6.1.5 software re-mapped I / O multiplexing function](https://translate.googleusercontent.com/translate_f#79) ............................................57

[6.1.6 GPIO locking mechanism](https://translate.googleusercontent.com/translate_f#79) .............................................57

[6.1.7 Input configuration](https://translate.googleusercontent.com/translate_f#79) .................. 58

2

|  |
| --- |
| **Page 4** |

[6.1.8 Output configuration](https://translate.googleusercontent.com/translate_f#80) .................. 58

[6.1.9 Reuse function configuration](https://translate.googleusercontent.com/translate_f#81) ..................59

[6.1.10 analog input configuration](https://translate.googleusercontent.com/translate_f#82) .............................................60

[6.1.11 GPIO peripheral configuration](https://translate.googleusercontent.com/translate_f#83) ........................................61

[6.2 Multiplexed function I/O and debugging configuration](https://translate.googleusercontent.com/translate_f#84) ................ .63

[6.2.1 The OSC\_IN / OSC\_OUT as GPIO ports PD0 / PD1](https://translate.googleusercontent.com/translate_f#85) ............................63

[6.2.2 SWD complex remapping function](https://translate.googleusercontent.com/translate_f#85) .........................................63

[6.3 GPIO registers described](https://translate.googleusercontent.com/translate_f#85) ....................................................63

[6.3.1 Port Configuration Low Register (GPIOx\_CRL) (x = AD)](https://translate.googleusercontent.com/translate_f#85) ............63

[6.3.2 Port Configuration High Register (GPIOx\_CRH) (x = AD)](https://translate.googleusercontent.com/translate_f#86) ..............64

[6.3.3 Port input data register (GPIOx\_IDR) (x = AD)](https://translate.googleusercontent.com/translate_f#87) ..............65

[6.3.4 Port output data register (GPIOx\_ODR) (x = AD)](https://translate.googleusercontent.com/translate_f#87) ............. 65

[6.3.5 Port Setting/Clearing Register (GPIOx\_BSRR) (x = AD)](https://translate.googleusercontent.com/translate_f#88) ............66

[6.3.6 Port Bit Clear Register (GPIOx\_BRR) (x = AD)](https://translate.googleusercontent.com/translate_f#88) ............. 66

[6.3.7 Port Configuration Lock Register (GPIOx\_LCKR) (x = AD)](https://translate.googleusercontent.com/translate_f#89) ............67

[6.3.8 Port Multiplex Function Low Register (GPIOx\_AFRL) (x = AD)](https://translate.googleusercontent.com/translate_f#90) ... 68

[6.3.9 Port Multiplex Function High Register (GPIOx\_AFRH) (x = AD)](https://translate.googleusercontent.com/translate_f#90) ... 68

[**7** Interrupts and events **(EXTI)**](https://translate.googleusercontent.com/translate_f#92)**…………..70**

[7.1 Nested Vectored Interrupt Controller](https://translate.googleusercontent.com/translate_f#92) ................. .70

[7.1.1 System Tick (SysTick) Calibration Value Register](https://translate.googleusercontent.com/translate_f#92) .................70

[7.1.2 interrupt and exception vectors](https://translate.googleusercontent.com/translate_f#92) .........................................70

[7.2 External Interrupt/Event Controller (EXTI)](https://translate.googleusercontent.com/translate_f#93) ........................72

[7.2.1 Main Features](https://translate.googleusercontent.com/translate_f#94) ................. .72

[7.2.2 Block diagram](https://translate.googleusercontent.com/translate_f#94) ........................... 72

[7.2.3 wake event management](https://translate.googleusercontent.com/translate_f#94) ..................................72

[7.2.4 Functional description](https://translate.googleusercontent.com/translate_f#95) ................... 73

[7.2.5 External Interrupt / Event line image](https://translate.googleusercontent.com/translate_f#95) ..........................................73

[7.3 EXTI register description](https://translate.googleusercontent.com/translate_f#96) ....................74

[7.3.1 Interrupt Mask Register (EXTI\_IMR)](https://translate.googleusercontent.com/translate_f#97) .................................75

[7.3.2 Event Mask Register (EXTI\_EMR)](https://translate.googleusercontent.com/translate_f#97) .............................................75

[7.3.3 Rising Edge Trigger Selection Register (EXTI\_RTSR)](https://translate.googleusercontent.com/translate_f#98) ............76

[7.3.4 Falling Edge Trigger Selection Register (EXTI\_FTSR)](https://translate.googleusercontent.com/translate_f#98) ............ 76

[7.3.5 Software interrupt event register (EXTI\_SWIER)](https://translate.googleusercontent.com/translate_f#99) ................. 77

[7.3.6 software interrupt event register (EXTI\_PR)](https://translate.googleusercontent.com/translate_f#100) ...................................78

[**8 DMA** controller **(DMA)**](https://translate.googleusercontent.com/translate_f#101)**…………………79**

[8.1 Introduction to DMA](https://translate.googleusercontent.com/translate_f#101) .................. .. 79

[8.2 Main Features of DMA](https://translate.googleusercontent.com/translate_f#101) .................. 79

[8.3 Functional description](https://translate.googleusercontent.com/translate_f#102) ....................80

[8.3.1 DMA processing](https://translate.googleusercontent.com/translate_f#102) ...................80

[8.3.2 Arbitrator](https://translate.googleusercontent.com/translate_f#102) .................... 80

[8.3.3 DMA channel](https://translate.googleusercontent.com/translate_f#103) ...................81

[8.3.4 Programmable data transmission width, alignment and data size end](https://translate.googleusercontent.com/translate_f#104) .......... 82

[8.3.5 Error Management](https://translate.googleusercontent.com/translate_f#105) ..................83

3

|  |
| --- |
| **Page 5** |

[8.3.6 Interruption](https://translate.googleusercontent.com/translate_f#105) ................... 83

[8.3.7 DMA request image](https://translate.googleusercontent.com/translate_f#106) .................... 84

[8.4 DMA register description](https://translate.googleusercontent.com/translate_f#107) ...................85

[8.4.1 DMA Interrupt Status Register (DMA\_ISR)](https://translate.googleusercontent.com/translate_f#107) ............... 85

[8.4.2 DMA Interrupt Flag Clear Register (DMA\_IFCR)](https://translate.googleusercontent.com/translate_f#108) ..............86

[8.4.3 DMA channel x configuration register (DMA\_CCRx) (x = 1…5)](https://translate.googleusercontent.com/translate_f#109) .......... 87

[8.4.4 DMA channel x transfer quantity register (DMA\_CNDTRx) (x = 1...5)](https://translate.googleusercontent.com/translate_f#111) ...........89

[8.4.5 DMA channel x peripheral address register (DMA\_CPARx) (x = 1…5)](https://translate.googleusercontent.com/translate_f#111) .......... 89

[8.4.6 DMA channel x memory address register (DMA\_CMARx) (x = 1…5)](https://translate.googleusercontent.com/translate_f#112) .......... 90

[**9** Analog **/** Digital Conversion **(ADC)**](https://translate.googleusercontent.com/translate_f#113)**………………….91**

[9.1 Introduction to ADC](https://translate.googleusercontent.com/translate_f#113) ................... 91

[9.2 ADC main features](https://translate.googleusercontent.com/translate_f#113) ......................... 91

[9.3 ADC function description](https://translate.googleusercontent.com/translate_f#113) ....................91

[9.3.1 ADC switch control](https://translate.googleusercontent.com/translate_f#114) ....................92

[9.3.2 Channel selection](https://translate.googleusercontent.com/translate_f#114) ....................92

[9.4 ADC working mode](https://translate.googleusercontent.com/translate_f#114) ................... 92

[9.4.1 Single conversion mode](https://translate.googleusercontent.com/translate_f#115) ............. 93

[9.4.2 Single-cycle scan mode](https://translate.googleusercontent.com/translate_f#115) .................................................. 93

[9.4.3 Continuous scan mode](https://translate.googleusercontent.com/translate_f#116) .......................................94

[9.4.4 DMA request](https://translate.googleusercontent.com/translate_f#117) .............95

[9.5 Data Alignment](https://translate.googleusercontent.com/translate_f#117) ..................95

[9.5.1 Programmable resolution](https://translate.googleusercontent.com/translate_f#118) ................... 96

[9.5.2 Programmable sample time](https://translate.googleusercontent.com/translate_f#118) ............................................96

[9.6 Externally triggered conversions](https://translate.googleusercontent.com/translate_f#118) .................. .. 96

[9.7 Temperature sensor](https://translate.googleusercontent.com/translate_f#118) ................... 97

[9.8 Internal reference voltage](https://translate.googleusercontent.com/translate_f#119) .................97

[9.9 AD conversion result monitoring in window comparator mode](https://translate.googleusercontent.com/translate_f#119) ..............

[9.10 ADC register description](https://translate.googleusercontent.com/translate_f#119) .................... 97

[9.10.1 A / D data register (ADC\_ADDATA)](https://translate.googleusercontent.com/translate_f#120) ..................................98

[9.10.2 A / D configuration register (ADC\_ADCFG)](https://translate.googleusercontent.com/translate_f#121) ........................................99

[9.10.3 A / D control register (ADC\_ADCR)](https://translate.googleusercontent.com/translate_f#122) .........................................100

[9.10.4 A/D channel selection register (ADC\_ADCHS)](https://translate.googleusercontent.com/translate_f#124) ............. 103

[9.10.5 A/D Window Compare Register (ADC\_ADCMPR)](https://translate.googleusercontent.com/translate_f#126) ............104

[9.10.6 A / D status register (ADC\_ADSTA)](https://translate.googleusercontent.com/translate_f#126) ........................................104

[9.10.7 A/D Data Register (ADC\_ADDR0, 4 ∼ 6, 9 ∼ 12, 14∼ 15)](https://translate.googleusercontent.com/translate_f#127) ........ 105

[9.10.8 A/D Extended Status Register (ADC\_ADSTA\_EXT)](https://translate.googleusercontent.com/translate_f#128) ... 106

[**10** Advanced Control Timer **(TIM1)**](https://translate.googleusercontent.com/translate_f#129)**………..107**

[10.1 Introduction to TIM1](https://translate.googleusercontent.com/translate_f#129) ..................... 107

[10.2 Main Features](https://translate.googleusercontent.com/translate_f#129) ................... 107

[10.3 Functional description](https://translate.googleusercontent.com/translate_f#130) ....................108

[10.3.1 Time base unit](https://translate.googleusercontent.com/translate_f#130) ...................108

[10.3.2 Counting mode](https://translate.googleusercontent.com/translate_f#132) ................... 110

[10.3.3 Repeat Counter](https://translate.googleusercontent.com/translate_f#141) ....................119

4

|  |
| --- |
| **Page 6** |

[10.3.4 Clock selection](https://translate.googleusercontent.com/translate_f#142) ................... 120

[10.3.5 Capture/Compare Channel](https://translate.googleusercontent.com/translate_f#145) ..................123

[10.3.6 input capture mode](https://translate.googleusercontent.com/translate_f#147) ........................................125

[10.3.7 PWM input mode](https://translate.googleusercontent.com/translate_f#148) .....................126

[10.3.8 forced output mode](https://translate.googleusercontent.com/translate_f#149) .......................................127

[10.3.9 Compare output mode](https://translate.googleusercontent.com/translate_f#149) .......................................127

[10.3.10 PWM mode](https://translate.googleusercontent.com/translate_f#150) .......................................128

[10.3.11 complementary outputs and dead-time insertion](https://translate.googleusercontent.com/translate_f#153) .........................................131

[10.3.12 Using the brake function](https://translate.googleusercontent.com/translate_f#154) ................. .. 132

[10.3.13 Clear OCxREF signal during external events](https://translate.googleusercontent.com/translate_f#156) ........134

[Generating output PWM six-step 10.3.14](https://translate.googleusercontent.com/translate_f#157) .........................................135

[10.3.15 Single pulse mode](https://translate.googleusercontent.com/translate_f#158) .............136

[10.3.16 encoder interface mode](https://translate.googleusercontent.com/translate_f#160) ........................................138

[10.3.17 timer-input exclusive OR function](https://translate.googleusercontent.com/translate_f#162) .........................................140

[10.3.18 Interface with Hall sensor](https://translate.googleusercontent.com/translate_f#162) ................

[10.3.19 TIMx timer and external trigger synchronization](https://translate.googleusercontent.com/translate_f#163) .............141

[10.3.20 Timer synchronization](https://translate.googleusercontent.com/translate_f#166) ...................144

[10.3.21 Debugging mode](https://translate.googleusercontent.com/translate_f#166) ..................144

[10.4 Register description](https://translate.googleusercontent.com/translate_f#166) ....................144

[10.4.1 Control Register. 1 (the TIMx\_CR1)](https://translate.googleusercontent.com/translate_f#167) .......................................145

[10.4.2 Control Register 2 (the TIMx\_CR2)](https://translate.googleusercontent.com/translate_f#168) .............................................146

[10.4.3 from the mode control register (TIMx\_SMCR)](https://translate.googleusercontent.com/translate_f#170) ...................................149

[10.4.4 DMA/Interrupt Enable Register (TIMX\_DIER)](https://translate.googleusercontent.com/translate_f#173) ..............151

[10.4.5 Status Register (TIMx\_SR)](https://translate.googleusercontent.com/translate_f#175) .............................................153

[10.4.6 event generation register (the TIMx\_EGR)](https://translate.googleusercontent.com/translate_f#177) .............................................155

[10.4.7 Capture/Compare Mode Register 1 (TIMx\_CCMR1)](https://translate.googleusercontent.com/translate_f#179) ... 157

[10.4.8 Capture/Compare Mode Register 2 (TIMx\_CCMR2)](https://translate.googleusercontent.com/translate_f#183) ............161

[10.4.9 Capture/Compare Enable Register (TIMx\_CCER)](https://translate.googleusercontent.com/translate_f#184) .............163

[10.4.10 counter (as TIMx\_CNT)...](https://translate.googleusercontent.com/translate_f#187) ....................................166

[10.4.11 Prescaler (TIMx\_PSC)](https://translate.googleusercontent.com/translate_f#188) ...........................166

[10.4.12 auto-reload register (in TIMx\_ARR)](https://translate.googleusercontent.com/translate_f#188) .............................................166

[10.4.13 Repeat Count Register (TIMx\_RCR)](https://translate.googleusercontent.com/translate_f#188) .............................................166

[10.4.14 capture / compare register. 1 (the TIMx\_CCR1)](https://translate.googleusercontent.com/translate_f#189) .................................167 is

[10.4.15 capture / compare register 2 (TIMx\_CCR2)](https://translate.googleusercontent.com/translate_f#189) .......................................168

[10.4.16 capture / compare register. 3 (TIMx\_CCR3)](https://translate.googleusercontent.com/translate_f#190) .......................................168

[10.4.17 Capture/Compare Register 4 (TIMx\_CCR4)](https://translate.googleusercontent.com/translate_f#191) ...............169

[10.4.18 Brake and Dead Band Register (TIMx\_BDTR)](https://translate.googleusercontent.com/translate_f#191) ..................169

[10.4.19 DMA control register (TIMx\_DCR)](https://translate.googleusercontent.com/translate_f#194) ........................................172

[10.4.20 Continuous Mode DMA Address (TIMx\_DMAR)](https://translate.googleusercontent.com/translate_f#195) .............173

[10.4.21 Capture/Compare Mode Register 3 (TIMx\_CCMR3)](https://translate.googleusercontent.com/translate_f#196) ... 174

[10.4.22 capture / compare register. 5 (TIMx\_CCR5)](https://translate.googleusercontent.com/translate_f#196) .......................................174

[**11 16** -bit general-purpose timer **(TIMx16 Bit)**](https://translate.googleusercontent.com/translate_f#198)

**176**

[11.1 Introduction to TIMx](https://translate.googleusercontent.com/translate_f#198) ................... .. 176

5

|  |
| --- |
| **Page 7** |

[11.2 TIMx main functions](https://translate.googleusercontent.com/translate_f#198) .................... [.176](https://translate.googleusercontent.com/translate_f#198)

[11.3 TIMx functional description](https://translate.googleusercontent.com/translate_f#199) ................... 177

[11.3.1 Time base unit](https://translate.googleusercontent.com/translate_f#199) ................... 177

[11.3.2 Counting mode](https://translate.googleusercontent.com/translate_f#201) .................... 179

[11.3.3 Clock selection](https://translate.googleusercontent.com/translate_f#209) .........................187

[11.3.4 Capture/Compare Channel](https://translate.googleusercontent.com/translate_f#212) .................191

[11.3.5 input capture mode](https://translate.googleusercontent.com/translate_f#214) ........................................193

[11.3.6 PWM input mode](https://translate.googleusercontent.com/translate_f#215) .......................................193

[11.3.7 forced output mode](https://translate.googleusercontent.com/translate_f#216) ........................................194

[11.3.8 Compare output mode](https://translate.googleusercontent.com/translate_f#216) .......................................194

[11.3.9 PWM mode](https://translate.googleusercontent.com/translate_f#217) ...................195

[11.3.10 Single pulse mode](https://translate.googleusercontent.com/translate_f#220) .............198

[11.3.11 Clear the OCxREF signal during external events](https://translate.googleusercontent.com/translate_f#222) ............200

[11.3.12 encoder interface mode](https://translate.googleusercontent.com/translate_f#222) ..................................200

[11.3.13 timer-input exclusive OR function](https://translate.googleusercontent.com/translate_f#224) .........................................203

[11.3.14 Synchronization of Timer and External Trigger](https://translate.googleusercontent.com/translate_f#225) ..............

[11.3.15 Timer Synchronization](https://translate.googleusercontent.com/translate_f#228) ....................206

[11.3.16 Debugging mode](https://translate.googleusercontent.com/translate_f#232) ............210

[11.4 TIMx register description](https://translate.googleusercontent.com/translate_f#232) ..............210

[11.4.1 Control Register. 1 (the TIMx\_CR1)](https://translate.googleusercontent.com/translate_f#233) .......................................211

[11.4.2 Control Register 2 (the TIMx\_CR2)](https://translate.googleusercontent.com/translate_f#235) .......................................213

[11.4.3 from the mode control register (TIMx\_SMCR)](https://translate.googleusercontent.com/translate_f#236) ...................................214

[11.4.4 DMA / Interrupt Enable Register (TIMx\_DIER)](https://translate.googleusercontent.com/translate_f#239) .................................217

[11.4.5 Status Register (TIMx\_SR)](https://translate.googleusercontent.com/translate_f#241) ........................................219

[11.4.6 event generation register (the TIMx\_EGR)](https://translate.googleusercontent.com/translate_f#242) .............................................220

[11.4.7 Capture/Compare Mode Register 1 (TIMx\_CCMR1)](https://translate.googleusercontent.com/translate_f#243) .............221

[11.4.8 Capture/Compare Mode Register 2 (TIMx\_CCMR2)](https://translate.googleusercontent.com/translate_f#247) ............225

[11.4.9 Capture/Compare Enable Register (TIMx\_CCER)](https://translate.googleusercontent.com/translate_f#249) .............227

[11.4.10 counter (TIMx\_CNT)...](https://translate.googleusercontent.com/translate_f#250) ....................................229

[11.4.11 Prescaler (TIMx\_PSC)](https://translate.googleusercontent.com/translate_f#251) ...........................229

[11.4.12 auto-reload register (in TIMx\_ARR)](https://translate.googleusercontent.com/translate_f#251) .............................................229

[11.4.13 Capture/Compare Register 1 (TIMx\_CCR1)](https://translate.googleusercontent.com/translate_f#251) .................229

[11.4.14 Capture/Compare Register 2 (TIMx\_CCR2)](https://translate.googleusercontent.com/translate_f#252) ................ 230

[11.4.15 Capture/Compare Register 3 (TIMx\_CCR3)](https://translate.googleusercontent.com/translate_f#252) ..............231

[11.4.16 Capture/Compare Register 4 (TIMx\_CCR4)](https://translate.googleusercontent.com/translate_f#253) ..............231

[11.4.17 DMA control register (TIMx\_DCR)](https://translate.googleusercontent.com/translate_f#254) ........................................232

[11.4.18 continuous mode DMA address (TIMx\_DMAR)](https://translate.googleusercontent.com/translate_f#255) .................................233

[**12 32** -bit general-purpose timer **(TIMx32 Bit)**](https://translate.googleusercontent.com/translate_f#256)**……….234**

[12.1 Introduction to TIMx](https://translate.googleusercontent.com/translate_f#256) .................... .. 234

[12.2 TIMx main functions](https://translate.googleusercontent.com/translate_f#256) .................... 234

[12.3 TIMx functional description](https://translate.googleusercontent.com/translate_f#257) .................... [.235](https://translate.googleusercontent.com/translate_f#257)

[12.3.1 Time base unit](https://translate.googleusercontent.com/translate_f#257) ...................235

[12.3.2 Counting mode](https://translate.googleusercontent.com/translate_f#259) ................... 237

6

|  |
| --- |
| **Page 8** |

[12.3.3 Clock selection](https://translate.googleusercontent.com/translate_f#267) ...................245

[12.3.4 Capture/Compare Channel](https://translate.googleusercontent.com/translate_f#270) ..................249

[12.3.5 input capture mode](https://translate.googleusercontent.com/translate_f#272) ..................................251

[12.3.6 PWM input mode](https://translate.googleusercontent.com/translate_f#273) .................................251

[12.3.7 forced output mode](https://translate.googleusercontent.com/translate_f#274) ........................................252

[12.3.8 Compare output mode](https://translate.googleusercontent.com/translate_f#274) .......................................252

[12.3.9 PWM mode](https://translate.googleusercontent.com/translate_f#275) ....................253

[12.3.10 Single pulse mode](https://translate.googleusercontent.com/translate_f#278) ...................256

[12.3.11 Clear OCxREF signal during external events...](https://translate.googleusercontent.com/translate_f#280) 258

[12.3.12 encoder interface mode](https://translate.googleusercontent.com/translate_f#280) .......................................258

[12.3.13 timer-input exclusive OR function](https://translate.googleusercontent.com/translate_f#282) ........................................261

[12.3.14 Synchronization of Timer and External Trigger](https://translate.googleusercontent.com/translate_f#283) .............

[12.3.15 Timer Synchronization](https://translate.googleusercontent.com/translate_f#286) ...................264

[12.3.16 Debugging mode](https://translate.googleusercontent.com/translate_f#290) ..................268

[12.4 TIMx register description](https://translate.googleusercontent.com/translate_f#290) ...................268

[12.4.1 Control Register. 1 (the TIMx\_CR1)](https://translate.googleusercontent.com/translate_f#291) .............................................269

[12.4.2 Control Register 2 (the TIMx\_CR2)](https://translate.googleusercontent.com/translate_f#293) .............................................271

[12.4.3 from the mode control register (TIMx\_SMCR)](https://translate.googleusercontent.com/translate_f#294) .......................................272

[12.4.4 DMA / Interrupt Enable Register (TIMx\_DIER)](https://translate.googleusercontent.com/translate_f#297) .......................................275

[12.4.5 Status Register (TIMx\_SR)](https://translate.googleusercontent.com/translate_f#299) ........................................277

[12.4.6 event generation register (the TIMx\_EGR)](https://translate.googleusercontent.com/translate_f#300) .............................................278

[12.4.7 Capture/Compare Mode Register 1 (TIMx\_CCMR1)](https://translate.googleusercontent.com/translate_f#301) ... 279

[12.4.8 Capture/Compare Mode Register 2 (TIMx\_CCMR2)](https://translate.googleusercontent.com/translate_f#305) ............283

[12.4.9 Capture / Compare Enable Register (TIMx\_CCER)](https://translate.googleusercontent.com/translate_f#307) ..............................285

[12.4.10 counter (as TIMx\_CNT)...](https://translate.googleusercontent.com/translate_f#308) ....................................287

[12.4.11 prescaler (TIMx\_PSC)](https://translate.googleusercontent.com/translate_f#309) .............................................287

[12.4.12 auto-reload register (in TIMx\_ARR)](https://translate.googleusercontent.com/translate_f#309) ...................................................287

[12.4.13 capture / compare register. 1 (the TIMx\_CCR1)](https://translate.googleusercontent.com/translate_f#310) .......................................288

[12.4.14 capture / compare register 2 (TIMx\_CCR2)](https://translate.googleusercontent.com/translate_f#310) .......................................288

[12.4.15 capture / compare register. 3 (TIMx\_CCR3)](https://translate.googleusercontent.com/translate_f#311) .......................................289

[12.4.16 capture / compare register. 4 (TIMx\_CCR4)](https://translate.googleusercontent.com/translate_f#311) .................................290 is

[12.4.17 DMA control register (TIMx\_DCR)](https://translate.googleusercontent.com/translate_f#312) ..................................290 is

[12.4.18 DMA address in continuous mode (TIMx\_DMAR)](https://translate.googleusercontent.com/translate_f#313) ............291

[**13** Basic timer **(TIM14)**](https://translate.googleusercontent.com/translate_f#315)

**293**

[13.1 Introduction to TIM14](https://translate.googleusercontent.com/translate_f#315) .................... .293

[13.2 TIM14 main features](https://translate.googleusercontent.com/translate_f#315) .............293

[13.3 TIM14 functional description](https://translate.googleusercontent.com/translate_f#316) ...................294

[13.3.1 Time base unit](https://translate.googleusercontent.com/translate_f#316) ...................294

[13.3.2 Counting mode](https://translate.googleusercontent.com/translate_f#317) .................... 295

[13.3.3 Repeat Counter](https://translate.googleusercontent.com/translate_f#320) ...................298

[13.3.4 Clock source](https://translate.googleusercontent.com/translate_f#321) ...................299

[13.3.5 Capture/Compare Channel](https://translate.googleusercontent.com/translate_f#322) .................300

[13.3.6 input capture mode](https://translate.googleusercontent.com/translate_f#323) ........................................302

7

|  |
| --- |
| **Page 9** |

[13.3.7 forced output mode](https://translate.googleusercontent.com/translate_f#324) ........................................302

[13.3.8 Compare output mode](https://translate.googleusercontent.com/translate_f#324) .......................................303

[13.3.9 PWM mode](https://translate.googleusercontent.com/translate_f#325) ...................304

[13.3.10 Debugging mode](https://translate.googleusercontent.com/translate_f#326) ..................305

[13.4 TIM14 register descriptions](https://translate.googleusercontent.com/translate_f#327) ....................................................305

[13.4.1 Control Register. 1 (TIM14\_CR1)](https://translate.googleusercontent.com/translate_f#327) .......................................305

[13.4.2 Control Register 2 (TIM14\_CR2)](https://translate.googleusercontent.com/translate_f#328) .......................................306

[13.4.3 Interrupt Enable Register (TIM14\_DIER)](https://translate.googleusercontent.com/translate_f#329) .......................................307

[13.4.4 Status Register (TIM14\_SR)](https://translate.googleusercontent.com/translate_f#330) .......................................308

[13.4.5 event generation register (TIM14\_EGR)](https://translate.googleusercontent.com/translate_f#331) .........................................309

[13.4.6 Capture/Compare Mode Register 1 (TIM14\_CCMR1)](https://translate.googleusercontent.com/translate_f#331) ... 309

[13.4.7 Capture/Compare Enable Register (TIM14\_CCER)](https://translate.googleusercontent.com/translate_f#334) ...........312

[13.4.8 counter (TIM14\_CNT)...](https://translate.googleusercontent.com/translate_f#336) ....................................314

[13.4.9 prescaler (TIM14\_PSC)](https://translate.googleusercontent.com/translate_f#336) .............................................314

[13.4.10 auto-reload register (TIM14\_ARR)](https://translate.googleusercontent.com/translate_f#336) .............................................314

[13.4.11 Repeat Count Register (TIM14\_RCR)](https://translate.googleusercontent.com/translate_f#337) ...................................315

[13.4.12 Capture/Compare Register 1 (TIM14\_CCR1)](https://translate.googleusercontent.com/translate_f#337) .............315

[13.4.13 brakes and dead register (TIM14\_BDTR)](https://translate.googleusercontent.com/translate_f#338) .......................................316

[13.4.14 DMA control register (TIM14\_DCR)](https://translate.googleusercontent.com/translate_f#338) .......................................316

[13.4.15 Full transfer address register (TIM14\_DMAR)](https://translate.googleusercontent.com/translate_f#339) ...................................317

[**14** Basic timer **(TIM16/17)**](https://translate.googleusercontent.com/translate_f#341)**………319**

[14.1 Introduction to TIM16/17](https://translate.googleusercontent.com/translate_f#341) .................... .319

[14.2 Main Features](https://translate.googleusercontent.com/translate_f#341) ................... 319

[14.3 Functional description](https://translate.googleusercontent.com/translate_f#342) ..................... 320

[14.3.1 Time base unit](https://translate.googleusercontent.com/translate_f#342) ................... 320

[14.3.2 Counting mode](https://translate.googleusercontent.com/translate_f#343) ................... 321

[14.3.3 Repeat Counter](https://translate.googleusercontent.com/translate_f#346) ....................324

[14.3.4 Clock source](https://translate.googleusercontent.com/translate_f#347) ...................325

[14.3.5 Capture/Compare Channel](https://translate.googleusercontent.com/translate_f#348) ..................326

[14.3.6 Input Capture mode](https://translate.googleusercontent.com/translate_f#349) ..................................328

[14.3.7 forced output mode](https://translate.googleusercontent.com/translate_f#350) ........................................328

[14.3.8 Compare output mode](https://translate.googleusercontent.com/translate_f#351) .................................329

[14.3.9 PWM mode](https://translate.googleusercontent.com/translate_f#352) ..............330

[14.3.10 complementary outputs and dead-time insertion](https://translate.googleusercontent.com/translate_f#353) ..................331

[14.3.11 brakes function](https://translate.googleusercontent.com/translate_f#355) ........................................333

[14.3.12 Single-pulse mode](https://translate.googleusercontent.com/translate_f#356) ...................335

[14.3.13 Debugging mode](https://translate.googleusercontent.com/translate_f#358) ..................336

[14.4 Register description](https://translate.googleusercontent.com/translate_f#358) ...................336

[14.4.1 TIM16/17 Control Register 1 (TIM16/17\_CR1)](https://translate.googleusercontent.com/translate_f#358) ... 336

[14.4.2 TIM16/17 Control Register 2 (TIM16/17\_CR2)](https://translate.googleusercontent.com/translate_f#359) ... 337

[14.4.3 TIM16/17 Interrupt Enable Register (TIM16/17\_DIER)](https://translate.googleusercontent.com/translate_f#360) .........338

[14.4.4 TIM16/17 Status Register (TIM16/17\_SR)](https://translate.googleusercontent.com/translate_f#361) .............339

[14.4.5 TIM16/17 Event Generation Register 1 (TIM16/17\_EGR)](https://translate.googleusercontent.com/translate_f#363) .........341

8

|  |
| --- |
| **Page 10** |

[14.4.6 TIM16/17 Capture/Compare Mode Register 1 (TIM16/17\_CCMR1)](https://translate.googleusercontent.com/translate_f#364) ........342

[14.4.7 TIM16/17 Capture/Compare Enable Register (TIM16/17\_CCER)](https://translate.googleusercontent.com/translate_f#367) .........345

[14.4.8 TIM16 / 17 Counter (TIM16 / 17\_CNT)...](https://translate.googleusercontent.com/translate_f#369) ..............................348

[14.4.9 TIM16/17 Prescaler Register (TIM16/17\_PSC)](https://translate.googleusercontent.com/translate_f#370) ... 348

[14.4.10 TIM16/17 Automatic Reload Register (TIM16/17\_ARR)](https://translate.googleusercontent.com/translate_f#370) ...........348

[14.4.11 TIM16/17 Repeat Count Register (TIM16/17\_RCR)](https://translate.googleusercontent.com/translate_f#370) ...............348

[14.4.12 TIM16/17 Capture/Compare Register 1 (TIM16/17\_CCR1)](https://translate.googleusercontent.com/translate_f#371) ............349

[14.4.13 TIM16/17 Brake and Dead Time Register (TIM16/17\_BDTR)](https://translate.googleusercontent.com/translate_f#371) ...............349

[14.4.14 TIM16/17 DMA Control Register (TIM16/17\_DCR)](https://translate.googleusercontent.com/translate_f#374) .........352

[14.4.15 TIM16/17 Full Transfer Address Register (TIM16/17\_DMAR)](https://translate.googleusercontent.com/translate_f#375) ...............353

[**15** Independent Watchdog **(IWDG)**](https://translate.googleusercontent.com/translate_f#377)**………..355**

[15.1 (Introduction to IWDG)](https://translate.googleusercontent.com/translate_f#377) .................................355

[15.2 IWDG Main Performance](https://translate.googleusercontent.com/translate_f#377) ...................355

[15.3 IWDG Functional Description](https://translate.googleusercontent.com/translate_f#377) ....................355

[15.3.1 Hardware watchdog](https://translate.googleusercontent.com/translate_f#378) ................... .356

[15.3.2 register access protection](https://translate.googleusercontent.com/translate_f#378) ........................................356

[15.3.3 Debugging mode](https://translate.googleusercontent.com/translate_f#378) ..................357

[15.4 IWDG register descriptions](https://translate.googleusercontent.com/translate_f#379) ....................................................357

[15.4.1 key register (IWDG\_KR)](https://translate.googleusercontent.com/translate_f#379) ..............................................357

[15.4.2 prescaler register IWDG\_PR](https://translate.googleusercontent.com/translate_f#379) .........................................357

[15.4.3 reload registers (IWDG\_RLR)](https://translate.googleusercontent.com/translate_f#380) ...................................358

[15.4.4 Status Register (IWDG\_SR)](https://translate.googleusercontent.com/translate_f#381) .......................................359

[15.4.5 IWDG control register (IWDG\_CR)](https://translate.googleusercontent.com/translate_f#381) .......................................359

[**16** Window Watchdog **(WWDG)**](https://translate.googleusercontent.com/translate_f#383)**………..361**

[16.1 Introduction to WWDG](https://translate.googleusercontent.com/translate_f#383) ....................... [.361](https://translate.googleusercontent.com/translate_f#383)

[16.2 Main Features of WWDG](https://translate.googleusercontent.com/translate_f#383) ..............361

[16.3 WWDG functional description](https://translate.googleusercontent.com/translate_f#383) .............................................361

[16.4 How to write a watchdog timeout procedure](https://translate.googleusercontent.com/translate_f#384) ...............................................362

[16.5 Debugging mode](https://translate.googleusercontent.com/translate_f#385) .........................363

[16.6 WWDG Register Description](https://translate.googleusercontent.com/translate_f#386) ...............364

[16.6.1 Control Register (the WWDG\_CR)](https://translate.googleusercontent.com/translate_f#386) .......................................364

[16.6.2 Configuration Register (WWDG\_CFR)](https://translate.googleusercontent.com/translate_f#386) .......................................364

[16.6.3 Status Register (WWDG\_SR)](https://translate.googleusercontent.com/translate_f#387) .......................................365

[**17** Serial Peripheral Interface **(SPI)**](https://translate.googleusercontent.com/translate_f#388)**………366**

[17.1 Brief description of SPI](https://translate.googleusercontent.com/translate_f#388) ...........................366

[17.2 Main Features](https://translate.googleusercontent.com/translate_f#388) ................... 366

[17.3 SPI functional description](https://translate.googleusercontent.com/translate_f#388) ...................366

[17.3.1 Overview](https://translate.googleusercontent.com/translate_f#388) ...................366

[17.3.2 SPI slave mode](https://translate.googleusercontent.com/translate_f#392) ..................371

[17.3.3 SPI Master Mode](https://translate.googleusercontent.com/translate_f#393) ..................371

[17.3.4 Status Marks](https://translate.googleusercontent.com/translate_f#394) .................... 372

[17.3.5 Baud rate setting](https://translate.googleusercontent.com/translate_f#394) ................... 373

[17.3.6 SPI communication using DMA](https://translate.googleusercontent.com/translate_f#395) .............

9

|  |
| --- |
| **Page 11** |

[17.4 memory mapped register file description](https://translate.googleusercontent.com/translate_f#395) .....................................373

[17.4.1 Transmit Data Register (SPI\_TXREG)](https://translate.googleusercontent.com/translate_f#396) ....................................374

[17.4.2 receive data register (SPI\_RXREG)](https://translate.googleusercontent.com/translate_f#396) ....................................374

[17.4.3 Current Status Register (SPI\_CSTAT)](https://translate.googleusercontent.com/translate_f#396) ..............................374

[17.4.4 Interrupt Status Register (SPI\_INTSTAT)](https://translate.googleusercontent.com/translate_f#397) ...................................375

[17.4.5 Interrupt Enable Register (SPI\_INTEN)](https://translate.googleusercontent.com/translate_f#399) ....................................377

[17.4.6 Interrupt Clear Register (SPI\_INTCLR)](https://translate.googleusercontent.com/translate_f#400) ....................................378

[17.4.7 global control register (SPI\_GCTL)](https://translate.googleusercontent.com/translate_f#401) ..............................379

[17.4.8 General Control Register (SPI\_CCTL)](https://translate.googleusercontent.com/translate_f#403) ....................................381

[17.4.9 baud rate generator (SPI\_SPBRG)](https://translate.googleusercontent.com/translate_f#404) ....................................382

[17.4.10 Received Data Number Register (SPI\_RXDNR)](https://translate.googleusercontent.com/translate_f#404) ............382

[17.4.11 Slave Chip Select Register (SPI\_NSSR)](https://translate.googleusercontent.com/translate_f#405) ....................................383

[17.4.12 Control Data Register (SPI\_EXTCTL)](https://translate.googleusercontent.com/translate_f#405) ....................................383

[**18 I2C** interface **(I2C)**](https://translate.googleusercontent.com/translate_f#407)**……..385**

[18.1 Introduction to I2C](https://translate.googleusercontent.com/translate_f#407) ..................... 385

[18.2 I2C Main Features](https://translate.googleusercontent.com/translate_f#407) .................... 385

[18.3 I2C Agreement](https://translate.googleusercontent.com/translate_f#407) ..........................385

[18.3.1 Start and stop conditions](https://translate.googleusercontent.com/translate_f#407) ..................385

[18.3.2 Slave Addressing Protocol](https://translate.googleusercontent.com/translate_f#408) ................... .386

[18.3.3 transmit and receive protocols...](https://translate.googleusercontent.com/translate_f#409) ....................................387

[18.3.4 Send buffer management and start, stop and repeat start condition generation](https://translate.googleusercontent.com/translate_f#411) ......... 389

[18.3.5 Multi-host arbitration](https://translate.googleusercontent.com/translate_f#412) ..................390

[18.3.6 Clock Synchronization](https://translate.googleusercontent.com/translate_f#413) ................... 391

[18.4 I2C working mode](https://translate.googleusercontent.com/translate_f#413) ................... 392

[18.4.1 Slave mode](https://translate.googleusercontent.com/translate_f#414) ..........................392

[18.4.2 Main Mode](https://translate.googleusercontent.com/translate_f#416) ...................394

[18.4.3 I2C stop the transfer](https://translate.googleusercontent.com/translate_f#417) ................................................... 395

[18.5 Communication using DMA](https://translate.googleusercontent.com/translate_f#417) ...................395

[18.6 I2C interruption](https://translate.googleusercontent.com/translate_f#418) .................... 396

[18.7 I2C register description](https://translate.googleusercontent.com/translate_f#419) .................... .397

[18.7.1 I2C control register (I2C\_CR)](https://translate.googleusercontent.com/translate_f#420) ......................................398

[18.7.2 I2C Destination Address Register (I2C\_TAR)](https://translate.googleusercontent.com/translate_f#422) ..............................400

[18.7.3 I2C Slave Address Register (I2C\_SAR)](https://translate.googleusercontent.com/translate_f#423) ... 401

[18.7.4 I2C Data Command Register (I2C\_DR)](https://translate.googleusercontent.com/translate_f#423) ................401

[18.7.5 Standard Mode I2C Clock High-level Counter Register (I2C\_SSHR)](https://translate.googleusercontent.com/translate_f#424) ...........402

[18.7.6 Standard Mode I2C Clock Low-Level Count Register (I2C\_SSLR)](https://translate.googleusercontent.com/translate_f#424) .......... 402

[18.7.7 Fast Mode I2C Clock High-level Counter Register (I2C\_FSHR)](https://translate.googleusercontent.com/translate_f#424) ...........402

[18.7.8 Fast Mode I2C Clock Low-Level Count Register (I2C\_FSLR)](https://translate.googleusercontent.com/translate_f#425) ... ...... 403

[18.7.9 I2C Interrupt Status Register (I2C\_ISR)](https://translate.googleusercontent.com/translate_f#425) ...................................403

[18.7.10 I2C Interrupt Mask Register (I2C\_IMR)](https://translate.googleusercontent.com/translate_f#425) ... 403

[18.7.11 I2C RAW Interrupt Register (I2C\_RAWISR)](https://translate.googleusercontent.com/translate_f#426) ..............404

[18.7.12 I2C Receive Threshold (I2C\_RXTLR)](https://translate.googleusercontent.com/translate_f#427) .............406

[18.7.13 I2C transmission threshold (I2C\_TXTLR)](https://translate.googleusercontent.com/translate_f#428) ..............406

10

|  |
| --- |
| **Page 12** |

[18.7.14 I2C Combined and Independent Interrupt Clear Register (I2C\_ICR)](https://translate.googleusercontent.com/translate_f#428) ......... 406

[18.7.15 I2C clears the RX\_UNDER interrupt register (I2C\_RX\_UNDER)](https://translate.googleusercontent.com/translate_f#428) ......... 407

[18.7.16 I2C clears the RX\_OVER interrupt register (I2C\_RX\_OVER)](https://translate.googleusercontent.com/translate_f#429) ........ 407

[18.7.17 I2C clear TX\_OVER interrupt register (I2C\_TX\_OVER)](https://translate.googleusercontent.com/translate_f#429) ........ 407

[18.7.18 I2C clears the RD\_REQ interrupt register (I2C\_RD\_REQ)](https://translate.googleusercontent.com/translate_f#429) .......... 407

[18.7.19 I2C clear TX\_ABRT interrupt register (I2C\_TX\_ABRT)](https://translate.googleusercontent.com/translate_f#430) ............408

[18.7.20 I2C clears RX\_DONE interrupt register (I2C\_RX\_DONE)](https://translate.googleusercontent.com/translate_f#430) .......... 408

[18.7.21 I2C Clear ACTIVITY Interrupt Register (I2C\_ACTIV)](https://translate.googleusercontent.com/translate_f#430) ... ...... 408

[18.7.22 I2C clear STOP\_DET interrupt register (I2C\_STOP)](https://translate.googleusercontent.com/translate_f#431) ...............409

[18.7.23 I2C clear START\_DET interrupt register (I2C\_START)](https://translate.googleusercontent.com/translate_f#431) ...............409

[18.7.24 I2C clears the GEN\_CALL interrupt register (I2C\_GC)](https://translate.googleusercontent.com/translate_f#431) .......... 410

[18.7.25 I2C enable register (I2C\_ENR)](https://translate.googleusercontent.com/translate_f#432) ..................410

[18.7.26 I2C Status Register (I2C\_SR)](https://translate.googleusercontent.com/translate_f#432) ... 410

[18.7.27 I2C Transmit Buffer Level Register (I2C\_TXFLR)](https://translate.googleusercontent.com/translate_f#433) ...........411

[18.7.28 I2C Receive Buffer Level Register (I2C\_RXFLR)](https://translate.googleusercontent.com/translate_f#433) .......... 411

[18.7.29 I2C SDA Hold Time Register (I2C\_HOLD)](https://translate.googleusercontent.com/translate_f#434) ... 412

[18.7.30 I2C DMA Control Register (I2C\_DMA)](https://translate.googleusercontent.com/translate_f#434) .............412

[18.7.31 I2C SDA Setup Time Register (I2C\_SETUP)](https://translate.googleusercontent.com/translate_f#435) ... 413

[18.7.32 I2C ACK Call Register (I2C\_GCR)](https://translate.googleusercontent.com/translate_f#435) ..............................413

[**19** Universal Asynchronous Transceiver **(UART)**](https://translate.googleusercontent.com/translate_f#436)**……..414**

[19.1 Introduction to UART](https://translate.googleusercontent.com/translate_f#436) .................... 414

[19.2 Main Features of UART](https://translate.googleusercontent.com/translate_f#436) ...................414

[19.3 UART function overview](https://translate.googleusercontent.com/translate_f#436) ...............414

[19.3.1 UART Characterization](https://translate.googleusercontent.com/translate_f#437) ................

[19.3.2 Transmitter](https://translate.googleusercontent.com/translate_f#438) ..................416

[19.3.3 Receiver](https://translate.googleusercontent.com/translate_f#439) ...................417

[19.3.4 9-bit data communication](https://translate.googleusercontent.com/translate_f#440) ..................418

[19.3.5 Multi-processor communication](https://translate.googleusercontent.com/translate_f#440) ....................419

[19.3.6 Single-line half-duplex communication](https://translate.googleusercontent.com/translate_f#441) ..................419

[19.3.7 Smart Card](https://translate.googleusercontent.com/translate_f#442) ...................420

[19.3.8 Fractional Baud Rate Generator](https://translate.googleusercontent.com/translate_f#443) ..................421

[19.3.9 Sampling](https://translate.googleusercontent.com/translate_f#443) ..........................422

[19.3.10 Verification Control](https://translate.googleusercontent.com/translate_f#444) .......................422

[19.3.11 Hardware Flow Control](https://translate.googleusercontent.com/translate_f#444) .................422

[19.3.12 Communication using DMA](https://translate.googleusercontent.com/translate_f#446) ...................424

[19.4 UART Interrupt Request](https://translate.googleusercontent.com/translate_f#446) ..................424

[19.5 UART registers is described](https://translate.googleusercontent.com/translate_f#447) .................................................... 425

[19.5.1 UART transmit data register (UART\_TDR)](https://translate.googleusercontent.com/translate_f#447) ..............................425

[19.5.2 UART Receive Data Register (UART\_RDR)](https://translate.googleusercontent.com/translate_f#447) ............426

[Current 19.5.3 UART Status Register (UART\_CSR)](https://translate.googleusercontent.com/translate_f#448) ....................................426

[19.5.4 UART Interrupt Status Register (UART\_ISR)](https://translate.googleusercontent.com/translate_f#449) .............427

[19.5.5 UART interrupt enable register (UART\_IER)](https://translate.googleusercontent.com/translate_f#450) .............428

[19.5.6 UART Interrupt Clear Register (UART\_ICR)](https://translate.googleusercontent.com/translate_f#451) ...............429

11

|  |
| --- |
| **Page 13** |

[19.5.7 UART Global Control Register (UART\_GCR)](https://translate.googleusercontent.com/translate_f#452) .............430

[19.5.8 UART Common Control Register (UART\_CCR)](https://translate.googleusercontent.com/translate_f#453) .............431

[19.5.9 UART Baud Rate Register (UART\_BRR)](https://translate.googleusercontent.com/translate_f#454) ...............432

[19.5.10 UART Fractional Baud Rate Register (UART\_FRA)](https://translate.googleusercontent.com/translate_f#455) ............433

[19.5.11 UART Receive Address Register (UART\_RXADDR)](https://translate.googleusercontent.com/translate_f#455) ............433

[19.5.12 UART Receive Mask Register (UART\_RXMASK)](https://translate.googleusercontent.com/translate_f#456) ............434

[19.5.13 UART SCR Register (UART\_SCR)](https://translate.googleusercontent.com/translate_f#456) ..............434

[**20** hardware division **(HWDIV)**](https://translate.googleusercontent.com/translate_f#458)**………….436**

[20.1 Introduction to Hardware Division](https://translate.googleusercontent.com/translate_f#458) .........................436

[20.2 Main Features of Hardware Division](https://translate.googleusercontent.com/translate_f#458) .................436

[20.3 Introduction to the hardware division function](https://translate.googleusercontent.com/translate_f#458) .................436

[20.4 Hardware Division Register](https://translate.googleusercontent.com/translate_f#458) ............436

[20.4.1 Dividend register (HWDIV\_DVDR)](https://translate.googleusercontent.com/translate_f#458) ................437

[20.4.2 Divisor register (HWDIV\_DVSR)](https://translate.googleusercontent.com/translate_f#459) .................437

[20.4.3 Quotient Register (HWDIV\_QUOTR)](https://translate.googleusercontent.com/translate_f#459) .................437

[I 20.4.4 Number of registers (HWDIV\_RMDR)](https://translate.googleusercontent.com/translate_f#460) .............................................438

[20.4.5 HWDIV Status Register (HWDIV\_SR)](https://translate.googleusercontent.com/translate_f#460) ................438

[20.4.6 HWDIV control register (HWDIV\_CR)](https://translate.googleusercontent.com/translate_f#460) .......................................438

[**21** System Configuration Controller **(SYSCFG)**](https://translate.googleusercontent.com/translate_f#462)**……………………440**

[21.1 SYSCFG register descriptions](https://translate.googleusercontent.com/translate_f#462) .........................................440

[21.1.1 SYSCFG Configuration Register (SYSCFG\_CFGR)](https://translate.googleusercontent.com/translate_f#462) ... 440

[21.1.2 External Interrupt Configuration Register 1 (SYSCFG\_EXTICR1)](https://translate.googleusercontent.com/translate_f#463) ... 441

[21.1.3 External Interrupt Configuration Register 2 (SYSCFG\_EXTICR2)](https://translate.googleusercontent.com/translate_f#464) ... 442

[21.1.4 External Interrupt Configuration Register 3 (SYSCFG\_EXTICR3)](https://translate.googleusercontent.com/translate_f#464) ... 442

[21.1.5 External Interrupt Configuration Register 4 (SYSCFG\_EXTICR4)](https://translate.googleusercontent.com/translate_f#465) ............443

[**22** Device Electronic Signature **(Device)**](https://translate.googleusercontent.com/translate_f#466)**…………………444**

[22.1 memory capacity register](https://translate.googleusercontent.com/translate_f#466) .........................................444

[22.1.1 product unique identity register (96)](https://translate.googleusercontent.com/translate_f#466) ..............................444

[22.2 CRS Register Description](https://translate.googleusercontent.com/translate_f#466) .............444

[22.2.1 unique identification code (UID1)](https://translate.googleusercontent.com/translate_f#466) ..................................444

[22.2.2 unique identification code (UID2)](https://translate.googleusercontent.com/translate_f#467) .......................................445

[22.2.3 unique identification code (UID3)](https://translate.googleusercontent.com/translate_f#467) ........................................445

[22.2.4 unique identification code (UID4)](https://translate.googleusercontent.com/translate_f#467) ..................................445

[**23** Debugging support **(DBG)**](https://translate.googleusercontent.com/translate_f#469)**……………….447**

[23.1 Overview](https://translate.googleusercontent.com/translate_f#469) .....................447

[23.2 Pin Assignment and Debug Port Pins](https://translate.googleusercontent.com/translate_f#470) ..................448

[23.2.1 SWD Debug Port Pins](https://translate.googleusercontent.com/translate_f#470) ................

[23.2.2 Internal pull-up and pull-down on SWD feet](https://translate.googleusercontent.com/translate_f#470) ........... 448

[23.3 ID Code and Locking Mechanism](https://translate.googleusercontent.com/translate_f#470) .................. 448

[23.3.1 microcontroller device ID code](https://translate.googleusercontent.com/translate_f#470) ........................................448

[23.3.2 Cortex JEDEC-106 ID codes...](https://translate.googleusercontent.com/translate_f#471) ..............................449

[23.4 SW debug port](https://translate.googleusercontent.com/translate_f#471) .................. 449

12

|  |
| --- |
| **Page 14** |

[23.4.1 Introduction to the SW Agreement](https://translate.googleusercontent.com/translate_f#471) .................

[23.4.2 SW protocol sequence](https://translate.googleusercontent.com/translate_f#471) .......................................449

[23.4.3 SW-DP state machine (Reset, idle states, ID code)](https://translate.googleusercontent.com/translate_f#472) ............ 450

[23.4.4 DP and AP read/write access](https://translate.googleusercontent.com/translate_f#472) .................451

[23.4.5 SW-DP Register](https://translate.googleusercontent.com/translate_f#473) ............451

[23.4.6 SW-AP register](https://translate.googleusercontent.com/translate_f#473) ..............451

[23.5 MCU debug module (MCUDBG)](https://translate.googleusercontent.com/translate_f#474) ..............................452

[23.5.1 debugging support low-power mode](https://translate.googleusercontent.com/translate_f#474) ....................................452

[23.5.2 supports timers, watchdog](https://translate.googleusercontent.com/translate_f#474) .........................................452

[23.5.3 debugging MCU configuration register](https://translate.googleusercontent.com/translate_f#474) ...................................452

[23.6 DBG register description](https://translate.googleusercontent.com/translate_f#474) ..............452

[23.6.1 DBG control register (DBG\_CR)](https://translate.googleusercontent.com/translate_f#474) ...................................453

[**24** modify records](https://translate.googleusercontent.com/translate_f#476)……………..**454**

13

|  |
| --- |
| **Page 15** |

illustration

[1](https://translate.googleusercontent.com/translate_f#23)

[System Architecture](https://translate.googleusercontent.com/translate_f#23) ..............................1[2](https://translate.googleusercontent.com/translate_f#31)

[Programming flow](https://translate.googleusercontent.com/translate_f#31) ...........................[3](https://translate.googleusercontent.com/translate_f#32)

[Register Flash page erase process](https://translate.googleusercontent.com/translate_f#32) ........................................[4](https://translate.googleusercontent.com/translate_f#33)

[Flash register full chip erase flow](https://translate.googleusercontent.com/translate_f#33) ...............................................[5](https://translate.googleusercontent.com/translate_f#34)

[Option byte programming flow](https://translate.googleusercontent.com/translate_f#34) .................... 12

[6](https://translate.googleusercontent.com/translate_f#35)

[Option byte erasure process](https://translate.googleusercontent.com/translate_f#35) .................... 13

[7](https://translate.googleusercontent.com/translate_f#45)

[A block diagram of a CRC calculation unit](https://translate.googleusercontent.com/translate_f#45) .................................................... .. 23

[8](https://translate.googleusercontent.com/translate_f#48)

[Power supply block diagram](https://translate.googleusercontent.com/translate_f#48) ........................ 26

[9](https://translate.googleusercontent.com/translate_f#49)

[Waveforms of power-on reset and power-down reset](https://translate.googleusercontent.com/translate_f#49) ...............[10](https://translate.googleusercontent.com/translate_f#50)

[PVD Thresholds](https://translate.googleusercontent.com/translate_f#50) ......................... 28

[11](https://translate.googleusercontent.com/translate_f#58)

[Reset Circuits](https://translate.googleusercontent.com/translate_f#58) ..................... 36

[12](https://translate.googleusercontent.com/translate_f#59)

[Clock Tree](https://translate.googleusercontent.com/translate_f#59) ......................... 37

[13](https://translate.googleusercontent.com/translate_f#60)

[Clock source](https://translate.googleusercontent.com/translate_f#60) ...................... 38

[14](https://translate.googleusercontent.com/translate_f#78)

[The basic structure of the I/O port location](https://translate.googleusercontent.com/translate_f#78) .................... .56

[15](https://translate.googleusercontent.com/translate_f#80)

[Enter floating/pull-up/pull-down configuration](https://translate.googleusercontent.com/translate_f#80) ................. .58

[16](https://translate.googleusercontent.com/translate_f#81)

[Output configuration](https://translate.googleusercontent.com/translate_f#81) ................... 59

[17](https://translate.googleusercontent.com/translate_f#82)

[Reuse function configuration](https://translate.googleusercontent.com/translate_f#82) .................... .. 60

[18](https://translate.googleusercontent.com/translate_f#83)

[High-impedance analog inputs configuration](https://translate.googleusercontent.com/translate_f#83) .........................................61

[19](https://translate.googleusercontent.com/translate_f#94)

[External Interrupt Controller block diagram / events](https://translate.googleusercontent.com/translate_f#94) ..........................................72

[20](https://translate.googleusercontent.com/translate_f#96)

[External Interrupt Universal I/O Image](https://translate.googleusercontent.com/translate_f#96) ...............74

[twenty one](https://translate.googleusercontent.com/translate_f#102)

[DMA block diagram](https://translate.googleusercontent.com/translate_f#102) ......................80

[twenty two](https://translate.googleusercontent.com/translate_f#106)

[Peripheral DMA request mapping](https://translate.googleusercontent.com/translate_f#106) ...................... .. 84

[twenty three](https://translate.googleusercontent.com/translate_f#114)

[ADC Block Diagram](https://translate.googleusercontent.com/translate_f#114) ........................92

[twenty four](https://translate.googleusercontent.com/translate_f#115)

[Single Conversion Mode Timing Chart](https://translate.googleusercontent.com/translate_f#115) .................. 93

[25](https://translate.googleusercontent.com/translate_f#116)

[Enabling channel conversion timing diagram (channel direction from low to high) under single cycle scan](https://translate.googleusercontent.com/translate_f#116) .......... 94

[26](https://translate.googleusercontent.com/translate_f#116)

[Enabling channel conversion timing diagram (channel direction from high to low) in single-cycle scanning](https://translate.googleusercontent.com/translate_f#116) .......... 94

[27](https://translate.googleusercontent.com/translate_f#117)

[Continuous scan mode enables channel conversion timing diagram (channel direction from low to high)](https://translate.googleusercontent.com/translate_f#117) ............95

[28](https://translate.googleusercontent.com/translate_f#117)

[Continuous scan mode enables channel conversion timing diagram (channel direction from high to low)](https://translate.googleusercontent.com/translate_f#117) ............95

[29](https://translate.googleusercontent.com/translate_f#118)

[Data Alignment](https://translate.googleusercontent.com/translate_f#118) .................... .96

[30](https://translate.googleusercontent.com/translate_f#130)

[Advanced control block diagram of the timer](https://translate.googleusercontent.com/translate_f#130) ..................................................... . 108

[31](https://translate.googleusercontent.com/translate_f#131)

[When the parameter of the prescaler changes from 1 to 2, the timing diagram of the counter](https://translate.googleusercontent.com/translate_f#131) ........ 109

[32](https://translate.googleusercontent.com/translate_f#132)

[When the parameter of the prescaler changes from 1 to 4, the timing diagram of the counter](https://translate.googleusercontent.com/translate_f#132) .......... 110

[33](https://translate.googleusercontent.com/translate_f#133)

[FIG timing counter, an internal clock divider factor of 1........](https://translate.googleusercontent.com/translate_f#133) ..........................111

[34](https://translate.googleusercontent.com/translate_f#133)

[FIG timing counter, an internal clock divider factor of 2........](https://translate.googleusercontent.com/translate_f#133) ..........................111

[35](https://translate.googleusercontent.com/translate_f#133)

[Counter timing diagram with internal clock division factor of 4](https://translate.googleusercontent.com/translate_f#133) ................ 111

[36](https://translate.googleusercontent.com/translate_f#134)

[Counter timing diagram with internal clock division factor of N](https://translate.googleusercontent.com/translate_f#134) ..............112

[37](https://translate.googleusercontent.com/translate_f#134)

[Counter timing diagram, update event when ARPE = 0 (TIMx\_ARR is not pre-loaded)](https://translate.googleusercontent.com/translate_f#134) ........ 112

[38](https://translate.googleusercontent.com/translate_f#135)

[Counter timing diagram, update event when ARPE = 1 (TIMx\_ARR pre-loaded)](https://translate.googleusercontent.com/translate_f#135) ......... 113

[39](https://translate.googleusercontent.com/translate_f#136)

[Counter timing diagram, internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#136) ................ 114

[40](https://translate.googleusercontent.com/translate_f#136)

[Counter timing diagram, internal clock division factor is 2](https://translate.googleusercontent.com/translate_f#136) ............... 114

[41](https://translate.googleusercontent.com/translate_f#136)

[Counter timing diagram, internal clock division factor is 4](https://translate.googleusercontent.com/translate_f#136) ............... 114

[42](https://translate.googleusercontent.com/translate_f#137)

[Counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#137) .............. 115

14

|  |
| --- |
| **Page 16** |

[43](https://translate.googleusercontent.com/translate_f#137)

[Counter timing diagram, update event when no repeat counter is used](https://translate.googleusercontent.com/translate_f#137) ........... 115

[44](https://translate.googleusercontent.com/translate_f#138)

[Counter timing diagram, internal clock division factor is 1, TIMx\_ARR = 0x6](https://translate.googleusercontent.com/translate_f#138) ......... 116

[45](https://translate.googleusercontent.com/translate_f#139)

[Counter timing diagram, internal clock division factor is 2](https://translate.googleusercontent.com/translate_f#139) ...............117

[46](https://translate.googleusercontent.com/translate_f#139)

[Counter timing diagram, internal clock division factor is 4, TIMx\_ARR = 0x36](https://translate.googleusercontent.com/translate_f#139) ........ 117

[47](https://translate.googleusercontent.com/translate_f#140)

[Counter timing diagram, the internal clock frequency dividing factor N](https://translate.googleusercontent.com/translate_f#140) .................................................1 18

[48](https://translate.googleusercontent.com/translate_f#140)

[Counter timing diagram, update event when ARPE = 1 (counter underflow)](https://translate.googleusercontent.com/translate_f#140) ........... 118

[49](https://translate.googleusercontent.com/translate_f#141)

[Counter timing diagram, update event when ARPE = 1 (counter overflow)](https://translate.googleusercontent.com/translate_f#141) .........119

[50](https://translate.googleusercontent.com/translate_f#142)

[Examples of update rates in different modes and TIMx\_RCR register settings](https://translate.googleusercontent.com/translate_f#142) ............ 120

[51](https://translate.googleusercontent.com/translate_f#143)

[For the control circuit in general mode, the internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#143) ............ 121

[52](https://translate.googleusercontent.com/translate_f#143)

[Examples TI2 external clock connection](https://translate.googleusercontent.com/translate_f#143) ........................................121

[53](https://translate.googleusercontent.com/translate_f#144)

[Control circuit in external clock mode 1](https://translate.googleusercontent.com/translate_f#144) ............122

[54](https://translate.googleusercontent.com/translate_f#144)

[External trigger input block diagram](https://translate.googleusercontent.com/translate_f#144) ...............122

[55](https://translate.googleusercontent.com/translate_f#145)

[Control Circuit in External Clock Mode 2](https://translate.googleusercontent.com/translate_f#145) ...............

[56](https://translate.googleusercontent.com/translate_f#146)

[Capture / compare channels (eg: Channel 1 input section)......](https://translate.googleusercontent.com/translate_f#146) 124............................

[57](https://translate.googleusercontent.com/translate_f#146)

[The main circuit of capture/compare channel 1](https://translate.googleusercontent.com/translate_f#146) .................

[58](https://translate.googleusercontent.com/translate_f#147)

[The output portion of the capture / compare channels (channels 1-3)](https://translate.googleusercontent.com/translate_f#147) ..............................125

[59](https://translate.googleusercontent.com/translate_f#147)

[The output portion of the capture / compare channels (channel 4)](https://translate.googleusercontent.com/translate_f#147) ....................................125

[60](https://translate.googleusercontent.com/translate_f#149)

[PWM input mode timing](https://translate.googleusercontent.com/translate_f#149) .........................................127

[61](https://translate.googleusercontent.com/translate_f#150)

[Compare output mode, flip OCl](https://translate.googleusercontent.com/translate_f#150) ..........................................128

[62](https://translate.googleusercontent.com/translate_f#151)

[Edge-aligned PWM waveform (ARR = 8)](https://translate.googleusercontent.com/translate_f#151) .....................129

[63](https://translate.googleusercontent.com/translate_f#152)

[Center-aligned PWM waveform (APR = 8)](https://translate.googleusercontent.com/translate_f#152) ...................130

[64](https://translate.googleusercontent.com/translate_f#153)

[Complementary output with dead-zone insertion](https://translate.googleusercontent.com/translate_f#153) ...............

[65](https://translate.googleusercontent.com/translate_f#154)

[The dead zone waveform delay is greater than the negative pulse](https://translate.googleusercontent.com/translate_f#154) ................. .. 132

[66](https://translate.googleusercontent.com/translate_f#154)

[The dead zone waveform delay is greater than the positive pulse](https://translate.googleusercontent.com/translate_f#154) ................ .. 132

[67](https://translate.googleusercontent.com/translate_f#156)

[Responding to the output of the brakes](https://translate.googleusercontent.com/translate_f#156) ...................134

[68](https://translate.googleusercontent.com/translate_f#157)

[Clear TIMx of OCxREF](https://translate.googleusercontent.com/translate_f#157) .......................................135

[69](https://translate.googleusercontent.com/translate_f#158)

[Generating six-step PWM, examples (OSSR = 1) the use of COM......](https://translate.googleusercontent.com/translate_f#158) ...........................136

[70](https://translate.googleusercontent.com/translate_f#159)

[Example of single pulse mode](https://translate.googleusercontent.com/translate_f#159) ...............137

[71](https://translate.googleusercontent.com/translate_f#161)

[Example of counter operation in encoder mode](https://translate.googleusercontent.com/translate_f#161) ..................

[72](https://translate.googleusercontent.com/translate_f#161)

[Examples of encoder interface mode IC1FP1 inverted](https://translate.googleusercontent.com/translate_f#161) ....................................139

[73](https://translate.googleusercontent.com/translate_f#163)

[Example of Hall sensor interface](https://translate.googleusercontent.com/translate_f#163) .................

[74](https://translate.googleusercontent.com/translate_f#164)

[Control circuit in reset mode](https://translate.googleusercontent.com/translate_f#164) ...............

[75](https://translate.googleusercontent.com/translate_f#164)

[Control circuit in gate control mode](https://translate.googleusercontent.com/translate_f#164) ............... 142

[76](https://translate.googleusercontent.com/translate_f#165)

[Control Circuit in Trigger Mode](https://translate.googleusercontent.com/translate_f#165) ...............

[77](https://translate.googleusercontent.com/translate_f#166)

[Control circuit in external clock mode 2 + trigger mode](https://translate.googleusercontent.com/translate_f#166) .........144

[78](https://translate.googleusercontent.com/translate_f#199)

[General Timer Block Diagram](https://translate.googleusercontent.com/translate_f#199) ...............177

[79](https://translate.googleusercontent.com/translate_f#200)

[When the parameter of the prescaler changes from 1 to 2, the timing diagram of the counter](https://translate.googleusercontent.com/translate_f#200) ......... 178

[80](https://translate.googleusercontent.com/translate_f#201)

[When the parameter of the prescaler changes from 1 to 4, the timing diagram of the counter](https://translate.googleusercontent.com/translate_f#201) ......... 179

[81](https://translate.googleusercontent.com/translate_f#202)

[FIG timing counter, an internal clock divider factor of 1........](https://translate.googleusercontent.com/translate_f#202) .......................... 180

[82](https://translate.googleusercontent.com/translate_f#202)

[FIG timing counter, an internal clock divider factor of 2........](https://translate.googleusercontent.com/translate_f#202) ..........................180

[83](https://translate.googleusercontent.com/translate_f#202)

[Counter timing diagram, internal clock division factor is 4](https://translate.googleusercontent.com/translate_f#202) ................ 180

[84](https://translate.googleusercontent.com/translate_f#203)

[Counter timing diagram, the internal clock frequency dividing factor N](https://translate.googleusercontent.com/translate_f#203) .....................................181

[85](https://translate.googleusercontent.com/translate_f#203)

[Counter timing diagram, update event when ARPE = 0 (TIMx\_ARR is not preloaded)](https://translate.googleusercontent.com/translate_f#203) ......... 181

[86](https://translate.googleusercontent.com/translate_f#204)

[Counter timing diagram, update event when ARPE = 1 (TIMx\_ARR pre-loaded)](https://translate.googleusercontent.com/translate_f#204) .......... 182

15

|  |
| --- |
| **Page 17** |

[87](https://translate.googleusercontent.com/translate_f#205)

[Counter timing diagram, internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#205) .................183

[88](https://translate.googleusercontent.com/translate_f#205)

[Counter timing diagram, internal clock division factor is 2](https://translate.googleusercontent.com/translate_f#205) .................183

[89](https://translate.googleusercontent.com/translate_f#205)

[Counter timing diagram, internal clock division factor is 4](https://translate.googleusercontent.com/translate_f#205) ................183

[90](https://translate.googleusercontent.com/translate_f#206)

[Counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#206) .................184

[91](https://translate.googleusercontent.com/translate_f#206)

[Counter timing diagram, update event when no repetition counter is used](https://translate.googleusercontent.com/translate_f#206) ............ 184

[92](https://translate.googleusercontent.com/translate_f#207)

[Counter timing diagram, internal clock division factor is 1, TIMx\_ARR = 0x6](https://translate.googleusercontent.com/translate_f#207) .........185

[93](https://translate.googleusercontent.com/translate_f#207)

[FIG timing counter, an internal clock divider factor of 2........](https://translate.googleusercontent.com/translate_f#207) ..........................185

[94](https://translate.googleusercontent.com/translate_f#208)

[Counter timing diagram, internal clock division factor is 4, TIMx\_ARR = 0x36](https://translate.googleusercontent.com/translate_f#208) ........ 186

[95](https://translate.googleusercontent.com/translate_f#208)

[Counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#208) ..............186

[96](https://translate.googleusercontent.com/translate_f#209)

[Counter timing diagram, update event when ARPE = 1 (counter underflow)](https://translate.googleusercontent.com/translate_f#209) .......... 187

[97](https://translate.googleusercontent.com/translate_f#209)

[Counter timing diagram, update event when ARPE = 1 (counter overflow)](https://translate.googleusercontent.com/translate_f#209) .......... 187

[98](https://translate.googleusercontent.com/translate_f#210)

[For the control circuit in general mode, the internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#210) ............ 188

[99](https://translate.googleusercontent.com/translate_f#210)

[Examples TI2 external clock connection](https://translate.googleusercontent.com/translate_f#210) .............................................188

[100 Control circuit in external clock mode 1](https://translate.googleusercontent.com/translate_f#211) ...............

[101 External Trigger Input Block Diagram](https://translate.googleusercontent.com/translate_f#212) .............190

[102 Control circuit in external clock mode 2](https://translate.googleusercontent.com/translate_f#212) .........190

[103 capture / compare channels (eg: Channel 1 input section)](https://translate.googleusercontent.com/translate_f#213) ....................................191

[104 Capture/Compare Channel 1 Main Circuit](https://translate.googleusercontent.com/translate_f#214) .................192

[Output portion 105 capture / compare channels (channel 1)](https://translate.googleusercontent.com/translate_f#214) ....................................192

[Output portion 106 capture / compare channels (channel 1)](https://translate.googleusercontent.com/translate_f#216) ....................................194

[107 output compare mode, flip OCl](https://translate.googleusercontent.com/translate_f#217) ..........................................195

[Edge-aligned PWM waveforms 108 (= the ARR. 8)](https://translate.googleusercontent.com/translate_f#219) .............................................197

[109 Center-aligned PWM waveform (APR = 8)](https://translate.googleusercontent.com/translate_f#220) ..................... 198

[110 Example of Single Pulse Mode](https://translate.googleusercontent.com/translate_f#221) ................... 199

[Clear TIMx 111 of the OCxREF](https://translate.googleusercontent.com/translate_f#222) ..................................200 is

[112 Counter Operation Example in Encoder Mode](https://translate.googleusercontent.com/translate_f#224) .................

[113 IC1FP1 inverter encoder interface mode example](https://translate.googleusercontent.com/translate_f#224) ..............................202

[114 Control circuit in reset mode](https://translate.googleusercontent.com/translate_f#225) ................203

[115 Control circuit in gate control mode](https://translate.googleusercontent.com/translate_f#226) ................. 204

[116 Control Circuit in Trigger Mode](https://translate.googleusercontent.com/translate_f#227) ................205

[117 External Clock Mode 2 + Trigger Mode Control Circuit](https://translate.googleusercontent.com/translate_f#228) ............206

[118 Examples of Master/Slave Timers](https://translate.googleusercontent.com/translate_f#228) .....................206

[119 Timer1 OC1REF Control Timer2](https://translate.googleusercontent.com/translate_f#229) ...............

[120 Timer 2 can be controlled by enabling timer 1](https://translate.googleusercontent.com/translate_f#230) .............208

[121 Update using timer 1 triggers timer 2](https://translate.googleusercontent.com/translate_f#231) ..............209

[122 Use Timer 1 to enable Timer 2](https://translate.googleusercontent.com/translate_f#231) ...............209

[123 Use the TI1 input of timer 1 to trigger timer 1 and timer 2](https://translate.googleusercontent.com/translate_f#232) ......... 210

[124 General Timer Block Diagram](https://translate.googleusercontent.com/translate_f#257) ...................... 235

[125 When the parameter of the prescaler changes from 1 to 2, the timing diagram of the counter](https://translate.googleusercontent.com/translate_f#258) ......... 236

[126 When the prescaler parameter changes from 1 to 4, the timing diagram of the counter](https://translate.googleusercontent.com/translate_f#259) ........... 237

[127 counter timing diagram, internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#260) ..............238

[FIG timing counter 128, an internal clock divider factor of 2........](https://translate.googleusercontent.com/translate_f#260) ..........................238

[FIG timing counter 129, an internal clock division factor of 4........](https://translate.googleusercontent.com/translate_f#260) ..........................238

[130 counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#261) ...............239

16

|  |
| --- |
| **Page 18** |

[131 Counter timing diagram, update event when ARPE = 0 (TIMx\_ARR is not preloaded)](https://translate.googleusercontent.com/translate_f#261) .........239

[132 Counter timing diagram, update event when ARPE = 1 (TIMx\_ARR pre-loaded)](https://translate.googleusercontent.com/translate_f#262) ........... 240

[FIG timing counter 133, an internal clock divider factor of 1........](https://translate.googleusercontent.com/translate_f#263) ..........................241

[FIG timing counter 134, an internal clock divider factor of 2........](https://translate.googleusercontent.com/translate_f#263) ..........................241

[FIG timing counter 135, an internal clock division factor of 4........](https://translate.googleusercontent.com/translate_f#263) ..........................241

[136 counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#264) ..............242

[137 Counter timing diagram, update event when no repeat counter is used](https://translate.googleusercontent.com/translate_f#264) .......... 242

[138 Counter timing diagram, internal clock division factor is 1, TIMx\_ARR = 0x6](https://translate.googleusercontent.com/translate_f#265) .........243

[139 counter timing diagram, internal clock division factor is 2](https://translate.googleusercontent.com/translate_f#265) ..............243

[140 counter timing diagram, internal clock division factor is 4, TIMx\_ARR = 0x36](https://translate.googleusercontent.com/translate_f#266) .......... 244

[141 Counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#266) ..............244

[142 Counter timing diagram, update event when ARPE = 1 (counter underflow)](https://translate.googleusercontent.com/translate_f#267) .......... 245

[143 Counter timing diagram, update event when ARPE = 1 (counter overflow)](https://translate.googleusercontent.com/translate_f#267) .......... 245

[144 Control circuit in general mode, internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#268) ..... 246

[145 TI2 external clock connection example](https://translate.googleusercontent.com/translate_f#268) ........................................246

[146 Control circuit in external clock mode 1](https://translate.googleusercontent.com/translate_f#269) ....................

[147 External trigger input block diagram](https://translate.googleusercontent.com/translate_f#270) ...................248

[148 Control Circuit in External Clock Mode 2](https://translate.googleusercontent.com/translate_f#270) .............

[149 capture / compare channels (eg: Channel 1 input section)](https://translate.googleusercontent.com/translate_f#271) ....................................249

[150 Capture/Compare Channel 1 Main Circuit](https://translate.googleusercontent.com/translate_f#272) ................. 250

[Output portion 151 capture / compare channels (channel 1)](https://translate.googleusercontent.com/translate_f#272) ..............................250

[Output portion 152 capture / compare channels (channel 1)](https://translate.googleusercontent.com/translate_f#274) ....................................252

[153 output compare mode, flip OCl](https://translate.googleusercontent.com/translate_f#275) ..........................................253

[Edge-aligned PWM waveforms 154 (= the ARR. 8)](https://translate.googleusercontent.com/translate_f#277) .............................................255

[155 center-aligned PWM waveforms (= the APR. 8)](https://translate.googleusercontent.com/translate_f#278) .............................................256

[156 Example of single pulse mode](https://translate.googleusercontent.com/translate_f#279) ................... 257

[157 Clear OCxREF of TIMx](https://translate.googleusercontent.com/translate_f#280) ..............258

[158 Counter Operation Example in Encoder Mode](https://translate.googleusercontent.com/translate_f#282) ...............

[159 IC1FP1 inverter encoder interface mode example](https://translate.googleusercontent.com/translate_f#282) ..............................260

[160 Control circuit in reset mode](https://translate.googleusercontent.com/translate_f#283) .............261

[161 Control circuit in gate control mode](https://translate.googleusercontent.com/translate_f#284) ..................262

[The control circuit 162 in the trigger mode](https://translate.googleusercontent.com/translate_f#285) ...........................................263

[163 Control circuit in external clock mode 2 + trigger mode](https://translate.googleusercontent.com/translate_f#286) .............264

[164 Examples of Master/Slave Timers](https://translate.googleusercontent.com/translate_f#286) ........................264

[OC1REF control timer 165 timer 12](https://translate.googleusercontent.com/translate_f#287) ....................................265

[166 Timer 2 can be controlled by enabling timer 1](https://translate.googleusercontent.com/translate_f#288) ............ 266

[167 using the updated trigger Timer 1 Timer 2](https://translate.googleusercontent.com/translate_f#289) ...........................................267

[168 1 using a timer so that the timer can trigger 2](https://translate.googleusercontent.com/translate_f#289) ..........................................267

[169 Use Timer 1's TI1 input to trigger Timer 1 and Timer 2](https://translate.googleusercontent.com/translate_f#290) ... 268

[170 basic timer block diagram](https://translate.googleusercontent.com/translate_f#316) ...................... .294

[171 When the prescaler parameter changes from 1 to 2, the counter timing diagram](https://translate.googleusercontent.com/translate_f#317) ........... 295

[172 When the parameter of the prescaler changes from 1 to 4, the timing chart of the counter](https://translate.googleusercontent.com/translate_f#317) .......... 295

[FIG timing counter 173, an internal clock divider factor of 1........](https://translate.googleusercontent.com/translate_f#318) ..........................296

[FIG timing counter 174, an internal clock divider factor of 2........](https://translate.googleusercontent.com/translate_f#318) ..........................296

17

|  |
| --- |
| **Page 19** |

[FIG timing counter 175, an internal clock division factor of 4........](https://translate.googleusercontent.com/translate_f#319) ..........................297

[FIG timing counter 176, an internal clock divider factor N](https://translate.googleusercontent.com/translate_f#319) ......................................297

[177 Counter timing diagram, update event when ARPE = 0 (TIM14\_ARR is not preloaded)](https://translate.googleusercontent.com/translate_f#319) .........297

[178 Counter timing diagram, update event when ARPE = 1 (TIM14\_ARR pre-loaded)](https://translate.googleusercontent.com/translate_f#320) ......... 298

[179 Examples of update rates for different modes and different TIMx\_RCR register settings](https://translate.googleusercontent.com/translate_f#321) ......... 299

[The control circuit 180 in the normal mode, the internal clock divider factor of 1](https://translate.googleusercontent.com/translate_f#322) ............................300

[181 capture / compare channels (eg: Channel 1 input section)](https://translate.googleusercontent.com/translate_f#322) ....................................300

[182 Capture/Compare Channel 1 Main Circuit](https://translate.googleusercontent.com/translate_f#323) ............301

[Output portion 183 capture / compare channels (channel 1)](https://translate.googleusercontent.com/translate_f#323) ....................................301

[184 output compare mode, flip OC1](https://translate.googleusercontent.com/translate_f#325) ..........................................303

[Edge-aligned PWM waveforms 185 (= the ARR. 8)](https://translate.googleusercontent.com/translate_f#326) .............................................304

[186 basic timers TIM16 and TIM17 block diagram](https://translate.googleusercontent.com/translate_f#342) ...................... 320

[187 When the parameter of the prescaler changes from 1 to 2, the timing chart of the counter](https://translate.googleusercontent.com/translate_f#343) .......... 321

[188 When the parameter of the prescaler changes from 1 to 4, the timing chart of the counter](https://translate.googleusercontent.com/translate_f#343) .......... 321

[189 counter timing diagram, internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#344) ..............322

[190 counter timing diagram, internal clock division factor is 2](https://translate.googleusercontent.com/translate_f#345) .............323

[191 counter timing diagram, internal clock division factor is 4](https://translate.googleusercontent.com/translate_f#345) .............323

[192 counter timing diagram, internal clock division factor is N](https://translate.googleusercontent.com/translate_f#345) .............323

[193 Counter timing diagram, update event when ARPE = 0 (TIMx\_ARR is not preloaded)](https://translate.googleusercontent.com/translate_f#346) .........324

[194 Counter timing diagram, update event when ARPE = 1 (TIMx\_ARR preload)](https://translate.googleusercontent.com/translate_f#346) .........324

[195 Examples of update rates under different modes and different TIMx\_RCR register settings](https://translate.googleusercontent.com/translate_f#347) .........325

[196 Control circuit in general mode, internal clock division factor is 1](https://translate.googleusercontent.com/translate_f#348) ..... 326

[197 capture / compare channels such as Channel 1 input portion :()](https://translate.googleusercontent.com/translate_f#348) .....................................326

[198 Capture/Compare Channel 1 Main Circuit](https://translate.googleusercontent.com/translate_f#349) .................327

[Output portion 199 capture / compare channels (channel 1)](https://translate.googleusercontent.com/translate_f#349) ....................................327

[200 output compare mode, flip OCl](https://translate.googleusercontent.com/translate_f#352) ..........................................330.

[Edge-aligned PWM waveforms 201 (= the ARR. 8)](https://translate.googleusercontent.com/translate_f#353) .............................................331

[202. Complementary output with dead zone insertion](https://translate.googleusercontent.com/translate_f#354) ..................332

[203 The dead-band waveform delay is greater than the negative pulse](https://translate.googleusercontent.com/translate_f#354) ............ .. 332

[204 The dead zone waveform delay is greater than the positive pulse](https://translate.googleusercontent.com/translate_f#354) ............. 332

[205 Respond to the output of the brakes](https://translate.googleusercontent.com/translate_f#356) ................... 334

[206 Monopulse mode example](https://translate.googleusercontent.com/translate_f#357) .................... [.335](https://translate.googleusercontent.com/translate_f#357)

[207 Independent Watchdog Block Diagram](https://translate.googleusercontent.com/translate_f#378) .....................356

[208 Watchdog Block Diagram](https://translate.googleusercontent.com/translate_f#384) ..........................362

[209 Window Watchdog Timing Chart](https://translate.googleusercontent.com/translate_f#385) ........................363

[210 SPI block diagram](https://translate.googleusercontent.com/translate_f#389) ...........................367

[211 Single Master and Single Slave Application](https://translate.googleusercontent.com/translate_f#390) ................. .. 368

[212 Data Clock Timing Diagram](https://translate.googleusercontent.com/translate_f#392) .................. 370

[213 Start and stop conditions](https://translate.googleusercontent.com/translate_f#408) .................. .386

[214 7-bit address format](https://translate.googleusercontent.com/translate_f#408) ........................386

[215 10-bit address format](https://translate.googleusercontent.com/translate_f#409) .................. 387

[216 Master Sending Agreement](https://translate.googleusercontent.com/translate_f#410) .................. 388

[217 Master Reception Agreement](https://translate.googleusercontent.com/translate_f#410) .................. 388

[218 Start byte transfer](https://translate.googleusercontent.com/translate_f#411) .......................389

18

|  |
| --- |
| **Page 20** |

[219 DR Registers](https://translate.googleusercontent.com/translate_f#412) .................. 390

[220 Main Send-Tx FIFO is empty](https://translate.googleusercontent.com/translate_f#412) ................... 390

[221 Main Receive-Tx FIFO is empty](https://translate.googleusercontent.com/translate_f#412) .................... .390

[222 Multi-host arbitration](https://translate.googleusercontent.com/translate_f#413) .................. 391

[223 Synchronization of multiple host clocks](https://translate.googleusercontent.com/translate_f#413) .................. 391

[224 I2C functional block diagram](https://translate.googleusercontent.com/translate_f#414) ..................... 392

[225 I2C Interface Host Flowchart](https://translate.googleusercontent.com/translate_f#417) ....................395

[226 Interruption mechanism](https://translate.googleusercontent.com/translate_f#419) ................... 397

[227 UART block diagram](https://translate.googleusercontent.com/translate_f#437) ....................415

[228 UART Timing](https://translate.googleusercontent.com/translate_f#438) .................... 416

[229 The status bit changes when sending](https://translate.googleusercontent.com/translate_f#439) .................. .417

[230 UART block diagram](https://translate.googleusercontent.com/translate_f#442) ....................420

[231 UART block diagram](https://translate.googleusercontent.com/translate_f#443) ...................421

[232 RX Pin Sampling Scheme](https://translate.googleusercontent.com/translate_f#444) ..................422

[233 Hardware flow control between two UARTs](https://translate.googleusercontent.com/translate_f#445) ...........423

[234 RTS flow control](https://translate.googleusercontent.com/translate_f#445) .........................423

[235 CTS flow control](https://translate.googleusercontent.com/translate_f#446) ................... 424

[236 MM32 series level and CPU level debug box](https://translate.googleusercontent.com/translate_f#469) ..............447

19

|  |
| --- |
| **Page 21** |

form

[1](https://translate.googleusercontent.com/translate_f#24)

[Memory image](https://translate.googleusercontent.com/translate_f#24) ......................2

[2](https://translate.googleusercontent.com/translate_f#26)

[Startup mode](https://translate.googleusercontent.com/translate_f#26) ................... ..5

[3](https://translate.googleusercontent.com/translate_f#28)

[Flash module structure](https://translate.googleusercontent.com/translate_f#28) .....................6

[4](https://translate.googleusercontent.com/translate_f#36)

[Flash interrupt request](https://translate.googleusercontent.com/translate_f#36) ...................... 14

[5](https://translate.googleusercontent.com/translate_f#36)

[Option byte format](https://translate.googleusercontent.com/translate_f#36) ................... .. 14

[6](https://translate.googleusercontent.com/translate_f#37)

[Option byte structure](https://translate.googleusercontent.com/translate_f#37) .................... .. 15

[7](https://translate.googleusercontent.com/translate_f#37)

[Option byte descriptions](https://translate.googleusercontent.com/translate_f#37) .................... .. 15

[8](https://translate.googleusercontent.com/translate_f#38)

[Overview FLASH memory location](https://translate.googleusercontent.com/translate_f#38) .............................................16

[9](https://translate.googleusercontent.com/translate_f#46)

[Overview of CRC Registers](https://translate.googleusercontent.com/translate_f#46) .........................

[10](https://translate.googleusercontent.com/translate_f#51)

[List of low power consumption modes](https://translate.googleusercontent.com/translate_f#51) ......................29

[11](https://translate.googleusercontent.com/translate_f#51)

[SLEEP NOW pattern](https://translate.googleusercontent.com/translate_f#51) .......................................30

[12](https://translate.googleusercontent.com/translate_f#52)

[SLEEP ON EXIT mode](https://translate.googleusercontent.com/translate_f#52) .......................................30

[13](https://translate.googleusercontent.com/translate_f#53)

[Shutdown mode](https://translate.googleusercontent.com/translate_f#53) .....................31

[14](https://translate.googleusercontent.com/translate_f#53)

[Standby mode](https://translate.googleusercontent.com/translate_f#53) .....................31

[15](https://translate.googleusercontent.com/translate_f#54)

[Overview of Power Control Registers](https://translate.googleusercontent.com/translate_f#54) ...................

[16](https://translate.googleusercontent.com/translate_f#62)

[Overview RCC register](https://translate.googleusercontent.com/translate_f#62) .............................................40

[17](https://translate.googleusercontent.com/translate_f#78)

[Port Configuration Table](https://translate.googleusercontent.com/translate_f#78) .................... 56

[18](https://translate.googleusercontent.com/translate_f#78)

[Output Mode Bits](https://translate.googleusercontent.com/translate_f#78) .................... 56

[19](https://translate.googleusercontent.com/translate_f#83)

[Advanced Timer TIM1](https://translate.googleusercontent.com/translate_f#83) .................... 61

[20](https://translate.googleusercontent.com/translate_f#83)

[General Timer TIM2/3/14/16/17](https://translate.googleusercontent.com/translate_f#83) ..........................61

[twenty one](https://translate.googleusercontent.com/translate_f#83)

[UART](https://translate.googleusercontent.com/translate_f#83) ....................... 62

[twenty two](https://translate.googleusercontent.com/translate_f#84)

[SPI](https://translate.googleusercontent.com/translate_f#84) ...........................62

[twenty three](https://translate.googleusercontent.com/translate_f#84)

[I2C](https://translate.googleusercontent.com/translate_f#84) ...........................62

[twenty four](https://translate.googleusercontent.com/translate_f#84)

[ADC](https://translate.googleusercontent.com/translate_f#84) .............................62

[25](https://translate.googleusercontent.com/translate_f#84)

[Other I/O pins](https://translate.googleusercontent.com/translate_f#84) .....................62

[26](https://translate.googleusercontent.com/translate_f#85)

[Debugging interface signal](https://translate.googleusercontent.com/translate_f#85) .....................63

[27](https://translate.googleusercontent.com/translate_f#85)

[Overview GPIO registers](https://translate.googleusercontent.com/translate_f#85) ..............................................63

[28](https://translate.googleusercontent.com/translate_f#92)

[Vector table for this series of products](https://translate.googleusercontent.com/translate_f#92) .................... 70

[29](https://translate.googleusercontent.com/translate_f#96)

[Overview EXTI register](https://translate.googleusercontent.com/translate_f#96) .............................................75

[30](https://translate.googleusercontent.com/translate_f#104)

[Programmable data transfer width and size end operation (when PINC = MINC = 1)](https://translate.googleusercontent.com/translate_f#104) .......... 82

[31](https://translate.googleusercontent.com/translate_f#105)

[DMA interrupt request](https://translate.googleusercontent.com/translate_f#105) ............. 84

[32](https://translate.googleusercontent.com/translate_f#106)

[List of DMA requests for each channel](https://translate.googleusercontent.com/translate_f#106) .................... .. 84

[33](https://translate.googleusercontent.com/translate_f#107)

[Overview DMA registers](https://translate.googleusercontent.com/translate_f#107) ........................................85

[34](https://translate.googleusercontent.com/translate_f#119)

[Overview ADC register](https://translate.googleusercontent.com/translate_f#119) .............................................97

[35](https://translate.googleusercontent.com/translate_f#160)

[Relationship between the encoder count direction signal](https://translate.googleusercontent.com/translate_f#160) .....................................138

[36](https://translate.googleusercontent.com/translate_f#166)

[Overview TIM1 register](https://translate.googleusercontent.com/translate_f#166) ........................................144

[37](https://translate.googleusercontent.com/translate_f#173)

[TIMx internal connection is triggered](https://translate.googleusercontent.com/translate_f#173) ..............................................151

[38](https://translate.googleusercontent.com/translate_f#187)

[Control bits of the complementary output channels OCx and OCxN with brake function](https://translate.googleusercontent.com/translate_f#187) ........... 165

[39](https://translate.googleusercontent.com/translate_f#223)

[Relationship between counting direction and encoder signal](https://translate.googleusercontent.com/translate_f#223) ......................

[40](https://translate.googleusercontent.com/translate_f#232)

[Overview TIMx register....](https://translate.googleusercontent.com/translate_f#232) ....................................211

[41](https://translate.googleusercontent.com/translate_f#239)

[TIMx trigger internal connections](https://translate.googleusercontent.com/translate_f#239) ..............................................217

[42](https://translate.googleusercontent.com/translate_f#250)

[Output channel control bits Ocx standard](https://translate.googleusercontent.com/translate_f#250) .........................................228

20

|  |
| --- |
| **Page 22** |

[43](https://translate.googleusercontent.com/translate_f#281)

[Relationship between the encoder count direction signal](https://translate.googleusercontent.com/translate_f#281) .....................................259

[44](https://translate.googleusercontent.com/translate_f#290)

[Overview TIMx register](https://translate.googleusercontent.com/translate_f#290) ........................................269

[45](https://translate.googleusercontent.com/translate_f#297)

[TIMx trigger internal connections](https://translate.googleusercontent.com/translate_f#297) ..............................................275

[46](https://translate.googleusercontent.com/translate_f#308)

[Output channel control bits Ocx standard](https://translate.googleusercontent.com/translate_f#308) .........................................286

[47](https://translate.googleusercontent.com/translate_f#327)

[Overview TIM14 register](https://translate.googleusercontent.com/translate_f#327) ..............................................305

[48](https://translate.googleusercontent.com/translate_f#335)

[Output channel control bits OCx standard](https://translate.googleusercontent.com/translate_f#335) ....................................314

[49](https://translate.googleusercontent.com/translate_f#358)

[Overview TIM16 / 17 registers](https://translate.googleusercontent.com/translate_f#358) ........................................336

[50](https://translate.googleusercontent.com/translate_f#368)

[Control bits of the complementary output channels OCx and OCxN with brake function](https://translate.googleusercontent.com/translate_f#368) .......... 347

[51](https://translate.googleusercontent.com/translate_f#378)

[Watchdog time-out time (40KHz input clock (LSI))](https://translate.googleusercontent.com/translate_f#378) ............356

[52](https://translate.googleusercontent.com/translate_f#379)

[Overview IWDG register](https://translate.googleusercontent.com/translate_f#379) ..............................................357

[53](https://translate.googleusercontent.com/translate_f#386)

[Overview WWDG register](https://translate.googleusercontent.com/translate_f#386) .........................................364

[54](https://translate.googleusercontent.com/translate_f#394)

[SPI status](https://translate.googleusercontent.com/translate_f#394) ......................372

[55](https://translate.googleusercontent.com/translate_f#395)

[Baud rate formula](https://translate.googleusercontent.com/translate_f#395) ...........................373

[56](https://translate.googleusercontent.com/translate_f#395)

[Overview SPI register](https://translate.googleusercontent.com/translate_f#395) .......................................373

[57](https://translate.googleusercontent.com/translate_f#409)

[The first byte of I2C](https://translate.googleusercontent.com/translate_f#409) ..................... .. 387

[58](https://translate.googleusercontent.com/translate_f#418)

[Interrupt bit setting and clearing](https://translate.googleusercontent.com/translate_f#418) .................... .396

[59](https://translate.googleusercontent.com/translate_f#419)

[Overview of I2C register descriptions](https://translate.googleusercontent.com/translate_f#419) .................

[60](https://translate.googleusercontent.com/translate_f#422)

[DISSLAVE(bit 6) and MASTER(bit 0) configuration](https://translate.googleusercontent.com/translate_f#422) ................. 400

[61](https://translate.googleusercontent.com/translate_f#446)

[UART interrupt request](https://translate.googleusercontent.com/translate_f#446) .......................................424

[62](https://translate.googleusercontent.com/translate_f#447)

[Overview UART registers](https://translate.googleusercontent.com/translate_f#447) ..............................................425

[63](https://translate.googleusercontent.com/translate_f#458)

[Overview of Hardware Division Registers](https://translate.googleusercontent.com/translate_f#458) ...............

[64](https://translate.googleusercontent.com/translate_f#462)

[Overview of the SYSCFG](https://translate.googleusercontent.com/translate_f#462) ........................................440

[65](https://translate.googleusercontent.com/translate_f#466)

[DESCRIPTION Overview The memory capacity register](https://translate.googleusercontent.com/translate_f#466) ..........................................444

[66](https://translate.googleusercontent.com/translate_f#470)

[SWJ Debug Port Pins](https://translate.googleusercontent.com/translate_f#470) ...................448

[68](https://translate.googleusercontent.com/translate_f#471)

[ID coding](https://translate.googleusercontent.com/translate_f#471) ...................... .. 449

[69](https://translate.googleusercontent.com/translate_f#471)

[Request packet (8 bits)](https://translate.googleusercontent.com/translate_f#471) ........................ 450

[70](https://translate.googleusercontent.com/translate_f#472)

[Request packet (3 bits)](https://translate.googleusercontent.com/translate_f#472) ......................... 450

[71](https://translate.googleusercontent.com/translate_f#472)

[Request packet (33 bits)](https://translate.googleusercontent.com/translate_f#472) ........................ 450

[73](https://translate.googleusercontent.com/translate_f#474)

[Overview DBG register](https://translate.googleusercontent.com/translate_f#474) ........................................452

[74](https://translate.googleusercontent.com/translate_f#476)

[Amendment Records](https://translate.googleusercontent.com/translate_f#476) ................... 454

twenty one

|  |
| --- |
| **Page 23** |

UM\_MM32F003\_q\_Ver1.19

1Memory and bus architecture

**1.1**The main system consists of the following parts:

• Two drive units:

**–** CPU core system bus (S-bus)

**–** General DMA

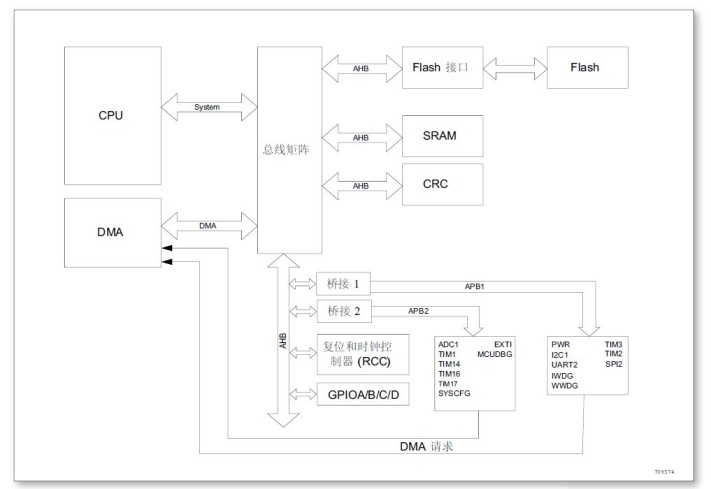
• Three passive units:

**–** Internal SRAM

**–** Internal flash memory

**–** AHB to APB bridge (APBx), which connects all AHB devices

These are connected to each other through a multi-level AHB bus architecture, as shown in Figure 1:



709574

CPU

System

DMA

DMA

ᙫ㓯⸙䱥

Flash ᧕ਓ

Flash

AHB

SRAM

AHB

AH

B

ẕ᧕ 1

ẕ᧕ 2

ἽՍ઼ᰦ䫏᧗

ࡦಘ (RCC)

GPIOA/B/C/D

APB2

APB1

PWR

I2C1

UART2

IWDG

WWDG

TIM3

TIM2

SPI2

ADC1

EXTI

MCUDBG

TIM1

TIM14

TIM16

TIM17

SYSCFG

DMA ∧≲

CRC

AHB

1.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

1/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 24** |

UM\_MM32F003\_q\_Ver1.19

This bus connects the system bus (peripheral bus) of the CPU core to the bus matrix. The bus matrix coordinates the core and the DMA.

Access.

**DMA**

This bus connects the DMA's AHB master interface with the bus matrix. The bus matrix coordinates the CPU and DMA to SRAM,

Access control of flash memory and peripherals.

**(BusMatrix)**

The bus matrix manages the access arbitration of the core system bus and the DMA bus. The bus matrix consists of the master module bus and slave modules

Bus composition.

The AHB peripheral is connected to the system bus through the bus matrix, allowing DMA access.

**AHB**

**APB**

**(AHB2APB bridges-APB)**

The AHB to APB bridge provides a synchronous connection between the AHB and APB buses. After each reset, all peripheral clocks

Both are closed (except SRAM and Flash). Before using a peripheral, you must open the corresponding RCC\_AHBENR,

The clock enable bit in the RCC\_APB2ENR or RCC\_APB1ENR register.

Note: When 8-bit or 16-bit access is made to the APB register, the access will be automatically converted to 32-bit access: the bridge will automatically

Expand 16-bit or 8-bit data to match 32-bit width.

**1.2**

**1.2.1**

Program memory, data memory, registers and I/O ports are addressed uniformly, and their linear address space reaches 4G.

Data bytes are stored in memory in little-endian format, and the lowest address byte in a word is considered the least significant word of the word

Bytes, and the highest address byte is the most significant byte.

The addressing space is divided into 8 blocks of 512MB each. All other memory spaces not allocated to on-chip memory and peripherals are

Is the reserved address space. Please refer to the memory map and register addressing chapters and peripheral chapters for details.

**1.2.2**

For the memory map, please refer to the memory map in the corresponding chapter of each peripheral.

The following table shows the starting addresses of all built-in peripherals.

1.

bus

Addressing range

size

Peripherals

Remarks

Flash

0x0000 0000-0x0000 3FFF

16 KB

Main flash memory, system memory

Or SRAM depends on BOOT configuration

0x0000 4000-0x07FF FFFF

∼ 128 MB

Reserved

0x0800 0000-0x0800 3FFF

16 KB

Main Flash memory

0x0802 0000-0x1FFD FFFF

∼ 256 MB

Reserved

0x1FFE 0000-0x1FFE 01FF

0.5 KB

Reserved

0x1FFE 0200-0x1FFE 0FFF

3 KB

Reserved

0x1FFE 1000-0x1FFE 1BFF

3 KB

Reserved

0x1FFE 1C00-0x1FFF F3FF

∼ 256 MB

Reserved

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

2/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 25** |

UM\_MM32F003\_q\_Ver1.19

bus

Addressing range

size

Peripherals

Remarks

Flash

0x1FFF F400-0x1FFF F7FF

1 KB

System memory

0x1FFF F800-0x1FFF F80F

16 B

Option bytes

0x1FFF F810-0x1FFF FFFF

∼ 2 KB

Reserved

SRAM

0x2000 0000-0x2000 07FF

2 KB

SRAM

0x2000 0800-0x2FFF FFFF

∼ 512 MB

Reserved

APB1

0x4000 0000-0x4000 03FF

1 KB

TIM2

0x4000 0400-0x4000 07FF

1 KB

TIM3

0x4000 0800-0x4000 0BFF

8 KB

Reserved

0x4000 2800-0x4000 2BFF

1 KB

Reserved

0x4000 2C00-0x4000 2FFF

1 KB

WWDG

0x4000 3000-0x4000 33FF

1 KB

IWDG

0x4000 3400-0x4000 37FF

1 KB

Reserved

0x4000 3800-0x4000 3BFF

1 KB

SPI2

0x4000 4000-0x4000 43FF

1 KB

Reserved

0x4000 4400-0x4000 47FF

1 KB

UART2

0x4000 4800-0x4000 4BFF

3 KB

Reserved

0x4000 5400-0x4000 57FF

1 KB

I2C1

0x4000 5800-0x4000 5BFF

1 KB

Reserved

0x4000 5C00-0x4000 5FFF

1 KB

Reserved

0x4000 6000-0x4000 63FF

1 KB

Reserved

0x4000 6400-0x4000 67FF

1 KB

Reserved

0x4000 6800-0x4000 6BFF

1 KB

Reserved

0x4000 6C00-0x4000 6FFF

1 KB

Reserved

0x4000 7000-0x4000 73FF

1 KB

PWR

0x4000 7400-0x4000 FFFF

35 KB

Reserved

APB2

0x4001 0000-0x4001 03FF

1 KB

SYSCFG

0x4001 0400-0x4001 07FF

1 KB

EXTI

0x4001 0800-0x4001 23FF

7 KB

Reserved

0x4001 2400-0x4001 27FF

1 KB

ADC1

0x4001 2800-0x4001 2BFF

1 KB

Reserved

0x4001 2C00-0x4001 2FFF

1 KB

TIM1

0x4001 3800-0x4001 3BFF

1 KB

Reserved

0x4001 3000-0x4001 33FF

1 KB

Reserved

0x4001 3400-0x4001 37FF

1 KB

DBGMCU

0x4001 3C00-0x4001 3FFF

1 KB

Reserved

0x4001 4000-0x4001 43FF

1 KB

TIM14

0x4001 4400-0x4001 47FF

1 KB

TIM16

0x4001 4800-0x4001 4BFF

1 KB

TIM17

0x4001 4C00-0x4001 63FF

7 KB

Reserved

0x4001 6400-0x4001 67FF

1 KB

Reserved

0x4001 6800-0x4001 7FFF

6 KB

Reserved

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

3/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 26** |

UM\_MM32F003\_q\_Ver1.19

bus

Addressing range

size

Peripherals

Remarks

AHB

0x4002 0000-0x4002 03FF

1 KB

DMA

0x4002 0400-0x4002 0FFF

3 KB

Reserved

0x4002 1000-0x4002 13FF

1 KB

RCC

0x4002 1400-0x4002 1FFF

3 KB

Reserved

0x4002 2000-0x4002 23FF

1 KB

Flash interface

0x4002 2400-0x4002 2FFF

3 KB

Reserved

0x4002 3000-0x4002 33FF

1 KB

CRC

0x4002 3400-0x4002 5FFF

11 KB

Reserved

0x4002 6000-0x4002 63FF

1 KB

Reserved

0x4002 6400-0x47FF FFFF

∼ 128 MB

Reserved

0x4800 0000-0x4800 03FF

1 KB

GPIOA

0x4800 0400-0x4800 07FF

1 KB

GPIOB

0x4800 0800-0x4800 0BFF

1 KB

GPIOC

0x4800 0C00-0x4800 0FFF

1 KB

GPIOD

0x4800 1000-0x5FFF FFFF

∼ 384 MB

Reserved

**1.3**

**SRAM**

Built-in static SRAM up to 2K bytes.

It can be accessed in bytes (8 bits), half words (16 bits) or words (32 bits). The starting address of SRAM is 0x2000 0000.

• Up to 2K bytes of SRAM on the data bus. Can be used by CPU or DMA with the fastest system clock and not

Insert anything waiting to be accessed.

**1.4**

Flash memory has two different storage areas:

• Main flash memory block, which includes application programs and user data area (if required)

• Information block, which contains four parts:

**–** Option bytes – Contains user configuration options for hardware and storage protection.

**–** System memory (System memory)-It contains the boot loader code. See built-in flash storage

Section.

The flash memory interface executes instructions and data access based on the AHB protocol. Its prefetch buffer function can speed up the CPU to execute code

speed.

**1.5**

**(Boot configuration)**

In the chip, three different startup modes can be selected by the level state of the BOOT0 pin and the configuration of the nBOOT1 bit, such as

As shown in the table below.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

4/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 27** |

UM\_MM32F003\_q\_Ver1.19

2.

Start mode selection

Boot mode

Explanation

**nBOOT1**

**BOOT0**

x

0

Main flash memory

The main flash memory is selected as the boot area

0

1

System memory

System memory selected as boot area

1

1

Built-in SRAM

The built-in SRAM is selected as the boot area

After the device is reset, the BOOT0 pin value is latched on the fourth rising edge of SYSCLK. The user can set

nBOOT1 bit cooperates with BOOT0 pin to select the boot mode.

When waking up from standby mode, the CPU will resample the BOOT0 pin value and nBOOT1 bit, so when there is a standby application

When it is used, it is necessary to keep the startup mode set. After the startup delay, the CPU gets the heap from address 0x0000 0000

The address at the top of the stack, and code execution starts from the address indicated by 0x0000 0004 in the boot memory.

Depending on the selected boot mode, the main flash memory, system memory or SRAM is accessed as follows:

• Boot from main flash memory: the main flash memory is mapped into the boot storage space (0x0000 0000), but still

Can be accessed from the original address space (0x800 0000). That is, the contents of the flash memory can be accessed from two addresses,

0x0000 0000 or 0x800 0000.

• Boot from system memory: The system memory is mapped into the boot space (0x0000 0000), but it can still be

Some address spaces (0x1FFF F400) are accessed.

• Boot from built-in SRAM: SRAM is mapped to boot space (0x0000 0000), but it can still be

Some address space (0x2000 0000) access

The boot loader is stored in the system memory, and the flash memory can be reprogrammed through UART1.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

5/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 28** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

2

**(FLASH)**

Embedded Flash (FLASH)

**2.1**

• Up to 16K bytes of flash memory

The characteristics of the flash memory interface are:

• Data interface with prefetch buffer (2 × 64 bits)

• Select byte loader

• Flash program/erase operation

• Access/write protection

• Low power mode

**2.2**

**2.2.1**

The flash memory space consists of 64-bit wide storage units, which can store both code and data. The main flash memory block is divided into 16 pages (each

Page 1K bytes) or 4 sectors (4K bytes per sector) block, set write protection in units of sectors (see storage protection phase

Content).

3. Flash

Module

name

address

Size **(** bytes **)**

Main memory block

Page 0

0x0800 0000-0x0800 03FF

1K

Page 1

0x0800 0400-0x0800 07FF

1K

Page 2

0x0800 0800-0x0800 0BFF

1K

Page 3

0x0800 0C00-0x0800 0FFF

1K

…

…

…

…

…

…

Page 12

0x0800 3000-0x0800 33FF

1K

Page 13

0x0800 3400-0x0800 37FF

1K

Page 14

0x0800 3800-0x0800 3BFF

1K

Page 15

0x0800 3C00-0x0800 3FFF

1K

Information block

Protection byte

0x1FFE 0000-0x1FFE 01FF

0.5K

Confidential space

0x1FFE 1000-0x1FFE 1BFF

3K

System memory

0x1FFF F400-0x1FFF F7FF

1K

Option byte

0x1FFF F800-0x1FFF F80F

16

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

6/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 29** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

Module

name

address

Size **(** bytes **)**

Flash memory interface register

FLASH\_ACR

0x4002 2000-0x4002 2003

4

FLASH\_KEYR

0x4002 2004-0x4002 2007

4

FLASH\_OPTKEYR

0x4002 2008-0x4002 200B

4

FLASH\_SR

0x4002 200C-0x4002 200F

4

FLASH\_CR

0x4002 2010-0x4002 2013

4

FLASH\_AR

0x4002 2014-0x4002 2017

4

Keep

0x4002 2018-0x4002 201B

4

FLASH\_OBR

0x4002 201C-0x4002 201F

4

FLASH\_WRPR

0x4002 2020-0x4002 2023

4

**2.2.2**

**FLASH**

The embedded Flash module can be directly addressed and accessed like a universal storage space. Any read operation on the contents of the Flash module

Work must go through a special judgment process.

Both instruction fetch and data fetch are read and accessed through the AHB bus, and can be accessed in accordance with the Flash access control register (FLASH\_

ACR) is performed in the manner specified by the options:

• Fetch instruction: the pre-fetch buffer can be enabled to increase the CPU running speed

• Incubation period: the number of waiting bits to ensure correct reading.

The CPU fetches instructions through the AHB bus. The function of the prefetch finger module is to improve the efficiency of fetching fingers.

Prefetch buffer (2 64-bit): automatically opened after each reset, due to the size of each buffer (64-bit)

It has the same bandwidth as the flash memory, so the content of the entire buffer can be updated by only one operation of reading the flash memory. Due to pre

With the presence of the buffer, the CPU can work at a higher frequency. Each time the CPU fetches a word with a maximum of 32 bits, take one

Instruction, the next instruction is already waiting in the buffer.

The prefetch controller will grasp the timing of accessing the flash based on the available space of the prefetch buffer. When the prefetch buffer exists

When there is at least one free space, the prefetch controller will initiate a read request. After reset, prefetch refers to the default state of the buffer

The state is open. Only when SYSCLK is lower than 24MHz and the AHB clock has not been divided

(SYSCLK must be equal to HCLK) to turn on/off the prefetch buffer. Normally, prefetch means that the buffer is in the initial

The switch state has already been decided during the process, and the MCU was running under the internal 8MHz oscillator.

Note: When the prescaler of the AHB clock is not equal to 1, prefetch means that the buffer must have an access latency.

In order to protect the correct reading of Flash, the middle finger of LATENCY[2:0] in the Flash access control register must be

The predetermined prefetch refers to the speed ratio of the controller. This value is equal to the amount of time required to be inserted between each access to the Flash and the next access.

The number of waiting cycles. After reset, this value defaults to zero, which is the state where no wait cycle is inserted.

**2.2.3**

**Flash**

The embedded flash memory supports online programming and in-application programming.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

7/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 30** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

ICP refers to the use of SWD to change the contents of Flash online and burn the user code into the microcontroller. ICP provided a

This simple and efficient method eliminates the problem of chip clamping when burning chips.

Unlike the ICP method, IAP (in-application programming) can use any communication interface (I/Os, USB,

UART, I2C, SPI, etc.) Download programs or data. IAP allows users to rewrite applications while running programs

Program, provided that part of the application program must be pre-programmed by ICP/ISP.

The programming and erasing operations can be completed within the entire product operating voltage range. This operation is completed by the following 7 registers:

• Keyword register (FLASH\_KEYR)

• Option byte keyword register (FLASH\_OPRKEYR)

• Flash control register (FLASH\_CR)

• Flash status register (FLASH\_SR)

• Flash address register (FLASH\_AR)

• Option byte register (FLASH\_OBR)

• Write protection register (FLASH\_WRPR)

As long as the CPU does not access the Flash space, the ongoing Flash write operation will not hinder the operation of the CPU. In other words,

While writing/erasing the Flash, any access to the Flash will stop the bus until the write/erase operation

The execution will continue after the operation is completed, which means that it is not possible to fetch and access the data while writing/erasing the Flash.

When writing/erasing the Flash space, the internal oscillator (HSI) must be on.

**Flash**

After reset, the flash memory is protected by default, which prevents accidental erasure. FLASH\_CR

The register is not allowed to be rewritten unless a series of unlock operations for the FLASH\_KEYR register are performed

Access permissions for FLASH\_CR. This series of operations consists of the following 2 write operations:

• Write keyword 1 = 0x45670123

• Write keyword 2 = 0xCDEF89AB

Any wrong sequence will lock FLASH\_CR until the next reset.

When a keyword error occurs, a hardware error interrupt is triggered by a bus error. KEY1 error will be interrupted immediately, KEY1

Correct but KEY2 error will cause an interrupt when KEY2 is wrong.

The main flash memory can be programmed with 16 bits at a time. When the PG bit in FLASH\_CR is 1, write a directly to the corresponding address

Half word (16 bits) is a programming operation. If you try to write other lengths than half words, it will cause a hardware error interrupt.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

8/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 31** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

170199

Read LOCK bit in

FLASH\_CR

LOCK bit in FLASH\_CR=1

Write PG bit in FLASH\_CR to 1

Perform half-word write at the

desired address

BSY bit in FLASH\_SR=1

Check the programmed value by

reading the programmed address

Perform unlock sequence

Yes

No

Yes

2.

The Flash memory interface will pre-read whether the byte to be programmed is all 1, if not, then the programming operation will automatically take

Cancel, and a programming error warning is displayed on the PGERR bit of the FLASH\_SR register.

If the write protection bit in FLASH\_WRPR corresponding to the address to be programmed is valid, there will also be no programming action.

This will also generate a programming error warning. After the programming action, the EOP bit in the FLASH\_SR register will give a prompt.

The programming process of the main flash memory in standard mode is as follows:

• Check the BSY bit in FLASH\_SR to confirm that the last operation has ended

• Set the PG bit in the FLASH\_CR register

• Write data to the target address in half-word units

• Wait for BSY in the FLASH\_SR register to return to zero

• Read data to verify

Note: When the BSY bit in FLASH\_SR is 1, these registers cannot be written.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

9/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 32** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

**Flash**

Flash memory can be erased in units of pages, or can be erased in one piece.

The steps to erase a page are as follows:

• Check the BSY bit in FLASH\_SR to confirm that the last operation has ended

• Set the PER bit in the FLASH\_CR register to 1

• Write FLASH\_AR register to select the page to be erased

• Set the STRT bit in the FLASH\_CR register to 1

• Wait for BSY in FLASH\_SR to return to zero

• Read erased pages to verify

227522

Read LOCK bit in

FLASH\_CR

LOCK bit in FLASH\_CR = 1

Write PER bit in FLASH\_CR

Write into FAR an address

within the page to erase

Write STRT bit in

FLASH\_CR to 1

BSY bit in FLASH\_SR = 1

Check the page is erased by

reading all the addresses

in the page

Preform unlock sequence

Yes

Yes

No

3. Flash

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

10/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 33** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

You can use the entire chip erase command to erase the entire Flash user area at once, but the information block will not be affected by this command.

The steps are as follows:

• Check the BSY bit in FLASH\_SR to confirm that the last operation has ended

• Set the MER bit in the FLASH\_CR register to 1

• Set the STRT bit in the FLASH\_CR register to 1

• Wait for the BSY bit to return to zero

• Read all pages and verify

079384

Read LOCK bit in

FLASH\_CR

Write MER bit in

FLASH\_CR to 1

Write STRT bit in

FLASH\_CR to 1

Check the erase operation by

reading all the addresses in the

user memory

LOCK bit in FLASH\_CR=1

Perform unlock sequence

BSY bit in FLASH\_SR=1

Yes

Yes

No

4. Flash

The programming of the option byte is different from the conventional user address, including 2 write protections and 1 hardware configuration. Remove Flash access restrictions

After that, you also need to complete the keyword write operation to the FLASH\_OPTKEYR register. After completing this operation, FLASH\_CR

The OPTWRE bit in the register will be set to '1', then the OPTPG bit in FLASH\_CR can be set first, and then

Write the target address in half-word units. The same will automatically check whether the option byte is 1, otherwise the relevant operation will be cancelled and

And the WRPRTERR bit in FLASH\_SR indicates an error. After the programming operation, it will be sent by FLASH\_SR

The EOP bit of the memory gives a hint.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

11/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 34** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

The option byte is 16-bit data, the effective data is the lower 8 bits, and the upper 8 bits are the inverse of the lower 8 bits. During the programming process, hard

The software will automatically set the upper 8 bits to the inverse of the lower 8 bits to ensure that the write value of the option byte is always correct. Proceed as follows:

• Check the BSY bit in the FLASH\_SR register to ensure the end of the last operation

• Unlock the OPTWRE bit in the FLASH\_CR register

• Set the OPTPG bit in the FLASH\_CR register to 1

• Write data (half word) to the target address

• Wait for the BSY bit to return to zero

• Read and verify that when the protection option byte is changed from the protected state to the unprotected state, it will automatically trigger a full chip erase.

If the user only wants to rewrite other bytes, it will not cause the entire chip to be erased. This mechanism is used to protect the contents of the Flash.

864155

**Read LOCK bit in**

**FLASH\_CR**

**LOCK bit in FLASH\_CR=1**

**Perform Unlock**

**Flash sequence**

**No**

**Yes**

**Unlock FLash OPTION**

**sequence**

**Write OPTPG bit in FLASH\_CR**

**to 1**

**Perform half-word write at**

**the desired address**

**BSY bit in FLASH\_SR=1**

**Check the programmed value**

**by reading the programmed**

**address**

**No**

**Yes**

5.

The process of erasing option bytes is as follows:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

12/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 35** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

• Check the BSY bit in the FLASH\_SR register to ensure the end of the last operation

• Unlock the OPTWRE bit in the FLASH\_CR register

• Set the OPTER bit in the FLASH\_CR register to 1

• Set the STRT bit in the FLASH\_CR register to 1

• Wait for the BSY bit to return to zero

• Read and verify

806825

**Read LOCK bit in**

**FLASH\_CR**

**LOCK bit in FLASH\_CR=1**

**Write FLASH OTPTION Based**

**Address to FLASH\_ACR**

**Write STRT bit in FLASH\_CR**

**to 1**

**BSY bit in FLASH\_SR=1**

**Check the Flash Op on erase**

**opera on by reading all the**

**addresses in the Flash Op on**

**memory**

**Perform Unlock**

**Flash sequence**

**No**

**Yes**

**No**

**Yes**

**Write OPTER bit in FLASH\_CR**

**to 1**

**Unlock FLash OPTION**

**sequence**

6.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

13/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 36** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

**2.3**

It can prevent the code in the Flash area of ​​​​​the user area from being read by untrusted code, and can also prevent the code from running while the program is flying.

For accidental erasure of Flash, the smallest unit of write protection is one sector (4 pages).

**2.3.1**

Write protection is controlled in units of one sector (4 pages), configure the WRP bit in the option byte, and subsequent system reset

The new option byte will be loaded to enable this protection. If you try to write or erase a protected sector, it will cause

The WRPRTERR flag in FLASH\_SR is set.

This situation is common in the user's own startup program that is programmed in the program:

• Use the OPTER bit of the flash control register (FLASH\_CR) to erase the entire option byte area;

• Perform a system reset and reload the option byte (including the new WRP byte); the write protection is released.

Using this method, the write protection of the entire main flash memory module except page 0 ∼ page 3 will be released. Page 0 ∼ page 3 are still in

Write protection.

**2.3.2**

By default, the option byte block is always readable and write-protected. To write to the option byte block (program/erase

Except) First write the correct key sequence in OPTKEYR (same as when locking), then allow the option byte block

For write operations, the OPTWRE bit in the FLASH\_CR register indicates that writing is allowed, and clearing this bit will prohibit the write operation.

**2.4 Flash**

4. Flash

Interrupt event

Event flag

Enable control bit

End of operation

EOP

EOPIE

Write protection error

WRPRTERR

ERRIE

Programming error

PGERR

ERRIE

**2.5**

The option byte is configured by the user according to the needs of the application; for example: you can choose to use the hardware mode watchdog or software watch

Door dog.

Each 32-bit word in the option byte is divided into the following formats:

5.

Bit **31** ∼ **24**

Bit **23** ∼ **16**

Bit **15** ∼ **8**

Bit **7** ∼ **0**

Inverse of option byte 1

Option byte 1

Inverse of option byte 0

Option byte 0

Note: The reverse code is automatically realized by the hardware, and the software write is invalid

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

14/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 37** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

The organization of option bytes in the option byte block is shown in the following table.

The option byte can be read from the memory address listed in the table below, or from the option byte register (FLASH\_OBR).

Note: The newly written option byte (user or read/write protected) will only take effect after the system is reset.

6.

address

**[31** : **24]**

**[23** : **16]**

**[15** : **8]**

**[7** : **0]**

0x1FFF F800

nUSER

USER

0x1FFF F804

nData1

Data1

nData0

Data0

0x1FFF F808

nWRP1

WRP1

nWRP0

WRP0

0x1FFF F80C

nWRP3

WRP3

nWRP2

WRP2

7.

Memory address

Option byte

0x1FFF F800

Bit [31:24] nUSER

Bit [23:16] USER: User option byte (stored in FLASH\_OBR[9:2]). This byte is used to configure the following

Features:

Select watchdog event: hardware or software

Note: Only bit [20] and bit [18:16] are used, and bit [23:21] and bit [19] are not used.

Bit 20: nBOOT1

Bit 18: nRST\_STDBY

0: Reset occurs when entering standby mode

1: No reset occurs when entering standby mode

Bit 17: nRST\_STOP

0: Reset when entering STOP mode

1: No reset occurs when entering STOP mode

Bit 16: WDG\_SW

0: hardware watchdog

1: Software watchdog

0x1FFF F804

Datax: 2 bytes of user data

This address can be programmed using option byte programming.

Bit [31:24]: nData1

Bit[23:16]: Data1 (stored in FLASH\_OBR[25:18])

Bit [15:8]: nData0

Bit[7:0]: Data0 (stored in FLASH\_OBR[17:10])

0x1FFF F808

WRPx: Flash write protection option byte

Bit [31:24]: nWRP1

Bit[23:16]: WRP1 (stored in FLASH\_WRPR[15:8])

Bit [15:8]: nWRP0

Bit[7:0]: WRP0 (stored in FLASH\_WRPR[7:0])

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

15/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 38** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

Memory address

Option byte

0x1FFF F80C

WRPx: Flash write protection option byte

Bit [31:24]: nWRP3

Bit[23:16]: WRP3 (stored in FLASH\_WRPR[31:24])

Bit [15:8]: nWRP2

Bit[7:0]: WRP2 (stored in FLASH\_WRPR[23:16])

Each bit in the option byte WRPx is used to protect 4 memory pages in main memory:

0: implement write protection

1: No write protection

Four user option bytes are used to protect a total of 128K bytes of main memory.

WRP0: Write protection on pages 0 to 31

WRP1: Write protection on pages 32 to 63

WRP2: Write protection on pages 64 to 95

WRP3: Write protection on pages 96 to 127

After each system reset, the option byte loader (OBL) reads the data of the information block and saves it in the option byte register (

FLASH\_OBR); each selection bit has its inverted bit in the information block, and the inverted bit is used for verification when the selection bit is loaded

To verify whether the selection bit is correct, if there is any difference, an option byte error flag (OPTERR) will be generated. Present

When an option byte error occurs, the corresponding option byte is forced to 0xFF. When the option byte and its complement are both 0xFF

(The state after erasing), then turn off the above verification function.

All selection bits (excluding their inverted bits) are used to configure the microcontroller, and the CPU can read the option byte register.

**2.6 Flash**

8. FLASH

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

FLASH\_ACR

Flash access control register

0x00000018

[Section 2.6.1](https://translate.googleusercontent.com/translate_f#38)

0x04

FLASH\_KEYR

FPEC key register

0xXXXXXXXX

[Section 2.6.2](https://translate.googleusercontent.com/translate_f#39)

0x08

FLASH\_OPTKEYR

Flash OPTKEY register

0xXXXXXXXX

[Section 2.6.3](https://translate.googleusercontent.com/translate_f#39)

0x0C

FLASH\_SR

Flash status register

0x00000000

[Section 2.6.4](https://translate.googleusercontent.com/translate_f#40)

0x10

FLASH\_CR

Flash control register

0x00000080

[Section 2.6.5](https://translate.googleusercontent.com/translate_f#41)

0x14

FLASH\_AR

Flash address register

0x00000000

[Section 2.6.6](https://translate.googleusercontent.com/translate_f#42)

0x1C

FLASH\_OBR

Option byte register

0x03FFFC1C

[Section 2.6.7](https://translate.googleusercontent.com/translate_f#42)

0x20

FLASH\_WRPR

Write protection register

0xFFFFFFFF

[Section 2.6.8](https://translate.googleusercontent.com/translate_f#43)

**2.6.1**

**(FLASH\_ACR)**

Address offset: 0x00

Reset value: 0x0000 0018

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

16/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 39** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

HLFCYA

rw

rw

Reserved

LATENCY

PRFTBE

Reserved

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 5

Reserved

Reserved, always read as 0.

4

PRFTBE

rw

0x01

Prefetch buffer enable

0: close prefetch buffer

1: enable prefetch buffer

3

HLFCYA

rw

0x01

Flash half cycle access enable

0: prohibit half-cycle access

1: Enable half-cycle access

2: 0

LATENCY

rw

0x00

Latency

These bits represent the ratio of SYSCLK (system clock) period to flash access time

example.

000: Zero wait state, when 0 <SYSCLK ≤ 24MHz

001: a waiting state, when 24MHz <SYSCLK ≤ 48MHz

**2.6.2**

**(FLASH\_KEYR)**

Address offset: 0x04

Reset value: 0xXXXX XXXX

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

FKEYR

FKEYR

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

FKEYR

w

0xXXXX

XXXX

FPEC key (Flash key)

These bits are used to enter the unlock key of FPEC.

Note: All these bits are write only and return 0 when read.

**2.6.3**

**OPTKEY**

**(FLASH\_OPTKEYR)**

Address offset: 0x08

Reset value: 0xXXXX XXXX

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

17/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 40** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

OPTKEYR

OPTKEYR

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

OPTKEYR

w

0xXXXX

XXXX

Option byte key

These bits are used to enter the key of the option byte to release OPTWRE.

Note: All these bits are write only and return 0 when read.

**2.6.4**

**(FLASH\_SR)**

Address offset: 0x0C

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

PGERR

r

Reserved

WRPRTE

RR

Reserved

BSY

Res.

Res.

EOP

rc\_w1

rc\_w1

rc\_w1

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 6

Reserved

Reserved, always read as 0.

5

EOP

rc\_w1

0x00

End of operation

When the flash operation (program/erase) is completed, the hardware sets this bit to '1' and writes

Enter '1' to clear this status.

Note: Each successful program or erase will set the EOP state.

4

WRPRTERR

rc\_w1

0x00

Write protection error

When trying to program a write-protected flash memory address, the hardware sets this bit to '1' and writes

Enter '1' to clear this status.

3

Reserved

Reserved, always read as 0.

2

PGERR

rc\_w1

0x00

Programming error

When trying to program an address whose content is not '0xFFFF', the hardware sets this bit

Is '1', write '1' to clear this status.

Note: Before programming operation, STRT of FLASH\_CR register must be cleared

Bit.

1

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

18/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 41** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

BSY

r

0x00

Busy

This bit indicates that the flash operation is in progress. At the beginning of the flash operation, this bit is

Set to '1'; this bit is cleared to '0' at the end of the operation or when an error occurs.

**2.6.5**

**(FLASH\_CR)**

Address offset: 0x10

Reset value: 0x0000 0080

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

MER

rw

Reserved

OPTPG

PG

PER

Res.

OPTER

rw

rw

rw

STRT

LOCK

Res.

OPTWRE

ERRIE

Res.

EOPIE

Reserved

rw

rw

rw

rw

rw

rc\_w0

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 13

Reserved

Reserved, always read as 0.

12

EOPIE

rw

0x00

End of operation interrupt enable

This bit allows the EOP bit in the FLASH\_SR register to become '1'

An interrupt is generated.

0: Disable interrupt generation

1: Allow interrupt generation

11

Reserved

Reserved, always read as 0.

10

ERRIE

rw

0x00

Error interrupt enable

This bit allows an interrupt to be generated when an FPEC error occurs (when FLASH\_SR sends

(When PGERR/WRPRTERR in the register is set to '1').

0: Disable interrupt generation

1: Allow interrupt generation

9

OPTWRE

rc\_w0

0x00

Option byte write enable

When this bit is '1', the option byte is allowed to be programmed. When

After the correct key sequence is written in the FLASH\_OPTKEYR register, this bit is

Set to '1'.

Software can write 0 to clear this bit.

8

Reserved

Reserved, always read as 0.

7

LOCK

rw

0x01

Lock

Can only write '1'. When this bit is '1', it means FPEC and FLASH\_CR

Is locked. After detecting the correct unlock sequence, the hardware automatically clears this bit

Is '0'.

After an unsuccessful unlock operation, this bit cannot be changed until the next system reset

Was changed.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

19/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 42** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6

STRT

rw

0x00

Start

When this bit is '1', an erase operation will be triggered. This bit can only be set by software

Is '1' and automatically clears '0' when BSY becomes '1'.

5

OPTER

rw

0x00

Erase option byte (Option byte erase)

Erase option bytes.

4

OPTPG

rw

0x00

Option byte programming

Program the option byte.

3

Reserved

Reserved, always read as 0.

2

MER

rw

0x00

Mass erase

Select to erase all user pages.

1

PER

rw

0x00

Page erase

Select Erase Page.

0

PG

rw

0x00

Programming

Select the programming operation.

**2.6.6**

**(FLASH\_AR)**

Address offset: 0x014

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

FAR

FAR

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

FAR

w

0x0000

0000

User Option Byte (Flash Address)

Select the address to be programmed when programming, and select the address to be erased when performing page erase

The erased page.

Note: When the BSY bit in FLASH\_SR is '1', this register cannot be written

Device.

Modified by hardware to the current/last used address. During the page erase operation, this register must be modified to specify the

page.

**2.6.7**

**(FLASH\_OBR)**

Address offset: 0x1C

Reset value: 0x03FF FC1C

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

20/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 43** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

Data1

OPTERR

Reserved

Reserved

Data0

Data0

Res.

WDG

\_SW

nRST\_

STOP

nRST\_

STDBY

r

r

r

nBOOT1

Res.

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 26

Reserved

Reserved, always read as 0.

25: 18

Data1

r

0xFF

Data1

17: 10

Data0

r

0xFF

Data0

9: 7

Reserved

Reserved, always read as 0.

6

nBOOT1

r

0x00

nBOOT1

5

Reserved

Reserved, always read as 0.

4

nRST\_STDBY r

0x01

Reset event when entering standby mode

0: Reset occurs when entering standby mode

1: No reset occurs when entering standby mode

3

nRST\_STOP

r

0x01

Reset event when entering stop mode

0: Reset when entering STOP mode

1: No reset occurs when entering STOP mode

2

WDG\_SW

r

0x01

Select watchdog event

0: hardware watchdog

1: Software watchdog

1

Reserved

Reserved, always read as 0.

0

OPTERR

r

0x00

Option byte error

When this bit is '1', it means that the option byte and its inverted code do not match.

Note: This bit is read-only.

The reset value of this register is related to the value written in the option byte. The reset value of the OPTERR bit is related to the load option word.

The result of comparing the option byte with its inverse code at the time of festival is related.

**2.6.8**

**(FLASH\_WRPR)**

Address offset: 0x20

Reset value: 0xFFFF FFFF

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

r

WRP

WRP

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

21/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 44** |

UM\_MM32F003\_q\_Ver1.19

(FLASH)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

WRP

r

0xFFFF

FFFF

Write protect

This register contains the write protection option byte loaded by OBL.

0: write protection takes effect

1: Write protection is invalid

Note: These bits are read-only.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

22/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 45** |

UM\_MM32F003\_q\_Ver1.19

(CRC)

3

**(CRC)**

Cyclic Redundancy Check Calculation Unit (CRC)

**3.1 CRC**

Cyclic Redundancy Check (CRC) calculation unit is based on a fixed generator polynomial to get the CRC calculation of any 32-bit full word

result. In other applications, CRC technology is mainly used to verify the accuracy and integrity of data transmission or data storage

Sex. The standard EN/IEC60335-1 provides a method to verify the integrity of flash memory. CRC calculation unit can

To calculate the software logo when the program is running, and then compare it with the reference logo generated when connecting, and then store it in the

Fixed memory space.

**3.2 CRC**

• Use CRC-32 (Ethernet) polynomial: 0x4C11DB7

X 32 + X 26 + X 23 + X 22 + X 16 + X 12 + X 11 + X 10 + X 8 + X 7 + X 5 + X 4 + X 2 + X +1

• A 32-bit data register is used for input/output.

• CRC calculation time: 4 AHB clock cycles (HCLK)

• General purpose 8-bit register (can be used to store temporary data)

The following figure is the block diagram of CRC calculation unit

027571

AHB ᙫ㓯

ᮠᦞᇴᆈಘ˄䗃ࠪ˅

CRC data: 0x4C11DB7)

ᮠᦞᇴᆈಘ (without inspection)

7. CRC

**3.3 CRC**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

23/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 46** |

UM\_MM32F003\_q\_Ver1.19

(CRC)

The CRC calculation unit contains a 32-bit data register:

• When writing to this register, as an input register, new data to be CRC calculated can be input.

• When reading this register, the result of the last CRC calculation is returned.

Each time the data register is written, the calculation result is the combination of the previous CRC calculation result and the new calculation result (for the entire

32-bit words are used for CRC calculation instead of byte-by-byte calculation).

The write operation is suspended during the CRC calculation, so the register CRC\_DR can be written back to back or continuously

Write-read operation.

The CRC\_DR register can be reset to 0xFFFF FFFF by setting the RESET bit of the register CRC\_CTRL.

This operation does not affect the data in the register CRC\_IDR.

**3.4 CRC**

The CRC calculation unit includes 2 data registers and a control register.

9. CRC

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

CRC\_DR

CRC data register

0xFFFFFFFF

[Section 3.4.1](https://translate.googleusercontent.com/translate_f#46)

0x04

CRC\_IDR

CRC Independent Data Register

0x00000000

[Section 3.4.2](https://translate.googleusercontent.com/translate_f#46)

0x08

CRC\_CTRL

CRC control register

0x00000000

[Section 3.4.3](https://translate.googleusercontent.com/translate_f#47)

**3.4.1**

**CRC**

**(CRC\_DR)**

Address offset: 0x00

Reset value: 0xFFFF FFFF

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

DR

DR

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

DR

rw

0xFFFF

FFFF

DR: Data register bits

Used as input register when writing new data of CRC calculator

Return CRC calculation result when reading

**3.4.2**

**CRC**

**(CRC\_IDR)**

Address offset: 0x04

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

24/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 47** |

UM\_MM32F003\_q\_Ver1.19

(CRC)

Reserved

Reserved

IDR

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 8

Reserved

Always read as 0.

7: 0

IDR

rw

0x00

IDR: General-purpose 8-bit data register bit (General-purpose 8-bit data

register bits)

It can be used to temporarily store 1 byte of data.

The CRC reset generated by the RESET bit of the register CRC\_CTRL

The memory has no effect.

Note: This register does not participate in CRC calculation and can store any data.

**3.4.3**

**CRC**

**(CRC\_CTRL)**

Address offset: 0x08

Reset value: 0x0000 0000

Reserved

Reserved

RESET

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 1

Reserved

Always read as 0.

0

RESET

w

0x00

RESET: Reset CRC calculation unit (CRC reset)

Set the data register to 0xFFFF FFFF.

You can only write '1' to this bit, it will be automatically cleared by hardware.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

25/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 48** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

4

**(PWR)**

Power control (PWR)

**4.1**

The chip's operating voltage (V DD ) is 2.0V ~ 5.5V. Provide the required 1.5V power supply through the built-in voltage regulator.

264088

A/D converter

Temp. sensor

Reset block

V DDA domain

Standby circuitry

(Wakeup logic,

IWDG)

Voltage Regulartor

V DD domain

Core

Memories

digital

peripherals

1.5V domain

(V the SSA ) V The the REF -

( from 2.4 V up to V DDA ) V REF+

(V DD ) V DDA

(V SS ) V SSA

V DD

V SS

I/O Ring

8.

Note: V DDA and V SSA must be connected to V DD and V SS respectively .

**4.1.1**

**A/D**

In order to improve the accuracy of the conversion, the ADC uses an independent power supply, filtering and shielding from the printed circuit board

Glitch interference.

• ADC power supply pin is V DDA

• Independent power ground V SSA

If there is a V REF− pin (depending on the package), it must be connected to V SSA .

**4.1.2**

The regulator is always enabled after reset. It works in 3 different modes depending on the application method.

• Operating mode: The regulator provides 1.5V power (core, memory and peripherals) in normal power consumption mode.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

26/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 49** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

• Stop mode: The regulator provides 1.5V power in low power mode to save the contents of registers and SRAM.

• Standby mode: The regulator stops supplying power. The contents of registers and SRAM are all lost.

**4.2**

**4.2.1**

**(POR)**

**(PDR)**

The chip has a complete power-on reset (POR) and power-down reset (PDR) circuit, the system when the supply voltage reaches 1.5V

It can work normally.

When V DD /V DDA is lower than the specified limit voltage V POR /V PDR , the system remains in the reset state without external reset

Circuit. For details on power-on reset and power-on reset, please refer to the electrical characteristics section of the data sheet.

086030

V DD /V DDA

POR

PDR

Reset

90 mV

hysteresis

Temporiza on

t RSTTEMPO

9.

**4.2.2**

**(PVD)**

The user can use PVD to compare the V DD voltage with the PLS bit in the power control register (PWR\_CR) to monitor

To control the power supply, these bits select the threshold for monitoring the voltage.

PVD is enabled by setting the PVDE bit.

The PVDO flag in the power control/status register (PWR\_CSR) is used to indicate whether V DD is above or below PVD

Voltage threshold. This event is internally connected to line 16 of the external interrupt, if the interrupt is in the external interrupt register

If it is enabled, the event will generate an interrupt. When V DD drops below the PVD threshold or when V DD rises to the PVD threshold

Above the value, the PVD interrupt will be generated according to the rising/falling edge trigger setting of the external interrupt line 16. Eg,

This feature can be used to perform emergency shutdown tasks.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

27/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 50** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

586945

V DD /V DDA

PVD threshold

PVD output

100 mV

hysteresis

10. PVD

**4.3**

After the system or power is reset, the microcontroller is in a running state. When the CPU does not need to continue to run, you can use a variety of

Low power consumption mode to save power consumption, such as waiting for an external event. Users need to be based on the lowest power consumption, the fastest

Conditions such as start-up time and available wake-up sources select an optimal low-power mode.

The chip has three low-power modes:

• Sleep mode (CPU stopped, all peripherals including CPU peripherals, such as NVIC, system clock (SysTick), etc.

Running)

• Stop mode (all clocks are stopped, the contents of registers and SRAM are still saved)

• Standby mode (1.5V power is off, the contents of registers and SRAM are all lost.)

In addition, in the operating mode, power consumption can be reduced in one of the following ways:

• Reduce the system clock frequency

• Turn off unused peripheral clocks on APB and AHB buses

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

28/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 51** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

10.

mode

enter

wake

To **1.5V** regional clock

Impact

Clock to **V**DD area

Impact

Voltage Regulator

Sleep (SLEEP

NOW or SLEEP

ON EXIT)

WFI (Wait for

Interrupt)

Any interruption

CPU clock is off, yes

Other clocks and ADC

Clock has no effect

no

open

WFE (Wait for

Event)

Wake-up event

Downtime

PDDS bit

SLEEPDEEP bit

WFI or WFE

Any external interrupt (at

External interrupt register

Settings)

All using 1.5V

The regional clocks are all

shut down

The vibration of HSI and HSE

Swing off

open

Standby

PDDS

SLEEPDEEP bit

WFI or WFE

On the WKUP pin

Rising edge, NRST lead

External reset on the foot,

IWDG reset

turn off

**4.3.1**

In the running mode, by programming the prescaler register, any system clock (SYSCLK, HCLK,

PCLK1, PCLK2) speed. Before entering sleep mode, you can also use the prescaler to reduce the peripheral clock.

For details, please refer to: Clock Configuration Register (RCC\_CFGR)

**4.3.2**

In the running mode, it can be reduced at any time by stopping the clock (HCLK and PCLKx) for peripherals and memory

Less power consumption.

To further reduce power consumption in sleep mode, all peripheral clocks can be turned off before executing WFI or WFE instructions.

By setting AHB peripheral clock enable register (RCC\_AHBENR), APB2 peripheral clock enable register (RCC

\_APB2ENR) and APB1 peripheral clock enable register (RCC\_APB1ENR) to switch each peripheral module

bell.

**4.3.3**

Go to sleep by executing WFI or WFE instruction. According to SLEEPONEXIT in the CPU system control register

Bit value, there are two options for selecting the sleep mode entry mechanism:

• SLEEP-NOW: If the SLEEPONEXIT bit is cleared, when WFI or WFE is executed, the microcontroller

Enter sleep mode.

• SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the system starts from the lowest priority interrupt handler

When exiting in the middle, the microcontroller immediately enters sleep mode.

In sleep mode, all I/O pins maintain their state when in run mode.

For more details on how to enter sleep mode, refer to Table [11 and Table](https://translate.googleusercontent.com/translate_f#51) 12.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

29/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 52** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

11. SLEEP NOW

**SLEEP NOW** mode

Explanation

enter

Execute WFI (Wait for Interrupt) or WFE (Wait for Event) instruction under the following conditions:

-SLEEPDEEP = 0 and

-SLEEPONEXIT = 0

Refer to the CPU system control register.

drop out

If WFI is executed to enter sleep mode: Interrupt: Refer to the interrupt vector table

If WFE is executed to enter sleep mode: wake-up event: refer to wake-up event management

Wakeup delay

no

12. SLEEP ON EXIT

**SLEEP ON EXIT** mode

Explanation

enter

Execute WFI (Wait for Interrupt) or WFE (Wait for Event) instruction under the following conditions:

-SLEEPDEEP = 0 and

-SLEEPONEXIT = 1

Refer to the CPU system control register.

drop out

If WFI is executed to enter sleep mode: interrupt or clear bit 1 of the CPU control register

If WFE is executed to enter sleep mode: wake-up event: refer to wake-up event management

Wakeup delay

no

**4.3.4**

The shutdown mode is based on the deep sleep mode of the CPU, which combines the clock control mechanism of the peripherals. In the shutdown mode, the voltage

The regulator can operate in normal mode. At this time, all clocks in the 1.5V power supply area are stopped, and HSI and HSE vibrate.

The function of the oscillator is disabled, and the contents of SRAM and registers are retained.

In the stop mode, all I/O pins maintain their state in the run mode.

For details on how to enter shutdown mode, see Table [13](https://translate.googleusercontent.com/translate_f#53) .

The following functions can be selected by programming independent control bits:

• Independent watchdog (IWDG): IWDG can be started by writing to the watchdog's key register or hardware selection.

• Internal oscillator (LSI oscillator): Set by the LSION bit of the control/status register (RCC\_CSR).

In shutdown mode, if the ADC is not turned off before entering this mode, these peripherals still consume current. by

This peripheral can be turned off by setting the ADON bit of register ADC\_CR2 to 0. Other unused GPIO need to be set

Set analog input, otherwise there is current consumption.

For details on how to exit the shutdown mode, see Table [13 for](https://translate.googleusercontent.com/translate_f#53) details .

When an interrupt or wake-up event causes the shutdown mode to exit, the HSI oscillator is selected as the system clock. Clock frequency is HSI

Divided by 6.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

30/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 53** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

When the voltage regulator is in normal power consumption mode and the system exits from shutdown mode, there will be an additional start-up delay.

13.

Stop mode

Explanation

enter

Execute WFI (Wait for Interrupt) or WFE (Wait for Event) instruction under the following conditions:

-Set the SLEEPDEEP bit in the CPU system control register

-Clear the PDDS bit in the power control register (PWR\_CR)

Note: In order to enter the stop mode, all external interrupt request bits (suspend register (EXTI\_PEND))

The flags must be cleared, otherwise the entry process in shutdown mode will be skipped and the program will continue to run.

drop out

The WFI (Wait for Interrupt) instruction is executed under the following conditions:

Any external interrupt pin is set to interrupt mode (the corresponding external interrupt vector must be enabled in NVIC).

See interrupt vector table

The WFE (Wait for Event) instruction is executed under the following conditions:

Any external interrupt lead is set to the event mode. See wake event management.

Wakeup delay

HSI wake-up time

**4.3.5**

Standby mode can achieve the lowest power consumption of the system. This mode is to turn off the voltage regulator during the CPU deep sleep mode. entire

The 1.5V power supply area was powered off. The HSI and HSE oscillators are also powered down. SRAM and register contents are lost. Standby

The road maintains power.

For details on how to enter standby mode, see Table [14](https://translate.googleusercontent.com/translate_f#53) .

You can select the following standby mode functions by setting independent control bits:

• Independent watchdog (IWDG): IWDG can be started by writing to the watchdog's key register or hardware selection.

• Internal oscillator (LSI oscillator): Set by the LSION bit of the control/status register (RCC\_CSR).

When an external reset (NRST pin), IWDG reset, or rising edge on WKUP pin, the microcontroller switches from standby mode

Quit. After waking up from standby, all registers are reset except the power control/status register (PWR\_CSR).

The code execution after waking up from standby mode is equivalent to the execution after reset (sampling the start mode pin, reading the reset vector, etc.).

The power control/status register (PWR\_CSR) will instruct the core to exit from standby.

For details on how to exit standby mode, see Table [14 for](https://translate.googleusercontent.com/translate_f#53) details .

14.

Standby mode

Explanation

enter

Execute WFI (Wait for Interrupt) or WFE (Wait for Event) instruction under the following conditions:

-Set the SLEEPDEEP bit in the CPU system control register

-Set the PDDS bit in the power control register (PWR\_CR)

-Clear the WUF bit in the power control/status register (PWR\_CSR)

drop out

Rising edge of WKUP pin, external reset on NRST pin, IWDG reset.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

31/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 54** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

Standby mode

Explanation

Wakeup delay

The voltage regulator starts during the reset phase.

**/**

In standby mode, all I/O pins are in a high-impedance state, except for the following pins:

• Reset pin (always active)

• TAMPER pin when set to anti-intrusion or calibration output

• Wake-up pin is enabled

By default, if the microprocessor is put into stop or standby mode while debugging the microprocessor, the debugging will be lost

connection. This is because the CPU core has lost its clock.

However, by setting some configuration bits in the DBGMCU\_CR register, you can debug the software while using the low power mode

Pieces. For more details, please refer to: Debug Support in Low Power Mode.

**4.4**

15.

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

PWR\_CR

Power control register

0x00000000

[Section 4.4.1](https://translate.googleusercontent.com/translate_f#54)

0x04

PWR\_CSR

Power control/status register

0x00000000

[Section 4.4.2](https://translate.googleusercontent.com/translate_f#55)

**4.4.1**

**(PWR\_CR)**

Address offset: 0x00

Reset value: 0x0000 0000 (cleared when waking up from standby mode)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

Reserved

PLS

PVDE CSBF CWUF PDDS

Reserved

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 13

Reserved

Always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

32/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 55** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

**Bit**

**Field**

**Type**

**Reset**

**Description**

12: 9

PLS

rw

0x00

PVD level selection

These bits are used to select the voltage threshold of the power supply voltage monitor

0000: 1.8V 0100: 3.0V 1000: 4.2V

0001: 2.1V 0101: 3.3V 1001: 4.5V

0010: 2.4V 0110: 3.6V 1010: 4.8V

0011: 2.7V 0111: 3.9V Others: reserved

Note: Refer to the electrical characteristics section of the data sheet for a detailed description.

8:5

Reserved

Always read as 0

4

PVDE

rw

0x00

Power voltage detector (PVD) enable (Power voltage detector en-

able)

1 = Turn on PVD

0 = Disable PVD

3

CSBF

rw

0x00

Clear standby flag (Clear standby flag)

Always read as 0

1 = Clear SBF standby bit (write)

0 = no effect

2

CWUF

rw

0x00

Clear wakeup flag

Always read as 0

1 = Clear WUF wakeup bit (write) after 2 system clock cycles

0 = no effect

1

PDDS

rw

0x00

Power down deepsleep

1 = CPU enters standby mode when it enters deep sleep

0 = CPU enters shutdown mode when entering deep sleep

0

Reserved

Always read as 0

**4.4.2**

**/**

**(PWR\_CSR)**

Address offset: 0x04

Reset value: 0x0000 0000 (not cleared when waking up from standby mode)

Compared to the standard APB read, the read register requires an additional APB cycle

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

EWUP

PVDO SBF

WUF

Reserved

Reserved

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 9

Reserved

Always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

33/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 56** |

UM\_MM32F003\_q\_Ver1.19

(PWR)

**Bit**

**Field**

**Type**

**Reset**

**Description**

8

EWUP

rw

0x00

Enable WKUP pin (Enable WKUP pin)

1 = WKUP pin is used to wake the CPU from standby mode, WKUP

The pin is forced to the input pull-down configuration (the rising edge on the WKUP pin will

(Wake up from standby mode)

0 = WKUP pin is general purpose I/O. The event on the WKUP pin cannot

CPU wakes up from standby mode

Note: This bit is cleared when the system is reset.

7:3

Reserved

Always read as 0

2

PVDO

rw

0x00

PVD output

This bit is only valid when PVD is enabled by PVDE bit

1 = V DD /V DDA is lower than the PVD threshold selected by PLS

0 = V DD /V DDA is higher than the PVD threshold selected by PLS

Note: PVD is stopped in standby mode. Therefore, after standby mode or after reset,

Before setting the PVDE bit, this bit is 0.

1

SBF

rw

0x00

Standby flag

This bit is set by hardware and can only be reset by POR/PDR (power-on/power-down reset) or

Set the CSBF bit of the power control register (PWR\_CR) to be cleared.

1 = System enters standby mode

0 = System is not in standby mode

0

WUF

rw

0x00

Wakeup flag

This bit is set by hardware and can only be reset by POR/PDR (power-on/power-down reset) or

Set the CWUF bit of the power control register (PWR\_CR) to clear.

1 = Wake-up event on WKUP pin

0 = no wake-up event has occurred

Note: When the WKUP pin is already high, use (by setting the EWUP bit)

When the WKUP pin is enabled, an additional event is detected.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

34/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 57** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

5

**(RCC)**

Reset and clock control (RCC)

**5.1**

Two reset modes are supported, namely system reset and power-on reset.

**5.1.1**

The system reset will reset the reset flag in the clock control register CSR, the standby and the power supply control register CSR

All registers except the wake flag.

When one of the following events occurs, a system reset is generated:

1. Low level on NRST pin (external reset)

2. Window watchdog count termination (WWDG reset)

3. Independent watchdog count termination (IWDG reset)

4. Software reset (SW reset)

The source of the reset event can be identified by looking at the reset status flag in the RCC\_CSR control status register.

By resetting the CPU interrupt application and resetting the SYSRESETREQ bit in the control register to '1', the software can be reset

Bit.

**5.1.2**

A power reset occurs when one of the following events occurs:

1. Power-on/power-off reset (POR/PDR reset)

2. Return from standby mode

Power reset will reset all registers.

[Figure](https://translate.googleusercontent.com/translate_f#58) 11 The reset source will eventually act on the RESET pin and remain low during the reset process. Reset entrance vector is fixed

Set at address 0x0000\_0004.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

35/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 58** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

112391

)

)

V DD / V DDA

External

Reset

NRST PAD

R PU

Pulse

Generator

~ 20 µ S

System Reset

WWDG Reset

IWDG Reset

Power on Reset

Software Reset

Standby Reset

11.

**5.2**

Three different clock sources can be used to drive the system clock (SYSCLK):

• HSI oscillator clock

• HSE oscillator clock

• LSI clock

When not in use, either clock source can be independently turned on or off, thereby optimizing system power consumption.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

36/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 59** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

280409

HSI

48 MHz

/ 6

HSI/6

AHB

Prescaler

/1,2..512

HSE OSC

2-24 MHz

LSI

40kHz

CSS

HSE

SYSCLK

SW

LSI

IWDGCLK

to Independent

Watchdog (IWDG)

HSI/6

HSE

SYSCLK

MCO

OSC\_IN

OSC\_OUT

Main

Clock Output

APB1

Prescaler

/1,2,4,8,16

/8

If (APB1 Prescaler=1) x 1

else

x 2

APB2

Prescaler

/1,2,4,8,16

If (APB2 Prescaler=1) x 1

else

x 2

ADC

Prescaler

/2,4,6,8

Clock

Enable (3 bits)

HCLK

to AHB bus, core

memory and DMA

to Cortex System mer

FCLK Cortex

Free running clock

Peripheral Clock

Enable (10 bits)

Peripheral Clock

Enable (2 bits)

Peripheral Clock

Enable (5bits)

Peripheral Clock

Enable (4 bit)

PCLK1

to APB1

peripherals

to TIM2,3

TIMXCLK

PCLK2

to APB2

peripherals

to TIM1,14,16,17

TIMXCLK

ADCCLK

to ADC

**Legend:**

HSE = high-speed external clock signal

HSI = high-speed internal clock signal

LSI = low-speed internal clock signal

MCO

HSI

LSICLK

LSI

HSI

Peripheral Clock

Enable

to TIM1

TIM.ADV

If (APB2 Prescaler!=1)

APB x 2

else if (AHB Prescale!=1)

AHB x 2

else

AHB CLK

12.

Users can configure the frequency of the AHB, high-speed APB (APB2) and low-speed APB (APB1) domains through multiple prescalers. AHB

With APB1, the maximum frequency of APB2 domain is 48MHz.

The RCC is supplied to the CPU system timer's (SysTick) external clock by dividing the AHB clock by 8. By SysTick

Control and status register settings, you can select the above clock or AHB clock as the SysTick clock. ADC clock by

Obtained after dividing the high-speed APB2 clock.

The timer clock frequency distribution is automatically set by the hardware according to the following two conditions:

1. If the corresponding APB prescaler coefficient is 1, the clock frequency of the timer is the same as the frequency of the APB bus.

2. Otherwise, the clock frequency of the timer is set to twice the frequency of the APB bus connected to it.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

37/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 60** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

FCLK is the free running clock of the CPU.

**5.2.1**

**HSE**

The high-speed external clock signal (HSE) is generated by the following two clock sources:

• HSE external crystal/ceramic resonator

• HSE user external clock

In order to reduce the distortion of the clock output and shorten the start-up settling time, the crystal/ceramic resonator and load capacitor must

Ground close to the oscillator pin. The load capacitance value must be adjusted according to the selected oscillator.

985606

(HiZ)

Hardware configuratiion

External clock

Crystal/Ceramic

resonators

Load capacitors

External source

C L1

C L2

OSC\_ IN

OSC\_OUT

OSC\_OUT

13.

**(HSE**

**)**

In this mode, an external clock must be provided. Its frequency can be up to 24MHz. Users can control

Use the HSEBYP and HSEON bits in the control register to select this mode. External clock signal (50% duty cycle

Wave, sine wave) must be connected to the OSC\_IN pin, while ensuring that the OSC\_OUT pin is floating.

**/**

**(HSE**

**)**

The external oscillator can provide a more accurate master clock for the system. Relevant hardware configuration can refer to Figure [13](https://translate.googleusercontent.com/translate_f#60) , further information can be

Refer to the electrical characteristics section of the data sheet.

The HSERDY bit in the clock control register RCC\_CR is used to indicate whether the high-speed external oscillator is stable. At startup

At this time, the clock is not released until this bit is set by hardware. If in the clock interrupt register RCC\_CIR

If an interrupt is allowed, a corresponding interrupt will be generated.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

38/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 61** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

The HSE crystal can be turned on and off by setting the HSEON bit in the RCC\_CR in the clock control register.

**5.2.2**

**HSI**

The HSI clock signal is generated by the internal HSI oscillator. The HSI oscillator can be used without any external components.

Provide the system clock. Its startup time is shorter than the HSE crystal oscillator. However, even after calibration, its clock frequency

The accuracy is still poor.

The manufacturing process determines that the oscillator frequency of different chips will be different, which is why the HSI clock frequency of each chip is

It has been calibrated to 1% (25 ◦ before leaving the factory)

C). When the system is reset, the factory calibration value is loaded into the clock control register

The HSICAL bit of the device.

The HSIRDY bit in the clock control register is used to indicate whether the HSI oscillator is stable. During clock start, until

This bit is set to '1' by hardware before the HSI oscillator output clock is released. The HSI oscillator can be controlled by the clock control register

The HSION bit to start and close.

If the HSE crystal oscillator fails, the HSI clock will be used as the backup clock source, refer to section [5.2.5](https://translate.googleusercontent.com/translate_f#61) .

**5.2.3**

**LSI**

The LSI oscillator acts as a low-power clock source, it can keep running in shutdown and standby mode, which is independent

The watchdog and automatic wake-up unit provide the clock. The LSI clock frequency is about 40KHz. Please refer to the data sheet for further information

Section on electrical characteristics.

The LSI oscillator can be turned on or off by the LSION bit in the control/status register (RCC\_CSR). In control/like

The LSIRDY bit in the status register (RCC\_CSR) indicates whether the low-speed internal oscillator is stable. During the startup phase, until

After this bit is set to '1' by hardware, the clock is released. If it is in the clock interrupt register (RCC\_CIR)

If enabled, an LSI interrupt request will be generated.

**5.2.4**

**(SYSCLK)**

After the system is reset, the HSI oscillator is selected as the system clock.

The switch from one clock source to another will only occur when the target clock source is ready. Being selected

When the clock source is not ready, the switching of the system clock will not occur. The switchover does not occur until the target clock source is ready.

The status bit in the clock configuration register (RCC\_CFGR) indicates which clock is ready and which clock is currently

Used as the system clock.

**5.2.5**

**(CSS)**

The clock security system can be activated by software. Once it is activated, the clock monitor will delay the start of the HSE oscillator

It is enabled afterwards and turned off after the HSE clock is turned off.

If the HSE clock fails, the HSE oscillator is automatically turned off and the clock failure event will be sent to the advanced timer TIM1

The brake input terminal, and generates a clock safety interrupt CSS, allowing the software to complete the rescue operation. This CSS breaks the connection to

The CPU's NMI is interrupted.

Note: Once the CSS is activated and the HSE clock fails, the CSS interrupt is generated and the NMI is also automatically generated. NMI will

It is executed continuously until the CSS interrupt pending bit is cleared. Therefore, it is necessary to set the clock interrupt in the NMI handler

The CSSC bit in the register (RCC\_CIR) is used to clear the CSS interrupt.

If the HSE oscillator is used directly or indirectly as the system clock, a clock failure will cause the system clock to automatically switch to HSI

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

39/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 62** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

Oscillator, and the external HSE oscillator is turned off.

**5.2.6**

If the independent watchdog has been started by hardware option or software, the LSI oscillator will be forced to be on

After the oscillator is stable, the clock is supplied to IWDG.

**5.2.7**

The microcontroller allows the output of clock signals to external MCO pins.

The corresponding GPIO port register must be configured for the corresponding function. The following 4 clock signals can be selected as the MCO clock:

• SYSCLK

• HSI

• HSE

• LSI

**– The** clock selection is controlled by the MCO[2:0] bits in the clock configuration register (RCC\_CFGR).

**5.3 RCC**

16. RCC

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

RCC\_CR

Clock control register

0x0000XX03

[Section 5.3.1](https://translate.googleusercontent.com/translate_f#62)

0x04

RCC\_CFGR

Clock configuration register

0x00000000

[Section 5.3.2](https://translate.googleusercontent.com/translate_f#64)

0x08

RCC\_CIR

Clock interrupt register

0x00000000

[Section 5.3.3](https://translate.googleusercontent.com/translate_f#66)

0x0C

RCC\_APB2RSTR

APB2 peripheral reset register

0x00000000

[Section 5.3.4](https://translate.googleusercontent.com/translate_f#68)

0x10

RCC\_APB1RSTR

APB1 peripheral reset register

0x00000000

[Section 5.3.5](https://translate.googleusercontent.com/translate_f#69)

0x14

RCC\_AHBENR

AHB peripheral clock enable register

0x00000014

[Section 5.3.6](https://translate.googleusercontent.com/translate_f#70)

0x18

RCC\_APB2ENR

APB2 peripheral clock enable register

0x00000000

[Section 5.3.7](https://translate.googleusercontent.com/translate_f#71)

0x1C

RCC\_APB1ENR

APB1 peripheral clock enable register

0x00000000

[Section 5.3.8](https://translate.googleusercontent.com/translate_f#72)

0x24

RCC\_CSR

Control status register

0x0C000000

[Section 5.3.9](https://translate.googleusercontent.com/translate_f#73)

0x40

RCC\_SYSCFG

System Configuration Register

0x00001400

[Section 5.3.10](https://translate.googleusercontent.com/translate_f#75)

**5.3.1**

**(RCC\_CR)**

Address offset: 0x00

Reset value: 0x0000 XX03

Access: No wait state, word, halfword and byte access

Reserved

CSSON

HSERDY

HSEBYP

HSEON

HSICAL

Reserved

HSIRDYHSION

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

r

r

r

r

r

r

rw

r

r

r

r

r

r

rw

r

rw

rw

rw

HSITEN

Reserved

rw

r

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

40/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 63** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 20

Reserved

Always read as 0.

19

CSSON

rw

0x00

CSSON: Clock security system en-

able)

Set or cleared by software to enable the clock monitor.

0: The clock monitor is off

1: If the external oscillator is ready, the clock monitor is turned on

18

HSEBYP

rw

0x00

HSEBYP: External high-speed clock bypass (External high-speed clock

bypass)

In debug mode, the external crystal oscillator is bypassed by software setting or clearing it.

This bit can only be written when the external oscillator is turned off.

0: The external oscillator is not bypassed

1: External external crystal oscillator is bypassed

17

HSERDY

r

0x00

HSERDY: External high-speed clock ready flag (External high-speed

clock ready flag)

Set by hardware to indicate that the external clock has stabilized.

0: external clock is not ready

1: External clock is ready

16

HSEON

rw

0x00

HSEON: External high-speed clock en-enable (External high-speed clock en-

able)

Set or cleared by software.

When entering standby and stop mode, this bit is cleared by hardware, when the external is turned off

bell. When an external clock is used or selected as the system clock, the

The bit cannot be cleared.

0: HSE oscillator is off

1: HSE oscillator is turned on

15: 14

Reserved

Always read as 0

13: 8

HSICAL

r

0xXX

HSICAL: Internal high-speed clock cal-

ibration)

At system startup, these bits are automatically initialized.

7: 3

Reserved

Always read as 0.

2

HSITEN

rw

0x00

HSITEN: Internal high-speed clock temperature calibration is enabled

Set or cleared by software.

0: Internal HSI clock does not automatically follow the temperature calibration

1: Internal HSI clock automatically follows temperature calibration

1

HSIRDY

r

0x01

HSIRDY: Internal high-speed clock ready flag (Internal high-speed

clock ready flag)

Set by hardware to indicate that the internal 8MHz clock has stabilized. At HSION

After the bit is cleared, this bit requires 6 internal 8MHz clock cycles to be cleared.

0: The internal 8MHz clock is not ready

1: Internal 8MHz clock is ready

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

41/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 64** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

HSION

rw

0x01

HSION: Internal high-speed clock en-enable (Internal high-speed clock en-

able)

Set or cleared by software.

Occurs when returning from standby and shutdown mode or as an external clock used as system clock

In case of failure, this bit is set to '1' by hardware to start the internal 8MHz oscillator.

When the internal 8MHz clock is directly or indirectly used or selected to be used

When the system clock is used, this bit cannot be cleared.

0: The internal 8MHz clock is off

1: The internal 8MHz clock is turned on

**5.3.2**

**(RCC\_CFGR)**

Address offset: 0x04

Reset value: 0x0000 0000

Access: No wait state, word, halfword and byte access

Only when the access occurs during a clock switch, 1 or 2 wait cycles are inserted.

Reserved

Reserved

Reserved

MCO

PPRE2

HPRE

SWS

PPRE1

SW

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

11

12

13

14

15

10

rw

r

r

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 27

Reserved

Always read as 0

26: 24

MCO

rw

0x00

MCO: Microcontroller clock output

Set or cleared by software.

00x: No clock output;

010: LSI clock output;

011: Reserved

100: System clock (SYSCLK) output;

101: HSI clock output;

110: HSE clock output;

note:

1. This clock output may be truncated when starting and switching the MCO clock source.

2. When the system clock is output to the MCO pin, make sure that the output clock frequency does not exceed

Over 50MHz (highest frequency of IO port)

23:14

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

42/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 65** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

13: 11

PPRE2

rw

0x00

PPRE2: High-speed APB prescaler (APB2) (APB high-speed

prescaler(APB2))

Set by software to '1' or clear '0' to control high-speed APB2 clock (PCLK2)

The prescaler coefficient.

0xx: HCLK is not divided

100: HCLK divided by 2

101: HCLK divided by 4

110: HCLK divided by 8

111: HCLK divided by 16

10: 8

PPRE1

rw

0x00

PPRE1: Low-speed APB prescaler (APB1) (APB low-speed

prescaler(APB1))0

Set by software to '1' or clear '0' to control low-speed APB1 clock (PCLK1)

The prescaler coefficient.

0xx: HCLK is not divided

100: HCLK divided by 2

101: HCLK divided by 4

110: HCLK divided by 8

111: HCLK divided by 16

7: 4

HPRE

rw

0x00

HPRE: AHB Prescaler

Set by software to '1' or clear '0' to control the prescaler of AHB clock.

0xxx: SYSCLK is not divided

1000: SYSCLK divided by 2

1001: SYSCLK divided by 4

1010: SYSCLK divided by 8

1011: SYSCLK divided by 16

1100: SYSCLK divided by 64

1101: SYSCLK divided by 128

1110: SYSCLK divided by 256

1111: SYSCLK divided by 512

note:

1. When the prescaler coefficient of the AHB clock is greater than 1, the prefetch buffer must be turned on. detailed

See the section on reading flash memory.

3: 2

SWS

r

0x00

SWS: System clock switch status

Set by hardware to '1' or clear '0' to indicate which clock source is used as the system

bell.

00: HSI divided by 6 is used as the system clock;

01: HSE is used as the system clock;

10: HSI as the system clock, forced to 1;

11: The LSI output is used as the system clock.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

43/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 66** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1: 0

SW

rw

0x00

SW: System clock switch

Set by software to '1' or clear '0' to select the system clock source.

When returning from stop or standby mode or directly or indirectly as the system clock

When the HSE of a fault occurs, it is forcedly selected by the hardware

00: HSI divided by 6 is used as the system clock;

01: HSE is used as the system clock;

10: HSI as the system clock, forced to 1;

11: LSI is used as the system clock.

**5.3.3**

**(RCC\_CIR)**

Address offset: 0x08

Reset value: 0x0000 0000

Access: No wait cycles, word, halfword and byte access

Reserved

CSSC

Reserved

HSE

RDYC

HSI

RDYC

LSI

RDYC

Reserved

HSE

RDYIE

HSI

RDYIE

LSI

RDYIE

CSSF

Reserved

HSE

RDYF

HSI

RDYF

LSI

RDYF

rc\_w1

rc\_w1

rc\_w1

rc\_w1

r

r

r

rw

rw

rw

r

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Res.

Res.

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 24

Reserved

Always read as 0

twenty three

CSSC

rc\_w1

0x00

CSSC: Clear the clock security system in (Clock security system in-

terrupt clear)

Set by software to clear the CSSF safety system interrupt flag CSSF.

0: No effect

1: Clear the CSSF safety system interrupt flag

22: 20

Reserved

Always read as 0

19

HSERDYC

rc\_w1

0x00

HSERDYC: Clear HSE ready interrupt (HSE ready interrupt

clear)

Set by software to '1' to clear the HSE ready interrupt flag bit HSERDYF.

0: No effect

1: Clear the HSE ready interrupt flag bit HSERDYF

18

HSIRDYC

rc\_w1

0x00

HSIRDYC: Clear HSI ready interrupt clear (HSI ready interrupt clear)

Set by software to clear HSI ready interrupt flag HSIRDYF.

0: No effect

1: Clear the HSI ready interrupt flag HSIRDYF

17

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

44/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 67** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

16

LSIRDYC

rc\_w1

0x00

LSIRDYC: Clear LSI ready interrupt clear

Set to '1' by software to clear the LSI ready interrupt flag LSIRDYF.

0: No effect

1: Clear the LSI ready interrupt flag bit LSIRDYF

15: 12

Reserved

Always read as 0

11

HSERDYIE

rw

0x00

HSERDYIE: HSE ready interrupt en-

able)

Set by software to '1' or clear '0' to enable or disable external oscillator ready

Break.

0: HSE ready interrupt is closed

1: HSE ready interrupt enable

10

HSIRDYIE

rw

0x00

HSIRDYIE: HSI ready interrupt enable (HSI ready interrupt enable)

Set by software to '1' or clear '0' to enable or disable the internal 8MHz oscillator

Ready interrupt.

0: HSI ready interrupt is closed

1: HSI ready interrupt enable

9

Reserved

Always read as 0

8

LSIRDYIE

rw

0x00

LSIRDYIE: LSI ready interrupt enable (LSI ready interrupt enable)

Set by software to '1' or clear '0' to enable or disable the internal 40KHz oscillator

Ready interrupt.

0: LSI ready interrupt is turned off

1: LSI ready interrupt enable

7

CSSF

r

0x00

CSSF: Clock security system interrupt flag (Clock security system in-

terrupt flag)

When the external oscillator clock fails, it is set to '1' by hardware.

Cleared by software by setting the '1' CSSC bit.

0: No safety system interruption due to HSE clock failure

1: HSE clock failure caused the clock security system to be interrupted

6: 4

Reserved

Always read as 0

3

HSERDYF

r

0x00

HSERDYF: HSE ready interrupt flag (HSE ready interrupt flag)

When the external low-speed clock is ready and the HSERDYIE bit is set to '1', the hard

Set to '1'.

Cleared by software by setting the HSERDYC bit to '1'.

0: No clock ready interrupt generated by external oscillator

1: External oscillator causes clock ready interrupt

2

HSIRDYF

r

0x00

HSIRDYF: HSI ready interrupt flag (HSI ready interrupt flag)

When the internal high-speed clock is ready and the HSIRDYIE bit is set to '1', the hardware

Set to '1'.

Cleared by software by setting the '1' HSIRDYC bit.

0: No clock ready interrupt generated by internal HSI oscillator

1: Internal HSI oscillator causes clock ready interrupt

1

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

45/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 68** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

LSIRDYF

r

0x00

LSIRDYF: LSI ready interrupt flag (LSI ready interrupt flag)

When the internal low-speed clock is ready and the LSIRDYIE bit is set to '1', the hardware

Set to '1'.

Cleared by software by setting the '1' LSIRDYC bit.

0: No clock ready interrupt generated by internal 40KHz oscillator;

1: The internal 40KHz oscillator causes the clock ready interrupt.

**5.3.4**

**APB2**

**(RCC\_APB2RSTR)**

Address offset: 0x0C

Reset value: 0x0000 0000

Access: No wait cycles, word, halfword and byte access

Reserved

Reserved

Reserved

SYSCFG

ADC1

TIM1 Res.

Reserved

DBGMCU

TIM17 TIM16 TIM14

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 23

Reserved

Always read as 0

twenty two

DBGMCU

rw

0x00

DBGMCU: DBGMCU reset

21: 19

Reserved

Always read as 0

18

TIM17

rw

0x00

TIM17: TIM17 timer reset (TIM17 timer reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset TIM17 timer

17

TIM16

rw

0x00

TIM16: TIM16 timer reset (TIM16 timer reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset TIM16 timer

16

TIM14

rw

0x00

TIM14: TIM14 timer reset (TIM14 timer reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset TIM14 timer

15: 12

Reserved

Always read as 0

11

TIM1

rw

0x00

TIM1: TIM1 timer reset (TIM1 timer reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset TIM1 timer

10

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

46/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 69** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9

ADC1

rw

0x00

ADC1: ADC1 interface reset (ADC1 interface reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset ADC1 interface

8: 1

Reserved

Always read as 0

0

SYSCFG

rw

0x00

SYSCFG: system configuration register reset (System Configuration reg-

ister reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset SYSCFG

**5.3.5**

**APB1**

**(RCC\_APB1RSTR)**

Address offset: 0x10

Reset value: 0x0000 0000

Access: No wait cycles, word, halfword and byte access

rw

rw

rw

rw

rw

rw

rw

PWR

Reserved

I2C1

Res.

Res.

Reserved

Reserved

TIM3 TIM2

WWDG

SPI2

Reserved

Reserved

UART2

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 29

Reserved

Always read as 0

28

PWR

rw

0x00

PWR: Power interface reset

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset power interface

27: 22

Reserved

Always read as 0

twenty one

I2C1

rw

0x00

I2C1: I2C1 reset (I2C1 reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset I2C1

20: 18

Reserved

Always read as 0

17

UART2

rw

0x00

UART2: UART2 reset (UART2 reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset UART2

16: 15

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

47/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 70** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

14

SPI2

rw

0x00

SPI2: SPI2 reset (SPI2 reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset SPI2

13: 12

Reserved

Always read as 0

11

WWDG

rw

0x00

WWDG: Window watchdog reset

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset window watchdog

10: 2

Reserved

Always read as 0

1

TIM3

rw

0x00

TIM3: Timer3 reset (Timer3 reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset TIM3 timer

0

TIM2

rw

0x00

TIM2: Timer2 reset (Timer2 reset)

Set to '1' or cleared to '0' by software.

0: No effect

1: Reset TIM2 timer

**5.3.6**

**AHB**

**(RCC\_AHBENR)**

Address offset: 0x14

Reset value: 0x0000 0014

Access: No wait cycles, word, halfword and byte access

Reserved

DMA

Res.

SRAM

Res.

FLITF

Reserved

Res.

GPIOA

GPIOB

GPIOC

GPIOD

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 21

Reserved

Always read as 0

20

GPIOD

rw

0x00

GPIOD: GPIOD clock enable (GPIOD clock enable)

0: GPIOD clock is off

1: GPIOD clock is on

19

GPIOC

rw

0x00

GPIOC: GPIOC clock enable (GPIOC clock enable)

0: GPIOC clock is off

1: GPIOC clock is on

18

GPIOB

rw

0x00

GPIOB: GPIOB clock enable (GPIOB clock enable)

0: GPIOB clock is off

1: GPIOB clock is on

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

48/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 71** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

17

GPIOA

rw

0x00

GPIOA: GPIOA clock enable (GPIOA clock enable)

0: GPIOA clock is off

1: GPIOA clock is on

16: 5

Reserved

Always read as 0

4

FLITF

rw

0x01

FLITF: Flash interface circuit clock enable (FLITF clock enable)

Set by software to '1' or clear '0' to enable or disable the flash memory interface during sleep mode

Circuit clock.

0: Flash memory interface circuit clock is off during sleep mode

1: The flash memory interface circuit clock is turned on during sleep mode

3

Reserved

Always read as 0

2

SRAM

rw

0x01

SRAM: SRAM interface clock enable

Set by software to '1' or clear '0' to turn SRAM on or off during sleep mode

clock.

0: SRAM clock is off during sleep mode

1: SRAM clock is turned on during sleep mode

1

Reserved

Always read as 0

0

DMA

rw

0x00

DMA: DMA clock enable

Set to '1' or cleared to '0' by software.

0: DMA clock is off

1: DMA clock is on

**5.3.7**

**APB2**

**(RCC\_APB2ENR)**

Address offset: 0x18

Reset value: 0x0000 0000

Access: No wait cycles, word, halfword and byte access

Note: When the peripheral clock is not started, the software cannot read the value of the peripheral register

Reserved

Reserved

Reserved

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

TIM1

ADC1

SYSCFG

TIM17 TIM16 TIM14

rw

rw

rw

Reserved

DBGMCU

rw

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 23

Reserved

Always read as 0

twenty two

DBGMCU

rw

0x00

DBGMCU enable

21: 19

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

49/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 72** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

18

TIM17

rw

0x00

TIM17: TIM17 timer enable

Set to '1' or cleared to '0' by software.

0: TIM17 timer clock is off

1: TIM17 timer clock is on

17

TIM16

rw

0x00

TIM16: TIM16 timer enable

Set to '1' or cleared to '0' by software.

0: TIM16 timer clock is off

1: TIM16 timer clock is on

16

TIM14

rw

0x00

TIM14: TIM14 timer enable (TIM14 timer enable)

Set to '1' or cleared to '0' by software.

0: TIM14 timer clock is off par 1: TIM14 timer clock is on

15:12

Reserved

Always read as 0

11

TIM1

rw

0x00

TIM1: TIM1 Timer clock enable

Set to '1' or cleared to '0' by software.

0: TIM1 timer clock is off

1: TIM1 timer clock is on

10

Reserved

Always read as 0

9

ADC1

rw

0x00

ADC1: ADC1 interface clock enable (ADC1 interface clock enable)

Set to '1' or cleared to '0' by software.

0: ADC1 interface clock is off

1: ADC1 interface clock is on

8: 1

Reserved

Always read as 0

0

SYSCFG

rw

0x00

SYSCFGEN: System configuration register clock enable (System configu-

ration register enable)

Set to '1' or cleared to '0' by software.

0: system configuration register clock is off

1: The system configuration register clock is turned on

**5.3.8**

**APB1**

**(RCC\_APB1ENR)**

Address offset: 0x1C

Reset value: 0x0000 0000

Access: No wait cycles, word, halfword and byte access

Note: When the peripheral clock is not started, the software cannot read the value of the peripheral register, the returned value is always 0x0

PWR

Res.

Res.

rw

rw

rw

rw

rw

rw

rw

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

Reserved

I2C1

UART2

TIM2

TIM3

WWDG

SPI2

Reserved

Reserved

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

50/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 73** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 29

Reserved

Always read as 0

28

PWR

rw

0x00

PWR: Power interface clock enable

Set to '1' or cleared to '0' by software.

0: power interface clock is off

1: The power interface clock is turned on

27: 22

Reserved

Always read as 0

twenty one

I2C1

rw

0x00

I2C1: I2C1 clock enable (I2C1 clock enable)

Set to '1' or cleared to '0' by software.

0: I2C1 clock is off

1: I2C1 clock is on

20:18

Reserved

Always read as 0

17

UART2

rw

0x00

UART2: UART2 clock enable

Set to '1' or cleared to '0' by software.

0: UART2 clock is off

1: UART2 clock is on

16:15

Reserved

Always read as 0

14

SPI2

rw

0x00

SPI2: SPI2 clock enable

Set to '1' or cleared to '0' by software.

0: SPI2 clock is off

1: SPI2 clock is on

13:12

Reserved

Always read as 0

11

WWDG

rw

0x00

WWDG: Window watchdog clock en-

able)

Set to '1' or cleared to '0' by software.

0: The window watchdog clock is closed

1: The window watchdog clock is on

10: 2

Reserved

Always read as 0

1

TIM3

rw

0x00

TIM3: Timer3 clock enable

Set to '1' or cleared to '0' by software.

0: Timer 3 clock is off

1: Timer 3 clock is on

0

TIM2

rw

0x00

TIM2: Timer2 clock enable

Set to '1' or cleared to '0' by software.

0: Timer 2 clock is off

1: Timer 2 clock is on

**5.3.9**

**(RCC\_CSR)**

Address offset: 0x24

Reset value: 0xXC00 0000

Access: 0 ~ 3 wait cycles, word, halfword and byte access

When accessing this register continuously, a wait state will be inserted.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

51/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 74** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

Except that the reset flag is cleared by system reset, the reset flag can only be cleared by power reset.

SFT

RSTF

WWDG

RSTF

Reserved

Reserved

Res.

LSIRDY

IWDG

RSTF

LSION

PIN

RSTF

Res.

POR

RSTF

RMVF

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

r

r

r

r

r

r

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31

Reserved

Always read as 0

30

WWDGRSTF

r

0x0x

WDGRSTF: Window watchdog re-sign

set flag)

Set by hardware when window watchdog reset occurs.

Cleared by software by writing RMVF bit.

0: No window watchdog reset occurs

1: A window watchdog reset occurred

29

IWDGRSTF

r

0x0x

IWDGRSTF: Independent watchdog reset flag (Independent watchdog

reset flag)

Set by hardware when the independent watchdog reset occurs in the VDD area.

Cleared by software by writing RMVF bit.

0: No independent watchdog reset occurs

1: An independent watchdog reset occurs

28

SFTRSTF

r

0x0x

SFTRSTF: Software reset flag

Set by hardware when a software reset occurs.

Cleared by software by writing RMVF bit.

0: No software reset occurs

1: A software reset has occurred

27

PORRSTF

r

0x01

PORRSTF: POR/PDR reset flag

Set to '1' by hardware when power-on/power-down reset occurs.

Cleared by software by writing RMVF bit.

0: No power-on/power-off reset occurs

1: A power-on/power-down reset occurs

26

PINRSTF

r

0x01

PINRSTF: NRST pin reset flag (PIN reset flag)

Set by hardware when the NRST pin reset occurs.

Cleared by software by writing RMVF bit.

0: No NRST pin reset occurs

1: NRST pin reset occurs

25

Reserved

Always read as 0

twenty four

RMVF

rw

0x00

RMVF: Remove reset flag

Set by software to clear the reset flag.

0: No effect

1: Clear the reset flag

23: 2

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

52/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 75** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

LSIRDY

r

0x00

LSIRDY: Internal low-speed oscillator ready (Internal low-speed oscillator

ready)

Set by hardware to '1' or clear '0' to indicate whether the internal 40KHz oscillator

thread.

After LSION is cleared, after 3 cycles of the internal 40KHz oscillator

LSIRDY is cleared.

0: The internal 40KHz oscillator clock is not ready

1: The internal 40KHz oscillator clock is ready

0

LSION

rw

0x00

LSION: Internal low-speed oscillator enabled (Internal low-speed oscillator

enable)

Set to '1' or cleared to '0' by software.

0: The internal 40KHz oscillator is off

1: The internal 40KHz oscillator is turned on

**5.3.10**

**(RCC\_SYSCFG)**

Address offset: 0x40

Reset value: 0x0000 1400

Access: 0 ~ 3 wait cycles, word, halfword and byte access

Except that the reset flag is cleared by system reset, the reset flag can only be cleared by power reset.

OSC.ITRIM

OSC.

LPFEN

Res.

Res.

Reserved

Reserved

OSC.RTRIM

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 15

Reserved

Always read as 0

14

OSC\_LPFEN

rw

0x00

OSC\_LPFEN: External crystal low-pass filter enable

0: prohibited

1: enable

13

Reserved

Always read as 0

12: 11

OSC\_ITRIM

rw

0x01

OSC\_ITRIM: Calibration value of external crystal drive current

If the crystal is abnormal, the drive current can be adjusted to adapt the crystal

00: 2mA

01: 4mA

10: 6mA

11: 8mA

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

53/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 76** |

UM\_MM32F003\_q\_Ver1.19

(RCC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

10: 8

OSC\_RTRIM

rw

0x04

OSC\_RTRIM: Calibration value of external crystal feedback resistance

If the crystal is abnormal, the drive current can be adjusted to adapt the crystal

000: 100KΩ

001: 200KΩ

010: 500KΩ

011: 700KΩ

100: 1MΩ

101: 2MΩ

110: 4MΩ

111: 8MΩ

7: 0

Reserved

Always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

54/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 77** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

6

Use **I/O (GPIO)**

General function I/O (GPIO)

**6.1 GPIO**

Each GPIO port has two 32-bit configuration registers (GPIOx\_CRL, GPIOx\_CRH), and two 32-bit data registers

Registers (GPIOx\_IDR and GPIOx\_ODR), one 32-bit bit/reset register (GPIOx\_BSRR), one

16-bit reset register (GPIOx\_BRR), a 32-bit lock register (GPIOx\_LCKR) and two alternate functions

Select the registers (GPIOx\_AFRH) and (GPIOx\_AFRL).

Each bit of the GPIO port can be individually configured into multiple modes by software.

• Input float

• Input pull-up

• Input drop-down

• Analog input

• Open drain output

• Push-pull output

• Push-pull multiplexing function

• Open drain multiplexing function

Each I/O port can be freely programmed, however, I/O port registers must be accessed according to 32-bit words (half words or words are not allowed

Section visit).

The GPIOx\_BSRR and GPIOx\_BRR registers allow independent access to read/change any GPIO register;

This way, there is no danger in generating IRQ between read change accesses.

The following figure shows the basic structure of an I/O port bit.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

55/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 78** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

146870

V DD

P-MOS

䗃ࠪ

᧗ࡦ

㠣⡷кཆ Toruo

⁑ᤏ䗃Check

༽⭘࣏㜭䗃Check

Don't

Tortoise

Pour out

䈫 **/** ߉

ᶕ㠚⡷кཆ懮

༽⭘࣏㜭䗃ࠪ

䗃ࠪ傡ࣘಘ

V SS

V SS

V DD

V DD

V SS

ᔰ **/** Check

ս

To

**/**

␵

ᇔᇴ

ᆈಘ

Ya

ࠪᮠ

ᦞᇴ

ᆈ

ಘ

Ya

Check

ᦞ

ᇴ

ᆈಘ

ᯭᇶ⢩䀖ਁಘ

ᔰ **/** Check

ᔰ **/** Check

N-MOS

؍ᣔ

Ҽᶱ㇑

؍ᣔ

Ҽᶱ㇑

I/ Oᕅ㝊

᧘ᥭǃ ᔰ┿

ᡆCheck

14. I/O

17.

Configuration mode

**CNF1**

**CNF0**

**MODE1**

**MODE0**

**PxODR** register

General output

Push-Pull

0

0

01

0 or 1

Open-Drain

1

0 or 1

Multiplex function output

Push-Pull

1

0

Do not use

Open-Drain

1

Do not use

enter

Analog input

0

0

00

Do not use

Floating input

1

Do not use

Drop-down input

1

0

0

Pull-up input

1

18.

**MODE[1** : **0]**

significance

00

enter

01

Output

**6.1.1**

Use **I/O (GPIO)**

During reset and immediately after reset, the multiplexing function is not turned on, and the I/O port is configured to float input mode (CNFx[1:0] = 01,

MODEx[1:0] = 00).

After reset, the SWD pin is put into input pull-up or pull-down mode:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

56/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 79** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

• PA14: SWCLK is put in pull-down mode

• PA13: SWDIO is placed in pull-up mode

When configured as an output, the value written to the output data register (GPIOx\_ODR) is output to the corresponding I/O pin. can

Use the output driver in push-pull mode or open-drain mode (when output 0, only the N-MOS is turned on).

The input data register (GPIOx\_IDR) captures the data on the I/O pin every AHB clock cycle.

All GPIO pins have an internal weak pull-up and weak pull-down. When configured as inputs, they can be activated or

disconnect.

**6.1.2**

When programming individual bits of GPIOx\_ODR, software does not need to disable interrupts:

In a single AHB write operation, only one or more bits can be changed.

This is done by writing to the desired bit in the'set/reset register' (GPIOx\_BSRR, reset is GPIOx\_BRR)

To achieve "1", the unselected bits will not be changed.

**6.1.3**

**/**

All ports have external interrupt capability. In order to use an external interrupt line, the port must be configured in input mode. More off

For information on external interrupts, refer to [Section](https://translate.googleusercontent.com/translate_f#85) 6.3: External Interrupt/Event Controller (EXTI).

**6.1.4**

use

The port bit configuration register must be programmed before using the default alternate function.

• For multiplexed input functions, the port must be configured in input mode (floating, pull-up or pull-down) and the input pin must be

External drive.

Note: The input pins of the multiplexing function can also be simulated by software. This simulation can be realized by programming the GPIO controller. this

When, the port should be set to the alternate function output mode. Obviously, at this time, the corresponding pin is no longer driven externally, but through GPIO

The controller is driven by software.

• For the multiplexed output function, the port must be configured in the multiplexed function output mode (push-pull or open-drain).

• For the bidirectional multiplexing function, the port bit must be configured with the multiplexing function output mode (push-pull or open-drain). At this time, the input drive

The device is configured for floating input mode.

If the port is configured for multiplexed output function, the pin is disconnected from the output register and connected to the output signal of the on-chip peripheral

Pick up. If the software configures a GPIO pin as an alternate output function, but the peripheral is not activated, its output will not

determine.

**6.1.5**

**I/O**

use

In order to optimize the number of peripheral I/O functions packaged in different devices, some multiplexed functions can be re-mapped to their

Some of his feet. This can be done by software configuring the corresponding register (refer to the AFR register description).

At this time, the multiplexed functions are no longer mapped to their original pins.

**6.1.6**

**GPIO**

The locking mechanism allows the IO configuration to be frozen. When the LOCK procedure is executed on a port bit, the next reset

Previously, the configuration of port bits could no longer be changed.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

57/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 80** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

**6.1.7**

When the I/O port is configured as an input:

• The output buffer is disabled

• Schmitt trigger input is activated

• Depending on the input configuration (pull-up, pull-down or floating), weak pull-up and pull-down resistors are connected

• The data appearing on the I/O pin is sampled into the input data register every AHB clock

• Read access to input data registers to get I/O status

The following figure shows the input configuration of I/O port bits:

250155

V DD

Don't

Tortoise

Pour out

䈫 **/** ߉

䗃ࠪ傡ࣘಘ

V SS

V DD

V SS

ᔰ

ս

To

**/**

␵

To

ᇴ

ᆈಘ

Ya

ࠪᮠ

ᦞᇴ

ᆈ

ಘ

Ya

Check

ᮠ

ᦞ

ᇴ

ᆈಘ

ᯭᇶ⢩䀖ਁಘ

ᔰ **/** Check

ᔰ **/** Check

؍ᣔ

Ҽᶱ㇑

؍ᣔ

Ҽᶱ㇑

I **/** Oᕅ㝊

15.

//

**6.1.8**

When the I/O port is configured as an output:

• The output buffer is activated

**–** Open-drain mode: '0' on the output register activates N-MOS, while '1' on the output register will port

Put into high impedance state (P-MOS is never activated)

**–** Push-pull mode: '0' on the output register activates N-MOS, while '1' on the output register will activate

P-MOS

• Schmitt input is activated

• Weak pull-up and pull-down resistors are prohibited

• The data appearing on the I/O pin is sampled into the input data register every AHB clock

• In open-drain mode, access to the input data register can get the I/O status

• In push-pull mode, read access to the output data register can get the last written value.

The following figure shows the output configuration of the I/O port bits:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

58/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 81** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

989530

PrMOS

䗃ࠪ

᧗ࡦ

Don't

Tortoise

Pour out

䈫 **/** ߉

䗃ࠪ傡ࣘಘ

V SS

V DD

V DD

V SS

on

ս

To

**/**

␵

To

ᇴ

ᆈಘ

Ya

ࠪᮠ

ᦞᇴ

ᆈ

ಘ

Ya

Check

ᦞᇴ

ᆈಘ

ᯭᇶ⢩䀖ਁಘ

NrMOS

؍ᣔ

Ҽᶱ㇑

؍ᣔ

Ҽᶱ㇑

I **/** Oᕅ㝊

᧘ᥭᡆᔰ┿

16.

**6.1.9**

use

When the I/O port is configured for alternate function:

• In open-drain or push-pull configuration, the output buffer is opened

• Signal driven output buffer with built-in peripherals (multiplexed function output)

• Schmitt trigger input is activated

• Weak pull-up and pull-down resistors are prohibited

• In every AHB clock cycle, the data appearing on the I/O pin is sampled into the input data register

• In open drain mode, I/O port status can be obtained when reading input data register

• In push-pull mode, the last written value can be obtained when reading the output data register

The following figure shows the configuration of the I/O port for multiplexing function. See AFIO register description for details.

A set of alternate function I/O registers allows the user to remap some alternate functions to different pins.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

59/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 82** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

240425

PrMOS

䗃ࠪ

᧗ࡦ

Sorrowful

༽⭘࣏㜭䗃Check

Don't

Tortoise

Pour out

䈫 **/** ߉

ᶕ㠚⡷кཆ懮

༽⭘࣏㜭䗃ࠪ

䗃ࠪ傡ࣘಘ

V SS

V DD

V DD

V SS

ᔰ

ս

To

**/**

␵

To

ᇴ

ᆈಘ

Ya

ࠪ

ᮠ

ᦞ

ᇴ

ᆈಘ

Ya

Check

ᦞ

ᇴ

ᆈಘ

ᯭᇶ⢩ 䀖ਁಘ

NrMOS

؍ᣔ

Ҽᶱ㇑

؍ᣔ

Ҽᶱ㇑

I/Oᕅ㝊

᧘ᥭᡆᔰ┿

17.

use

**6.1.10**

When the I/O port is configured as an analog input configuration:

• Output buffer disabled

• The Schmitt trigger input is disabled, achieving zero consumption on each analog I/O pin. Schmitt trigger output value is forced

Is '0'

• Weak pull-up and pull-down resistors are prohibited

• The value is '0' when reading the input data register

The following figure shows the high-impedance input configuration of the I/O port bits:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

60/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 83** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

272001

Sorrowful

⁑ᤏ䗃Check

Don't

Tortoise

Pour out

䈫 **/** ߉

ᶕ㠚⡷кཆ懮

䗃ࠪ傡ࣘಘ

V DD

V SS

Check

ս

To

**/**

␵

ᇔᇴ

ᆈಘ

Ya

ࠪᮠ

ᦞ

ᇴ

ᆈಘ

Ya

Check

ᦞ

ᇴ

ᆈಘ

ᯭᇶ⢩䀖ਁಘ

؍ᣔ

Ҽᶱ㇑

؍ᣔ

Ҽᶱ㇑

I **/** Oᕅ㝊

0

18.

**6.1.11**

**GPIO**

The following table lists the pin configuration of each peripheral:

19.

TIM1

**TIM1** pin

Configuration

**GPIO** configuration

TIM1\_CHx

Input capture channel x

Floating input

Output comparison channel x

Push-pull multiplexed output

TIM1\_CHxN

Complementary output channel x

Push-pull multiplexed output

TIM1\_BKIN

Brake input

Floating input

TIM1\_ETR

External trigger clock input

Floating input

20.

With TIM2/3/14/16/17

**TIM2/3/14/16/17**

Pin

Configuration

**GPIO** configuration

TIM2/3/14/

16/17\_CHx

Input capture channel x

Floating input

Output comparison channel x

Push-pull multiplexed output

TIM2/3\_ETR

External trigger clock input x

Floating input

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

61/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 84** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

21. UART

**UART** pin

Configuration

**GPIO** configuration

UARTx\_TX

Serial port sending

Push-pull multiplexed output

UARTx\_RX

Serial port reception

Floating input or with pull-up input

UARTx\_RTS

Hardware flow control

Push-pull multiplexed output

UARTx\_CTS

Hardware flow control

Floating input or with pull-up input

22. SPI

**SPI** pin

Configuration

**GPIO** configuration

SPIx\_SCK

Main mode

Push-pull multiplexed output

Slave mode

Floating input

SPIx\_MOSI

Full-duplex mode/master mode

Push-pull multiplexed output

Full-duplex mode/slave mode

Floating input or with pull-up input

Simple bidirectional data line/master mode

Push-pull multiplexed output

Simple bidirectional data line/slave mode

Not used, can be used as general purpose I/O

SPIx\_MISO

Full-duplex mode/master mode

Floating input or with pull-up input

Full-duplex mode/slave mode

Push-pull multiplexed output

Simple bidirectional data line/master mode

Not used, can be used as general purpose I/O

Simple bidirectional data line/slave mode

Push-pull multiplexed output

SPIx\_NSS

Hardware master/slave mode

Floating input or with pull-up input or with pull-down input

Enter

Hardware master mode/NSS output enable

Push-pull multiplexed output

Software mode

Not used, can be used as general purpose I/O

23. I2C

**I2C** pin

Configuration

**GPIO** configuration

I2Cx\_SCL

I2C clock

Open-drain multiplexed output

I2Cx\_SDA

I2C data

Open-drain multiplexed output

24. ADC

**ADC** pin

**GPIO** configuration

ADC

Analog input

25.

I/O

Pin

Configuration

**GPIO** configuration

MCO

Clock output

Push-pull multiplexed output

EXTI input line

External interrupt input

Floating input or with pull-up input or pull-down input

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

62/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 85** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

**6.2**

With **I/O**

In order to optimize the number of peripherals, some multiplexed functions can be remapped to other pins. Setting the alternate function register (AFR)

Re-mapping pins. At this time, the multiplexing function no longer maps to their original allocation.

**6.2.1**

**OSC\_IN/OSC\_OUT**

**GPIO**

**PD0/PD1**

The external oscillator pin OSC\_IN/OSC\_OUT can be used as PD0/PD1 of GPIO.

Clock, and then by setting the alternate function register (AFR).

Note: External interrupt/event functions are not remapped.

**6.2.2**

**SWD**

use

Debug interface signals are mapped to GPIO ports, as shown in the table below

26.

Reuse function

**GPIO** port

SWDIO

PA13

SWCLK

PA14

In order to use more GPIOs during debugging, you can change by setting multiplex remapping and multiplex function registers

The above remapping configuration.

**6.3 GPIO**

27. GPIO

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

GPIOx\_CRL

Port configuration low register

0x44444444

[Section 6.3.1](https://translate.googleusercontent.com/translate_f#85)

0x04

GPIOx\_CRH

Port configuration high register

0x44444444

[Section 6.3.2](https://translate.googleusercontent.com/translate_f#86)

0x08

GPIOx\_IDR

Port input data register

0x0000XXXX

[Section 6.3.3](https://translate.googleusercontent.com/translate_f#87)

0x0C

GPIOx\_ODR

Port output data register

0x00000000

[Section 6.3.4](https://translate.googleusercontent.com/translate_f#87)

0x10

GPIOx\_BSRR

Port setting/clearing register

0x00000000

[Section 6.3.5](https://translate.googleusercontent.com/translate_f#88)

0x14

GPIOx\_BRR

Port bit clear register

0x00000000

[Section 6.3.6](https://translate.googleusercontent.com/translate_f#88)

0x18

GPIOx\_LCKR

Port configuration lock register

0x00000000

[Section 6.3.7](https://translate.googleusercontent.com/translate_f#89)

0x20

GPIOx\_AFRL

Port multiplex function low register

0x00000000

[Section 6.3.8](https://translate.googleusercontent.com/translate_f#90)

0x24

GPIOx\_AFRH

Port multiplexing function high register

0x00000000

[Section 6.3.9](https://translate.googleusercontent.com/translate_f#90)

**6.3.1**

**(GPIOx\_CRL)(x = A..D)**

Offset address: 0x00

Reset value: 0x4444 4444

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

63/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 86** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNF7

CNF5

CNF6

CNF4

CNF3

CNF2

CNF1

CNF0

MODE7

MODE5

MODE6

MODE4

MODE3

MODE1

MODE2

MODE0

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 30

27: 26

23: 22

19: 18

15: 14

11: 10

7: 6

3: 2

CNFy

rw

0x01

Port x configuration bits (0…7)

The software configures the corresponding I/O port through these bits, please refer to Table 17 Port Bits

Configuration table

In input mode (MODE = 00):

00: Analog input mode

01: floating input mode

10: Pull-up/pull-down input mode

11: Reserved

In output mode (MODE> 00):

00: Universal push-pull output mode

01: General open-drain output mode

10: Push-pull output mode with multiplexing function

11: Open-drain output mode with multiplexing function

29: 28

25: 24

21: 20

17: 16

13: 12

9: 8

5: 4

1: 0

MODEy

rw

0x00

Port x mode bits (y = 0…7) (Port x mode bits)

The software configures the corresponding I/O port through these bits, please refer to Table 17 Port Bits

Configuration table

00: Input mode (state after reset)

01: output mode

10: reserved

11: Reserved

**6.3.2**

**(GPIOx\_CRH)(x = A..D)**

Offset address: 0x04

Reset value: 0x4444 4444

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNF15

CNF13

CNF14

CNF12

CNF11

CNF10

CNF9

CNF8

MODE15

MODE13

MODE14

MODE12

MODE11

MODE9

MODE10

MODE8

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

64/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 87** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 30

27: 26

23: 22

19: 18

15: 14

11: 10

7: 6

3: 2

CNFy

rw

0x01

Port x configuration bits (8…15)

The software configures the corresponding I/O port through these bits, please refer to Table 17 Port Bits

Configuration table

In input mode (MODE = 00):

00: Analog input mode

01: floating input mode

10: Pull-up/pull-down input mode

11: Reserved

In output mode (MODE[1: 0]> 00):

00: Universal push-pull output mode

01: General open-drain output mode

10: Push-pull output mode with multiplexing function

11: Open-drain output mode with multiplexing function

29: 28

25: 24

21: 20

17: 16

13: 12

9: 8

5: 4

1: 0

MODEy

rw

0x00

Port x mode bits (y = 8…15) (Port x mode bits)

The software configures the corresponding I/O port through these bits, please refer to Table 17 Port Bits

Configuration table

00: Input mode (state after reset)

01: output mode

10: reserved

11: Reserved

**6.3.3**

**(GPIOx\_IDR)(x = A..D)**

Offset address: 0x08

Reset value: 0x0000 XXXX

IDR

Reserved

r

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

IDRy

r

0xXXXX

Port input data (y = 0..15) (Port input data)

These bits are read-only and can only be read in the form of words (16 bits), the value read is right

The status of the corresponding I/O port.

**6.3.4**

**(GPIOx\_ODR)(x = A..D)**

Offset address: 0x0C

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

65/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 88** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

Reset value: 0x0000 0000

ODR

Reserved

rw

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

ODRy

rw

0x00

Port output data (y = 0..15) (Port output data)

These bits are readable and can only be operated as words (16 bits).

Note: For GPIOx\_BSRR (x = A…E), each ODR can be entered separately

Line independent setting/clearing.

**6.3.5**

**/**

**(GPIOx\_BSRR)(x = A..D)**

Offset address: 0x10

Reset value: 0x0000 0000

BR15 BR14

BR11

BR12

BR13

BR10 BR9

BR1

BR2

BR3

BR4

BR5

BR6

BR7

BR8

BR0

BS15 BS14

BS11

BS12

BS13

BS10 BS9

BS1

BS2

BS3

BS4

BS5

BS6

BS7

BS8

BS0

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

BRy

w

0x00

Clear bit x of port x (y = 0…15) (Port x Reset bit y)

These bits can only be written and can only be manipulated as words (16 bits).

0: No effect on the corresponding ODRy bit

1: Clear the corresponding ODRy bit to 0

15: 0

BSy

w

0x00

Set port x bit y (y = 0..15) (Port x Set bit y)

These bits can only be written and can only be manipulated as words (16 bits).

0: The corresponding ODRy bit has no effect

1: Set the corresponding ODRy bit to 1

**6.3.6**

**(GPIOx\_BRR)(x = A..D)**

Offset address: 0x14

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

66/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 89** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

BR

Reserved

w

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0

15: 0

BRy

w

0x00

Clear bit x of port x (y = 0…15) (Port x Reset bit y)

These bits can only be written and can only be manipulated as words (16 bits).

0: The corresponding ODRy bit has no effect

1: Clear the corresponding ODRy bit to 0

**6.3.7**

**(GPIOx\_LCKR)(x = A..D)**

When bit 16 (LCKK) is set by executing the correct write sequence, this register is used to lock the port bit configuration. Bit[15:0]

Used to lock the configuration of the GPIO port. During the specified write operation, LCKP[15:0] cannot be changed. When responding

After the LOCK sequence is executed on the port bit, the configuration of the port bit cannot be changed until the next system reset.

Each lock bit locks the corresponding 4 bits in the control register (CRL, CRH).

Address offset: 0x18

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

LCK

LCKK

Reserved

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 17

Reserved

Always read as 0

16

LCKK

rw

0x00

Lock key

The given bit can be read out at any time, it can only be modified by the lock key write sequence.

0: Port configuration lock key activation

1: The port configuration lock key bit is activated, and GPIOx\_LCKR before the next system reset

Register is locked

Write sequence of key lock:

Write 1-> Write 0-> Write 1-> Read 0-> Read 1

The last read can be omitted, but it can be used to confirm that the lock key has been activated.

Note: The value of LCK cannot be changed while operating the write sequence of the key lock. Operation lock key write

Any errors entered in the sequence will not activate the key lock.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

67/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 90** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 0

LCKy

rw

0x00

Port x Lock bit y (y = 0…15) (Port x Lock bit y)

These bits are readable and writable but can only be written when the LCKK bit is 0.

0: Do not lock the port configuration

1: Lock port configuration

**6.3.8**

Use **(GPIOx\_AFRL) (x = A..D)**

Offset address: 0x20

Reset value: 0x0000 0000

rw

rw

rw

rw

rw

rw

rw

rw

AFR3

AFR2

AFR1

AFR0

AFR4

AFR5

AFR6

AFR7

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

AFRy

rw

0x0000

0000

Multiplexing function selection of bit y (y = 0…7) of port x

These bits can be written in software to configure the IO multiplexing function.

0000: AF0

0001: AF1

0010: AF2

0011: AF3

0100: AF4

0101: AF5

0110: AF6

0111: AF7

**6.3.9**

Use **(GPIOx\_AFRH) (x = A..D)**

Offset address: 0x24

Reset value: 0x0000 0000

rw

rw

rw

rw

rw

rw

rw

rw

AFR11

AFR10

AFR9

AFR8

AFR12

AFR13

AFR14

AFR15

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

68/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 91** |

UM\_MM32F003\_q\_Ver1.19

Use I/O (GPIO)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

AFRy

rw

0x0000

0000

Multiplexing function selection of bit y (y = 8…15) of port x

These bits can be written in software to configure the IO multiplexing function.

0000: AF0

0001: AF1

0010: AF2

0011: AF3

0100: AF4

0101: AF5

0110: AF6

0111: AF7

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

69/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 92** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

7

**(EXTI)**

Interrupts and events (EXTI)

**7.1**

feature

• Interrupts can be masked (except NMI)

• 16 programmable priority levels (4-bit interrupt priority used)

• Low-latency exception and interrupt handling

• Power management control

• Implementation of system control register

Nested Vectored Interrupt Controller (NVIC) is closely connected to the interface of the processor core, which can realize low-latency interrupt processing and

Handle late interrupts efficiently.

The nested vector interrupt controller manages interrupts including nuclear exceptions. For more abnormal and NVIC programming instructions, please refer to

Examine the CPU technical reference manual.

**7.1.1**

**(SysTick)**

The system tick calibration value is fixed at 9000, when the system tick clock is set to 3MHz (HCLK/8, HCLK = 24MHz),

Generate a 1ms time base.

**7.1.2**

The following table lists the vector table of this series of products.

28.

position

priority

Priority type

name

Explanation

address

-

-

-

Keep

0x0000\_0000

-3

fixed

Reset

Reset

0x0000\_0004

-2

fixed

NMI

Non-maskable interrupt

RCC Clock Security System (CSS) connection

To NMI vector

0x0000\_0008

-1

fixed

Hardware failure (HardFault)

All types of failures

0x0000\_000C

0

Configurable

Storage Management (MemManage)

Memory management

0x0000\_0010

1

Configurable

Bus fault (BusFault)

Prefetch finger failed, memory access failed

0x0000\_0014

2

Configurable

Fault Application (UsageFault)

Undefined instruction or illegal status

0x0000\_0018

-

-

-

Keep

0x0000\_001C

∼

0x0000\_002B

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

70/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 93** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

position

priority

Priority type

name

Explanation

address

3

Configurable

SVCall

System service call via SWI instruction

0x0000\_002C

4

Configurable

Debug Monitor (DebugMonitor)

Debug monitor

0x0000\_0030

-

-

-

Keep

0x0000\_0034

5

Configurable

PendSV

Suspendable system services

0x0000\_0038

6

Configurable

SysTick

System tick timer

0x0000\_003C

0

7

Configurable

WWDG/IWDG

Window/Independent Watchdog Timer Interrupt

0x0000\_0040

1

8

Configurable

PVD

Power supply voltage detection connected to EXTI16

(PVD) Interrupt

0x0000\_0044

2

9

-

-

Keep

0x0000\_0048

3

10

Configurable

Flash

Flash Global Interrupt

0x0000\_004C

4

11

Configurable

RCC

RCC global interrupt

0x0000\_0050

5

12

Configurable

EXTI0\_1

EXTI line [1:0] interrupted

0x0000\_0054

6

13

Configurable

EXTI2\_3

EXTI line [3:2] interrupted

0x0000\_0058

7

14

Configurable

EXTI4\_15

EXTI line [15:4] interrupted

0x0000\_005C

8

-

-

-

Keep

0x0000\_0060

9

16

Configurable

DMA1 channel 1

DMA1 channel 1 global interrupt

0x0000\_0064

10

17

Configurable

DMA1 channels 2, 3

DMA1 channel 2, 3 global interrupt

0x0000\_0068

11

18

Configurable

DMA1 channels 4, 5

DMA1 channel 4, 5 global interrupt

0x0000\_006C

12

19

Configurable

ADC

ADC interrupt

0x0000\_0070

13

20

Configurable

TIM1\_BRK\_UP\_TRG\_COM

TIM1 brake, update, trigger, COM

Interrupt

0x0000\_0074

14

twenty one

Configurable

TIM1\_CC

TIM1 capture compare interrupt

0x0000\_0078

15

twenty two

Configurable

TIM2

TIM2 global interrupt

0x0000\_007C

16

twenty three

Configurable

TIM3

TIM3 global interrupt

0x0000\_0080

17

-

-

-

Keep

0x0000\_0084

18

-

-

-

Keep

0x0000\_0088

19

26

Configurable

TIM14

TIM14 global interrupt

0x0000\_008C

20

-

-

-

Keep

0x0000\_0090

twenty one

28

Configurable

TIM16

TIM16 global interrupt

0x0000\_0094

twenty two

29

Configurable

TIM17

TIM17 global interrupt

0x0000\_0098

twenty three

30

Configurable

I2C1

I2C1 global interrupt

0x0000\_009C

twenty four

-

-

-

Keep

0x0000\_00A0

25

-

-

-

Keep

0x0000\_00A4

26

33

Configurable

SPI2

SPI2 global interrupt

0x0000\_00A8

27

-

-

-

Keep

0x0000\_00AC

28

35

Configurable

UART2

UART2 global interrupt

0x0000\_00B0

29

-

-

-

Keep

0x0000\_00B4

30

-

-

-

Keep

0x0000\_00B8

31

-

-

-

Keep

0x0000\_00BC

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

71/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 94** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

**7.2**

**/**

**(EXTI)**

External interrupt and event controller (EXTI) manages external and internal asynchronous events/interrupts and generates corresponding event requests to

CPU/interrupt controller and wake-up request to power management.

Edge detector capable of generating event/interrupt request. Each input line can be independently configured for input type (pulse or suspend) and

Corresponding trigger event (both rising edge or falling edge or both edges trigger). Each input line can be shielded independently. hang

The interrupt register holding the status line holds the request.

**7.2.1**

The main features of the EXTI controller are as follows:

• Each interrupt/event has an independent trigger and mask

• Each interrupt line has a dedicated status bit

• Support software interrupt/event request

• Detect an external signal whose pulse width is less than the APB2 clock width. Refer to the relevant parameters in the electrical characteristics section of the data sheet

number.

**7.2.2**

677630

AMBA APB bus

Peripheral interface

Interrupt

mask

register

Rending

request

register

Software

interrupt

event

register

Rising

trigger

selection

register

Falling

trigger

selection

register

Event

mask

register

Edge detect

circuit

Input

Line

Pulse

generator

To NVIC Interrupt

Controller

PCLK 2

19.

/

**7.2.3**

Can handle external or internal events to wake up the core (WFE). The wake-up event can be generated by the following configuration:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

72/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 95** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

• An interrupt is enabled in the control register of the peripheral, but not in the NVIC, and is also registered in the system control register of the CPU

The SEVONPEND bit is enabled in the device.

When the CPU recovers from WFE, it is necessary to clear the interrupt suspend bit of the corresponding peripheral and the peripheral NVIC interrupt channel suspend bit

(In the NVIC interrupt clear pending register).

• Configure an external or internal EXTI line as the event mode, when the CPU recovers from WFE, because the corresponding event line

The suspend bit is not set, it is not necessary to clear the interrupt suspend bit of the corresponding peripheral or the NVIC interrupt channel suspend bit.

To use external I/O port as wake-up event, please refer to the function description in the next section.

**7.2.4**

To generate an interrupt, you must first configure and enable the interrupt line. Set 2 trigger registers according to the required edge detection, same as

When writing '1' in the corresponding bit of the interrupt mask register, the interrupt request is allowed. When the expected edge occurs on the external interrupt line,

An interrupt request will be generated and the corresponding suspend bit will also be set to '1'. Write '1' in the corresponding bit of the suspend register

Clear the interrupt request.

If you need to generate an event, you must first configure and enable the event line. Set 2 triggers according to the required edge detection

At the same time, write '1' in the corresponding bit of the event mask register to allow the event request. When something needed happens on the event line

At the edge, an event request pulse will be generated and the corresponding pending bit will not be set to '1'.

By writing '1' in the software interrupt/event register, an interrupt/event request can also be generated by software.

Use the following procedure to configure multiple lines as interrupt sources:

• Configure interrupt line mask bit (EXTI\_IMR)

• Configure the trigger selection bits (EXTI\_RTSR and EXTI\_FTSR) of the selected interrupt line

• Configure the enable and mask bits of the NVIC interrupt channel corresponding to the external interrupt controller (EXTI) so that the

The request can be responded correctly

Through the following process, you can configure multiple lines as event sources:

• Configure the mask bit of the event line (EXTI\_EMR)

• Configure the trigger select bits of the event line (EXTI\_RTSR and EXTI\_FTSR)

**/**

Multiple lines can be configured as software interrupt/event lines. The following is the process of generating a software interrupt:

• Configure interrupt/event line mask bits (EXTI\_IMR, EXTI\_EMR)

• Set the request bit (EXTI\_SWIER) of the software interrupt register

**7.2.5**

**/**

The general-purpose I/O ports are connected to 16 external interrupt/event lines in the following diagram:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

73/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 96** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

268009

EXTI0

EXTI0[3:0] bits in SYSCFG \_EXTICR1 register

EXTI13

EXTI13[3:0] bits in SYSCFG \_EXTICR4 register

EXTI3

EXTI3[3:0] bits in SYSCFG \_EXTICR1 register

EXTI4

EXTI4[3:0] bits in SYSCFG \_EXTICR2 register

EXTI15

EXTI15[3:0] bits in SYSCFG \_EXTICR4 register

PA0

PB0

PC0

PD0

PA3

PB3

PC3

PD3

PA4

PB4

PC4

PD4

PA13

PB13

PC13

PD13

PA15

PB15

PC15

PD15

20.

With I/O

Note: The GPIO corresponding to the above figure may be different due to the actual chip package, and the actual chip shall prevail.

In addition, the connection of other external interrupt/event controllers is as follows:

• EXTI line 16 is connected to the PVD output

• EXTI line 24 is connected to IWDG interrupt

**7.3 EXTI**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

74/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 97** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

29. EXTI

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

EXTI\_IMR

Interrupt mask register

0x00000000

[Section 7.3.1](https://translate.googleusercontent.com/translate_f#97)

0x04

EXTI\_EMR

Event mask register

0x00000000

[Section 7.3.2](https://translate.googleusercontent.com/translate_f#97)

0x08

EXTI\_RTSR

Rising edge trigger selection register

0x00000000

[Section 7.3.3](https://translate.googleusercontent.com/translate_f#98)

0x0C

EXTI\_FTSR

Falling edge trigger selection register

0x00000000

[Section 7.3.4](https://translate.googleusercontent.com/translate_f#98)

0x10

EXTI\_SWIER

Software interrupt event register

0x00000000

[Section 7.3.5](https://translate.googleusercontent.com/translate_f#99)

0x14

EXTI\_PR

Software interrupt event registration

0x00000000

[Section 7.3.6](https://translate.googleusercontent.com/translate_f#100)

**7.3.1**

**(EXTI\_IMR)**

Offset address: 0x00

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

IMR16

Reserved

IMR15 IMR14 IMR13 IMR12 IMR11 IMR10 IMR9 IMR8 IMR7 IMR6 IMR5 IMR4 IMR3

IMR2 IMR1 IMR0

IMR24

rw

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 25

Reserved

Always read as 0.

twenty four

IMRx

rw

0x00

Interrupt Mask on line x

1 = Open interrupt request from line x

0 = mask interrupt request from line x

23: 17

Reserved

Always read as 0.

16: 0

IMRx

rw

0x00

Interrupt Mask on line x

1 = Open interrupt request from line x

0 = mask interrupt request from line x

**7.3.2**

**(EXTI\_EMR)**

Offset address: 0x04

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

EMR16

Reserved

EMR15 EMR14 EMR13 EMR12 EMR11 EMR10 EMR9 EMR8 EMR7 EMR6 EMR5 EMR4 EMR3

EMR2 EMR1 EMR0

EMR24

Reserved

rw

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

75/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 98** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 25

Reserved

Always read as 0.

twenty four

EMRx

rw

0x00

Event Mask on line x (Event Mask on line x)

1 = open event request from line x

0 = block event requests from line x

23: 17

Reserved

Always read as 0.

16: 0

EMRx

rw

0x00

Event Mask on line x (Event Mask on line x)

1 = open event request from line x

0 = block event requests from line x

**7.3.3**

**(EXTI\_RTSR)**

Offset address: 0x08

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

TR16

Reserved

TR15

TR14

TR13

TR12

TR11

TR10

TR9

TR8

TR7

TR6

TR5

TR4

TR3

TR2

TR1

TR0

TR24

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 25

Reserved

Always read as 0.

twenty four

TRx

rw

0x00

Rising trigger event con-bit on line x (Rising trigger event con-bit on line x

figuration bit of line x)

1 = Enable rising edge trigger on input line x (interrupts and events)

0 = Disable rising edge trigger on input line x (interrupts and events)

23: 17

Reserved

Always read as 0.

16: 0

TRx

rw

0x00

Rising trigger event con-bit on line x (Rising trigger event con-bit on line x

figuration bit of line x)

1 = Enable rising edge trigger on input line x (interrupts and events)

0 = Disable rising edge trigger on input line x (interrupts and events)

**7.3.4**

**(EXTI\_FTSR)**

Offset address: 0x0C

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

76/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 99** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

TR16

Reserved

TR15

TR14

TR13

TR12

TR11

TR10

TR9

TR8

TR7

TR6

TR5

TR4

TR3

TR2

TR1

TR0

TR24

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 25

Reserved

Always read as 0.

twenty four

TRx

rw

0x00

Falling trigger event con-bit on line x (Falling trigger event con-bit

figuration bit of line x)

1 = Enable falling edge trigger on input line x (interrupts and events)

0 = Disable falling edge trigger on input line x (interrupts and events)

23: 17

Reserved

Always read as 0.

16: 0

TRx

rw

0x00

Falling trigger event con-bit on line x (Falling trigger event con-bit

figuration bit of line x)

1 = Enable falling edge trigger on input line x (interrupts and events)

0 = Disable falling edge trigger on input line x (interrupts and events)

**7.3.5**

**(EXTI\_SWIER)**

Offset address: 0x10

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

SWIER16

Reserved

SWIER15SWIER14SWIER13SWIER12SWIER11SWIER10 SWIER9 SWIER8 SWIER7 SWIER6 SWIER5 SWIER4 SWIER3 SWIER2 SWIER1 SWIER0

SWIER24

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 25

Reserved

Always read as 0.

twenty four

SWIERx

rw

0x00

Software interrupt on line x (Software interrupt on line x)

When this bit is '0', writing '1' will set the corresponding suspend in EXTI\_PR

Bit. If allowed in EXTI\_INTMASK and EXTI\_EVNTMASK

When this interrupt is generated, an interrupt will be generated at this time.

Note: This bit can be cleared by clearing the corresponding bit of EXTI\_PEND (writing '1')

Is '0'.

23: 17

Reserved

Always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

77/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 100** |

UM\_MM32F003\_q\_Ver1.19

(EXTI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

16: 0

SWIERx

rw

0x00

Software interrupt on line x (Software interrupt on line x)

When this bit is '0', writing '1' will set the corresponding suspend in EXTI\_PR

Bit. If allowed in EXTI\_INTMASK and EXTI\_EVNTMASK

When this interrupt is generated, an interrupt will be generated at this time.

Note: This bit can be cleared by clearing the corresponding bit of EXTI\_PEND (writing '1')

Is '0'.

**7.3.6**

**(EXTI\_PR)**

Offset address: 0x14

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

rc\_w1

PR16

Reserved

PR15

PR14

PR13

PR12

PR11

PR10

PR9

PR8

PR7

PR6

PR5

PR4

PR3

PR2

PR1

PR0

PR24

Reserved

rc\_w1

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 25

Reserved

Always read as 0.

twenty four

PRx

rc\_w1

0x00

Pending bit

1 = A selected trigger request has occurred

0 = no trigger request occurred

When a selected edge event occurs on the external interrupt line, this bit is set to '1'.

Writing '1' in this bit can clear it, or it can be detected by changing the edge

The polarity is cleared.

23: 17

Reserved

Always read as 0.

16: 0

PRx

rc\_w1

0x00

Pending bit

1 = A selected trigger request has occurred

0 = no trigger request occurred

When a selected edge event occurs on the external interrupt line, this bit is set to '1'.

Writing '1' in this bit can clear it, or it can be detected by changing the edge

The polarity is cleared.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

78/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 101** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

8 **DMA**

**(DMA)**

DMA controller (DMA)

**8.1 DMA**

Direct memory access is used to provide high-speed data transfer between peripherals and memory or between memory and memory.

Without any CPU intervention, data can be moved quickly through DMA. This saves CPU resources to do other

operating.

The DMA controller has 5 channels, and each channel specializes in managing multiple peripheral requests.

**8.2 DMA**

• 5 independent configurable channels.

• Each channel is directly connected to a dedicated hardware DMA request, and each channel also supports software triggering. these functions

Configured via software.

• Priority between 5 requests can be set by software programming (there are four levels: very high, high, medium and low), if

When equal priority is determined by hardware (request 0 takes priority over request 1, and so on).

• The transmission width (byte, halfword, fullword) of independent source and target data areas, simulating the process of packing and unpacking. Yuanhe

The target address must be aligned according to the data transmission width.

• Support circular buffer management.

• Each channel has 3 event flags (DMA half transfer, DMA transfer completed and DMA transfer error), these 3

The event flag logic may become a separate interrupt request.

• Transfer between memories.

• Peripherals and memory, memory and peripheral transmission.

• Flash, SRAM, peripheral SRAM, APB1, APB2, and AHB peripherals can be used as the source and destination of access.

• Programmable number of data transfers: the maximum is 65536.

The following is a functional block diagram:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

79/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 102** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

628500

Cortex-M0

DMA

Ch.1

Ch. 2

up to

Ch.5

Arbiter

AHB Slave

System

DMA

Flash᧕ਓ

Flash

SRAM

༽Ս઼ᰦ䫏᧗ࡦ

ಘ˄RCC˅

GPIOA

GPIOB

GPIOC

GPIOD

ẕ᧕

I2C1

TIM3

TIM2

UART1

SPI1

ADC1

TIM1

TIM16

TIM17

APB

DMA~≲

ᙫ

㓯

⸙䱥

UART2

21. DMA

**8.3**

The DMA controller and CPU core share the system data bus to perform direct memory data transfer. When CPU and DMA are the same

When accessing the same target (RAM or peripheral), DMA requests may stop the CPU from accessing the system bus up to several

Cycle, the bus arbiter performs circular scheduling to ensure that the CPU can get at least half of the system bus (memory or external

Set) bandwidth.

**8.3.1**

**DMA**

After an event occurs, the peripheral sends a request signal to the DMA controller. DMA controller according to channel priority

Right to process requests. When the DMA controller starts to access the peripheral, the DMA controller immediately sends a response to the peripheral

signal. When the response signal is received from the DMA controller, the peripheral immediately releases its request. When the peripheral releases this please

The DMA controller also cancels the response signal at the same time. If more requests occur, the peripheral can start the next processing.

In summary, each DMA transfer consists of 3 operations:

1. Perform the load operation from the peripheral data register or from the memory location specified by the DMA\_CMARx register.

2. Store data in the peripheral data register or store data in the memory location at the specified address in the DMA\_CMARx register.

3. Perform a decrement operation of the DMA\_CNDTRx register. This register contains the number of outstanding operations.

**8.3.2**

The arbiter initiates peripheral/memory access based on the priority requested by the channel. Priority management is divided into 2 stages:

• Software: The priority of each channel can be set in the DMA\_CCRx register with 4 levels:

**–** Highest priority

**–** High priority

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

80/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 103** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

**–** Medium priority

**–** Low priority

• Hardware: If 2 requests have the same software priority, the channel with the lower number

Tao has higher priority. For example, channel 2 has priority over channel 4.

**8.3.3**

**DMA**

Each channel can perform DMA transfers between peripheral registers with fixed addresses and memory addresses. DMA transfer

The amount of data is programmable up to 65536. The register containing the number of data items to be transferred, at each transfer

After decreasing.

The amount of data transferred from peripherals and memory can be programmed through the PSIZE and MSIZE bits in the DMA\_CCRx register.

By setting the PINC and MINC flags in the DMA\_CCRx register, the pointers of peripherals and memory are transferred every time

After the loss, the automatic increment can be selectively completed. When set to incremental mode, the next address to be transmitted will be the previous one

The address plus the increment value depends on the selected data width being 1, 2, or 4. The first transmitted address is stored in

DMA\_CPARx / DMA\_CMARx register. When the channel is configured in acyclic mode, after the end of transmission (ie transmission

The count becomes 0) DMA operation will no longer occur.

The following is the process of configuring DMA channel x (x represents the channel number):

1. Set the address of the peripheral register in the DMA\_CPARx register. When a peripheral data transfer request occurs, this place

The address will be the source or destination of the data transmission.

2. Set the address of the data memory in the DMA\_CMARx register. When a peripheral data transmission request occurs, the transmitted

Data will be read from or written to this address.

3. Set the amount of data to be transferred in the DMA\_CNDTRx register. After each data transmission, this value is decremented.

4. Set the channel priority in the PL[1:0] bits of the DMA\_CCRx register.

5. Set the direction of data transfer, cyclic mode, incremental mode of peripherals and memory in the DMA\_CCRx register,

Peripheral and memory data width, half of the transmission interrupt or transfer completion interrupt.

6. Set the ENABLE bit of the DMA\_CCRx register to start the channel. When the DMA channel is started, it can

Respond to DMA requests from peripherals on this channel.

When half of the data is transferred, the half transfer flag (HTIF) is set to 1, when the half transfer interrupt enable bit (HTIE) is set,

An interrupt request will be generated. After the data transmission is completed, the transmission completion flag (TCIF) is set to 1, when the transmission permission is set

When the completion interrupt bit (TCIE) is input, an interrupt request will be generated.

Circular mode is used to handle circular buffers and continuous data transmission (such as the scanning mode of the ADC). Send at DMA\_CCRx

The CIRC bit in the memory is used to enable this function. When the cyclic mode is activated and the number of data transmissions becomes 0, the

It will be automatically restored to the initial value set when configuring the channel, and the DMA operation will continue.

The operation of the DMA channel can be performed without peripheral requests. This operation is called memory-to-memory mode.

After setting the MEM2MEM bit in the DMA\_CCRx register, the DMA\_CCRx register is set in software

When the EN bit in the device starts the DMA channel, the DMA transfer will start immediately. When the DMA\_CNDTRx register becomes 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

81/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 104** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

, The DMA transfer ends. The memory-to-memory mode cannot be used simultaneously with the loop mode.

**8.3.4**

When PSIZE and MSIZE are different, the DMA module performs data alignment according to the following table.

30.

(PINC = MINC = 1)

Source

width

aims

width

transmission

number

Source **(** Address **/** Data **)**

Transfer operation

Target **(** Address **/** Data **)**

8

8

4

0x0/B0

0x1/B1

0x2/B2

0x3/B3

1: Read B0[7:0] at 0x0, write B0[7:0] at 0x0

2: Read B1[7:0] at 0x1, write B1[7:0] at 0x1

3: Read B2[7:0] at 0x2, write B2[7:0] at 0x2

4: Read B3[7:0] at 0x3, write B3[7:0] at 0x3

0x0/B0

0x1/B1

0x2/B2

0x3/B3

8

16

4

0x0/B0

0x1/B1

0x2/B2

0x3/B3

1: Read B0[7:0] at 0x0, write 00B0[15:0] at 0x0

2: Read B1[7:0] at 0x1 and write 00B1[15:0] at 0x2

3: Read B2[7:0] at 0x2, and write 00B2[15:0] at 0x4

4: Read B3[7:0] at 0x3, write 00B3[15:0] at 0x6

0x0/00B0

0x2/00B1

0x4/00B2

0x6/00B3

8

32

4

0x0/B0

0x1/B1

0x2/B2

0x3/B3

1: Read B0[7:0] at 0x0, write 000000B0[31:0] at 0x0

2: Read B1[7:0] at 0x1, write 000000B1[31:0] at 0x4

3: Read B2[7:0] at 0x2, write 000000B2[31:0] at 0x8

4: Read B3[7:0] at 0x3, write 000000B3[31:0] at 0xC

0x0/000000B0

0x4/000000B1

0x8/000000B2

0xC/000000B3

16

8

4

0x0/B1B0

0x2/B3B2

0x4/B5B4

0x6/B7B6

1: Read B1B0[15:0] at 0x0, write B0[7:0] at 0x0

2: Read B3B2[15:0] at 0x2, write B2[7:0] at 0x1

3: Read B5B4[15:0] at 0x4, write B4[7:0] at 0x2

4: Read B7B6[15:0] at 0x6, write B6[7:0] at 0x3

0x0/B0

0x1/B2

0x2/B4

0x3/B6

16

16

4

0x0/B1B0

0x2/B3B2

0x4/B5B4

0x6/B7B6

1: Read B1B0[15:0] at 0x0, write B1B0[15:0] at 0x0

2: Read B3B2[15:0] at 0x2, write B3B2[15:0] at 0x2

3: Read B5B4[15:0] at 0x4, write B5B4[15:0] at 0x4

4: Read B7B6[15:0] at 0x6, write B7B6[15:0] at 0x6

0x0/B1B0

0x2/B3B2

0x4/B5B4

0x6/B7B6

16

32

4

0x0/B1B0

0x2/B3B2

0x4/B5B4

0x6/B7B6

1: Read B1B0[15:0] at 0x0, write 0000B1B0[31:0] at 0x0

2: Read B3B2[15:0] at 0x2, and write 0000B3B2[31:0] at 0x4

3: Read B5B4[15:0] at 0x4 and write 0000B5B4[31:0] at 0x8

4: Read B7B6[15:0] at 0x6 and write 0000B7B6[31:0] at 0xC

0x0/0000B1B0

0x4/0000B3B2

0x8/0000B5B4

0xC/0000B7B6

32

8

4

0x0/B3B2B1B0

0x4/B7B6B5B4

0x8/BBBAB9B8

0xC/BFBEBDBC

1: Read B3B2B1B0[31:0] at 0x0, write B0[7:0] at 0x0

2: Read B7B6B5B4[31:0] at 0x4, write B4[7:0] at 0x1

3: Read BBBAB9B8[31:0] at 0x8, write B8[7:0] at 0x2

4: Read BFBEBDBC[31:0] at 0xC, write BC[7:0] at 0x3

0x0/B0

0x1/B4

0x2/B8

0x3/BC

32

16

4

0x0/B3B2B1B0

0x4/B7B6B5B4

0x8/BBBAB9B8

0xC/BFBEBDBC

1: Read B3B2B1B0[31:0] at 0x0, write B1B0[15:0] at 0x0

2: Read B7B6B5B4[31:0] at 0x4, write B5B4[15:0] at 0x2

3: Read BBBAB9B8[31:0] at 0x8, write B9B8[15:0] at 0x4

4: Read BFBEBDBC[31:0] at 0xC, write BDBC[15:0] at 0x6

0x0/B1B0

0x2/B5B4

0x4/B9B8

0x6/BDBC

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

82/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 105** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

Source

width

aims

width

transmission

number

Source **(** Address **/** Data **)**

Transfer operation

Target **(** Address **/** Data **)**

32

32

4

0x0/B3B2B1B0

0x4/B7B6B5B4

0x8/BBBAB9B8

0xC/BFBEBDBC

1: Read B3B2B1B0[31:0] at 0x0, write B3B2B1- at 0x0

B0[31:0]

2: Read B7B6B5B4[31:0] at 0x4, write B7B6B5- at 0x4

B4[31:0]

3: Read BBBAB9B8[31:0] at 0x8, write BBBAB9- at 0x8

B8[31:0]

4: Read BFBEBDBC[31:0] at 0xC, write BFBEBD- at 0xC-

BC[31:0]

0x0/B3B2B1B0

0x4/B7B6B5B4

0x8/BBBAB9B8

0xC/BFBEBDBC

**AHB**

When the DMA module starts an AHB byte or halfword write operation, the data will not be in the HWDATA[31:0] bus

The part used is repeated. Therefore, if the DMA writes AHB in bytes or halfwords that does not support byte or halfword write operations

When the device (that is, HSIZE is not suitable for the module), no error will occur, and the DMA will write to 32 bits according to the following two examples

HWDATA data:

• When HSIZE is half word, write half word '0xABCD', DMA will set HWDATA bus to

'0xABCDABCD'.

• When HSIZE is byte, write byte '0xAB', DMA will set HWDATA bus to

'0xABABABAB'.

Assuming that the AHB/APB bridge is an AHB 32-bit slave device, it does not process HSIZE parameters, it will follow the

Transfer any bytes or halfwords on AHB to APB in 32 bits:

• An AHB operation on the write byte data '0xB0' at address 0x0 (or 0x1, 0x2 or 0x3) will be converted to

APB operates on the written data '0xB0B0B0B0' at address 0x0.

• An AHB write half word data '0xB1B0' operation at address 0x0 (or 0x2) will be converted to APB to ground

Write data '0xB1B0B1B0' at address 0x0.

For example, if you want to write to the APB backup register (a 16-bit register aligned with a 32-bit address), you need to configure the memory

The data source width (MSIZE) is '16 bits' and the peripheral target data width (PSIZE) is '32 bits'.

**8.3.5**

Reading and writing a reserved address area will cause a DMA transfer error. DMA transfer occurs during DMA read and write operations

When an error is input, the hardware will automatically clear the channel configuration register (DMA\_CCRx) corresponding to the channel where the error occurred

EN bit, the channel operation is stopped. At this time, the transfer error interrupt flag corresponding to the channel in the DMA\_IFT register

Bit (TEIF) will be set, if the transfer error interrupt enable bit is set in the DMA\_CCRx register, it will be generated

Interrupt.

**8.3.6**

Each DMA channel can generate an interrupt when the DMA transfer is over half, the transfer is completed, and the transfer error. Flexible for application

For sexual considerations, these interrupts are turned on by setting different bits of the register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

83/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 106** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

31. DMA

Interrupt event

Event flag

Enable control bit

More than half of the transmission

HTIF

HTIE

Transfer complete

TCIF

TCIE

Transmission error

TEIF

TEIE

**8.3.7**

**DMA**

**DMA**

5 requests from peripheral TIMx, ADC, SPI, I2C and UART, through logical OR input to DMA control

This means that only one request can be active at a time. See the DMA request image below.

Peripheral DMA requests can be turned on or off independently by setting control bits in corresponding peripheral registers.

040581

Channel 1

Fixed hardware priority

High priority

Internal

DMA

request

HW request 1

SW trigger (MEM 2MEM bit)

ADC1 (1)

TIM2\_CH3

TIM17\_CH1 (1)

TIM17\_UP (1)

HW request 2

SW trigger (MEM 2MEM bit)

ADC1 (2)

I2C1\_TX

TIM1\_CH1 TIM2\_UP

TIM3\_CH3

TIM17\_CH1

TIM17\_UP (2)

HW request 3

SW trigger (MEM 2MEM bit)

SPI1\_TX UART1\_RX (1)

I2C1\_RX

TIM1\_CH2 TIM2\_CH2

TIM3\_CH4 TIM3\_UP

TIM16\_CH1 (1) TIM16\_UP (1)

HW request 4

SW trigger (MEM 2MEM bit)

UART2\_TX TIM1\_CH4

TIM1\_TRIG TIM1\_COM

TIM2\_CH4

TIM3\_CH1 TIM3\_TRIG

TIM16\_CH1 (2)

TIM16\_UP (2)

HW request 5

SW trigger (MEM 2MEM bit)

UART2\_RX

TIM1\_UP TIM1\_CH3

TIM2\_CH1

Channel 2

Channel 3

Channel 4

Channel 5

(2)

SPI2\_RX

SPI2\_TX

twenty two.

DMA

32.

DMA

Peripherals

Channel **1**

Channel **2**

Channel **3**

Channel **4**

Channel **5**

ADC

ADC1 (1)

ADC1 (2)

SPI

SPI2\_RX

SPI2\_TX

UART

UART2\_TX

UART2\_RX

I2C

I2C1\_TX

I2C1\_RX

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

84/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 107** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

Peripherals

Channel **1**

Channel **2**

Channel **3**

Channel **4**

Channel **5**

TIM1

TIM1\_CH1

TIM1\_CH2

TIM1\_CH4

TIM1\_TRIG

TIM1\_COM

TIM1\_UP

TIM1\_CH3

TIM2

TIM2\_CH3

TIM2\_UP

TIM2\_CH2

TIM2\_CH4

TIM2\_CH1

TIM3

TIM3\_CH3

TIM3\_CH4

TIM3\_UP

TIM3\_CH1

TIM3\_TRIG

TIM16

TIM16\_CH1 (1)

TIM16\_UP (1)

TIM16\_CH1 (2)

TIM16\_UP (2)

TIM17

TIM17\_CH1 (1)

TIM17\_UP (1)

TIM17\_CH1 (2)

TIM17\_UP (2)

1. If the mapping bit in the SYSCFG\_CFGR register is cleared, the DMA request is mapped on this DMA channel.

2. If the mapping bit of the SYSCFG\_CFGR register is set, the DMA request is mapped on this DMA channel.

**8.4 DMA**

33. DMA

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

DMA\_ISR

DMA interrupt status register

0x00000000

[Section 8.4.1](https://translate.googleusercontent.com/translate_f#107)

0x04

DMA\_IFCR

DMA interrupt flag clear register

0x00000000

[Section 8.4.2](https://translate.googleusercontent.com/translate_f#108)

0x08 + 20 × (n-1)

DMA\_CCRx

DMA channel x configuration register

0x00000000

[Section 8.4.3](https://translate.googleusercontent.com/translate_f#109)

0x0C + 20 × (n-1)

DMA\_CNDTRx

DMA channel x transfer quantity register

0x00000000

[Section 8.4.4](https://translate.googleusercontent.com/translate_f#111)

0x10 + 20 × (n-1)

DMA\_CPARx

DMA channel x peripheral address register

0x00000000

[Section 8.4.5](https://translate.googleusercontent.com/translate_f#111)

0x14 + 20 × (n-1)

DMA\_CMARx

DMA channel x memory address register

0x00000000

[Section 8.4.6](https://translate.googleusercontent.com/translate_f#112)

**8.4.1**

**DMA**

**(DMA\_ISR)**

Offset address: 0x00

Reset value: 0x0000 0000

TCIF5

HTIF5

TEIF5

GIF5

TEIF4 HTIF4

TEIF3

GIF4

TCIF4

HTIF3 TCIF3

TCIF1

HTIF1

TEIF1

GIF2

TCIF2

HTIF2

TEIF2

GIF3

GIF1

Reserved

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 20

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

85/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 108** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

**Bit**

**Field**

**Type**

**Reset**

**Description**

19,15,11,

7, 3

TEIFx

r

0x00

Channel x transfer error flag (x = 1 …5) (Channel x transfer error

flag)

The hardware sets these bits. Write '1' in the corresponding bit of the DMA\_IFCR register

You can clear the corresponding flag here.

0: No transmission error (TE) on channel x

1: A transmission error (TE) occurred on channel x

18,14,10,

6, 2

HTIFx

r

0x00

Channel x half transfer flag (x = 1 …5) (Channel x half transfer flag

flag)

The hardware sets these bits. Write '1' in the corresponding bit of the DMA\_IFCR register

You can clear the corresponding flag here.

0: There is no half transmission event (HT) on channel x

1: A half transmission event (HT) occurred on channel x

17,13,9,

5, 1

TCIFx

r

0x00

Channel x transfer completion flag (x = 1 …5) (Channel x transfer com-

plete flag)

The hardware sets these bits. Write '1' in the corresponding bit of the DMA\_IFCR register

You can clear the corresponding flag here.

0: No transmission completion event (TC) on channel x

1: A transmission completion event (TC) was generated on channel x

16,12,8,

4, 0

GIFx

r

0x00

Channel x global interrupt flag (x = 1 …5) (Channel x global inter-

rupt flag)

The hardware sets these bits. Write '1' in the corresponding bit of the DMA\_IFCR register

You can clear the corresponding flag here.

0: There is no TE, HT or TC event on channel x

1: TE, HT or TC event was generated on channel x

**8.4.2**

**DMA**

**(DMA\_IFCR)**

Offset address: 0x04

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

CTE

IF1

CTE

IF2

CTE

IF3

CTE

IF4

CHT

IF1

CHT

IF2

CHT

IF3

CHT

IF4

CTC

IF1

CTC

IF2

CTC

IF3

CTC

IF4

CG

IF4

CG

IF3

CG

IF2

CG

IF1

CTE

IF5

CHT

IF5

CTC

IF5

CG

IF5

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 20

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

86/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 109** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

**Bit**

**Field**

**Type**

**Reset**

**Description**

19,15,11,

7, 3

CTEIFx

w

0x00

Clear the channel x transfer error flag (x = 1 …5) (Channel x transfer

error clear)

These bits are set and cleared by software.

0: not working

1: Clear the corresponding TEIF flag in the DMA\_ISR register

18,14,10,

6, 2

CHTIFx

w

0x00

Clear the half transmission flag of channel x (x = 1 …5) (Channel x half trans-

fer clear)

These bits are set and cleared by software.

0: not working

1: Clear the corresponding HTIF flag in the DMA\_ISR register

17,13,9,

5, 1

CTCIFx

w

0x00

Clear the transfer completion flag of channel x (x = 1 …5) (Channel x transfer

complete clear)

These bits are set and cleared by software.

0: not working

1: Clear the corresponding TCIF flag in the DMA\_ISR register

16,12,8,

4, 0

CGIFx

w

0x00

Clear the global interrupt flag of channel x (x = 1 …5) (Channel x global

interrupt clear)

These bits are set and cleared by software.

0: not working

1: Clear the corresponding GIF, TEIF, HTIF and DMA\_ISR registers

TCIF logo

**8.4.3**

**DMA**

**x**

**(DMA\_CCRx) (x = 1 5)**

Offset address: 0x08 + 20 x (channel number-1)

Reset value: 0x0000 0000

ARE

MEM2

MEM

PL

TCIE

HTIE

TEIE

DIR

CIRC

PINC

MINC

EN

MSIZE

PSIZE

Reserved

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Reserved, always read as 0.

15

ARE

rw

0x00

Auto-Reload Enable

This bit is set and cleared by software. After aborting the transmission, the NDT of each channel,

Whether the PADDR and MADDR registers return to the initial values ​​​​set:

1: Initial value of automatic reload setting after aborting transmission

0: Disable automatic reload function

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

87/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 110** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

**Bit**

**Field**

**Type**

**Reset**

**Description**

14

MEM2MEM

rw

0x00

Memory to memory mode

This bit is set and cleared by software.

0: non-memory to memory mode

1: boot memory to memory mode

13: 12

PL

rw

0x00

Channel priority level

These bits are set and cleared by software.

00: low

01: Medium

10: high

11: highest

11: 10

MSIZE

rw

0x00

Memory data width (Memory size)

These bits are set and cleared by software.

00: 8 bits

01: 16 bits

10: 32 bits

11: reserved

9: 8

PSIZE

rw

0x00

Peripheral size

These bits are set and cleared by software.

00: 8 bits

01: 16 bits

10: 32 bits

11: reserved

7

MINC

rw

0x00

Memory increment mode

This bit is set and cleared by software.

0: Do not perform memory address increment operation

1: Perform memory address increment operation

6

PINC

rw

0x00

Peripheral increment mode

This bit is set and cleared by software.

0: Do not perform peripheral address increment operation

1: Perform peripheral address increment operation

5

CIRC

rw

0x00

Circular mode

This bit is set and cleared by software.

0: Do not perform loop operation

1: Perform loop operation

4

DIR

rw

0x00

Data transfer direction

This bit is set and cleared by software.

0: read from peripheral

1: read from memory

3

TEIE

rw

0x00

Transfer error interrupt enable

This bit is set and cleared by software.

0: Disable TE interrupt

1: enable TE interrupt

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

88/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 111** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

HTIE

rw

0x00

Allow half transfer interrupt enable

This bit is set and cleared by software.

0: Disable HT interrupt

1: Enable HT interrupt

1

TCIE

rw

0x00

Transfer complete interrupt enable

This bit is set and cleared by software.

0: Disable TC interrupt

1: Allow TC interrupt

0

EN

rw

0x00

Channel enable

This bit is set and cleared by software.

0: channel does not work

1: Channel is on

**8.4.4**

**DMA**

**x**

**(DMA\_CNDTRx) (x = 1 5)**

Offset address: 0x0C + 20 x (channel number-1)

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

NDT

Reserved

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Reserved, always read as 0.

15: 0

NDT

rw

0x0000

Number of data to transfer

The number of data transfers is from 0 to 65535. This register can only work in the channel

Write (DMA\_CCRx EN = 0). After the channel is turned on, this register

Becomes read-only, indicating the remaining number of bytes to be transferred. The register content is in

Decreases after each DMA transfer. After the data transfer is completed, the contents of the register or

Becomes 0; or when the channel is configured in auto-reload mode, the register

The content of will be automatically reloaded to the value of the previous configuration.

When the content of the register is 0, no matter whether the channel is turned on, it will not happen

Any data transmission.

**8.4.5**

**DMA**

**x**

**(DMA\_CPARx) (x = 1 5)**

Offset address: 0x10 + 20 x (channel number-1)

Reset value: 0x0000 0000

This register cannot be written when the channel is turned on (EN of DMA\_CCRx = 1).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

89/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 112** |

UM\_MM32F003\_q\_Ver1.19

DMA

(DMA)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

PA

PA

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

PA

rw

0x0000

0000

Peripheral address

The base address of the peripheral data register is used as the source or destination of the data transfer.

When PSIZE = '01' (16 bits), the PA[0] bit is not used. Operation automatically

Align with halfword address.

When PSIZE = '10' (32 bits), the PA[1:0] bits are not used. Automatic operation

Ground is aligned with the word address.

**8.4.6**

**DMA**

**x**

**(DMA\_CMARx) (x = 1 5)**

Offset address: 0x14 + 20 x (channel number-1)

Reset value: 0x0000 0000

This register cannot be written when the channel is turned on (EN of DMA\_CCRx = 1).

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

MA

MA

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

MA

rw

0x0000

0000

Memory address

The memory address is used as the source or destination of data transfer.

When MSIZE = '01' (16 bits), the MA[0] bit is not used. Operation automatically

Align with halfword address.

When MSIZE = '10' (32 bits), the MA[1:0] bits are not used. Operate from

Align with the word address dynamically.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

90/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 113** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

9

**/**

**(ADC)**

Analog/Digital Conversion (ADC)

**9.1 ADC**

The 12-bit ADC is a successive approximation analog-to-digital converter (SAR A/D converter).

The A/D converter supports multiple working modes: single conversion and continuous conversion mode, and can select channel automatic scanning. A/D

The starting method of conversion includes software setting, external pin trigger and the start of each timer.

The window comparator (analog watchdog) allows the application to detect whether the input voltage exceeds the high/low threshold set by the user.

The input clock of the ADC must not exceed 15MHz. It is generated by dividing PCLK2.

**9.2 ADC**

• Up to 12-bit programmable resolution SAR ADC, up to 8 external input channels

• Up to 1Msps conversion rate

• Support multiple working modes:

**–** Single conversion mode: A/D conversion completes a conversion on the designated channel

**-** Single-Cycle Scan MODE: A / D Conversion in IS Completed The One Cycle All of specified Channel (The Channel Number from Low to High

Serial number channel) conversion

**–** Continuous scan mode: A/D conversion continuously executes single-cycle scan mode until the software stops A/D conversion

• Channel sampling time, resolution can be configured by software

• Support DMA transfer

• A/D conversion start conditions:

**–** Software start

**-** External Start the Trigger

**–** Timer match

• Analog watchdog, the conversion result can be compared with the specified value, when the conversion value matches the set value, the user can set

Whether to generate an interrupt request

**9.3 ADC**

The following figure shows the ADC block diagram

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

91/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 114** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

424697

AGND

VREF

Analog Multiplexer

12 people

SAR

ADC

AIN. 1

AIN. 2

AIN. 3

AIN. 4

AIN. 5

AIN. 6

AIN. 0

...

⑙ᓖՐᝏಘ

ⵥ঻৲㘳⭥঻

A

nalog M

U

X

A

P

B

B

us

A

D

C

onfigure R

egister

(A

D

C

FG

)

A

D

D

A

TA

R

egister 0

(A

D

D

R

0)

A

D

C

hanel S

elect

(A

D

C

H

S

)

A

D

C

ontrol R

egister

(A

D

C

R

)

A

D

D

A

TA

R

egister 1

(A

D

D

R

1)

...

A

D

C

om

pare R

egister

(A

D

C

M

P

R

)

A

D

S

tatus R

egister

(A

D

S

TA

)

Compare data

CMPF

A

D

E

O

C

F

AD interrupt

...

Australia

23. ADC

**9.3.1**

**ADC**

The ADC can be powered on by setting the ADEN bit in the ADCFG register. When the ADEN bit is set for the first time, it will

The ADC wakes up from the power-off state.

After the ADC powers up for a period of time, set the ADST bit in the ADCR register to start conversion.

The conversion can be stopped by clearing the ADST bit, and the power-down mode can be placed by clearing the ADEN bit.

**9.3.2**

Contains multiple external input channels, internal temperature sensor channels and internal 1.2V reference voltage channel. Each external input

Each channel has an independent enable bit, which can be set by setting the corresponding bit in the ADCHS register.

**9.4 ADC**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

92/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 115** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**9.4.1**

In the single conversion mode, A/D conversion is performed only once on the corresponding channel. The specific process is as follows:

• Set A/D conversion by software, external trigger input and ADST of ADCR register set by timer overflow.

• When the A/D conversion is completed, the data value of the A/D conversion will be stored in the A/D data registers ADDATA and ADDRn

in.

• When the A/D conversion is completed, the ADIF bit of the status register ADSTA is set. If the ADIE of the control register ADCR at this time

Set to 1, will generate an AD conversion end interrupt request.

• During the A/D conversion, the ADST bit remains at 1. At the end of the A/D conversion, the ADST bit is automatically cleared to 0, and the A/D converter

Enter idle mode.

Note: In single conversion mode, if the software enables more than one channel, the channel with the lowest serial number is converted, and the other channels are ignored.

307539

**ADCLK**

**ADST**

**SAMPLE**

**ADDATA[11:0]**

**ADEOCF**

SAMCTL[3:0]

RSLTCTL[2:0]

**Sample**

**Vin(n)**

**ADDATA[11:0]**

Australia

twenty four.

**9.4.2**

In the single-cycle scan mode, the channels that can be started will be sequenced (scanning can be selected through the configuration register bit SCAN\_DIR

Channel direction) To perform an A/D conversion, the operation steps are as follows:

• Set by software or external trigger to start ADST, the direction setting defaults from the smallest serial number channel to the largest serial number channel A/D

Conversion can also be set according to the program, from the largest serial number channel to the smallest serial number channel A/D conversion.

• After each A/D conversion is completed, the A/D conversion values ​​​​will be loaded into the data register of the corresponding channel in order, and the ADIF conversion

The conversion end flag is set. If the conversion end interrupt is set, an interrupt will be generated after the conversion of all channels is completed.

begging.

• After the conversion is completed, the ADST bit is automatically cleared, and the A/D converter enters the idle state.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

93/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 116** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

603371

**ADST**

**Chsel[2:0]**

**sample**

**ADDATA[11:0]**

**SAR[11:0]**

**3'b000**

**3'b010**

**3'b111**

**3'b101**

**R0**

**R2**

**R7**

**R5**

**R0**

**R2**

**R5**

**R7**

**DMA** ᩜ䎠

**DMA** ᩜ䎠

**DMA** ᩜ䎠

**DMA** ᩜ䎠

অ ઘᵏ ᢛ ᧿䙊䚃 **0, 2 ,.. 7. 5 ,.** of The The ADCHS **[.. 8** ˖ **0] = 9'b010100101** ˅

Australia

25.

()

040923

**ADST**

**Chsel[2:0]**

**sample**

**ADDATA[11:0]**

**SAR[11:0]**

**3'b111**

**3'b101**

**3'b000**

**3'b010**

**R0**

**R2**

**R7**

**R5**

**R0**

**R2**

**R5**

**R7**

**DMA** ᩜ䎠

**DMA** ᩜ䎠

**DMA** ᩜ䎠

**DMA** ᩜ䎠

অ ઘᵏ ᢛ ᧿䙊䚃 **0, 2 ,.. 7. 5 ,.** of The The ADCHS **[.. 8** ˖ **0] = 9'b010100101** ˅

Australia

26.

()

**9.4.3**

In continuous scan mode, A/D conversion is performed sequentially on the channel where the CHENn bit in the ADCHS register is enabled

(The scan channel direction can be selected through the configuration register bit SCAN\_DIR), the operation steps are as follows:

• Set by software or external trigger to start ADST, external trigger can be configured by software trigger delay, direction setting defaults from the minimum

The A/D conversion of the serial number channel to the largest serial number channel can also be set according to the program, from the largest serial number channel to the smallest serial number

Channel A/D conversion.

• After the A/D conversion of all channels is completed, the A/D conversion values ​​​​will be loaded into the corresponding data registers in order,

ADIF conversion end flag is set, if the conversion end interrupt is set, it will be generated after all channels conversion is completed

Interrupt request.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

94/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 117** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

• As long as the ADST bit remains at 1, A/D conversion continues. When the ADST bit is cleared to 0, A/D conversion is completed, A/D

The converter enters the idle state. When ADST is cleared to 0, the A/D conversion will complete the current conversion.

550679

**ADST**

**Chsel[2:0]**

**sample**

**ADDATA[11:0]**

**3'b000 3'b010**

**3'b111**

**3'b101**

䘎㔝ᢛ ᧿䙊䚃 **0, 2 ,.. 7. 5 ,.** of The The ADCHS **[.. 8** ˖ **0] = 9'b010100101** ˅

**3'b000 3'b010**

**3'b101**

**3'b111**

**3'b000**

Lost ␵ 䲔 **ADST**

Australia

27.

()

453605

**ADST**

**Chsel[2:0]**

**sample**

**ADDATA[11:0]**

**3'b111 3'b101**

**3'b000**

**3'b010**

䘎㔝ᢛ ᧿䙊䚃 **0, 2 ,.. 7. 5 ,.** of The The ADCHS **[.. 8** ˖ **0] = 9'b010100101** ˅

**3'b111 3'b101**

**3'b010**

**3'b000**

**3'b111**

Lost ␵ 䲔 **ADST**

Australia

28.

()

**9.4.4**

**DMA**

The channel conversion values ​​​​during single-cycle scanning and continuous scanning are stored in the data register (ADDRn) of the respective channel, the most recent

The result of the second conversion is also stored in the ADDATA register. DMA transmission can choose to transfer a specific channel

Data, or transmit the results of all scan channels.

**9.5**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

95/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 118** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

The ALIGN bit in the ADCR register selects the alignment of the data storage after conversion. Data can be aligned left or right,

As shown below.

730272

0

0

0

0

D11 D10 D9

D8

D7

D6

D5

D4

D3

D2

D1

D0

D11 D10 D9

D8

D7

D6

D5

D4

D3

D2

D1

D0

0

0

0

0

Right-justified data

Data is left aligned

29.

**9.5.1**

The ADC conversion effective number of bits can be changed by the RSLTCTL[2:0] bits in the ADC\_CFG register to speed up the data

For conversion rate, the effective data bits are aligned at the high order of the 12-bit data.

**9.5.2**

The ADC clock ADCLK is obtained by dividing PCLK2. The division factor can be set by AD- of ADCFG register.

It is determined by the CPRE bit, that is, the PCLK2 / (N+1) / 2 frequency is used as the ADC clock.

Set the ADC resolution to n bits (n=8,9,10,11,12), each channel sampling period is m, the number of sampling periods can be

Changed by the SAMCTL bit in the ADC\_CFG register.

The sampling frequency and sampling time are calculated as follows:

F sample = F ADCLK / (m + n + 1.5).

Eg:

When ADCCLK = 15MHz, the resolution is configured as 12Bit, and the sampling time is 1.5 cycles

F sample = F ADCLK / 15.

T CONV = 1.5 + 13.5 = 15 period = 1µs

**9.6**

ADC conversion can be triggered by external events (eg timer capture, EXTI line). If the ADCR register is set

TRGEN bit, you can use external events to trigger conversion. The external trigger source can be selected by setting the TRGSEL bit.

For specific external trigger source selection, please refer to the description of the relevant bits in the AD control register.

External trigger can be set to delay control, please refer to the description of TRGSHIFT in ADCR[21:19].

After the trigger signal is generated, it delays N PCLK2 clock cycles to start sampling. If it is triggered scan mode, only

The sampling of the first channel is delayed, and the remaining channels start immediately after the last sampling.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

96/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 119** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**9.7**

The temperature sensor can be used to measure the ambient temperature device (T A ).

The temperature sensor is internally connected to the internal signal source channel of the ADC. This channel converts the voltage output by the sensor into

Numeric value. When the sensor is not in use, the corresponding bit of the register can be set to turn off separately.

The output voltage of the temperature sensor changes linearly with the temperature. Due to the change of the production process, the deviation of the temperature change curve is different.

There will be a difference on the chip.

The internal temperature sensor is more suitable for detecting temperature changes rather than measuring absolute temperatures. If you need to measure accurately

For temperature, an external temperature sensor should be used.

The temperature value is calculated as follows:

T(

◦

C) = (V SENSE -V 25 ) / Avg\_Slope + 25

V 25 : 25

◦

V SENSE value at C

V SENSE : Current output voltage of temperature sensor

V SENSE = Value \* V dd / 4096 (Value is the conversion result data of ADC)

Avg\_Slope: Average slope of temperature and V SENSE curve (in mV/ ◦

C or µV/

◦

C means)

For typical values ​​​​of of V 25 and Avg\_Slope, please refer to the temperature sensor chapter of the data sheet.

**9.8**

The internal signal source channel of the ADC is connected to an internal reference voltage, the size of which is 1.2V.

The test voltage output is converted to a digital value.

The internal reference voltage has a separate start bit, which can be turned on or off by setting the corresponding bit of the register.

**9.9**

**AD**

In compare mode, two compare registers are provided: upper limit and lower limit. The monitoring channel can be selected by software setting CMPCH bit.

When CPMHDATA ≥ CPMLDATA, the comparison result is greater than or equal to CMPHDATA in the ADCMPR register

If the specified value is less than the specified value of CMPLDATA, the ADWIF bit of the status register ADSTA is set.

When CPMHDATA <CPMLDATA, if the comparison result is equal to the value specified by CPMHDATA or in two

Between fixed values, the ADWIF bit of the status register ADSTA is set. If the ADWIE of the control register ADCR is set

Bit, an interrupt request will be generated.

**9.10 ADC**

34. ADC

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

ADC\_ADDATA

A/D data register

0x00000000

[Section 9.10.1](https://translate.googleusercontent.com/translate_f#120)

0x04

ADC\_ADCFG

A/D configuration register

0x00000000

[Section 9.10.2](https://translate.googleusercontent.com/translate_f#121)

0x08

ADC\_ADCR

A/D control register

0x00000000

[Section 9.10.3](https://translate.googleusercontent.com/translate_f#122)

0x0C

ADC\_ADCHS

A/D channel selection register

0x00000000

[Section 9.10.4](https://translate.googleusercontent.com/translate_f#124)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

97/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 120** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x10

ADC\_ADCMPR

A/D window compare register

0x00000000

[Section 9.10.5](https://translate.googleusercontent.com/translate_f#126)

0x14

ADC\_ADSTA

A/D status register

0x00000000

[Section 9.10.6](https://translate.googleusercontent.com/translate_f#126)

0x18

ADC\_ADDR0

A/D data register

0x00000000

[Section 9.10.7](https://translate.googleusercontent.com/translate_f#127)

0x28∼0x30

ADC\_ADDR4 ∼ 6

A/D data register

0x00000000

[Section 9.10.7](https://translate.googleusercontent.com/translate_f#127)

0x3C∼0x48

ADC\_ADDR9 ∼ 12

A/D data register

0x00000000

[Section 9.10.7](https://translate.googleusercontent.com/translate_f#127)

0x50∼ 0x54

ADC\_ADDR14 ∼ 15

A/D data register

0x00000000

[Section 9.10.7](https://translate.googleusercontent.com/translate_f#127)

0x58

ADC\_ADSTA\_EXT

A/D extended status register

0x00000000

[Section 9.10.8](https://translate.googleusercontent.com/translate_f#128)

**9.10.1 A/D**

**(ADC\_ADDATA)**

Address offset: 0x00

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

DATA

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

OVER

RUN

CHANNELSEL

VAILD

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 22

Reserved

Reserved, always read as 0.

twenty one

VALID

r

0x00

Valid flag (read only) (Valid flag)

1 = DATA[11:0] bit data is valid

0 = DATA[11:0] Bit data is invalid

After the conversion of the corresponding analog channel is completed, set this bit and read the ADDATA register

After the device, this bit is cleared by hardware.

20

OVERRUN

r

0x00

Data overlay flag (read only) (Overrun flag)

1 = DATA[11:0] The data is overwritten

0 = DATA[11:0] The last conversion result of the data

Before the new conversion result is loaded into the register, if the data of DATA[11:0]

If not read, OVERRUN will be set to 1. After reading the ADDATA register,

This bit is cleared by hardware.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

98/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 121** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

19: 16

CHANNELSEL r

0x00

The 4 digits show the channel corresponding to the current data (Channel selection)

0000 = Conversion data of channel 0

0100 = Conversion data of channel 4

0101 = Conversion data of channel 5

0110 = Conversion data of channel 6

1001 = Conversion data of channel 9

1010 = Conversion data of channel 10

1011 = Conversion data for channel 11

1100 = Conversion data of channel 12

1110 = Conversion data of the temperature sensor

1111 = Conversion data of internal reference voltage

Other: invalid

15: 0

DATA

r

0x00

12-bit A/D conversion result (Transfer data)

Align left or right according to the setting.

**9.10.2 A/D**

**(ADC\_ADCFG)**

Address offset: 0x04

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

VSEN

SAMCTL

RSLTCTL

ADCPRE

TSEN ADWEN ADEN

ADCPRE

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 15

Reserved

Reserved, always read as 0.

14

ADCPRE

rw

0x00

ADC prescaler

As the least significant bit of ADCPRE[3:0], used with Bit[6:4]

13: 10

SAMCTL

rw

0x00

Channel x Sample time selection

These bits are used to independently select the sampling time of each channel. During the sampling period

The middle channel selection bit must remain unchanged.

0000: 1.5 cycles

0100: 41.5 cycles

0001: 7.5 cycles

0101: 55.5 cycles

0010: 13.5 cycles

0110: 71.5 cycles

0011: 28.5 cycles

0111: 239.5 cycles

1000: 2.5 cycles

1001: 3.5 cycles

1010: 4.5 cycles

1011: 5.5 cycles

1100: 6.5 cycles

Other: reserved

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

99/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 122** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9: 7

RSLTCTL

rw

0x00

Select ADCx conversion data resolution (resolution)

000: 12 bits are valid

001: 11 bits are valid

010: 10 digits are valid

011: 9 digits are valid

100: 8 bits are valid

6: 4

ADCPRE

rw

0x00

ADC prescaler

Set by software to '1' or clear '0' to determine the ADC clock frequency.

When Bit[14] is 0, the actual frequency division coefficient is (2 \* (ADCPRE + 1))

When Bit[14] is 1, the actual frequency division coefficient is (2 \* (ADCPRE + 1) + 1)

3

VSEN

rw

0x00

Internal Voltage Reference Enable (Voltage Sensor enable)

1: The internal voltage sensor is enabled

0: The internal voltage sensor is disabled

2

TSEN

rw

0x00

Temperature sensor enable control bit (Temperature sensor enable)

1 = Temperature sensor is enabled

0 = temperature sensor disabled

1

ADWEN

rw

0x00

A/D window comparator enable (ADC window comparison enable)

1 = A/D window comparator is enabled

0 = A/D window comparator is disabled

0

ADEN

rw

0x00

A/D conversion enable (ADC enable)

1 = enabled

0 = disabled

**9.10.3 A/D**

**(ADC\_ADCR)**

Address offset: 0x08

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CMPCH

ADMD

ALIGN

ADST

Res.

DMAEN

Reserved

TRGSEL

TRGEN ADWIE

ADIE

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

TRGSHIFT

TRGSEL

SCANDIR

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 22

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

100/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 123** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

21: 19

TRGSHIFT

rw

0x00

External trigger shift sample

After the trigger signal is generated, delay N PCLK2 clock cycles before starting

sampling.

If it is the trigger scan mode, the other channels are set after the last sampling

Start now.

0: no delay

1: 4 cycles

2: 16 cycles

3: 32 cycles

4: 64 cycles

5: 128 cycles

6: 256 cycles

7: 512 cycles

18: 17

TRGSEL

rw

0x00

External trigger selection

Share with Bit[6:4]

16

SCANDIR

rw

0x00

ADC scan direction

In single cycle scan or continuous scan mode, set the scan channel sequence

0: ADC channel selection registers are scanned in order from low to high

1: The ADC channel selection registers are scanned in order from high to low

15: 12

CMPCH

rw

0x00

Window comparison channel selec-

tion) selection)

0000 = Compare channel 0 conversion result selected

0001 = Compare channel 4 conversion result selected

0010 = Compare channel 5 conversion result selected

0011 = Compare channel 6 conversion result selected

0100 = Compare channel 9 conversion result selected

0101 = Select compare channel 10 conversion result

0110 = Compare channel 11 conversion result selected

0111 = Select compare channel 12 conversion result

1010 = Select to compare conversion results of temperature sensors

1011 = Select to compare internal reference voltage conversion results

1111 = all scan channels

Other: invalid

11

ALIGN

rw

0x00

Data alignment

0: right aligned

1: Left aligned

10: 9

ADMD

rw

0x00

A/D conversion mode (ADC mode)

00: single conversion

01: Single cycle scan

10: Continuous scanning

When changing the conversion mode, the software must first disable the ADST bit.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

101/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 124** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

8

ADST

rw

0x00

A/D conversion start (ADC start)

1 = Start of conversion

0 = end of conversion or enter idle state

There are two ways to set ADST:

In single mode or single cycle mode, after the conversion is completed, ADST will be hard

The file is automatically cleared.

In continuous scan mode, A/D conversion will continue until the software writes '0'

To this bit or system reset.

7

Reserved

Reserved, always read as 0.

6: 4

TRGSEL

rw

0x00

External trigger selection (External trigger selection), bits [18:17,6:4]

Select external trigger source

00000: TIM1\_CC1

00001: TIM1\_CC2

00010: TIM1\_CC3

00011: TIM2\_CC2

00100: TIM3\_TRGO

00101: TIM1\_CC4 and CC5

00110: TIM3\_CC1

00111: EXTI line 11

01000: TIM1\_TRGO

01011: TIM2\_CC1

01100: TIM3\_CC4

01101: TIM2\_TRGO

01111: EXTI line 15

10000: TIM1\_CC4

10001: TIM1\_CC5

Other: invalid

3

DMAEN

rw

0x00

DMA enable (Direct memory access enable)

1 = DMA request is enabled

0 = DMA disabled

2

TRGEN

rw

0x00

External trigger enable (External trigger enable)

1 = Use external trigger signal to start A/D conversion

0 = Start A/D conversion without external trigger signal

1

ADWIE

rw

0x00

A/D window comparator interrupt enable (ADC window comparator inter-

rupt enable)

1 = A/D window comparator interrupt is enabled

0 = Disable A/D window comparator interrupt

0

ADIE

rw

0x00

A/D interrupt enable (ADC interrupt enable)

1 = A/D interrupt is enabled

0 = Disable A/D interrupt

If ADIF is set, an interrupt request is generated after the A/D conversion is completed.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

102/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 125** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**9.10.4 A/D**

**(ADC\_ADCHS)**

Address offset: 0x0C

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CHEN9

CHEN11

CHEN12

CHEN6

Reserved

CHEN10

CHEN5 CHEN4

CHEN0

rw

rw

rw

rw

rw

rw

rw

rw

CHENVS CHENTS

rw

rw

Res.

Reserved

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Reserved, always read as 0.

15

CHENVS

rw

0x00

Internal Voltage Reference Enable (Voltage Sensor enable)

1 = enabled

0 = disabled

14

CHENTS

rw

0x00

Temperature Sensor enable

1 = enabled

0 = disabled

13

Reserved

Reserved, always read as 0.

12

CHEN12

rw

0x00

Analog input channel 12 enable (Analog input channel 12 enable)

1 = enabled

0 = disabled

11

CHEN11

rw

0x00

Analog input channel 11 enable (Analog input channel 11 enable)

1 = enabled

0 = disabled

10

CHEN10

rw

0x00

Analog input channel 10 enable (Analog input channel 10 enable)

1 = enabled

0 = disabled

9

CHEN9

rw

0x00

Analog input channel 9 enable (Analog input channel 9 enable)

1 = enabled

0 = disabled

8: 7

Reserved

Reserved, always read as 0.

6

CHEN6

rw

0x00

Analog input channel 6 enable (Analog input channel 6 enable)

1 = enabled

0 = disabled

5

CHEN5

rw

0x00

Analog input channel 5 enable (Analog input channel 5 enable)

1 = enabled

0 = disabled

4

CHEN4

rw

0x00

Analog input channel 4 enable (Analog input channel 4 enable)

1 = enabled

0 = disabled

3: 1

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

103/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 126** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

CHEN0

rw

0x00

Analog input channel 0 enable (Analog input channel 0 enable)

1 = enabled

0 = disabled

Note: If the channel enable is 0, channel 0 is enabled.

**9.10.5 A/D**

**(ADC\_ADCMPR)**

Address offset: 0x10

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CMPHDATA

CMPLDATA

Reserved

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 28

Reserved

Reserved, always read as 0.

27: 16

CMPHDATA

rw

0x00

Compare data high limit

The 12-bit value will be compared with the conversion result of the specified channel.

15: 12

Reserved

Reserved, always read as 0.

11: 0

CMPLDATA

rw

0x00

Compare data low limit

The 12-bit value will be compared with the conversion result of the specified channel.

**9.10.6 A/D**

**(ADC\_ADSTA)**

Address offset: 0x14

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Res.

VALID

OVERRUN

VALID

CHANNEL

BUSY

ADWIF

ADIF

r

r

r

r

r

r

r

r

r

r

rc\_w1

r

r

r

r

r

r

rc\_w1

r

r

r

r

r

r

r

r

r

r

r

r

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 20

OVERRUN

r

0x0000

Channel 0, 4 ∼ 6, 9 ∼ 11 data overrun flag (Overrun flag)

Read only.

19: 8

VALID

r

0x0000

Valid flags of channels 0, 4 ∼ 6, 9 ∼ 11

Read only.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

104/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 127** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7: 4

CHANNEL

r

0x00

Current conversion channel

These 4 bits indicate the channel being converted when BUSY = 1. BUSY = 0

The time indicates the channel that can be converted next.

3

Reserved

Reserved, always read as 0.

2

BUSY

r

0x00

Busy

1 = A/D converter is busy

0 = A/D converter is idle

1

ADWIF

rc\_w1

0x00

Comparison flag (ADC window comparator interrupt flag)

Selected A/D conversion channel, the result is greater than or equal to ADCMPHR or less than

ADCMPLR, this bit is '1'.

Writing a '1' to this flag bit clears it.

0

ADIF

rc\_w1

0x00

A/D conversion end flag (ADC interrupt flag)

This bit is set by hardware at the end of the channel group conversion and cleared by software.

1 = A/D conversion completed

0 = A/D conversion is not completed

Writing a '1' to this flag bit clears it.

**9.10.7 A/D**

**(ADC\_ADDR0,4** ∼ **6,9** ∼ **12,14** ∼ **15)**

Address offset: 0x18, 0x28 − 0x30, 0x3C − 0x48, 0x50 − 0x54

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

DATA

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

OVER

RUN

Reserved

VAILD

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 22

Reserved

Reserved, always read as 0.

twenty one

VALID

r

0x00

Valid flag (read only) (Valid flag)

1 = DATA[11:0] bit data is valid

0 = DATA[11:0] Bit data is invalid

After the conversion of the corresponding analog channel is completed, set this bit and read the ADDATA register

After the device, this bit is cleared by hardware.

20

OVERRUN

r

0x00

Data overlay flag (read only) (Overrun flag)

1 = DATA [11:0] Data is overwritten

0 = DATA [11:0] The last conversion result of the data

Before the new conversion result is loaded into the register, if the data of DATA[11:0]

If it is not read, OVERRUN will be set to '1'. After reading the ADDATA register,

This bit is cleared by hardware.

19: 16

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

105/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 128** |

UM\_MM32F003\_q\_Ver1.19

/ (ADC)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 0

DATA

r

0x00

12-bit A/D conversion result of channel (Transfer data)

Align left or right according to the setting.

**9.10.8 A/D**

**(ADC\_ADSTA\_EXT)**

Address offset: 0x58

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

r

r

r

r

OVERRUN

VALID

r

r

r

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 8

Reserved

Reserved, always read as 0.

7: 4

OVERRUN

r

0x00

Channel data overlay flag (Overrun flag)

1000: channel 15 (V\_SENSOR)

0100: Channel 14 (T\_SENSOR)

0001: Channel 12

3: 0

VALID

r

0x00

Valid flag of the channel (Valid flag)

1000: channel 15 (V\_SENSOR)

0100: Channel 14 (T\_SENSOR)

0001: Channel 12

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

106/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 129** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

10

**(TIM1)**

Advanced Control Timer (TIM1)

**10.1 TIM1**

The advanced control timer (TIM1) consists of a 16-bit auto-reload counter, which consists of a programmable prescaler

drive.

It is suitable for a variety of purposes, including measuring the pulse width of an input signal (input capture), or generating an output waveform (output ratio

Compare, PWM, complementary PWM with embedded dead time, etc.).

Using the timer prescaler and RCC clock to control the prescaler, the pulse width and waveform period can be adjusted from several microseconds.

Adjustment from seconds to several milliseconds.

The advanced control timer (TIM1) and general-purpose timer (TIMx) are completely independent, and they do not share any resources. they

It can be operated synchronously. For detailed description, please refer to the chapter of general timer synchronization.

**10.2**

The functions of the TIM1 timer include:

• 16-bit up, down, up/down autoload register

• 16-bit programmable (can be modified in real time) prescaler, the counter clock frequency division factor is between 1 and 65536

Any value

• Up to 4 independent channels

**–** Input capture

**–** Output comparison

**–** PWM generation (edge ​​​or center alignment mode)

**–** Single pulse mode output

• Complementary output with programmable dead time

• Use external signals to control the timer and timer interconnection synchronization circuit

• Allow repeat counters to update timer registers after a specified number of counter cycles

• The brake input signal can put the timer output signal into a reset state or a known state

• An interrupt/DMA is generated when the following events occur:

**–** Update: counter overflow/downflow, counter initialization (triggered by software or internal/external)

**–** Trigger events (counter start, stop, initialization or counting by internal/external trigger)

**–** Input capture

**–** Output comparison

**-** Brake Signal the INPUT

• Supports incremental (quadrature) encoder and Hall sensor circuits for positioning

• Trigger input as external clock or cycle-by-cycle current management

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

107/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 130** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

459861

enter

filter

ITR0

ITR1

ITR2

ITR3

ITR

ETRP

TRGI

TRC

TI1FP1

TI2FP2

ETRF

(CK\_INT)

TRGO

ETR

CNT counter

+/-

PSC

Prescaler

CK\_CNT

CK\_PSC

Repeat register

U

ETRF

Prescaler

IC1PS U

Prescaler

IC2PS

Prescaler

IC3PS

Prescaler

IC4PS

IC1

TI1FP1

TI1FP2

TRC

TI2FP1

TI2FP2

TRC

IC2

IC3

IC4

TRC

TRC

TI1

XOR

TI2

TI3

TIMx\_ETR

TIMx\_CH1

TIMx\_CH2

TIMx\_CH3

TIMx\_CH4

TI4

U

U

U

DTG

OC1REF

CC1I

CC2I

DTG

OC2REF

DTG

OC3REF

CC3I

CC4I

OC4REF

TGI

CC1I

CC2I

CC3I

CC4I

U

UI

DTG registers

OC1N

OC1

OC2

TIMx\_CH1

OC2N

OC3

OC3N

OC4

Polarity selection

Clock failure event from the clock controller

CSS (Clock Security System)

TIMx\_BKIN

BI

TI1F\_ED

TI3FP3

TI3FP4

TI4FP3

TI4FP4

TIMx\_CH1N

TIMx\_CH2

TIMx\_CH2N

TIMx\_CH3

TIMx\_CH3N

TIMx\_CH4

Internal clock

CK\_TIM from RCC

Polarity selection, edge

Detector and prescaler

trigger

Controller

Slave mode

Controller

Encoder

interface

Reset, enable, increment */* decrement, count

To other timers

To DAC/ADC

Auto-reload register

REP register

Input filter and

Edge detector

Input filter and

Edge detector

Input filter and

Edge detector

Input filter and

Edge detector

Capture/Compare 1 register

Capture/Compare 3 register

Capture/Compare 2 register

Capture/Compare 4 registers

Output

control

Output

control

Output

control

Output

control

Stop, clear, or increase */* decrease

Capture/compare 5 registers

⌘˖

ṩᦞ᧗ࡦսˈ൘ਁ⭏ 8 һԦਾˈ亴㻵䖭ᇴᆈಘcheck ᇩ䖜〫ࡠᴹ᭸ᇴᆈಘ

һԦ

ѝᯝ઼'0$䗃ࠪ

U

Reg

CC5I

30.

**10.3**

**10.3.1**

The main part of the programmable advanced control timer is a 16-bit counter and its associated auto-load register. This

Each counter can count up, count down, or count up and down in both directions. This counter clock is divided by the prescaler

get.

The counter, auto-load register and prescaler register can be read and written by software, even if the counter is still running.

It works.

The time base unit contains:

• Counter register (TIMx\_CNT)

• Prescaler register (TIMx\_PSC)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

108/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 131** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

• Auto load register (TIMx\_ARR)

• Repeat number register (TIMx\_RCR)

The auto-reload register is pre-loaded, and writing or reading the auto-reload register will access the pre-load register. According to TIMx\_CR1

The setting of the autoload preload enable bit (ARPE) in the register, the content of the preload register is immediately or every time

The update event UEV is transferred to the shadow register. When the counter reaches an overflow condition (underflow condition when counting down)

And when the UDIS bit in the TIMx\_CR1 register is equal to 0, an update event is generated. Update events can also be produced by the software

Health. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK\_CNT of the prescaler, only if the counter in the TIMx\_CR1 register is set

CK\_CNT is valid only when the counter enables bit (CEN). (For more details on enabling the counter, see the controller

The description of the slave mode).

Note: The counter starts counting one clock cycle after the CEN bit of the TIMx\_CR register is set.

The prescaler can divide the counter clock frequency by any value between 1 and 65536. It is based on a (TIMx\_PSC

The 16-bit counter controlled by the 16-bit register in the register). Because this control register has a buffer, it can

It is changed at runtime. The parameters of the new prescaler are adopted when the next update event arrives.

The following two figures respectively give examples of changing the counter parameters while the prescaler is running.

810652

CK\_PSC

CEN

ᇊᰦಘᰦ䫏 = CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

02

03

0

1

1

0

0

0

0

0

0

1

1

1

1

TIMx\_PSCᇴᆈಘ

31.

1

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

109/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 132** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

646475

CK\_PSC

CEN

ᇊᰦಘᰦ䫏=CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

0

3

3

0

0

0

2

2

0

1

3

1

3

TIMx\_PSCᇴᆈಘ

32.

1

4

**10.3.2**

In the up-counting mode, the counter counts from 0 to the autoload value (the content of the TIMx\_ARR counter), and then restarts

Start counting from 0 and generate a counter overflow event.

If the repetition counter function is used, when the up count reaches the set repetition count (TIMx\_RCR), a

Update event (UEV); otherwise, the update event is generated every time the counter overflows.

Setting the UG bit in the TIMx\_EGR register (either through software or using a slave mode controller) can also produce

Create an update event.

Setting the UDIS bit in the TIMx\_CR1 register can disable the update event; this can avoid registering to the preload

The shadow register is updated when a new value is written in the device. Until the UDIS bit is cleared, no update event will be generated. But in response

When the update event occurs, the counter will still be cleared to 0, and the count of the prescaler is also requested to 0 (but the value of the prescaler

constant). In addition, if the URS bit in the TIMx\_CR1 register is set (select an update request), setting the UG bit will

An update event UEV is generated, but the UIF flag is not set by the hardware (that is, no interrupt or DMA request is generated). This is for

Avoid generating update and capture interrupts simultaneously when the counter is cleared in capture mode.

When an update event occurs, all registers are updated, and the hardware sets the update flag at the same time (according to the URS bit)

Bit (UIF bit in the TIMx\_SR register).

• The repetition counter is reloaded with the contents of the TIMx\_RCR register.

• The autoload shadow register is reset to the value of the preload register (TIMx\_ARR).

• The buffer of the prescaler is placed in the value of the preload register (the content of the TIMx\_PSC register).

The following figure shows some examples. When TIMx\_ARR = 0x36, the counter will operate at different clock frequencies.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

110/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 133** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

446234

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

31

32 33 34 35 36

05

Counter overflow

Update interrupt flag (UIF)

00 01 02 03 04

06 07

33.

1

036407

0034

0035

0000

0036

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

34.

2

959219

0035

0000

0036

0001

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

35.

4

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

111/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 134** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

345045

1F

20

00

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

36.

N

800181

Write new value in TIMx\_ARR

31

32

33 34

35

36

05

00 01

02

03

04

06

07

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

37.

ARPE = 0

(TIMx\_ARR

)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

112/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 135** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

175157

Write new value in TIMx\_ARR

F0

F1 F2 F3 F4 F5

05

00 01 02 03 04

06 07

F5

36

F5

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

38.

ARPE = 1

(TIMx\_ARR)

In down mode, the counter counts down to 0 from the automatically loaded value (TIMx\_ARR counter value), and then

It restarts from the automatically loaded value and generates a counter underflow event.

If a repeat counter is used, after counting down the number of times set in the repeat count register (TIMx\_RCR),

An update event (UEV) will be generated, otherwise the update event will be generated every time the counter underflows.

Setting the UG bit in the TIMx\_EGR register (either through software or using a slave mode controller) can also produce

Create an update event.

Setting the UDIS bit in the TIMx\_CR1 register can disable UEV events. This can avoid pre-loading registration

The shadow register is updated when a new value is written in the device. Therefore, no update event will be generated until the UDIS bit is cleared to 0. However,

The counter will still restart counting from the current auto-load value, and the prescaler counter restarts from 0 (but the prescaler

The frequency of the frequency converter cannot be modified).

In addition, if the URS bit in the TIMx\_CR1 register is set (select update request), setting the UG bit will generate a

Update event UEV without setting the UIF flag (so no interrupts and DMA requests are generated), this is to avoid

When the event is captured and the counter is cleared, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and (according to the setting of the URS bit) update the flag bit (TIMx\_SR

The UIF bit in the register) is also set.

• The repetition counter is reset to the contents of the TIMx\_RCR register.

• The prescaler buffer is loaded with the preloaded value (the value of the TIMx\_PSC register).

• The current autoload register is updated to the preload value (the content in the TIMx\_ARR register).

Note: Autoload is updated before the counter is reloaded, so the next cycle will be the expected value. Here are some when TIMx\_ARR

= 0x36, example of counter operation at different clock frequencies.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

113/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 136** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

878146

05

04

03

02

01

00

31

36

35

34

33

32

30

2F

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

39.

1

100845

0002

0001

0036

0000

0035

0034

0033

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

40.

2

247112

0001

0036

0000

0035

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

41.

4

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

114/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 137** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

044951

20

1F

00

36

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

42.

N

844225

05

04

03 02

01

00

31

36 35

34

33

32

30

2F

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

Write new value in TIMx\_ARR

43.

use

**(**

**/**

**)**

In center-aligned mode, the counter starts counting from 0 to the automatically loaded value (TIMx\_ARR register)-1, generating a

Counter overflow event, then count down to 1 and generate a counter underflow event; then start again from 0

count.

In this mode, the DIR direction bit in TIMx\_CR1 cannot be written. It is updated by the hardware and indicates the current counting method

to.

The update event can be generated at each count overflow and each count underflow; can also be controlled by (software or using slave mode

Controller) Set the UG bit in the TIMx\_EGR register to generate. At this time, the counter starts counting from 0 again, prescaler

The counter starts counting from 0 again.

Setting the UDIS bit in the TIMx\_CR1 register can disable UEV events. This can avoid pre-loading registration

The shadow register is updated when a new value is written in the device. Therefore, no update event will be generated until the UDIS bit is cleared to 0. However,

The counter will continue to count up or down based on the current auto-reload value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

115/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 138** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

In addition, if the URS bit in the TIMx\_CR1 register is set (select update request), setting the UG bit will generate a

Update event UEV without setting the UIF flag (so no interrupts and DMA requests are generated), this is to avoid

When the event is captured and the counter is cleared, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and (according to the setting of the URS bit) update the flag bit (TIMx\_SR

The UIF bit in the register) is also set.

• The repetition counter is reset to the contents of the TIMx\_RCR register.

• The prescaler buffer is loaded with the preload (TIMx\_PSC register) value.

• The current autoload register is updated to the preload value (the content in the TIMx\_ARR register).

Note: If an update occurs due to a counter overflow, the automatic reload will be updated before the counter is reloaded, so the next cycle

It will be the expected value (the counter is loaded with the new value).

The following are some examples of counter operations at different clock frequencies:

519069

04

03 02 01 00 01

05

02 03 04 05 06

04 03

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Counter underflow

44.

1TIMx\_ARR = 0x6

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

116/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 139** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

897437

0003

0002

0000

0001

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

45.

2

365016

0034

0036

0035

0035

Note: Center-aligned mode 2 or mode 3 is used in conjunction with the overflow UIF.

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

46.

4TIMx\_ARR = 0x36

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

117/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 140** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

918695

20

1F

00

01

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

47.

N

608655

06

05

04 03

02

01

05

00

01 02

03

04

06

07

FD

36

FD

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

Write new value in TIMx\_ARR

48.

ARPE = 1

()

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

118/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 141** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

712179

F7

F8

F9 FA FB FC

31

36

35 34

33

32

30

2F

FD

36

FD

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

Write new value in TIMx\_ARR

49.

ARPE = 1

()

**10.3.3**

The'time base unit' explains how the update event (UEV) is generated when the counter overflows/underflows, but in fact it can only

Generated when the repeat count reaches 0. This feature is very useful for generating PWM signals.

This means that every N count overflows or underflows, the data is transferred from the preload register to the shadow register (TIMx\_ARR

Automatic reload register, TIMx\_PSC preload register, and capture/compare register in compare mode (TIMx\_CCRx),

N is

TIMx\_RCR repeats the value in the register.

The repetition counter decrements when any of the following conditions are true:

• Each time the counter overflows in up-counting mode,

• Each time the counter underflows in down-counting mode,

• Every time overflow and underflow in center alignment mode. Although this limits the maximum PWM cycle to 128,

However, it can update the duty cycle twice per PWM cycle. In center-aligned mode, because the waveform is symmetrical, such as

If the compare register is refreshed only once per PWM cycle, the maximum resolution is 2xTck.

The repetition counter is automatically loaded, and the repetition rate is defined by the value of the TIMx\_RCR register (see Figure [50](https://translate.googleusercontent.com/translate_f#142) ). When more

New events are generated by software (by setting the UG bit in TIMx\_EGR) or by hardware slave mode controller,

No matter what the value of the repetition counter is, an update event occurs immediately, and the content of the TIMx\_RCR register is reset

Load into the repeat counter.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

119/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 142** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

848184

counter

TIMX\_CNT

TIMX\_RCR=0 UEV

UEV

UEV

UEV

UEV

And resynchronize

UEV

(By software)

(By software)

(By software)

Update time:

The preload register is transferred to the active register and an update interrupt is generated.

If the repetitive counter underflow occurs when the counter is equal to the auto-reload value, the event is updated.

Center alignment mode

Edge alignment mode

Count up

Count down

TIMX\_RCR=1

TIMX\_RCR=2

TIMX\_RCR=3

TIMX\_RCR=3

50.

TIMx\_RCR

**10.3.4**

The counter clock can be provided by the following clock sources:

• Internal clock (CK\_INT).

• External clock mode 1: External input pin (TIx).

• External clock mode 2: External trigger input (ETR).

• Internal trigger input (ITRx): use one timer as the prescaler of another timer, if you can configure one

Timer1 acts as a prescaler for another timer, Timer2.

**(CK\_INT)**

If the slave mode controller is disabled (SMS = 000), the CEN, DIR (TIMx\_CR1 register) and UG bits (TIMx\_EGR

(Register) is the de facto control bit and can only be modified by software (UG bit is still automatically cleared). When the CEN bit is written

When set to 1, the prescaler clock is provided by the internal clock CK\_INT.

The figure below shows the operation of the control circuit and the up-counter in normal mode without the prescaler.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

120/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 143** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

397736

Internal clock

CEN = CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

CNT\_INT

UG

31

32

33 34

35

36

05

00

01

02

03

04

06

07

51.

1

**1**

When SMS = 111 in the TIMx\_SMCR register, this mode is selected. The counter can be

Count on a rising or falling edge.

756508

Encoder mode

External clock

Mode 1

External clock

Mode 2

Internal clock

mode

CK\_PSC

filter

ICF[3:0]

edge

Detector

CC2P

TS[2:0]

SMS[2:0]

ECE

TI2

TIMx\_CCMR1

TIMx\_CCER

TI2F\_Rising

TI2F\_Falling

ITRx

TI1F\_ED

TI1FP1

TI2FP2

ETRF

001

100

101

110

111

TRGI

ETRF

CK\_INT

(Internal clock)

TIMx\_SMCR

TI2F

TI1F

or

or

or

TIMx\_SMCR

0

1

52. TI2

For example, to configure the up counter to count on the rising edge of the T12 input, use the following steps:

1. Configure the TIMx\_CCMR1 register CC2S = 01, and configure channel 2 to detect the rising edge of the TI2 input.

2. Configure IC2F[3:0] in the TIMx\_CCMR1 register and select the input filter bandwidth (if the filter is not required, ensure

Hold IC2F = 0000).

3. Configure CC2P = 0 of the TIMx\_CCER register to select the rising edge polarity.

4. Configure SMS = 111 in the TIMx\_SMCR register to select timer external clock mode 1.

5. Configure TS = 110 in the TIMx\_SMCR register and select TI2 as the trigger input source.

6. Set CEN = 1 in the TIMx\_CR1 register to start the counter.

Note: The capture prescaler is not used as a trigger, so there is no need to configure it. When the rising edge appears on TI2, the counter counts once,

And the TIF flag is set. The delay between the rising edge of TI2 and the actual clock of the counter depends on the resynchronization at the input of TI2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

121/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 144** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

Circuit.

274714

TI2

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

Write TIF = 0

34

36

TIF

35

53.

1

**2**

To select this mode, set ECE = 1 in the TIMx\_SMCR register.

The counter can externally trigger every rising or falling edge of ETR to count.

The following figure is the overall block diagram of the external trigger input:

160574

Encoder mode

External clock

Mode 1

External clock

Mode 2

Internal clock

mode

CK\_PSC

filter

Down counter

Crossover

/1,/2,/4,/8

SMS[2:0]

ECE

ETR pin

ETR

TRGI

ETRF

CK\_INT

(Internal clock)

TI2F

TI1F

or

or

or

0

1

ETP

ETRP

CK\_INT

ETF[3:0]

ETPS[1:0]

TIMx\_SMCR

TIMx\_SMCR

TIMx\_SMCR

TIMx\_SMCR

54.

For example, to configure an up counter that counts every 2 rising edges under ETR, use the following steps:

• No filter is required in this example, set ETF[3:0] = 0000 in the TIMx\_SMCR register

• Set the prescaler and set ETPS[1:0] = 01 in the TIMx\_SMCR register

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

122/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 145** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

• Select rising edge detection of ETR, set ETP = 0 in TIMx\_SMCR register

• Turn on external clock mode 2 and write ECE = 1 in the TIMx\_SMCR register

• Start the counter and write CEN = 1 in the TIMx\_CR1 register

The counter counts every 2 rising edges of ETR.

The delay between the rising edge of ETR and the actual clock of the counter depends on the resynchronization circuit at the ETRP signal

096922

ETRP

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

34

36

ETRF

35

ETR

f CK\_INT

55.

2

**10.3.5**

**/**

Each capture/compare channel surrounds a capture/compare register (including shadow registers), including the captured input

Part (digital filtering, multiplexing and prescaler), and output part (comparator and output control).

[Figures](https://translate.googleusercontent.com/translate_f#146) 56 to 59 are an overview of the capture/compare channels.

The input section samples the corresponding TIx input signal and generates a filtered signal TIxF. Then, one with polarity

The selected edge monitor generates a signal (TIxFPx), which can be used as an input trigger from the mode controller or as

Capture control. This signal enters the capture register (ICxPS) through prescaler.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

123/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 146** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

468148

IC1PS

filter

Down counter

ICF[3:0]

edge

Detector

CC1P

ICPS[1:0]

TI1

TIMx\_CCMR1

TIMx\_CCER

TI1F\_Rising

TI1F\_Falling

01

10

11

(Since mode controller)

0

1

f DTS

TI1F

0

1

divider

/1,/2,/4,/8

IC1

CC1E

CC1S[1:0]

TIMx\_CCMR1

TIMx\_CCER

TRC

TI2F\_Rising

TI2F\_Falling

(from channel 2)

(from channel 2)

TI2FP1

TI1FP1

TI1F\_ED

To slave mode controller

56.

/ ( 1

)

523832

MCU peripheral interface

APB bus

Capture/compare preload registers

high

8

CC1S[1]

OC1PE

TIMx\_CCMR1

UEV

Reading CCR1L

Reading CCR1H

CC1S[1]

CC1S[0]

IC1PS

CC1E

CC1G

TIMx\_EGR

read\_in\_progress

enter

mode

8

low

(Assuming 16 bits)

Capture/Compare Shadow Register

capture

counter

Comparators

capture\_transfer

compare\_transfer

write\_in\_progress

Output

mode

CNT> CCR1

CNT = CCR1

(From time base unit)

Write to CCR1H

Write to CCR1L

CC1S[0]

OC1PE

S

R

S

R

57.

/ 1

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

124/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 147** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

180377

Output mode

Controller

OC1M[2:0]

TIMx\_CCMR1

CNT>CCR1

CNT=CCR1

ETRF

OC1ref

To slave mode controller

0

1

CC1P

TIMx\_CCER

Output enable

Circuit

CC1E TIMx\_CCER

OC1

58.

/ ( 1

3)

062055

Output mode

Controller

OC2M[2:0]

TIMx\_CCMR2

CNT>CCR4

CNT=CCR4

ETRF

OC4ref

To slave mode controller

0

1

CC4P

TIMx\_CCER

Output enable

Circuit

CC4E

TIMx\_CCER

OC4

MOE

OIS4

TIMx\_BDTR

TIM1\_CR2

OSSI

59.

/ (4)

The capture/compare module consists of a preload register and a shadow register. The read and write process only operates preload registration

Device. In capture mode, the capture occurs on the shadow register, and then copied into the preload register.

In the comparison mode, the contents of the preload register are copied to the shadow register, and then the contents of the shadow register and

Counter.

**10.3.6**

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched to capture/ratio

Compare register (TIMx\_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx\_SR register)

Set to 1, if an interrupt or DMA operation is enabled, an interrupt or DMA request will be generated. If a capture event occurs

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

125/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 148** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

When the CCxIF flag is already high, the repeated capture flag CCxOF (TIMx\_SR register) is set. Write CCxIF

= 0 can clear CCxIF, or read the captured data stored in the TIMx\_CCRx register can also clear CCxIF. write

CCxOF = 0 can clear CCxOF.

The following example shows how to capture the counter value into the TIMx\_CCR1 register on the rising edge of the TI1 input, step

as follows:

• Select valid input: TIMx\_CCR1 must be connected to the TI1 input, so write to the TIMx\_CCMR1 register

CC1S = 01, when CC1S is not 00, the channel is configured as an input, and the TIMx\_CCR1 register becomes

Is read-only.

• According to the characteristics of the input signal, configure the input filter to the required bandwidth (that is, when the input is TIx, the input filter controls

The control bit is the ICxF bit in the TIMx\_CCMRx register). Assuming that the input signal is at most 5 clock cycles

Internal jitter, we must configure the bandwidth of the filter to be longer than 5 clock cycles; therefore we can connect (at fDTS frequency)

Continue sampling 8 times to confirm the last real edge transition on TI1, which is written in the TIMx\_CCMR1 register

IC1F = 0011.

• Select the valid conversion edge of the TI1 channel and write CC1P = 0 (rising edge) in the TIMx\_CCER register.

• Configure input prescaler. In this example, we want to capture every valid level-shifting moment, so

The prescaler is disabled (write IC1PS = 00 in the TIMx\_CCMR1 register).

• Set CC1E = 1 in the TIMx\_CCER register to allow the capture counter value to be captured in the capture register.

• If necessary, enable the relevant interrupt request by setting the CC1IE bit in the TIMx\_DIER register, by setting

The CC1DE bit in the TIMx\_DIER register allows DMA requests.

When an input capture occurs:

• When a valid level transition occurs, the counter value is transferred to the TIMx\_CCR1 register.

• The C1IF flag is set (interrupt flag). When at least 2 consecutive captures occur, and CC1IF has not been cleared,

CC1OF is also set to 1.

• If the CC1IE bit is set, an interrupt will be generated.

• If the CC1DE bit is set, a DMA request will also be generated.

In order to deal with the capture overflow, it is recommended to read the data before reading the capture overflow flag, this is to avoid losing the read capture

Capture overflow information that may be generated after the overflow flag is obtained and before the data is read.

Note: By setting the corresponding CCxG bit in the TIMx\_EGR register, an input capture interrupt and/or DMA request can be generated by software.

**10.3.7 PWM**

This mode is a special case of the input capture mode, except for the following differences, the operation is the same as the input capture mode:

• Two ICx signals are mapped to the same TIx input.

• The two ICx signals are edge-active, but have opposite polarities.

• One of the TIxFP signals is used as a trigger input signal, and the slave mode controller is configured in reset mode. For example, you

Need to measure the length of the PWM signal input to TI1 (TIMx\_CCR1 register) and duty cycle (TIMx\_CCR2

Register), the specific steps are as follows (depending on the frequency of CK\_INT and the value of prescaler)

• Select the valid input of TIMx\_CCR1: Set CC1S = 01 in the TIMx\_CCMR1 register (TI1 selected).

• Select the effective polarity of TI1FP1 (used to capture data into TIMx\_CCR1 and clear the counter): set CC1P =

0 (rising edge valid).

• Select the valid input of TIMx\_CCR2: Set CC2S = 10 in the TIMx\_CCMR1 register (select TI1).

• Select the effective polarity of TI1FP2 (capture data to TIMx\_CCR2): set CC2P = 1 (active on falling edge).

• Select a valid trigger input signal: set TS = 101 in the TIMx\_SMCR register (select TI1FP1).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

126/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 149** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

• Configure the slave mode controller to reset mode: set SMS = 100 in TIMx\_SMCR.

• Enable capture: Set CC1E = 1 and CC2E = 1 in the TIMx\_CCER register.

065889

0004

0000

0001

0002

0003

0004

0000

0004

0002

TIMx\_CNT

TIMx\_CCR1

TIMx\_CCR2

TI1

IC1 capture

IC2 capture

Reset counter

IC2 capture

Pulse width measurement

IC1 capture

Period measurement

60. PWM

Because only TI1FP1 and TI2FP2 are connected to the slave mode controller, the PWM input mode can only use TIMx\_CH1/TIMx\_CH2

signal.

**10.3.8**

In the output mode (CCxS = 00 in the TIMx\_CCMRx register), the comparison signal (OCxREF and the corresponding

OCx/OCxN) can be forced into valid or invalid state directly by software without relying on output compare register and count

Comparison results between devices.

Set the corresponding OCxM = 101 in the TIMx\_CCMRx register to force the output compare signal (OCxREF/OCx)

Is valid. In this way, OCxREF is forced to high level (OCxREF is always active high), and OCx is

To CCxP signal with opposite polarity.

For example: CCxP = 0 (OCx high level effective), then OCx is forced to high level.

Set OCxM = 100 in the TIMx\_CCMRx register to force the OCxREF signal low.

In this mode, the comparison between the TIMx\_CCRx shadow register and the counter is still in progress, and the corresponding flag will also

modified. Therefore, corresponding interrupts and DMA requests will still be generated. This will be in the output comparison mode section below

Introduction.

**10.3.9**

This function is used to control an output waveform or indicate when a given time has expired.

When the contents of the counter and the capture/compare register are the same, the output compare function does the following:

• Register the output compare mode (OCxM bit in the TIMx\_CCMRx register) and output polarity (TIMx\_CCER

The value defined by the CCxP bit in the device is output to the corresponding pin. The output pin can hold it when comparing and matching

Level (OCxM = 000), set to the active level (OCxM = 001), set to no active level (OCxM

= 010) or roll over (OCxM = 011).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

127/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 150** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

• Set the flag bit in the interrupt status register (CCxIF bit in the TIMx\_SR register).

• If the corresponding interrupt mask (CCxIE bit in the TIMx\_DIER register) is set, an interrupt is generated.

• If the corresponding enable bit is set (CCxDE bit in the TIMx\_DIER register, the TIMx\_CR2 register

CCDS bit selects DMA request function), then generate a DMA request.

The OCxPE bit in TIMx\_CCMRx selects whether the TIMx\_CCRx register needs to use a preload register.

In output comparison mode, the update event UEV has no effect on the OCxREF and OCx outputs.

The accuracy of synchronization can reach one counting cycle of the counter. Output compare mode (in single pulse mode) can also be used to output

A single pulse is output.

Configuration steps of output comparison mode:

• Select counter clock (internal, external, prescaler)

• Write the corresponding data to the TIMx\_ARR and TIMx\_CCRx registers

• If an interrupt request is to be generated, set the CCxIE bit

• Select the output mode, for example:

**-** Toggle OCx CCRx match the when required and at The counter Provided the Output pins OCxM = 011

**–** Set OCxPE = 0 to disable the preload register

**–** Set CCxP = 0 to select the polarity as active high

**-** the SET to enable the Output at The CCxE = 1

• Set the CEN bit of the TIMx\_CR1 register to start the counter

The TIMx\_CCRx register can be updated by software at any time to control the output waveform, provided the pre-use is not used

Load register (OCxPE = '0', otherwise the shadow register of TIMx\_CCRx can only be updated in the next event

Is updated). The figure below gives an example.

484761

0039

003A

003B

B200

B201

003A

B201

Write B201h in CCR1 register

A match was detected on CCR1

If the interrupt is enabled, an interrupt is generated

TIMX\_CNT

TIMX\_CCR1

OC1REF=OC1

61.

OC1

**10.3.10 PWM**

Pulse width modulation mode can generate a frequency determined by the TIMx\_ARR register and confirmed by the TIMx\_CCRx register

A signal with a fixed duty cycle.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

128/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 151** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

Write '110' (PWM mode 1) or '111' (PWM mode) in the OCxM bit in the TIMx\_CCMRx register

2), each OCx output channel can be set independently to generate a PWM. Must be sent by setting TIMx\_CCMRx

The OCxPE bit of the register enables the corresponding preload register, and finally the ARPE bit of the TIMx\_CR1 register must be set

Enable auto-reloading of preload registers (in up-count or center-symmetric mode).

Because only when an update event occurs, the preload register can be transferred to the shadow register, so in the count

Before the device starts counting, all registers must be initialized by setting the UG bit in the TIMx\_EGR register.

The polarity of OCx can be set by the CCxP bit in the TIMx\_CCER register by software, it can be set to high level

Active or active low. OCx output enable pass (in TIMx\_CCER and TIMx\_BDTR registers) CCxE

, CCxNE, MO E, OSSI and OSSR bit combination control. See the description of the TIMx\_CCER register for details.

In PWM mode (mode 1 or mode 2), TIMx\_CNT and TIMx\_CCRx are always compared, (based on the count

The counting direction of the device) to determine whether it meets TIMx\_CCRx ≤ TIMx\_CNT or TIMx\_CNT ≤ TIMx\_CCRx.

Depending on the state of the CMS bit in the TIMx\_CR1 register, the timer can generate an edge-aligned PWM signal or center

Aligned PWM signal.

**PWM**

Counting up is performed when the DIR bit in the TIMx\_CR1 register is low. See section [10.3.2](https://translate.googleusercontent.com/translate_f#132) .

The following is an example of PWM mode 1. When TIMx\_CNT <TIMx\_CCRx, the PWM reference signal OCxREF

High, otherwise low. If the comparison value in TIMx\_CCRx is greater than the auto-reload value (TIMx\_ARR), then OCxREF

Keep it as '1'. If the comparison value is 0, OCxREF remains at '0'. Figure [62](https://translate.googleusercontent.com/translate_f#151) shows the edge when TIMx\_ARR = 8

Example of aligned PWM waveforms.

652093

0 1 2 3 4 5 6 7 8 0 1

'1'

'0'

Counter register

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx=4

CCRx=8

CCRx>8

CCRx=0

62.

PWM

(ARR = 8)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

129/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 152** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

Countdown is performed when the DIR bit of the TIMx\_CR1 register is high. See section [10.3.2](https://translate.googleusercontent.com/translate_f#132) .

In PWM mode 1, the reference signal OCxREF is low when TIMx\_CNT> TIMx\_CCRx, otherwise it is high. Such as

If the comparison value in TIMx\_CCRx is greater than the auto-reload value in TIMx\_ARR, OCxREF remains at '1'.

In this mode, a 0% PWM waveform cannot be generated.

**PWM**

When the CMS bit in the TIMx\_CR1 register is not '00', it is the center-aligned mode (all other configuration pairs are OCxREF/OCx

Signals have the same effect). Depending on the setting of the different CMS bits, the comparison flag can be used when the counter is counting up

Set to 1, set when the counter counts down, or set to '1' when the counter counts up and down. TIMx\_CR1 mail

The count direction bit (DIR) in the memory is updated by hardware. Do not modify it by software. See section [10.3.](https://translate.googleusercontent.com/translate_f#132) 2 for center alignment

mode.

[Figure](https://translate.googleusercontent.com/translate_f#152) 63 shows some examples of center-aligned PWM waveforms

• TIMx\_ARR = 8

• PWM mode 1

• CMS = 01 in the TIMx\_CR1 register, in center-aligned mode 1, the comparison is set when the counter counts down

Sign

931803

0 1 2 3 4 5 6 7 8 7 6

'1'

'0'

Counter register

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx=4

CCRx=8

CCRx>8

CCRx=0

CMS=01

CMS=10

CMS=11

CMS=10 or 11

'1'

5 4 3 2 1 0 1

CCRx=7

OCxREF

CCxIF

CMS=01

CMS=10

CMS=11

CMS=01

CMS=10

CMS=11

CMS=01

CMS=10

CMS=11

63.

PWM

(APR = 8)

use

• When entering the center alignment mode, the current up/down counting configuration is used; this means whether the counter is counting up or down

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

130/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 153** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

It depends on the current value of the DIR bit in the TIMx\_CR1 register. In addition, the software cannot modify DIR and CMS at the same time

Bit.

• It is not recommended to rewrite the counter when running in center-aligned mode, as it will produce unpredictable results. specifically :

**–** If the value written to the counter is greater than the value of auto-reload (TIMx\_CNT> TIMx\_ARR), the direction

Will not be updated

**–** For example, if the counter is counting up, it will continue to count up

**–** If the value of 0 or TIMx\_ARR is written to the counter, the direction is updated, but no update event is generated

UEV

• The safest way to use the center alignment mode is to generate a software update (set TIMx\_EGR before starting the counter)

UG bit), do not modify the counter value while counting is in progress.

**10.3.11**

The advanced control timer (TIM1) can output two complementary signals, and can manage the instantaneous turn-off and turn-on of the output. This

The period of time is usually called a dead zone, and the user should use the connected output devices and their characteristics (level-delayed delay, power

Source switch delay, etc.) to adjust the dead time.

Configure the CCxP and CCxNP bits in the TIMx\_CCER register to independently select the polarity for each output (main

Output OCx or complementary output OCxN).

The complementary signals OCx and OCxN are controlled by a combination of the following control bits: CCxE of the TIMx\_CCER register

And CCxNE bits, MOE, OISx, OISxN, OSSI and in the TIMx\_BDTR and TIMx\_CR2 registers

For OSSR bits, see Table 38 Control bits for complementary output channels OCx and OCxN with brake function. In particular, in

When transitioning to IDLE state (MOE drops to 0) the dead zone is activated.

Setting the CCxE and CCxNE bits at the same time will insert a dead zone, and if there is a brake circuit, then also set the MOE bit. each

Each channel has a 10-bit dead zone generator. The reference signal OCxREF can generate 2 output OCx and

OCxN. If OCx and OCxN are high effective:

• The OCx output signal is the same as the reference signal, except that its rising edge has a delay relative to the rising edge of the reference signal.

• The OCxN output signal is opposite to the reference signal, except that its rising edge has a delay relative to the falling edge of the reference signal.

If the delay is greater than the currently valid output width (OCx or OCxN), no corresponding pulse will be generated.

The following figures show the relationship between the output signal of the dead zone generator and the current reference signal OCxREF. (Assuming

CCxP = 0, CCxNP = 0, MOE = 1, CCxE = 1 and CCxNE = 1).

376699

delay

delay

OCxN

OCx

OCxREF

64.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

131/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 154** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

393374

delay

OCxN

OCx

OCxREF

65.

118234

delay

OCxN

OCx

OCxREF

66.

The dead time delay of each channel is the same, which is configured by the DTG bit in the TIMx\_BDTR register. detailed

See [section 10.4.18](https://translate.googleusercontent.com/translate_f#191) for delay calculation.

**OCxREF**

**OCx**

**OCxN**

In output mode (forced, output compare or PWM), by configuring CCxE and CCxNE of the TIMx\_CCER register

Bit, OCxREF can be redirected to the output of OCx or OCxN.

This function can send a special waveform (such as PWM) on an output when the complementary output is at an invalid level

Or static effective level). Another function is to make the two outputs at the inactive level at the same time, or at the effective level and band

Complementary output of dead zone.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it will not be inverted, and will go high immediately when OCxREF is active. Eg,

If CCxNP = 0, then OCxN = OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE = CCxNE

= 1), OCx is valid when OCxREF is high, and OCxN is the opposite, OCxN becomes valid when OCxREF is low.

**10.3.12**

use

When using the brake function, according to the corresponding control bits (MOE, OSSI and OSSR in the TIMx\_BDTR register

Bit, the OISx and OISxN bits in the TIMx\_CR2 register), the output enable signal and the invalid level will be modified. but

At any time, the OCx and OCxN outputs cannot be at the same time at the same time. For details, please refer to the register table with brake

Control bits for the complementary output channels OCx and OCxN of the car function.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

132/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 155** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

The brake source can be either a brake input pin or a clock failure event. Clock failure events are controlled by reset clock

The clock security system in the device is generated.

After the system is reset, the brake circuit is disabled and the MOE bit is low. Setting the BKE bit in the TIMx\_BDTR register can enable

Can brake function. The polarity of the brake input signal can be selected by configuring the BKP bit in the same register. BKE and

BKP can be modified at the same time.

Because the falling edge of MOE can be asynchronous, the actual signal (acting at the output) and the synchronization control bit (at TIMx\_BDTR

There is a resynchronization circuit between the registers). This resynchronization circuit will produce between the asynchronous signal and the synchronous signal

Health delay. In particular, if MOE=1 is written when it is low, a delay (null instruction) must be inserted before reading it

In order to read the correct value. This is because asynchronous signals are written and synchronous signals are read.

When braking occurs (the selected level appears at the brake input), there are the following actions:

• The MOE bit is asynchronously cleared, placing the output in an inactive state, an idle state, or a reset state (selected by the OSSI bit).

This feature is still effective when the MCU oscillator is turned off.

• When MOE = 0, the output level of each output channel is determined by the OISx bit in the TIMx\_CR2 register. Such as

If OSSI = 0, the timer releases the enable output, otherwise the enable output is always high.

• When using complementary outputs:

**– The** output is first put into a reset state, that is, an invalid state (depending on polarity). This is an asynchronous operation, even if

This function is also effective when the timer has no clock.

**–** If the clock of the timer still exists, the dead zone generator will take effect again, according to OISx after the dead zone

And the level indicated by the OISxN bit drive the output port. Even in this case, OCx and OCxN

It cannot be driven to a valid level at the same time. Note, because the MOE is resynchronized, the dead time is longer than usual

Make it longer (about 2 CK\_TIM clock cycles).

**–** If OSSI = 0, the timer releases the enable output, otherwise keeps the enable output; or when CCxE and CCxNE

When one of them goes high, the enable output goes high.

• If the BIE bit in the TIMx\_DIER register is set, the brake status flag (BIF in the TIMx\_SR register

When bit) is '1', an interrupt is generated. If the BDE bit in the TIMx\_DIER register is set, it generates

One DMA request.

• If the AOE bit in the TIMx\_BDTR register is set, the MOE bit is set by the next update event UEV

Set automatically; for example, this can be used for shaping. Otherwise, MOE remains low until it is set to '1' again; at this time,

This feature can be used for safety, you can connect the brake input to the power supply-driven alarm output, thermal sensing

Device or other security device.

Note: The brake input is level effective. Therefore, when the brake input is valid, the MOE cannot be set simultaneously (either automatically or through software).

At the same time, the status flag BIF cannot be cleared.

The brake is generated by the BRK input, its effective polarity is programmable, and is determined by the BKE bit in the TIMx\_BDTR register

Open.

In addition to brake input and output management, write protection is also implemented in the brake circuit to ensure the safety of the application. It allows

The user freezes several configuration parameters (dead zone length, OCx/OCxN polarity and prohibited status, OCxM configuration, brake enable

Energy and polarity). The user can select one of the three levels of protection through the LOCK bit in the TIMx\_BDTR register.

See [section](https://translate.googleusercontent.com/translate_f#183) 10.4.8. The LOCK bit can only be modified once after the MCU is reset.

The following figure shows an example of output in response to braking:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

133/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 156** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

844716

OCxN

OCx

OCxREF

(OCxN is not used, CCxp=0, OISx=1)

(OCxN is not used, CCxp=0, OISx=0)

OCx

OCx

(OCxN is not used, CCxp=1, OISx=1)

OCx

(OCxN is not used, CCxp=1, OISx=1)

OCx

delay

delay

delay

delay

delay

delay

(CCxE=1, CCxP=0, OISx=0, CCxNE=1, CCxNP=0, OISxN=1)

OCxN

OCx

OCx

OCxN

delay

delay

OCx

OCxN

OCx

OCxN

BREAK(MOE)

(CCxE=1, CCxP=0, OISx=1, CCxNE=1, CCxNP=1, OISxN=1)

(CCxE=1, CCxP=0, OISx=0, CCxNE=0, CCxNP=0, OISxN=1)

(CCxE=1, CCxP=0, CCxNP=0, OISx=OISxN=0 or OISx=OISxN=1)

(CCxE=1, CCxP=0, OISx=1, CCxNE=0, CCxNP=0, OISxN=0)

67.

**10.3.13**

**OCxREF**

For a given channel, at the ETRF input (set the corresponding OCxCE bit in the TIMx\_CCMRx register

High level of '1') can pull the OCxREF signal low, the OCxREF signal will remain low until the next occurrence

Update event UEV.

This function can only be used for output comparison and PWM mode, but not for forced mode.

For example, the OCxREF signal can be connected to an external input. At this time, ETR must be configured as follows:

• The external trigger prescaler must be turned off: ETPS[1:0] = 00 in the TIMx\_SMCR register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

134/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 157** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

• External clock mode 2: ECE = 0 in the TIMx\_SMCR register must be disabled.

• External trigger polarity (ETP) and external trigger filter (ETF) can be configured as required.

The figure below shows the behavior of the OCxREF signal corresponding to different OCxCE values ​​​​when the ETRF input goes high. At this

In this example, the timer TIMx is placed in PWM mode.

769364

Counter (CNT)

(CCRx)

ETRF

OCxREF

(OCxCE='0')

OCxREF

(OCxCE='1')

OCREF\_CLR

OCREF\_CLR

Go high

Keep high

68.

TIMx

OCxREF

**10.3.14**

**PWM**

When complementary output is required on one channel, the preload bits are OCxM, CCxE and CCxNE. Happening com

During the commutation event, these preload bits are transferred to the shadow register bits. So that you can pre-set the next step

And modify the configuration of all channels at the same time. COM can be set by setting the TIMx\_EGR register

The COM bit is generated by software, or generated by hardware on the rising edge of TRGI.

When a COM event occurs, a flag (COMIF bit in the TIMx\_SR register) is set. If it is set at this time

If the COMIE bit of the TIMx\_DIER register is generated, an interrupt is generated; if the TIMx\_DIER register has been set

The COMDE bit generates a DMA request.

The following figure shows the OCx and OCxN output in three different configurations when a COM event occurs.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

135/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 158** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

186221

Counter (CNT)

(CCRx)

Write COM to 1

Write 100 to OCxM

CCxE=1

CCxNE=0

OCxM=100 (Forced invalid)

CCxE=1

CCxNE=0

OCxM=100

CCxE=1

CCxNE=0

OCxM=100 (Forced invalid)

Write 1 to OCxNE

And write 101 to OCxM

CCxE=0

CCxNE=1

OCxM=101

Write 0 to OCxNE

And write 100 to OCxM

CCxE=1

CCxNE=0

OCxM=100

OCxREF

COM event

OCx

OCxN

OCx

OCxN

OCx

OCxN

Example 1

Example 2

Example 3

CCxE=1

CCxNE=0

OCxM=100 (Forced invalid)

69.

COM for PWM

(OSSR = 1)

**10.3.15**

The single pulse mode (OPM) is a special case of the aforementioned many modes. This mode allows the counter to respond to an stimulus and

A programmable pulse is generated after a programmable delay.

The counter can be started by the slave mode controller to generate a waveform in the output comparison mode or PWM mode. Set up

The OPM bit in the TIMx\_CR1 register will select the one-shot mode, which allows the counter to automatically generate the next pulse

Stopped during an update event UEV.

A pulse can be generated only when the comparison value is different from the initial value of the counter. Before starting (when the timer is waiting for a touch

Issue), must be configured as follows:

• Up counting method: counter CNT <CCRx ≤ ARR (in particular, 0 <CCRx)

• Down counting method: counter CNT> CCRx

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

136/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 159** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

947210

OC1

TI2

OC1REF

t DELAY

t PULSE

t

0

TIMx\_ARR

TIMx\_CCR1

count

Device

70.

For example, you need to detect a rising edge on the input pin of TI2, delay t DELAY, and then generate it on OC1.

Generate a positive pulse of length t PULSE .

Assume TI2FP2 as trigger 1:

• Set CC2S = 01 in the TIMx\_CCMR1 register to map TI2FP2 to TI2.

• Set CC2P = 0 in the TIMx\_CCER register to enable TI2FP2 to detect the rising edge.

• Set TS = 110 in the TIMx\_SMCR register to use TI2FP2 as the trigger (TRGI) of the slave mode controller.

• Set SMS = 110 (trigger mode) in the TIMx\_SMCR register and TI2FP2 is used to start the counter.

The OPM waveform is determined by the value written to the compare register (the clock frequency and counter prescaler must be considered)

• t DELAY is defined by the value in the TIMx\_CCR1 register.

• t PULSE is defined by the difference between the autoload value and the comparison value (TIMx\_ARR-TIMx\_CCR1).

• Assume that a waveform from 0 to 1 is generated when a compare match occurs, and a slave is generated when the counter reaches the preload value

1 to 0 waveform; first set OC1M = 111 in TIMx\_CCMR1 register to enter PWM mode 2; root

Selectively enable preload registers as needed: set OC1PE = 1 and TIMx\_CR1 in TIMx\_CCMR1

ARPE in the register; then fill in the comparison value in the TIMx\_CCR1 register, in the TIMx\_ARR register

Fill in the autoload value, set the UG bit to generate an update event, and then wait for an external trigger on TI2

event. In this example, CC1P = 0.

In this example, the DIR and CMS bits in the TIMx\_CR1 register should be set low.

Because only one pulse is required, OPM = 1 in the TIMx\_CR1 register must be set, and the next update event

(When the counter rolls over from the autoload value to 0) stop counting.

**OCx**

In the one-shot mode, the edge detection logic on the TIx input pin sets the CEN bit to start the counter. Then the counter and

The comparison operation between the comparison values ​​​​produces a conversion of the output. But these operations require a certain clock cycle, so it limits

The minimum delay t DELAY available .

If you want to output the waveform with minimum delay, you can set the OCxFE bit in the TIMx\_CCMRx register;

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

137/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 160** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

OCxREF (and OCx) directly respond to the excitation and no longer depend on the result of the comparison, the output waveform matches the waveform when the comparison

same. OCxFE only works when the channel is configured in PWM1 and PWM2 modes.

**10.3.16**

The method to select the encoder interface mode is: if the counter only counts on the edge of TI2, set TIMx\_SMCR to register

SMS = 001 in the device; if only counting on the edge of TI1, set SMS = 010; if the counter is in both TI1 and TI1

TI2 edge count, then set SMS = 011.

The polarity of TI1 and TI2 can be selected by setting the CC1P and CC2P bits in the TIMx\_CCER register; eg

If required, the input filter can also be programmed. The two inputs TI1 and TI2 are used as the interface for the incremental encoder.

Referring to Table [35](https://translate.googleusercontent.com/translate_f#160) , assuming that the counter has started (CEN = 1 in the TIMx\_CR1 register), the counter

Active transition drive on TI1FP1 or TI2FP2. TI1FP1 and TI2FP2 are TI1 and TI2 filtering through the input

The signal after the controller and polarity control; if there is no filtering and phase change, TI1FP1 = TI1; if there is no filtering and phase change, then

TI2FP2 = TI2. According to the transition sequence of the two input signals, a count pulse and a direction signal are generated. Based on two losses

The transition sequence of the input signal, the counter counts up or down, and the hardware advances the DIR bit of the TIMx\_CR1 register

Line accordingly. Whether the counter counts on TI1, counts on TI2, or counts on both TI1 and TI2,

The transition at any input (TI1 or TI2) will recalculate the DIR bit.

The encoder interface mode is basically equivalent to using an external clock with direction selection. This means that the counter is only on

Continuous counting from 0 to the autoload value of the TIMx \_ARR register (according to the direction, or 0 to ARR counting, or

(ARR to 0 count). So TIMx\_ARR must be configured before starting counting; similarly, the catcher, comparator,

The prescaler, repetition counter, and trigger output characteristics still work as usual. Encoder mode and external clock mode 2 are not both

Content, so they cannot be operated at the same time.

In this mode, the counter is automatically modified according to the speed and direction of the incremental encoder, so the content of the counter

Finally indicates the position of the encoder. The counting direction corresponds to the direction of rotation of the connected sensor. The following table lists all possible

, Assuming that TI1 and TI2 are not transformed simultaneously.

35.

Effective edge

Relative signal level **(TITFP1** corresponds to **TI2** ,

**TI2FP2** corresponds to **TI1)**

**TI1FP1** signal

**TI2FP2** signal

rise

decline

rise

decline

Only count at TI1

high

Count down

Count up

Do not count

Do not count

low

Count up

Count down

Do not count

Do not count

Counting on TI2 only

high

Do not count

Do not count

Count up

Count down

low

Do not count

Do not count

Count down

Count up

Count on TI1 and TI2

number

high

Count down

Count up

Count up

Count down

low

Count up

Count down

Count down

Count up

An external incremental encoder can be directly connected to the MCU without the need for external interface logic. However, the general use ratio

The comparator converts the differential output of the encoder to a digital signal, which greatly increases the ability to resist noise interference. Encoder output

The third signal represents the mechanical zero point, which can be connected to an external interrupt input and trigger a counter reset.

The following figure is an example of counter operation, showing the generation of count signals and direction control. It also shows when selected

How to suppress the input jitter on both sides of the edge; jitter may be generated when the sensor position is close to a transition point.

In this example, we assume the configuration is as follows:

• CC1S = '01' (TIMx\_CCMR1 register, IC1FP1 is mapped to TI1)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

138/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 161** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

• CC2S = '01' (TIMx\_CCMR2 register, IC2FP2 is mapped to TI2)

• CC1P = '0' (TIMx\_CCER register, IC1FP1 is not inverted, IC1FP1=TI1)

• C2P = '0' (TIMx\_CCER register, IC2FP2 is not inverted, IC2FP2=TI2)

• SMS = '011' (TIMx\_SMCR register, all inputs are valid on rising and falling edges).

• CEN = '1' (TIMx\_CR1 register, counter enable)

352612

Increment

TI1

counter

Diminishing

Increment

TI2

Forward

Jitter

Reverse

Jitter

Forward

71.

136200

Diminishing

TI1

counter

Increment

Diminishing

TI2

Forward

Jitter

Reverse

Jitter

Forward

72. IC1FP1

When the timer is configured in encoder interface mode, it provides information about the current position of the sensor. Use the second configuration in the capture

The mode timer measures the interval between two encoder events and can obtain dynamic information (speed, acceleration, deceleration). Finger

The encoder output showing the mechanical zero point can be used for this purpose. According to the interval between two events, you can follow a fixed time

Read the counter. If possible, you can latch the counter value to the third input capture register (capture signal

Must be periodic and can be generated by another timer). It can also use a DMA generated by a real-time clock

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

139/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 162** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

Request to read its value.

**10.3.17**

The TI1S bit in the TIMx\_CR2 register allows the input filter of channel 1 to be connected to the output of an XOR gate.

The three inputs of the OR gate are TIMx\_CH1, TIMx\_CH2, and TIMx\_CH3.

The XOR output can be used for all timer input functions such as triggering or input capture. Section [10.3.](https://translate.googleusercontent.com/translate_f#162) 18 gives this special

An example of how to connect Hall sensors.

**10.3.18**

When using advanced control timer (TIM1) to generate PWM signal to drive the motor, another general-purpose TIMx (TIM2) can be used

Or TIM3) The timer is used as an "interface timer" to connect the Hall sensor, see Figure [73,](https://translate.googleusercontent.com/translate_f#163) 3 timer input pins (CC1

CC2, CC3) is connected to the TI1 input channel through an XOR gate (by setting TI1S in the TIMx\_CR2 register

Bit to select), the'interface timer' captures this signal.

The slave mode controller is configured in reset mode and the slave input is TI1F\_ED. Whenever one of the three inputs changes, the counter

Start counting from 0 again. This produces a time reference triggered by any change in the Hall input.

The capture/compare channel 1 on the'interface timer' is configured in capture mode, and the capture signal is TRC (see Figure [56](https://translate.googleusercontent.com/translate_f#146) ). Captured value

It reflects the time delay between two input changes and gives information about the motor speed.

'Interface timer' can be used to generate a pulse in output mode, this pulse can (by triggering a COM event

) Used to change the attributes of each channel of the advanced timer TIM1, and the advanced control timer generates a PWM signal to drive the horse

Reach. Therefore, the "interface timer" channel must be programmed after a specified delay (output compare or PWM mode)

A positive pulse is generated and this pulse is sent to the advanced control timer TIM1 through the TRGO output.

Example: The Hall input is connected to the TIMx timer, which requires a specific designation after each change on any Hall input

Time, change the PWM configuration of the advanced control timer TIMx.

• Set the TI1S bit of the TIMx\_CR2 register to '1' and configure three timer input logic ORs to the TI1 input,

• Time base programming: Set TIMx\_ARR to its maximum value (counter must be cleared by TI1 change). Set prescaler

A maximum counter period is obtained, which is longer than the time interval between two changes on the sensor.

• Set channel 1 to capture mode (select TRC): set CC1S = 01 in the TIMx\_CCMR1 register, if necessary,

Digital filters can also be set.

• Set channel 2 to PWM2 mode with the required delay: set OC2M in the TIMx\_CCMR1 register

= 111 and CC2S = 00.

• Select OC2REF as the trigger output on TRGO: set MMS = 101 in the TIMx\_CR2 register.

In the advanced control register TIM1, the correct ITR input must be a trigger input and the timer is programmed to generate

PWM signal, capture/compare control signal is pre-loaded (CCPC = 1 in TIMx\_CR2 register), trigger at the same time

The input controls the COM event (CCUS = 1 in the TIMx\_CR2 register). After a COM event, write to the next step

PWM control bits (CCxE, OCxM), which can be implemented in the interrupt subroutine that handles the rising edge of OC2REF.

The following figure shows this example:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

140/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 163** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

190862

TIH1

TIH2

TIH3

Counter (CNT)

(CCR2)

TRGO = OC2REF

COM

OC1

OC1N

OC2

OC2N

OC3

OC3N

Write CCxE, CCxNE

And OCxM for the next step

Advanced control timer

Interface timer

CCR1

C7A3

C7A8

C794

C7A5

C7AB

C796

73.

**10.3.19 TIMx**

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gating mode, and trigger mode.

When a trigger input event occurs, the counter and its prescaler can be re-initialized; meanwhile, if TIMx\_CR1

The URS bit of the register is low, and an update event UEV is also generated; then all the preload registers (TIMx\_ARR

, TIMx\_CCRx) have been updated.

In the following example, the rising edge of the TI1 input causes the up counter to be cleared:

• Configure channel 1 to detect the rising edge of TI1. Configure the bandwidth of the input filter (in this example, no filtering is required

Device, so keep IC1F = 0000). The capture prescaler is not used in the trigger operation, so no configuration is required. CC1S

The bit selects only the input capture source, that is, CC1S = 01 in the TIMx\_CCMR1 register. Set the TIMx\_CCER register

CC1P = 0 to determine the polarity (only the rising edge is detected).

• Set SMS = 100 in the TIMx\_SMCR register to configure the timer in reset mode; set the TIMx\_SMCR register

TS = 101, select TI1 as the input source.

• Set CEN = 1 in the TIMx\_CR1 register to start the counter.

The counter starts counting according to the internal clock, and then runs normally until TI1 has a rising edge; at this time, the counter is

Clear to zero and restart counting from 0. At the same time, the trigger flag (TIF bit in the TIMx\_SR register) is set, the root

According to the settings of the TIE (Interrupt Enable) bit and TDE (DMA Enable) bit in the TIMx\_DIER register, please generate an interrupt.

Request or a DMA request.

The figure below shows the behavior when the auto-reload register TIMx\_ARR = 0x36. At the rising edge of TI1 and the actual counter

The delay between resets depends on the resynchronization circuit at the input of TI1.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

141/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 164** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

770275

TI1

UG

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

33

32

31

30

00

01 02 03

36

00

01

02

34 35

03

74.

The enable of the counter depends on the level of the selected input.

In the following example, the counter only counts up when TI1 is low:

• Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (in this example, no filtering is required, so

Keep IC1F = 0000). The capture prescaler is not used in the trigger operation, so no configuration is required. The CC1S bit is used

Select the input capture source and set CC1S = 01 in the TIMx\_CCMR1 register. Set in TIMx\_CCER register

CC1P = 1 to determine the polarity (only low level detection).

• Set SMS = 101 in the TIMx\_SMCR register to configure the timer in gating mode; set the TIMx\_SMCR register

TS = 101, select TI1 as the input source.

• Set CEN = 1 in the TIMx\_CR1 register to start the counter. In gated mode, if CEN = 0, count

The device cannot be started regardless of the trigger input level.

As long as TI1 is low, the counter starts counting based on the internal clock and stops counting when TI1 goes high. When the counter starts or

When stopping, set the TIF flag in TIMx\_SR.

The delay between the rising edge of TI1 and the actual stop of the counter depends on the resynchronization circuit at the input of TI1.

470760

TI1

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

34

Counter register

TIF

33

32

31

30

35 36 37 38

Write TIF = 0

75.

The enabling of the counter depends on the event on the selected input.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

142/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 165** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

In the following example, the counter starts counting up on the rising edge of the TI2 input:

• Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (in this example, no filter is required, ensure that

Hold IC2F = 0000). The capture prescaler is not used in the trigger operation, and no configuration is required. CC2S bit is for selection only

CC2P = 1 in the input catcher to determine the polarity (only low level detection).

• Set SMS = 110 in the TIMx\_SMCR register to configure the timer in trigger mode; set the TIMx\_SMCR register

TS = 110, select TI2 as the input source.

When TI2 has a rising edge, the counter starts to count under the drive of the internal clock and sets the TIF flag. On TI2

The delay between the rising edge and the counter starting counting depends on the resynchronization circuit at the input of TI2.

873472

TI2

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

34

35

Counter register

36

TIF

37 38

76.

**2 +**

External clock mode 2 can be used with another slave mode (except external clock mode 1 and encoder mode). This

When the ETR signal is used as an input to an external clock, another one can be selected in reset mode, gating mode or trigger mode

Input as a trigger input. It is not recommended to use the TS bit of the TIMx\_SMCR register to select ETR as TRGI.

In the following example, when a rising edge appears on TI1, the counter counts up on each rising edge of ETR

once:

• Configure the external trigger input circuit through the TIMx\_SMCR register:

**–** ETF = 0000: no filtering

**–** ETPS = 00: without prescaler

**–** ETP = 0: Detect the rising edge of ETR and set ECE = 1 to enable external clock mode 2.

• Configure channel 1 as follows to detect the rising edge of TI:

**–** IC1F = 0000: no filtering

**– The** capture prescaler is not used in the trigger operation and no configuration is required

**–** Set CC1S = 01 in the TIMx\_CCMR1 register to select the input capture source

**–** Set CC1P = 0 in the TIMx\_CCER register to determine the polarity (only the rising edge is detected)

• Set SMS = 110 in the TIMx\_SMCR register to configure the timer in trigger mode. Set TIMx\_SMCR register

TS = 101, select TI1 as the input source.

When a rising edge occurs on TI1, the TIF flag is set and the counter starts counting on the rising edge of ETR.

The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronization circuit at the input of ETRP.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

143/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 166** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

608005

TI1

CEN/CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

34

35

Counter register

36

TIF

ETR

77.

2

**10.3.20**

All TIM timers are connected internally for timer synchronization or linking. See chapter TIM2/3/4 for details.

**10.3.21**

When the microcontroller enters the debug mode (CPU core is stopped), according to the setting of DBG\_TIMx\_STOP in the DBG module

When set, the TIMx counter can either continue normal operation or stop. See the subsequent commissioning chapter for details.

**10.4**

36. TIM1

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

TIMx\_CR1

Control register 1

0x00000000

[Section 10.4.1](https://translate.googleusercontent.com/translate_f#167)

0x04

TIMx\_CR2

Control register 2

0x00000000

[Section 10.4.2](https://translate.googleusercontent.com/translate_f#168)

0x08

TIMx\_SMCR

Slave mode control register

0x00000000

[Section 10.4.3](https://translate.googleusercontent.com/translate_f#170)

0x0C

TIMX\_DIER

DMA/interrupt enable register

0x00000000

[Section 10.4.4](https://translate.googleusercontent.com/translate_f#173)

0x10

TIMx\_SR

Status register

0x00000000

[Section 10.4.5](https://translate.googleusercontent.com/translate_f#175)

0x14

TIMx\_EGR

Event generation register

0x00000000

[Section 10.4.6](https://translate.googleusercontent.com/translate_f#177)

0x18

TIMx\_CCMR1

Capture/Compare Mode Register 1

0x00000000

[Section 10.4.7](https://translate.googleusercontent.com/translate_f#179)

0x1C

TIMx\_CCMR2

Capture/Compare Mode Register 2

0x00000000

[Section 10.4.8](https://translate.googleusercontent.com/translate_f#183)

0x20

TIMx\_CCER

Capture/compare enable register

0x00000000

[Section 10.4.9](https://translate.googleusercontent.com/translate_f#184)

0x24

TIMx\_CNT

counter

0x00000000

[Section 10.4.10](https://translate.googleusercontent.com/translate_f#187)

0x28

TIMx\_PSC

Prescaler

0x00000000

[Section 10.4.11](https://translate.googleusercontent.com/translate_f#188)

0x2C

TIMx\_ARR

Autoload register

0x00000000

[Section 10.4.12](https://translate.googleusercontent.com/translate_f#188)

0x30

TIMx\_RCR

Repeat count register

0x00000000

[Section 10.4.13](https://translate.googleusercontent.com/translate_f#188)

0x34

TIMx\_CCR1

Capture/Compare Register 1

0x00000000

[Section 10.4.14](https://translate.googleusercontent.com/translate_f#189)

0x38

TIMx\_CCR2

Capture/Compare Register 2

0x00000000

[Section 10.4.15](https://translate.googleusercontent.com/translate_f#189)

0x3C

TIMx\_CCR3

Capture/Compare Register 3

0x00000000

[Section 10.4.16](https://translate.googleusercontent.com/translate_f#190)

0x40

TIMx\_CCR4

Capture/Compare Register 4

0x00000000

[Section 10.4.17](https://translate.googleusercontent.com/translate_f#191)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

144/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 167** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x44

TIMx\_BDTR

Brake and dead zone registers

0x00000000

[Section 10.4.18](https://translate.googleusercontent.com/translate_f#191)

0x48

TIMx\_DCR

DMA control register

0x00000000

[Section 10.4.19](https://translate.googleusercontent.com/translate_f#194)

0x4C

TIMx\_DMAR

Continuous mode DMA address

0x00000000

[Section 10.4.20](https://translate.googleusercontent.com/translate_f#195)

0x54

TIMx\_CCMR3

Capture/Compare Mode Register 3

0x00000000

[Section 10.4.21](https://translate.googleusercontent.com/translate_f#196)

0x58

TIMx\_CCR5

Capture/Compare Register 5

0x00000000

[Section 10.4.22](https://translate.googleusercontent.com/translate_f#196)

**10.4.1**

**1(TIMx\_CR1)**

Offset address: 0x00

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

ARPE

UDIS

CMS

URS

OPM

DIR

CEN

CKD

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, always read as 0

9:8

CKD

rw

0x00

Clock division

These 2 bits are defined in the timer clock (CK\_INT) frequency, dead time and by

Between the dead zone generator and the sampling clock used by the digital filter (ETR, TIx)

The frequency division ratio.

00: t DTS = t CK\_INT

01: t DTS = 2 xt CK\_INT

10: t DTS = 4 xt CK\_INT

11: Reserved, do not use this configuration

7

ARPE

rw

0x00

Auto-reload preload enable bit (Auto-reload preload enable)

0: TIMx\_ARR register is not buffered

1: TIMx\_ARR register is loaded into the buffer

6:5

CMS

rw

0x00

Select Center-aligned mode selection

00: Edge alignment mode. Counter up or down according to the direction bit (DIR)

count

01: Center alignment mode 1. The counter alternately counts up and down. Configuration

Is the output of the output channel (CCxS = 00 in the TIMx\_CCMRx register)

The compare interrupt flag bit is only set when the counter is counting down

10: Center alignment mode 2. The counter alternately counts up and down. Configuration

Is the output of the output channel (CCxS = 00 in the TIMx\_CCMRx register)

Compare interrupt flag bit is set only when the counter counts up

11: Center alignment mode 3. The counter alternately counts up and down. Configuration

Is the output of the output channel (CCxS = 00 in the TIMx\_CCMRx register)

Compare interrupt flag bit is set when the counter counts up and down

Note: When the counter is turned on (CEN = 1), it is not allowed to switch from edge-aligned mode

Switch to center alignment mode.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

145/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 168** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

4

DIR

rw

0x00

Direction

0: the counter counts up

1: The counter counts down

: When the counter is configured in center-aligned mode or encoder mode, this bit is read-only.

3

OPM

rw

0x00

One pulse mode

0: The counter does not stop when an update event occurs

1: When the next update event occurs (clears the CEN bit), the counter stops

2

URS

rw

0x00

Update request source (Update request source) software selects through this bit

The source of the UEV event.

0: If update interrupt or DMA request is allowed, any one of the following events

An update interrupt or DMA request is generated:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller

1: If an update interrupt or DMA request is allowed, only the counter overflows

An update interrupt or DMA request is generated only after the overflow/underflow

1

UDIS

rw

0x00

Disable disable (Update disable) Software enables/disables UEV through this bit

Event generation

0: UEV is allowed. Update (UEV) events are generated by any of the following events:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller are cached registers are loaded into their

Preload value.

1: UEV is prohibited. No update event occurs, shadow registers (ARR, PSC,

CCRx) keep their values. If the UG bit or slave mode controller is set

A hardware reset is issued and the counter and prescaler are reinitialized

0

CEN

rw

0x00

Counter enable

0: disable counter

1: Enable the counter.

Note: After the CEN bit is set by the software, the external clock, gating mode and encoder mode

To work. The trigger mode can automatically set the CEN bit through hardware.

**10.4.2**

**2(TIMx\_CR2)**

Offset address: 0x04

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

146/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 169** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Res. OIS4 OIS3N OIS3 OIS2N OIS2 OIS1N OIS1 TI1S

MMS

CCDSCCUS Res. CCPC

OIS5

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 17

Reserved

Reserved, always read as 0

16

OIS5

rw

0x00

Output idle state 5 (OC5 output). See bit OIS1.

15

Reserved

Reserved, always read as 0

14

OIS4

rw

0x00

Output idle state 4 (OC4 output). See bit OIS1.

13

OIS3N

rw

0x00

Output idle state 3 (OC3N output). See bit OIS1N.

12

OIS3

rw

0x00

Output idle state 3 (OC3 output). See bit OIS1.

11

OIS2N

rw

0x00

Output idle state 2 (OC2N output). See bit OIS1N.

10

OIS2

rw

0x00

Output idle state 2 (OC2 output). See bit OIS1.

9

OIS1N

rw

0x00

Output Idle state 1 (OC1N output) (Output Idle state 1)

0: When MOE = 0, OC1N = 0 after the dead zone

1: When MOE = 0, OC1N = 1 after the dead zone

Note: After LOCK (TIMx\_BKR register) level 1, 2, or 3 has been set, the

The bit cannot be modified.

8

OIS1

rw

0x00

Output Idle state 1 (OC1 output) (Output Idle state 1)

0: When MOE = 0, if OC1N is implemented, then OC1 = 0 after the dead zone

1: When MOE = 0, if OC1N is implemented, then OC1 = 1 after the dead zone

Note: After LOCK (TIMx\_BKR register) level 1, 2, or 3 has been set, the

The bit cannot be modified.

7

TI1S

rw

0x00

TI1 selection (TI1 selection)

0: TIMx\_CH1 pin is connected to TI1 input

1: TIMx\_CH1, TIMx\_CH2 and TIMx\_CH3 pins are connected through XOR

To TI1 input

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

147/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 170** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6:4

MMS

rw

0x00

Master mode selection

These two bits are used to select the synchronization information sent to the slave timer in master mode

(TRGO). The possible combinations are as follows:

000: Reset – the UG bit in the TIMx\_EGR register is used as a trigger input

Out (TRGO). If trigger input (slave mode controller is in reset mode)

Reset occurs, the signal on TRGO will have a delay relative to the actual reset

late.

001: Enable – The counter enable signal CNT\_EN is used as a trigger input

Out (TRGO). Sometimes it is necessary to start multiple timers or controls at the same time

Enable the slave timer for a period of time. Counter enable signal is through CEN

Logic OR generation of the trigger input signal in control bit and gating mode. Count

When the counter enable signal is controlled by the trigger input, there will be a delay on TRGO,

Unless master/slave mode is selected (see the MSM bit in the TIMx\_SMCR register

description of).

010: Update-The update event is selected as the trigger input (TRGO). For example, one

The clock of each master timer can be used as a prescaler for the slave timer.

011: Comparison pulse-when a capture occurs or a comparison is successful, when setting

When the CC1IF flag is set (even if it is already high), the trigger output sends a positive

Pulse (TRGO).

100: Compare – OC1REF signal is used as a trigger output (TRGO)

101: Compare – OC2REF signal is used as a trigger output (TRGO)

110: Compare – OC3REF signal is used as a trigger output (TRGO)

111: Compare – OC4REF signal is used as a trigger output (TRGO)

3

CCDS

rw

0x00

Capture/compare DMA selection

0: Send CCx DMA request when CCx event occurs

1: When an update event occurs, the CCx DMA request is sent

2

CCUS

rw

0x00

Capture/compare control update selection (Capture/compare control update

selection)

0: If the capture/compare control bit is preloaded (CCPC = 1), only pass

Update them by setting the COM bit

1: If the capture/compare control bit is preloaded (CCPC = 1), you can pass

Update them by setting the COM bit or a rising edge on TRGI

Note: This bit only affects channels with complementary outputs.

1

Reserved

Reserved, always read as 0

0

CCPC

rw

0x00

Capture/compare preloaded control bit (Capture/compare preloaded con-

trol)

0: CCxE, CCxNE and OCxM bits are not pre-loaded

1: CCxE, CCxNE and OCxM bits are preloaded; after setting this bit,

They are only updated after the COM bit is set

Note: This bit only affects channels with complementary outputs.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

148/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 171** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**10.4.3**

**(TIMx\_SMCR)**

Offset address: 0x08

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

ETP

ETPS

ECE

ETF

MSM

TS

Res.

SMS

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

ETP

rw

0x00

External trigger polarity

This bit selects whether to use ETR or the inverse of ETR as the trigger operation.

0: ETR is not inverted, high level or rising edge is valid

1: ETR is inverted, low level or falling edge is valid

14

ECE

rw

0x00

External clock enable (External clock enable)

This bit enables external clock mode 2.

0: Disable external clock mode 2

1: Enable external clock mode 2, the counter is valid by any on the ETRF signal

Rising edge drive

Note 1: Set ECE bit and select external clock mode 1 and connect TRGI to

ETRF (SMS = 111 and TS = 111) have the same effect. Note 2: The following

Mode can be used simultaneously with external clock mode 2: reset mode, gating mode and touch

Transmit mode; however, TRGI cannot be connected to ETRF at this time (TS bit cannot be 111). Note

3: When external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock

The input is ETRF.

13:12

ETPS

rw

0x00

External trigger prescaler

The frequency of the external trigger signal ETRP must be at most the frequency of TIMxCLK

1/4. When inputting a faster external clock, you can use prescaler to reduce ETRP

Frequency of.

00: turn off prescaler

01: ETRP frequency divided by 2

10: ETRP frequency divided by 4

11: ETRP frequency divided by 8

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

149/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 172** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

11:8

ETF

rw

0x00

External trigger filter

These bits define the sampling frequency of ETRP signal and the digital filtering of ETRP

The bandwidth of the wave. In fact, the digital filter is an event counter, which records

After N events, an output transition will occur.

0000: no filter, sampling with fDTS

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

0110: Sampling frequency f SAMPLING = f DTS /4, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

7

MSM

rw

0x00

Master/slave mode

0: No effect

1: The event on the trigger input (TRGI) is delayed to allow the current setting

The perfect synchronization between the timer (via TRGO) and its slave timer, this is necessary

It is very useful when you want to synchronize several timers to a single external event

of

6:4

TS

rw

0x00

Trigger selection

These 3 bits select the trigger input for synchronization counter.

000: Internal trigger 0 (ITR0)

001: Internal trigger 1 (ITR1)

010: Internal trigger 2 (ITR2)

011: Internal trigger 3 (ITR3)

100: TI1's edge detector (TI1F\_ED)

101: Filtered timer input 1 (TI1FP1)

110: Filtered timer input 2 (TI2FP2)

111: External trigger input (ETRF)

For more details about ITRx, see the table below.

Note: These bits can only be changed when not used (eg SMS = 000), to avoid changing

Wrong edge detection when changing time.

3

Reserved

Reserved, always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

150/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 173** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2: 0

SMS

rw

0x00

Slave mode selection

When the external signal is selected, the effective edge of the trigger signal (TRGI) and the selected

External input polarity is related (see input control register and control register

Bright)

000: Slave mode off-if CEN = 1, the prescaler is directly internally

Clock driven.

001: Encoder mode 1-according to the level of TI1FP1, the counter is in TI2FP2

Counts up/down.

010: Encoder mode 2-according to the level of TI2FP2, the counter is in TI1FP1

Counts up/down.

011: Encoder mode 3-according to the level of another input, the counter is

The edges of TI1FP1 and TI2FP2 count up/down.

100: Reset mode-the rising edge of the selected trigger input (TRGI) is reset

Initialize the counter and generate a signal to update the register.

101: Gating mode-when the trigger input (TRGI) is high, the time of the counter

The bell opens. When the trigger input goes low, the counter stops (but does not reset). meter

The start and stop of the counter are controlled.

110: Trigger mode-the counter starts on the rising edge of the trigger input TRGI (but

No reset), only the start of the counter is controlled.

111: External clock mode 1-rising edge of the selected trigger input (TRGI)

Drive counter.

Note: If TI1F\_EN is selected as the trigger input (TS = 100), do not use the gate

Control mode. This is because TI1F\_ED outputs a pulse every time TI1F changes,

However, the gating mode is to check the level of the trigger input.

37. TIMx

Slave timer

**ITR0**

**ITR1**

**ITR2**

**ITR3**

TIM1

no

TIM2

TIM3

no

TIM2

TIM1

no

TIM3

no

TIM3

TIM1

TIM2

no

no

**10.4.4 DMA/**

**(TIMX\_DIER)**

Offset address: 0x0C

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Res. TDE

BIE

UIE

COM

DE

CC4

DE

CC3

DE

CC2

DE

CC1

DE

UDE

TIE

COM

IE

CC4

IE

CC3

IE

CC2

IE

CC1

IE

Reserved

CC5

DE

CC5

IE

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

151/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 174** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:18

Reserved

Reserved, always read as 0

17

CC5DE

rw

0x00

Allow Capture/Compare 5 DMA request (Capture/Compare 5 DMA

request enable)

0: Disable capture/compare 5 DMA request

1: Allow capture/compare 5 DMA request

16

CC5IE

rw

0x00

Allow Capture/Compare 5 interrupt en-

able)

0: Disable capture/compare 5 interrupt

1: Allow capture/compare 5 interrupt

15

Reserved

Reserved, always read as 0

14

TDE

rw

0x00

Allow trigger DMA request (Trigger DMA request enable)

0: Disable triggering of DMA request

1: Allow to trigger DMA request

13

COMDE

rw

0x00

Allow COM DMA request (COM DMA request enable)

0: Disable COM DMA request

1: Allow COM DMA request

12

CC4DE

rw

0x00

Allow Capture/Compare 4 DMA request (Capture/Compare 4 DMA

request enable)

0: Disable capture/compare 4 DMA request

1: Allow capture/compare 4 DMA request

11

CC3DE

rw

0x00

Allow Capture/Compare 3 DMA request (Capture/Compare 3 DMA

request enable)

0: Disable capture/compare 3 DMA request

1: Allow capture/compare 3 DMA request

10

CC2DE

rw

0x00

Allow Capture/Compare 2 DMA request (Capture/Compare 2 DMA

request enable)

0: Disable capture/compare 2 DMA request

1: Allow capture/compare 2 DMA request

9

CC1DE

rw

0x00

Allow Capture/Compare 1 DMA request (Capture/Compare 1 DMA

request enable)

0: Disable capture/compare 1 DMA request

1: Allow capture/compare 1 DMA request

8

UDE

rw

0x00

Update DMA request enable

0: Disable update DMA request

1: Allow updated DMA requests

7

BIE

rw

0x00

Break interrupt enable

0: Disable brake interruption

1: Allow the brake to be interrupted

6

TIE

rw

0x00

Trigger interrupt enable

0: Disable trigger interrupt

1: Enable trigger interrupt

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

152/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 175** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

5

COMIE

rw

0x00

COM interrupt enable

0: Disable COM interrupt

1: Allow COM interrupt

4

CC4IE

rw

0x00

Allow Capture/Compare 4 interrupt en-

able)

0: Disable capture/compare 4 interrupt

1: Enable capture/compare 4 interrupt

3

CC3IE

rw

0x00

Allow Capture/Compare 3 interrupt en-

able)

0: Disable capture/compare 3 interrupt

1: Allow capture/compare 3 interrupt

2

CC2IE

rw

0x00

Allow Capture/Compare 2 interrupt en-

able)

0: Disable capture/compare 2 interrupt

1: Enable capture/compare 2 interrupt

1

CC1IE

rw

0x00

Allow Capture/Compare 1 interrupt en-

able)

0: Disable capture/compare 1 interrupt

1: Enable capture/compare 1 interrupt

0

UIE

rw

0x00

Update interrupt enable

0: Disable update interrupt

1: Allow update interrupt

**10.4.5**

**(TIMx\_SR)**

Offset address: 0x10

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

CC4

OF

CC3

OF

CC2

OF

CC1

OF

Res. BIF

TIF

UIF

COM

IF

CC4

IF

CC3

IF

CC2

IF

CC1

IF

CC5

IF

Reserved

CC5

OF

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:18

Reserved

Reserved, always read as 0

17

CC5OF

rc\_w0

0x00

Capture/Compare 4 Overcapture (Capture/Compare 4 overcap-

ture flag)

See CC1OF description.

16

CC5IF

rc\_w0

0x00

Capture/Compare 5 interrupt flag

Refer to CC1IF description.

15:13

Reserved

Reserved, always read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

153/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 176** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

12

CC4OF

rc\_w0

0x00

Capture/Compare 4 overcap-capture (Capture/Compare 4 overcap-

ture flag)

See CC1OF description.

11

CC3OF

rc\_w0

0x00

Capture/Compare 3 overcap-capture (Capture/Compare 3 overcap-

ture flag)

See CC1OF description.

10

CC2OF

rc\_w0

0x00

Capture/Compare 2 overcap-capture (Capture/Compare 2 overcap-

ture flag)

See CC1OF description.

9

CC1OF

rc\_w0

0x00

Capture/Compare 1 overcap-capture (Capture/Compare 1 overcap-

ture flag)

This flag can be set by hardware only when the corresponding channel is configured for input capture

1. Write 0 to clear this bit.

0: No repeated capture;

1: When the value of the counter is captured in the TIMx\_CCR1 register, the CC1IF

The status is already 1.

8

Reserved

Reserved, always read as 0

7

BIF

rc\_w0

0x00

Break interrupt flag

When the brake input is valid, this bit is set by hardware. If there is no brake input

Valid, the bit can be cleared to "0" by software

0: No braking event

1: An effective level is detected on the brake input

6

TIF

rc\_w0

0x00

Trigger interrupt flag

When a trigger event occurs (when the slave mode controller is in a mode other than gating mode)

In other modes, a valid edge is detected at the TRGI input, or gated mode

Any edge under the formula) is set to 1 by hardware. It is cleared by software.

0: No trigger event is generated

1: Trigger interrupt waiting for response

5

COMIF

rc\_w0

0x00

COM interrupt flag

When a COM event is generated (when the capture/compare control bits: CCxE, CCxNE,

(OCxM has been updated) This bit is set by hardware. It is cleared by software.

0: No COM event is generated

1: COM interrupt waiting for response

4

CC4IF

rc\_w0

0x00

Capture/Compare 4 interrupt flag

Refer to CC1IF description.

3

CC3IF

rc\_w0

0x00

Capture/Compare 3 interrupt flag

Refer to CC1IF description.

2

CC2IF

rc\_w0

0x00

Capture/Compare 2 interrupt flag

Refer to CC1IF description.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

154/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 177** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

CC1IF

rc\_w0

0x00

Capture/Compare 1 interrupt flag

If channel CC1 is configured in output mode:

When the counter value matches the comparison value, the bit is set to '1' by the hardware, but in the center

Except in symmetric mode (refer to the CMS bit of the TIMx\_CR1 register). it

Cleared by software to '0'.

0: No match occurred

1: The value of TIMx\_CNT matches the value of TIMx\_CCR1

If channel CC1 is configured for input mode:

When the capture event occurs, this bit is set by hardware, it is cleared by software or through

Clear '0' after reading TIMx\_CCR1.

0: No input capture

1: The counter value has been captured (copied) to TIMx\_CCR1 (checked on IC1

An edge with the same polarity as the selected one is measured)

0

UIF

rc\_w0

0x00

Update interrupt flag

This bit is set to '1' by hardware when an update event occurs. It is cleared to '0' by the software.

0: No update event is generated

1: Update event is waiting for response. When the register is updated, this bit is set to '1' by hardware:

-If UDIS = 0 in the TIMx\_CR1 register, when REP\_CNT = 0

An update event is generated (when the down counter overflows or underflows repeatedly)

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, when TIMx\_EGR

An update event occurs when UG = 1 in the register (software resets counter CNT

New initialization)

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, the counter

An update event is generated when CNT is triggered by event re-initialization. (Refer to synchronous control

(Description of register)

**10.4.6**

**(TIMx\_EGR)**

Offset address: 0x14

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

BG

UG

TG COMG CC4G CC3G CC2G CC1G

Reserved

CC5G

w

w

w

w

w

w

w

w

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:17

Reserved

Reserved, always read as 0

16

CC5G

w

0x00

Generate Capture/Compare 5 generation (Capture/Compare 5 generation)

Refer to CC1G description.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

155/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 178** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, always read as 0

7

BG

w

0x00

Break generation

This bit is set to '1' by the software and is used to generate a braking event, automatically by the hardware

Clear '0'.

0: No action

1: Generate a braking event. At this time MOE = 0, BIF = 1, if you enable

Corresponding interrupts and DMA, corresponding interrupts and DMA are generated

6

TG

w

0x00

Trigger generation

This bit is set by software to generate a trigger event, which is automatically generated by hardware

Clear '0'.

0: No action

1: TIF = 1 in the TIMx\_SR register, if the corresponding interrupt and DMA are enabled,

Then generate the corresponding interrupt and DMA

5

COMG

w

0x00

Capture/compare events and generate control updates (Capture/Compare control

update generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: When CCPC = 1, it is allowed to update the CCxE, CCxNE, OCxM bits

Note: This bit is only valid for channels with complementary outputs.

4

CC4G

w

0x00

Generate Capture/Compare 4 generation

Refer to CC1G description.

3

CC3G

w

0x00

Generate Capture/Compare 3 generation

Refer to CC1G description.

2

CC2G

w

0x00

Generate Capture/Compare 2 generation

Refer to CC1G description.

1

CC1G

w

0x00

Generate Capture/Compare 1 generation

This bit is set by software to generate a capture/compare event.

Reset to 0.

0: No action

1: Generate a capture/compare event on channel CC1:

If channel CC1 is configured as an output:

Set CC1IF=1, if the corresponding interrupt and DMA are turned on, the corresponding

Interrupt and DMA.

If channel CC1 is configured as input:

The current counter value is captured into the TIMx\_CCR1 register and set

CC1IF = 1, if the corresponding interrupt and DMA are turned on, the corresponding

Off and DMA. If CC1IF is already 1, set CC1OF = 1.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

156/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 179** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

UG

w

0x00

Generate an update event (Update generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: Re-initialize the counter and generate an update event. Note the prescaler

The counter of the counter is also cleared to '0' (but the prescaler coefficient is unchanged). If in the center

In symmetric mode or DIR = 0 (count up), the counter is cleared to '0'; if

DIR = 1 (count down), the counter takes the value of TIMx\_ARR.

**10.4.7**

**/**

**1(TIMx\_CCMR1)**

Offset address: 0x18

Reset value: 0x0000

The channel can be used for input (capture mode) or output (comparison mode), and the direction of the channel is defined by the corresponding CCxS. The deposit

The function of other bits of the device is different from the output mode. OCxx describes the function of the channel in output mode, ICxx describes

The function of the channel in output mode. Therefore, it must be noted that the function of the same bit in output mode and input mode is not

Same.

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OC2M

CC1S

OC2

PE

OC2

FE

OC1M

OC1

PE

OC1

FE

IC2F

IC2PSC

CC2S

OC1

CE

IC1F

IC1PSC

OC2

CE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

OC2CE

rw

0x00

Output compare 2 clear enable

14:12

OC2M

rw

0x00

Output compare 2 mode

11

OC2PE

rw

0x00

Output compare 2 preload en-

able)

10

OC2FE

rw

0x00

Output compare 4 fast enable (Output compare 4 fast enable)

9:8

CC2S

rw

0x00

Capture/Compare 2 selection

This bit defines the direction of the channel (input/output) and the selection of input pins:

00: CC2 channel is configured as output

01: CC2 channel is configured as input, IC2 is mapped on TI2

10: CC2 channel is configured as input, IC2 is mapped on TI1

11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC2S is only when the channel is closed (CC2E = 0 in the TIMx\_CCER register)

Is writable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

157/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 180** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7

OC1CE

rw

0x00

Output compare 1 clear enable

0: OC1REF is not affected by ETRF input

1: When high level of ETRF input is detected, clear OC1REF = 0

6:4

OC1M

rw

0x00

Output compare 1 mode

These 3 bits define the action of the output reference signal OC1REF, and OC1REF

Decide the values ​​​​of OC1 and OC1N. OC1REF is active high, and

The effective levels of OC1 and OC1N depend on the CC1P and CC1NP bits.

000: freeze. Output compare register TIMx\_CCR1 and counter

Comparison between TIMx\_CNT has no effect on OC1REF

001: Set channel 1 to effective level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is high

010: Set channel 1 to invalid level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is low

011: Flip. When TIMx\_CCR1=TIMx\_CNT, flip OC1REF

Level

100: Forced to invalid level. Force OC1REF low

101: Forced to effective level. Force OC1REF high

110: PWM mode 1-When counting up, when TIMx\_CNT <

When TIMx\_CCR1, channel 1 is active level, otherwise it is inactive level;

When counting down, when TIMx\_CNT> TIMx\_CCR1, channel 1 is none

Effective level (OC1REF = 0), otherwise the effective level (OC1REF = 1)

111: PWM mode 2-When counting up, when TIMx\_CNT <

When TIMx\_CCR1, channel 1 is invalid level, otherwise it is valid level;

When counting down, when TIMx\_CNT> TIMx\_CCR1, channel 1 is

Effective level, otherwise invalid level

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = 00 (the channel is configured as an output), this bit cannot be modified. Note

2: In PWM mode 1 or PWM mode 2, only when the comparison result changes

Or when switching from freeze mode to PWM mode in output compare mode, OC1REF

The level changes.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

158/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 181** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

OC1PE

rw

0x00

Output compare 1 preload en-

able)

0: Disable the preload function of the TIMx\_CCR1 register, which can be written at any time

TIMx\_CCR1 register, and the newly written value takes effect immediately

1: Turn on the preload function of the TIMx\_CCR1 register.

Preload register operation, the preload value of TIMx\_CCR1 is updated to

Is loaded into the current register when it comes

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = 00 (the channel is configured as an output), this bit cannot be modified. Note 2:

In single pulse mode only (OPM = 1 in the TIMx\_CR1 register), you can

The PWM mode is used when the preload register is recognized, otherwise its operation is undefined.

2

OC1FE

rw

0x00

Output compare 1 fast enable (Output compare 1 fast enable)

This bit is used to speed up the response of the CC output to trigger input events.

0: CC1 operates normally according to the value of the counter and CCR1, even if it is triggered

The device is turned on. When the trigger input has a valid edge, activate CC1

The minimum output delay is 5 clock cycles

1: The valid edge input to the trigger acts as if a comparison match has occurred.

Therefore, OC is set to the comparison level regardless of the comparison result. Sample trigger

The delay between the effective edge of the device and the output of CC1 is reduced to 3 clock cycles.

OCFE only works when the channel is configured in PWM1 or PWM2 mode

1:0

CC1S

rw

0x00

Capture/Compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: CC1 channel is configured as input, IC1 is mapped on TI2

11: CC1 channel is configured as input, IC1 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIMx\_CCER register)

Is writable.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:12

IC2F

rw

0x00

Input capture 2 filter

11:10

IC2PSC

rw

0x00

Input capture 2 prescaler

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

159/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 182** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9:8

CC2S

rw

0x00

Capture/Compare 2 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC2 channel is configured as output

01: CC2 channel is configured as input, IC2 is mapped on TI2

10: CC2 channel is configured as input, IC2 is mapped on TI1

11: CC2 channel is configured as input, IC2 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC2S is only when the channel is closed (CC2E = 0 in the TIMx\_CCER register)

Is writable.

7:4

IC1F

rw

0x00

Input capture 1 filter

These bits define the sampling frequency and digital filter length of the TI1 input. number

The word filter consists of an event counter, which will record after N events

Produces an output transition:

0000: no filter, sampling with fDTS

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

0110: Sampling frequency f SAMPLING = f DTS /4, N = 6

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

3:2

IC1PSC

rw

0x00

Input capture 1 prescaler

These 2 bits define the prescaler coefficient of the CC1 input (IC1).

When CC1E = 0 (in the TIMx\_CCER register), the prescaler is reset.

00: No prescaler, every edge detected on the capture input port is touched

Send a capture

01: Capture is triggered every 2 events

10: Capture is triggered every 4 events

11: Trigger every 8 events

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

160/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 183** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1:0

CC1S

rw

0x00

Capture/compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: CC1 channel is configured as input, IC1 is mapped on TI2

11: CC1 channel is configured as input, IC1 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIMx\_CCER register)

Is writable.

**10.4.8**

**/**

**2(TIMx\_CCMR2)**

Offset address: 0x1C

Reset value: 0x0000

See the description of the CCMR1 register above.

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OC4M

CC3S

OC4

PE

OC4

FE

OC3M

OC3

PE

OC3

FE

OC4

CE

IC4F

IC4PSC

CC4S

OC3

CE

IC3F

IC3PSC

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

OC4CE

rw

0x00

Output compare 4 clear enable

14:12

OC4M

rw

0x00

Output compare 4 mode (Output compare 4 mode)

11

OC4PE

rw

0x00

Output compare 4 preload en-

able)

10

OC4FE

rw

0x00

Output compare 4 fast enable (Output compare 4 fast enable)

9:8

CC4S

rw

0x00

Capture/Compare 4 selection

The 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC4 channel is configured as output

01: CC4 channel is configured as input, IC4 is mapped on TI4

10: CC4 channel is configured as input, IC4 is mapped on TI3

11: CC4 channel is configured as input, IC4 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC4S is only when the channel is closed (CC4E = 0 in the TIMx\_CCER register)

Is writable.

7

OC3CE

rw

0x00

Output compare 3 clear '0' enable (Output compare 3 clear enable)

6:4

OC3M

rw

0x00

Output compare 3 mode

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

161/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 184** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

OC3PE

rw

0x00

Output compare 3 preload en-

able)

2

OC3FE

rw

0x00

Output compare 3 fast enable (Output compare 3 fast enable)

1:0

CC3S

rw

0x00

Capture/Compare 3 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC3 channel is configured as output

01: CC3 channel is configured as input, IC3 is mapped on TI3

10: CC3 channel is configured as input, IC3 is mapped on TI4

11: CC3 channel is configured as input, IC3 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC3S is only when the channel is closed (CC3E = 0 in the TIMx\_CCER register)

Is writable.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:12

IC4F

rw

0x00

Input capture 4 filter

11:10

IC4PSC

rw

0x00

Input capture 4 prescaler

9:8

CC4S

rw

0x00

Capture/Compare 4 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC4 channel is configured as output

01: CC4 channel is configured as input, IC4 is mapped on TI4

10: CC4 channel is configured as input, IC4 is mapped on TI3

11: CC4 channel is configured as input, IC4 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC4S is only when the channel is closed (CC4E = 0 in the TIMx\_CCER register)

Is writable.

7:4

IC3F

rw

0x00

Input capture 3 filter

3:2

IC3PSC

rw

0x00

Input capture 3 prescaler

1:0

CC3S

rw

0x00

Capture/compare 3 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC3 channel is configured as output

01: CC3 channel is configured as input, IC3 is mapped on TI3

10: CC3 channel is configured as input, IC3 is mapped on TI4

11: CC3 channel is configured as input, IC3 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC3S is only when the channel is closed (CC3E = 0 in the TIMx\_CCER register)

Is writable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

162/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 185** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**10.4.9**

**/**

**(TIMx\_CCER)**

Offset address: 0x20

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

CC1P

CC2E

CC1E

CC4P CC4E

CC3

NP

CC3

NE

CC3P CC3E

CC2

NE

CC2

NP

CC2P

CC1

NP

CC1

NE

CC5P CC5E

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:18

Reserved

Reserved, always read as 0

17

CC5P

rw

0x00

Capture/Compare 5 output polarity

Refer to the description of CC1P.

16

CC5E

rw

0x00

Capture/Compare 5 output enable

Refer to the description of CC1E.

15:14

Reserved

Reserved, always read as 0

13

CC4P

rw

0x00

Capture/Compare 4 output polarity

Refer to the description of CC1P.

12

CC4E

rw

0x00

Capture/Compare 4 output enable

Refer to the description of CC1E.

11

CC3NP

rw

0x00

Input/Capture 3 Complementary output polarity (Capture/Compare 3 comple-

mentary output polarity)

Refer to the description of CC1NP.

10

CC3NE

rw

0x00

Input/capture 3 complementary output enable (Capture/Compare 3 comple-

mentary output enable)

Refer to the description of CC1NE.

9

CC3P

rw

0x00

Capture/Compare 3 output polarity

Refer to the description of CC1P.

8

CC3E

rw

0x00

Capture/Compare 3 output enable

Refer to the description of CC1E.

7

CC2NP

rw

0x00

Input/Capture 2 Complementary output polarity (Capture/Compare 2 comple-

mentary output polarity)

Refer to the description of CC1NP.

6

CC2NE

rw

0x00

Input/capture 2 complementary output enable (Capture/Compare 2 comple-

mentary output enable)

Refer to the description of CC1NE.

5

CC2P

rw

0x00

Capture/Compare 2 output polarity

Refer to the description of CC1P.

4

CC2E

rw

0x00

Capture/Compare 2 output enable

Refer to the description of CC1E.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

163/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 186** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

CC1NP

rw

0x00

Input/Capture 1 Complementary output polarity (Capture/Compare 1 comple-

mentary output polarity)

0: OC1N active high

1: OC1N active low

Note: When the LOCK level (LCCK bit in the TIMx\_BDTR register) is set to 3 or

2 When CC1S = 00 (channel is configured as output), this bit cannot be modified.

2

CC1NE

rw

0x00

Input/capture 1 complementary output enable (Capture/Compare 1 comple-

mentary output enable)

0: Off-OC1N disables output, so the output level of OC1N depends on

The values ​​​​of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1E bits

1: On-OC1N signal is output to the corresponding output pin, its output level

Depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1E bits

Value of

1

CC1P

rw

0x00

Capture/Compare 1 output polarity

CC1 channel is configured as output:

0: OC1 active high

1: OC1 active low

The CC1 channel is configured as an input:

This bit selects whether IC1 or IC1's inverted signal is used as a trigger or capture signal.

0: No inversion: capture occurs on the rising edge of IC1; when used as an external trigger

, IC1 is not inverted

1: Invert: capture occurs on the falling edge of IC1; when used as an external trigger,

IC1 invert

0

CC1E

rw

0x00

Capture/Compare 1 output enable

CC1 channel is configured as output:

0: Off-OC1 disables output, so the output level of OC1 depends on

The values ​​​​of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits

1: On-OC1 signal is output to the corresponding output pin, and its output level depends on

Depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits

Value of

The CC1 channel is configured as an input:

This bit determines whether the value of the counter can be captured into the TIMx\_CCR1 register.

0: capture prohibited

1: Capture enable

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

164/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 187** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

38.

OCx

OCxN

Control bit

Output status (1)

**MOE** bit **OSSI**

Bit

**OSSR**

Bit

**CCxE**

Bit

**CCxNE**

Bit

**OCx** output status

**OCxN** output status

1

X

0

0

0

Output prohibited (disconnected from timer)

OCx = 0, OCx\_EN = 0

Output prohibited (disconnected from timer)

OCxN = 0, OCxN\_EN = 0

0

0

1

Output prohibited (disconnected from timer)

OCx = 0, OCx\_EN = 0

OCxREF + polarity,

OCxN = OCxREF xor CCxNP,

OCxN\_EN = 1

0

1

0

OCxREF + polarity,

OCx = OCxREF xor CCxP,

OCx\_EN = 1

Output prohibited (disconnected from timer)

OCxN = 0, OCxN\_EN = 0

0

1

1

OCxREF + polarity + dead zone,

OCx\_EN=1

OCxREF Invert + Polarity + Dead Zone,

OCxN\_EN = 1

1

0

0

Output prohibited (disconnected from timer)

OCx = CCxP, OCx\_EN = 0

Output disabled (disconnected from timer) OCxN =

CCxNP, OCxN\_EN = 0

1

0

1

Off state (output is enabled and inactive

level)

OCx = CCxP, OCx\_EN = 1

OCxREF + polarity,

OCxN = OCxREF xor CCxNP,

OCxN\_EN = 1

1

1

0

OCxREF + polarity,

OCx = OCxREF xor CCxP,

OCx\_EN = 1

Off state (output is enabled and inactive

level)

OCxN = CCxNP, OCxN\_EN = 1

1

1

1

OCxREF + polarity + dead zone,

OCx\_EN = 1

OCxREF Invert + Polarity + Dead Zone,

OCxN\_EN = 1

0

0

X

0

0

Output prohibited (disconnected from timer)

0

0

1

Asynchronously: OCx = CCxP, OCx\_EN = 0, OCxN = CCxNP,

0

1

0

OCxN\_EN = 0;

0

1

1

If the clock exists: after a dead time

OCx = OISx, OCxN = OISxN,

It is assumed that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN.

1

0

0

Off state (output is enabled and inactive level)

1

0

1

Asynchronously: OCx = CCxP, OCx\_EN = 1, OCxN = CCxNP,

1

1

0

OCxN\_EN = 1;

1

1

1

If the clock exists: after a dead time

OCx = OISx, OCxN = OISxN,

It is assumed that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN.

1. If the two outputs of a channel are not used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP

And CCxNP must be cleared.

Note 1: The state of the external I/O pins connected to complementary OCx and OCxN channels depends on the state of the OCx and OCxN channels

And GPIO and AFIO registers.

2: If CCxE=0 and CCxNE=0, OCx and OCxN are high impedance after the output is disabled.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

165/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 188** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**10.4.10**

**(TIMx\_CNT)**

Offset address: 0x24

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CNT

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CNT

rw

0x0000

Counter value

**10.4.11**

**(TIMx\_PSC)**

Offset address: 0x28

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

PSC

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

PSC

rw

0x0000

Prescaler value

The clock frequency of the counter (CK\_CNT) is equal to f CK\_PSC /(PSC + 1).

The PSC includes each time the update event occurs, load the current prescaler

The value of the memory. Update events include the counter being cleared by the UG bit of TIM\_EGR

'0' or the slave controller working in reset mode clears '0'.

**10.4.12**

**(TIMx\_ARR)**

Offset address: 0x2C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

ARR

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

ARR

rw

0x0000

Prescaler value

ARR contains the value to be loaded into the actual auto-reload register.

For details, refer to section [10.3.](https://translate.googleusercontent.com/translate_f#130) 1: Updates and actions related to ARR.

When the value of auto-reload is empty, the counter does not work.

**10.4.13**

**(TIMx\_RCR)**

Offset address: 0x30

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

166/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 189** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

REP

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, always read as 0.

7:0

REP

rw

0x00

Repetition counter value

After the preload function is enabled, these bits allow the user to set the compare register

Update rate (ie periodically transferred from the preload register to the current register);

If the update interrupt is allowed, it will also affect the speed at which the update interrupt is generated

rate.

Each time the down counter REP\_CNT reaches 0, an update event will be generated

And the counter REP\_CNT starts counting from the REP value again. due to

REP\_CNT only reloads REP when the periodic update event U\_RC occurs

Value, so the new value written to the TIMx\_RCR register is only changed in the next cycle

Only take effect when new events occur.

This means that in PWM mode, (REP+1) corresponds to:

-In edge-aligned mode, the number of PWM cycles

-In center-symmetric mode, the number of PWM half cycles

**10.4.14**

**/**

**1(TIMx\_CCR1)**

Offset address: 0x34

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CCR1

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR1

rw

0x0000

Capture/Compare 1 value

If the CC1 channel is configured as an output:

CCR1 contains the value loaded into the current capture/compare 1 register (preload

value).

If the preload is not selected in the TIMx\_CCMR1 register (OC1PE bit)

Load function, the written value will be transferred to the current register immediately. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 1

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC1 port.

If the CC1 channel is configured as an input:

CCR1 contains the count transmitted by the last input capture 1 event (IC1)

器值。 Device value. Device value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

167/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 190** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**10.4.15**

**/**

**2(TIMx\_CCR2)**

Offset address: 0x38

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CCR2

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR2

rw

0x0000

Capture/Compare 2 value

If the CC2 channel is configured as an output:

CCR2 contains the value loaded into the current capture/compare 2 register (preload

value).

If the preload is not selected in the TIMx\_CCMR2 register (OC2PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 2

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC2 port.

If the CC2 channel is configured as an input:

CCR2 contains the count transmitted by the last input capture 2 event (IC2)

器值。 Device value. Device value.

**10.4.16**

**/**

**3(TIMx\_CCR3)**

Offset address: 0x3C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CCR3

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR3

rw

0x0000

Capture/Compare 3 value

If the CC3 channel is configured as an output:

CCR3 contains the value loaded into the current capture/compare 3 register (preload

value).

If the preload is not selected in the TIMx\_CCMR3 register (OC3PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 3

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC3 port.

If the CC3 channel is configured as input:

CCR3 contains the count transmitted by the last input capture 3 event (IC3)

器值。 Device value. Device value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

168/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 191** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**10.4.17**

**/**

**4(TIMx\_CCR4)**

Offset address: 0x40

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CCR4

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR4

rw

0x0000

Capture/Compare 4 value

If the CC4 channel is configured as output:

CCR4 contains the value loaded into the current capture/compare 4 register (preload

value).

If the preload is not selected in the TIMx\_CCMR4 register (OC4PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 4

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC4 port.

If the CC4 channel is configured as input:

CCR4 contains the count transmitted by the last input capture 4 event (IC4)

器值。 Device value. Device value.

**10.4.18**

**(TIMx\_BDTR)**

Offset address: 0x44

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OSSI

DTG

MOE AOE BKP BKE OSSR

LOCK

DOE

Reserved

Note: According to the lock setting, the AOE, BKP, BKE, OSSI, OSSR and DTG bits can be write-protected, it is necessary to

Configure them when writing to the TIMx\_BDTR register.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:17

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

169/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 192** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

16

DOE

rw

0x00

Direct output (Direct output enable)

When the brake is valid and the MOE is set to zero, it is valid.

1: The idle state is output immediately, and it is output after waiting for the dead time.

0: After the brake input, wait for a dead time and output the idle state.

15

MOE

rw

0x00

Main output enable

When the brake input is valid, this bit is asynchronously cleared to '0' by hardware. According to AOE bit

The setting value of this bit can be cleared to 0 by software or set to 1 automatically. It only

Valid for channels configured as output.

0: Disable OC and OCN output or force to idle state

1: If the corresponding enable bit (TIMx\_CCER register is set)

CCxE, CCxNE bit), then OC and OCN output is turned on

For details on OC/OCN enable, see Section [10.4.9](https://translate.googleusercontent.com/translate_f#184) , Capture/Compare Enable

Energy register (TIMx\_CCER).

14

AOE

rw

0x00

Automatic output enable

0: MOE can only be set to '1' by the software

1: MOE can be set to '1' by the software or automatically set to 1 at the next update event (eg

(If the brake input is invalid)

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 1

At this time, this bit cannot be modified.

13

BKP

rw

0x00

Brake input polarity (Break polarity)

0: Low level of brake input is effective

1: Brake input high level is effective

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 1

At this time, this bit cannot be modified.

12

BKE

rw

0x00

Break enable

0: Disable brake input (BRK and BRK\_ACTH)

1: Turn on the brake input (BRK and BRK\_ACTH)

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 1

At this time, this bit cannot be modified.

11

OSSR

rw

0x00

Off-state selection for Run in Run mode

mode)

This bit is used when MOE = 1 and the channel is a complementary output. No complementary output

There is no OSSR bit in the timer.

Refer to OC/OCN enable detailed description (Section [10.4.9](https://translate.googleusercontent.com/translate_f#184) , Capture/Compare Enable

Energy register (TIMx\_CCER)).

0: Disable OC/OCN output when the timer does not work (OC/OCN enables

Can output signal = 0)

1: When the timer does not work, if CCxE = 1 or CCxNE = 1, the first

First turn on OC/OCN and output invalid level, then set OC/OCN to enable

Output signal = 1

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 2

At this time, this bit cannot be modified.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

170/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 193** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

10

OSSI

rw

0x00

Off-state selection for Idle in idle mode

mode)

This bit is used when MOE = 0 and the channel is set to output.

Refer to OC/OCN enable detailed description (Section [10.4.9](https://translate.googleusercontent.com/translate_f#184) , Capture/Compare Enable

Energy register (TIMx\_CCER)).

0: Disable OC/OCN output when the timer does not work (OC/OCN enables

Can output signal = 0)

1: When the timer is not working, if CCxE = 1 or CCxNE = 1,

OC/OCN first outputs its idle level, then OC/OCN enables output

Signal = 1

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 2

At this time, this bit cannot be modified.

9:8

LOCK

rw

0x00

Lock configuration

This bit provides write protection to prevent software errors.

00: Lock off, no register write protection

01: Lock level 1, cannot write DTG, TIMx\_BDTR register,

OISx/OISxN of BKE, BKP, AOE bits and TIMx\_CR2 register

Bit

10: Lock level 2, cannot write to everyone in lock level 1, nor write

Enter the CC polarity bit (when the relevant channel is set to output through the CCxS bit, CC

The polarity bit is the CCxP/CCNxP bit in the TIMx\_CCER register) and

OSSR/OSSI bit

11: Lock level 3, you cannot write to everyone in lock level 2, nor can you write

Into the CC control bit (when the relevant channel is set to output through the CCxS bit, CC

The control bit is the OCxM/OCxPE bit in the TIMx\_CCMRx register)

Note: After the system is reset, the LOCK bit can only be written once, when written to TIMx\_BDTR

When registering, its content is frozen until reset.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

171/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 194** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7:0

DTG

rw

0x00

Dead-time generator setup (Dead-time generator setup)

These bits define the duration of the dead zone inserted between complementary outputs. Assuming DT

Indicates its duration:

DTG[7: 5] = 0xx:

DT = (DTG[7: 0] + 1) × t dtg , t dtg = t DTS ;

DTG[7: 5] = 10x:

DT = (DTG[5: 0] + 1 + 64) × t dtg , t dtg = 2 × t DTS ;

DTG[7: 5] = 110:

DT = (DTG[4: 0] + 1 + 32) × t dtg , t dtg = 8 × t DTS ;

DTG[7: 5] = 111:

DT = (DTG[4: 0] + 1 + 32) × t dtg , t dtg = 16 × t DTS ;

Example: If t DTS = 125ns (8MHz), the possible dead time is:

125ns to 15875ns (step time is 125ns),

16µs to 31750ns (step time is 250ns),

32µs to 63µs (step time is 1µs),

64µs to 126µs (step time is 2µs).

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 1,

At 2 or 3, these bits cannot be modified.

**10.4.19 DMA**

**(TIMx\_DCR)**

Offset address: 0x48

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

DBA

DBL

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

172/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 195** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

12:8

DBL

rw

0x00

DMA continuous transfer length (DMA burst length)

These bits define the transfer length of DMA in continuous mode (when the

When the TIMx\_DMAR register is read or written, the timer

Continued transmission), that is: define the number of transmissions, the transmission can be half-word (double-byte)

Or bytes:

00000: 1 transmission 00001: 2 transmissions

00010: 3 transmissions...

...10001: 18 transmissions

Example: We consider such a transmission: DBL = 7, DBA = TIM2\_CR1

-If DBL = 7, DBA = TIM2\_CR1 indicates the address of the data to be transmitted,

Then the transmitted address is given by:

(Address of TIMx\_CR1) + DBA + (DMA index), where DMA index

= DBL

Among them (the address of TIMx\_CR1) + DBA plus 7, gives the write

Enter or read the address of the data, so that the data transfer will occur at the slave address

(Address of TIMx\_CR1) + 7 registers starting with DBA. According to DMA

The setting of the data length may occur as follows:

-If the data is set to halfword (16 bits), the data will be transferred to all 7

Registers.

-If the data is set to bytes, the data will still be transferred to all 7 registers:

The first register contains the first MSB byte, the second register contains the first

One LSB byte, and so on. Therefore, for the timer, the user must indicate

Set the width of data transferred by DMA.

7:5

Reserved

Reserved, always read as 0.

4:0

DBA

rw

0x00

DMA base address (DMA base address) These bits define the DMA in

Base address in continuous mode (when reading the TIMx\_DMAR register or

When writing), DBA is defined as starting from the address where the TIMx\_CR1 register is located

Offset:

00000: TIMx\_CR1

00001: TIMx\_CR2

00010: TIMx\_SMCR

...

**10.4.20**

**DMA**

**(TIMx\_DMAR)**

Offset address: 0x4C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

DMAB

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

173/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 196** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

DMAB

rw

0x0000

DMA register for burst accesses

Reading or writing to the TIMx\_DMAR register will cause

Memory access operation:

TIMx\_CR1 address + DBA + DMA index, where:'TIMx\_CR1

Address' is the address where the control register 1 (TIMx\_CR1) is located;

'DBA' is the base address defined in the TIMx\_DCR register;

'DMA index' is the offset automatically controlled by DMA, it depends on

DBL defined in the TIMx\_DCR register.

**10.4.21**

**/**

**3(TIMx\_CCMR3)**

Offset address: 0x54

Reset value: 0x0000

Channel only for output (comparison mode)

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

OC5

CE

OC5M

OC5

PE

OC5

FE

Reserved

rw

rw

rw

rw

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, always read as 0.

7

OC5CE

rw

0x00

Output compare 5 clear '0' enable (Output compare 5 clear enable)

6:4

OC5M

rw

0x00

Output compare 5 mode

3

OC5PE

rw

0x00

Output compare 5 preload en-

able)

2

OC5FE

rw

0x00

Output compare 5 fast enable (Output compare 5 fast enable)

1:0

Reserved

Reserved, always read as 0.

**10.4.22**

**/**

**5(TIMx\_CCR5)**

Offset address: 0x58

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CCR5

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

174/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 197** |

UM\_MM32F003\_q\_Ver1.19

(TIM1)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR5

rw

0x0000

Capture/Compare 5 value

The CC5 channel can only be configured as an output:

CCR5 contains the value loaded into the current capture/compare 5 register (preload

value).

If the preload is not selected in the TIMx\_CCMR3 register (OC5PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When the update event occurs, this preload value is transferred to the current capture/compare

5 in the register.

The current capture/compare register participates in the comparison with the counter TIMx\_CNT, and

Generate an output signal on the OC5 port.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

175/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 198** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

11 **16**

Use **(TIMx16 Bit)**

16-bit general-purpose timer (TIMx16 Bit)

**11.1 TIMx**

The general-purpose timer is a 16-bit auto-loading counter driven by a programmable prescaler. It is suitable for many

Occasions include measuring the pulse length of an input signal (input capture) or generating an output waveform (output comparison and PWM).

Using timer prescaler and RCC clock controller prescaler, pulse length and waveform period can be in a few microseconds

Adjust to a few milliseconds.

The TIMx timer is completely independent and does not share any resources with each other. They can be operated synchronously together.

**11.2 TIMx**

General TIMx (TIM3) timer functions include:

• 16-bit up, down, up/down auto-loading counter

• 16-bit programmable (can be modified in real time) prescaler, counter clock frequency division factor is between 1 ∼ 65536

Any value of

• 4 independent channels

**–** Input capture

**–** Output comparison

**–** PWM generation (edge ​​​or center alignment mode)

**–** Single pulse mode output

• Use external signal to control timer and timer interconnection synchronization circuit

• An interrupt/DMA is generated when the following events occur:

**–** Update: counter overflow/downflow, counter initialization (triggered by software or internal/external)

**–** Trigger events (counter start, stop, initialization or counting by internal/external trigger)

**–** Input capture

**–** Output comparison

• Supports incremental (quadrature) encoder and Hall sensor circuits for positioning

• Trigger input as external clock or cycle-by-cycle current management

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

176/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 199** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

073662

enter

filter

ITR0

ITR1

ITR2

ITR3

ITR

ETRP

TRGI

TRC

TI1FP1

TI2FP2

ETRF

(CK\_INT)

TRGO

ETR

CNT counter

+/-

PSC

Prescaler

CK\_CNT

CK\_PSC

U

ETRF

Prescaler

IC1PS U

Prescaler

IC2PS

Prescaler

IC3PS

Prescaler

IC4PS

IC1

TI1FP1

TI1FP2

TRC

TI2FP1

TI2FP2

TRC

IC2

IC3

IC4

TRC

TRC

TI1

XOR

TI2

TI3

TIMx\_ETR

TIMx\_CH1

TIMx\_CH2

TIMx\_CH3

TIMx\_CH4

TI4

U

U

U

OC1REF

CC1I

CC2I

OC2REF

OC3REF

CC3I

CC4I

OC4REF

TGI

CC1I

CC2I

CC3I

CC4I

U

UI

OC1N

OC1

OC2

TIMx\_CH1

OC2N

OC3

OC3N

OC4

TI1F\_ED

TI3FP3

TI3FP4

TI4FP3

TI4FP4

TIMx\_CH1N

TIMx\_CH2

TIMx\_CH2N

TIMx\_CH3

TIMx\_CH3N

TIMx\_CH4

Internal clock

TIMxCLK from RCC

Polarity selection, edge

Detector and prescaler

trigger

Controller

Slave mode

Controller

Encoder

interface

Reset, enable, increment */* decrement, count

To other timers

To DAC/ADC

Auto-reload register

Input filter and

Edge detector

Input filter and

Edge detector

Input filter and

Edge detector

Input filter and

Edge detector

Capture/Compare 1 register

Capture/Compare 3 register

Capture/Compare 2 register

Capture/Compare 4 registers

Output

control

Output

control

Output

control

Output

control

Stop, clear, or increase */* decrease

⌘˖

ṩᦞ᧗ࡦսˈ൘ਁ⭏ 8 һԦਾˈ亴㻵䖭ᇴᆈಘcheck ᇩ䖜〫ࡠᴹ᭸ᇴᆈಘ

һԦ

ѝᯝ઼'0$䗃ࠪ

U

Reg

78.

use

**11.3 TIMx**

**11.3.1**

The main part of the programmable general-purpose timer is a 16-counter and its associated auto-load register. This count

The device can count up, down, or up and down. The counter clock is divided by the prescaler.

The counter, auto-load register and prescaler register can be read and written by software, and can still be read and written while the counter is running,

The time base unit contains:

• Counter register (TIMx\_CNT)

• Prescaler register (TIMx\_PSC)

• Auto load register (TIMx\_ARR)

The auto-reload register is pre-loaded, and writing or reading the auto-reload register will access the pre-load register. According to

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

177/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 200** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

The setting of the autoload preload enable bit (ARPE) in the TIMx\_CR1 register, the content of the preload register is set

That is, it is transferred to the shadow register at each update event UEV. When the counter reaches the overflow condition (counting down

Underflow condition) and when the UDIS bit in the TIMx\_CR1 register is equal to 0, an update event is generated. Update events are also available

To be generated by the software. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK\_CNT of the prescaler, only if the counter in the TIMx\_CR1 register is set

CK\_CNT is valid only when the counter enables bit (CEN). (For details on counter enable, please refer to the slave

Mode description).

The prescaler can divide the counter clock frequency by any value between 1 and 65536. It is based on a (in

A 16-bit counter controlled by a 16-bit register in the TIMx\_PSC register. Because this control register is buffered

It can be changed while working. The parameters of the new prescaler are adopted when the next update event arrives.

The following two figures respectively give examples of changing the counter parameters while the prescaler is running.

059785

CK\_PSC

CEN

ᇊᰦಘᰦ䫏 = CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

02

03

0

1

1

0

0

0

0

0

0

1

1

1

1

TIMx\_PSCᇴᆈಘ

79.

1

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

178/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 201** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

763391

CK\_PSC

CEN

ᇊᰦಘᰦ䫏=CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

0

3

3

0

0

0

2

2

0

1

3

1

3

TIMx\_PSCᇴᆈಘ

80.

1

4

**11.3.2**

In the up-counting mode, the counter counts from 0 to the autoload value (the content of the TIMx\_ARR counter), and then restarts

Start counting from 0 and generate a counter overflow event.

An update event can be generated each time the counter overflows, and the UG bit is set in the TIMx\_EGR register (via software

Or use slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIMx\_CR1 register can disable the update event; this can avoid registering to the preload

The shadow register is updated when a new value is written in the device. Until the UDIS bit is cleared, no update event will be generated. But in response

When the update event occurs, the counter will still be cleared to 0, and the count of the prescaler will also be cleared to 0 (but the value of the prescaler

constant). In addition, if the URS bit in the TIMx\_CR1 register is set (select an update request), setting the UG bit will

An update event UEV is generated, but the UIF flag is not set by the hardware (that is, no interrupt or DMA request is generated). This is for

Avoid generating update and capture interrupts simultaneously when the counter is cleared in capture mode.

When an update event occurs, all registers are updated, and the hardware sets the update flag at the same time (according to the URS bit)

Bit (UIF bit in the TIMx\_SR register).

• The buffer of the prescaler is placed in the value of the preload register (the content of the TIMx\_PSC register)

• The autoload shadow register is reset to the value of the preload register (TIMx\_ARR)

The following figure shows some examples. When TIMx\_ARR = 0x36, the counter will operate at different clock frequencies:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

179/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 202** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

894901

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

31

32 33 34 35 36

05

Counter overflow

Update interrupt flag (UIF)

00 01 02 03 04

06 07

81.

1

100720

0034

0035

0000

0036

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

82.

2

564964

0035

0000

0036

0001

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

83.

4

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

180/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 203** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

372226

1F

20

00

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

84.

N

370803

Write new value in TIMx\_ARR

31

32

33 34

35

36

05

00 01

02

03

04

06

07

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

85.

ARPE = 0

(TIMx\_ARR

)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

181/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 204** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

874537

Write new value in TIMx\_ARR

F0

F1 F2 F3 F4 F5

05

00 01 02 03 04

06 07

F5

36

F5

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

86.

ARPE = 1

(TIMx\_ARR)

In down mode, the counter counts down to 0 from the automatically loaded value (TIMx\_ARR counter value), and then

It restarts from the automatically loaded value and generates a counter underflow event.

An update event can be generated each time the counter overflows, and the UG bit is set in the TIMx\_EGR register (via software

Or use slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIMx\_CR1 register can disable UEV events. This can avoid preloading registers

The shadow register is updated when a new value is written. Therefore, no update event will be generated until the UDIS bit is cleared to 0. However, counting

The counter will still restart counting from the current auto-load value, and the counter of the prescaler restarts from 0 (but the prescaler

The speed of the device cannot be modified).

In addition, if the URS bit in the TIMx\_CR1 register is set (select update request), setting the UG bit will generate a

Update event UEV without setting the UIF flag (so no interrupts and DMA requests are generated), this is to avoid

When the event is captured and the counter is cleared, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and (according to the setting of the URS bit) update the flag bit (TIMx\_SR

The UIF bit in the register) is also set.

• The prescaler buffer is set to the value of the preload register (the value of the TIMx\_PSC register).

• The current autoload register is updated to the preload value (the content in the TIMx\_ARR register).

Note: Autoload is updated before the counter is reloaded, so the next cycle will be the expected value.

The following are some examples of counter operation at different clock frequencies when TIMx\_ARR = 0x36:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

182/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 205** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

721721

05

04

03

02

01

00

31

36

35

34

33

32

30

2F

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

87.

1

819787

0002

0001

0036

0000

0035

0034

0033

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

88.

2

213411

0001

0036

0000

0035

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

89.

4

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

183/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 206** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

264550

20

1F

00

36

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

90.

N

367681

05

04

03 02

01

00

31

36 35

34

33

32

30

2F

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

Write new value in TIMx\_ARR

91.

use

**(**

**/**

**)**

In center-aligned mode, the counter starts counting from 0 to the automatically loaded value (TIMx\_ARR register) -1, generating a

Counter overflow event, then count down to 1 and generate a counter underflow event; then start again from 0

count.

In this mode, the DIR direction bit in TIMx\_CR1 cannot be written. It is updated by the hardware and indicates the current counting method

to. The update event can be generated at every count overflow and every count underflow; can also be passed (software or use slave mode

Controller) Set the UG bit in the TIMx\_EGR register to generate, at this time, the counter starts counting again from 0

The frequency counter starts counting from 0 again.

Setting the UDIS bit in the TIMx\_CR1 register can disable UEV events. This can avoid pre-loading registration

The shadow register is updated when a new value is written in the device. Therefore, no update event will be generated until the UDIS bit is cleared to 0. However,

The counter will continue to count up or down based on the current auto-reload value.

In addition, if the URS bit in the TIMx\_CR1 register is set (select update request), setting the UG bit will generate a

Update event UEV without setting the UIF flag (so no interrupts and DMA requests are generated), this is to avoid

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

184/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 207** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

When the event is captured and the counter is cleared, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and (according to the setting of the URS bit) update the flag bit (TIMx\_SR

The UIF bit in the register) is also set.

• The prescaler buffer is loaded with the preload (TIMx\_PSC register) value

• The current autoload register is updated to the preload value (the content in the TIMx\_ARR register)

Note: If an update occurs due to a counter overflow, the automatic reload will be updated before the counter is reloaded, so the next cycle

It will be the expected value (the counter is loaded with the new value).

The following are some examples of counter operations at different clock frequencies:

775077

04

03

02 01

00 01

05

02 03 04 05 06

04 03

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Counter underflow

92.

1TIMx\_ARR = 0x6

824268

0003

0002

0000

0001

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

93.

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

185/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 208** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

464735

0034

0036

0035

0035

Note: Center-aligned mode 2 or mode 3 is used in conjunction with the overflow UIF.

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

94.

4TIMx\_ARR = 0x36

220046

20

1F

00

01

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

95.

N

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

186/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 209** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

482632

06

05

04 03

02

01

05

00

01 02

03

04

06

07

FD

36

FD

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

Write new value in TIMx\_ARR

96.

ARPE = 1

()

970767

F7

F8

F9 FA FB FC

31

36

35 34

33

32

30

2F

FD

36

FD

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

Write new value in TIMx\_ARR

97.

ARPE = 1

()

**11.3.3**

The counter clock can be provided by the following clock sources:

• Internal clock (CK\_INT)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

187/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 210** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

• External clock mode 1: External input pin (TIx)

• External clock mode 2: External trigger input (ETR)

• Internal trigger input (ITRx): use one timer as the prescaler of another timer, if you can configure one

Timer1 acts as a prescaler for another timer, Timer2.

**(CK\_INT)**

If the slave mode controller is disabled (SMS = 000), the CEN, DIR (TIMx\_CR1 register) and UG bits (TIMx\_

EGR register) is the de facto control bit and can only be modified by software (UG bit is still automatically cleared). When the CEN bit

When written as 1, the prescaler clock is provided by the internal clock CK\_INT.

The figure below shows the operation of the control circuit and the up-counter in normal mode without the prescaler.

199864

31

32

33 34

35

36

05

00

01

02

03

04

06

07

Internal clock

CEN = CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

CNT\_INT

UG

98.

1

**1**

When SMS = 111 in the TIMx\_SMCR register, this mode is selected. The counter can be

Count on a rising or falling edge.

356901

Encoder mode

External clock

Mode 1

External clock

Mode 2

Internal clock

mode

CK\_PSC

filter

ICF[3:0]

edge

Detector

CC2P

TS[2:0]

SMS[2:0]

ECE

TI2

TIMx\_CCMR1

TIMx\_CCER

TI2F\_Rising

TI2F\_Falling

ITRx

TI1F\_ED

TI1FP1

TI2FP2

ETRF

001

100

101

110

111

TRGI

ETRF

CK\_INT

(Internal clock)

TIMx\_SMCR

TI2F

TI1F

or

or

or

TIMx\_SMCR

0

1

99. TI2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

188/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 211** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

For example, to configure the up counter to count on the rising edge of the T12 input, use the following steps:

1. Configure TIMx\_CCMR1 register CC2S = 01, configure channel 2 to detect the rising edge of TI2 input

2. Configure IC2F[3:0] in the TIMx\_CCMR1 register and select the input filter bandwidth (if the filter is not required, ensure

Hold IC2F = 0000)

Note: The capture prescaler is not used as a trigger, so there is no need to configure it

3. Configure CC2P = 0 of the TIMx\_CCER register to select the polarity of the rising edge

4. Configure SMS = 111 in the TIMx\_SMCR register to select the timer external clock mode 1

5. Configure TS = 110 in the TIMx\_SMCR register and select TI2 as the trigger input source

6. Set CEN = 1 in the TIMx\_CR1 register to start the counter

When the rising edge appears on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the resynchronization circuit at the input of TI2.

518995

TI2

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

Write TIF = 0

34

36

TIF

35

100.

1

**2**

The method to select this mode is: let ECE = 1 in the TIMx\_SMCR register to trigger the ETR externally

Each rising or falling edge counts. The following figure is the overall block diagram of the external trigger input:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

189/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 212** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

132558

Encoder mode

External clock

Mode 1

External clock

Mode 2

Internal clock

mode

CK\_PSC

filter

Down counter

Crossover

/1,/2,/4,/8

SMS[2:0]

ECE

ETR pin

ETR

TRGI

ETRF

CK\_INT

(Internal clock)

TI2F

TI1F

or

or

or

0

1

ETP

ETRP

CK\_INT

ETF[3:0]

ETPS[1:0]

TIMx\_SMCR

TIMx\_SMCR

TIMx\_SMCR

TIMx\_SMCR

101.

For example, to configure an up counter that counts every 2 rising edges under ETR, use the following steps:

1. No filter is needed in this example, set ETF[3:0] = 0000 in the TIMx\_SMCR register

2. Set the prescaler and set ETPS[1:0] = 01 in the TIMx\_SMCR register

3. Set the detection on the rising edge of ETR, set ETP = 0 in the TIMx\_SMCR register

4. Turn on external clock mode 2 and set ECE = 1 in the TIMx\_SMCR register

5. Start the counter and set CEN = 1 in the TIMx\_CR1 register

The counter counts every 2 rising edges of ETR.

The delay between the rising edge of ETR and the actual clock of the counter depends on the resynchronization circuit at the ETRP signal terminal.

052284

34

36

35

ETRP

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

ETRF

ETR

f CK\_INT

102.

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

190/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 213** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.3.4**

**/**

Each capture/compare channel surrounds a capture/compare register (including shadow registers), including the captured input

Part (digital filtering, multiplexing and prescaler), and output part (comparator and output control).

The following figures are an overview of the capture/compare channels. The input section samples the corresponding TIx input signal and generates a

Filtered signal TIxF. Then, an edge monitor with polarity selection generates a signal (TIxFPx), which

Can be triggered as input from the mode controller or as a capture control. This signal enters the capture register through prescaler

(ICxPS).

565511

IC1PS

filter

Down counter

ICF[3:0]

edge

Detector

CC1P

ICPS[1:0]

TI1

TIMx\_CCMR1

TIMx\_CCER

TI1F\_Rising

TI1F\_Falling

01

10

11

(Since mode controller)

0

1

f DTS

TI1F

0

1

divider

/1,/2,/4,/8

IC1

CC1E

CC1S[1:0]

TIMx\_CCMR1

TIMx\_CCER

TRC

TI2F\_Rising

TI2F\_Falling

(from channel 2)

(from channel 2)

TI2FP1

TI1FP1

TI1F\_ED

To slave mode controller

103.

/ ( 1

)

The output section generates an intermediate waveform OCxRef (high effective) as a reference, the end of the chain determines the pole of the final output signal

Sex.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

191/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 214** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

903065

MCU peripheral interface

APB bus

Capture/compare preload registers

high

8

CC1S[1]

OC1PE

TIMx\_CCMR1

UEV

Reading CCR1L

Reading CCR1H

CC1S[1]

CC1S[0]

IC1PS

CC1E

CC1G

TIMx\_EGR

read\_in\_progress

enter

mode

8

low

(Assuming 16 bits)

Capture/Compare Shadow Register

capture

counter

Comparators

capture\_transfer

compare\_transfer

write\_in\_progress

Output

mode

CNT> CCR1

CNT = CCR1

(From time base unit)

Write to CCR1H

Write to CCR1L

CC1S[0]

OC1PE

S

R

S

R

104.

/ 1

358425

Output mode

Controller

OC1M[2:0]

TIMx\_CCMR1

CNT>CCR1

CNT=CCR1

ETRF

OC1ref

To slave mode controller

0

1

CC1P

TIMx\_CCER

Output enable

Circuit

CC1E TIMx\_CCER

OC1

105.

/ ( 1)

The capture/compare module consists of a preload register and a shadow register. The read and write process only operates preload registration

Device. In capture mode, the capture occurs on the shadow register, and then copied into the preload register.

In the comparison mode, the contents of the preload register are copied to the shadow register, and then the contents of the shadow register and

Counter.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

192/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 215** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.3.5**

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched to capture/ratio

Compare register (TIMx\_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx\_SR register) is

Set to 1, if interrupt or DMA operation is enabled, an interrupt or DMA operation will be generated. If a capture event occurs

When the CCxIF flag is already high, the repeated capture flag CCxOF (TIMx\_SR register) is set. Write CCxIF

= 0 can clear CCxIF, or read the captured data stored in the TIMx\_CCRx register can also clear CCxIF. write

CCxOF = 0 can clear CCxOF.

The following example shows how to capture the counter value into the TIMx\_CCR1 register on the rising edge of the TI1 input, step

as follows:

• Select valid input: TIMx\_CCR1 must be connected to TI1 input, so write to TIMx\_CCR1 register

CC1S = 01, when CC1S is not 00, the channel is configured as an input, and the TM1\_CCR1 register becomes

Is read-only.

• According to the characteristics of the input signal, configure the input filter to the required bandwidth (that is, when the input is TIx, the input filter controls

The control bit is the ICxF bit in the TIMx\_CCMRx register). Assuming that the input signal is at most 5 clock cycles

For internal jitter, we must configure the filter bandwidth to be longer than 5 clock cycles. So we can connect (at fDTS frequency)

Continue sampling 8 times to confirm the last real edge transition on TI1, which is written in the TIMx\_CCMR1 register

IC1F = 0011.

• Select the valid conversion edge of the TI1 channel and write CC1P = 0 (rising edge) in the TIMx\_CCER register.

• Configure input prescaler. In this example, we want to capture every valid level-shifting moment, so

The prescaler is disabled (write IC1PS = 00 in the TIMx\_CCMR1 register).

• Set CC1E = 1 in the TIMx\_CCER register to allow the capture counter value to be captured in the capture register.

• If necessary, enable the relevant interrupt request by setting the CC1IE bit in the TIMx\_DIER register, by setting

The CC1DE bit in the TIMx\_DIER register allows DMA requests.

When an input is captured:

• When a valid level transition occurs, the counter value is transferred to the TIMx\_CCR1 register.

• The CC1IF flag is set (interrupt flag). When at least 2 consecutive captures occur, CC1IF has not been cleared.

• CC1OF is also set to 1.

• If the CC1IE bit is set, an interrupt will be generated.

• If the CC1DE bit is set, a DMA request will also be generated.

In order to deal with the capture overflow, it is recommended to read the data before reading the capture overflow flag, this is to avoid losing the read capture

Capture overflow information that may be generated after the overflow flag is obtained and before the data is read.

Note: By setting the corresponding CCxG bit in the TIMx\_EGR register, an input capture interrupt and/or DMA request can be generated by software.

**11.3.6 PWM**

This mode is a special case of the input capture mode, except for the following differences, the operation is the same as the input capture mode:

• Two ICx signals are mapped to the same TIx input

• 2 ICx signals are edge valid, but the polarity is reversed

• One of the TIxFP signals is used as a trigger input signal, and the slave mode controller is configured in reset mode

For example, you need to measure the length (TIMx\_CCR1 register) and duty cycle of the PWM signal input to TI1

(TIMx\_CCR2 register), the specific steps are as follows (depending on the frequency of CK\_INT and the value of prescaler)

• Select the valid input of TIMx\_CCR1: Set CC1S = 01 in the TIMx\_CCMR1 register (select TI1)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

193/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 216** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

• Select the effective polarity of TI1FP1 (used to capture data into TIMx\_CCR1 and clear the counter): set CC1P =

0 (rising edge valid).

• Select the valid input of TIMx\_CCR2: Set CC2S = 10 in the TIMx\_CCMR1 register (select TI1).

• Select the effective polarity of TI1FP2 (capture data to TIMx\_CCR2): set CC2P = 1 (active on falling edge).

• Select a valid trigger input signal: set TS = 101 in the TIMx\_SMCR register (select TI1FP1).

• Configure the slave mode controller to reset mode: set SMS = 100 in TIMx\_SMCR.

• Enable capture: Set CC1E = 1 and CC2E = 1 in the TIMx\_CCER register.

789042

0004

0000

0001

0002

0003

0004

0000

0004

0002

TI1

TIMx\_CNT

TIMx\_CCR1

TIMx\_CCR2

IC1 capture

IC2 capture

Reset counter

IC2 capture

Pulse width measurement

IC1 capture

Period measurement

106.

/ ( 1)

Since only TI1FP1 and TI2FP2 are connected to the slave mode controller. So PWM input mode can only use TIMx\_CH1

/ TIMx\_CH2 signal.

**11.3.7**

In the output mode (CCxS = 00 in the TIMx\_CCMRx register), the comparison signal (OCxREF and the corresponding

OCx) can be forced into valid or invalid state directly by software, without depending on the output compare register and counter

Comparing results.

Set the corresponding OCxM = 101 in the TIMx\_CCMRx register to force the output compare signal (OCxREF/OCx)

Is valid. In this way, OCxREF is forced to high level (OCxREF is always active high), and OCx is

To the opposite value of the CCxP polarity bit. For example: CCxP = 0 (OCx high level effective), then OCx is forced to high level.

Set OCxM = 100 in the TIMx\_CCMRx register to force the OCxREF signal low. In this mode, in

The comparison between the TIMx\_CCRx shadow register and the counter is still in progress, and the corresponding flag will be modified. Therefore

Corresponding interrupts and DMA requests will still be generated. This will be described in the section on output comparison mode below.

**11.3.8**

This function is used to control an output waveform or indicate when a given period of time has expired.

When the contents of the counter and the capture/compare register are the same, the output compare function does the following:

• Register the output compare mode (OCxM bit in the TIMx\_CCMRx register) and output polarity (TIMx\_CCER

The value defined by the CCxP bit in the device is output to the corresponding pin. The output pin can hold it when comparing and matching

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

194/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 217** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

Level (OCxM = 000), set to the active level (OCxM = 001), set to no active level (OCxM

= 010) or roll over (OCxM = 011).

• Set the flag bit in the interrupt status register (CCxIF bit in the TIMx\_SR register).

• If the corresponding interrupt mask (CCXIE bit in the TIMx\_DIER register) is set, an interrupt is generated.

• If the corresponding enable bit is set (CCxDE bit in the TIMx\_DIER register, the TIMx\_CR2 register

CCDS bit selects DMA request function), then generate a DMA request.

The OCxPE bit in TIMx\_CCMRx selects whether the TIMx\_CCRx register needs to use a preload register. Losing

In the compare mode, the update event UEV has no effect on the OCxREF and OCx outputs. Synchronization accuracy can be achieved

One counting cycle of the counter. The output compare mode (in single pulse mode) can also be used to output a single pulse.

Configuration steps of output comparison mode:

1. Select counter clock (internal, external, prescaler)

2. Write the corresponding data to the TIMx\_ARR and TIMx\_CCRx registers

3. If an interrupt request and/or a DMA request is to be generated, set the CCxIE bit and/or CCxDE bit

4. Select the output mode, for example: OCxM = '011', OCxPE = '0', CCxP = '0' and

CCxE = '1', when the counter CNT matches CCRx, the output pin of OCx is toggled, CCRx is not preloaded

Use, enable OCx output and high level is effective

5. Set the CEN bit of the TIMx\_CR1 register to start the counter

The TIMx\_CCRx register can be updated by software at any time to control the output waveform, provided the pre-use is not used

Load register (OCxPE = '0', otherwise the TIMx\_CCRx shadow register can only be used when the next update event occurs

Updated). The figure below gives an example.

803963

0039

003A

003B

B200

B201

003A

B201

TIMX\_CNT

TIMX\_CCR1

OC1REF = OC1

Write B201h in CCR1 register

A match was detected on CCR1

If the interrupt is enabled, an interrupt is generated

107.

OC1

**11.3.9 PWM**

Pulse width modulation mode can generate a frequency determined by the TIMx\_ARR register and confirmed by the TIMx\_CCRx register

A signal with a fixed duty cycle.

Write '110' (PWM mode 1) or '111' (PWM mode) in the OCxM bit in the TIMx\_CCMRx register

2), each OCx output channel can be set independently to generate a PWM. The TIMx\_CCMRx register must be set

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

195/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 218** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

OCxPE bit to enable the corresponding preload register, and finally set the ARPx bit of the TIMx\_CR1 register to enable

Preload registers for automatic reload (in up-count or center-symmetric mode).

Because only when an update event occurs, the preload register can be transferred to the shadow register, so in the count

Before the device starts counting, all registers must be initialized by setting the UG bit in the TIMx\_EGR register.

The polarity of OCx can be set by the CCxP bit in the TIMx\_CCER register by software, it can be set to high

Active level or active low level. The CCxE bit in the TIMx\_CCER register controls OCx output enable. See more

Description of the TIMx\_CCERx register.

In PWM mode (mode 1 or mode 2), TIMx\_CNT and TIM1\_CCRx are always compared, (based on the count

The counting direction of the device) to determine whether it meets TIM1\_CCRx ≤ TIM1\_CNT or TIM1\_CNT ≤ TIM1\_CCRx.

However, in order to interact with the function of OCREF\_CLR (before the next PWM cycle, an external event on the ETR signal

Can clear OCxREF). The OCxREF signal can only be generated under the following conditions:

• When the result of the comparison changes

• When the output compare mode (OCxM bit in the TIMx\_CCMRx register) is changed from'frozen' (no comparison, OCxM =

'000') Switch to a PWM mode (OCxM = '110' or '111')

In this way, the PWM output can be forced by software during operation. According to the state of the CMS bit in the TIMx\_CR1 register,

The timer can generate an edge-aligned PWM signal or a center-aligned PWM signal.

**PWM**

Count up when the DIR bit in the TIMx\_CR1 register is low.

The following is an example of PWM mode 1. PWM signal reference OCxREF when TIMx\_CNT <TIMx\_CCRx

High, otherwise low. If the comparison value in TIMx\_CCRx is greater than the auto-reload value (TIMx\_ARR), then OCxREF

Keep it as '1'. If the comparison value is 0, OCxREF remains at '0'. The following figure shows the edge pair when TIMx\_ARR = 8

Examples of homogeneous PWM waveforms.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

196/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 219** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

458343

0 1 2 3 4 5 6 7 8 0 1

þ1ÿ

þ0ÿ

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx = 4

CCRx = 8

CCRx >8

CCRx =0

Counter register

108.

PWM

(ARR = 8)

Countdown is performed when the DIR bit of the TIMx\_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low when TIMx\_CNT> TIMx\_CCRx, otherwise it is high. Such as

If the comparison value in TIMx\_CCRx is greater than the auto-reload value in TIMx\_ARR, OCxREF remains at '1'.

In this mode, a 0% PWM waveform cannot be generated.

**PWM**

When the CMS bit in the TIMx\_CR1 register is not '00', it is the center-aligned mode (all other configurations

/OCx signals have the same effect). According to the setting of different CMS bits, the comparison flag can count up in the counter

Set to 1 at time, set to 1 when the counter counts down, or set to 1 when the counter counts up and down. TIMx\_CR1

The count direction bit (DIR) in the register is updated by hardware, do not modify it by software. See the center alignment mode chapter.

The following figure shows some examples of center-aligned PWM waveforms

• TIMx\_ARR = 8

• PWM mode 1

• CMS = 01 in the TIMx\_CR1 register, in center-aligned mode 1, the ratio is set when the counter counts down

More mark

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

197/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 220** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

546141

0 1 2 3 4 5 6 7 8 7 6

þ1ÿ

þ0ÿ

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx = 4

CCRx = 8

CCRx >8

CCRx =0

CMS =01

CMS =10

CMS =11

CMS =10 or 11

CMS =01

CMS =10

CMS =11

þ1ÿ

CMS =01

CMS =10

CMS =11

CMS =01

CMS =10

CMS =11

5 4 3 2 1 0 1

CCRx =7

OCxREF

CCxIF

Counter register

109.

PWM

(APR = 8)

use

• When entering the center alignment mode, the current up/down counting configuration is used; this means whether the counter is counting up or down

It depends on the current value of the DIR bit in the TIMx\_CR1 register. In addition, the software cannot modify DIR and CMS at the same time

Bit.

• It is not recommended to rewrite the counter when running in center-aligned mode, as it will produce unpredictable results. specifically :

**–** If the value written to the counter is greater than the value of auto-reload (TIMx\_CNT> TIMx\_ARR), the direction

Will not be updated. For example, if the counter is counting up, it will continue to count up.

**–** If the value of 0 or TIMx\_ARR is written to the counter, the direction is updated, but no update event is generated

UEV

• The safest way to use the center alignment mode is to generate a software update (set TIMx\_

UG bit in the EGR bit), do not modify the counter value while the count is in progress.

**11.3.10**

The single pulse mode (OPM) is a special case of the aforementioned many modes. This mode allows the counter to respond to an stimulus and

A programmable pulse is generated after a programmable delay.

The counter can be started by the slave mode controller to generate a waveform in the output comparison mode or PWM mode. Set up

The OPM bit in the TIMx\_CR1 register will select the one-shot mode, which allows the counter to automatically generate the next pulse

Stopped during an update event UEV.

A pulse can be generated only when the comparison value is different from the initial value of the counter. Before starting (when the timer is waiting for a touch

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

198/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 221** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

Issue), must be configured as follows:

• Up counting method: CNT <CCRx ≤ ARR (in particular, 0 <CCRx)

• Countdown method: CNT> CCRx

296813

t DELAY

t PULSE

t

0

meter

Counter

TIM1\_ARR

TIM1\_CCR1

OC1

OC1REF

TI2

110.

For example, you need to detect a rising edge on the input pin of TI2, delay t DELAY, and then generate it on OC1.

Generate a positive pulse of length t PULSE .

Assume TI2FP2 as trigger 1:

• Set CC2S = 01 in the TIMx\_CCMR1 register to map TI2FP2 to TI2

• Set CC2P = 0 in the TIMx\_CCER register to enable TI2FP2 to detect the rising edge

• Set TS = 110 in the TIMx\_SMCR register to use TI2FP2 as the slave mode controller trigger (TRGI)

• Set SMS = 110 (trigger mode) in the TIMx\_SMCR register, TI2FP2 is used to start the counter

The OPM waveform is determined by the value written to the compare register (the clock frequency and counter prescaler must be considered).

• t DELAY is defined by the value written to the TIMx\_CCR1 register.

• t PULSE is defined by the difference between the autoload value and the comparison value (TIMx\_ARR-TIMx\_CCR1).

• Assume that a waveform from 0 to 1 is generated when a compare match occurs, and a slave is generated when the counter reaches the preload value

1 to 0 waveform; first set OC1M = 111 in TIMx\_CCMR1 register to enter PWM mode 2; root

Selectively enable preload registers as needed: set OC1PE = 1 and TIMx\_CR1 in TIMx\_CCMR1

ARPE in the register; then fill in the comparison value in the TIMx\_CCR1 register, in the TIMx\_ARR register

Fill in the autoload value, modify the UG bit to generate an update event, and then wait for an external trigger on TI2

event. In this example, CC1P = 0.

In this example, the DIR and CMS bits in the TIMx\_CR1 register should be set low.

Because only one pulse is needed, it is necessary to set OPM = 1 in the TIMx\_CR1 register for the next update event

(When the counter rolls over from the autoload value to 0) stop counting.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

199/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 222** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**OCx**

In the one-shot mode, the edge detection logic on the TIx input pin sets the CEN bit to start the counter. Then the counter and

The comparison operation between the comparison values ​​​​produces a conversion of the output. But these operations require a certain clock cycle, so it limits

The minimum delay t DELAY available .

If you want to output the waveform with minimum delay, you can set the OCxFE bit in the TIMx\_CCMRx register;

OCxREF (and OCx) are forced to respond to the excitation and no longer depend on the result of the comparison. The output waveform matches the comparison waveform.

Same shape. OCxFE only works when the channel is configured in PWM1 and PWM2 modes.

**11.3.11**

**OCxREF**

For a given channel, set the corresponding OCxCE bit in the TIMx\_CCMRx register at the ETRF input

'1') high level can pull the OCxREF signal low, the OCxREF signal will remain low until the next change occurs

New event UEV.

This function can only be used for output comparison and PWM mode, but not for forced mode.

For example, the OCxREF signal can be connected to an external input. At this time, ETR must be configured as follows:

• The external trigger prescaler must be off: ETPS[1:0] = 00 in the TIMx\_SMCR register

• External clock mode 2 must be disabled: ECE = 0 in the TIMx\_SMCR register

• External trigger polarity (ETP) and external trigger filter (ETF) can be configured as required

The figure below shows the behavior of the OCxREF signal corresponding to different OCxCE values ​​​​when the ETRF input goes high. At this

In this example, the timer TIMx is placed in PWM mode.

079702

Counter ( CNT )

( CCRx )

ETRF

OCxREF

( OCxCE ='0')

OCxREF

( OCxCE ='1')

OCREF \_ CLR

OCREF \_ CLR

Go high

Keep high

111.

TIMx

OCxREF

**11.3.12**

The method to select the encoder interface mode is: if the counter only counts on the edge of TI2, set TIMx\_SMCR to register

SMS = 001 in the device; if only counting on the edge of TI1, set SMS = 010; if the counter is in both TI1 and TI1

TI2 edge count, then set SMS = 011.

The polarity of TI1 and TI2 can be selected by setting the CC1P and CC2P bits in the TIMx\_CCER register; if required

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

200/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 223** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

If necessary, the input filter can also be programmed.

The two inputs TI1 and TI2 are used as the interface for the incremental encoder. The following table assumes that the counter has started (TIMx\_CR1

CEN = 1) in the register, the counter is driven by each valid transition on TI1FP1 or TI2FP2. TI1FP1

And TI2FP2 are the signals after TI1 and TI2 pass the input filter and polarity control; if there is no filtering and disguise, then

TI1FP1 = TI1; if there is no filtering and disguise, then TI2FP2 = TI2. According to the transition sequence of the two input signals, the

Count pulse and direction signal are generated. According to the transition sequence of the two input signals, the counter counts up or down, and at the same time

The hardware sets the DIR bit of the TIMx\_CR1 register accordingly. Regardless of whether the counter counts on TI1,

TI2 counts or counts on TI1 and TI2 simultaneously. The transition at any input (TI1 or TI2) is recalculated

DIR bit.

The encoder interface mode is basically equivalent to using an external clock with direction selection. This means that the counter is only on

Continuous counting from 0 to the autoload value of the TIMx\_ARR register (according to the direction, either 0 to ARR count, or

ARR to 0 count). Therefore, TIMx\_ARR must be configured before starting counting; similarly, the catcher, comparator, pre-

The frequency divider and trigger output characteristics still work as usual.

In this mode, the counter is automatically modified according to the speed and direction of the incremental encoder, so the content of the counter

Finally indicates the position of the encoder. The counting direction corresponds to the direction of rotation of the connected sensor. The following table lists all possible

, Assuming that TI1 and TI2 are not transformed simultaneously.

39.

Effective edge

Relative signal level **(TI1FP1** pair

Should **TI2 of** , **on TI2FP2** corresponds to **TI1)**

**TI1FP1** signal

**TI1FP2** signal

rise

decline

rise

decline

Only count at TI1

high

Count down

Count up

Do not count

Do not count

low

Count up

Count down

Do not count

Do not count

Counting on TI2 only

high

Do not count

Do not count

Count up

Count down

low

Do not count

Do not count

Count down

Count up

Count on TI1 and TI2

high

Count down

Count up

Count up

Count down

low

Count up

Count down

Count down

Count up

An external incremental encoder can be directly connected to the MCU without the need for external interface logic. However, the general use ratio

The comparator converts the differential output of the encoder to a digital signal, which greatly increases the ability to resist noise interference. Encoder output

The third signal represents the mechanical zero point, which can be connected to an external interrupt input and trigger a counter reset.

The following figure is an example of counter operation, showing the generation of count signals and direction control. It also shows when selected

How to suppress the input jitter on both sides of the edge; jitter may be generated when the sensor position is close to a transition point.

In this example, we assume the configuration is as follows:

• C1S = '01' (TIMx\_CCMR1 register, IC1FP1 is mapped to TI1)

• CC2S = '01' (TIMx\_CCMR2 register, IC2FP2 is mapped to TI2)

• CC1P = '0' (TIMx\_CCER register, IC1FP1 is not inverted, IC1FP1 = TI1)

• CC2P = '0' (TIMx\_CCER register, IC2FP2 is not inverted, IC2FP2 = TI2)

• SMS = '011' (TIMx\_SMCR register, all inputs are valid on rising and falling edges)

• CEN = '1' (TIMx\_CR1 register, counter enable)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

201/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 224** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

119663

Increment

TI1

counter

Diminishing

Increment

TI2

Forward

Jitter

Reverse

Jitter

Forward

112.

The following figure shows the operation example of the counter when the polarity of IC1FP1 is inverted (CC1P = '1', other configurations are the same as the above example)

002625

Diminishing

TI1

counter

Increment

Diminishing

TI2

Forward

Jitter

Reverse

Jitter

Forward

113. IC1FP1

When the timer is configured in encoder interface mode, it provides information about the current position of the sensor. Use the second configuration in the capture

The mode timer measures the interval between two encoder events and can obtain dynamic information (speed, acceleration, deceleration). Finger

The encoder output showing the mechanical zero point can be used for this purpose. According to the interval between two events, you can follow a fixed time

Read the counter. If possible, you can latch the counter value to the third input capture register (capture signal

Must be periodic and can be generated by another timer). It can also use a DMA generated by a real-time clock

Request to read its value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

202/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 225** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.3.13**

The TI1S bit in the TIMx\_CR2 register allows the input filter of channel 1 to be connected to the output of an XOR gate.

The three inputs of the OR gate are TIMx\_CH1, TIMx\_CH2, and TIMx\_CH3.

The XOR output can be used for all timer input functions such as triggering or input capture. The Advanced Control Timer chapter gives

An example of using this feature to connect a Hall sensor is presented.

**11.3.14**

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gating mode, and trigger mode.

When a trigger input event occurs, the counter and its prescaler can be re-initialized; meanwhile, if TIMx\_CR1

The URS bit of the register is low, and an update event UEV is also generated; then all the preload registers (TIMx\_ARR

, TIMx\_CCRx) have been updated.

• In the following example, the rising edge of the TI1 input causes the up counter to be cleared

• Configure channel 1 to detect the rising edge of TI1. Configure the bandwidth of the input filter (in this example, no filtering is required

Device, so keep IC1F = 0000). The capture prescaler is not used in the trigger operation, so no configuration is required. CC1S

The bit selects only the input capture source, that is, CC1S = 01 in the TIMx\_CCMR1 register. Set the TIMx\_CCER register

CC1P = 0 to determine the polarity (only the rising edge is detected)

• Set SMS = 100 in the TIMx\_SMCR register to configure the timer in reset mode; set the TIMx\_SMCR register

TS = 101, select TI1 as input source

• Set CEN = 1 in the TIMx\_CR1 register to start the counter

The counter starts counting according to the internal clock, and then runs normally until TI1 has a rising edge; at this time, the counter is

Clear to zero and restart counting from 0. At the same time, the trigger flag (TIF bit in the TIMx\_SR register) is set, the root

According to the settings of the TIE (Interrupt Enable) bit and TDE (DMA Enable) bit in the TIMx\_DIER register, please generate an interrupt.

Request or a DMA request.

The figure below shows the behavior when the auto-reload register TIMx\_ARR = 0x36. At the rising edge of TI1 and the actual counter

The delay between resets depends on the resynchronization circuit at the input of TI1.

729416

33

32

31

30

00

01 02 03

36

00

01

02

34 35

03

TI1

UG

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

114.

The enable of the counter depends on the level of the selected input.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

203/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 226** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

In the following example, the counter only counts up when TI1 is low:

• Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (in this example, no filtering is required, so keep

Hold IC1F = 0000). The capture prescaler is not used in the trigger operation, so no configuration is required. CC1S bit is used for selection

Enter the capture source and set CC1S = 01 in the TIMx\_CCMR1 register. Set CC1P in TIMx\_CCER register

= 1 to determine the polarity (only low level detection).

• Set SMS = 101 in the TIMx\_SMCR register to configure the timer in gating mode; set the TIMx\_SMCR register

TS = 101, select TI1 as the input source.

• Set CEN = 1 in the TIMx\_CR1 register to start the counter. In gated mode, if CEN = 0, count

The device cannot be started regardless of the trigger input level.

As long as TI1 is low, the counter starts counting based on the internal clock and stops counting when TI1 goes high. When the counter starts or

When stopping, set the TIF flag in TIMx\_SR.

The delay between the rising edge of TI1 and the actual stop of the counter depends on the resynchronization circuit at the input of TI1.

704741

**34**

**33**

**32**

**31**

**30**

**35**

**36 37 38**

Write TIF = 0

TI1

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

115.

The enabling of the counter depends on the event on the selected input.

In the following example, the counter starts counting up on the rising edge of the TI2 input:

• Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (in this example, no filter is required, ensure that

Hold IC2F = 0000). The capture prescaler is not used in the trigger operation, and no configuration is required. CC2S bit is for selection only

Enter the capture source and set CC2S = 01 in the TIMx\_CCMR1 register. Set CC1P in TIMx\_CCER register

= 1 to determine the polarity (only low level detection).

• Set SMS = 110 in the TIMx\_SMCR register to configure the timer in trigger mode; set the TIMx\_SMCR register

TS = 110, select TI2 as the input source.

When TI2 has a rising edge, the counter starts to count under the drive of the internal clock and sets the TIF flag.

The delay between the rising edge of TI2 and the start of the counter depends on the resynchronization circuit at the input of TI2.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

204/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 227** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

991811

34

35

36

37

38

TI2

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

116.

**2 +**

External clock mode 2 can be used with another slave mode (except external clock mode 1 and encoder mode). This

, The ETR signal is used as the input of the external clock, and another one can be selected in reset mode, gating mode or trigger mode

Inputs are used as trigger inputs. It is not recommended to use the TS bit of the TIMx\_SMCR register to select ETR as TRGI.

In the following example, when a rising edge occurs on TI1, the counter counts up on each rising edge of ETR

once:

• Configure the external trigger input circuit through the TIMx\_SMCR register:

**–** ETF = 0000: no filtering

**–** ETPS = 00: without prescaler

**–** ETP = 0: detect the rising edge of ETR, set ECE = 1 to enable external clock mode 2

• Configure channel 1 as follows to detect the rising edge of TI:

**–** IC1F = 0000: no filtering

**– The** capture prescaler is not used in the trigger operation and no configuration is required

**–** Set CC1S = 01 in the TIMx\_CCMR1 register to select the input capture source

**–** Set CC1P = 0 in the TIMx\_CCER register to determine the polarity (only the rising edge is detected)

• Set SMS = 110 in the TIMx\_SMCR register to configure the timer in trigger mode. Set TIMx\_SMCR register

TS = 101, select TI1 as the input source.

When a rising edge occurs on TI1, the TIF flag is set and the counter starts counting on the rising edge of ETR.

The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronization circuit at the input of ETRP.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

205/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 228** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

262859

34

35

36

TI1

CEN/CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

ETR

117.

2 +

**11.3.15**

All TIMx timers are connected internally for timer synchronization or linking. When a timer is in main mode, it

You can reset, start, stop, or provide a clock to the counter of another timer in slave mode.

The figure below shows an overview of the trigger selection and main mode selection modules.

use

115977

Main mode

control

clock

MMS

UEV

counter

Prescaler

Slave mode

control

SMS

counter

Prescaler

CK \_ PSC

ITR 0

TS

TRGO 1

TIMER 1

TIMER 2

Input trigger selection

118.

/

For example, Timer 1 can be configured as Timer 2 prescaler. Refer to the figure above and do the following:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

206/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 229** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

• Configure Timer 1 as the main mode, which can output a periodic trigger signal at each update event UEV.

When MMS = '010' in the TIM1\_CR2 register, input on TRGO1 whenever an update event occurs

There is a rising edge signal.

• Connect TRGO1 of timer 1 to timer 2 and set TS = '000' in TIM2\_SMCR register,

Configure Timer 2 to use ITR1 as the internally triggered slave mode.

• Then put the slave mode controller in external clock mode 1 (SMS = 111 in TIM2\_SMCR register);

Timer 2 can be driven by the periodic rising edge of Timer 1 (that is, the timer 1 counter overflows).

• Finally, the corresponding (TIMx\_CR1 register) CEN bit must be set to start the two timers respectively.

Note: If OCx has been selected as the trigger output of Timer 1 (MMS = 1xx), its rising edge is used to drive the count of Timer 2

Device.

use

In this example, the operation of Timer 2 is controlled by the output comparison of Timer 1. Refer to the figure below. Only when timer 1

When OC1REF is high, Timer 2 counts the divided internal clock. The clock frequency of both timers is determined by

The prescaler divides CK\_INT by 3 (f CK\_CNT = f CK\_INT/3 ).

• Configure Timer 1 as the main mode and send its output comparison reference signal (OC1REF) as the trigger output (TIM1\_CR2

Register MMS = 100)

• Configure the OC1REF waveform of timer 1 (TIM1\_CCMR1 register)

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 001 in TIM2\_SMCR register)

• Configure Timer 2 for gating mode (SMS = 101 in TIM2\_SMCR register)

• Set CEN = 1 in the TIM2\_CR1 register to enable timer 2

• Set CEN = 1 in the TIM1\_CR1 register to enable timer 1

Note: The timer 2 clock is not synchronized with the timer 1 clock, this mode only affects the timer 2 counter enable signal.

949443

FE

00

01

FC

3046

3048

3045

3047

FD

FF

CK\_INT

TIMER1-OC1REF

TIMER1-CNT

TIMER2-CNT

TIMER2-TIF

Write TIF = 0

119.

1

OC1REF

2

In the example above, their counters and prescalers were not initialized before Timer 2 started, so they

Start counting from the current value. It is possible to reset 2 timers before starting timer 1, so that they are from the given value

Start, that is, write any desired value in the timer counter. Reset by writing the UG bit in the TIMx\_EGR register

Timer.

In the next example, Timer 1 and Timer 2 need to be synchronized. Timer 1 is the main mode and starts from 0, the timer

2 is the slave mode and starts from 0xE7; the prescaler coefficients of the two timers are the same. Write 0 to CEN of TIM1\_CR1

The bit will disable Timer 1, and Timer 2 will stop immediately.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

207/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 230** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

• Configure Timer 1 as the main mode and send the output compare 1 reference signal (OC1REF) as the trigger output (TIM1\_CR2

Register MMS = 100).

• Configure the OC1REF waveform of timer 1 (TIM1\_CCMR1 register).

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 for gating mode (SMS = 101 in TIM2\_SMCR register)

• Set UG = 1 in the TIM1\_EGR register to reset timer 1.

• Set UG = 1 in the TIM2\_EGR register to reset timer 2.

• Write 0xE7 to the timer 2 counter (TIM2\_CNT) and initialize it to 0xE7.

• Set CEN = 1 in the TIM2\_CR1 register to enable timer 2.

• Set CEN = 1 in the TIM1\_CR1 register to start Timer 1.

• Set CEN = 0 in the TIM1\_CR1 register to stop timer 1.

381843

00

01

02

75

00

E8

E9

AB

E7

CK\_INT

TIMER1-CEN = CNT\_EN

TIMER1-CNT\_INIT

TIMER1-CNT

TIMER2-CNT

TIMER2-CNT\_INIT

TIMER2 write CNT

TIMER2-TIF

Write TIF = 0

120.

1

2

use

In this example, the update event of timer 1 is used to enable timer 2. Refer to the figure below. When Timer 1 generates more

When a new event occurs, Timer 2 starts counting from its current value (which may be non-zero) according to the divided internal clock. in

When the trigger signal is received, the CEN bit of timer 2 is automatically set to 1, and the counter starts counting until 0 is written to

The CEN bit of the TIM2\_CR1 register. The clock frequency of both timers is divided by the prescaler to CK\_INT

3(fCK\_CNT=fCK\_INT/3).

• Configure Timer 1 as the main mode and send its update event (UEV) as the trigger output (TIM1\_CR2 register)

MMS = 010).

• Configure the period of Timer 1 (TIM1\_ARR register).

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 as trigger mode (SMS = 110 in TIM2\_SMCR register)

• Set CEN = 1 in the TIM1\_CR1 register to start Timer 1.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

208/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 231** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

227169

FF

01

02

FD

46

48

45

47

Write TIF = 0

FE

00

CK\_INT

TIMER1\_UEV

TIMER1\_CNT

TIMER2\_CNT

TIMER2\_CEN = CNT\_EN

TIMER2\_TIF

121.

Use 1

2

In the previous example, the two counters can be initialized before counting is started. The following figure shows the same configuration as 0

Next, use the trigger mode instead of the gating mode (SMS = 110 in the TIM2\_SMCR register).

449102

00

01

02

75

00

E8

E9

CD

EA

E7

CK\_INT

TIMER1-CEN = CNT\_EN

TIMER1-CNT\_INIT

TIMER1-CNT

TIMER2-CNT

TIMER2-CNT\_INIT

TIMER2-CNT\_INIT

TIMER2-TIF

Write TIF = 0

122.

Use 1

2

use

This example uses Timer 1 as the prescaler for Timer 2. The configuration is as follows:

• Configure Timer 1 as the main mode and send its update event UEV as the trigger output (TIM1\_CR2 register

MMS = '010'). Then every time the counter overflows, it outputs a periodic signal.

• Configure the period of Timer 1 (TIM1\_ARR register).

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 to use external clock mode (SMS = 111 in TIM2\_SMCR register)

• Set CEN = 1 in the TIM1\_CR2 register to start timer 2

• Set CEN = 1 in the TIM1\_CR1 register to start timer 1

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

209/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 232** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

Use **2**

In this example, when the TI1 input of timer 1 rises, timer 1 is enabled and timer 1 is enabled, and timer 2 is enabled. for

To ensure the alignment of the counter, Timer 1 must be configured in master/slave mode (corresponding to TI1 being a slave, and corresponding to Timer 2 being a master):

• Configure Timer 1 as the main mode and send its enable as a trigger output (MMS='001' in the TIM1\_CR2 register

)

• Configure Timer 1 in Slave mode and obtain input trigger from TI1 (TS = '100' in TIM1\_SMCR register)

• Configure Timer 1 as the trigger mode (SMS='110' in the TIM1\_SMCR register)

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 as trigger mode (SMS = 110 in TIM2\_SMCR register)

When a rising edge appears on TI1 of timer 1, the two timers start counting according to the internal clock synchronously.

The TIF flag is also set at the same time.

Note: In this example, both timers are initialized (set the corresponding UG bit) before starting, and both counters start from 0

At the beginning, you can insert an offset between timers by writing to any counter register (TIMx\_CNT). You can see in the picture below

There is a delay between CNT\_EN and CK\_PSC of Timer 1 in master/slave mode.

753049

00

01

07

02 03 04 05 06

08 09

00

01

07

02 03 04 05 06

08 09

CK\_INT

TIMER 1-TI1

TIMER 1-CEN = CNT\_EN

TIMER 1-CK\_PSC

TIMER1-CNT

TIMER1-TIF

TIMER2-CEN = CNT\_EN

TIMER2-CK\_PSC

TIMER2-CNT

TIMER2-TIF

123.

Use 1

TI1

1

2

**11.3.16**

When the microcontroller enters debug mode (CPU core is stopped), according to the setting of DBG\_TIMx\_STOP in the DBG module,

The TIMx counter either continues to operate normally or stops. See the debugging module chapter for details.

**11.4 TIMx**

These peripheral registers can be operated in halfword (16-bit) or word (32-bit) mode.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

210/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 233** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

40. TIMx

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

TIMx\_CR1

Control register 1

0x00000000

[Section 12.4.1](https://translate.googleusercontent.com/translate_f#291)

0x04

TIMx\_CR2

Control register 2

0x00000000

[Section 12.4.2](https://translate.googleusercontent.com/translate_f#293)

0x08

TIMx\_SMCR

Slave mode control register

0x00000000

[Section 12.4.3](https://translate.googleusercontent.com/translate_f#294)

0x0C

TIMx\_DIER

DMA/interrupt enable register

0x00000000

[Section 12.4.4](https://translate.googleusercontent.com/translate_f#297)

0x10

TIMx\_SR

Status register

0x00000000

[Section 12.4.5](https://translate.googleusercontent.com/translate_f#299)

0x14

TIMx\_EGR

Event generation register

0x00000000

[Section 12.4.6](https://translate.googleusercontent.com/translate_f#300)

0x18

TIMx\_CCMR1

Capture/Compare Mode Register 1

0x00000000

[Section 12.4.7](https://translate.googleusercontent.com/translate_f#301)

0x1C

TIMx\_CCMR2

Capture/Compare Mode Register 2

0x00000000

[Section 12.4.8](https://translate.googleusercontent.com/translate_f#305)

0x20

TIMx\_CCER

Capture/compare enable register

0x00000000

[Section 12.4.9](https://translate.googleusercontent.com/translate_f#307)

0x24

TIMx\_CNT

counter

0x00000000

[Section 12.4.10](https://translate.googleusercontent.com/translate_f#308)

0x28

TIMx\_PSC

Prescaler

0x00000000

[Section 12.4.11](https://translate.googleusercontent.com/translate_f#309)

0x2C

TIMx\_ARR

Autoload register

0x00000000

[Section 12.4.12](https://translate.googleusercontent.com/translate_f#309)

0x34

TIMx\_CCR1

Capture/Compare Register 1

0x00000000

[Section 12.4.13](https://translate.googleusercontent.com/translate_f#310)

0x38

TIMx\_CCR2

Capture/Compare Register 2

0x00000000

[Section 12.4.14](https://translate.googleusercontent.com/translate_f#310)

0x3C

TIMx\_CCR3

Capture/Compare Register 3

0x00000000

[Section 12.4.15](https://translate.googleusercontent.com/translate_f#311)

0x40

TIMx\_CCR4

Capture/Compare Register 4

0x00000000

[Section 12.4.16](https://translate.googleusercontent.com/translate_f#311)

0x48

TIMx\_DCR

DMA control register

0x00000000

[Section 12.4.17](https://translate.googleusercontent.com/translate_f#312)

0x4C

TIMx\_DMAR

Continuous mode DMA address

0x00000000

[Section 12.4.18](https://translate.googleusercontent.com/translate_f#313)

**11.4.1**

**1(TIMx\_CR1)**

Offset address: 0x00

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CKD

ARPE

CMS

DIR

OPM

URS UDIS CEN

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

9:8

CKD

rw

0x00

Clock division

These 2 bits are defined in the timer clock (CK\_INT) frequency, dead time and by

Between the dead zone generator and the sampling clock used by the digital filter (ETR, TIx)

The frequency division ratio.

00: t DTS = t CK\_INT

01: t DTS = 2 xt CK\_INT

10: t DTS = 4 xt CK\_INT

11: Reserved, do not use this configuration

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

211/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 234** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7

ARPE

rw

0x00

Auto-reload preload enable bit (Auto-reload preload enable)

0: TIMx\_ARR register is not buffered

1: TIMx\_ARR register is loaded into the buffer

6:5

CMS

rw

0x00

Select Center-aligned mode selection

00: Edge alignment mode. Counter up or down according to the direction bit (DIR)

count

01: Center alignment mode 1. The counter alternately counts up and down. Configuration

Is the output of the output channel (CCxS = 00 in the TIMx\_CCMRx register)

The compare interrupt flag bit is only set when the counter is counting down

10: Center alignment mode 2. The counter alternately counts up and down. counter

Count up and down alternately. Channels configured as outputs (TIMx\_CCMRx

The CCxS = 00) output compare interrupt flag bit in the register, only in the counter

Set when counting up

11: Center alignment mode 3. The counter alternately counts up and down. counter

Count up and down alternately. Channels configured as outputs (TIMx\_CCMRx

The CCxS = 00) output compare interrupt flag bit in the register

Set when counting up and down

Note: When the counter is turned on (CEN = 1), it is not allowed to switch from edge-aligned mode

Switch to center alignment mode.

4

DIR

rw

0x00

Direction

0: the counter counts up

1: The counter counts down

Note: When the counter is configured for center-aligned mode or encoder mode, this bit is

Read only.

3

OPM

rw

0x00

One pulse mode (One pulse mode) 0: When an update event occurs, the counter

Counter does not stop

1: When the next update event occurs (clears the CEN bit), the counter stops

2

URS

rw

0x00

Update request source (Update request source) software selects through this bit

The source of the UEV event.

0: If update interrupt or DMA request is allowed, any one of the following events

An update interrupt or DMA request is generated:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller

1: If an update interrupt or DMA request is allowed, only the counter overflows

An update interrupt or DMA request is generated only after the overflow/underflow

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

212/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 235** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

UDIS

rw

0x00

Disable disable (Update disable) Software enables/disables UEV through this bit

Event generation

0: UEV is allowed. Update (UEV) events are generated by any of the following events:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller are cached registers are loaded into their

Preload value.

1: UEV is prohibited. No update event occurs, shadow registers (ARR, PSC,

CCRx) keep their values. If the UG bit or slave mode controller is set

A hardware reset is issued and the counter and prescaler are reinitialized

0

CEN

rw

0x00

Counter enable

0: disable counter

1: Enable the counter.

Note: After the CEN bit is set by the software, the external clock, gating mode and encoder mode

To work. The trigger mode can automatically set the CEN bit through hardware.

**11.4.2**

**2(TIMx\_CR2)**

Offset address: 0x04

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

Reserved

TI1S

MMS

CCDS

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 8

Reserved

Reserved, read as 0

7

TI1S

rw

0x00

TI1 selection (TI1 selection)

0: TIMx\_CH1 pin is connected to TI1 input

1: TIMx\_CH1, TIMx\_CH2 and TIMx\_CH3 pins are connected through XOR

To TI1 input

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

213/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 236** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6:4

MMS

rw

0x00

Master mode selection

These two bits are used to select the synchronization information sent to the slave timer in master mode

(TRGO). The possible combinations are as follows:

000: Reset – the UG bit in the TIMx\_EGR register is used as a trigger input

Out (TRGO). If trigger input (slave mode controller is in reset mode)

Reset occurs, the signal on TRGO will have a delay relative to the actual reset

late.

001: Enable – The counter enable signal CNT\_EN is used as a trigger input

Out (TRGO). Sometimes it is necessary to start multiple timers or controls at the same time

Enable the slave timer for a period of time. Counter enable signal is through CEN

Logic OR generation of the trigger input signal in control bit and gating mode. Count

When the counter enable signal is controlled by the trigger input, there will be a delay on TRGO,

Unless master/slave mode is selected (see the MSM bit in the TIMx\_SMCR register

description of).

010: Update-The update event is selected as the trigger input (TRGO). For example, one

The clock of each master timer can be used as a prescaler for the slave timer.

011: Comparison pulse-when a capture occurs or a comparison is successful, when setting

When the CC1IF flag is set (even if it is already high), the trigger output sends a positive

Pulse (TRGO).

100: Compare – OC1REF signal is used as a trigger output (TRGO)

101: Compare – OC2REF signal is used as a trigger output (TRGO)

110: Compare – OC3REF signal is used as a trigger output (TRGO)

111: Compare – OC4REF signal is used as a trigger output (TRGO)

3

CCDS

rw

0x00

Capture/compare DMA selection

0: Send CCx DMA request when CCx event occurs

1: When an update event occurs, the CCx DMA request is sent

2: 0

Reserved

Reserved, read as 0

**11.4.3**

**(TIMx\_SMCR)**

Offset address: 0x08

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

ETP

ECE

ETPS

ETF

MSM

TS

Res.

SMS

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

ETP

rw

0x00

External trigger polarity

This bit selects whether to use ETR or the inverse of ETR as the trigger operation.

0: ETR is not inverted, high level or rising edge is valid

1: ETR is inverted, low level or falling edge is valid

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

214/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 237** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

14

ECE

rw

0x00

External clock enable (External clock enable)

This bit enables external clock mode 2.

0: Disable external clock mode 2

1: Enable external clock mode 2, the counter is valid by any on the ETRF signal

Rising edge drive

Note 1: Set ECE bit and select external clock mode 1 and connect TRGI to

ETRF (SMS = 111 and TS = 111) have the same effect.

Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gate

Control mode and trigger mode; however, TRGI cannot be connected to ETRF (TS bit does not

Can be 111).

Note 3: When external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock

The input of the clock is ETRF.

13:12

ETPS

rw

0x00

External trigger prescaler

The frequency of the external trigger signal ETRP must be at most the frequency of TIMxCLK

1/4. When inputting a faster external clock, you can use prescaler to reduce ETRP

Frequency of.

00: turn off prescaler

01: ETRP frequency divided by 2

10: ETRP frequency divided by 4

11: ETRP frequency divided by 8

11:8

ETF

rw

0x00

External trigger filter

These bits define the sampling frequency of ETRP signal and the digital filtering of ETRP

The bandwidth of the wave. In fact, the digital filter is an event counter, which records

After N events, an output transition will occur.

0000: no filter, sampling with f DTS

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

0110: Sampling frequency f SAMPLING = f DTS /4, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

215/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 238** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7

MSM

rw

0x00

Master/slave mode

0: No effect

1: The event on the trigger input (TRGI) is delayed to allow the current setting

The perfect synchronization between the timer (via TRGO) and its slave timer, this is necessary

It is very useful when you want to synchronize several timers to a single external event

of

6:4

TS

rw

0x00

Trigger selection

These 3 bits select the trigger input for synchronization counter.

000: Internal trigger 0 (ITR0)

001: Internal trigger 1 (ITR1)

010: Internal trigger 2 (ITR2)

011: Internal trigger 3 (ITR3)

100: TI1's edge detector (TI1F\_ED)

101: Filtered timer input 1 (TI1FP1)

110: Filtered timer input 2 (TI2FP2)

111: External trigger input (ETRF)

For more details about ITRx, see the table below.

Note: These bits can only be changed when not used (eg SMS = 000), to avoid changing

Wrong edge detection when changing time.

3

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

216/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 239** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2: 0

SMS

rw

0x00

Slave mode selection

When the external signal is selected, the effective edge of the trigger signal (TRGI) and the selected

External input polarity is related (see input control register and control register

Bright)

000: Slave mode off-if CEN = 1, the prescaler is directly internally

Clock driven.

001: Encoder mode 1-according to the level of TI1FP1, the counter is in TI2FP2

Counts up/down.

010: Encoder mode 2-according to the level of TI2FP2, the counter is in TI1FP1

Counts up/down.

011: Encoder mode 3-according to the level of another input, the counter is

The edges of TI1FP1 and TI2FP2 count up/down.

100: Reset mode-the rising edge of the selected trigger input (TRGI) is reset

Initialize the counter and generate a signal to update the register.

101: Gating mode-when the trigger input (TRGI) is high, the time of the counter

The bell opens. When the trigger input goes low, the counter stops (but does not reset). meter

The start and stop of the counter are controlled.

110: Trigger mode-the counter starts on the rising edge of the trigger input TRGI (but

No reset), only the start of the counter is controlled.

111: External clock mode 1-rising edge of the selected trigger input (TRGI)

Drive counter.

Note: If TI1F\_EN is selected as the trigger input (TS = 100), do not use the gate

Control mode. This is because TI1F\_ED outputs a pulse every time TI1F changes,

However, the gating mode is to check the level of the trigger input.

41. TIMx

Slave timer

**ITR0 (TS = 000)**

**ITR1 (TS = 001)**

**ITR2 (TS = 010)**

**ITR3 (TS = 011)**

TIM3

TIM1

TIM2

no

no

**11.4.4 DMA/**

**(TIMx\_DIER)**

Offset address: 0x0C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Res. TDE Res. CC4DE

UDE

Res.

TIE

Res. CC4IE

UIE

CC3DE CC2DECC1DE

CC3IE CC2IE CC1IE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

217/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 240** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

14

TDE

rw

0x00

Allow trigger DMA request (Trigger DMA request enable)

0: Disable triggering of DMA request

1: Allow to trigger DMA request

13

Reserved

Reserved, read as 0

12

CC4DE

rw

0x00

Allow Capture/Compare 4 DMA request (Capture/Compare 4 DMA

request enable)

0: Disable capture/compare 4 DMA request

1: Allow capture/compare 4 DMA request

11

CC3DE

rw

0x00

Allow Capture/Compare 3 DMA request (Capture/Compare 3 DMA

request enable)

0: Disable capture/compare 3 DMA request

1: Allow capture/compare 3 DMA request

10

CC2DE

rw

0x00

Allow Capture/Compare 2 DMA request (Capture/Compare 2 DMA

request enable)

0: Disable capture/compare 2 DMA request

1: Allow capture/compare 2 DMA request

9

CC1DE

rw

0x00

Allow Capture/Compare 1 DMA request (Capture/Compare 1 DMA

request enable)

0: Disable capture/compare 1 DMA request

1: Allow capture/compare 1 DMA request

8

UDE

rw

0x00

Update DMA request enable

0: Disable update DMA request

1: Allow updated DMA requests

7

Reserved

Reserved, read as 0

6

TIE

rw

0x00

Trigger interrupt enable

0: Disable trigger interrupt

1: Enable trigger interrupt

5

Reserved

Reserved, read as 0

4

CC4IE

rw

0x00

Allow Capture/Compare 4 interrupt en-

able)

0: Disable capture/compare 4 interrupt

1: Enable capture/compare 4 interrupt

3

CC3IE

rw

0x00

Allow Capture/Compare 3 interrupt en-

able)

0: Disable capture/compare 3 interrupt

1: Allow capture/compare 3 interrupt

2

CC2IE

rw

0x00

Allow Capture/Compare 2 interrupt en-

able)

0: Disable capture/compare 2 interrupt

1: Enable capture/compare 2 interrupt

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

218/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 241** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

CC1IE

rw

0x00

Allow Capture/Compare 1 interrupt en-

able)

0: Disable capture/compare 1 interrupt

1: Enable capture/compare 1 interrupt

0

UIE

rw

0x00

Update interrupt enable

0: Disable update interrupt

1: Allow update interrupt

**11.4.5**

**(TIMx\_SR)**

Offset address: 0x10

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rc\_w0

rc\_w0

rc\_w0

rc\_w0

Res.

CC4OF

Res.

TIF

Res. CC4IF

UIF

CC3OF CC2OF CC1OF

CC3IF CC2IF CC1IF

rc\_w0 rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, read as 0

12

CC4OF

rc\_w0

0x00

Capture/Compare 4 overcap-capture (Capture/Compare 4 overcap-

ture flag)

See CC1OF description.

11

CC3OF

rc\_w0

0x00

Capture/Compare 3 overcap-capture (Capture/Compare 3 overcap-

ture flag)

See CC1OF description.

10

CC2OF

rc\_w0

0x00

Capture/Compare 2 overcap-capture (Capture/Compare 2 overcap-

ture flag)

See CC1OF description.

9

CC1OF

rc\_w0

0x00

Capture/Compare 1 overcap-capture (Capture/Compare 1 overcap-

ture flag)

This flag can be set by hardware only when the corresponding channel is configured for input capture

1. Write 0 to clear this bit.

0: No repeated capture;

1: When the value of the counter is captured in the TIMx\_CCR1 register, the CC1IF

The status is already 1.

8: 7

Reserved

Reserved, read as 0

6

TIF

rc\_w0

0x00

Trigger interrupt flag

When a trigger event occurs (when the slave mode controller is in a mode other than gating mode)

In other modes, a valid edge is detected at the TRGI input, or gated mode

Any edge under the formula) is set to 1 by hardware. It is cleared by software.

0: No trigger event is generated

1: Trigger interrupt waiting for response

5

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

219/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 242** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

4

CC4IF

rc\_w0

0x00

Capture/Compare 4 interrupt flag

Refer to CC1IF description.

3

CC3IF

rc\_w0

0x00

Capture/Compare 3 interrupt flag

Refer to CC1IF description.

2

CC2IF

rc\_w0

0x00

Capture/Compare 2 interrupt flag

Refer to CC1IF description.

1

CC1IF

rc\_w0

0x00

Capture/Compare 1 interrupt flag

If channel CC1 is configured in output mode:

When the counter value matches the comparison value, the bit is set to '1' by the hardware, but in the center

Except in symmetric mode (refer to the CMS bit of the TIMx\_CR1 register). it

Cleared by software to '0'.

0: No match occurred

1: The value of TIMx\_CNT matches the value of TIMx\_CCR1

If channel CC1 is configured for input mode:

When the capture event occurs, this bit is set by hardware, it is cleared by software or through

Clear '0' after reading TIMx\_CCR1.

0: No input capture

1: The counter value has been captured (copied) to TIMx\_CCR1 (checked on IC1

An edge with the same polarity as the selected one is measured)

0

UIF

rc\_w0

0x00

Update interrupt flag

This bit is set to '1' by hardware when an update event occurs. It is cleared to '0' by the software.

0: No update event is generated

1: Update event is waiting for response. When the register is updated, this bit is set to '1' by hardware:

-If UDIS = 0 in the TIMx\_CR1 register, when REP\_CNT = 0

An update event is generated (when the down counter overflows or underflows repeatedly)

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, when TIMx\_EGR

An update event occurs when UG = 1 in the register (software resets counter CNT

New initialization)

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, the counter

An update event is generated when CNT is triggered by event re-initialization. (Refer to synchronous control

(Description of register)

**11.4.6**

**(TIMx\_EGR)**

Offset address: 0x14

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

w

w

w

w

TG

Res. CC4G CC3G CC2G CC1G

UG

w

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:7

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

220/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 243** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6

TG

w

0x00

Trigger generation

This bit is set to '1' by the software and is used to generate a braking event, automatically by the hardware

Clear '0'.

0: No action

1: TIF = 1 in the TIMx\_SR register, if the corresponding interrupt and DMA are enabled,

Then generate the corresponding interrupt and DMA

5

Reserved

Reserved, read as 0

4

CC4G

w

0x00

Generate Capture/Compare 4 generation

Refer to CC1G description.

3

CC3G

w

0x00

Generate Capture/Compare 3 generation

Refer to CC1G description.

2

CC2G

w

0x00

Generate Capture/Compare 2 generation

Refer to CC1G description.

1

CC1G

w

0x00

Generate Capture/Compare 1 generation

This bit is set by software to generate a capture/compare event.

Reset to 0.

0: No action

1: Generate a capture/compare event on channel CC1:

If channel CC1 is configured as an output:

Set CC1IF=1, if the corresponding interrupt and DMA are turned on, the corresponding

Interrupt and DMA.

If channel CC1 is configured as input:

The current counter value is captured into the TIMx\_CCR1 register and set

CC1IF = 1, if the corresponding interrupt and DMA are turned on, the corresponding

Off and DMA. If CC1IF is already 1, set CC1OF = 1.

0

UG

w

0x00

Generate an update event (Update generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: Re-initialize the counter and generate an update event. Note the prescaler

The counter of the counter is also cleared to '0' (but the prescaler coefficient is unchanged). If in the center

In symmetric mode or DIR = 0 (count up), the counter is cleared to '0'; if

DIR = 1 (count down), the counter takes the value of TIMx\_ARR.

**11.4.7**

**/**

**1(TIMx\_CCMR1)**

Offset address: 0x18

Reset value: 0x0000

The channel can be used for input (capture mode) or output (comparison mode), and the direction of the channel is defined by the corresponding CCxS. The deposit

The function of other bits of the device is different from the output mode. OCxx describes the function of the channel in output mode, ICxx describes

The function of the channel in output mode. Therefore, it must be noted that the function of the same bit in output mode and input mode is not

Same.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

221/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 244** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OC2M

OC2PE OC2FE

CC2S

OC1CE

OC1M

IC1F

OC1PE OC1FE

IC1PSC

CC1S

IC2PSC

IC2F

OC2CE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

OC2CE

rw

0x00

Output compare 2 clear enable

14:12

OC2M

rw

0x00

Output compare 2 mode

11

OC2PE

rw

0x00

Output compare 2 preload en-

able)

10

OC2FE

rw

0x00

Output compare 4 fast enable (Output compare 4 fast enable)

9:8

CC2S

rw

0x00

Capture/Compare 2 selection

This bit defines the direction of the channel (input/output) and the selection of input pins:

00: CC2 channel is configured as output

01: CC2 channel is configured as input, IC2 is mapped on TI2

10: CC2 channel is configured as input, IC2 is mapped on TI1

11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC2S is only when the channel is closed (CC2E = 0 in the TIMx\_CCER register)

Is writable.

7

OC1CE

rw

0x00

Output compare 1 clear enable

0: OC1REF is not affected by ETRF input

1: When high level of ETRF input is detected, clear OC1REF = 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

222/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 245** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6:4

OC1M

rw

0x00

Output compare 1 mode

These 3 bits define the action of the output reference signal OC1REF, and OC1REF

Decide the values ​​​​of OC1 and OC1N. OC1REF is active high, and

The effective levels of OC1 and OC1N depend on the CC1P and CC1NP bits.

000: freeze. Output compare register TIMx\_CCR1 and counter

Comparison between TIMx\_CNT has no effect on OC1REF

001: Set channel 1 to effective level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is high

010: Set channel 1 to invalid level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is low

011: Flip. When TIMx\_CCR1=TIMx\_CNT, flip OC1REF

Level

100: Forced to invalid level. Force OC1REF low

101: Forced to effective level. Force OC1REF high

110: PWM mode 1-When counting up, when TIMx\_CNT <

When TIMx\_CCR1, channel 1 is active level, otherwise it is inactive level;

When counting down, when TIMx\_CNT> TIMx\_CCR1, channel 1 is none

Effective level (OC1REF = 0), otherwise the effective level (OC1REF = 1)

111: PWM mode 2-When counting up, when TIMx\_CNT <

When TIMx\_CCR1, channel 1 is invalid level, otherwise it is valid level;

When counting down, when TIMx\_CNT> TIMx\_CCR1, channel 1 is

Effective level, otherwise invalid level

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = 00 (the channel is configured as an output), this bit cannot be modified. Note

2: In PWM mode 1 or PWM mode 2, only when the comparison result changes

Or when switching from freeze mode to PWM mode in output compare mode, OC1REF

The level changes.

3

OC1PE

rw

0x00

Output compare 1 preload en-

able)

0: Disable the preload function of the TIMx\_CCR1 register, which can be written at any time

TIMx\_CCR1 register, and the newly written value takes effect immediately

1: Turn on the preload function of the TIMx\_CCR1 register.

Preload register operation, the preload value of TIMx\_CCR1 is updated to

Is loaded into the current register when it comes

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = 00 (the channel is configured as an output), this bit cannot be modified. Note 2:

In single pulse mode only (OPM = 1 in the TIMx\_CR1 register), you can

The PWM mode is used when the preload register is recognized, otherwise its operation is undefined.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

223/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 246** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

OC1FE

rw

0x00

Output compare 1 fast enable (Output compare 1 fast enable)

This bit is used to speed up the response of the CC output to trigger input events.

0: CC1 operates normally according to the value of the counter and CCR1, even if it is triggered

The device is turned on. When the trigger input has a valid edge, activate CC1

The minimum output delay is 5 clock cycles

1: The valid edge input to the trigger acts as if a comparison match has occurred.

Therefore, OC is set to the comparison level regardless of the comparison result. Sample trigger

The delay between the effective edge of the device and the output of CC1 is reduced to 3 clock cycles.

OCFE only works when the channel is configured in PWM1 or PWM2 mode

1:0

CC1S

rw

0x00

Capture/Compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: CC1 channel is configured as input, IC1 is mapped on TI2

11: CC1 channel is configured as input, IC1 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIMx\_CCER register)

Is writable.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:12

IC2F

rw

0x00

Input capture 2 filter

11:10

IC2PSC

rw

0x00

Input capture 2 prescaler

9:8

CC2S

rw

0x00

Capture/Compare 2 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC2 channel is configured as output

01: CC2 channel is configured as input, IC2 is mapped on TI2

10: CC2 channel is configured as input, IC2 is mapped on TI1

11: CC2 channel is configured as input, IC2 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC2S is only when the channel is closed (CC2E = 0 in the TIMx\_CCER register)

Is writable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

224/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 247** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7:4

IC1F

rw

0x00

Input capture 1 filter

These bits define the sampling frequency and digital filter length of the TI1 input. number

The word filter consists of an event counter, which will record after N events

Produces an output transition:

0000: no filter, sampling with fDTS

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

0110: Sampling frequency f SAMPLING = f DTS /4, N = 6

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

3:2

IC1PSC

rw

0x00

Input capture 1 prescaler

These 2 bits define the prescaler coefficient of the CC1 input (IC1).

When CC1E = 0 (in the TIMx\_CCER register), the prescaler is reset.

00: No prescaler, every edge detected on the capture input port is touched

Send a capture

01: Capture is triggered every 2 events

10: Capture is triggered every 4 events

11: Trigger every 8 events

1:0

CC1S

rw

0x00

Capture/compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: CC1 channel is configured as input, IC1 is mapped on TI2

11: CC1 channel is configured as input, IC1 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIMx\_CCER register)

Is writable.

**11.4.8**

**/**

**2(TIMx\_CCMR2)**

Offset address: 0x1C

Reset value: 0x0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

225/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 248** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

See the description of the CCMR1 register above

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OC4M

OC4PE OC4FE

CC4S

OC3CE

OC3M

IC3F

OC3PE OC3FE

IC3PSC

CC3S

IC4PSC

IC4F

OC4CE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

OC4CE

rw

0x00

Output compare 4 clear enable

14:12

OC4M

rw

0x00

Output compare 4 mode (Output compare 4 mode)

11

OC4PE

rw

0x00

Output compare 4 preload en-

able)

10

OC4FE

rw

0x00

Output compare 4 fast enable (Output compare 4 fast enable)

9:8

CC4S

rw

0x00

Capture/Compare 4 selection

The 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC4 channel is configured as output

01: CC4 channel is configured as input, IC4 is mapped on TI4

10: CC4 channel is configured as input, IC4 is mapped on TI3

11: CC4 channel is configured as input, IC4 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC4S is only when the channel is closed (CC4E = 0 in the TIMx\_CCER register)

Is writable.

7

OC3CE

rw

0x00

Output compare 3 clear '0' enable (Output compare 3 clear enable)

6:4

OC3M

rw

0x00

Output compare 3 mode

3

OC3PE

rw

0x00

Output compare 3 preload en-

able)

2

OC3FE

rw

0x00

Output compare 3 fast enable (Output compare 3 fast enable)

1:0

CC3S

rw

0x00

Capture/Compare 3 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC3 channel is configured as output

01: CC3 channel is configured as input, IC3 is mapped on TI3

10: CC3 channel is configured as input, IC3 is mapped on TI4

11: CC3 channel is configured as input, IC3 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC3S is only when the channel is closed (CC3E = 0 in the TIMx\_CCER register)

Is writable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

226/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 249** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:12

IC4F

rw

0x00

Input capture 4 filter

11:10

IC4PSC

rw

0x00

Input capture 4 prescaler

9:8

CC4S

rw

0x00

Capture/Compare 4 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC4 channel is configured as output

01: CC4 channel is configured as input, IC4 is mapped on TI4

10: CC4 channel is configured as input, IC4 is mapped on TI3

11: CC4 channel is configured as input, IC4 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC4S is only when the channel is closed (CC4E = 0 in the TIMx\_CCER register)

Is writable.

7:4

IC3F

rw

0x00

Input capture 3 filter

3:2

IC3PSC

rw

0x00

Input capture 3 prescaler

1:0

CC3S

rw

0x00

Capture/compare 3 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC3 channel is configured as output

01: CC3 channel is configured as input, IC3 is mapped on TI3

10: CC3 channel is configured as input, IC3 is mapped on TI4

11: CC3 channel is configured as input, IC3 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC3S is only when the channel is closed (CC3E = 0 in the TIMx\_CCER register)

Is writable.

**11.4.9**

**/**

**(TIMx\_CCER)**

Offset address: 0x20

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

Res.

CC4P

Res.

Res.

CC4E

CC3P CC3E

CC2P CC2E

CC1P CC1E

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:14

Reserved

Reserved, read as 0

13

CC4P

rw

0x00

Capture/Compare 4 output polarity

Refer to the description of CC1P.

12

CC4E

rw

0x00

Capture/Compare 4 output enable

Refer to the description of CC1E.

11:10

Reserved

Reserved, read as 0

9

CC3P

rw

0x00

Capture/Compare 3 output polarity

Refer to the description of CC1P.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

227/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 250** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

8

CC3E

rw

0x00

Capture/Compare 3 output enable

Refer to the description of CC1E.

7:6

Reserved

Reserved, read as 0

5

CC2P

rw

0x00

Capture/Compare 2 output polarity

Refer to the description of CC1P.

4

CC2E

rw

0x00

Capture/Compare 2 output enable

Refer to the description of CC1E.

3:2

Reserved

Reserved, read as 0

1

CC1P

rw

0x00

Capture/Compare 1 output polarity

CC1 channel is configured as output:

0: OC1 active high

1: OC1 active low

The CC1 channel is configured as an input:

This bit selects whether IC1 or IC1's inverted signal is used as a trigger or capture signal.

0: No inversion: capture occurs on the rising edge of IC1; when used as an external trigger

, IC1 is not inverted

1: Invert: capture occurs on the falling edge of IC1; when used as an external trigger,

IC1 invert

Note: When the LOCK level (LCCK bit in the TIMx\_BDTR register) is set to 3 or

At 2, the bit cannot be modified.

0

CC1E

rw

0x00

Capture/Compare 1 output enable

CC1 channel is configured as output:

0: Off-OC1 disables output, so the output level of OC1 depends on

The values ​​​​of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits

1: On-OC1 signal is output to the corresponding output pin, and its output level depends on

Depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits

Value of

The CC1 channel is configured as an input:

This bit determines whether the value of the counter can be captured into the TIMx\_CCR1 register.

0: capture prohibited

1: Capture enable

42.

Ocx

**CCxE** bit

**OCx** output status

0

Disable output (OCx = 0, OCx\_EN = 0)

1

OCx = OCxREF + polarity, OCx\_EN = 1

Note: The state of external I/O pins connected to standard OCx channels depends on the state of the OCx channel and GPIO and AFIO

register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

228/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 251** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.4.10**

**(TIMx\_CNT)**

Offset address: 0x24

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CNT

rw

0x0000

Counter value

**11.4.11**

**(TIMx\_PSC)**

Offset address: 0x28

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

PSC

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

PSC

rw

0x0000

Prescaler value

The clock frequency of the counter (CK\_CNT) is equal to f CK\_PSC /(PSC + 1).

The PSC includes each time the update event occurs, load the current prescaler

The value of the memory. Update events include the counter being cleared by the UG bit of TIM\_EGR

'0' or the slave controller working in reset mode clears '0'.

**11.4.12**

**(TIMx\_ARR)**

Offset address: 0x2C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

ARR

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

ARR

rw

0x0000

Prescaler value

ARR contains the value to be loaded into the actual auto-reload register.

For details, refer to section [12.3.](https://translate.googleusercontent.com/translate_f#257) 1: Updates and actions related to ARR.

When the value of auto-reload is empty, the counter does not work.

**11.4.13**

**/**

**1(TIMx\_CCR1)**

Offset address: 0x34

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

229/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 252** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR1

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR1

rw

0x0000

Capture/Compare 1 value

If the CC1 channel is configured as an output:

CCR1 contains the value loaded into the current capture/compare 1 register (preload

value).

If the preload is not selected in the TIMx\_CCMR1 register (OC1PE bit)

Load function, the written value will be transferred to the current register immediately. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 1

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC1 port.

If the CC1 channel is configured as an input:

CCR1 contains the count transmitted by the last input capture 1 event (IC1)

器值。 Device value. Device value.

**11.4.14**

**/**

**2(TIMx\_CCR2)**

Offset address: 0x38

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR2

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR2

rw

0x0000

Capture/Compare 2 value

If the CC2 channel is configured as an output:

CCR2 contains the value loaded into the current capture/compare 2 register (preload

value).

If the preload is not selected in the TIMx\_CCMR2 register (OC2PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 2

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC2 port.

If the CC2 channel is configured as an input:

CCR2 contains the count transmitted by the last input capture 2 event (IC2)

器值。 Device value. Device value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

230/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 253** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.4.15**

**/**

**3(TIMx\_CCR3)**

Offset address: 0x3C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR3

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR3

rw

0x0000

Capture/Compare 3 value

If the CC3 channel is configured as an output:

CCR3 contains the value loaded into the current capture/compare 3 register (preload

value).

If the preload is not selected in the TIMx\_CCMR3 register (OC3PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 3

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC3 port. If the CC3 channel is equipped

Set as input:

CCR3 contains the count transmitted by the last input capture 3 event (IC3)

器值。 Device value. Device value.

**11.4.16**

**/**

**4(TIMx\_CCR4)**

Offset address: 0x40

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR4

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR4

rw

0x0000

Capture/Compare 4 value

If the CC4 channel is configured as output:

CCR4 contains the value loaded into the current capture/compare 4 register (preload

value).

If the preload is not selected in the TIMx\_CCMR4 register (OC4PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 4

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC4 port.

If the CC4 channel is configured as input:

CCR4 contains the count transmitted by the last input capture 4 event (IC4)

器值。 Device value. Device value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

231/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 254** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.4.17 DMA**

**(TIMx\_DCR)**

Offset address: 0x48

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

DBL

Res.

Res.

DBA

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, always read as 0.

12:8

DBL

rw

0x00

DMA continuous transfer length (DMA burst length)

These bits define the transfer length of DMA in continuous mode (when the

When the TIMx\_DMAR register is read or written, the timer

Continued transmission), that is: define the number of transmissions, the transmission can be half-word (double-byte)

Or bytes:

00000: 1 transmission 00001: 2 transmissions

00010: 3 transmissions...

...10001: 18 transmissions

Example: We consider such a transmission: DBL = 7, DBA = TIM2\_CR1

-If DBL = 7, DBA = TIM2\_CR1 indicates the address of the data to be transmitted,

Then the transmitted address is given by:

(Address of TIMx\_CR1) + DBA + (DMA index), where DMA index

= DBL

Among them (the address of TIMx\_CR1) + DBA plus 7, gives the write

Enter or read the address of the data, so that the data transfer will occur at the slave address

(Address of TIMx\_CR1) + 7 registers starting with DBA. According to DMA

The setting of the data length may occur as follows:

-If the data is set to halfword (16 bits), the data will be transferred to all 7

Registers.

-If the data is set to bytes, the data will still be transferred to all 7 registers:

The first register contains the first MSB byte, the second register contains the first

One LSB byte, and so on. Therefore, for the timer, the user must indicate

Set the width of data transferred by DMA.

7:5

Reserved

Reserved, always read as 0.

4:0

DBA

rw

0x00

DMA base address (DMA base address) These bits define the DMA in

Base address in continuous mode (when reading the TIMx\_DMAR register or

When writing), DBA is defined as starting from the address where the TIMx\_CR1 register is located

Offset:

00000: TIMx\_CR1

00001: TIMx\_CR2

00010: TIMx\_SMCR

...

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

232/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 255** |

UM\_MM32F003\_q\_Ver1.19

16

Use (TIMX16 BIT)

**11.4.18**

**DMA**

**(TIMx\_DMAR)**

Offset address: 0x4C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

DMAB

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

DMAB

rw

0x0000

DMA register for burst accesses

Reading or writing to the TIMx\_DMAR register will cause

Memory access operation:

TIMx\_CR1 address + DBA + DMA index, where:'TIMx\_CR1

Address' is the address where the control register 1 (TIMx\_CR1) is located;

'DBA' is the base address defined in the TIMx\_DCR register;

'DMA index' is the offset automatically controlled by DMA, it depends on

DBL defined in the TIMx\_DCR register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

233/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 256** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

12 **32**

Use **(TIMx32 Bit)**

32-bit general-purpose timer (TIMx32 Bit)

**12.1 TIMx**

The general-purpose timer is a 32-bit auto-loading counter driven by a programmable prescaler. It is suitable for many

Occasions include measuring the pulse length of an input signal (input capture) or generating an output waveform (output comparison and PWM).

Using timer prescaler and RCC clock controller prescaler, pulse length and waveform period can be in a few microseconds

Adjust to a few milliseconds.

The TIMx timer is completely independent and does not share any resources with each other. They can be operated synchronously together.

**12.2 TIMx**

General TIMx (TIM2) timer functions include:

• 32-bit up, down, up/down auto-loading counter

• 16-bit programmable (can be modified in real time) prescaler, counter clock frequency division factor is between 1 ∼ 65536

Any value of

• 4 independent channels

**–** Input capture

**–** Output comparison

**–** PWM generation (edge ​​​or center alignment mode)

**–** Single pulse mode output

• Use external signal to control timer and timer interconnection synchronization circuit

• An interrupt/DMA is generated when the following events occur:

**–** Update: counter overflow/downflow, counter initialization (triggered by software or internal/external)

**–** Trigger events (counter start, stop, initialization or counting by internal/external trigger)

**–** Input capture

**–** Output comparison

• Supports incremental (quadrature) encoder and Hall sensor circuits for positioning

• Trigger input as external clock or cycle-by-cycle current management

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

234/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 257** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

073662

enter

filter

ITR0

ITR1

ITR2

ITR3

ITR

ETRP

TRGI

TRC

TI1FP1

TI2FP2

ETRF

(CK\_INT)

TRGO

ETR

CNT counter

+/-

PSC

Prescaler

CK\_CNT

CK\_PSC

U

ETRF

Prescaler

IC1PS U

Prescaler

IC2PS

Prescaler

IC3PS

Prescaler

IC4PS

IC1

TI1FP1

TI1FP2

TRC

TI2FP1

TI2FP2

TRC

IC2

IC3

IC4

TRC

TRC

TI1

XOR

TI2

TI3

TIMx\_ETR

TIMx\_CH1

TIMx\_CH2

TIMx\_CH3

TIMx\_CH4

TI4

U

U

U

OC1REF

CC1I

CC2I

OC2REF

OC3REF

CC3I

CC4I

OC4REF

TGI

CC1I

CC2I

CC3I

CC4I

U

UI

OC1N

OC1

OC2

TIMx\_CH1

OC2N

OC3

OC3N

OC4

TI1F\_ED

TI3FP3

TI3FP4

TI4FP3

TI4FP4

TIMx\_CH1N

TIMx\_CH2

TIMx\_CH2N

TIMx\_CH3

TIMx\_CH3N

TIMx\_CH4

Internal clock

TIMxCLK from RCC

Polarity selection, edge

Detector and prescaler

trigger

Controller

Slave mode

Controller

Encoder

interface

Reset, enable, increment */* decrement, count

To other timers

To DAC/ADC

Auto-reload register

Input filter and

Edge detector

Input filter and

Edge detector

Input filter and

Edge detector

Input filter and

Edge detector

Capture/Compare 1 register

Capture/Compare 3 register

Capture/Compare 2 register

Capture/Compare 4 registers

Output

control

Output

control

Output

control

Output

control

Stop, clear, or increase */* decrease

⌘˖

ṩᦞ᧗ࡦսˈ൘ਁ⭏ 8 һԦਾˈ亴㻵䖭ᇴᆈಘcheck ᇩ䖜〫ࡠᴹ᭸ᇴᆈಘ

һԦ

ѝᯝ઼'0$䗃ࠪ

U

Reg

124.

use

**12.3 TIMx**

**12.3.1**

The main part of the programmable general-purpose timer is a 32-bit counter and its associated auto-load register. This meter

The counter can count up, down, or up and down. The counter clock is divided by the prescaler.

The counter, auto-load register and prescaler register can be read and written by software, and can still be read and written while the counter is running,

The time base unit contains:

• Counter register (TIMx\_CNT)

• Prescaler register (TIMx\_PSC)

• Auto load register (TIMx\_ARR)

The auto-reload register is pre-loaded, and writing or reading the auto-reload register will access the pre-load register. According to TIMx

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

235/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 258** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

The setting of the autoload preload enable bit (ARPE) in the \_CR1 register, the content of the preload register is immediately or

It is transferred to the shadow register at each update event UEV. When the counter reaches the overflow condition (underflow when counting down

Condition) and when the UDIS bit in the TIMx\_CR1 register is equal to 0, an update event is generated. Update events can also be

Software generated. The generation of update events under each configuration will be described in detail later.

The counter is driven by the clock output CK\_CNT of the prescaler, only if the counter in the TIMx\_CR1 register is set

CK\_CNT is valid only when the counter enables bit (CEN). (For details on counter enable, please refer to the slave

Mode description).

The prescaler can divide the counter clock frequency by any value between 1 and 65536. It is based on a (in

The 32-bit counter controlled by the 16-bit register in the TIMx\_PSC register. Because this control register is buffered

It can be changed while working. The parameters of the new prescaler are adopted when the next update event arrives.

The following two figures respectively give examples of changing the counter parameters while the prescaler is running.

059785

CK\_PSC

CEN

ᇊᰦಘᰦ䫏 = CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

02

03

0

1

1

0

0

0

0

0

0

1

1

1

1

TIMx\_PSCᇴᆈಘ

125.

1

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

236/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 259** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

763391

CK\_PSC

CEN

ᇊᰦಘᰦ䫏=CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

0

3

3

0

0

0

2

2

0

1

3

1

3

TIMx\_PSCᇴᆈಘ

126.

1

4

**12.3.2**

In the up-counting mode, the counter counts from 0 to the autoload value (the content of the TIMx\_ARR counter), and then restarts

Start counting from 0 and generate a counter overflow event.

An update event can be generated each time the counter overflows, and the UG bit is set in the TIMx\_EGR register (via software

Or use slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIMx\_CR1 register can disable the update event; this can avoid registering to the preload

The shadow register is updated when a new value is written in the device. Until the UDIS bit is cleared, no update event will be generated. But in response

When the update event occurs, the counter will still be cleared to 0, and the count of the prescaler is also requested to 0 (but the value of the prescaler

constant). In addition, if the URS bit in the TIMx\_CR1 register is set (select an update request), setting the UG bit will

An update event UEV is generated, but the UIF flag is not set by the hardware (that is, no interrupt or DMA request is generated). This is for

Avoid generating update and capture interrupts simultaneously when the counter is cleared in capture mode.

When an update event occurs, all registers are updated, and the hardware sets the update flag at the same time (according to the URS bit)

Bit (UIF bit in the TIMx\_SR register).

• The buffer of the prescaler is placed in the value of the preload register (the content of the TIMx\_PSC register)

• The autoload shadow register is reset to the value of the preload register (TIMx\_ARR)

The following figure shows some examples. When TIMx\_ARR = 0x36, the counter will operate at different clock frequencies:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

237/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 260** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

894901

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

31

32 33 34 35 36

05

Counter overflow

Update interrupt flag (UIF)

00 01 02 03 04

06 07

127.

1

100720

0034

0035

0000

0036

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

128.

2

564964

0035

0000

0036

0001

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

129.

4

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

238/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 261** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

372226

1F

20

00

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

130.

N

370803

Write new value in TIMx\_ARR

31

32

33 34

35

36

05

00 01

02

03

04

06

07

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

131.

ARPE = 0

(TIMx\_ARR

)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

239/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 262** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

874537

Write new value in TIMx\_ARR

F0

F1 F2 F3 F4 F5

05

00 01 02 03 04

06 07

F5

36

F5

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

132.

ARPE = 1

(TIMx\_ARR)

In down mode, the counter counts down to 0 from the automatically loaded value (TIMx\_ARR counter value), and then

It restarts from the automatically loaded value and generates a counter underflow event.

An update event can be generated each time the counter overflows, and the UG bit is set in the TIMx\_EGR register (via software

Or use slave mode controller) can also generate an update event.

Setting the UDIS bit in the TIMx\_CR1 register can disable UEV events. This can avoid preloading registers

The shadow register is updated when a new value is written. Therefore, no update event will be generated until the UDIS bit is cleared to 0. However, counting

The counter will still restart counting from the current auto-load value, and the counter of the prescaler restarts from 0 (but the prescaler

The speed of the device cannot be modified).

In addition, if the URS bit in the TIMx\_CR1 register is set (select update request), setting the UG bit will generate a

Update event UEV without setting the UIF flag (so no interrupts and DMA requests are generated), this is to avoid

When the event is captured and the counter is cleared, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and (according to the setting of the URS bit) update the flag bit (TIMx\_

The UIF bit in the SR register) is also set.

• The prescaler buffer is set to the value of the preload register (the value of the TIMx\_PSC register).

• The current autoload register is updated to the preload value (the content in the TIMx\_ARR register).

Note: Autoload is updated before the counter is reloaded, so the next cycle will be the expected value.

The following are some examples of counter operation at different clock frequencies when TIMx\_ARR = 0x36:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

240/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 263** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

721721

05

04

03

02

01

00

31

36

35

34

33

32

30

2F

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

133.

1

819787

0002

0001

0036

0000

0035

0034

0033

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

134.

2

213411

0001

0036

0000

0035

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

135.

4

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

241/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 264** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

264550

20

1F

00

36

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

136.

N

367681

05

04

03 02

01

00

31

36 35

34

33

32

30

2F

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

Write new value in TIMx\_ARR

137.

use

**(**

**/**

**)**

In center-aligned mode, the counter starts counting from 0 to the automatically loaded value (TIMx\_ARR register) -1, generating a

Counter overflow event, then count down to 1 and generate a counter underflow event; then start again from 0

count.

In this mode, the DIR direction bit in TIMx\_CR1 cannot be written. It is updated by the hardware and indicates the current counting method

to. The update event can be generated at every count overflow and every count underflow; can also be passed (software or use slave mode

Controller) Set the UG bit in the TIMx\_EGR register to generate, at this time, the counter starts counting again from 0

The frequency counter starts counting from 0 again.

Setting the UDIS bit in the TIMx\_CR1 register can disable UEV events. This can avoid pre-loading registration

The shadow register is updated when a new value is written in the device. Therefore, no update event will be generated until the UDIS bit is cleared to 0. However,

The counter will continue to count up or down based on the current auto-reload value.

In addition, if the URS bit in the TIMx\_CR1 register is set (select update request), setting the UG bit will generate a

Update event UEV without setting the UIF flag (so no interrupts and DMA requests are generated), this is to avoid

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

242/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 265** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

When the event is captured and the counter is cleared, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and (according to the setting of the URS bit) update the flag bit (TIMx\_SR

The UIF bit in the register) is also set.

• The prescaler buffer is loaded with the preload (TIMx\_PSC register) value

• The current autoload register is updated to the preload value (the content in the TIMx\_ARR register)

Note: If an update occurs due to a counter overflow, the automatic reload will be updated before the counter is reloaded, so the next cycle

It will be the expected value (the counter is loaded with the new value).

The following are some examples of counter operations at different clock frequencies:

775077

04

03

02 01

00 01

05

02 03 04 05 06

04 03

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Counter underflow

138.

1TIMx\_ARR = 0x6

824268

0003

0002

0000

0001

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

139.

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

243/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 266** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

464735

0034

0036

0035

0035

Note: Center-aligned mode 2 or mode 3 is used in conjunction with the overflow UIF.

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

140.

4TIMx\_ARR = 0x36

220046

20

1F

00

01

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

141.

N

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

244/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 267** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

482632

06

05

04 03

02

01

05

00

01 02

03

04

06

07

FD

36

FD

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

Write new value in TIMx\_ARR

142.

ARPE = 1

()

970767

F7

F8

F9 FA FB FC

31

36

35 34

33

32

30

2F

FD

36

FD

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

Write new value in TIMx\_ARR

143.

ARPE = 1

()

**12.3.3**

The counter clock can be provided by the following clock sources:

• Internal clock (CK\_INT)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

245/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 268** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

• External clock mode 1: External input pin (TIx)

• External clock mode 2: External trigger input (ETR)

• Internal trigger input (ITRx): use one timer as the prescaler of another timer, if you can configure one

Timer1 acts as a prescaler for another timer, Timer2.

**(CK\_INT)**

If the slave mode controller is disabled (SMS = 000), the CEN, DIR (TIMx\_CR1 register) and UG bits (TIMx\_

EGR register) is the de facto control bit and can only be modified by software (UG bit is still automatically cleared). When the CEN bit

When written as 1, the prescaler clock is provided by the internal clock CK\_INT.

The figure below shows the operation of the control circuit and the up-counter in normal mode without the prescaler.

199864

31

32

33 34

35

36

05

00

01

02

03

04

06

07

Internal clock

CEN = CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

CNT\_INT

UG

144.

1

**1**

When SMS = 111 in the TIMx\_SMCR register, this mode is selected. The counter can be

Count on a rising or falling edge.

356901

Encoder mode

External clock

Mode 1

External clock

Mode 2

Internal clock

mode

CK\_PSC

filter

ICF[3:0]

edge

Detector

CC2P

TS[2:0]

SMS[2:0]

ECE

TI2

TIMx\_CCMR1

TIMx\_CCER

TI2F\_Rising

TI2F\_Falling

ITRx

TI1F\_ED

TI1FP1

TI2FP2

ETRF

001

100

101

110

111

TRGI

ETRF

CK\_INT

(Internal clock)

TIMx\_SMCR

TI2F

TI1F

or

or

or

TIMx\_SMCR

0

1

145. TI2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

246/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 269** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

For example, to configure the up counter to count on the rising edge of the T12 input, use the following steps:

1. Configure TIMx\_CCMR1 register CC2S = 01, configure channel 2 to detect the rising edge of TI2 input

2. Configure IC2F[3:0] in the TIMx\_CCMR1 register and select the input filter bandwidth (if the filter is not required, ensure

Hold IC2F = 0000)

Note: The capture prescaler is not used as a trigger, so there is no need to configure it

3. Configure CC2P = 0 of the TIMx\_CCER register to select the polarity of the rising edge

4. Configure SMS = 111 in the TIMx\_SMCR register to select the timer external clock mode 1

5. Configure TS = 110 in the TIMx\_SMCR register and select TI2 as the trigger input source

6. Set CEN = 1 in the TIMx\_CR1 register to start the counter

When the rising edge appears on TI2, the counter counts once and the TIF flag is set.

The delay between the rising edge of TI2 and the actual clock of the counter depends on the resynchronization circuit at the input of TI2.

518995

TI2

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

Write TIF = 0

34

36

TIF

35

146.

1

**2**

The method to select this mode is: let ECE = 1 in the TIMx\_SMCR register to trigger the ETR externally

Each rising or falling edge counts. The following figure is the overall block diagram of the external trigger input:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

247/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 270** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

132558

Encoder mode

External clock

Mode 1

External clock

Mode 2

Internal clock

mode

CK\_PSC

filter

Down counter

Crossover

/1,/2,/4,/8

SMS[2:0]

ECE

ETR pin

ETR

TRGI

ETRF

CK\_INT

(Internal clock)

TI2F

TI1F

or

or

or

0

1

ETP

ETRP

CK\_INT

ETF[3:0]

ETPS[1:0]

TIMx\_SMCR

TIMx\_SMCR

TIMx\_SMCR

TIMx\_SMCR

147.

For example, to configure an up counter that counts every 2 rising edges under ETR, use the following steps:

1. No filter is needed in this example, set ETF[3:0] = 0000 in the TIMx\_SMCR register

2. Set the prescaler and set ETPS[1:0] = 01 in the TIMx\_SMCR register

3. Set the detection on the rising edge of ETR, set ETP = 0 in the TIMx\_SMCR register

4. Turn on external clock mode 2 and set ECE = 1 in the TIMx\_SMCR register

5. Start the counter and set CEN = 1 in the TIMx\_CR1 register

The counter counts every 2 rising edges of ETR.

The delay between the rising edge of ETR and the actual clock of the counter depends on the resynchronization circuit at the ETRP signal terminal.

052284

34

36

35

ETRP

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

ETRF

ETR

f CK\_INT

148.

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

248/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 271** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**12.3.4**

**/**

Each capture/compare channel surrounds a capture/compare register (including shadow registers), including the captured input

Part (digital filtering, multiplexing and prescaler), and output part (comparator and output control).

The following figures are an overview of the capture/compare channels. The input section samples the corresponding TIx input signal and generates a

Filtered signal TIxF. Then, an edge monitor with polarity selection generates a signal (TIxFPx), which

Can be triggered as input from the mode controller or as a capture control. This signal enters the capture register through prescaler

(ICxPS).

565511

IC1PS

filter

Down counter

ICF[3:0]

edge

Detector

CC1P

ICPS[1:0]

TI1

TIMx\_CCMR1

TIMx\_CCER

TI1F\_Rising

TI1F\_Falling

01

10

11

(Since mode controller)

0

1

f DTS

TI1F

0

1

divider

/1,/2,/4,/8

IC1

CC1E

CC1S[1:0]

TIMx\_CCMR1

TIMx\_CCER

TRC

TI2F\_Rising

TI2F\_Falling

(from channel 2)

(from channel 2)

TI2FP1

TI1FP1

TI1F\_ED

To slave mode controller

149.

/ ( 1

)

The output section generates an intermediate waveform OCxRef (high effective) as a reference, the end of the chain determines the pole of the final output signal

Sex.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

249/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 272** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

903065

MCU peripheral interface

APB bus

Capture/compare preload registers

high

8

CC1S[1]

OC1PE

TIMx\_CCMR1

UEV

Reading CCR1L

Reading CCR1H

CC1S[1]

CC1S[0]

IC1PS

CC1E

CC1G

TIMx\_EGR

read\_in\_progress

enter

mode

8

low

(Assuming 16 bits)

Capture/Compare Shadow Register

capture

counter

Comparators

capture\_transfer

compare\_transfer

write\_in\_progress

Output

mode

CNT> CCR1

CNT = CCR1

(From time base unit)

Write to CCR1H

Write to CCR1L

CC1S[0]

OC1PE

S

R

S

R

150.

/ 1

358425

Output mode

Controller

OC1M[2:0]

TIMx\_CCMR1

CNT>CCR1

CNT=CCR1

ETRF

OC1ref

To slave mode controller

0

1

CC1P

TIMx\_CCER

Output enable

Circuit

CC1E TIMx\_CCER

OC1

151.

/ ( 1)

The capture/compare module consists of a preload register and a shadow register. The read and write process only operates preload registration

Device. In capture mode, the capture occurs on the shadow register, and then copied into the preload register.

In the comparison mode, the contents of the preload register are copied to the shadow register, and then the contents of the shadow register and

Counter.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

250/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 273** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**12.3.5**

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched to capture/ratio

Compare register (TIMx\_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx\_SR register) is

Set to 1, if interrupt or DMA operation is enabled, an interrupt or DMA operation will be generated. If a capture event occurs

When the CCxIF flag is already high, the repeated capture flag CCxOF (TIMx\_SR register) is set. Write CCxIF

= 0 can clear CCxIF, or read the captured data stored in the TIMx\_CCRx register can also clear CCxIF. write

CCxOF = 0 can clear CCxOF.

The following example shows how to capture the counter value into the TIMx\_CCR1 register on the rising edge of the TI1 input, step

as follows:

• Select valid input: TIMx\_CCR1 must be connected to TI1 input, so write to TIMx\_CCR1 register

CC1S = 01, when CC1S is not 00, the channel is configured as an input, and the TM1\_CCR1 register becomes

Is read-only.

• According to the characteristics of the input signal, configure the input filter to the required bandwidth (that is, when the input is TIx, the input filter controls

The control bit is the ICxF bit in the TIMx\_CCMRx register). Assuming that the input signal is at most 5 clock cycles

For internal jitter, we must configure the filter bandwidth to be longer than 5 clock cycles. So we can connect (at fDTS frequency)

Continue sampling 8 times to confirm the last real edge transition on TI1, which is written in the TIMx\_CCMR1 register

IC1F = 0011.

• Select the valid conversion edge of the TI1 channel and write CC1P = 0 (rising edge) in the TIMx\_CCER register.

• Configure input prescaler. In this example, we want to capture every valid level-shifting moment, so

The prescaler is disabled (write IC1PS = 00 in the TIMx\_CCMR1 register).

• Set CC1E = 1 in the TIMx\_CCER register to allow the capture counter value to be captured in the capture register.

• If necessary, enable the relevant interrupt request by setting the CC1IE bit in the TIMx\_DIER register, by setting

The CC1DE bit in the TIMx\_DIER register allows DMA requests.

When an input is captured:

• When a valid level transition occurs, the counter value is transferred to the TIMx\_CCR1 register.

• The CC1IF flag is set (interrupt flag). When at least 2 consecutive captures occur, CC1IF has not been cleared.

• CC1OF is also set to 1.

• If the CC1IE bit is set, an interrupt will be generated.

• If the CC1DE bit is set, a DMA request will also be generated.

In order to deal with the capture overflow, it is recommended to read the data before reading the capture overflow flag, this is to avoid losing the read capture

Capture overflow information that may be generated after the overflow flag is obtained and before the data is read.

Note: By setting the corresponding CCxG bit in the TIMx\_EGR register, an input capture interrupt and/or DMA request can be generated by software.

**12.3.6 PWM**

This mode is a special case of the input capture mode, except for the following differences, the operation is the same as the input capture mode:

• Two ICx signals are mapped to the same TIx input

• 2 ICx signals are edge valid, but the polarity is reversed

• One of the TIxFP signals is used as a trigger input signal, and the slave mode controller is configured in reset mode

For example, you need to measure the length of the PWM signal input to TI1 (TIMx\_CCR1 register) and the duty cycle (TIMx\_

CCR2 register), the specific steps are as follows (depending on the frequency of CK\_INT and the value of prescaler)

• Select the valid input of TIMx\_CCR1: Set CC1S = 01 in the TIMx\_CCMR1 register (select TI1)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

251/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 274** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

• Select the effective polarity of TI1FP1 (used to capture data into TIMx\_CCR1 and clear the counter): set CC1P =

0 (rising edge valid).

• Select the valid input of TIMx\_CCR2: Set CC2S = 10 in the TIMx\_CCMR1 register (select TI1).

• Select the effective polarity of TI1FP2 (capture data to TIMx\_CCR2): set CC2P = 1 (active on falling edge).

• Select a valid trigger input signal: set TS = 101 in the TIMx\_SMCR register (select TI1FP1).

• Configure the slave mode controller to reset mode: set SMS = 100 in TIMx\_SMCR.

• Enable capture: Set CC1E = 1 and CC2E = 1 in the TIMx\_CCER register.

789042

0004

0000

0001

0002

0003

0004

0000

0004

0002

TI1

TIMx\_CNT

TIMx\_CCR1

TIMx\_CCR2

IC1 capture

IC2 capture

Reset counter

IC2 capture

Pulse width measurement

IC1 capture

Period measurement

152.

/ ( 1)

Since only TI1FP1 and TI2FP2 are connected to the slave mode controller. So PWM input mode can only use TIMx\_CH1

/ TIMx\_CH2 signal.

**12.3.7**

In the output mode (CCxS = 00 in the TIMx\_CCMRx register), the comparison signal (OCxREF and the corresponding

OCx) can be forced into valid or invalid state directly by software, without depending on the output compare register and counter

Comparing results.

Set the corresponding OCxM = 101 in the TIMx\_CCMRx register to force the output compare signal (OCxREF/OCx)

Is valid. In this way, OCxREF is forced to high level (OCxREF is always active high), and OCx is

To the opposite value of the CCxP polarity bit. For example: CCxP = 0 (OCx high level effective), then OCx is forced to high level.

Set OCxM = 100 in the TIMx\_CCMRx register to force the OCxREF signal low. In this mode, in

The comparison between the TIMx\_CCRx shadow register and the counter is still in progress, and the corresponding flag will be modified. Therefore

Corresponding interrupts and DMA requests will still be generated. This will be described in the section on output comparison mode below.

**12.3.8**

This function is used to control an output waveform or indicate when a given period of time has expired.

When the contents of the counter and the capture/compare register are the same, the output compare function does the following:

• Register the output compare mode (OCxM bit in the TIMx\_CCMRx register) and output polarity (TIMx\_CCER

The value defined by the CCxP bit in the device is output to the corresponding pin. The output pin can hold it when comparing and matching

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

252/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 275** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

Level (OCxM = 000), set to the active level (OCxM = 001), set to no active level (OCxM

= 010) or roll over (OCxM = 011).

• Set the flag bit in the interrupt status register (CCxIF bit in the TIMx\_SR register).

• If the corresponding interrupt mask (CCXIE bit in the TIMx\_DIER register) is set, an interrupt is generated.

• If the corresponding enable bit is set (CCxDE bit in the TIMx\_DIER register, the TIMx\_CR2 register

CCDS bit selects DMA request function), then generate a DMA request.

The OCxPE bit in TIMx\_CCMRx selects whether the TIMx\_CCRx register needs to use a preload register. Losing

In the compare mode, the update event UEV has no effect on the OCxREF and OCx outputs. Synchronization accuracy can be achieved

One counting cycle of the counter. The output compare mode (in single pulse mode) can also be used to output a single pulse.

Configuration steps of output comparison mode:

1. Select counter clock (internal, external, prescaler)

2. Write the corresponding data to the TIMx\_ARR and TIMx\_CCRx registers

3. If an interrupt request and/or a DMA request is to be generated, set the CCxIE bit and/or CCxDE bit

4. Select the output mode, for example: OCxM = '011', OCxPE = '0', CCxP = '0' and

CCxE = '1', when the counter CNT matches CCRx, the output pin of OCx is toggled, CCRx is not preloaded

Use, enable OCx output and high level is effective

5. Set the CEN bit of the TIMx\_CR1 register to start the counter

The TIMx\_CCRx register can be updated by software at any time to control the output waveform, provided the pre-use is not used

Load register (OCxPE = '0', otherwise the TIMx\_CCRx shadow register can only be used when the next update event occurs

Updated). The figure below gives an example.

803963

0039

003A

003B

B200

B201

003A

B201

TIMX\_CNT

TIMX\_CCR1

OC1REF = OC1

Write B201h in CCR1 register

A match was detected on CCR1

If the interrupt is enabled, an interrupt is generated

153.

OC1

**12.3.9 PWM**

Pulse width modulation mode can generate a frequency determined by the TIMx\_ARR register and confirmed by the TIMx\_CCRx register

A signal with a fixed duty cycle.

Write '110' (PWM mode 1) or '111' (PWM mode) in the OCxM bit in the TIMx\_CCMRx register

2), each OCx output channel can be set independently to generate a PWM. The TIMx\_CCMRx register must be set

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

253/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 276** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

OCxPE bit to enable the corresponding preload register, and finally set the ARPx bit of the TIMx\_CR1 register to enable

Preload registers for automatic reload (in up-count or center-symmetric mode).

Because only when an update event occurs, the preload register can be transferred to the shadow register, so in the count

Before the device starts counting, all registers must be initialized by setting the UG bit in the TIMx\_EGR register.

The polarity of OCx can be set by the CCxP bit in the TIMx\_CCER register by software, it can be set to high

Active level or active low level. The CCxE bit in the TIMx\_CCER register controls OCx output enable. See more

Description of the TIMx\_CCERx register.

In PWM mode (mode 1 or mode 2), TIMx\_CNT and TIM1\_CCRx are always compared, (based on the count

The counting direction of the device) to determine whether it meets TIM1\_CCRx ≤ TIM1\_CNT or TIM1\_CNT ≤ TIM1\_CCRx.

However, in order to interact with the function of OCREF\_CLR (before the next PWM cycle, an external event on the ETR signal

Can clear OCxREF). The OCxREF signal can only be generated under the following conditions:

• When the result of the comparison changes

• When the output compare mode (OCxM bit in the TIMx\_CCMRx register) is changed from'frozen' (no comparison, OCxM =

'000') Switch to a PWM mode (OCxM = '110' or '111')

In this way, the PWM output can be forced by software during operation. According to the state of the CMS bit in the TIMx\_CR1 register,

The timer can generate an edge-aligned PWM signal or a center-aligned PWM signal.

**PWM**

Count up when the DIR bit in the TIMx\_CR1 register is low.

The following is an example of PWM mode 1. PWM signal reference OCxREF when TIMx\_CNT <TIMx\_CCRx

High, otherwise low. If the comparison value in TIMx\_CCRx is greater than the auto-reload value (TIMx\_ARR), then OCxREF

Keep it as '1'. If the comparison value is 0, OCxREF remains at '0'. The following figure shows the edge pair when TIMx\_ARR = 8

Examples of homogeneous PWM waveforms.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

254/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 277** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

458343

0 1 2 3 4 5 6 7 8 0 1

þ1ÿ

þ0ÿ

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx = 4

CCRx = 8

CCRx >8

CCRx =0

Counter register

154.

PWM

(ARR = 8)

Countdown is performed when the DIR bit of the TIMx\_CR1 register is high.

In PWM mode 1, the reference signal OCxREF is low when TIMx\_CNT> TIMx\_CCRx, otherwise it is high. Such as

If the comparison value in TIMx\_CCRx is greater than the auto-reload value in TIMx\_ARR, OCxREF remains at '1'.

In this mode, a 0% PWM waveform cannot be generated.

**PWM**

When the CMS bit in the TIMx\_CR1 register is not '00', it is the center-aligned mode (all other configurations

/OCx signals have the same effect). According to the setting of different CMS bits, the comparison flag can count up in the counter

Set to 1 at time, set to 1 when the counter counts down, or set to 1 when the counter counts up and down. TIMx\_CR1

The count direction bit (DIR) in the register is updated by hardware, do not modify it by software. See the center alignment mode chapter.

The following figure shows some examples of center-aligned PWM waveforms

• TIMx\_ARR = 8

• PWM mode 1

• CMS = 01 in the TIMx\_CR1 register, in center-aligned mode 1, the ratio is set when the counter counts down

More mark

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

255/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 278** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

546141

0 1 2 3 4 5 6 7 8 7 6

þ1ÿ

þ0ÿ

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx = 4

CCRx = 8

CCRx >8

CCRx =0

CMS =01

CMS =10

CMS =11

CMS =10 or 11

CMS =01

CMS =10

CMS =11

þ1ÿ

CMS =01

CMS =10

CMS =11

CMS =01

CMS =10

CMS =11

5 4 3 2 1 0 1

CCRx =7

OCxREF

CCxIF

Counter register

155.

PWM

(APR = 8)

use

• When entering the center alignment mode, the current up/down counting configuration is used; this means whether the counter is counting up or down

It depends on the current value of the DIR bit in the TIMx\_CR1 register. In addition, the software cannot modify DIR and CMS at the same time

Bit.

• It is not recommended to rewrite the counter when running in center-aligned mode, as it will produce unpredictable results. specifically :

**–** If the value written to the counter is greater than the value of auto-reload (TIMx\_CNT> TIMx\_ARR), the direction

Will not be updated. For example, if the counter is counting up, it will continue to count up.

**–** If the value of 0 or TIMx\_ARR is written to the counter, the direction is updated, but no update event is generated

UEV

• The safest way to use the center alignment mode is to generate a software update (set TIMx\_

UG bit in the EGR bit), do not modify the counter value while the count is in progress.

**12.3.10**

The single pulse mode (OPM) is a special case of the aforementioned many modes. This mode allows the counter to respond to an stimulus and

A programmable pulse is generated after a programmable delay.

The counter can be started by the slave mode controller to generate a waveform in the output comparison mode or PWM mode. Set up

The OPM bit in the TIMx\_CR1 register will select the one-shot mode, which allows the counter to automatically generate the next pulse

Stopped during an update event UEV.

A pulse can be generated only when the comparison value is different from the initial value of the counter. Before starting (when the timer is waiting for a touch

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

256/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 279** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

Issue), must be configured as follows:

• Up counting method: CNT <CCRx ≤ ARR (in particular, 0 <CCRx)

• Countdown method: CNT> CCRx

296813

t DELAY

t PULSE

t

0

meter

Counter

TIM1\_ARR

TIM1\_CCR1

OC1

OC1REF

TI2

156.

For example, you need to detect a rising edge on the input pin of TI2, delay t DELAY, and then generate it on OC1.

Generate a positive pulse of length t PULSE .

Assume TI2FP2 as trigger 1:

• Set CC2S = 01 in the TIMx\_CCMR1 register to map TI2FP2 to TI2

• Set CC2P = 0 in the TIMx\_CCER register to enable TI2FP2 to detect the rising edge

• Set TS = 110 in the TIMx\_SMCR register to use TI2FP2 as the slave mode controller trigger (TRGI)

• Set SMS = 110 (trigger mode) in the TIMx\_SMCR register, TI2FP2 is used to start the counter

The OPM waveform is determined by the value written to the compare register (the clock frequency and counter prescaler must be considered).

• t DELAY is defined by the value written to the TIMx\_CCR1 register.

• t PULSE is defined by the difference between the autoload value and the comparison value (TIMx\_ARR-TIMx\_CCR1).

• Assume that a waveform from 0 to 1 is generated when a compare match occurs, and a slave is generated when the counter reaches the preload value

1 to 0 waveform; first set OC1M = 111 in TIMx\_CCMR1 register to enter PWM mode 2; root

Selectively enable preload registers as needed: set OC1PE = 1 and TIMx\_CR1 in TIMx\_CCMR1

ARPE in the register; then fill in the comparison value in the TIMx\_CCR1 register, in the TIMx\_ARR register

Fill in the autoload value, modify the UG bit to generate an update event, and then wait for an external trigger on TI2

event. In this example, CC1P = 0.

In this example, the DIR and CMS bits in the TIMx\_CR1 register should be set low.

Because only one pulse is needed, it is necessary to set OPM = 1 in the TIMx\_CR1 register for the next update event

(When the counter rolls over from the autoload value to 0) stop counting.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

257/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 280** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**OCx**

In the one-shot mode, the edge detection logic on the TIx input pin sets the CEN bit to start the counter. Then the counter and

The comparison operation between the comparison values ​​​​produces a conversion of the output. But these operations require a certain clock cycle, so it limits

The minimum delay tDELAY available.

If you want to output the waveform with minimum delay, you can set the OCxFE bit in the TIMx\_CCMRx register;

OCxREF (and OCx) are forced to respond to the excitation and no longer depend on the result of the comparison. The output waveform matches the comparison waveform.

Same shape. OCxFE only works when the channel is configured in PWM1 and PWM2 modes.

**12.3.11**

**OCxREF**

For a given channel, set the corresponding OCxCE bit in the TIMx\_CCMRx register at the ETRF input

'1') high level can pull the OCxREF signal low, the OCxREF signal will remain low until the next change occurs

New event UEV.

This function can only be used for output comparison and PWM mode, but not for forced mode.

For example, the OCxREF signal can be connected to an external input. At this time, ETR must be configured as follows:

• The external trigger prescaler must be off: ETPS[1:0] = 00 in the TIMx\_SMCR register

• External clock mode 2 must be disabled: ECE = 0 in the TIMx\_SMCR register

• External trigger polarity (ETP) and external trigger filter (ETF) can be configured as required

The figure below shows the behavior of the OCxREF signal corresponding to different OCxCE values ​​​​when the ETRF input goes high. At this

In this example, the timer TIMx is placed in PWM mode.

079702

Counter ( CNT )

( CCRx )

ETRF

OCxREF

( OCxCE ='0')

OCxREF

( OCxCE ='1')

OCREF \_ CLR

OCREF \_ CLR

Go high

Keep high

157.

TIMx

OCxREF

**12.3.12**

The method to select the encoder interface mode is: if the counter only counts on the edge of TI2, set TIMx\_SMCR to register

SMS = 001 in the device; if only counting on the edge of TI1, set SMS = 010; if the counter is in both TI1 and TI1

TI2 edge count, then set SMS = 011.

The polarity of TI1 and TI2 can be selected by setting the CC1P and CC2P bits in the TIMx\_CCER register; if required

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

258/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 281** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

If necessary, the input filter can also be programmed.

The two inputs TI1 and TI2 are used as the interface for the incremental encoder. The following table assumes that the counter has started (TIMx\_CR1

CEN = 1) in the register, the counter is driven by each valid transition on TI1FP1 or TI2FP2. TI1FP1

And TI2FP2 are the signals after TI1 and TI2 pass the input filter and polarity control; if there is no filtering and disguise, then

TI1FP1 = TI1; if there is no filtering and disguise, then TI2FP2 = TI2. According to the transition sequence of the two input signals, the

Count pulse and direction signal are generated. According to the transition sequence of the two input signals, the counter counts up or down, and at the same time

The hardware sets the DIR bit of the TIMx\_CR1 register accordingly. Regardless of whether the counter counts on TI1,

TI2 counts or counts on TI1 and TI2 simultaneously. The transition at any input (TI1 or TI2) is recalculated

DIR bit.

The encoder interface mode is basically equivalent to using an external clock with direction selection. This means that the counter is only on

Continuous counting from 0 to the autoload value of the TIMx\_ARR register (according to the direction, either 0 to ARR count, or

ARR to 0 count). Therefore, TIMx\_ARR must be configured before starting counting; similarly, the catcher, comparator, pre-

The frequency divider and trigger output characteristics still work as usual.

In this mode, the counter is automatically modified according to the speed and direction of the incremental encoder, so the content of the counter

Finally indicates the position of the encoder. The counting direction corresponds to the direction of rotation of the connected sensor. The following table lists all possible

, Assuming that TI1 and TI2 are not transformed simultaneously.

43.

Effective edge

Relative signal level

**(TI1FP1** corresponds to **TI2** ,

**TI2FP2** corresponds to **TI1)**

**TI1FP1** signal

**TI1FP2** signal

rise

decline

rise

decline

Only count at TI1

high

Count down

Count up

Do not count

Do not count

low

Count up

Count down

Do not count

Do not count

Counting on TI2 only

high

Do not count

Do not count

Count up

Count down

low

Do not count

Do not count

Count down

Count up

On TI1 and TI2

count

high

Count down

Count up

Count up

Count down

low

Count up

Count down

Count down

Count up

An external incremental encoder can be directly connected to the MCU without the need for external interface logic. However, the general use ratio

The comparator converts the differential output of the encoder to a digital signal, which greatly increases the ability to resist noise interference. Encoder output

The third signal represents the mechanical zero point, which can be connected to an external interrupt input and trigger a counter reset.

The following figure is an example of counter operation, showing the generation of count signals and direction control. It also shows when selected

How to suppress the input jitter on both sides of the edge; jitter may be generated when the sensor position is close to a transition point.

In this example, we assume the configuration is as follows:

• C1S = '01' (TIMx\_CCMR1 register, IC1FP1 is mapped to TI1)

• CC2S = '01' (TIMx\_CCMR2 register, IC2FP2 is mapped to TI2)

• CC1P = '0' (TIMx\_CCER register, IC1FP1 is not inverted, IC1FP1 = TI1)

• CC2P = '0' (TIMx\_CCER register, IC2FP2 is not inverted, IC2FP2 = TI2)

• SMS = '011' (TIMx\_SMCR register, all inputs are valid on rising and falling edges)

• CEN = '1' (TIMx\_CR1 register, counter enable)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

259/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 282** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

119663

Increment

TI1

counter

Diminishing

Increment

TI2

Forward

Jitter

Reverse

Jitter

Forward

158.

The following figure shows the operation example of the counter when the polarity of IC1FP1 is inverted (CC1P = '1', other configurations are the same as the above example)

002625

Diminishing

TI1

counter

Increment

Diminishing

TI2

Forward

Jitter

Reverse

Jitter

Forward

159. IC1FP1

When the timer is configured in encoder interface mode, it provides information about the current position of the sensor. Use the second configuration in the capture

The mode timer measures the interval between two encoder events and can obtain dynamic information (speed, acceleration, deceleration). Finger

The encoder output showing the mechanical zero point can be used for this purpose. According to the interval between two events, you can follow a fixed time

Read the counter. If possible, you can latch the counter value to the third input capture register (capture signal

Must be periodic and can be generated by another timer). It can also use a DMA generated by a real-time clock

Request to read its value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

260/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 283** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**12.3.13**

The TI1S bit in the TIMx\_CR2 register allows the input filter of channel 1 to be connected to the output of an XOR gate.

The three inputs of the OR gate are TIMx\_CH1, TIMx\_CH2, and TIMx\_CH3.

The XOR output can be used for all timer input functions such as triggering or input capture. Section [10.3.](https://translate.googleusercontent.com/translate_f#162) 18 gives this special

An example of how to connect Hall sensors.

**12.3.14**

The TIMx timer can be synchronized with an external trigger in multiple modes: reset mode, gating mode, and trigger mode.

When a trigger input event occurs, the counter and its prescaler can be re-initialized; meanwhile, if TIMx\_CR1

The URS bit of the register is low, and an update event UEV is also generated; then all the preload registers (TIMx\_ARR

, TIMx\_CCRx) have been updated.

• In the following example, the rising edge of the TI1 input causes the up counter to be cleared

• Configure channel 1 to detect the rising edge of TI1. Configure the bandwidth of the input filter (in this example, no filtering is required

Device, so keep IC1F = 0000). The capture prescaler is not used in the trigger operation, so no configuration is required. CC1S

The bit selects only the input capture source, that is, CC1S = 01 in the TIMx\_CCMR1 register. Set the TIMx\_CCER register

CC1P = 0 to determine the polarity (only the rising edge is detected)

• Set SMS = 100 in the TIMx\_SMCR register to configure the timer in reset mode; set the TIMx\_SMCR register

TS = 101, select TI1 as input source

• Set CEN = 1 in the TIMx\_CR1 register to start the counter

The counter starts counting according to the internal clock, and then runs normally until TI1 has a rising edge; at this time, the counter is

Clear to zero and restart counting from 0. At the same time, the trigger flag (TIF bit in the TIMx\_SR register) is set, the root

According to the settings of the TIE (Interrupt Enable) bit and TDE (DMA Enable) bit in the TIMx\_DIER register, please generate an interrupt.

Request or a DMA request.

The figure below shows the behavior when the auto-reload register TIMx\_ARR = 0x36. At the rising edge of TI1 and the actual counter

The delay between resets depends on the resynchronization circuit at the input of TI1.

729416

33

32

31

30

00

01 02 03

36

00

01

02

34 35

03

TI1

UG

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

160.

The enable of the counter depends on the level of the selected input.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

261/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 284** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

In the following example, the counter only counts up when TI1 is low:

• Configure channel 1 to detect a low level on TI1. Configure the input filter bandwidth (in this example, no filtering is required, so keep

Hold IC1F = 0000). The capture prescaler is not used in the trigger operation, so no configuration is required. CC1S bit is used for selection

Enter the capture source and set CC1S = 01 in the TIMx\_CCMR1 register. Set CC1P in TIMx\_CCER register

= 1 to determine the polarity (only low level detection).

• Set SMS = 101 in the TIMx\_SMCR register to configure the timer in gating mode; set the TIMx\_SMCR register

TS = 101, select TI1 as the input source.

• Set CEN = 1 in the TIMx\_CR1 register to start the counter. In gated mode, if CEN = 0, count

The device cannot be started regardless of the trigger input level.

As long as TI1 is low, the counter starts counting based on the internal clock and stops counting when TI1 goes high. When the counter starts or

When stopping, set the TIF flag in TIMx\_SR.

The delay between the rising edge of TI1 and the actual stop of the counter depends on the resynchronization circuit at the input of TI1.

704741

**34**

**33**

**32**

**31**

**30**

**35**

**36 37 38**

Write TIF = 0

TI1

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

161.

The enabling of the counter depends on the event on the selected input.

In the following example, the counter starts counting up on the rising edge of the TI2 input:

• Configure channel 2 to detect the rising edge of TI2. Configure the input filter bandwidth (in this example, no filter is required, ensure that

Hold IC2F = 0000). The capture prescaler is not used in the trigger operation, and no configuration is required. CC2S bit is for selection only

Enter the capture source and set CC2S = 01 in the TIMx\_CCMR1 register. Set CC1P in TIMx\_CCER register

= 1 to determine the polarity (only low level detection).

• Set SMS = 110 in the TIMx\_SMCR register to configure the timer in trigger mode; set the TIMx\_SMCR register

TS = 110, select TI2 as the input source.

When TI2 has a rising edge, the counter starts to count under the drive of the internal clock and sets the TIF flag.

The delay between the rising edge of TI2 and the start of the counter depends on the resynchronization circuit at the input of TI2.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

262/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 285** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

991811

34

35

36

37

38

TI2

CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

162.

**2 +**

External clock mode 2 can be used with another slave mode (except external clock mode 1 and encoder mode). This

, The ETR signal is used as the input of the external clock, and another one can be selected in reset mode, gating mode or trigger mode

Inputs are used as trigger inputs. It is not recommended to use the TS bit of the TIMx\_SMCR register to select ETR as TRGI.

In the following example, when a rising edge occurs on TI1, the counter counts up on each rising edge of ETR

once:

• Configure the external trigger input circuit through the TIMx\_SMCR register:

**–** ETF = 0000: no filtering

**–** ETPS = 00: without prescaler

**–** ETP = 0: detect the rising edge of ETR, set ECE = 1 to enable external clock mode 2

• Configure channel 1 as follows to detect the rising edge of TI:

**–** IC1F = 0000: no filtering

**– The** capture prescaler is not used in the trigger operation and no configuration is required

**–** Set CC1S = 01 in the TIMx\_CCMR1 register to select the input capture source

**–** Set CC1P = 0 in the TIMx\_CCER register to determine the polarity (only the rising edge is detected)

• Set SMS = 110 in the TIMx\_SMCR register to configure the timer in trigger mode. Set TIMx\_SMCR register

TS = 101, select TI1 as the input source.

When a rising edge occurs on TI1, the TIF flag is set and the counter starts counting on the rising edge of ETR.

The delay between the rising edge of the ETR signal and the actual reset of the counter depends on the resynchronization circuit at the input of ETRP.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

263/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 286** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

262859

34

35

36

TI1

CEN/CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

TIF

ETR

163.

2 +

**12.3.15**

All TIMx timers are connected internally for timer synchronization or linking. When a timer is in main mode, it

You can reset, start, stop, or provide a clock to the counter of another timer in slave mode.

The figure below shows an overview of the trigger selection and main mode selection modules.

use

115977

Main mode

control

clock

MMS

UEV

counter

Prescaler

Slave mode

control

SMS

counter

Prescaler

CK \_ PSC

ITR 0

TS

TRGO 1

TIMER 1

TIMER 2

Input trigger selection

164.

/

For example, Timer 1 can be configured as Timer 2 prescaler. Refer to the figure above and do the following:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

264/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 287** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

• Configure Timer 1 as the main mode, which can output a periodic trigger signal at each update event UEV.

When MMS = '010' in the TIM1\_CR2 register, input on TRGO1 whenever an update event occurs

There is a rising edge signal.

• Connect TRGO1 of timer 1 to timer 2 and set TS = '000' in TIM2\_SMCR register,

Configure Timer 2 to use ITR1 as the internally triggered slave mode.

• Then put the slave mode controller in external clock mode 1 (SMS = 111 in TIM2\_SMCR register);

Timer 2 can be driven by the periodic rising edge of Timer 1 (that is, the timer 1 counter overflows).

• Finally, the corresponding (TIMx\_CR1 register) CEN bit must be set to start the two timers respectively.

Note: If OCx has been selected as the trigger output of Timer 1 (MMS = 1xx), its rising edge is used to drive the count of Timer 2

Device.

use

The this Example the In, The Operation of the Timer 2 The IS Output Controlled by the Timer 1. Comparison of the Refer to Figure [165](https://translate.googleusercontent.com/translate_f#287) . The Timer only

When OC1REF of 1 is high, Timer 2 counts the divided internal clock. The clock frequency of both timers is

It is obtained by dividing CK\_INT by 3 (f CK\_CNT = f CK\_INT /3) by the prescaler .

• Configure Timer 1 as the main mode and send its output comparison reference signal (OC1REF) as the trigger output (TIM1\_CR2

Register MMS = 100)

• Configure the OC1REF waveform of timer 1 (TIM1\_CCMR1 register)

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 001 in TIM2\_SMCR register)

• Configure Timer 2 for gating mode (SMS = 101 in TIM2\_SMCR register)

• Set CEN = 1 in the TIM2\_CR1 register to enable timer 2

• Set CEN = 1 in the TIM1\_CR1 register to enable timer 1

Note: The timer 2 clock is not synchronized with the timer 1 clock, this mode only affects the timer 2 counter enable signal.

949443

FE

00

01

FC

3046

3048

3045

3047

FD

FF

CK\_INT

TIMER1-OC1REF

TIMER1-CNT

TIMER2-CNT

TIMER2-TIF

Write TIF = 0

165.

1

OC1REF

2

In the example above, their counters and prescalers were not initialized before Timer 2 started, so they

Start counting from the current value. It is possible to reset 2 timers before starting timer 1, so that they are from the given value

Start, that is, write any desired value in the timer counter. Reset by writing the UG bit in the TIMx\_EGR register

Timer.

In the next example, Timer 1 and Timer 2 need to be synchronized. Timer 1 is the main mode and starts from 0, the timer

2 is the slave mode and starts from 0xE7; the prescaler coefficients of the two timers are the same. Write 0 to CEN of TIM1\_CR1

The bit will disable Timer 1, and Timer 2 will stop immediately.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

265/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 288** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

• Configure Timer 1 as the main mode and send the output compare 1 reference signal (OC1REF) as the trigger output (TIM1\_CR2

Register MMS = 100).

• Configure the OC1REF waveform of timer 1 (TIM1\_CCMR1 register).

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 for gating mode (SMS = 101 in TIM2\_SMCR register)

• Set UG = 1 in the TIM1\_EGR register to reset timer 1.

• Set UG = 1 in the TIM2\_EGR register to reset timer 2.

• Write 0xE7 to the timer 2 counter (TIM2\_CNT) and initialize it to 0xE7.

• Set CEN = 1 in the TIM2\_CR1 register to enable timer 2.

• Set CEN = 1 in the TIM1\_CR1 register to start Timer 1.

• Set CEN = 0 in the TIM1\_CR1 register to stop timer 1.

381843

00

01

02

75

00

E8

E9

AB

E7

CK\_INT

TIMER1-CEN = CNT\_EN

TIMER1-CNT\_INIT

TIMER1-CNT

TIMER2-CNT

TIMER2-CNT\_INIT

TIMER2 write CNT

TIMER2-TIF

Write TIF = 0

166.

1

2

use

The this Example the In, The Update Timer Event. 1 of Timer 2. IS Used to enable the Refer to Figure [167 is](https://translate.googleusercontent.com/translate_f#289) . The When the Timer. 1 IS Updated

When an event occurs, Timer 2 starts counting from its current value (which can be non-zero) according to the divided internal clock. After receiving the touch

When signaled, the CEN bit of timer 2 is automatically set to 1, and the counter starts counting until 0 is written to TIM2\_CR1

The CEN bit of the register. The clock frequency of both timers is divided by the prescaler CK\_INT by 3 (f CK\_CNT =

f CK\_INT /3).

• Configure Timer 1 as the main mode and send its update event (UEV) as the trigger output (TIM1\_CR2 register)

MMS = 010).

• Configure the period of Timer 1 (TIM1\_ARR register).

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 as trigger mode (SMS = 110 in TIM2\_SMCR register)

• Set CEN = 1 in the TIM1\_CR1 register to start Timer 1.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

266/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 289** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

227169

FF

01

02

FD

46

48

45

47

Write TIF = 0

FE

00

CK\_INT

TIMER1\_UEV

TIMER1\_CNT

TIMER2\_CNT

TIMER2\_CEN = CNT\_EN

TIMER2\_TIF

167.

Use 1

2

In the previous example, the two counters can be initialized before counting is started. The following figure shows the same configuration as 0

Next, use the trigger mode instead of the gating mode (SMS = 110 in the TIM2\_SMCR register).

449102

00

01

02

75

00

E8

E9

CD

EA

E7

CK\_INT

TIMER1-CEN = CNT\_EN

TIMER1-CNT\_INIT

TIMER1-CNT

TIMER2-CNT

TIMER2-CNT\_INIT

TIMER2-CNT\_INIT

TIMER2-TIF

Write TIF = 0

168.

Use 1

2

use

This example uses Timer 1 as the prescaler for Timer 2. The configuration is as follows:

• Configure Timer 1 as the main mode and send its update event UEV as the trigger output (TIM1\_CR2 register

MMS = '010'). Then every time the counter overflows, it outputs a periodic signal.

• Configure the period of Timer 1 (TIM1\_ARR register).

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 to use external clock mode (SMS = 111 in TIM2\_SMCR register)

• Set CEN = 1 in the TIM1\_CR2 register to start timer 2

• Set CEN = 1 in the TIM1\_CR1 register to start timer 1

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

267/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 290** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

Use **2**

In this example, when the TI1 input of timer 1 rises, timer 1 is enabled and timer 1 is enabled, and timer 2 is enabled. for

To ensure the alignment of the counter, Timer 1 must be configured in master/slave mode (corresponding to TI1 being a slave, and corresponding to Timer 2 being a master):

• Configure Timer 1 as the main mode and send its enable as a trigger output (MMS='001' in the TIM1\_CR2 register

)

• Configure Timer 1 in Slave mode and obtain input trigger from TI1 (TS = '100' in TIM1\_SMCR register)

• Configure Timer 1 as the trigger mode (SMS='110' in the TIM1\_SMCR register)

• Configure Timer 2 to obtain input trigger from Timer 1 (TS = 000 in TIM2\_SMCR register)

• Configure Timer 2 as trigger mode (SMS = 110 in TIM2\_SMCR register)

When a rising edge appears on TI1 of timer 1, the two timers start counting according to the internal clock synchronously.

The TIF flag is also set at the same time.

Note: In this example, both timers are initialized (set the corresponding UG bit) before starting, and both counters start from 0

At the beginning, you can insert an offset between timers by writing to any counter register (TIMx\_CNT). You can see in the picture below

There is a delay between CNT\_EN and CK\_PSC of Timer 1 in master/slave mode.

753049

00

01

07

02 03 04 05 06

08 09

00

01

07

02 03 04 05 06

08 09

CK\_INT

TIMER 1-TI1

TIMER 1-CEN = CNT\_EN

TIMER 1-CK\_PSC

TIMER1-CNT

TIMER1-TIF

TIMER2-CEN = CNT\_EN

TIMER2-CK\_PSC

TIMER2-CNT

TIMER2-TIF

169.

Use 1

TI1

1

2

**12.3.16**

When the microcontroller enters debug mode (CPU core is stopped), according to the setting of DBG\_TIMx\_STOP in the DBG module,

The TIMx counter either continues to operate normally or stops. See the debugging module chapter for details.

**12.4 TIMx**

These peripheral registers can be operated in halfword (16-bit) or word (32-bit) mode.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

268/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 291** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

44. TIMx

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

TIMx\_CR1

Control register 1

0x00000000

[Section 12.4.1](https://translate.googleusercontent.com/translate_f#291)

0x04

TIMx\_CR2

Control register 2

0x00000000

[Section 12.4.2](https://translate.googleusercontent.com/translate_f#293)

0x08

TIMx\_SMCR

Slave mode control register

0x00000000

[Section 12.4.3](https://translate.googleusercontent.com/translate_f#294)

0x0C

TIMx\_DIER

DMA/interrupt enable register

0x00000000

[Section 12.4.4](https://translate.googleusercontent.com/translate_f#297)

0x10

TIMx\_SR

Status register

0x00000000

[Section 12.4.5](https://translate.googleusercontent.com/translate_f#299)

0x14

TIMx\_EGR

Event generation register

0x00000000

[Section 12.4.6](https://translate.googleusercontent.com/translate_f#300)

0x18

TIMx\_CCMR1

Capture/Compare Mode Register 1

0x00000000

[Section 12.4.7](https://translate.googleusercontent.com/translate_f#301)

0x1C

TIMx\_CCMR2

Capture/Compare Mode Register 2

0x00000000

[Section 12.4.8](https://translate.googleusercontent.com/translate_f#305)

0x20

TIMx\_CCER

Capture/compare enable register

0x00000000

[Section 12.4.9](https://translate.googleusercontent.com/translate_f#307)

0x24

TIMx\_CNT

counter

0x00000000

[Section 12.4.10](https://translate.googleusercontent.com/translate_f#308)

0x28

TIMx\_PSC

Prescaler

0x00000000

[Section 12.4.11](https://translate.googleusercontent.com/translate_f#309)

0x2C

TIMx\_ARR

Autoload register

0x00000000

[Section 12.4.12](https://translate.googleusercontent.com/translate_f#309)

0x34

TIMx\_CCR1

Capture/Compare Register 1

0x00000000

[Section 12.4.13](https://translate.googleusercontent.com/translate_f#310)

0x38

TIMx\_CCR2

Capture/Compare Register 2

0x00000000

[Section 12.4.14](https://translate.googleusercontent.com/translate_f#310)

0x3C

TIMx\_CCR3

Capture/Compare Register 3

0x00000000

[Section 12.4.15](https://translate.googleusercontent.com/translate_f#311)

0x40

TIMx\_CCR4

Capture/Compare Register 4

0x00000000

[Section 12.4.16](https://translate.googleusercontent.com/translate_f#311)

0x48

TIMx\_DCR

DMA control register

0x00000000

[Section 12.4.17](https://translate.googleusercontent.com/translate_f#312)

0x4C

TIMx\_DMAR

Continuous mode DMA address

0x00000000

[Section 12.4.18](https://translate.googleusercontent.com/translate_f#313)

**12.4.1**

**1(TIMx\_CR1)**

Offset address: 0x00

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CKD

ARPE

CMS

DIR

OPM

URS UDIS CEN

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

9:8

CKD

rw

0x00

Clock division

These 2 bits are defined in the timer clock (CK\_INT) frequency, dead time and by

Between the dead zone generator and the sampling clock used by the digital filter (ETR, TIx)

The frequency division ratio.

00: t DTS = t CK\_INT

01: t DTS = 2 xt CK\_INT

10: t DTS = 4 xt CK\_INT

11: Reserved, do not use this configuration

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

269/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 292** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7

ARPE

rw

0x00

Auto-reload preload enable bit (Auto-reload preload enable)

0: TIMx\_ARR register is not buffered

1: TIMx\_ARR register is loaded into the buffer

6:5

CMS

rw

0x00

Select Center-aligned mode selection

00: Edge alignment mode. Counter up or down according to the direction bit (DIR)

count

01: Center alignment mode 1. The counter alternately counts up and down. Configuration

Is the output of the output channel (CCxS = 00 in the TIMx\_CCMRx register)

The compare interrupt flag bit is only set when the counter is counting down

10: Center alignment mode 2. The counter alternately counts up and down. counter

Count up and down alternately. Channels configured as outputs (TIMx\_CCMRx

The CCxS = 00) output compare interrupt flag bit in the register, only in the counter

Set when counting up

11: Center alignment mode 3. The counter alternately counts up and down. counter

Count up and down alternately. Channels configured as outputs (TIMx\_CCMRx

The CCxS = 00) output compare interrupt flag bit in the register

Set when counting up and down

Note: When the counter is turned on (CEN = 1), it is not allowed to switch from edge-aligned mode

Switch to center alignment mode.

4

DIR

rw

0x00

Direction

0: the counter counts up

1: The counter counts down

Note: When the counter is configured for center-aligned mode or encoder mode, this bit is

Read only.

3

OPM

rw

0x00

One pulse mode (One pulse mode) 0: When an update event occurs, the counter

Counter does not stop

1: When the next update event occurs (clears the CEN bit), the counter stops

2

URS

rw

0x00

Update request source (Update request source) software selects through this bit

The source of the UEV event.

0: If update interrupt or DMA request is allowed, any one of the following events

An update interrupt or DMA request is generated:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller

1: If an update interrupt or DMA request is allowed, only the counter overflows

An update interrupt or DMA request is generated only after the overflow/underflow

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

270/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 293** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

UDIS

rw

0x00

Disable disable (Update disable) Software enables/disables UEV through this bit

Event generation

0: UEV is allowed. Update (UEV) events are generated by any of the following events:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller are cached registers are loaded into their

Preload value.

1: UEV is prohibited. No update event occurs, shadow registers (ARR, PSC,

CCRx) keep their values. If the UG bit or slave mode controller is set

A hardware reset is issued and the counter and prescaler are reinitialized

0

CEN

rw

0x00

Counter enable

0: disable counter

1: Enable the counter.

Note: After the CEN bit is set by the software, the external clock, gating mode and encoder mode

To work. The trigger mode can automatically set the CEN bit through hardware.

**12.4.2**

**2(TIMx\_CR2)**

Offset address: 0x04

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

Reserved

TI1S

MMS

CCDS

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 8

Reserved

Reserved, read as 0

7

TI1S

rw

0x00

TI1 selection (TI1 selection)

0: TIMx\_CH1 pin is connected to TI1 input

1: TIMx\_CH1, TIMx\_CH2 and TIMx\_CH3 pins are connected through XOR

To TI1 input

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

271/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 294** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6:4

MMS

rw

0x00

Master mode selection

These two bits are used to select the synchronization information sent to the slave timer in master mode

(TRGO). The possible combinations are as follows:

000: Reset – the UG bit in the TIMx\_EGR register is used as a trigger input

Out (TRGO). If trigger input (slave mode controller is in reset mode)

Reset occurs, the signal on TRGO will have a delay relative to the actual reset

late.

001: Enable – The counter enable signal CNT\_EN is used as a trigger input

Out (TRGO). Sometimes it is necessary to start multiple timers or controls at the same time

Enable the slave timer for a period of time. Counter enable signal is through CEN

Logic OR generation of the trigger input signal in control bit and gating mode. Count

When the counter enable signal is controlled by the trigger input, there will be a delay on TRGO,

Unless master/slave mode is selected (see the MSM bit in the TIMx\_SMCR register

description of).

010: Update-The update event is selected as the trigger input (TRGO). For example, one

The clock of each master timer can be used as a prescaler for the slave timer.

011: Comparison pulse-when a capture occurs or a comparison is successful, when setting

When the CC1IF flag is set (even if it is already high), the trigger output sends a positive

Pulse (TRGO).

100: Compare – OC1REF signal is used as a trigger output (TRGO)

101: Compare – OC2REF signal is used as a trigger output (TRGO)

110: Compare – OC3REF signal is used as a trigger output (TRGO)

111: Compare – OC4REF signal is used as a trigger output (TRGO)

3

CCDS

rw

0x00

Capture/compare DMA selection

0: Send CCx DMA request when CCx event occurs

1: When an update event occurs, the CCx DMA request is sent

2: 0

Reserved

Reserved, read as 0

**12.4.3**

**(TIMx\_SMCR)**

Offset address: 0x08

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

ETP

ECE

ETPS

ETF

MSM

TS

Res.

SMS

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

ETP

rw

0x00

External trigger polarity

This bit selects whether to use ETR or the inverse of ETR as the trigger operation.

0: ETR is not inverted, high level or rising edge is valid

1: ETR is inverted, low level or falling edge is valid

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

272/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 295** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

14

ECE

rw

0x00

External clock enable (External clock enable)

This bit enables external clock mode 2.

0: Disable external clock mode 2

1: Enable external clock mode 2, the counter is valid by any on the ETRF signal

Rising edge drive

Note 1: Set ECE bit and select external clock mode 1 and connect TRGI to

ETRF (SMS = 111 and TS = 111) have the same effect.

Note 2: The following slave modes can be used simultaneously with external clock mode 2: reset mode, gate

Control mode and trigger mode; however, TRGI cannot be connected to ETRF (TS bit does not

Can be 111).

Note 3: When external clock mode 1 and external clock mode 2 are enabled at the same time, the external clock

The input of the clock is ETRF.

13:12

ETPS

rw

0x00

External trigger prescaler

The frequency of the external trigger signal ETRP must be at most the frequency of TIMxCLK

1/4. When inputting a faster external clock, you can use prescaler to reduce ETRP

Frequency of.

00: turn off prescaler

01: ETRP frequency divided by 2

10: ETRP frequency divided by 4

11: ETRP frequency divided by 8

11:8

ETF

rw

0x00

External trigger filter

These bits define the sampling frequency of ETRP signal and the digital filtering of ETRP

The bandwidth of the wave. In fact, the digital filter is an event counter, which records

After N events, an output transition will occur.

0000: no filter, sampling with f DTS

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

0110: Sampling frequency f SAMPLING = f DTS /4, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

273/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 296** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7

MSM

rw

0x00

Master/slave mode

0: No effect

1: The event on the trigger input (TRGI) is delayed to allow the current setting

The perfect synchronization between the timer (via TRGO) and its slave timer, this is necessary

It is very useful when you want to synchronize several timers to a single external event

of

6:4

TS

rw

0x00

Trigger selection

These 3 bits select the trigger input for synchronization counter.

000: Internal trigger 0 (ITR0)

001: Internal trigger 1 (ITR1)

010: Internal trigger 2 (ITR2)

011: Internal trigger 3 (ITR3)

100: TI1's edge detector (TI1F\_ED)

101: Filtered timer input 1 (TI1FP1)

110: Filtered timer input 2 (TI2FP2)

111: External trigger input (ETRF)

For more details about ITRx, see the table below.

Note: These bits can only be changed when not used (eg SMS = 000), to avoid changing

Wrong edge detection when changing time.

3

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

274/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 297** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2: 0

SMS

rw

0x00

Slave mode selection

When the external signal is selected, the effective edge of the trigger signal (TRGI) and the selected

External input polarity is related (see input control register and control register

Bright)

000: Slave mode off-if CEN = 1, the prescaler is directly internally

Clock driven.

001: Encoder mode 1-according to the level of TI1FP1, the counter is in TI2FP2

Counts up/down.

010: Encoder mode 2-according to the level of TI2FP2, the counter is in TI1FP1

Counts up/down.

011: Encoder mode 3-according to the level of another input, the counter is

The edges of TI1FP1 and TI2FP2 count up/down.

100: Reset mode-the rising edge of the selected trigger input (TRGI) is reset

Initialize the counter and generate a signal to update the register.

101: Gating mode-when the trigger input (TRGI) is high, the time of the counter

The bell opens. When the trigger input goes low, the counter stops (but does not reset). meter

The start and stop of the counter are controlled.

110: Trigger mode-the counter starts on the rising edge of the trigger input TRGI (but

No reset), only the start of the counter is controlled.

111: External clock mode 1-rising edge of the selected trigger input (TRGI)

Drive counter.

Note: If TI1F\_EN is selected as the trigger input (TS = 100), do not use the gate

Control mode. This is because TI1F\_ED outputs a pulse every time TI1F changes,

However, the gating mode is to check the level of the trigger input.

45. TIMx

Slave timer

**ITR0 (TS = 000)**

**ITR1 (TS = 001)**

**ITR2 (TS = 010)**

**ITR3 (TS = 011)**

TIM2

TIM1

no

TIM3

no

**12.4.4 DMA/**

**(TIMx\_DIER)**

Offset address: 0x0C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Res. TDE Res. CC4DE

UDE

Res.

TIE

Res. CC4IE

UIE

CC3DE CC2DECC1DE

CC3IE CC2IE CC1IE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

275/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 298** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

14

TDE

rw

0x00

Allow trigger DMA request (Trigger DMA request enable)

0: Disable triggering of DMA request

1: Allow to trigger DMA request

13

Reserved

Reserved, read as 0

12

CC4DE

rw

0x00

Allow Capture/Compare 4 DMA request (Capture/Compare 4 DMA

request enable)

0: Disable capture/compare 4 DMA request

1: Allow capture/compare 4 DMA request

11

CC3DE

rw

0x00

Allow Capture/Compare 3 DMA request (Capture/Compare 3 DMA

request enable)

0: Disable capture/compare 3 DMA request

1: Allow capture/compare 3 DMA request

10

CC2DE

rw

0x00

Allow Capture/Compare 2 DMA request (Capture/Compare 2 DMA

request enable)

0: Disable capture/compare 2 DMA request

1: Allow capture/compare 2 DMA request

9

CC1DE

rw

0x00

Allow Capture/Compare 1 DMA request (Capture/Compare 1 DMA

request enable)

0: Disable capture/compare 1 DMA request

1: Allow capture/compare 1 DMA request

8

UDE

rw

0x00

Update DMA request enable

0: Disable update DMA request

1: Allow updated DMA requests

7

Reserved

Reserved, read as 0

6

TIE

rw

0x00

Trigger interrupt enable

0: Disable trigger interrupt

1: Enable trigger interrupt

5

Reserved

Reserved, read as 0

4

CC4IE

rw

0x00

Allow Capture/Compare 4 interrupt en-

able)

0: Disable capture/compare 4 interrupt

1: Enable capture/compare 4 interrupt

3

CC3IE

rw

0x00

Allow Capture/Compare 3 interrupt en-

able)

0: Disable capture/compare 3 interrupt

1: Allow capture/compare 3 interrupt

2

CC2IE

rw

0x00

Allow Capture/Compare 2 interrupt en-

able)

0: Disable capture/compare 2 interrupt

1: Enable capture/compare 2 interrupt

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

276/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 299** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

CC1IE

rw

0x00

Allow Capture/Compare 1 interrupt en-

able)

0: Disable capture/compare 1 interrupt

1: Enable capture/compare 1 interrupt

0

UIE

rw

0x00

Update interrupt enable

0: Disable update interrupt

1: Allow update interrupt

**12.4.5**

**(TIMx\_SR)**

Offset address: 0x10

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rc\_w0

rc\_w0

rc\_w0

rc\_w0

Res.

CC4OF

Res.

TIF

Res. CC4IF

UIF

CC3OF CC2OF CC1OF

CC3IF CC2IF CC1IF

rc\_w0 rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, read as 0

12

CC4OF

rc\_w0

0x00

Capture/Compare 4 overcap-capture (Capture/Compare 4 overcap-

ture flag)

See CC1OF description.

11

CC3OF

rc\_w0

0x00

Capture/Compare 3 overcap-capture (Capture/Compare 3 overcap-

ture flag)

See CC1OF description.

10

CC2OF

rc\_w0

0x00

Capture/Compare 2 overcap-capture (Capture/Compare 2 overcap-

ture flag)

See CC1OF description.

9

CC1OF

rc\_w0

0x00

Capture/Compare 1 overcap-capture (Capture/Compare 1 overcap-

ture flag)

This flag can be set by hardware only when the corresponding channel is configured for input capture

1. Write 0 to clear this bit.

0: No repeated capture;

1: When the value of the counter is captured in the TIMx\_CCR1 register, the CC1IF

The status is already 1.

8: 7

Reserved

Reserved, read as 0

6

TIF

rc\_w0

0x00

Trigger interrupt flag

When a trigger event occurs (when the slave mode controller is in a mode other than gating mode)

In other modes, a valid edge is detected at the TRGI input, or gated mode

Any edge under the formula) is set to 1 by hardware. It is cleared by software.

0: No trigger event is generated

1: Trigger interrupt waiting for response

5

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

277/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 300** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

4

CC4IF

rc\_w0

0x00

Capture/Compare 4 interrupt flag

Refer to CC1IF description.

3

CC3IF

rc\_w0

0x00

Capture/Compare 3 interrupt flag

Refer to CC1IF description.

2

CC2IF

rc\_w0

0x00

Capture/Compare 2 interrupt flag

Refer to CC1IF description.

1

CC1IF

rc\_w0

0x00

Capture/Compare 1 interrupt flag

If channel CC1 is configured in output mode:

When the counter value matches the comparison value, the bit is set to '1' by the hardware, but in the center

Except in symmetric mode (refer to the CMS bit of the TIMx\_CR1 register). it

Cleared by software to '0'.

0: No match occurred

1: The value of TIMx\_CNT matches the value of TIMx\_CCR1

If channel CC1 is configured for input mode:

When the capture event occurs, this bit is set by hardware, it is cleared by software or through

Clear '0' after reading TIMx\_CCR1.

0: No input capture

1: The counter value has been captured (copied) to TIMx\_CCR1 (checked on IC1

An edge with the same polarity as the selected one is measured)

0

UIF

rc\_w0

0x00

Update interrupt flag

This bit is set to '1' by hardware when an update event occurs. It is cleared to '0' by the software.

0: No update event is generated

1: Update event is waiting for response. When the register is updated, this bit is set to '1' by hardware:

-If UDIS = 0 in the TIMx\_CR1 register, when REP\_CNT = 0

An update event is generated (when the down counter overflows or underflows repeatedly)

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, when TIMx\_EGR

An update event occurs when UG = 1 in the register (software resets counter CNT

New initialization)

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, the counter

An update event is generated when CNT is triggered by event re-initialization. (Refer to synchronous control

(Description of register)

**12.4.6**

**(TIMx\_EGR)**

Offset address: 0x14

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

w

w

w

w

TG

Res. CC4G CC3G CC2G CC1G

UG

w

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:7

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

278/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 301** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6

TG

w

0x00

Trigger generation

This bit is set to '1' by the software and is used to generate a braking event, automatically by the hardware

Clear '0'.

0: No action

1: TIF = 1 in the TIMx\_SR register, if the corresponding interrupt and DMA are enabled,

Then generate the corresponding interrupt and DMA

5

Reserved

Reserved, read as 0

4

CC4G

w

0x00

Generate Capture/Compare 4 generation

Refer to CC1G description.

3

CC3G

w

0x00

Generate Capture/Compare 3 generation

Refer to CC1G description.

2

CC2G

w

0x00

Generate Capture/Compare 2 generation

Refer to CC1G description.

1

CC1G

w

0x00

Generate Capture/Compare 1 generation

This bit is set by software to generate a capture/compare event.

Reset to 0.

0: No action

1: Generate a capture/compare event on channel CC1:

If channel CC1 is configured as an output:

Set CC1IF=1, if the corresponding interrupt and DMA are turned on, the corresponding

Interrupt and DMA.

If channel CC1 is configured as input:

The current counter value is captured into the TIMx\_CCR1 register and set

CC1IF = 1, if the corresponding interrupt and DMA are turned on, the corresponding

Off and DMA. If CC1IF is already 1, set CC1OF = 1.

0

UG

w

0x00

Generate an update event (Update generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: Re-initialize the counter and generate an update event. Note the prescaler

The counter of the counter is also cleared to '0' (but the prescaler coefficient is unchanged). If in the center

In symmetric mode or DIR = 0 (count up), the counter is cleared to '0'; if

DIR = 1 (count down), the counter takes the value of TIMx\_ARR.

**12.4.7**

**/**

**1(TIMx\_CCMR1)**

Offset address: 0x18

Reset value: 0x0000

The channel can be used for input (capture mode) or output (comparison mode), and the direction of the channel is defined by the corresponding CCxS. The deposit

The function of other bits of the device is different from the output mode. OCxx describes the function of the channel in output mode, ICxx describes

The function of the channel in output mode. Therefore, it must be noted that the function of the same bit in output mode and input mode is not

Same.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

279/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 302** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OC2M

OC2PE OC2FE

CC2S

OC1CE

OC1M

IC1F

OC1PE OC1FE

IC1PSC

CC1S

IC2PSC

IC2F

OC2CE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

OC2CE

rw

0x00

Output compare 2 clear enable

14:12

OC2M

rw

0x00

Output compare 2 mode

11

OC2PE

rw

0x00

Output compare 2 preload en-

able)

10

OC2FE

rw

0x00

Output compare 4 fast enable (Output compare 4 fast enable)

9:8

CC2S

rw

0x00

Capture/Compare 2 selection

This bit defines the direction of the channel (input/output) and the selection of input pins:

00: CC2 channel is configured as output

01: CC2 channel is configured as input, IC2 is mapped on TI2

10: CC2 channel is configured as input, IC2 is mapped on TI1

11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC2S is only when the channel is closed (CC2E = 0 in the TIMx\_CCER register)

Is writable.

7

OC1CE

rw

0x00

Output compare 1 clear enable

0: OC1REF is not affected by ETRF input

1: When high level of ETRF input is detected, clear OC1REF = 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

280/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 303** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

6:4

OC1M

rw

0x00

Output compare 1 mode

These 3 bits define the action of the output reference signal OC1REF, and OC1REF

Decide the values ​​​​of OC1 and OC1N. OC1REF is active high, and

The effective levels of OC1 and OC1N depend on the CC1P and CC1NP bits.

000: freeze. Output compare register TIMx\_CCR1 and counter

Comparison between TIMx\_CNT has no effect on OC1REF

001: Set channel 1 to effective level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is high

010: Set channel 1 to invalid level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is low

011: Flip. When TIMx\_CCR1=TIMx\_CNT, flip OC1REF

Level

100: Forced to invalid level. Force OC1REF low

101: Forced to effective level. Force OC1REF high

110: PWM mode 1-When counting up, when TIMx\_CNT <

When TIMx\_CCR1, channel 1 is active level, otherwise it is inactive level;

When counting down, when TIMx\_CNT> TIMx\_CCR1, channel 1 is none

Effective level (OC1REF = 0), otherwise the effective level (OC1REF = 1)

111: PWM mode 2-When counting up, when TIMx\_CNT <

When TIMx\_CCR1, channel 1 is invalid level, otherwise it is valid level;

When counting down, when TIMx\_CNT> TIMx\_CCR1, channel 1 is

Effective level, otherwise invalid level

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = 00 (the channel is configured as an output), this bit cannot be modified. Note

2: In PWM mode 1 or PWM mode 2, only when the comparison result changes

Or when switching from freeze mode to PWM mode in output compare mode, OC1REF

The level changes.

3

OC1PE

rw

0x00

Output compare 1 preload en-

able)

0: Disable the preload function of the TIMx\_CCR1 register, which can be written at any time

TIMx\_CCR1 register, and the newly written value takes effect immediately

1: Turn on the preload function of the TIMx\_CCR1 register.

Preload register operation, the preload value of TIMx\_CCR1 is updated to

Is loaded into the current register when it comes

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = 00 (the channel is configured as an output), this bit cannot be modified. Note 2:

In single pulse mode only (OPM = 1 in the TIMx\_CR1 register), you can

The PWM mode is used when the preload register is recognized, otherwise its operation is undefined.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

281/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 304** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

OC1FE

rw

0x00

Output compare 1 fast enable (Output compare 1 fast enable)

This bit is used to speed up the response of the CC output to trigger input events.

0: CC1 operates normally according to the value of the counter and CCR1, even if it is triggered

The device is turned on. When the trigger input has a valid edge, activate CC1

The minimum output delay is 5 clock cycles

1: The valid edge input to the trigger acts as if a comparison match has occurred.

Therefore, OC is set to the comparison level regardless of the comparison result. Sample trigger

The delay between the effective edge of the device and the output of CC1 is reduced to 3 clock cycles.

OCFE only works when the channel is configured in PWM1 or PWM2 mode

1:0

CC1S

rw

0x00

Capture/Compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: CC1 channel is configured as input, IC1 is mapped on TI2

11: CC1 channel is configured as input, IC1 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIMx\_CCER register)

Is writable.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:12

IC2F

rw

0x00

Input capture 2 filter

11:10

IC2PSC

rw

0x00

Input capture 2 prescaler

9:8

CC2S

rw

0x00

Capture/Compare 2 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC2 channel is configured as output

01: CC2 channel is configured as input, IC2 is mapped on TI2

10: CC2 channel is configured as input, IC2 is mapped on TI1

11: CC2 channel is configured as input, IC2 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC2S is only when the channel is closed (CC2E = 0 in the TIMx\_CCER register)

Is writable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

282/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 305** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7:4

IC1F

rw

0x00

Input capture 1 filter

These bits define the sampling frequency and digital filter length of the TI1 input. number

The word filter consists of an event counter, which will record after N events

Produces an output transition:

0000: no filter, sampling with fDTS

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

0110: Sampling frequency f SAMPLING = f DTS /4, N = 6

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

3:2

IC1PSC

rw

0x00

Input capture 1 prescaler

These 2 bits define the prescaler coefficient of the CC1 input (IC1).

When CC1E = 0 (in the TIMx\_CCER register), the prescaler is reset.

00: No prescaler, every edge detected on the capture input port is touched

Send a capture

01: Capture is triggered every 2 events

10: Capture is triggered every 4 events

11: Trigger every 8 events

1:0

CC1S

rw

0x00

Capture/compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: CC1 channel is configured as input, IC1 is mapped on TI2

11: CC1 channel is configured as input, IC1 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIMx\_CCER register)

Is writable.

**12.4.8**

**/**

**2(TIMx\_CCMR2)**

Offset address: 0x1C

Reset value: 0x0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

283/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 306** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

See the description of the CCMR1 register above

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

OC4M

OC4PE OC4FE

CC4S

OC3CE

OC3M

IC3F

OC3PE OC3FE

IC3PSC

CC3S

IC4PSC

IC4F

OC4CE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

OC4CE

rw

0x00

Output compare 4 clear enable

14:12

OC4M

rw

0x00

Output compare 4 mode (Output compare 4 mode)

11

OC4PE

rw

0x00

Output compare 4 preload en-

able)

10

OC4FE

rw

0x00

Output compare 4 fast enable (Output compare 4 fast enable)

9:8

CC4S

rw

0x00

Capture/Compare 4 selection

The 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC4 channel is configured as output

01: CC4 channel is configured as input, IC4 is mapped on TI4

10: CC4 channel is configured as input, IC4 is mapped on TI3

11: CC4 channel is configured as input, IC4 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC4S is only when the channel is closed (CC4E = 0 in the TIMx\_CCER register)

Is writable.

7

OC3CE

rw

0x00

Output compare 3 clear '0' enable (Output compare 3 clear enable)

6:4

OC3M

rw

0x00

Output compare 3 mode

3

OC3PE

rw

0x00

Output compare 3 preload en-

able)

2

OC3FE

rw

0x00

Output compare 3 fast enable (Output compare 3 fast enable)

1:0

CC3S

rw

0x00

Capture/Compare 3 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC3 channel is configured as output

01: CC3 channel is configured as input, IC3 is mapped on TI3

10: CC3 channel is configured as input, IC3 is mapped on TI4

11: CC3 channel is configured as input, IC3 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC3S is only when the channel is closed (CC3E = 0 in the TIMx\_CCER register)

Is writable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

284/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 307** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:12

IC4F

rw

0x00

Input capture 4 filter

11:10

IC4PSC

rw

0x00

Input capture 4 prescaler

9:8

CC4S

rw

0x00

Capture/Compare 4 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC4 channel is configured as output

01: CC4 channel is configured as input, IC4 is mapped on TI4

10: CC4 channel is configured as input, IC4 is mapped on TI3

11: CC4 channel is configured as input, IC4 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC4S is only when the channel is closed (CC4E = 0 in the TIMx\_CCER register)

Is writable.

7:4

IC3F

rw

0x00

Input capture 3 filter

3:2

IC3PSC

rw

0x00

Input capture 3 prescaler

1:0

CC3S

rw

0x00

Capture/compare 3 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC3 channel is configured as output

01: CC3 channel is configured as input, IC3 is mapped on TI3

10: CC3 channel is configured as input, IC3 is mapped on TI4

11: CC3 channel is configured as input, IC3 is mapped on TRC, this mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

(TS bit selection of the device)

Note: CC3S is only when the channel is closed (CC3E = 0 in the TIMx\_CCER register)

Is writable.

**12.4.9**

**/**

**(TIMx\_CCER)**

Offset address: 0x20

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

Res.

CC4P

Res.

Res.

CC4E

CC3P CC3E

CC2P CC2E

CC1P CC1E

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:14

Reserved

Reserved, always read as 0

13

CC4P

rw

0x00

Capture/Compare 4 output polarity

Refer to the description of CC1P.

12

CC4E

rw

0x00

Capture/Compare 4 output enable

Refer to the description of CC1E.

11:10

Reserved

Reserved, always read as 0

9

CC3P

rw

0x00

Capture/Compare 3 output polarity

Refer to the description of CC1P.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

285/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 308** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

8

CC3E

rw

0x00

Capture/Compare 3 output enable

Refer to the description of CC1E.

7:6

Reserved

Reserved, always read as 0

5

CC2P

rw

0x00

Capture/Compare 2 output polarity

Refer to the description of CC1P.

4

CC2E

rw

0x00

Capture/Compare 2 output enable

Refer to the description of CC1E.

3:2

Reserved

Reserved, always read as 0

1

CC1P

rw

0x00

Capture/Compare 1 output polarity

CC1 channel is configured as output:

0: OC1 active high

1: OC1 active low

The CC1 channel is configured as an input:

This bit selects whether IC1 or IC1's inverted signal is used as a trigger or capture signal.

0: No inversion: capture occurs on the rising edge of IC1; when used as an external trigger

, IC1 is not inverted

1: Invert: capture occurs on the falling edge of IC1; when used as an external trigger,

IC1 invert

Note: When the LOCK level (LCCK bit in the TIMx\_BDTR register) is set to 3 or

At 2, the bit cannot be modified.

0

CC1E

rw

0x00

Capture/Compare 1 output enable

CC1 channel is configured as output:

0: Off-OC1 disables output, so the output level of OC1 depends on

The values ​​​​of the MOE, OSSI, OSSR, OIS1, OIS1N, and CC1NE bits

1: On-OC1 signal is output to the corresponding output pin, and its output level depends on

Depends on MOE, OSSI, OSSR, OIS1, OIS1N and CC1NE bits

Value of

The CC1 channel is configured as an input:

This bit determines whether the value of the counter can be captured into the TIMx\_CCR1 register.

0: capture prohibited

1: Capture enable

46.

Ocx

**CCxE** bit

**OCx** output status

0

Disable output (OCx = 0, OCx\_EN = 0)

1

OCx = OCxREF + polarity, OCx\_EN = 1

Note: The state of external I/O pins connected to standard OCx channels depends on the state of the OCx channel and GPIO and AFIO

register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

286/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 309** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**12.4.10**

**(TIMx\_CNT)**

Offset address: 0x24

Reset value: 0x0000 0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

CNT

rw

0x0000

High counter value of counter (High counter value)

15:0

CNT

rw

0x0000

Low counter value of counter (Low counter value)

**12.4.11**

**(TIMx\_PSC)**

Offset address: 0x28

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

PSC

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

PSC

rw

0x0000

Prescaler value

The clock frequency of the counter (CK\_CNT) is equal to f CK\_PSC /(PSC + 1).

The PSC includes each time the update event occurs, load the current prescaler

The value of the memory. Update events include the counter being cleared by the UG bit of TIM\_EGR

'0' or the slave controller working in reset mode clears '0'.

**12.4.12**

**(TIMx\_ARR)**

Offset address: 0x2C

Reset value: 0x0000 0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

ARR

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

ARR

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

287/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 310** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

ARR

rw

0x0000

High auto-reload value

15:0

ARR

rw

0x0000

Low auto-reload value

ARR contains the value to be loaded into the actual auto-reload register.

For details, refer to section [12.3.](https://translate.googleusercontent.com/translate_f#257) 1: Updates and actions related to ARR.

When the value of auto-reload is empty, the counter does not work.

**12.4.13**

**/**

**1(TIMx\_CCR1)**

Offset address: 0x34

Reset value: 0x0000 0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR1

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR1

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

CCR1

rw

0x0000

High Capture/Compare 1 value

15:0

CCR1

rw

0x0000

Capture/Compare 1 Low Capture/Compare 1 value

If the CC1 channel is configured as an output:

CCR1 contains the value loaded into the current capture/compare 1 register (preload

value).

If the preload is not selected in the TIMx\_CCMR1 register (OC1PE bit)

Load function, the written value will be transferred to the current register immediately. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 1

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC1 port.

If the CC1 channel is configured as an input:

CCR1 contains the count transmitted by the last input capture 1 event (IC1)

器值。 Device value. Device value.

**12.4.14**

**/**

**2(TIMx\_CCR2)**

Offset address: 0x38

Reset value: 0x0000 0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR2

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

288/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 311** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

CCR2

rw

0x0000

High Capture/Compare 2 value

15:0

CCR2

rw

0x0000

Capture/Compare 2 Low Capture/Compare 2 value

If the CC2 channel is configured as an output:

CCR2 contains the value loaded into the current capture/compare 2 register (preload

value).

If the preload is not selected in the TIMx\_CCMR2 register (OC2PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 2

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC2 port.

If the CC2 channel is configured as an input:

CCR2 contains the count transmitted by the last input capture 2 event (IC2)

器值。 Device value. Device value.

**12.4.15**

**/**

**3(TIMx\_CCR3)**

Offset address: 0x3C

Reset value: 0x0000 0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR3

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR3

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

CCR3

rw

0x0000

High Capture/Compare 3 value

15:0

CCR3

rw

0x0000

Capture/Compare 3 Low Capture/Compare 3 value

If the CC3 channel is configured as an output:

CCR3 contains the value loaded into the current capture/compare 3 register (preload

value).

If the preload is not selected in the TIMx\_CCMR3 register (OC3PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 3

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC3 port. If the CC3 channel is equipped

Set as input:

CCR3 contains the count transmitted by the last input capture 3 event (IC3)

器值。 Device value. Device value.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

289/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 312** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**12.4.16**

**/**

**4(TIMx\_CCR4)**

Offset address: 0x40

Reset value: 0x0000 0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR4

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR4

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

CCR4

rw

0x0000

High Capture/Compare 4 value

15:0

CCR4

rw

0x0000

Capture/Compare 4 Low Capture/Compare 4 value

If the CC4 channel is configured as output:

CCR4 contains the value loaded into the current capture/compare 4 register (preload

value).

If the preload is not selected in the TIMx\_CCMR4 register (OC4PE bit)

Load characteristics, the written value will be immediately transferred to the current register. Otherwise only

When an update event occurs, this preload value is transferred to the current capture/compare 4

In the register. The current capture/compare register participates in the same counter as TIMx\_CNT

Comparison and generate an output signal on the OC4 port.

If the CC4 channel is configured as input:

CCR4 contains the count transmitted by the last input capture 4 event (IC4)

器值。 Device value. Device value.

**12.4.17 DMA**

**(TIMx\_DCR)**

Offset address: 0x48

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

DBL

Res.

Res.

DBA

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

290/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 313** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

12:8

DBL

rw

0x00

DMA continuous transfer length (DMA burst length)

These bits define the transfer length of DMA in continuous mode (when the

When the TIMx\_DMAR register is read or written, the timer

Continued transmission), that is: define the number of transmissions, the transmission can be half-word (double-byte)

Or bytes:

00000: 1 transmission 00001: 2 transmissions

00010: 3 transmissions...

...10001: 18 transmissions

Example: We consider such a transmission: DBL = 7, DBA = TIM2\_CR1

-If DBL = 7, DBA = TIM2\_CR1 indicates the address of the data to be transmitted,

Then the transmitted address is given by:

(Address of TIMx\_CR1) + DBA + (DMA index), where DMA index

= DBL

Among them (the address of TIMx\_CR1) + DBA plus 7, gives the write

Enter or read the address of the data, so that the data transfer will occur at the slave address

(Address of TIMx\_CR1) + 7 registers starting with DBA. According to DMA

The setting of the data length may occur as follows:

-If the data is set to halfword (16 bits), the data will be transferred to all 7

Registers.

-If the data is set to bytes, the data will still be transferred to all 7 registers:

The first register contains the first MSB byte, the second register contains the first

One LSB byte, and so on. Therefore, for the timer, the user must indicate

Set the width of data transferred by DMA.

7:5

Reserved

Reserved, always read as 0.

4:0

DBA

rw

0x00

DMA base address (DMA base address) These bits define the DMA in

Base address in continuous mode (when reading the TIMx\_DMAR register or

When writing), DBA is defined as starting from the address where the TIMx\_CR1 register is located

Offset:

00000: TIMx\_CR1

00001: TIMx\_CR2

00010: TIMx\_SMCR

...

**12.4.18**

**DMA**

**(TIMx\_DMAR)**

Offset address: 0x4C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

DMAB

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

291/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 314** |

UM\_MM32F003\_q\_Ver1.19

32

Use (TIMX32 BIT)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

DMAB

rw

0x0000

DMA register for burst accesses

Reading or writing to the TIMx\_DMAR register will cause

Memory access operation:

TIMx\_CR1 address + DBA + DMA index, where:'TIMx\_CR1

Address' is the address where the control register 1 (TIMx\_CR1) is located;

'DBA' is the base address defined in the TIMx\_DCR register;

'DMA index' is the offset automatically controlled by DMA, it depends on

DBL defined in the TIMx\_DCR register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

292/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 315** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

13

**(TIM14)**

Basic timer (TIM14)

**13.1 TIM14**

The basic timer TIM14 consists of a 16-bit auto-load counter, which is driven by a programmable prescaler.

It is suitable for a variety of purposes, including measuring the pulse width of an input signal (input capture), or generating an output waveform (output comparison

And PWM).

Using the timer prescaler and RCC clock to control the prescaler, the pulse width and waveform period can be adjusted from several microseconds.

Adjustment from seconds to several milliseconds.

The basic timer TIM14 is completely independent and does not share any resources.

**13.2 TIM14**

• 16-bit auto-loading counter

• 16-bit programmable (can be modified in real time) prescaler, counter clock frequency division factor is between 1 ∼ 65536

Any value of

• Independent channel

**–** Input capture

**–** Output comparison

**–** PWM generation (edge-aligned mode)

• An interrupt/DMA is generated when the following events occur:

**–** Update: counter overflow, counter initialization (via software)

**–** Input capture

**–** Output comparison

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

293/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 316** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

667481

trigger

Controller

CNT counter

+/-

U

UI

U

PSC

Prescaler

C1I

Prescaler IC1PS

U

Output

control

Input filter and

Edge detector

IC1

OC1REF

CC1I

OC1

TIMx \_CH 1

Enable, count

TIMx\_CH1

Internal clock (CK\_INT)

Stop, clear

CK\_PSC

CK\_CNT

TI1FP1

TI1

⌘˖

ṩᦞ᧗ࡦսˈ൘ਁ⭏ 8 һԦਾˈ亴㻵䖭ᇴᆈಘcheck ᇩ䖜〫ࡠᴹ᭸ᇴᆈಘ

һԦ

ѝᯝ઼'0$䗃ࠪ

U

Reg

Auto-reload register

Capture/Compare 1 register

170.

**13.3 TIM14**

**13.3.1**

The main part of the programmable basic timer is a 16-bit counter and its associated auto-load register. This meter

The counter can count up. The counter clock is divided by the prescaler. Counter, auto-load register and pre-division

The frequency register can be read and written by software, and can still be read and written while the counter is running. The time base unit contains:

• Counter register (TIM14\_CNT)

• Prescaler register (TIM14\_PSC)

• Auto load register (TIM14\_ARR)

The auto-reload register is pre-loaded, and writing or reading the auto-reload register will access the pre-load register. According to TIM14\_CR1

The setting of the autoload preload enable bit (ARPE) in the register, the content of the preload register is immediately or every time

The update event UEV is transferred to the shadow register. When the counter reaches the overflow condition and when the TIM14\_CR1 register

When the UDIS bit in is equal to 0, an update event is generated. Update events can also be generated by software. Later will describe in detail each

Generation of update events under one configuration.

The counter is driven by the clock output CK\_CNT of the prescaler, only if the counter in the TIM14\_CR1 register is set

CK\_CNT is valid only when the counter enables bit (CEN). (For details on counter enable, please refer to the slave

Mode description).

Note: The real counter enable signal CNT\_EN is set after one clock cycle of CEN.

The prescaler can divide the counter clock frequency by any value between 1 and 65536. It is based on a (in

The 16-bit counter controlled by the 16-bit register in the TIM14\_PSC register. This control register has a buffer,

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

294/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 317** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

It can be changed while working. The parameters of the new prescaler are adopted when the next update event arrives.

The following figures show examples of changing counter parameters when the prescaler is running.

059785

CK\_PSC

CEN

ᇊᰦಘᰦ䫏 = CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

02

03

0

1

1

0

0

0

0

0

0

1

1

1

1

TIMx\_PSCᇴᆈಘ

171.

1

2

763391

CK\_PSC

CEN

ᇊᰦಘᰦ䫏=CK\_CNT

Sorrowful

ᴤᯠһԦ(UEV)

Sorrowfulness

Sorrowful

Benevolence

F7

F8 F9 FA FB FC

00

01

0

3

3

0

0

0

2

2

0

1

3

1

3

TIMx\_PSCᇴᆈಘ

172.

1

4

**13.3.2**

In the up-counting mode, the counter counts from 0 to the autoload value (the content of the TIM14\_ARR counter), and then

It starts counting from 0 and generates a counter overflow event.

Setting the UG bit in the TIM14\_EGR register can also generate an update event.

Setting the UDIS bit in the TIM14\_CR1 register can disable the update event; this can avoid sending to the preload

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

295/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 318** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

The shadow register is updated when a new value is written in the memory. Until the UDIS bit is cleared, no update event will be generated. But when

When an update event should occur, the counter will still be cleared to 0, and the count of the prescaler is also requested to 0 (but the number of prescalers

Value unchanged). In addition, if the URS bit in the TIM14\_CR1 register is set (select an update request), set UG

Bit will generate an update event UEV, but the hardware does not set the UIF flag (that is, no interrupt is generated): This is to avoid catching

When the counter is cleared in acquisition mode, both update and capture interrupts are generated.

When an update event occurs, all registers are updated, and the hardware sets the update flag at the same time (according to the URS bit)

Bit (UIF bit in the TIM14\_SR register).

• The autoload shadow register is reset to the value of the preload register (TIM14\_ARR)

• The buffer of the prescaler is placed in the value of the preload register (the content of the TIM14\_PSC register)

The following figure gives some examples. When TIM14\_ARR = 0x36, the counter will operate at different clock frequencies:

894901

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

31

32 33 34 35 36

05

Counter overflow

Update interrupt flag (UIF)

00 01 02 03 04

06 07

173.

1

100720

0034

0035

0000

0036

0001

0002

0003

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

174.

2

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

296/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 319** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

564964

0035

0000

0036

0001

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

175.

4

372226

1F

20

00

CK\_PSC

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

176.

N

370803

Write new value in TIMx\_ARR

31

32

33 34

35

36

05

00 01

02

03

04

06

07

FF

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Auto-reload register

177.

ARPE = 0

(TIM14\_ARR

)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

297/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 320** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

874537

Write new value in TIMx\_ARR

F0

F1 F2 F3 F4 F5

05

00 01 02 03 04

06 07

F5

36

F5

36

CK\_PSC

CNT\_EN

Timer clock = CK\_CNT

Counter register

Update event (UEV)

Counter overflow

Update interrupt flag (UIF)

Automatic reload of preload registers

Auto-reload shadow register

178.

ARPE = 1

(TIM14\_ARR)

**13.3.3**

'Time base unit' explains how the update event (UEV) is generated when the counter overflows, but in fact it can only repeat

Generated when the count reaches 0. This feature is very useful for generating PWM signals.

This means that every N count overflows, data is transferred from the preload register to the shadow register (TIMx\_ARR automatically

Reload register, TIMx\_PSC preload register, and capture/compare register in compare mode (TIMx\_CCRx),

N is the value in the TIMx\_RCR repeat count register.

The repetition counter decrements when any of the following conditions are true:

• Every time the counter overflows in up-counting mode

The repetition counter is automatically loaded, and the repetition rate is defined by the value of the TIMx\_RCR register. When the update event is made by the software

Generated (by setting the UG bit in TIMx\_EGR) or generated by the hardware slave mode controller, regardless of repetition

What is the value of the counter, an update event occurs immediately, and the contents of the TIMx\_RCR register are reloaded to repeat

counter.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

298/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 321** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

505243

counter

TIMX\_CNT

TIMX\_RCR=0

UEV

UEV

TIMX\_RCR=1

UEV

TIMX\_RCR=2

UEV

TIMX\_RCR=3

UEV

TIMX\_RCR=3

UEV

(by SW)

Update event:

Edge alignment mode

Count up

And resynchronize

The preload register is transferred to the active register and an update interrupt is generated.

179.

TIMx\_RCR

**13.3.4**

The counter clock is provided by the internal clock (CK\_INT).

CEN (TIM14\_CR1 register) and UG bit (TIM14\_EGR register) are de facto control bits and can only

Modified by software (except UG bit is still cleared automatically). As long as the CEN bit is written as '1', the prescaler clock is determined by

The internal clock CK\_INT is provided.

The figure below shows the operation of the control circuit and the up-counter in normal mode without the prescaler.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

299/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 322** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

199864

31

32

33 34

35

36

05

00

01

02

03

04

06

07

Internal clock

CEN = CNT\_EN

Counter clock = CK\_CNT = CK\_PSC

Counter register

CNT\_INT

UG

180.

1

**13.3.5**

**/**

Each capture/compare channel surrounds a capture/compare register (including shadow registers), including the captured input

Part (digital filtering, multiplexing and prescaler), and output part (comparator and output control).

The following figures are an overview of the capture/compare channels. The input section samples the corresponding TIx input signal and generates a

Filtered signal TIxF. Then, an edge detector with polarity selection generates a signal (TIxFPx), which

Can be triggered as input from the mode controller or as a capture control. This signal enters the capture register through prescaler

(ICxPS).

565511

**IC1PS**

filter

Down counter

**ICF[3:0]**

edge

Detector

**CC1P**

**ICPS[1:0]**

**TI1**

**TIMx\_CCMR1**

**TIMx\_CCER**

**TI1F\_Rising**

**TI1F\_Falling**

**01**

**10**

**11**

**0**

**1**

**f DTS**

**TI1F**

**divider**

**/1,/2,/4,/8**

**IC1**

**CC1E**

**CC1S[1:0]**

**TIMx\_CCMR1**

**TIMx\_CCER**

Keep

**TI1FP1**

**TI1F\_ED**

To slave mode controller

Keep

181.

/ ( 1

)

The output section generates an intermediate waveform OCxREF (high effective) as a reference, the end of the chain determines the pole of the final output signal

Sex.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

300/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 323** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

903065

MCU peripheral interface

APB bus

Capture/compare preload registers

high

8

CC1S[1]

OC1PE

TIMx\_CCMR1

UEV

Reading CCR1L

Reading CCR1H

CC1S[1]

CC1S[0]

IC1PS

CC1E

CC1G

TIMx\_EGR

read\_in\_progress

enter

mode

8

low

(Assuming 16 bits)

Capture/Compare Shadow Register

capture

counter

Comparators

capture\_transfer

compare\_transfer

write\_in\_progress

Output

mode

CNT> CCR1

CNT = CCR1

(From time base unit)

Write to CCR1H

Write to CCR1L

CC1S[0]

OC1PE

S

R

S

R

182.

/ 1

358425

Output mode

Controller

OC1M[2:0]

TIMx\_CCMR1

CNT>CCR1

CNT=CCR1

ETRF

OC1ref

To slave mode controller

0

1

CC1P

TIMx\_CCER

Output enable

Circuit

CC1E TIMx\_CCER

OC1

183.

/ ( 1)

The capture/compare module consists of a preload register and a shadow register. The read and write process only operates preload registration

Device. In capture mode, the capture occurs on the shadow register, and then copied into the preload register.

In the comparison mode, the contents of the preload register are copied to the shadow register, and then the contents of the shadow register and

Counter.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

301/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 324** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**13.3.6**

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched to capture/ratio

Compare register (TIM14\_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIM14\_SR register)

Set to 1, if an interrupt or DMA operation is enabled, an interrupt or DMA operation will be generated. If a capture event occurs

When the CCxIF flag is already high, the repeated capture flag CCxOF (TIM14\_SR register) is set. Write CCxIF

= 0 can clear CCxIF, or read the captured data stored in the TIM14\_CCRx register can also clear CCxIF. write

CCxOF = 0 can clear CCxOF.

The following example shows how to capture the value of the counter into the TIM14\_CCR1 register on the rising edge of the TI1 input, step

The steps are as follows:

1. Select a valid input: TIM14\_CCR1 must be connected to the TI1 input, so write to TIM14\_CCMR1 to register

CC1S = 01 in the device, when CC1S is not 00, the channel is configured as an input, and TIM14\_CCR1 sends

The memory becomes read-only.

2. According to the characteristics of the input signal, configure the input filter to the required bandwidth (that is, when the input is TIx, the input filter controls

The control bit is the ICxF bit in the TIM14\_CCMRx register). Assuming that the input signal is at most 5 clock cycles

For inter-time jitter, we must configure the filter bandwidth to be longer than 5 clock cycles. So we can (at fDTS frequency)

8 consecutive samples to confirm the last real edge transition on TI1, which is in the TIM14\_CCMR1 register

Write IC1F = 0011.

3. Select the effective conversion edge of the TI1 channel and write CC1P = 0 and CC1NP in the TIM14\_CCER register

= 0 (rising edge).

4. Configure the input prescaler. In this example, we want to capture every valid level-shifting moment, so

The prescaler is disabled (write IC1PS = 00 in the TIM14\_CCMR1 register).

5. Set CC1E = 1 in the TIM14\_CCER register to allow the capture counter value to be captured in the capture register.

6. If necessary, enable the relevant interrupt request by setting the CC1IE bit in the TIM14\_DIER register, by setting

The CC1DE bit in the TIM14\_DIER register allows DMA requests.

When an input is captured:

• When a valid level transition occurs, the counter value is transferred to the TIM14\_CCR1 register. CC1IF flag is set

Set (interrupt flag). When at least 2 consecutive captures occur and CC1IF has not been cleared, CC1OF is also set to 1.

• If the CC1IE bit is set, an interrupt will be generated.

In order to deal with the capture overflow, it is recommended to read the data before reading the capture overflow flag, this is to avoid losing the read capture

Capture overflow information that may be generated after the overflow flag is obtained and before the data is read.

Note: Set the corresponding CCxG bit in the TIM14\_EGR register to generate an input capture interrupt request through software.

**13.3.7**

In the output mode (CCxS = 00 in the TIM14\_CCMRx register), the comparison signal (OCxREF and corresponding

OCx) can be forced into valid or invalid state directly by software, without relying on the output compare register and counter

Comparison results.

Set the corresponding OCxM = 101 in the TIM14\_CCMRx register to force the output compare signal (OCxREF/OCx)

Is valid. In this way, OCxREF is forced to high level (OCxREF is always active high), and OCx is

To the opposite value of the CCxP polarity bit. For example: CCxP = 0 (OCx high level effective), then OCx is forced to high level.

Set OCxM = 100 in the TIM14\_CCMRx register to force the OCxREF signal low. In this mode, in

The comparison between the TIM14\_CCRx shadow register and the counter is still in progress, and the corresponding flag will be modified. Therefore

The corresponding interrupt request will still be generated. This will be described in the section on output comparison mode below.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

302/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 325** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**13.3.8**

This function is used to control an output waveform or indicate when a given period of time has expired.

When the contents of the counter and the capture/compare register are the same, the output compare function does the following:

1. Set the output compare mode (OCxM bit in the TIM14\_CCMRx register) and the output polarity (TIM14\_CCER

The value defined by the CCxP bit in the register is output to the corresponding pin. When comparing and matching, the output pin can be

Hold its level (OCxM = 000), set to active level (OCxM = 001), set to no active level

(OCxM = 010) or roll over (OCxM = 011).

2. Set the flag bit in the interrupt status register (CCxIF bit in the TIM14\_SR register).

3. If the corresponding interrupt mask (CCXIE bit in the TIM14\_DIER register) is set, an interrupt is generated.

The OCxPE bit in TIM14\_CCMRx selects whether the TIM14\_CCRx register needs to use a preload register. in

In output comparison mode, the update event UEV has no effect on the OCxREF and OCx outputs. Synchronization accuracy can reach

One count cycle to the counter. The output compare mode (in single pulse mode) can also be used to output a single pulse.

Configuration steps of output comparison mode:

• Select counter clock (internal, external, prescaler)

• Write the corresponding data to the TIM14\_ARR and TIM14\_CCRx registers

• If an interrupt request and/or a DMA request is to be generated, set the CCxIE bit and/or CCxDE bit

• Choose output mode,

**–** Write OCxM = '011', toggle the output pin of OCx when the counter CNT matches CCRx

**–** Write OCxPE = '0', prohibit pre-loading

**-** the Write CCxP = '0', Which Selects High

**–** Write CCxE = '1' to allow output

• Set the CEN bit of the TIM14\_CR1 register to start the counter

The TIM14\_CCRx register can be updated by software at any time to control the output waveform, provided that the pre-use is not used

Load register (OCxPE = '0', otherwise TIM14\_CCRx shadow register can only be updated at the next event

Is updated). The figure below gives an example.

803963

0039

003A

003B

B200

B201

003A

B201

TIMX\_CNT

TIMX\_CCR1

OC1REF = OC1

Write B201h in CCR1 register

A match was detected on CCR1

If the interrupt is enabled, an interrupt is generated

184.

OC1

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

303/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 326** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**13.3.9 PWM**

Pulse width modulation mode can generate a frequency determined by the TIM14\_ARR register and registered by the TIM14\_CCRx

The detector determines the duty cycle signal.

Write '110' (PWM mode 1) or '111' (PWM mode) in the OCxM bit in the TIM14\_CCMRx register

2), each OCx output channel can be set independently to generate a PWM. The TIM14\_CCMRx register must be set

OCxPE bit to enable the corresponding preload register, and finally set the ARPE bit of the TIM14\_CR1 register to enable

Preload registers that can be automatically reloaded (in up-count mode).

Only when an update event occurs, the preload register can be transferred to the shadow register, so the counter is turned on

Before starting counting, all registers must be initialized by setting the UG bit in the TIM14\_EGR register.

The polarity of OCx can be set by the CCxP bit in the TIM14\_CCER register by software, it can be set to high

Active level or active low level. The CCxE bit in the TIM14\_CCER register controls OCx output enable. See more

Description of the TIM14\_CCERx register.

In PWM mode (mode 1 or mode 2), TIM14\_CNT and TIM14\_CCRx are always comparing whether they match

TIM14\_CNT ≤ TIM14\_CCRx.

Because this counter counts up, it can only generate edge-aligned PWM mode.

**PWM**

The following is an example of PWM mode 1. PWM signal reference when TIM14\_CNT <TIM14\_CCRx

OCxREF is high, otherwise low. If the comparison value in TIM14\_CCRx is greater than the auto-reload value (TIM14\_ARR),

Then OCxREF remains '1'. If the comparison value is 0, OCxREF remains at '0'. The following picture shows TIM14\_ARR

= 8 example of an edge-aligned PWM waveform.

458343

0 1 2 3 4 5 6 7 8 0 1

þ1ÿ

þ0ÿ

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx = 4

CCRx = 8

CCRx >8

CCRx =0

Counter register

185.

PWM

(ARR = 8)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

304/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 327** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**13.3.10**

When the microcontroller enters the debug mode (Cortex TM -M0 core is stopped), according to the DBG\_TIMx\_STOP in the DBG module

When set, the TIM14 counter will either continue normal operation or stop.

**13.4 TIM14**

47. TIM14

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

TIM14\_CR1

Control register 1

0x00000000

[Section 13.4.1](https://translate.googleusercontent.com/translate_f#327)

0x04

TIM14\_CR2

Control register 2

0x00000000

[Section 13.4.2](https://translate.googleusercontent.com/translate_f#328)

0x0C

TIM14\_DIER

Interrupt enable register

0x00000000

[Section 13.4.3](https://translate.googleusercontent.com/translate_f#329)

0x10

TIM14\_SR

Status register

0x00000000

[Section 13.4.4](https://translate.googleusercontent.com/translate_f#330)

0x14

TIM14\_EGR

Event generation register

0x00000000

[Section 13.4.5](https://translate.googleusercontent.com/translate_f#331)

0x18

TIM14\_CCMR1

Capture/Compare Mode Register 1

0x00000000

[Section 13.4.6](https://translate.googleusercontent.com/translate_f#331)

0x20

TIM14\_CCER

Capture/compare enable register

0x00000000

[Section 13.4.7](https://translate.googleusercontent.com/translate_f#334)

0x24

TIM14\_CNT

counter

0x00000000

[Section 13.4.8](https://translate.googleusercontent.com/translate_f#336)

0x28

TIM14\_PSC

Prescaler

0x00000000

[Section 13.4.9](https://translate.googleusercontent.com/translate_f#336)

0x2C

TIM14\_ARR

Autoload register

0x00000000

[Section 13.4.10](https://translate.googleusercontent.com/translate_f#336)

0x30

TIM14\_RCR

Repeat count register

0x00000000

[Section 13.4.11](https://translate.googleusercontent.com/translate_f#337)

0x34

TIM14\_CCR1

Capture/Compare Register 1

0x00000000

[Section 13.4.12](https://translate.googleusercontent.com/translate_f#337)

0x44

TIM14\_BDTR

Brake and dead zone registers

0x00000000

[Section 13.4.13](https://translate.googleusercontent.com/translate_f#338)

0x48

TIM14\_DCR

TIM14DMA control register

0x00000000

[Section 13.4.14](https://translate.googleusercontent.com/translate_f#338)

0x4C

TIM14\_DMAR

TIM14 full transfer address register

0x00000000

[Section 13.4.15](https://translate.googleusercontent.com/translate_f#339)

**13.4.1**

**1(TIM14\_CR1)**

Offset address: 0x00

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

CKD

ARPE

Reserved

URS UDIS CEN

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

305/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 328** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9:8

CKD

rw

0x00

Clock division

These 2 bits are defined in the timer clock (CK\_INT) frequency, dead time and by

Between the deadband generator and the sampling frequency used by the digital filter (ETR, TIx)

The frequency division ratio.

00: t DTS = t CK\_INT

01: t DTS = 2 xt CK\_INT

10: t DTS = 4 xt CK\_INT

11: Reserved, do not use this configuration

7

ARPE

rw

0x00

Auto-reload preload enable bit (Auto-reload preload enable)

0: TIM14\_ARR register is not buffered

1: TIM14\_ARR register is loaded into the buffer

6:3

Reserved

Reserved, read as 0

2

URS

rw

0x00

Update request source (Update request source) software selects through this bit

The source of the UEV event.

0: If UEV is enabled, any of the following events will generate UEV:

-Counter overflow

-Set UG bit

-Updates generated from the mode controller

1: If UEV is enabled, only the counter overflow will generate a UEV event

1

UDIS

rw

0x00

Disable disable (Update disable) Software enables/disables UEV through this bit

Event generation

0: UEV is allowed. Update (UEV) events are generated by any of the following events:

-Counter overflow

-Set UG bit

1: UEV is prohibited. No update event occurs, shadow registers (ARR, PSC,

CCRx) keep their values. If the UG bit is set, the counter and pre-minute

The frequency converter is re-initialized.

0

CEN

rw

0x00

Counter enable

0: disable counter

1: Enable the counter.

**13.4.2**

**2(TIM14\_CR2)**

Offset address: 0x04

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

OIS1N OIS1

CCDSCCUS Res. CCPC

Reserved

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

306/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 329** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9

OIS1N

rw

0x00

Output Idle state 1 (OC1N output) (Output Idle state 1)

0: When MOE = 0, OC1N = 0 after the dead zone

1: When MOE = 1, OC1N = 1 after the dead zone

Note: After LOCK (TIMx\_BKR register) level 1, 2, or 3 has been set, the

The bit cannot be modified.

8

OIS1

rw

0x00

Output Idle state 1 (OC1 output) (Output Idle state 1)

0: When MOE = 0, if OC1N is implemented, then OC1 = 0 after the dead zone

1: When MOE = 1, if OC1N is implemented, then OC1 = after dead zone

1.

Note: After LOCK (TIMx\_BKR register) level 1, 2, or 3 has been set, the

The bit cannot be modified.

7:4

Reserved

Reserved, read as 0

3

CCDS

rw

0x00

Capture/compare DMA selection

0: Send CCx DMA request when CCx event occurs

1: When an update event occurs, the CCx DMA request is sent

2

CCUS

rw

0x00

Capture/compare control update selection (Capture/compare control update

selection)

0: When the capture/compare control bit is preloaded (CCPC = 1), only pass

Only set COMG bit will be updated

1: When the capture/compare control bit is preloaded (CCPC = 1), set

The COMG bit or TRGI will be updated when a rising edge is generated

Note: This bit is only effective when the channel has complementary output.

1

Reserved

Reserved, read as 0

0

CCPC

rw

0x00

Capture/compare preloaded control bit (Capture/compare preloaded con-

trol)

0: CCxE, CCxNE and OCxM bits are not preloaded

1: CCxE, CCxNE and OCxM bits are preloaded after being written, only

Only updated when the COM bit is set to '1'

Note: This bit is only effective when the channel has complementary output.

**13.4.3**

**(TIM14\_DIER)**

Offset address: 0x0C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

UIE

CC1IE

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:2

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

307/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 330** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

CC1IE

rw

0x00

Capture/Compare 1 interrupt en-

able)

0: CC1 interrupt disable

1: CC1 interrupt enable

0

UIE

rw

0x00

Update interrupt enable

0: Disable update interrupt

1: Allow update interrupt

**13.4.4**

**(TIM14\_SR)**

Offset address: 0x10

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

UIF

CC1OF

CC1IF

rc\_w0 rc\_w0

rc\_w0

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

9

CC1OF

rc\_w0

0x00

Capture/Compare 1 overcap-capture (Capture/Compare 1 overcap-

ture flag)

This flag can be set by hardware only when the corresponding channel is configured for input capture

1. Write 0 to clear this bit.

0: No repeated capture;

1: When the counter value is captured in the TIM14\_CCR1 register, CC1IF

'S status is already 1.

8: 2

Reserved

Reserved, read as 0

1

CC1IF

rc\_w0

0x00

Capture/Compare 1 interrupt flag

If channel CC1 is configured in output mode:

When the counter value matches the comparison value, this bit is set to '1' by hardware, and it is set by software

Clear '0'.

0: No match occurred

1: The value of TIM14\_CNT matches the value of TIM14\_CCR1

When the content of TIM14\_CCR1 is greater than TIM14\_ARR, the counter overflows

The CC1IF bit goes high when out.

If channel CC1 is configured in input mode: this bit is set by the capture event

Set to '1' by hardware, it is cleared to 0 by software or by reading TIM14\_CCR1.

0: No input capture

1: The counter value has been captured (copied) to TIM14\_CCR1 (checked on IC1

An edge with the same polarity as the selected one is measured)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

308/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 331** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

UIF

rc\_w0

0x00

Update interrupt flag

This bit is set to '1' by hardware when an update event occurs. It is cleared to '0' by the software.

0: No update event is generated

1: Update interrupt waiting for response. When the register is updated, this bit is set to '1' by hardware:

－If UDIS of TIM14\_CR1 register = 0, when the count overflows;

-If UDIS = 0 and URS = 0 in the TIM14\_CR1 register, pass the soft

When writing the UG bit in the TIM14\_EGR register to re-initialize the timer.

**13.4.5**

**(TIM14\_EGR)**

Offset address: 0x14

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

CC1G

UG

w

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:2

Reserved

Reserved, read as 0

1

CC1G

w

0x00

Generate Capture/Compare 1 generation

This bit is set by software to generate a capture/compare event.

Reset to 0.

0: No action

1: Generate a capture/compare event on channel CC1:

If channel CC1 is configured as an output:

Set CC1IF=1, if the corresponding interrupt is turned on, the corresponding interrupt will be generated.

If channel CC1 is configured as input:

The current counter value is captured into the TIM14\_CCR1 register and set

CC1IF = 1, if the corresponding interrupt is turned on, the corresponding interrupt is generated. If

CC1IF is already 1, set CC1OF = 1.

0

UG

w

0x00

Generate an update event (Update generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: Re-initialize the counter and generate an update event. Note the prescaler

The counter of the counter is also cleared to '0' (but the prescaler coefficient is unchanged). Counter was

Clear '0'.

**13.4.6**

**/**

**1(TIM14\_CCMR1)**

Offset address: 0x18

Reset value: 0x0000

The channel can be used for input (capture mode) or output (comparison mode), and the direction of the channel is defined by the corresponding CCxS. The deposit

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

309/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 332** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

The function of other bits of the device is different from the output mode. OCxx describes the function of the channel in output mode, ICxx describes

The function of the channel in output mode. Therefore, it must be noted that the function of the same bit in output mode and input mode is not

Same.

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

Res.

OC1M

IC1F

OC1PE OC1FE

IC1PSC

CC1S

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:7

Reserved

Reserved, read as 0

6:4

OC1M

rw

0x00

Output compare 1 mode

These 3 bits define the action of the output reference signal OC1REF, and OC1REF

Decided the value of OC1. OC1REF is active high, and OC1 has

The effect level depends on the CC1P bit.

000: freeze. Output compare register TIM14\_CCR1 and counter

Comparison between TIM14\_CNT has no effect on OC1REF

001: Set channel 1 to effective level when matching. When the counter TIM14\_CNT

Is the same as the capture/compare register 1 (TIM14\_CCR1), it is forced

OC1REF is high

010: Set channel 1 to invalid level when matching. When the counter TIM14\_CNT

Is the same as the capture/compare register 1 (TIM14\_CCR1), it is forced

OC1REF is low

011: Flip. When TIM14\_CCR1=TIM14\_CNT, flip

OC1REF level

100: Forced to invalid level. Force OC1REF low

101: Forced to effective level. Force OC1REF high

110: PWM mode 1, when TIM14\_CNT <TIM14\_CCR1

Channel 1 is the active level, otherwise it is the inactive level

111: PWM mode 2, when TIM14\_CNT <TIM14\_CCR1

Channel 1 is an invalid level, otherwise it is an effective level.

Note: In PWM mode 1 or PWM mode 2, only when the comparison result changes

Or when switching from freeze mode to PWM mode in output compare mode, OC1REF

The level changes.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

310/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 333** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

OC1PE

rw

0x00

Output compare 1 preload en-

able)

0: Disable the preload function of the TIM14\_CCR1 register, which can be written at any time

TIM14\_CCR1 register, and the newly written value takes effect immediately

1: Enable the preload function of TIM14\_CCR1 register, read and write operations are only

For preload register operation, the preload value of TIM14\_CCR1 is being updated

The file is loaded into the current register when it arrives

Note: Only in single pulse mode (OPM = '1' in TIM14\_CR1 register),

To use the PWM mode without confirming the preload register, otherwise its action will not

determine.

2

OC1FE

rw

0x00

Output compare 1 fast enable (Output compare 1 fast enable)

This bit is used to speed up the response of the CC output to trigger input events.

0: CC1 operates normally according to the value of the counter and CCR1, even if it is triggered

The device is turned on. When the trigger input has a valid edge, activate CC1

The minimum output delay is 5 clock cycles

1: The valid edge input to the trigger acts as if a comparison match has occurred.

Therefore, OC is set to the comparison level regardless of the comparison result. Sample trigger

The delay between the effective edge of the device and the output of CC1 is reduced to 3 clock cycles.

OCFE only works when the channel is configured in PWM1 or PWM2 mode

1:0

CC1S

rw

0x00

Capture/Compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: reserved

11: reserved

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIM14\_CCER register)

Is writable.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

311/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 334** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7:4

IC1F

rw

0x00

Input capture 1 filter

These bits define the sampling frequency and digital filter length of the TI1 input. number

The word filter consists of an event counter, which will record after N events

Produces an output transition:

0000: no filter, sampling with fDTS

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

0001: Sampling frequency f SAMPLING = f CK\_INT , N = 2

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

0100: Sampling frequency f SAMPLING = f DTS /2, N = 6

1100: Sampling frequency f SAMPLING = f DTS /16, N = 8

0101: Sampling frequency f SAMPLING = f DTS /2, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

0110: Sampling frequency f SAMPLING = fDTS/4, N = 6

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

0111: Sampling frequency f SAMPLING = f DTS /4, N = 8

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

3:2

IC1PSC

rw

0x00

Input capture 1 prescaler

These 2 bits define the prescaler coefficient of the CC1 input (IC1).

When CC1E = 0 (in the TIM14\_CCER register), the prescaler is reset.

00: No prescaler, every edge detected on the capture input port is touched

Send a capture

01: Capture is triggered every 2 events

10: Capture is triggered every 4 events

11: Trigger every 8 events

1:0

CC1S

rw

0x00

Capture/compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output

01: CC1 channel is configured as input, IC1 is mapped on TI1

10: reserved

11: reserved

Note: CC1S is only when the channel is closed (CC1E = 0 in the TIM14\_CCER register)

Is writable.

**13.4.7**

**/**

**(TIM14\_CCER)**

Offset address: 0x20

Reset value: 0x0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

312/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 335** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

Reserved

CC1NP

CC1P CC1E

Res.

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:4

Reserved

Reserved, read as 0

3

CC1NP

rw

0x00

Capture/Compare 1 complementary output polarity (Capture/Compare 1 comple-

mentary output Polarity)

When channel CC1 is configured as an output, CC1NP must be operated to clear, CC1NP

= 0;

When channel CC1 is configured as input, CC1NP and CC1P are jointly controlled

Refer to CC1P description for the polarity of TI1FP1

2

Reserved

Reserved, read as 0

1

CC1P

rw

0x00

Capture/Compare 1 output polarity

CC1 channel is configured as output:

0: OC1 active high

1: OC1 active low

The CC1 channel is configured as an input:

CC1P/CC1NP is used to select the signal TI1FP1 and

The polarity of TI2FP1, this bit IC1 is also the inverted signal of IC1.

00: No inversion/rising edge: capture occurs on the rising edge of TIxFP1 (capture mode

Type), TIxFP1 is not inverted;

01: Inverting/falling edge: Capture occurs on the falling edge of TIxFP1 (capture mode

Type), TIxFP1 is inverted;

10: Reserved, do not use this configuration;

11: No inversion/rising and falling edges: capture occurs on the rising edge of TIxFP1

With the falling edge (capture mode), TIxFP1 is not inverted.

Note: When the LOCK level is set to 2 or 3 (LOCK in the TIMx\_BDTR register

Bit) and CC1S = '00' (the channel is configured as an output), this bit cannot be modified

change.

0

CC1E

rw

0x00

Capture/Compare 1 output enable

CC1 channel is configured as output:

0: Off-OC1 output prohibited

1: On-OC1 signal is output to the corresponding output pin

CC1 channel is configured as input: this bit determines whether the counter value can be captured

Into the TIM14\_CCR1 register.

0: capture prohibited

1: Capture enable

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

313/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 336** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

48.

OCx

**CCxE** bit

**OCx** output status

0

Disable output (OCx = 0, OCx\_EN = 0)

1

OCx = OCxREF + polarity, OCx\_EN = 1

Note: The status of external I/O pins connected to standard OCx channels depends on the OCx channel status and GPIO registers.

**13.4.8**

**(TIM14\_CNT)**

Offset address: 0x24

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CNT

rw

0x0000

Counter value

**13.4.9**

**(TIM14\_PSC)**

Offset address: 0x28

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

PSC

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

PSC

rw

0x0000

Prescaler value

The clock frequency of the counter (CK\_CNT) is equal to f CK\_PSC /(PSC + 1).

The PSC includes each time the update event occurs, load the current prescaler

The value of the memory.

**13.4.10**

**(TIM14\_ARR)**

Offset address: 0x2C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

ARR

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

314/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 337** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

ARR

rw

0x0000

Auto-reload value

ARR contains the value to be loaded into the actual auto-reload register.

For details, please refer to the time base unit chapter: ARR update and action.

When the value of auto-reload is empty, the counter does not work.

**13.4.11**

**(TIM14\_RCR)**

Offset address: 0x30

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

REP

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, always read as 0.

7:0

REP

rw

0x00

Repetition counter value

After the preload register is enabled, these bits allow the user to set the compare register

The update rate (ie periodically transferred from the preload register to the current register

If the update interrupt is allowed, it will also affect the generation of update interrupt

s speed.

Each time the down counter REP\_CNT reaches 0, an update event will be generated

And the counter REP\_CNT starts counting from the REP value again. by

REP\_CNT is only reloaded when the periodic update event U\_RC occurs

REP value, so the new value written to the TIMx\_RCR register

It takes effect only when the periodic update event occurs. This means that in PWM mode,

(REP+1) corresponds to the number of PWM cycles in edge-aligned mode.

**13.4.12**

**/**

**1(TIM14\_CCR1)**

Offset address: 0x34

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CCR1

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

315/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 338** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR1

rw

0x0000

Capture/Compare 1 value

If the CC1 channel is configured as an output:

CCR1 contains the value loaded into the current capture/compare 1 register (preload

value).

If not selected in the TIM14\_CCMR1 register (OC1PE bit)

Preload function, the written value will be transferred to the current register immediately. no

Then only when the update event occurs, this preload value is transmitted to the current capture

Get/Compare 1 register. The current capture/compare register participates in the same counter

Comparison of TIM14\_CNT, and output signal is generated on OC1 port.

If the CC1 channel is configured as an input:

CCR1 contains the count transmitted by the last input capture 1 event (IC1)

器值。 Device value. Device value.

**13.4.13**

**(TIM14\_BDTR)**

Offset address: 0x44

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

Reserved

MOE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

MOE

rw

0x00

MOE: Main output enable

0: output prohibited

1: If the corresponding enable bit is set (TIM14\_CCER register

CCxE, CCxNE bit), then turn on the output

For details on OC/OCN enable, see Section [13.4.7](https://translate.googleusercontent.com/translate_f#334) , Capture/Compare Enable

Energy register (TIM14\_CCER).

14:0

Reserved

Reserved, always read as 0

**13.4.14 DMA**

**(TIM14\_DCR)**

Offset address: 0x48

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

DBA

DBL

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

316/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 339** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

**Bit**

**Field**

**Type**

**Reset**

**Description**

12:8

DBL

rw

0x00

DMA continuous transfer length (DMA burst length)

These bits define the transfer length of DMA in continuous mode (when the

When the TIMx\_DMAR register is read or written, the timer

Continue to send), that is: define the number of bytes transmitted:

00000: 1 byte

00001: 2 bytes

00010: 3 bytes

...

...

10001: 18 bytes

7:5

Reserved

Reserved, always read as 0.

4:0

DBA

rw

0x00

DMA base address (DMA base address) These bits define the DMA

The base address of the transfer (when reading or writing the TIMx\_DMAR register),

DBA is defined as the offset from the address where the TIMx\_CR1 register is located.

Eg:

00000: TIMx\_CR1

00001: TIMx\_CR2

00010: TIMx\_SMCR

...

Example: To complete the following transmission: DBL = 7, DBA = TIMx\_CR1

At this time, the transfer starts from the address of TIMx\_CR1 to/from 7 consecutive registers

Proceed.

**13.4.15**

**(TIM14\_DMAR)**

Offset address: 0x4C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

DMAB

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

DMAB

rw

0x0000

DMA register for burst accesses

Reading or writing to the TIMx\_DMAR register will cause

Memory access operation:

TIMx\_CR1 address + (DBA + DMA index) × 4,

Among them:'TIMx\_CR1 address' is the control register 1 (TIMx\_CR1)

The address at;'DBA' is the base defined in the TIMx\_DCR register

Address;'DMA index' is the offset automatically controlled by DMA, it depends on

DBL defined in the TIMx\_DCR register.

Examples of how to use DMA concurrent operations

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

317/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 340** |

UM\_MM32F003\_q\_Ver1.19

(TIM14)

In this example, the concurrent function of the timer DMA is used to transfer the contents of the CCRx register in halfword mode.

Update to CCRx register. Follow the steps below:

1. Configure related DMA channels:

(a) The DMA channel device address is the DMAR register address

(b) The DMA channel memory address is the RAM buffer containing the data to be transferred to the CCRx register via DMA

District address

(c) Number of transmitted data = 3 (see note below)

(d) Prohibition in notification mode

2. Configure the DBA and DBL bits of the DCR register: DBL = 3 transfers, DBA = 0xE.

3. Enable TIMx update DMA request (set UDE bit of DIER register)

4. Enable TIMx

5. Enable DMA channel

Note: In this example, all CCRx registers are updated at once. If the CCRx register needs to be updated twice, the number

The data quantity should be 6, and the RAM buffer should contain data1, data2, data3, data4, data5 and data6. The data is as follows

The process is transferred to the CCRx register: on the first update DMA request, data1 is transferred to CCR2, data2 is transferred to

CCR3, data3 are transferred to CCR4, and in the second DMA update interrupt request, data4 is transferred to CCR2, data5 are

Transferred to CCR3, data6 is transferred to CCR4.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

318/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 341** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

14

**(TIM16/17)**

Basic timer (TIM16/17)

**14.1 TIM16/17**

The basic timer TIM16/17 consists of a 16-bit auto-load counter, driven by a programmable prescaler.

It is suitable for a variety of purposes, including measuring the pulse width of an input signal (input capture), or generating an output waveform (output comparison

And PWM).

Using the timer prescaler and RCC clock to control the prescaler, the pulse width and waveform period can be adjusted from several microseconds.

Adjustment from seconds to several milliseconds.

The basic timers TIM16/17 are completely independent, they do not share any resources.

**14.2**

• 16-bit auto-reload register

• 16-bit programmable (can be modified in real time) prescaler, counter clock frequency division factor is between 1 ∼ 65536

Any value of

• 1 independent channel:

**–** Input capture

**–** Output comparison

**–** PWM generation (edge-aligned mode)

**–** Single pulse mode output

• Complementary output with programmable dead time

• Allow repeat counters to update timer registers after a specified number of counter cycles

• The brake input signal can put the timer output signal into a reset state or a known state

• An interrupt/DMA is generated when the following events occur:

**–** Update: Counter overflow

**–** Input capture

**–** Output comparison

**-** Brake Signal the INPUT

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

319/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 342** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

999664

TIMx\_CH1

Internal clock (CK\_INT)

U

TIMx\_CH1N

Counter enable (CEN)

+/- CNT counter

TIMx\_CH1

TIMx\_BKIN

TI1

BRK

Input filter and

Edge detector

Polarity selection

BI

IC1

C1I

CK\_PSC

CK\_CNT

IC1PS

TI1FP1

CC1I

OC1REF

Repeat register

DTG register

UI

OC1

OC1N

DTG

Output

control

PSC

Prescaler

Prescaler

U

U

Auto-reload register

Capture/Compare 1 register

Stop, clear, or increase */* decrease

Clock failure event from the clock controller

⌘˖

ṩᦞ᧗ࡦսˈ൘ਁ⭏ 8 һԦਾˈ亴㻵䖭ᇴᆈಘcheck ᇩ䖜〫ࡠᴹ᭸ᇴᆈಘ

һԦ

ѝᯝ઼'0$䗃ࠪ

U

Reg

REP register

186.

TIM16

TIM17

**14.3**

**14.3.1**

The main part of the programmable basic timer is a 16-bit counter and its associated auto-load register. This meter

The counter can count up, the counter clock is divided by the prescaler. Counter, auto-load register and pre-division

Frequency register can be read and written by software, and can still be read and written while the counter is running. The time base unit contains:

• Counter register (TIMx\_CNT)

• Prescaler register (TIMx\_PSC)

• Auto load register (TIMx\_ARR)

• Repeat number register (TIMx\_RCR)

The auto-reload register is pre-loaded, and writing or reading the auto-reload register will access the pre-load register. According to TIMx

The setting of the autoload preload enable bit (ARPE) in the \_CR1 register, the content of the preload register is immediately or

It is transferred to the shadow register at each update event UEV. When the counter reaches the overflow condition and when TIMx\_CR1 is sent

When the UDIS bit in the memory is equal to 0, an update event is generated. Update events can also be generated by software. Will be described in detail later

Describe the generation of update events for each configuration.

The counter is driven by the clock output CK\_CNT of the prescaler, only if the counter in the TIMx\_CR1 register is set

CK\_CNT is valid only when the counter enables bit (CEN). (For details on counter enable, please refer to the slave

Mode description).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

320/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 343** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Note: The real counter starts counting after one clock cycle of CEN is set.

The prescaler can divide the counter clock frequency by any value between 1 and 65536. It is based on a (in

A 16-bit counter controlled by a 16-bit register in the TIMx\_PSC register. This control register has a buffer, it

Can be changed at work. The new prescaler parameters are adopted when the next update event arrives.

The figure below shows an example of changing the counter parameters while the prescaler is running.

741411

CK\_PSC

CEN

Sorrowful

ᴤᯠһԦ

Sorrowfulness

Sorrowful

Benevolence

F7 F8 F9 FA FB FC

00

01

02

03

0

1

1

0

0

0

0

0

0

1

1

1

1

በᮠ٬㠣

(UEV)

TIMx\_PSC register

ᇊᰦಘᰦ䫏 = CK\_CNT

187.

1

2

994427

CK\_PSC

CEN

ᇊᰦಘᰦ䫏 = CK\_CNT

Sorrowful

ᴤᯠһԦ (UEV)

Sorrowfulness

Sorrowful

Benevolence

F7 F8 F9 FA FB FC

00

01

0

3

3

0

0

0

2

2

0

1

3

1

3

በᮠ٬㠣TIMx\_PSC ᇴᆈಘ

188.

1

4

**14.3.2**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

321/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 344** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

In the up-counting mode, the counter counts from 0 to the autoload value (the content of the TIMx\_ARR counter), and then restarts

Start counting from 0 and generate a counter overflow event.

If the repetition counter function is used, it is only generated when the up count reaches the set number of repetition counts (TIMx\_RCR)

Generate an update event (UEV); otherwise, an update event will be generated each time the counter overflows. It can be produced every time the counter overflows

To generate an update event, set the UG bit in the TIMx\_EGR register (by software or using the slave mode controller)

An update event can also be generated.

Setting the UDIS bit in the TIMx\_CR1 register can disable the update event; this can avoid registering to the preload

The shadow register is updated when a new value is written in the device. No update event will be generated until the UDIS bit is cleared to '0'. But when

When an update event should occur, the counter will still be cleared to '0', and the count of the prescaler will also be cleared to '0' (but the prescaler

Coefficient remains unchanged). In addition, if the URS bit in the TIMx\_CR1 register is set (select an update request), set UG

Bit will generate an update event UEV, but the hardware does not set the UIF flag (ie, no interrupt or DMA request is generated); this is

To avoid clearing the counter in capture mode, both update and capture interrupts are generated. When an update event occurs,

All registers are updated, and the hardware sets the update flag bit (based on the URS bit) (in the TIMx\_SR register

UIF bit).

• The repetition counter is reloaded with the contents of the TIMx\_RCR register.

• The autoload shadow register is reset to the value of the preload register (TIMx\_ARR).

• The buffer of the prescaler is placed in the value of the preload register (the content of the TIMx\_PSC register).

The following figure shows some examples. When TIMx\_ARR = 0x36, the counter will operate at different clock frequencies.

902323

31

32 33 34 35 36

05

00 01 02 03 04

06 07

CK\_INT

CNT\_EN

Timer clock = CK\_CNT

Counter register

Counter overflow

Update event (UEV)

Update interrupt flag (UIF)

189.

1

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

322/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 345** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

241716

0034

0035

0000

0036

0001

0002

0003

CK\_INT

CNT\_EN

Timer clock = CK\_CNT

Counter register

Counter overflow

Update event (UEV)

Update interrupt flag (UIF)

190.

2

664226

0035

0000

0036

0001

CK\_INT

CNT\_EN

Timer clock = CK\_CNT

Counter register

Counter overflow

Update event (UEV)

Update interrupt flag (UIF)

191.

4

948833

1F

20

00

CK\_INT

Timer clock = CK\_CNT

Counter register

Counter overflow

Update event (UEV)

Update interrupt flag (UIF)

192.

N

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

323/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 346** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

618172

31

32

33 34

35 36

05

00 01 02 03 04

06 07

FF

36

Autoload register

Write new value in TIMx\_ARR

CK\_INT

CNT\_EN

Timer clock = CK\_CNT

Counter register

Counter overflow

Update event (UEV)

Update interrupt flag (UIF)

193.

ARPE = 0

(TIMx\_ARR

)

006973

F0

F1 F2 F3 F4 F5

05

00 01 02 03 04

06 07

F5

36

F5

36

Automatic reload of preload registers

CK\_INT

CNT\_EN

Timer clock = CK\_CNT

Counter register

Counter overflow

Update event (UEV)

Update interrupt flag (UIF)

Auto-reload shadow register

Write new value in TIMx\_ARR

194.

ARPE = 1

(TIMx\_ARR

)

**14.3.3**

'Time base unit' explains how the update event (UEV) is generated when the counter overflows, but in fact it can only repeat

Generated when the count reaches 0. This feature is very useful for generating PWM signals.

This means that every N count overflows, data is transferred from the preload register to the shadow register (TIMx\_ARR automatically

Reload register, TIMx\_PSC preload register, and capture/compare register in compare mode (TIMx\_CCRx),

N is the value in the TIMx\_RCR repeat count register.

The repetition counter decrements when any of the following conditions are true:

Each time the counter overflows in the up-counting mode, the repetition counter is automatically loaded, and the repetition rate is determined by TIMx\_RCR

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

324/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 347** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Definition of register value. When the update event is generated by software (by setting the UG bit in TIMx\_EGR) or by hardware

Generated by the slave mode controller, no matter what the value of the repetition counter is, an update event occurs immediately, and TIMx\_RCR

The contents of the register are reloaded into the repetition counter.

505243

counter

TIMX\_CNT

TIMX\_RCR=0

UEV

UEV

TIMX\_RCR=1

UEV

TIMX\_RCR=2

UEV

TIMX\_RCR=3

UEV

TIMX\_RCR=3

UEV

(by SW)

Update event:

Edge alignment mode

Count up

And resynchronize

The preload register is transferred to the active register and an update interrupt is generated.

195.

TIMx\_RCR

**14.3.4**

The counter clock can be provided by the following clock sources:

• Internal clock (CK\_INT).

**(CK\_INT)**

If the slave mode controller is disabled (SMS = 000 in the TIMx\_SMCR register), then CEN, DIR (TIMx\_CR1

Register) and UG bit (TIMx\_EGR register) are de facto control bits and can only be modified by software (UG bit is still

Is automatically cleared). As long as the CEN bit is written as '1', the prescaler clock is provided by the internal clock CK\_INT.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

325/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 348** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

The figure below shows the operation of the control circuit and the up-counter in normal mode without the prescaler.

977915

31

32

33 34

35

36

05

00 01

02

03

04

06

07

CK\_INT

CEN = CNT\_EN

UG

CNT\_INIT

Counter clock = CK\_CNT = CK\_PSC

Counter register

196.

1

**14.3.5**

**/**

Each capture/compare channel surrounds a capture/compare register (including shadow registers), including the captured input

Part (digital filtering, multiplexing and prescaler), and output part (comparator and output control). The following pictures are

An overview of capture/compare channels.

The input section samples the corresponding TIx input signal and generates a filtered signal TIxF. Then, one with polarity

The selected edge detector generates a signal (TIxFPx), which can be used as an input trigger from the mode controller or as

Capture control. This signal enters the capture register (ICxPS) through prescaler.

699822

IC1PS

filter

Down counter

ICF[3:0]

edge

Detector

CC1P

ICPS[1:0]

TI1

TIMx\_CCMR1

TIMx\_CCER

TI1F\_Rising

TI1F\_Falling

01

10

11

(Since mode controller)

0

1

f DTS

TI1F

0

1

divider

/1,/2,/4,/8

IC1

CC1E

CC1S[1:0]

TIMx\_CCMR1

TIMx\_CCER

TRC

TI2F\_Rising

TI2F\_Falling

(from channel 2)

(from channel 2)

TI2FP1

TI1FP1

TI1F\_ED

To slave mode controller

197.

/ ( 1

)

The output section generates an intermediate waveform OCxREF (high effective) as a reference, the end of the chain determines the pole of the final output signal

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

326/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 349** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Sex.

097990

MCU peripheral interface

APB bus

Capture/compare preload registers

high

8

CC1S[1]

OC1PE

TIMx\_CCMR1

UEV

Reading CCR1L

Reading CCR1H

CC1S[1]

CC1S[0]

IC1PS

CC1E

CC1G

TIMx\_EGR

read\_in\_progress

enter

mode

8

low

(Assumed to be

16 bits)

Capture/Compare Shadow Register

capture

counter

Comparators

capture\_transfer

compare\_transfer

write\_in\_progress

Output

mode

CNT> CCR1

CNT = CCR1

(From time base unit)

Write to CCR1H

Write to CCR1L

CC1S[0]

OC1PE

S

R

S

R

198.

/ 1

264837

ETRF

OC1CE OC1M[3:0]

TIM1\_CCMR1

TIM1\_BDTR

DTG[7:0]

CC1NE CC1E

TIM1\_CCER

CC1NP

TIM1\_CCER

MOE OSSI

TIM1\_BDTR

OSSR

CNT>CCR1

CNT=CCR1

OCxREF

OC5REF

Output mode

Controller

Output

Selector

OC1REF

dead zone

generate

OC1REFC

To main mode controller

OC1DT

OC1DTN

'0'

'0'

(1)

x0

01

11

11

10

0X

0

1

0

1

CC1P

TIM1\_CCER

OC1

OC1N

CC1NE CC1E TIM1\_CCER

Output

Enable

Circuit

Output

Enable

Circuit

199.

/ ( 1)

The capture/compare module consists of a preload register and a shadow register. The read and write process only operates preload registration

Device. In capture mode, the capture occurs on the shadow register, and then copied into the preload register. In comparison mode

Under the formula, the content of the preload register is copied to the shadow register, and then the content of the shadow register and the counter

Compare.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

327/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 350** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**14.3.6**

In input capture mode, when the corresponding edge on the ICx signal is detected, the current value of the counter is latched to capture/ratio

Compare register (TIMx\_CCRx). When a capture event occurs, the corresponding CCxIF flag (TIMx\_SR register)

Set to '1'. If interrupt or DMA operation is enabled, an interrupt or DMA operation will be generated. If something catches

When the CCxIF flag is already high when the device occurs, the repeated capture flag CCxOF (TIMx\_SR register) is set to '1'.

Write CCxIF = 0 to clear CCxIF, or read the captured data stored in the TIMx\_CCRx register to clear

CCxIF. Write CCxOF = 0 to clear CCxOF.

The following example shows how to capture the counter value into the TIMx\_CCR1 register on the rising edge of the TI1 input, step

as follows:

• Select valid input: TIMx\_CCR1 must be connected to the TI1 input, so write to the TIMx\_CCMR1 register

CC1S = 01, as long as CC1S is not '00', the channel is configured as an input, and the TM1\_CCR1 register

Becomes read-only.

• According to the characteristics of the input signal, configure the input filter to the required bandwidth (that is, when the input is TIx, the input filter controls

Bit is the ICxF bit in the TIMx\_CCMRx register). Assuming that the input signal is at most 5 internal clock cycles

For inter-time jitter, we must configure the filter bandwidth to be longer than 5 clock cycles. So we can (at f DTS frequency) connect

Continue sampling 8 times to confirm the last real edge transition on TI1, which is written in the TIMx\_CCMR1 register

IC1F = 0011.

• Select the effective conversion edge of the TI1 channel and write CC1P = 0 in the TIMx\_CCER register (in this example, it is rising

along).

• Configure input prescaler. In this example, we want to capture every valid level-shifting moment, so

The prescaler is disabled (write IC1PS = 00 in the TIMx\_CCMR1 register).

• Set CC1E = 1 in the TIMx\_CCER register to allow the capture counter value to be captured in the capture register.

• If necessary, enable the relevant interrupt request by setting the CC1IE bit in the TIMx\_DIER register, by setting

The CC1DE bit in the TIMx\_DIER register allows DMA requests.

When an input capture occurs:

• When a valid level transition occurs, the counter value is transferred to the TIMx\_CCR1 register.

• The C1IF flag is set (interrupt flag). When at least 2 consecutive captures occur, and CC1IF has not been cleared,

CC1OF is also set to 1.

• If the CC1IE bit is set, an interrupt will be generated.

• If the CC1DE bit is set, a DMA request will also be generated.

In order to deal with the capture overflow, it is recommended to read the data before reading the capture overflow flag, this is to avoid losing the read capture

Capture overflow information that may be generated after the overflow flag is obtained and before the data is read.

Note: By setting the corresponding CCxG bit in the TIMx\_EGR register, an input capture interrupt and/or DMA request can be generated by software.

**14.3.7**

In the output mode (CCxS = 00 in the TIMx\_CCMRx register), the comparison signal (OCxREF and the corresponding

OCx/OCxN is forced into valid or invalid state by software, and does not depend on the comparison between the output compare register and the counter

result.

Set the corresponding OCxM = 101 in the TIMx\_CCMRx register to force the output compare signal (OCxREF/OCx)

Is valid. In this way, OCxREF is forced to high level (OCxREF is always active high), and OCx is

To the opposite value of the CCxP polarity bit.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

328/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 351** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

For example: CCxP = 0 (OCx high level effective), then OCx is forced to high level.

Set OCxM = 100 in the TIMx\_CCMRx register to force the OCxREF signal low.

In this mode, the comparison between the TIMx\_CCRx shadow register and the counter is still in progress, and the corresponding flag will also

modified. Therefore, corresponding interrupts and DMA requests will still be generated. This will be in the output comparison mode section below

Introduction.

**14.3.8**

This function is used to control an output waveform or indicate when a given time has expired.

When the contents of the counter and the capture/compare register are the same, the output compare function does the following:

• Register the output compare mode (OCxM bit in the TIMx\_CCMRx register) and output polarity (TIMx\_CCER

The value defined by the CCxP bit in the device is output to the corresponding pin. The output pin can hold it when comparing and matching

Level (OCxM = 000), set to the active level (OCxM = 001), set to no active level (OCxM

= 010) or roll over (OCxM = 011).

• Set the flag bit in the interrupt status register (CCxIF bit in the TIMx\_SR register).

• If the corresponding interrupt mask (CCxIE bit in the TIMx\_DIER register) is set, an interrupt is generated.

• If the corresponding enable bit is set (CCxDE bit in the TIMx\_DIER register, the TIMx\_CR2 register

CCDS bit selects DMA request function), then generate a DMA request.

The OCxPE bit in TIMx\_CCMRx selects whether the TIMx\_CCRx register needs to use a preload register. Losing

In the compare mode, the update event UEV has no effect on the OCxREF and OCx outputs. Synchronization accuracy can be achieved

One counting cycle of the counter. The output compare mode (in single pulse mode) can also be used to output a single pulse.

Configuration steps of output comparison mode:

• Select counter clock (internal, external, prescaler)

• Write the corresponding data to the TIMx\_ARR and TIMx\_CCRx registers

• If an interrupt request is to be generated, set the CCxIE bit

• Select output mode:

**-** the Write OCxM = '011', and the CNT When the OCx The Toggle Output pins matching the CCRx

**-** the Write OCxPE = '0', unused PRELOAD

**–** Write CCxP = '0', active high

**-** the Write CCxE = '. 1', The Opening the OCx Output

• Set the CEN bit of the TIMx\_CR1 register to start the counter

The TIMx\_CCRx register can be updated by software at any time to control the output waveform, provided the pre-use is not used

Load register (OCxPE = '0', otherwise the shadow register of TIMx\_CCRx can only be updated in the next event

Is updated). The figure below gives an example.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

329/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 352** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

655133

0039

003A

003B

B200

B201

003A

B201

TIMX\_CNT

TIMX\_CCR1

OC1REF=OC1

Write B201h in CCR1 register

A match was detected on CCR1

If the interrupt is enabled, an interrupt is generated

200.

OC1

**14.3.9 PWM**

Pulse width modulation mode can generate a frequency determined by the TIMx\_ARR register and confirmed by the TIMx\_CCRx register

A signal with a fixed duty cycle.

Write '110' (PWM mode 1) or '111' (PWM mode 2) in the OCxM bit in the TIMx\_CCMRx register,

Each OCx output channel can be independently set to generate a PWM. The TIMx\_CCMRx register must be set

The OCxPE bit enables the corresponding preload register, and finally set the ARPE bit of the TIMx\_CR1 register, (in

(In up-counting mode) Preload register with automatic reload enabled. Only when an update event occurs, preload

The register can be transferred to the shadow register, so before the counter starts counting, it must be set by TIMx\_EGR

Use the UG bit in the register to initialize all registers.

The polarity of OCx can be set by the CCxP bit in the TIMx\_CCER register by software, it can be set to high power

Level effective or low level effective. CCxE, CCxNE, MOE, TIMx\_CCER and TIMx\_BDTR registers

The OSSI OSSR bit controls the OCx output enable. See the description of the TIMx\_CCERx register for details.

In PWM mode (mode 1 or mode 2), TIMx\_CNT and TIMx\_CCRx are always compared, (based on the count

The counting direction of the device) to determine whether it meets TIMx\_CCRx ≤ TIMx\_CNT or TIMx\_CNT ≤ TIMx\_CCRx.

Depending on the state of the CMS bit in the TIMx\_CR1 register, the timer can generate an edge-aligned PWM signal or center

Aligned PWM signal.

**PWM**

Count up when the DIR bit in the TIMx\_CR1 register is low. See: Up-counting mode below

An example of PWM mode 1. When TIMx\_CNT <TIMx\_CCRx, the PWM signal reference OCxRE is high,

Otherwise it is low. If the comparison value in TIMx\_CCRx is greater than the auto-reload value (TIMx\_ARR), OCxREF

Hold as '1'. If the comparison value is 0, OCxREF remains at '0'. Figure [201](https://translate.googleusercontent.com/translate_f#353) shows the edge pair when TIMx\_ARR = 8

Examples of homogeneous PWM waveforms.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

330/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 353** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

202517

0 1 2 3 4 5 6 7 8 0 1

þ1ÿ

þ0ÿ

Counter register

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

OCxREF

CCxIF

CCRx=4

CCRx=8

CCRx>8

CCRx=0

201.

PWM

(ARR = 8)

**14.3.10**

The basic timer TIM16/17 can output two complementary signals, and can manage the instantaneous turn-off and turn-on of the output. This

The period of time is usually called a dead zone, and the user should use the connected output devices and their characteristics (level-delayed delay, power

Source switch delay, etc.) to adjust the dead time.

Configure the CCxP and CCxNP bits in the TIMx\_CCER register to independently select the polarity for each output (main

Output OCx or complementary output OCxN).

The complementary signals OCx and OCxN are controlled by a combination of the following control bits: CCxE of the TIMx\_CCER register

And CCxNE bits, MOE, OISx, OISxN, OSSI and in the TIMx\_BDTR and TIMx\_CR2 registers

OSSR bits, see Table [50](https://translate.googleusercontent.com/translate_f#368) : Control bits of complementary output channels OCx and OCxN with brake function. especially,

When transitioning to the IDLE state (MOE drops to 0), the dead zone is activated.

Setting the CCxE and CCxNE bits at the same time will insert a dead zone, and if there is a brake circuit, then also set the MOE bit. each

Each channel has a 10-bit dead zone generator. The reference signal OCxREF can generate 2 output OCx and

OCxN. If OCx and OCxN are high effective:

• The OCx output signal is the same as the reference signal, except that its rising edge has a delay relative to the rising edge of the reference signal.

• The OCxN output signal is opposite to the reference signal, except that its rising edge has a delay relative to the falling edge of the reference signal.

If the delay is greater than the currently valid output width (OCx or OCxN), no corresponding pulse will be generated. The following pictures

The figure shows the relationship between the output signal of the dead zone generator and the current reference signal OCxREF. (Assuming CCxP = 0,

CCxNP = 0, MOE = 1, CCxE = 1 and CCxNE = 1).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

331/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 354** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

570898

delay

delay

OCxN

OCx

OCxREF

202.

554467

delay

OCxN

OCx

OCxREF

203.

817039

delay

OCxN

OCx

OCxREF

204.

The dead time delay of each channel is the same, which is configured by the DTG bit in the TIMx\_BDTR register. detailed

See [section 14.4.13](https://translate.googleusercontent.com/translate_f#371) Delay calculation in the TIM16/17 Brake and Dead Time Register (TIMx\_BDTR).

Redirect OCxREF to OCx or OCxN in output mode (forced, output compare or PWM), through configuration

The CCxE and CCxNE bits in the TIMx\_CCER register, OCxREF can be redirected to OCx or OCxN

Output.

This function can send a special waveform (such as PWM) on an output when the complementary output is at an invalid level

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

332/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 355** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Or static effective level). Another function is to make the two outputs at the inactive level at the same time, or at the effective level and band

Complementary output of dead zone.

Note: When only OCxN is enabled (CCxE = 0, CCxNE = 1), it will not be inverted, and will be effective immediately when OCxREF goes high. Eg,

If CCxNP = 0, then OCxN = OCxREF. On the other hand, when both OCx and OCxN are enabled (CCxE = CCxNE

= 1), OCx is valid when OCxREF is high, and OCxN is the opposite, OCxN becomes valid when OCxREF is low.

**14.3.11**

use

When using the brake function, according to the corresponding control bits (MOE/OSSI and OSSR bits in the TIMx\_BDTR register,

The OISx and OISxN bits in the TIMx\_CR2 register), the output enable signal and the invalid level will be modified. But no

Whenever, the OCx and OCxN outputs cannot be on the active level at the same time. See table 50 for details

The control bits of the complementary output channels OCx and OCxN. The brake source can be either a brake input pin or a time

Clock failure event. The clock failure event is generated by resetting the clock security system (CSS) in the clock controller.

System-wide chapter. After the system is reset, the brake circuit is disabled and the MOE bit is low. Set the TIMx\_BDTR register

The BKE bit can enable the brake function, the polarity of the brake input signal can be configured by the BKP bit in the same register

select. BKE and BKP can be modified at the same time. When writing BKE and BKP bits, there will be 1 before the actual writing

APB clock cycle delay, so you need to wait for one APB clock cycle before you can correctly read back the written

Bit.

Because the falling edge of MOE can be asynchronous, the actual signal (acting at the output) and the synchronization control bit (at TIMx\_BDTR

There is a resynchronization circuit between the registers). This resynchronization circuit will produce between the asynchronous signal and the synchronous signal

Health delay. In particular, if MOE = 1 is written when it is low, a delay must be inserted before reading it

Order) to read the correct value. This is because asynchronous signals are written and synchronous signals are read.

When braking occurs (the selected level appears at the brake input), there are the following actions:

• The MOE bit is asynchronously cleared, placing the output in an inactive state, an idle state, or a reset state (selected by the OSSI bit).

This feature is still effective when the MCU oscillator is turned off.

• When MOE = 0, the output level of each output channel is determined by the OISx bit in the TIMx\_CR2 register. Such as

If OSSI = 0, the timer releases the enable output, otherwise the enable output is always high.

• When using complementary outputs:

**– The** output is first put into a reset state, that is, an invalid state (depending on polarity). This is an asynchronous operation, even if

This function is also effective when the timer has no clock.

**–** If the clock of the timer still exists, the dead zone generator will take effect again, according to OISx after the dead zone

And the level indicated by the OISxN bit drive the output port. Even in this case, OCx and OCxN

It cannot be driven to a valid level at the same time. Note: Because of the resynchronization of MOE, the dead time is more than usual

Lower the length (about 2 ck\_tim clock cycles).

**–** If OSSI = 0, the timer releases the enable output, otherwise keeps the enable output; or when CCxE and CCxNE

When one of them goes high, the enable output goes high.

• If the BIE bit in the TIMx\_DIER register is set, the brake status flag (BIF in the TIMx\_SR register

When bit) is '1', an interrupt is generated. If the BDE bit in the TIMx\_DIER register is set, it generates

One DMA request.

• If the AOE bit in the TIMx\_BDTR register is set, the MOE bit is set by the next update event UEV

Set automatically; for example, this can be used for shaping. Otherwise, MOE remains low until it is set to '1' again; at this time,

This feature can be used for safety, you can connect the brake input to the power supply-driven alarm output, thermal sensing

Device or other security device.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

333/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 356** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Note: The brake input is level effective. Therefore, when the brake input is valid, the MOE cannot be set simultaneously (either automatically or through software).

At the same time, the status flag BIF cannot be cleared.

The brake is generated by the BRK input, its effective polarity is programmable, and is determined by the BKE bit in the TIMx\_BDTR register

Open.

In addition to brake input and output management, write protection is also implemented in the brake circuit to ensure the safety of the application. It allows

The user freezes several configuration parameters (dead zone length, OCx/OCxN polarity and prohibited status, OCxM configuration, brake enable

Energy and polarity). The user can select one of the three levels of protection through the LOCK bit in the TIMx\_BDTR register.

See [section](https://translate.googleusercontent.com/translate_f#371) 14.4.13 Brake and dead zone register (TIM16/17\_BDTR). The LOCK bit can only be modified after MCU reset

Change it once.

The following figure shows an example of output in response to braking:

199280

OCxN

OCx

OCxREF

( OCxN is not used, CCxP=0, OISx=1)

(

OCx

OCx

(

OCx

(

OCx

(CCxE=1, CCxP=0, OISx=1, CCxNE=0, CCxNP=0, OISxN=0)

delay

delay

delay

delay

delay

delay

(CCxE=1, CCxP=0, OISx=0, CCxNE=1, CCxNP=0, OISxN=1)

OCxN

OCx

(CCxE=1, CCxP=0, OISx=1, CCxNE=1, CCxNP=1, OISxN=1)

OCx

OCxN

(CCxE=1, CCxP=0, OISx=0, CCxNE=0, CCxNP=0, OISxN=1)

delay

delay

OCx

OCxN

OCx

OCxN

(CCxE=1, CCxP=0, CCxNE=0, CCxNP=0, OISx=OISxN=0 or OISx=OISxN=1)

BREAK(MOE)

(OCxN is not used, CCxP=0, OISx=0)

OCxN is not used, CCxP=1, OISx=1)

OCxN is not used, CCxP=1, OISx=0)

205.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

334/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 357** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**14.3.12**

The single pulse mode (OPM) is a special case of the aforementioned many modes. This mode allows the counter to respond to an stimulus and

After a programmable delay, a pulse with a programmable pulse width is generated. Can be started by slave mode controller

The counter generates a waveform in output compare mode or PWM mode. Set OPM in TIMx\_CR1 register

The bit will select the one-shot mode, which allows the counter to automatically stop when the next update event UEV occurs.

A pulse can be generated only when the comparison value is different from the initial value of the counter. Before starting (when the timer is waiting for a touch

Issue), must be configured as follows:

• Up counting method: CNT <CCRx ≤ ARR (in particular, 0 <CCRx)

408060

OC1

TI2

OC1REF

t DELAY

t PULSE

t

0

TIM1\_ARR

TIM1\_CCR1

counter

206.

For example, you need to detect a rising edge on the input pin of TI2, delay t DELAY, and then generate it on OC1.

Generate a positive pulse of length t PULSE .

Assume TI2FP2 as trigger 1:

• Set CC2S = '01' in the TIMx\_CCMR1 register to map TI2FP2 to TI2.

• Set CC2P = '0' in the TIMx\_CCER register to enable TI2FP2 to detect the rising edge.

• Set TS = '110' in the TIMx\_SMCR register and TI2FP2 is used as the trigger (TRGI) of the slave mode controller.

• Set SMS = '110' (trigger mode) in the TIMx\_SMCR register and TI2FP2 is used to start the counter.

• The OPM waveform is determined by the value written to the compare register (the clock frequency and counter prescaler must be considered)

• t DELAY is defined by the value written to the TIMx\_CCR1 register.

• t PULSE is defined by the difference between the autoload value and the comparison value (TIMx\_ARR-TIMx\_CCR1).

• Assume that a waveform from '0' to '1' is generated when a comparison match occurs, and is generated when the counter reaches the preload value

A waveform from '1' to '0'; first, set OC1M = '111' in the TIMx\_CCMR1 register, enter

Enter PWM mode 2; selectively enable the preload register as needed: set OC1PE in TIMx\_CCMR1

='1' and ARPE in the TIMx\_CR1 register; then fill in the comparison value in the TIMx\_CCR1 register, in

Fill the auto-load value in the TIMx\_ ARR register, modify the UG bit to generate an update event, and then wait for

An external trigger event on TI2. In this example, CC1P = '0'.

In this example, the DIR and CMS bits in the TIMx\_CR1 register should be set low. Because only one pulse is needed, all

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

335/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 358** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

It is necessary to set OPM = '1' in the TIMx\_CR1 register at the next update event (when the counter is loaded from

It will stop counting when the value rolls over to 0). Setting OPM = '0' in the TIMx\_CR1 register will select the repeat mode.

**14.3.13**

When the microcontroller enters the debug mode (Cortex TM -M0 core is stopped), according to DBG\_TIMx\_STOP in the DBG module

Setting, the TIMx counter can either continue normal operation or stop.

**14.4**

49. TIM16/17

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

TIM16/17\_CR1

TIM16/17 control register 1

0x00000000

[Section 14.4.1](https://translate.googleusercontent.com/translate_f#358)

0x04

TIM16/17\_CR2

TIM16/17 control register 2

0x00000000

[Section 14.4.2](https://translate.googleusercontent.com/translate_f#359)

0x0C

TIM16/17\_DIER

TIM16/17 interrupt enable register

0x00000000

[Section 14.4.3](https://translate.googleusercontent.com/translate_f#360)

0x10

TIM16/17\_SR

TIM16/17 status register

0x00000000

[Section 14.4.4](https://translate.googleusercontent.com/translate_f#361)

0x14

TIM16/17\_EGR

TIM16/17 event generation register 1

0x00000000

[Section 14.4.5](https://translate.googleusercontent.com/translate_f#363)

0x18

TIM16/17\_CCMR1

TIM16/17 capture/compare mode register 1

0x00000000

[Section 14.4.6](https://translate.googleusercontent.com/translate_f#364)

0x20

TIM16/17\_CCER

TIM16/17 capture/compare enable register

0x00000000

[Section 14.4.7](https://translate.googleusercontent.com/translate_f#367)

0x24

TIM16/17\_CNT

TIM16/17 counter

0x00000000

[Section 14.4.8](https://translate.googleusercontent.com/translate_f#369)

0x28

TIM16/17\_PSC

TIM16/17 prescaler register

0x00000000

[Section 14.4.9](https://translate.googleusercontent.com/translate_f#370)

0x2C

TIM16/17\_ARR

TIM16/17 auto-reload register

0x00000000

[Section 14.4.10](https://translate.googleusercontent.com/translate_f#370)

0x30

TIM16/17\_RCR

TIM16/17 repeat count register

0x00000000

[Section 14.4.11](https://translate.googleusercontent.com/translate_f#370)

0x34

TIM16/17\_CCR1

TIM16/17 capture/compare register 1

0x00000000

[Section 14.4.12](https://translate.googleusercontent.com/translate_f#371)

0x44

TIM16/17\_BDTR

TIM16/17 Braking and Dead Time Register

0x00000000

[Section 14.4.13](https://translate.googleusercontent.com/translate_f#371)

0x48

TIM16/17\_DCR

TIM16/17DMA control register

0x00000000

[Section 14.4.14](https://translate.googleusercontent.com/translate_f#374)

0x4C

TIM16/17\_DMAR

TIM16/17 full transfer address register

0x00000000

[Section 14.4.15](https://translate.googleusercontent.com/translate_f#375)

**14.4.1 TIM16/17**

**1(TIM16/17\_CR1)**

Offset address: 0x00

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

ARPE

UDIS

URS

OPM

CEN

CKD

Reserved

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

336/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 359** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9:8

CKD

rw

0x00

Clock division

Defined in the timer clock (CK\_INT) frequency and digital filter (ETR, TIx)

The division ratio between the sampling frequencies used.

00: t DTS = t CK\_INT

01: t DTS = 2 xt CK\_INT

10: t DTS = 4 xt CK\_INT

11: reserved

7

ARPE

rw

0x00

Auto-reload preload enable bit (Auto-reload preload enable)

0: TIMx\_ARR register is not buffered

1: TIMx\_ARR register buffer is valid

6:4

Reserved

Reserved, always read as 0.

3

OPM

rw

0x00

One pulse mode

0: The counter does not stop when an update event occurs

1: When the next update event occurs (clears the CEN bit), the counter stops

2

URS

rw

0x00

Update request source (Update request source) software selects through this bit

The source of the UEV event.

0: If update interrupt or DMA request is allowed, any one of the following events

An update interrupt or DMA request is generated:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller

1: If the update interrupt or DMA request is enabled, only the counter overflows/

Only underflow will generate update interrupt or DMA request.

1

UDIS

rw

0x00

Disable disable (Update disable) Software enables/disables UEV through this bit

Event generation

0: UEV is allowed. Update (UEV) events are generated by any of the following events:

-Counter overflow/underflow

-Set UG bit

-Updates generated from the mode controller with cached registers are loaded into them

Preload value.

1: UEV is prohibited. No update event occurs, shadow registers (ARR, PSC,

CCRx) keep their values. If the UG bit or slave mode controller is set

A hardware reset is issued and the counter and prescaler are reinitialized

0

CEN

rw

0x00

Counter enable

0: disable counter

1: Enable the counter.

In single pulse mode, when an update event occurs, CEN is automatically cleared.

Note: After the CEN bit is set by the software, the external clock, gating mode and encoder mode

To work. The trigger mode can automatically set the CEN bit through hardware.

**14.4.2 TIM16/17**

**2(TIM16/17\_CR2)**

Offset address: 0x04

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

337/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 360** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

OIS1N OIS1

CCDSCCUS Res. CCPC

Reserved

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

9

OIS1N

rw

0x00

Output Idle state 1 (OC1N output) (Output Idle state 1)

0: When MOE = 0, OC1N = 0 after the dead zone

1: When MOE = 1, OC1N = 1 after the dead zone

Note: After LOCK (TIMx\_BKR register) level 1, 2, or 3 has been set, the

The bit cannot be modified.

8

OIS1

rw

0x00

Output Idle state 1 (OC1 output) (Output Idle state 1)

0: When MOE = 0, if OC1N is implemented, then OC1 = 0 after the dead zone

1: When MOE = 1, if OC1N is implemented, then OC1 = after dead zone

1.

Note: After LOCK (TIMx\_BKR register) level 1, 2, or 3 has been set, the

The bit cannot be modified.

7:4

Reserved

Reserved, read as 0

3

CCDS

rw

0x00

Capture/compare DMA selection

0: Send CCx DMA request when CCx event occurs

1: When an update event occurs, the CCx DMA request is sent

2

CCUS

rw

0x00

Capture/compare control update selection (Capture/compare control update

selection)

0: When the capture/compare control bit is preloaded (CCPC = 1), only pass

Only set COMG bit will be updated

1: When the capture/compare control bit is preloaded (CCPC = 1), set

The COMG bit or TRGI will be updated when a rising edge is generated

Note: This bit is only effective when the channel has complementary output.

1

Reserved

Reserved, read as 0

0

CCPC

rw

0x00

Capture/compare preloaded control bit (Capture/compare preloaded con-

trol)

0: CCxE, CCxNE and OCxM bits are not preloaded

1: CCxE, CCxNE and OCxM bits are preloaded after being written, only

Only updated when the COM bit is set to '1'

Note: This bit is only effective when the channel has complementary output.

**14.4.3 TIM16/17**

**(TIM16/17\_DIER)**

Offset address: 0x0C

Reset value: 0x0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

338/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 361** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Res.

BIE

UIE

CC1

DE

UDE

COM

IE

CC1

IE

Reserved

Reserved

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

9

CC1DE

rw

0x00

Allow capture/compare 1 DMA request

0: CC1 DMA request prohibited

1: CC1 DMA request allowed

8

UDE

rw

0x00

Update DMA request allowed

0: Update DMA request prohibited

1: Update DMA request permission

7

BIE

rw

0x00

Braking interruption allowed

0: brake interruption prohibited

1: Braking interruption allowed

6

Reserved

Reserved, read as 0

5

COMIE

rw

0x00

COM interrupt enable

0: COM interrupt disabled

1: COM interrupt enable

4:2

Reserved

Reserved, read as 0

1

CC1IE

rw

0x00

Capture/Compare 1 interrupt enable

0: CC1 interrupt disable

1: CC1 interrupt enable

0

UIE

rw

0x00

Update interrupt enable

0: disable update interrupt

1: Update interrupt enable

**14.4.4 TIM16/17**

**(TIM16/17\_SR)**

Offset address: 0x10

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

CC1

OF

BIF

Res.

TIF

COM

IF

CC1

IF

UIF

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

rc\_w0

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:10

Reserved

Reserved, read as 0

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

339/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 362** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9

CC1OF

rc\_w0

0x00

Capture/Compare 1 overcap-capture (Capture/Compare 1 overcap-

ture flag)

This flag can be set by hardware only when the corresponding channel is configured for input capture

1. Write 0 to clear this bit.

0: No repeated capture;

1: When the value of the counter is captured in the TIMx\_CCR1 register, the CC1IF

The status is already 1.

8

Reserved

Reserved, read as 0

7

BIF

rc\_w0

0x00

Break interrupt flag

When the brake input is valid, the hardware is '1' for this position. If the brake input is invalid,

Then this bit can be cleared to "0" by software

0: No braking event

1: An effective level is detected on the brake input

6

TIF

rc\_w0

0x00

Trigger interrupt flag

When a trigger event occurs (when the slave mode controller is in a mode other than gating mode)

In other modes, a valid edge is detected at the TRGI input, or gated mode

Any edge below) is set to '1' by hardware. It is cleared to '0' by the software.

0: No trigger event is generated;

1: Trigger interrupt waiting for response.

5

COMIF

rc\_w0

0x00

COM interrupt flag

When a COM event is generated (when the capture/compare control bits: CCxE, CCxNE,

When OCxM has been updated), this bit is set by hardware. It is cleared by software.

0: No COM event is generated

1: COM interrupt waiting for response

4:2

Reserved

Reserved, read as 0

1

CC1IF

rc\_w0

0x00

Capture/Compare 1 interrupt flag

If channel CC1 is configured in output mode:

When the counter value matches the comparison value, this bit is set to '1' by hardware, and it is set by software

Clear '0'.

0: No match occurred

1: The value of TIMx\_CNT matches the value of TIMx\_CCR1

When the content of TIMx\_CCR1 is greater than TIMx\_ARR, the counter overflows

The CC1IF bit goes high.

If channel CC1 is configured for input mode:

When the capture event occurs, this bit is set to '1' by hardware, it is cleared to '0' by software or

Clear '0' by reading TIMx\_CCR1.

0: No input capture

1: The counter value has been captured (copied) to TIMx\_CCR1 (checked on IC1

An edge with the same polarity as the selected one is measured)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

340/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 363** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

UIF

rc\_w0

0x00

Update interrupt flag

This bit is set to '1' by hardware when an update event occurs. It is cleared to '0' by the software.

0: No update event is generated

1: Update interrupt waiting for response. When the register is updated, this bit is set to '1' by hardware:

-If UDIS = 0 in the TIMx\_CR1 register, the corresponding repeat counter

The value overflows (update if the repeat counter = 0) generates an update event;

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, the

The UG of the TIMx\_EGR register reinitializes the counter CNT

Update events are generated when;

-If UDIS = 0 and URS = 0 in the TIMx\_CR1 register, the counter

Generated when CNT is reinitialized by a triggered event.

**14.4.5 TIM16/17**

**1(TIM16/17\_EGR)**

Offset address: 0x14

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

BG

UG

COMG

CC1G

Reserved

TG

w

w

w

w

w

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, read as 0

7

BG

w

0x00

Break generation

This bit is set to '1' by the software and is used to generate a braking event, automatically by the hardware

Clear '0'.

0: No action

1: Brake event occurs, clear MOET, set the TIMx\_SR register

TIF = 1, if the corresponding interrupt and DMA are turned on, the corresponding interrupt and

DMA.

6

TG

w

0x00

Trigger generation

This bit is set by software to generate a trigger event, which is automatically generated by hardware

Clear '0'.

0: no action;

1: TIF=1 in the TIMx\_SR register, if the corresponding interrupt and DMA are enabled,

Then generate the corresponding interrupt and DMA.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

341/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 364** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

5

COMG

w

0x00

Capture/Compare control update generation (Capture/Compare control update

generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: It is possible to update CCxE, CCxNE and CCxE when the CCPC bit is set

OCxM bits

Note: This bit is only effective when the channel has complementary output.

4:2

Reserved

Reserved, read as 0

1

CC1G

w

0x00

Capture/Compare 1 produces

This bit is set to '1' by software to generate a capture/compare event.

Automatically clear '0'.

0: No action

1: Generate a capture/compare event on channel CC1

If channel CC1 is configured as output: set CC1IF = 1, if the corresponding

Interrupt and DMA, then generate the corresponding interrupt and DMA. If channel CC1 is equipped

Set as input, the current counter value is captured to the TIMx\_CCR1 register; set

Set CC1IF = 1, if the corresponding interrupt and DMA are turned on, the corresponding

Interrupt and DMA. If CC1IF is already 1, set CC1OF = 1.

0

UG

w

0x00

Generate an update event (Update generation)

This bit is set to '1' by software and automatically cleared to '0' by hardware.

0: No action

1: Re-initialize the counter and generate an update event. Note the prescaler

The counter of the counter is also cleared to '0' (but the prescaler coefficient is unchanged).

**14.4.6 TIM16/17**

**/**

**1(TIM16/17\_CCMR1)**

Offset address: 0x18

Reset value: 0x0000

The channel can be used for input (capture mode) or output (comparison mode), and the direction of the channel is defined by the corresponding CCxS. The deposit

The function of other bits of the device is different from the output mode. OCxx describes the function of the channel in output mode, ICxx describes

The function of the channel in output mode. Therefore, it must be noted that the function of the same bit in output mode and input mode is not

Same.

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

rw

rw

rw

CC1S

OC1M

OC1

PE

OC1

FE

IC1F

IC1PSC

Res.

Reserved

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

342/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 365** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:7

Reserved

Reserved, always read as 0.

6:4

OC1M

rw

0x00

Output compare 1 mode

These 3 bits define the action of the output reference signal OC1REF, and OC1REF

Determine the values ​​​​of OC1 and OC1N.

OC1REF is active high, while the effective levels of OC1 and OC1N are taken

Depends on the CC1P and CC1PN bits.

000: freeze. Output compare register TIMx\_CCR1 and counter

The comparison between TIMx\_CNT has no effect on OC1REF; (This mode is only used

To generate a time base).

001: Set channel 1 to effective level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is high.

010: Set channel 1 to invalid level when matching. When the counter TIMx\_CNT

When the value of is the same as capture/compare register 1 (TIMx\_CCR1), force

OC1REF is low.

011: Flip. When TIMx\_CCR1 = TIMx\_CNT, flip OC1REF

Level.

100: Forced to invalid level. Force OC1REF low.

101: Forced to effective level. Force OC1REF high.

110: PWM mode 1, channel 1 when TIMx\_CNT <TIMx\_CCR1

Is an effective level, otherwise it is an invalid level

111: PWM mode 2, when TIMx\_CNT <TIMx\_CCR1, the channel

1 is invalid level, otherwise it is valid level.

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = '00' (the channel is configured as an output), this bit cannot be modified.

Note 2: In PWM mode 1 or PWM mode 2, only when the comparison result changes

Or when switching from freeze mode to PWM mode in output compare mode, OC1REF

The level changes.

3

OC1PE

rw

0x00

Output compare 1 preload en-

able)

0: Disable the preload function of the TIMx\_CCR1 register, which can be written at any time

TIMx\_CCR1 register, and the newly written value takes effect immediately.

1: Turn on the preload function of the TIMx\_CCR1 register.

Preload register operation, the preload value of TIMx\_CCR1 is updated to

The incoming time is transferred to the current register.

Note 1: When the LOCK level is set to 3 (LOCK bit in the TIMx\_BDTR register)

And when CC1S = '00' (the channel is configured as an output), this bit cannot be modified.

Note 2: Only in single pulse mode (OPM = '1' in TIMx\_CR1 register),

To use the PWM mode without confirming the preload register, otherwise its action will not

determine.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

343/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 366** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

OC1FE

rw

0x00

Output compare 1 fast enable (Output compare 1 fast enable)

This bit is used to speed up the response of the CC output to trigger input events.

0: CC1 operates normally according to the value of the counter and CCR1, even if it is triggered

The device is turned on. When the trigger input has a valid edge, activate CC1

The minimum output delay is 5 clock cycles.

1: The valid edge input to the trigger acts as if a comparison match has occurred.

Therefore, OC is set to the comparison level regardless of the comparison result. Sample trigger

The delay between the effective edge of the device and the output of CC1 is reduced to 3 clock cycles.

OCFE only works when the channel is configured in PWM1 or PWM2 mode.

1:0

CC1S

rw

0x00

Capture/Compare 1 selection

These 2 bits define the direction of the channel (input/output) and the selection of input pins:

00: CC1 channel is configured as output;

01: CC1 channel is configured as input, IC1 is mapped on TI1;

10: CC1 channel is configured as input, IC1 is mapped on TI2;

11: The CC1 channel is configured as an input, and IC1 is mapped on the TRC. This mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

The TS bit of the controller).

Note: CC1S is only when the channel is closed (CC1E = '0' of the TIMx\_CCER register

) Is writable.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

344/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 367** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7:4

IC1F

rw

0x00

Input capture 1 filter (Input capture 1 ?lter)

These bits define the sampling frequency and digital filter length of the TI1 input. number

The word filter consists of an event counter, which will record after N events

Produces an output transition:

0000: no filter, sampling with f DTS

0001: sampling frequency f SAMPLING = f CK\_INT , N = 2

0010: Sampling frequency f SAMPLING = f CK\_INT , N = 4

0011: Sampling frequency f SAMPLING = f CK\_INT , N = 8

0100: sampling frequency f SAMPLING = f DTS , N = 6

0101: Sampling frequency f SAMPLING = f DTS , N = 8

0110: Sampling frequency f SAMPLING = f DTS , N = 6

0111: Sampling frequency f SAMPLING = DTS /4, N = 8

1000: Sampling frequency f SAMPLING = f DTS /8, N = 6

1001: Sampling frequency f SAMPLING = f DTS /8, N = 8

1010: Sampling frequency f SAMPLING = f DTS /16, N = 5

1011: Sampling frequency f SAMPLING = f DTS /16, N = 6

1100: Sampling frequency f SAMPLING = ff DTS /16, N = 8

1101: Sampling frequency f SAMPLING = f DTS /32, N = 5

1110: Sampling frequency f SAMPLING = f DTS /32, N = 6

1111: Sampling frequency f SAMPLING = f DTS /32, N = 8

3:2

IC1PSC

rw

0x00

Input capture 1 prescaler

These 2 bits define the prescaler coefficient of the CC1 input (IC1). When CC1E =

When '0' (in TIMx\_CCER register), the prescaler is reset.

00: No prescaler, every edge detected on the capture input port is touched

Send a capture

01: Capture is triggered every 2 events

10: Capture is triggered every 4 events

11: Trigger every 8 events

1:0

CC1S

rw

0x00

Capture/Compare 1 Select these 2 bits to define the direction of the channel (input/output), and input

Choice of foot:

00: CC1 channel is configured as output;

01: CC1 channel is configured as input, IC1 is mapped on TI1;

10: CC1 channel is configured as input, IC1 is mapped on TI2;

11: The CC1 channel is configured as an input, and IC1 is mapped on the TRC. This mode

Only works when the internal trigger input is selected (registered by TIMx\_SMCR

The TS bit of the controller).

Note: CC1S is only when the channel is closed (CC1E = '0' of the TIMx\_CCER register

) Is writable.

**14.4.7 TIM16/17**

**/**

**(TIM16/17\_CCER)**

Offset address: 0x20

Reset value: 0x0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

345/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 368** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

CC1P CC1E

CC1

NP

CC1

NE

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:4

Reserved

Reserved, always read as 0.

3

CC1NP

rw

0x00

Capture/Compare 1 output polarity (Capture/Compare 1 complemen-

tary output Polarity)

0: OC1N high effective

1: OC1N active low

2

CC1NE

rw

0x00

Capture/Compare 1 complementary output enable (Capture/Compare 1 comple-

mentary output enable)

0: Off-OC1N is invalid

1: On-the signal output by OC1N to the relevant output pin depends on the MOE,

OSSI, OSSR, OIS1, OIS1 and CC1E bits

1

CC1P

rw

0x00

Capture/Compare 1 output polarity

Channel CC1 is configured as output:

0: OC1 high effective

1: OC1 low active

The CC1 channel is configured as an input:

CC1P/CC1NP is used to select the signal TI1FP1 and

The polarity of TI2FP1, this bit IC1 is also the inverted signal of IC1.

00: No inversion/rising edge: capture occurs on the rising edge of TIxFP1 (capture mode

Type), TIxFP1 is not inverted;

01: Inverting/falling edge: Capture occurs on the falling edge of TIxFP1 (capture mode

Type), TIxFP1 is inverted;

10: Reserved, do not use this configuration;

11: No inversion/rising and falling edges: capture occurs on the rising edge of TIxFP1

With the falling edge (capture mode), TIxFP1 is not inverted.

0

CC1E

rw

0x00

Capture/Compare 1 output enable

CC1 channel is configured as output:

0: Off-OC1 output prohibited

1: On-the OC1 signal output to the corresponding output pin depends on the MOE,

OSSI, OSSR, OIS1, OIS1N and CC1E bits

The CC1 channel is configured as an input:

This bit determines whether the value of the counter can be captured into the TIMx\_CCR1 register.

0: capture prohibited

1: Capture enable

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

346/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 369** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

50.

OCx

OCxN

Control bit

Output status (1)

**MOE** bit **OSSI**

Bit

**OSSR**

Bit

**CCxE**

Bit

**CCxNE**

Bit

**OCx** output status

**OCxN** output status

1

X

0

0

0

Output prohibited (disconnected from timer)

OCx = 0, OCx\_EN = 0

Output prohibited (disconnected from timer)

OCxN = 0, OCxN\_EN = 0

0

0

1

Output prohibited (disconnected from timer)

OCx = 0, OCx\_EN = 0

OCxREF + polarity,

OCxN = OCxREF xor CCxNP,

OCxN\_EN = 1

0

1

0

OCxREF + polarity,

OCx = OCxREF xor CCxP,

OCx\_EN = 1

Output prohibited (disconnected from timer)

OCxN = 0, OCxN\_EN = 0

0

1

1

OCxREF + polarity + dead zone,

OCx\_EN=1

OCxREF Invert + Polarity + Dead Zone,

OCxN\_EN = 1

1

0

0

Output prohibited (disconnected from timer)

OCx = CCxP, OCx\_EN = 0

Output disabled (disconnected from timer) OCxN =

CCxNP, OCxN\_EN = 0

1

0

1

Off state (output is enabled and inactive

level)

OCx = CCxP, OCx\_EN = 1

OCxREF + polarity,

OCxN = OCxREF xor CCxNP,

OCxN\_EN = 1

1

1

0

OCxREF + polarity,

OCx = OCxREF xor CCxP,

OCx\_EN = 1

Off state (output is enabled and inactive

level)

OCxN = CCxNP, OCxN\_EN = 1

1

1

1

OCxREF + polarity + dead zone,

OCx\_EN = 1

OCxREF Invert + Polarity + Dead Zone,

OCxN\_EN = 1

0

0

X

0

0

Output prohibited (disconnected from timer)

0

0

1

Asynchronously: OCx = CCxP, OCx\_EN = 0, OCxN = CCxNP,

0

1

0

OCxN\_EN = 0;

0

1

1

If the clock exists: after a dead time

OCx = OISx, OCxN = OISxN,

It is assumed that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN.

1

0

0

Off state (output is enabled and inactive level)

1

0

1

Asynchronously: OCx = CCxP, OCx\_EN = 1, OCxN = CCxNP,

1

1

0

OCxN\_EN = 1;

1

1

1

If the clock exists: after a dead time

OCx = OISx, OCxN = OISxN,

It is assumed that OISx and OISxN do not both correspond to the effective levels of OCx and OCxN.

1. If the two outputs of a channel are not used (CCxE = CCxNE = 0), then OISx, OISxN, CCxP

And CCxNP must be cleared.

Note: The state of the external I/O pins where the pins are connected to complementary OCx and OCxN channels depends on the state of the OCx and OCxN channels

And GPIO and AFIO registers.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

347/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 370** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**14.4.8 TIM16/17**

**(TIM16/17\_CNT)**

Offset address: 0x24

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CNT

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CNT

rw

0x0000

Counter value

**14.4.9 TIM16/17**

**(TIM16/17\_PSC)**

Offset address: 0x28

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

PSC

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

PSC

rw

0x0000

Prescaler value

The clock frequency of the counter (CK\_CNT) is equal to f CK\_PSC /(PSC + 1).

PSC includes the update event (including the use of TIMx\_EGR register

UG or clear the counter by triggering the controller when configured in reset mode)

The value loaded into the current prescaler register when it is generated.

**14.4.10 TIM16/17**

**(TIM16/17\_ARR)**

Offset address: 0x2C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

ARR

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

ARR

rw

0x0000

Prescaler value

ARR contains the value to be loaded into the actual auto-reload register.

For details, refer to section [14.3.](https://translate.googleusercontent.com/translate_f#342) 1: Update and actions related to ARR.

When the value of auto-reload is empty, the counter does not work.

**14.4.11 TIM16/17**

**(TIM16/17\_RCR)**

Offset address: 0x30

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

348/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 371** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

REP

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:8

Reserved

Reserved, always read as 0.

7:0

REP

rw

0x00

Repetition counter value

After the preload register is enabled, these bits allow the user to set the compare register

The update rate (ie periodically transferred from the preload register to the current register

If the update interrupt is allowed, it will also affect the generation of update interrupt

s speed.

Each time the down counter REP\_CNT reaches 0, an update event will be generated

And the counter REP\_CNT starts counting from the REP value again. by

REP\_CNT is only reloaded when the periodic update event U\_RC occurs

REP value, so the new value written to the TIMx\_RCR register

It takes effect only when the periodic update event occurs. This means that in PWM mode,

(REP+1) corresponds to the number of PWM cycles in edge-aligned mode.

**14.4.12 TIM16/17**

**/**

**1(TIM16/17\_CCR1)**

Offset address: 0x34

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

CCR1

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

CCR1

rw

0x0000

Capture/Compare 1 value

If the CC1 channel is configured as an output:

CCR1 contains the value loaded into the current capture/compare 1 register (preload

value). If not selected in the TIMx\_CCMR1 register (OC1PE bit)

If you select the preload feature, the written value will be transferred to the current register immediately.

Otherwise, only when an update event occurs, this preload value is transmitted to the current capture

Get/Compare 1 register. The current capture/compare register participates in the same counter

TIMx\_CNT comparison and generates output signal on OC1 port.

If the CC1 channel is configured as an input:

CCR1 contains the count transmitted by the last input capture 1 event (IC1)

器值。 Device value. Device value.

**14.4.13 TIM16/17**

**(TIM16/17\_BDTR)**

Offset address: 0x44

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

349/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 372** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

OSSI

DTG

MOE AOE BKP BKE OSSR

LOCK

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Note: According to the lock setting, the AOE, BKP, BKE, OSSI, OSSR and DTG[7:0] bits can all be write-protected.

Configure them when writing to the TIMx\_BDTR register at one time.

**Bit**

**Field**

**Type**

**Reset**

**Description**

15

MOE

rw

0x00

Main output enable

When the brake input is valid, this bit is asynchronously cleared to '0' by hardware. According to AOE bit

The setting value of this bit can be cleared to 0 by software or set to 1 automatically. It only

Valid for channels configured as output.

0: Disable OC and OCN output or force to idle state

1: If the corresponding enable bit (TIMx\_CCER register is set)

CCxE, CCxNE bit), then OC and OCN output is turned on

ON the Details OC the For / OCN enable, See Section 14.4.7 [,](https://translate.googleusercontent.com/translate_f#367) TIM16 / 17 Capture

Obtain/compare enable register (TIM16/17\_CCER).

14

AOE

rw

0x00

Automatic output enable

0: MOE can only be set to '1' by the software

1: MOE can be set to '1' by the software or automatically set to '1' at the next update event

(If the brake input is invalid)

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to '1'

At this time, this bit cannot be modified.

13

BKP

rw

0x00

Brake input polarity (Break polarity)

0: Low level of brake input is effective

1: Brake input high level is effective

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to '1'

, The bit cannot be modified.

Note: Any write operation to this bit requires an APB clock delay before

kick in.

12

BKE

rw

0x00

Break enable

0: Brake input prohibited (BRK and CCS clock failure events)

1: Braking input enable (BRK and CCS clock failure events)

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to '1'

At this time, this bit cannot be modified.

Note: Any write operation to this bit requires an APB clock delay before

kick in.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

350/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 373** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

11

OSSR

rw

0x00

Off-state selection for Run in Run mode

mode)

This bit is used when MOE = 1 and the channel is a complementary output. No complement

There is no OSSR bit in the output timer. Refer to OC/OCN enable

Detailed description (Section [14.4.7](https://translate.googleusercontent.com/translate_f#367) , TIM1 capture/compare enable register

(TIMx\_CCER)).

0: Disable OC/OCN output when the timer does not work (OC/OCN enables

Can output signal = 0)

1: When the timer is not working, if CCxE = 1 or CCxNE = 1,

OC/OCN enable and output invalid level, then set OC/OCN enable input

Out signal = 1

Note: When the LOCK level (the LOCK bit in the TIMx\_BDTR register) is set to '2'

At this time, this bit cannot be modified.

10

OSSI

rw

0x00

Off-state selection fo Idle in running mode

mode)

This bit is used when the channel is output when MOE = 0.

Refer to the OC/OCN enable detailed description (Section [14.4.7](https://translate.googleusercontent.com/translate_f#367) , TIM1 capture/ratio

Compare enable register (TIMx\_CCER)).

0: Disable OC/OCN output when the timer does not work (OC/OCN enables

Can output signal = 0);

1: When the timer is not working, if CCxE = 1 or CCxNE = 1,

OC/OCN is forced to output idle level, set OC/OCN to enable output signal

No. = 1.

Note: When the LOCK level (LOC bit in the TIMx\_BDTR register) is set to '2'

At this time, this bit cannot be modified.

9:8

LOCK

rw

0x00

Lock configuration

This bit provides write protection to prevent software errors.

00: The lock is closed and the register is not write protected;

01: Lock level 1, cannot write DTG, TIMx\_BDTR register,

OISx/OISxN of BKE, BKP, AOE bits and TIMx\_CR2 register

Bit

10: Lock level 2, cannot write to everyone in lock level 1, nor write

Enter the CC polarity bit (when the relevant channel is set to output through the CCxS bit, CC

The polarity bit is the CCxP/CCNxP bit in the TIMx\_CCER register) and

OSSR/OSSI bit;

11: Lock level 3, you cannot write to everyone in lock level 2, nor can you write

Into the CC control bit (when the relevant channel is set to output through the CCxS bit, CC

The control bit is the OCxM/OCxPE bit in the TIMx\_CCMRx register);

Note: After the system is reset, the LOCK bit can only be written once, when written to TIMx\_BDTR

When registering, its content is frozen until reset.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

351/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 374** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

7:0

DTG

rw

0x00

Dead-time generator setup (Dead-time generator setup)

These bits define the duration of the dead zone inserted between complementary outputs. Assuming DT

Indicates its duration:

DTG[7: 5] = 0xx:

DT = (DTG[7: 0] + 1) × t dtg , t dtg = t DTS ;

DTG[7: 5] = 10x:

DT = (DTG[5: 0] + 1 + 64) × t dtg , t dtg = 2 × t DTS ;

DTG[7: 5] = 110:

DT = (DTG[4: 0] + 1 + 32) × t dtg , t dtg = 8 × t DTS ;

DTG[7: 5] = 111:

DT = (DTG[4: 0] + 1 + 32) × t dtg , t dtg = 16 × t DTS ;

Example: If t DTS = 125ns (8MHz), the possible dead time is:

125ns to 15875ns (step time is 125ns),

16µs to 31750ns (step time is 250ns),

32µs to 63µs (step time is 1µs),

64µs to 126µs (step time is 2µs).

Note: When the LOCK level (LOCK bit in the TIMx\_BDTR register) is set to 1,

At 2 or 3, these bits cannot be modified.

**14.4.14 TIM16/17DMA**

**(TIM16/17\_DCR)**

Offset address: 0x48

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

DBA

DBL

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:13

Reserved

Reserved, always read as 0.

12:8

DBL

rw

0x00

DMA continuous transfer length (DMA burst length)

These bits define the transfer length of DMA in continuous mode (when the

When the TIMx\_DMAR register is read or written, the timer

Continue to send), that is: define the number of bytes transmitted:

00000: 1 byte

00001: 2 bytes

00010: 3 bytes

...

...

10001: 18 bytes

7:5

Reserved

Reserved, always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

352/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 375** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

**Bit**

**Field**

**Type**

**Reset**

**Description**

4:0

DBA

rw

0x00

DMA base address (DMA base address) These bits define the DMA

The base address of the transfer (when reading or writing the TIMx\_DMAR register),

DBA is defined as the offset from the address where the TIMx\_CR1 register is located.

Eg:

00000: TIMx\_CR1

00001: TIMx\_CR2

00010: TIMx\_SMCR

...

Example: To complete the following transmission: DBL = 7, DBA = TIMx\_CR1

At this time, the transfer starts from the address of TIMx\_CR1 to/from 7 consecutive registers

Proceed.

**14.4.15 TIM16/17**

**(TIM16/17\_DMAR)**

Offset address: 0x4C

Reset value: 0x0000

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

DMAB

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:0

DMAB

rw

0x0000

DMA register for burst accesses

Reading or writing to the TIMx\_DMAR register will cause

Memory access operation:

TIMx\_CR1 address + (DBA + DMA index) × 4,

Among them:'TIMx\_CR1 address' is the control register 1 (TIMx\_CR1)

The address at;'DBA' is the base defined in the TIMx\_DCR register

Address;'DMA index' is the offset automatically controlled by DMA, it depends on

DBL defined in the TIMx\_DCR register.

Examples of how to use DMA concurrent operations

In this example, the concurrent function of the timer DMA is used to transfer the contents of the CCRx register in halfword mode.

Update to CCRx register. Follow the steps below:

1. Configure related DMA channels:

(a) The DMA channel device address is the DMAR register address

(b) The DMA channel memory address is the RAM buffer containing the data to be transferred to the CCRx register via DMA

District address

(c) Number of transmitted data = 3 (see note below)

(d) Prohibition in notification mode

2. Configure the DBA and DBL bits of the DCR register: DBL = 3 transfers, DBA = 0xE.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

353/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 376** |

UM\_MM32F003\_q\_Ver1.19

(TIM16/17)

3. Enable TIMx update DMA request (set UDE bit of DIER register)

4. Enable TIMx

5. Enable DMA channel

Note: In this example, all CCRx registers are updated at once. If the CCRx register needs to be updated twice, the number

The data quantity should be 6, and the RAM buffer should contain data1, data2, data3, data4, data5 and data6. The data is as follows

The process is transferred to the CCRx register: on the first update DMA request, data1 is transferred to CCR2, data2 is transferred to

CCR3, data3 are transferred to CCR4, and in the second DMA update interrupt request, data4 is transferred to CCR2, data5 are

Transferred to CCR3, data6 is transferred to CCR4.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

354/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 377** |

UM\_MM32F003\_q\_Ver1.19

(IWDG)

15

**(IWDG)**

Independent Watchdog (IWDG)

**15.1 (IWDG**

**)**

Built-in two watchdogs provide higher security, time accuracy and flexibility of use. Two watchdog devices

(Independent watchdog and window watchdog) can be used to detect and resolve faults caused by software errors; when the counter reaches a given

When the timeout value is exceeded, an interrupt is triggered (only applicable to window watchdog) or a system reset is generated.

The independent watchdog (IWDG) is driven by a dedicated low-speed clock (LSI), which is still effective even if the main clock fails.

The window watchdog is driven by the clock obtained after dividing the APB1 clock and detects the application through a configurable time window

The sequence is abnormal, too late or too early.

IWDG is most suitable for those who need watchdog as a main program, can work completely independently, and

Where time accuracy is low. WWDG is best for applications that require a watchdog to function in a precise timing window

sequence.

**15.2 IWDG**

• Free running down counter

• Clock is provided by an independent oscillator (can work in stop and standby mode)

• After the watchdog is activated, a reset is generated when the counter counts to 0x0000.

**15.3 IWDG**

The following figure is the functional block diagram of the independent watchdog module.

Write 0xCCCC in the key register (IWDG\_KR). Start the independent watchdog; the counter starts to reset from it

The value 0xFFF counts down. When the counter counts to the end of 0x000, a reset signal (IWGD\_RESET) is generated.

Whenever you write 0xAAAA in the key register IWDG\_KR, the value in IWDG\_RLR will be reset

Load into the counter to avoid watchdog reset.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

355/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 378** |

UM\_MM32F003\_q\_Ver1.19

(IWDG)

426852

Benevolent ᇴᆈಘ

IWDG\_PR

⣦ᘱᇴᆈಘ

IWDG\_SR

ヵ㻵䖭ᇴᆈಘ

IWDG\_RLR

ᇴᆈಘ

IWDG\_KR

8ս

㠶ren ಘ

LSI

(40kHz)

12ս䟽㻵䖭ᮠ٬

12 surrender

1.8V׋⭥४

VDD׋⭥४

IWDG༽ս

207.

Note: The watchdog function is in the V DD power supply area, that is, it can still work normally in shutdown and standby mode.

51.

(40KHz

(LSI))

Prescaler

**PR[2:0]** bits

Minimum time **RL[11:0]=0x000**

longest time

/4

0

0.1

409.6

/8

1

0.2

819.2

/32

3

0.8

3276.8

/64

4

1.6

6553.6

/128

5

3.2

13107.2

/256

(6 or 7)

6.4

26214.4

Note: These times are given according to the 40KHz clock. In fact, the frequency of the oscillator inside the MCU will be between 30KHz and 60KHz

To change.

In addition, even if the frequency of the oscillator is accurate, the exact timing still depends on the phase between the APB interface clock and the oscillator clock

Poor, so there will always be a complete oscillator period that is uncertain.

**15.3.1**

If the user activates the'Hardware Watchdog' function in the selection byte (please refer to the "Embedded Flash Memory" section), the system

After the system is powered on and reset, the watchdog will automatically start running; if the software does not register to the key before the counter counts

Write the corresponding value, the system will generate a reset.

**15.3.2**

The IWDG\_PR and IWDG\_RLR registers are write-protected. To modify the values ​​​​of of these two registers, you must first

Write 0x5555 in the IWDG\_KR register. Writing to this register with different values ​​​​will disrupt the sequence of operations.

Will be protected again. The reload operation (ie writing 0xAAAA) will also enable the write protection function.

The status register indicates whether the prescaler value and the down counter are being updated.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

356/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 379** |

UM\_MM32F003\_q\_Ver1.19

(IWDG)

**15.3.3**

When the microcontroller enters the debug mode (the CPU core is stopped), configure it according to DBG\_IWDG\_STOP in the debug module

When set, the IWDG counter can continue to work or stop. For details, see the chapter of the debugging module.

**15.4 IWDG**

52. IWDG

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

IWDG\_KR

Key register

0x00000000

[Section 15.4.1](https://translate.googleusercontent.com/translate_f#379)

0x04

IWDG\_PR

Prescaler register

0x00000000

[Section 15.4.2](https://translate.googleusercontent.com/translate_f#379)

0x08

IWDG\_RLR

Reload register

0x00000FFF

[Section 15.4.3](https://translate.googleusercontent.com/translate_f#380)

0x0C

IWDG\_SR

Status register

0x00000000

[Section 15.4.4](https://translate.googleusercontent.com/translate_f#381)

0x10

IWDG\_CR

Control register

0x00000000

[Section 15.4.5](https://translate.googleusercontent.com/translate_f#381)

**15.4.1**

**(IWDG\_KR)**

Offset address: 0x00

Reset value: 0x0000 0000 (reset in standby mode)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

w

Reserved

KEY

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

KEY

w

0x00

Key value (only write register, read value is 0x0000) (Key value)

Software must write 0xAAAA at certain intervals, otherwise, when the counter is 0

, The watchdog will reset.

Write 0x5555 to allow access to IWDG\_PR and IWDG\_RLR

Memory.

Write 0xCCCC to start watchdog operation.

**15.4.2**

**IWDG\_PR**

Offset address: 0x04

Reset value: 0x0000 0000 (reset in standby mode)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

357/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 380** |

UM\_MM32F003\_q\_Ver1.19

(IWDG)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

Reserved

Reserved

PR

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 3

Reserved

Always read as 0.

2: 0

PR

rw

0x00

Prescaler divider

These bits have write protection settings. When setting the bits to select the counter

The prescaler of the clock. To change the prescaler factor, the IWDG\_SR register

The PVU bit must be 0.

000: prescaler factor = 4

100: prescaler factor = 64

001: prescaler factor = 8

101: prescaler factor = 128

010: Prescaler factor = 16

110: prescaler factor = 256

011: prescaler factor = 32

111: prescaler factor = 256

Note: Reading this register will return the prescaler value from the V DD voltage domain.

If the write operation is in progress, the value read back may be invalid. Therefore, only

When the PUV bit of the IWDG\_SR register is 0, the read value is valid.

**15.4.3**

**(IWDG\_RLR)**

Offset address: 0x08

Reset value: 0x0000 0FFF (reset in standby mode)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

RL

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 12

Reserved

Always read as 0.

11: 0

RL

rw

0xFFF

Watchdog counter reload value

These bits are write-protected. Used to define the reload value of the watchdog counter, every

When writing 0xAAAA to the IWDG\_KR register, the reload value will be transferred

Sent to the counter. The counter then counts down from this value. Janitor

The dog timeout period can be calculated by the value of the second reload and the clock prescaler value.

Note: Reading this register will return the prescaler value from the V DD voltage domain. Such as

If the write operation is in progress, the value read back may be invalid. Therefore, only if

When the RUV bit of the IWDG\_SR register is 0, the read value is valid.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

358/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 381** |

UM\_MM32F003\_q\_Ver1.19

(IWDG)

**15.4.4**

**(IWDG\_SR)**

Offset address: 0x0C

Reset value: 0x0000 0000 (do not reset in standby mode)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

r

r

Reserved

Reserved

PVU

RVU

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 2

Reserved

Always read as 0.

1

RVU

r

0x00

Watchdog counter reload value update (Watchdog counter reload

value update)

This bit is set by hardware to indicate that the reload value update is in progress. when

After the reload update in the V DD field is completed, this bit is cleared to 0 by the hardware (most

5 more 40KHz oscillator cycles are required) The reload value is only in RVU

It can only be updated after the bit is cleared to '0'.

0

PVU

r

0x00

Watchdog prescaler value update

This bit is set by hardware to indicate that the prescaler value update is in progress. when

After the update of the prescaler value in the V DD domain is completed, this bit is cleared to '0' by the hardware (most

5 more 40KHz oscillator cycles are required) The prescaler value is only available in RVU

It can only be updated after the bit is cleared to '0'.

Note: If multiple reload values ​​​​or prescaler values ​​​​are used in the application, the preload must be changed again after the RVU bit is cleared

The load value can only be changed again after the PVU bit is cleared. However, after the prescaler and/or reload value update, there is no need to wait

RVU or PVU reset, you can continue to execute the following code. (Even in low power mode, the secondary write operation will continue to be performed

to make)

**15.4.5 IWDG**

**(IWDG\_CR)**

Address offset: 0x10

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

Reserved

IRQ\_

CLR

IRQ\_

SEL

rw

rw

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

359/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 382** |

UM\_MM32F003\_q\_Ver1.19

(IWDG)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:2

Reserved

Reserved, always read as 0.

1

IRQ\_CLR

rw

0x00

IWDG interrupt clear

1: Write 1 to clear interrupt

0: No operation

0

IRQ\_SEL

rw

0x00

IWDG overflow operation selection

1: Generate an interrupt after overflow

0: Reset after overflow

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

360/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 383** |

UM\_MM32F003\_q\_Ver1.19

(WWDG)

16

**(WWDG)**

Window Watchdog (WWDG)

**16.1 WWDG**

Window watchdogs are often used to monitor application deviations caused by external disturbances or unforeseen logic conditions

Software failure caused by running sequence. Unless the value of the down counter is refreshed before the T6 bit becomes 0, the watchdog circuit

When the preset time period is reached, an MCU reset is generated. After the down counter reaches the value of the window register

Previously, if the 7-bit down counter value (in the control register) was refreshed, an MCU reset will also be generated.

This indicates that the down counter needs to be refreshed within a limited time window.

**16.2 WWDG**

• Programmable free running down counter

• Condition reset:

**–** When the value of the down counter is less than 0x40, (if the watchdog is activated), a reset is generated.

**–** When the down counter is reloaded outside the window, (if the watchdog is activated), a reset is generated

• If the watchdog is enabled and interrupts are enabled, an early wakeup interrupt (EWI) is generated when the down counter is equal to 0x40,

It can be used to reload the counter to avoid WWDG reset.

**16.3 WWDG**

If the watchdog is enabled (the WDGA bit in the WWDG\_CR register is set to '1'), and when it is 7 bits (T[6:0])

When the down counter rolls over from 0x40 to 0x3F ((T6 bit is cleared), a reset is generated. If the software is in the counter value

Reload the counter when it is greater than the value in the window register, a reset will be generated.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

361/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 384** |

UM\_MM32F003\_q\_Ver1.19

(WWDG)

055172

-

W6 W5 W4 W3 W2 W1 W0

WDGA T6

T5

T4

T3

T2

T1

T0

CMP

Watchdog control register (WWDG\_CR)

Watchdog configura on register (WWDG\_CFR)

6-bit downcounter (CNT)

WDG prescaler

(WDGTB)

Comparator

= 1 when

T6:0> W6:0

Write WWDG\_CR

PCLK1

(from RCC clock controller)

RESET

208.

The application program must periodically write to the WWDG\_CR register during normal operation to prevent the MCU from resetting.

Only when the counter value is less than the value of the window register can write operation. Stored in WWDG\_CR register

The value must be between 0xFF and 0xC0:

• Start watchdog

After the system is reset, the watchdog is always off, and the WDGA bit in the WWDG\_CR register can be turned on

The watchdog, after which it cannot be turned off again unless a reset occurs.

• Control down counter

The down counter is in a free-running state. Even if the watchdog is disabled, the down counter continues to count down. Be a janitor

When the dog is enabled, the T6 bit must be set to prevent an immediate reset.

The T[5:0] bit contains the number of times before the watchdog generates a reset; the delay time before reset is between a minimum value and a

The maximum value varies because the prescaler value is unknown when writing to the WWDG\_CR register.

The upper limit of the window is included in the configuration register (WWDG\_CFR): to avoid reset, the down counter must be

When the value is less than the value of the window register and is greater than 0x3F, it is reloaded. The above figure describes the operation of the window register.

Cheng.

Another way to reload the counter is to use the early wake-up interrupt (EWI). Set the WWDG\_CFR register

The WEI bit turns on the interrupt. When the down counter reaches 0x40, this interrupt is generated, and the corresponding interrupt service routine (ISR)

Can be used to load counters to prevent WWDG reset. Write '0' in the WWDG\_SR register to clear this

Break.

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

**16.4**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

362/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 385** |

UM\_MM32F003\_q\_Ver1.19

(WWDG)

The following figure shows the linearity between the 6-bit count value loaded into the watchdog counter (CNT) and the watchdog delay time

Relationship (in mS). This figure can be used as a reference for quick calculations without taking into account the time deviation. Such as

If you need higher accuracy, you can use the calculation formula provided in the figure below.

Warning: When writing to the WWDG\_CR register, always set the T6 bit to '1' to avoid an immediate reset.

399420

**WDGTB**

**0**

**1**

**2**

**3**

ᴰሿ厵ᰦ٬

**113µS**

**227µS**

**455µS**

**910µS**

ᴰབྷ䎵ᰦ٬

**7.28mS**

**14.56mS**

**29.12mS**

**58.25mS**

൘PCLK1 = 36MHzᰦⲴᴰሿ-ᴰབྷ䎵ᰦ٬

T PCLK1 ˖ APB1ԕmSѪ অ ս Ⲵᰦ䫏䰤䳄

T WWDG ˖ WWDG䎵ᰦᰦ䰤

T WWDG = T PCLK1 × 4096 × 2 WDGTB × (T[5:0]+1)

(mS)

Check

㇑㇇䎵ᰦ Ⲵ ྲл˖

༽Ս

T6ս

T[6:0]CNTs are out of order

ᰦ䰤

デエョᯠ ᡧᯠデਓ

3Fh

W[6:0]

209.

**16.5**

When the microcontroller enters the debug mode (the CPU core is stopped), according to the DBG\_WWDG\_STOP configuration in the debug module

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

363/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 386** |

UM\_MM32F003\_q\_Ver1.19

(WWDG)

When set, the WWDG counter can continue to work or stop. See the chapter about debugging modules for details.

**16.6 WWDG**

53. WWDG

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

WWDG\_CR

Control register

0x0000007F

[Section 16.6.1](https://translate.googleusercontent.com/translate_f#386)

0x04

WWDG\_CFR

Configuration register

0x0000007F

[Section 16.6.2](https://translate.googleusercontent.com/translate_f#386)

0x08

WWDG\_SR

Status register

0x00000000

[Section 16.6.3](https://translate.googleusercontent.com/translate_f#387)

**16.6.1**

**(WWDG\_CR)**

Offset address: 0x00

Reset value: 0x0000 007F

rw

rw

WDGA

T

Reserved

Reserved

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:8

Reserved

Reserved, always read as 0

7

WDGA

rw

0x00

Activation bit

This bit is set to '1' by software, but can only be cleared to '0' by hardware after reset. when

When WDGA = 1, the watchdog can generate a reset.

0: watchdog prohibited

1: Start watchdog

6:0

T

rw

0x7F

7-bit counter (MSB to LSB) (7-bit counter)

These bits are used to store the counter value of the watchdog. Every (4096x2 WDGTB )

PCLK1 period is decremented by 1. When the counter value changes from 40h to 3Fh (T6 changes

0), a watchdog reset is generated.

**16.6.2**

**(WWDG\_CFR)**

Offset address: 0x04

Reset value: 0x0000 007F

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

364/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 387** |

UM\_MM32F003\_q\_Ver1.19

(WWDG)

rw

rw

rw

WDGTB

W

Reserved

Reserved

EWI

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:10

Reserved

Reserved, always read as 0

9

EWI

rw

0x00

Early wakeup interrupt

If this bit is set to '1', an interrupt will be generated when the counter value reaches 40h.

This interrupt can only be cleared by hardware after reset.

8:7

WDGTB

rw

0x00

Time base (Timer base)

The time base of the prescaler can be modified as follows:

00: CK timer clock (PCLK1 divided by 4096) divided by 1

01: CK timer clock (PCLK1 divided by 4096) divided by 2

10: CK timer clock (PCLK1 divided by 4096) divided by 4

11: CK timer clock (PCLK1 divided by 4096) divided by 8

6:0

W

rw

0x7F

7-bit window value

These bits contain the window value used for comparison with the down counter.

**16.6.3**

**(WWDG\_SR)**

Offset address: 0x08

Reset value: 0x0000 0000

rc\_w0

EWIF

Reserved

Reserved

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:1

Reserved

Reserved, always read as 0

0

EWIF

rc\_w0

0x00

Early wakeup interrupt flag

When the counter value reaches 40h, this bit is set by hardware. It must pass

Software writes '0' to clear.

Writing '1' to this bit has no effect. If the interrupt is not enabled, this bit will also be set to '1'.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

365/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 388** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

17

**(SPI)**

Serial Peripheral Interface (SPI)

**17.1 SPI**

The SPI interface is widely used for board-level communication between different devices, such as extended serial Flash, ADC, etc. Many IC manufacturers

All produced devices support SPI interface.

SPI allows the MCU to communicate with external devices in full-duplex, synchronous, and serial mode. Application software can check status

Or SPI interrupt to communicate.

**17.2**

• Fully compatible with Motorola's SPI specification

• Support DMA request

• Support full duplex synchronous transmission on 3 lines

• 16-bit programmable baud rate generator

• Support master mode and slave mode

• 8-byte receive/transmit FIFO

• SPI as the master mode SPI clock can be up to pclk/2 (pclk is APB clock), as the slave mode

The fastest SPI clock can be up to pclk/4

• Programmable clock polarity and phase

• Programmable data sequence, MSB first or LSB first

• Supports one master and multiple slave operations

• Supports simultaneous transmission and reception of data lengths of 1 to 32 bits

• Except for 8-bit data transmission and reception, the remaining 1 to 32-bit data transmission and reception only support LSB mode, not MSB mode.

• Supports 8 transmit buffers and receive buffers corresponding to each configuration data bit (Data size)

• Interrupt drive operation

**– The** sending end is empty, the sending end overflows

**– The** received data is valid, and the data at the receiving end overflows

**–** Complete reception in SPI master mode, the transmitter is empty

**17.3 SPI**

**17.3.1**

The block diagram of SPI is shown below

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

366/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 389** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

020437

APB

Bridge

DMA

Bus

Interface

Logic

Register

Logic

TX

BUFF

ER

Control Logic

Tx shift register

Rx shift register

Baud Rate

Generator

M

aster block

Master/Slave Select

TX shift register

RX shift register Slave block

RX

BUFF

ER

SPI

210. SPI

SPI supports simultaneous reception and transmission of 1 to 32 bits of data. SPI can be configured in slave mode or in a host ring

Configure the main mode in the environment. Four possible timing relationships can be selected by configuring clock polarity CPOL and phase CPHA

system. Programmable data sequence, MSB first or LSB first.

The same clock is used for the sending and receiving parts. Data is output on the rising or falling edge of the clock, the opposite of SCLK

Data is latched on the valid edge. Because SPI is used to exchange data, the data must be read after the transfer, even if the data

Is not valid data. In SPI mode, the clock phase and polarity of the master and the slaves that communicate with it must be the same.

Usually SPI is connected to external devices through 4 pins:

• MISO: Master input/slave output pin. This pin sends data in slave mode and receives data in master mode

according to.

• MOSI: Master device output/slave device input pin. This pin sends data in master mode and receives data in slave mode

according to.

• SCK: serial port clock, used as the output of the master device and the input of the slave device.

• NSS: Select from the device. This is an optional pin used to select the master/slave device. Its function is used as a'piece

Select the pin', so that the master device can communicate with a specific slave device individually, to avoid conflicts on the data line. Slave NSS

The pin can be driven by the master device as a standard IO. Once enabled, the NSS pin can also be used as an output

Pin, and pulled low when the SPI is set to master mode; at this time, all NSS pins are connected to the master device NSS pin

The SPI device will detect a low level.

The figure below is an example of a single master and single slave device interconnection.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

367/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 390** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

236771

MSBit

LSBit

MSBit

LSBit

**8-bit shift register**

**8-bit shift register**

**SPI clock**

**generator**

**Not used if NSS is**

**managed by software**

**NSS (1)**

**SCK**

**SCK**

**MOSI**

**MOSI**

**MISO**

Slave

Master

**V DD**

**MISO**

**NSS (1)**

211.

use

The MOSI pins are connected to each other, and the MISO pins are connected to each other. In this way, data is transferred serially between the master and slave (MSB bit is

before).

Communication is always initiated by the master device. The master device sends data to the slave device through the MOSI pin, and the slave device uses the MISO

The feet return data. This means that the data output and data input of full-duplex communication are synchronized with the same clock signal;

The signal is provided by the master device through the SCK pin.

The CPOL and CPHA bits of the SPI\_CCTL register can be combined into four possible timing relationships. CPOL (Clock

Polarity) bit controls the idle state level of the SCK clock during no data transmission. This bit is for devices in master mode and slave mode

All works. If CPOL is cleared to '0', the SCK pin remains low in the idle state, that is, low power between two transmissions

Level; if CPOL is set to '1', the SCK pin remains high in the idle state, that is, high between the two transfers.

If the CPHA (clock phase) bit is set to '1', the first data bit is locked on the second clock edge of the SCK clock

Save (CPOL bit is 0 is the falling edge, CPOL bit is 1 is the rising edge), and the first received

The data bits are sampled. SPI changes the serial data when the first SCK clock transfer is transferred (the clock is now idle)

Change in the opposite direction of the status), and capture data on the next edge.

If the CPHA (clock phase) bit is cleared to '0', the first data bit is locked on the first clock edge of the SCK clock

Save (CPOL bit is 0 is the falling edge, CPOL bit is 1 is the rising edge), and the first received

The data bits are sampled. The SPI captures serial data when the first SCK clock transfer is transferred (the clock is now idle)

Changes in the opposite direction of the state), the data changes on the next edge.

The combination of CPOL clock polarity and CPHA clock phase selects the clock edge for data capture. Figure [212 shows the](https://translate.googleusercontent.com/translate_f#392) SPI transfer

The 4 kinds of CPHA and CPOL bit combinations lost. This figure can explain the SCK pin, MISO pin, and

Timing diagram of the master or slave directly connected to the MOSI pin.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

368/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 391** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

For the sensitivity to board-level delay in high-speed transmission mode, TXEDGE and RXEDGE in the SPI\_CCTL register

The control bits perform time adjustments on the transmit phase and receive samples.

• In slave mode, when TXEDGE is 1, the transmission data is immediately sent to the data bus, used in high-speed mode (SPBRG

= 4); when 0, send data is sent to the data bus after a valid clock edge, used in low-speed mode (SPBRG

> 4).

• In master mode, when RXEDGE is 1, the data is sampled in the middle of the transmitted data bit; when it is 0, the data bit is transmitted

Sample data at the trailing edge of the clock (for high-speed mode)

1. Before changing the CPOL/CPHA bit, the SPIEN bit must be cleared to disable the SPI.

2. Master and slave must be configured to the same timing mode.

3. The idle state of SCK must be consistent with the polarity specified by the SPI\_CCTL register (when CPOL is 1, SCK should be pulled up when idle

It is high level; when CPOL is 0, SCK should be pulled low when idle).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

369/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 392** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

240135

MSBit

MSBit

LSBit

LSBit

**CPHA=1**

**CPHA=0**

CPOL = 1

CPOL = 0

MISO

(from master)

MOSI

(from slave)

NSS

(to slave)

CAPTURE STROBE

MSBit

MSBit

LSBit

CPOL = 1

CPOL = 0

(from master)

CAPTURE STROBE

MISO

(from slave)

MOSI

LSBit

(to slave)

NSS

212.

According to the LSBFE bit in the SPI\_CCTL register, the MSB or LSB can be used first when outputting data bits.

According to the SPILEN bit of the SPI\_CCTL register, each data frame can be 7 or 8 bits. Selected data

The frame format is valid for both sending and/or receiving.

In addition, register SPI\_EXTCTL can be set to configure the data frame length from 1 to 32 bits. When using this configuration, you need to configure

Set: DW8\_32 bit of SPI\_GCTL register is '0', and LSBFE bit of SPI\_CCTL register is configured as

'1', the SPILEN bit is configured as '1'. In conjunction with DMA data transmission, the data length of DMA needs to be configured to 8 bits.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

370/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 393** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**17.3.2 SPI**

In the slave configuration, the SCK pin is used to receive the serial clock from the master device. Settings in the SPI\_SPBRG register

Does not affect the data transfer rate.

1. Set the SPILEN bit to define the data frame format as 7 or 8 bits

2. Select the CPOL and CPHA bits to define the phase relationship between data transmission and serial clock. To ensure the correct number

According to the transmission, the CPOL and CPHA bits of the slave device and the master device must be configured in the same way.

3. The frame format (MSB first or LSB first depends on the LSBFE bit in the SPI\_CCTL register) must be

The equipment is the same.

4. Clear the MDOE bit and set the SPIE bit to make the corresponding pin work in SPI mode. In this configuration, MOSI

Pin is data input, MISO pin is data output.

In a write operation, data words are written to the transmit buffer in parallel.

When the slave device receives the clock signal and the first data bit appears on the MOSI pin, the transmission process starts, the first

The bits are sent out. The remaining bits are loaded into the shift register. When the data in the transmit buffer is transferred to the shift register

At this time, the TX\_INTF flag in the SPI\_INTSTAT register is set. If the SPI\_INTEN register is set

The TXIEN bit will generate an interrupt.

For the receiver, when the data reception is completed:

• The data in the shift register is transferred to the receive buffer, and the RX\_INTF flag in the SPI\_INTSTAT register is set

Set.

• If the RXIEN bit in the SPI\_INTEN register is set, an interrupt is generated.

After the last sampling clock edge, the RXNE bit is set to '1' and the data byte received in the shift register is transferred

To the receive buffer. When reading the SPI\_RXREG register, the SPI device returns this value.

**17.3.3 SPI**

In the main configuration, the serial clock is generated on the SCK pin.

Configuration steps

1. Define the serial clock baud rate through the SPI\_SPBRG register.

2. Select the CPOL and CPHA bits to define the phase relationship between data transmission and serial clock.

3. Set the SPILEN bit to define the 8 or 7-bit data frame format.

4. Configure the LSBFE bit of the SPI\_CCTL register to define the frame format.

5. If only receiving without sending data, configure the SPI\_RNDNR register to define the number of bytes that need to be received.

6. The MDOE and SPIE bits must be set.

In this configuration, the MOSI pin is the data output, the MISO pin is the data input, and the NSS is the slave device selection signal input.

Out.

When a byte is written into the transmit buffer, the transmit process begins. When the first data bit is sent, the data word is

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

371/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 394** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

Through the internal bus) into the shift register, and then serially moved out to the MOSI pin; MSB first or LSB first, take

It depends on the LSBFE bit in the SPI\_CCTL register. TX\_INTF when data is transferred from the transmit buffer to the shift register

The flag will be set. If the TXIEN bit in the SPI\_INTEN register is set, an interrupt will be generated.

For the receiver, when the data transfer is complete:

• The data in the shift register is transferred to the receive buffer, and the RX\_INTF flag in the SPI\_INTSTAT register is set

Set.

• If the RXIEN bit in the SPI\_INTEN register is set, an interrupt is generated.

After the last sampling clock edge, the RXNE bit is set to '1' and the data byte received in the shift register is transferred

To the receive buffer. When reading the SPI\_RXREG register, the SPI device returns this value.

If you only receive without sending data, after receiving the number of bytes defined by RXDNR, the RXMATCH\_INTF bit is set to '1',

Indicates that all data has been received and no clock signal is sent in the main mode.

**17.3.4**

For the convenience of software operation, the application can be monitored by 4 current status flags and 7 interrupt status flags

The status of the SPI bus. The current status flag is read-only and is automatically set and cleared by hardware. Interrupt status flag is in the event

It is set when it occurs and generates a CPU interrupt when the interrupt is enabled, which is cleared by software.

There is an 8-byte transmit buffer and receive buffer inside the SPI respectively, according to the setting of DW8\_32 bit of SPI\_GCTL

Set, the CPU can read and write 1 or 4 bytes at a time. According to the setting of DW8\_32, the sending and receiving buffers have

A byte or a valid data status flag.

54. SPI

classification

Status flag

Buffer and signal status

Interrupted state

TX\_INTF

According to the DW8\_32 setting, there is at least one space for valid data,

Can complete a write operation of the send data register

RX\_INTF

According to the DW8\_32 setting, there is at least one valid data,

Can complete a read operation of the received data register

UNDERRUN\_INTF

Send buffer is empty and repeated

RXOERR\_INTF

Receive buffer is not empty and is covered

RXMATCH\_INTF

Not empty, the last data is transferred to the receive buffer

RXFULL\_INTF

The receive buffer is full and no new data can be received

TXEPT\_INTF

The transmit buffer is empty and can no longer be sent

Current status

RXAVL\_4BYTE

Receive buffer has more than 4 bytes of valid data

TXFULL

Send buffer full

TXEPT

Transmit buffer empty

RXAVL

Receive buffer is not empty, at least one byte can be received

When TXTLF of SPI\_GCTL register is 00, TX\_INTF is set when there is 1 or more free data space in the transmit buffer

Bit; when TXTLF is 01, TX\_INTF is set when the transmit buffer has more than half of the free space.

When the RXTLF of the SPI\_GCTL register is 00, and the receive buffer has one or more valid data, RX\_INTF is set;

When RXTLF is 01, RX\_INTF is set when the receive buffer has more than half of the valid data.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

372/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 395** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**17.3.5**

The baud rate is the frequency of the generated SCLK, which is generally the frequency division of PCLK. BRG is a 16-bit baud rate generator.

The SPBREG register controls the counting period of the 16-bit counter.

Provide the desired baud rate and f pclk (the frequency of the APB module), use the formula shown in the table below to calculate the approximate number assignment

The value is given to the SPBRG register. The X in the table below is equal to the value of the SPBRG register (2 ∼ 65535).

55.

mode

formula

SPI mode

Baud rate = f pclk /X

**17.3.6**

With **DMA**

**SPI**

In order to reach the maximum communication speed, the buffer needs to be sent to the SPI in time, and the data in the receive buffer must also be

It must be read in time to prevent overflow. To facilitate high-speed data transmission, SPI implements a simple request/application

Answer the DMA mechanism.

When the DMAEN bit in the SPI\_GCTL register is set, the SPI module can issue a DMA data transfer request.

Both the DMA requests of the transmit buffer and the receive buffer are enabled by DMAEN.

• When sending, when the TXTLF of the SPI\_GCTL register is 00, the transmit buffer has 1 or more free numbers

DMA transfer request is made when there is space; when TXTLF is 01, the transmit buffer has more than half of the free space

The DMA request is made. There is only one DMA transfer per request. Each DMA transfer data size and send

Each data size of the send buffer is determined by DW8\_32.

• When receiving, when the RXTLF of the SPI\_GCTL register is 00, the receive buffer has 1 or more valid numbers

A DMA transfer request is made when the data is available; when RXTLF is 01, the receive buffer has more than half of the valid data.

Make a DMA request. There is only one DMA transfer per request. Each DMA transfer data size and receive buffer

Each data size of the punch is determined by DW8\_32.

**17.4**

56. SPI

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

SPI\_TXREG

Transmit data register

0x00000000

[Section 17.4.1](https://translate.googleusercontent.com/translate_f#396)

0x04

SPI\_RXREG

Receive data register

0x00000000

[Section 17.4.2](https://translate.googleusercontent.com/translate_f#396)

0x08

SPI\_CSTAT

Current status register

0x00000001

[Section 17.4.3](https://translate.googleusercontent.com/translate_f#396)

0x0C

SPI\_INTSTAT

Interrupt status register

0x00000000

[Section 17.4.4](https://translate.googleusercontent.com/translate_f#397)

0x10

SPI\_INTEN

Interrupt enable register

0x00000000

[Section 17.4.5](https://translate.googleusercontent.com/translate_f#399)

0x14

SPI\_INTCLR

Interrupt clear register

0x00000000

[Section 17.4.6](https://translate.googleusercontent.com/translate_f#400)

0x18

SPI\_GCTL

Global control register

0x00000004

[Section 17.4.7](https://translate.googleusercontent.com/translate_f#401)

0x1C

SPI\_CCTL

General control register

0x00000008

[Section 17.4.8](https://translate.googleusercontent.com/translate_f#403)

0x20

SPI\_SPBRG

Baud rate generator

0x00000002

[Section 17.4.9](https://translate.googleusercontent.com/translate_f#404)

0x24

SPI\_RXDNR

Receive data number register

0x00000001

[Section 17.4.10](https://translate.googleusercontent.com/translate_f#404)

0x28

SPI\_NSSR

Slave Chip Select Register

0x000000FF

[Section 17.4.11](https://translate.googleusercontent.com/translate_f#405)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

373/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 396** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x2C

SPI\_EXTCTL

Data control register

0x00000008

[Section 17.4.12](https://translate.googleusercontent.com/translate_f#405)

**17.4.1**

**SPI\_TXREG**

Offset address: 0x00

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

TXREG

TXREG

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:0

TXREG

rw

0x0000

0000

Transmit data register (Transmit data register)

The valid data bits are controlled by DW8\_32.

0: Only the lower 8 bits are valid

1: TXREG[31:0] are valid

**17.4.2**

**SPI\_RXREG**

Offset address: 0x04

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

RXREG

RXREG

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:0

RXREG

r

0x0000

0000

Receive data register (Receive data register)

The valid data bits are controlled by DW8\_32.

0: Only the lower 8 bits are valid

1: RXREG[31:0] are valid

This register can be read and written.

**17.4.3**

**SPI\_CSTAT**

Offset address: 0x08

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

374/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 397** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

Reset value: 0x0000 0001

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

RXAVL

TXFULL

TXEPT

RXAVL\_

4BYTE

RXFADDR

TXFADDR

r

r

r

r

r

r

r

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:12

Reserved

Always read as 0.

11:8

RXFADDR

r

0x00

Current receive buffer address (Receive FIFO address)

7:4

TXFADDR

r

0x00

Current transmit buffer address (Transmit FIFO address)

3

RXAVL\_4BYTE r

0x00

The valid data in the receive buffer reaches 4 bytes flag bit (Receive

available 4 byte data message)

1: There are more than 4 bytes in the receive buffer

0: The data in the receive buffer is less than 4 bytes

2

TXFULL

r

0x00

Transmitter FIFO full status bit

1: Transmit buffer is full

0: Transmit buffer is not full

1

RXAVL

r

0x00

Receive valid byte data information bit (Receive available byte data

message)

This bit is set when the receiving buffer receives a complete byte of data.

1: The receiving buffer has received a valid byte of data

0: Receiver buffer empty

This bit is read-only and is automatically set and cleared by hardware.

0

TXEPT

r

0x01

Transmitter empty bit

1: Transmit buffer and transmit shift register are empty

0: The sender is not empty

This bit is read-only and is automatically set and cleared by hardware.

**17.4.4**

**SPI\_INTSTAT**

Offset address: 0x0C

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

375/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 398** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

TXEPT\_

INTF

RXFULL

\_INTF

RX

MATCH

\_INFT

RXO

ERR\_

INFT

UNDE

RRUN

\_INTF

RX\_

INTF

TX\_

INTF

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:7

Reserved

Always read as 0.

6

TXEPT\_INTF

r

0x00

Transmitter empty interrupt flag bit (Transmitter empty interrupt flag bit)

It is automatically set by hardware and cleared by writing the TXEPT\_ICLR bit in the INTCLR register.

1: Transmit buffer and TX shift register are empty

0: The sender is not empty

Note: This bit is the interrupt status signal and TXEPT is the status signal.

5

RXFULL\_INTF r

0x00

Receiver buffer full interrupt flag bit (RX FIFO full interrupt flag bit)

It is automatically set by hardware and cleared by writing the RXFULL\_ICLR bit in the INTCLR register.

1: RX buffer is full

0: RX buffer is not full

4

RXMATCH\_

INTF

r

0x00

Receive the specified byte number interrupt flag (Receive data match the

RXDNR number, the receive process will be completed

and generate the interrupt)

Set automatically by hardware, write to INTCLR register RXMATCH\_ICLR bit clear

except.

1: Received the number of bytes specified by the RXDNR register

0: The number of bytes specified by the RXDNR register is not completed

3

RXOERR\_

INTF

r

0x00

Receive overrun error interrupt flag (Receive overrun error inter-

rupt flag bit)

Set automatically by hardware, write RXOERR\_ICLR bit in INTCLR register to clear

except.

1: overflow error

0: No overflow error

2

UNDERRUN\_

INTF

r

0x00

SPI underrun interrupt flag bit (SPI underrun interrupt flag bit)

Automatically set by hardware, write UNDERRUN\_ICLR bit in INTCLR register

Clear.

1: underflow error

0: No underflow error

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

376/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 399** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

RX\_INTF

r

0x00

Receive data available interrupt flag (Receive data available inter-

rupt flag bit)

Set by hardware automatically, write RX\_ICLR bit in INTCLR register to clear.

When the receiving buffer receives a complete byte of data.

1: There is valid byte data in the receiving buffer

0: Receiver buffer empty

0

TX\_INTF

r

0x00

Transmit buffer effective interrupt flag (the number of bytes sent

Data) (Transmit FIFO available interrupt flag bit)

It is automatically set by hardware and cleared by writing the TX\_ICLR bit in the INTCLR register.

1: The sender buffer is valid

0: Invalid sender buffer

**17.4.5**

**SPI\_INTEN**

Offset address: 0x10

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

TXEPT\_

IEN

RXFULL

\_IEN

RX

MATCH

\_IEN

RXO

ERR\_

IEN

UNDE

RRUN

\_IEN

RX\_

IEN

TX\_

IEN

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:7

Reserved

Always read as 0.

6

TXEPT\_IEN

rw

0x00

Transmit empty interrupt enable bit (Transmit empty interrupt enable bit)

1: Interrupt enable

0: Disable interrupt

5

RXFULL\_IEN rw

0x00

Receive buffer full interrupt enable bit (Receive FIFO full interrupt

enable bit)

1: Interrupt enable

0: Disable interrupt

4

RXMATCH\_

IEN

rw

0x00

Receive the specified byte number interrupt enable bit (Receive data complete inter-

rupt enable bit)

1: Interrupt enable

0: Disable interrupt

3

RXOERR\_

IEN

rw

0x00

Receiver overflow error interrupt enable bit (Overrun error interrupt en-

able bit)

1: Interrupt enable

0: Disable interrupt

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

377/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 400** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

UNDERRUN\_

IEN

rw

0x00

SPI slave mode underflow interrupt enable bit (SPI slave mode) (Transmitter

underrun interrupt enable bit (SPI slave mode only))

1: Interrupt enable

0: Disable interrupt

1

RX\_IEN

rw

0x00

Receive data interrupt enable bit (Receive FIFO interrupt enable

bit)

1: Interrupt enable

0: Disable interrupt

0

TX\_IEN

rw

0x00

Transmit FIFO empty interrupt enable bit (Transmit FIFO empty interrupt enable bit

enable bit)

1: Interrupt enable

0: Disable interrupt

**17.4.6**

**SPI\_INTCLR**

Offset address: 0x14

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

w

w

w

w

w

w

w

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

TXEPT\_

ICLR

RXFULL

\_ICLR

RX

MATCH

\_ICLR

RXO

ERR\_

ICLR

UNDE

RRUN

\_ICLR

RX\_

ICLR

TX\_

ICLR

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:7

Reserved

Always read as 0.

6

TXEPT\_ICLR w

0x00

Transmitter empty interrupt clear bit

bit)

1: Interrupt clear

0: Interrupt is not cleared

5

RXFULL\_ICLR w

0x00

Receiver buffer full interrupt clear bit (Receiver buffer full interrupt

clear bit)

1: Interrupt clear

0: Interrupt is not cleared

4

RXMATCH\_

ICLR

w

0x00

Receive the specified interrupt interrupt clear bit (Receive completed interrupt

clear bit)

1: Interrupt clear

0: Interrupt is not cleared

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

378/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 401** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

RXOERR\_

ICLR

w

0x00

Receiver overflow error interrupt clear bit (Overrun error interrupt clear bit

bit)

1: Interrupt clear

0: Interrupt is not cleared

2

UNDERRUN\_

ICLR

w

0x00

SPI slave mode underflow interrupt clear bit (SPI slave mode) (Transmitter

underrun interrupt clear bit (SPI slave mode only))

1: Interrupt clear

0: Interrupt is not cleared

1

RX\_ICLR

w

0x00

Receive interrupt clear bit (Receive interrupt clear bit)

1: Interrupt clear

0: Interrupt is not cleared

0

TX\_ICLR

w

0x00

Transmit FIFO empty interrupt clear bit (Transmitter FIFO empty inter-

rupt clear bit)

1: Interrupt clear

0: Interrupt is not cleared

**17.4.7**

**SPI\_GCTL**

Offset address: 0x18

Reset value: 0x0000 0004

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

DW8\_

32

NSS

DMAEN

TXTLF

RXTLF

RXEN TXEN MODE INTEN SPIEN

NSS

TOG

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:13

Reserved

Always read as 0.

12

NSSTOG

rw

0x00

Slave select toggle automatically from the device selection signal

1: NSS signal automatically flips after each data is transmitted

0: NSS signal does not flip

Note: This bit is only valid in master mode

11

DW8\_32

rw

0x00

Send and receive data register valid data selection (Valid byte or double-

word data select signal)

0: Only the lower 8 bits are valid

1: 32-bit data is valid

Note: Whether it is through the CPU or DMA, you must use the specified data format to access

ask.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

379/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 402** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

10

NSS

rw

0x00

Hardware or software controls NSS output in main mode (NSS select signal

that from software or hardware)

0: controlled by the NSSR register value

1: Automatic hardware control during data transmission

9

DMAEN

rw

0x00

DMA access mode enable for reception and transmission

0: DMA mode disabled

1: DMA mode enable

8:7

TXTLF

rw

0x00

TX buffer trigger DMA request edge selection (TX FIFO trigger

level bit)

00: It is performed when there is more than or equal to 1 free data space in the transmit buffer

DMA request or send interrupt request

01: DMA is performed when the transmit buffer has more than half of the free space

Request or send an interrupt request

1x: reserved

Note: When DW8\_32 is 0, one data space represents 1 byte;

At 1, a data space represents 4 bytes.

6:5

RXTLF

rw

0x00

Receive buffer trigger DMA request edge selection (RX FIFO trigger

level bit)

00: DMA is performed when the receive buffer has 1 or more valid data

Request or receive interrupt request

01: Perform DMA when the receive buffer has more than half of the valid data.

Request or receive interrupt request

1x: reserved

Note: When DW8\_32 is 0, one valid data represents 1 byte;

At 1, a valid data represents 4 bytes.

4

RXEN

rw

0x00

Receive enable bit (Receive enable bit)

1: Receive enable

0: Reception prohibited. You can also clear the RX buffer

Note: When the SPI only works in the host receive mode, txen must be set to

0.

3

TXEN

rw

0x00

Transmit enable bit

1: Send enable

0: transmission prohibited. You can also clear the TX buffer

Note: When sending and receiving in the master mode at the same time.

2

MODE

rw

0x01

Master mode bit

1: Master mode (serial clock generated by internal BRG)

0: Slave mode (serial clock from external master)

1

INTEN

rw

0x00

SPI interrupt enable bit (SPI interrupt enable bit)

1: Enable SPI interrupt

0: Disable SPI interrupt

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

380/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 403** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

SPIEN

rw

0x00

SPI select bit (SPI select bit)

0: SPI disabled (reset state)

1: SPI enable

**17.4.8**

Use **SPI\_CCTL**

Offset address: 0x1C

Reset value: 0x0000 0008

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

TX

EDGE

RX

EDGE

LSB

FE

CPOL

CPH

ASEL

SPI

LEN

CPHA

HISPD

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:8

Reserved

Always read as 0.

7

HISPD

rw

0x00

High speed slave mode

1: SPI runs in high-speed slave mode

0: SPI running in low speed slave mode

Note: This bit is only valid in slave mode.

6

CPHASEL

rw

0x00

CPHA register polarity selection (CPHA polarity select)

1: When CPHA is 1, the first data bit is sampled from the second clock edge

Start

0: When CPHA is 0, it means that the first data bit sample is from the first clock edge

Start

5

TXEDGE

rw

0x00

Transmit data edge select bit (slave mode) (Transmit data edge select)

1: Send data to the data bus immediately

Can be used in high-speed mode (SPBRG = 4).

0: Transmit data is sent to the data bus after a valid clock edge

Can be used in low speed mode (SPBRG> 4).

4

RXEDGE

rw

0x00

Receive data sampling clock edge selection bit (master mode) (Receive data edge

select)

1: Data is sampled at the trailing clock edge of the transmitted data bits (used in high-speed mode)

0: sample data in the middle of the transmitted data bits

3

SPILEN

rw

0x01

SPI data length bit (SPI character length bit)

This bit takes effect after DW8\_32 is set (DW8\_32=0).

1: 8-bit data (default)

0: 7-bit data

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

381/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 404** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

LSBFE

rw

0x00

LSBFE: LSB first enable bit (LSI first enable bit)

1: The least significant bit of data transmission or reception is first

0: The most significant bit of data transmission or reception is first

1

CPOL

rw

0x00

Clock polarity select bit

1: Clock is high in idle state (between transfers)

0: The clock is low in idle state (between transfers)

0

CPHA

rw

0x00

Clock phase select bit (Clock phase select bit)

1: The first data bit sampling starts from the first clock edge

0: The first data bit is sampled from the second clock edge

**17.4.9**

**SPI\_SPBRG**

Offset address: 0x20

Reset value: 0x0000 0002

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

SPBRG

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

Reserved

Always read as 0.

15:0

SPBRG

rw

0x02

The SPI baud rate control register is used to generate the baud rate (SPI baud rate con-

trol register for baud rate)

Baud rate formula:

Baud rate = fpclk/SPBRG

(F pclk /is the APB clock frequency)

Note: Do not write 0 and 1 to this register.

**17.4.10**

**SPI\_RXDNR**

Offset address: 0x24

Reset value: 0x0000 0001

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

382/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 405** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

RXDNR

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:16

Reserved

Always read as 0.

15:0

RXDNR

rw

0x01

This register is used to store the number of bytes that need to be received in the next receiving process

(The register is used to hold a count of to be received

bytes in next receive process)

The value of this register is valid when the SPI is the host receiving mode. The default value is 1.

This register value is changed by the MCU write value.

Note: Do not write '0' value to this register.

**17.4.11**

**SPI\_NSSR**

Offset address: 0x28

Reset value: 0x0000 00FF

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

Reserved

NSS

**Bit**

**Field**

**Type**

**Reset**

**Description**

15:1

Reserved

Always read as 0.

0

NSS

rw

0xFF

Chip select output signal in main mode. Active low, this bit is invalid in slave mode (Chip

select output signal in Master mode).

0: Slave is selected

1: Slave is not selected

**17.4.12**

**SPI\_EXTCTL**

Offset address: 0x2C

Reset value: 0x0000 0008

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

383/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 406** |

UM\_MM32F003\_q\_Ver1.19

(SPI)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

EXTLEN

Reserved

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:5

Reserved

Always read as 0.

4:0

EXTLEN

rw

0x08

Control SPI data length

0 0000: 32 bit

0 0001: 1 bit

0 0010: 2 bit

0 0011: 3 bit

…

1 1100: 28 bit

1 1101: 29 bit

1 1110: 30 bit

1 1111: 31 bit

Note: Only valid when the DW8\_32 bit of the SPI\_GCTL register is '0', and

The LSBFE bit of the SPI\_CCTL register must be configured as '1', SPILEN

The bit must also be configured as '1'.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

384/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 407** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

18 **I2C**

**(I2C)**

I2C interface (I2C)

**18.1 I2C**

The I2C (inter-chip) bus interface connects the microcontroller and the serial I2C bus. It provides multi-host functions and controls all I2C

Bus specific timing, protocol, arbitration and timing.

The I2C bus is a two-wire serial interface, where two-wire serial data (SDA) and serial clock (SCL) lines are connected

Transfer information between bus devices. Each device has a unique address identification, and can be sent as a

Or receiver. In addition to the transmitter and receiver, the device can also be regarded as a master or slave when performing data transmission.

The host is a device that initiates data transmission on the bus and generates a clock signal that allows transmission. At this time, any addressed device

The pieces are considered slaves.

I2C can work in standard mode (data transmission rate is 0 ∼ 100 Kbps), fast mode (maximum data transmission rate

400 Kbps).

**18.2 I2C**

• Parallel bus I2C bus protocol converter

• Half-duplex synchronous operation

• Support master-slave mode

• Support 7-bit address and 10-bit address

• Support standard mode 100 Kbps, fast mode 400 Kbps

• Generate Start, Stop, Resend Start, answer Acknowledge signal detection

• Only one host is supported in master mode

• There are 2 bytes of send and receive buffer respectively

• Added glitch-free circuits on SCLI and SDAI

• Support DMA operation

• Support interrupt and query operations

**18.3 I2C**

**18.3.1**

When the bus is idle, SCL and SDA are simultaneously pulled high by external pull-up resistors. When the host starts data

When transmitting, a start condition must be generated first. When the SCL line is high, the SDA line switches from high to low

Change means the starting condition. When the host finishes the transmission, it sends a stop condition. The SCL line is high and the SDA line is low

Switching the level to high indicates the stop condition. The following figure shows the timing diagram of the start and stop conditions. During data transmission,

When SCL is 1, SDA must remain stable.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

385/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 408** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

755520

S

Start Condition

P

SCL

Change of Data

Allowed

Data line Stable

Data Valid

Change of Data

Allowed

Stop Condition

SDA

213.

**18.3.2**

I2C has two address formats: 7-bit address format and 10-bit address format.

**7**

The following figure shows that the first 7 bits (bit 7: 1) of a byte sent after the start condition (S) are the slave address, the lowest bit

(Bit 0) is the data direction bit. When bit 0 is 0, it means that the master writes data to the slave, and 1 means that the master reads data from the slave.

389349

S

A6

A5

A4

A3

A2

A1

A0

R/W

ACK

MSB

LSB

Slave Address

sent by slave

S = START condition

R/W = Read/Write Pulse

ACK = Acknowledge

214.7

**10**

In the 10-bit address format, send 2 bytes to transfer the 10-bit address. The bit description of the first byte sent is as follows

Bottom: The first 5 bits (bit 7:3) are used to inform the slave that the next 10 bits are transmitted. The last two words of the first byte

The bit (bit 2:1) bit 9:8 of the slave address, the least significant bit (bit 0) is the data direction bit (R/W). Second transmission

One byte is the lower eight bits of the 10-bit address.

The details are shown below:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

386/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 409** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

611694

S

0

A9 A8

A7 A6 A5 A4 A3 A2 A1 A0 ACK

ACK

1

1

1

1

Reserved for 10-Bit Address

sent by slave

sent by slave

ACK = Acknowledge

R/W = Read/Write Pulse

S = START condition

R/W

215. 10

The following table defines the special purpose and reserved address of the first byte of I2C:

57. I2C

Slave address

**R/W** bit

description

0000 000

0

General call address. I2C puts the data into the receive buffer and generates a broadcast call interrupt

0000 000

1

Start byte

0000 001

X

CBUS address. I2C interface ignores this access

0000 010

X

Keep

0000 011

X

Keep

0000 1xx

X

Keep

1111 1xx

X

Keep

1111 0xx

X

10-bit slave addressing

**18.3.3**

The host initiates data transmission and sends or receives data from the bus as a master sender or master receiver. Slave responds to master

Machine request to send or receive data as a slave sender or slave receiver.

All data is transmitted in byte format, and there is no limit to the number of bytes transmitted each time. When the host finishes sending the address and R/W bit

Or the master sends a byte of data to the slave, and the slave receiver must generate a response signal (ACK). When receiving

The receiver cannot generate an ACK response signal, and the host will generate a stop condition to abort the transmission. When the slave cannot respond, it must

SDA must be released to high level to make the host generate a stop condition.

When the master transmitter transmits data as shown in the figure below, the slave receiver generates an ACK in response to each byte received

The main transmitter.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

387/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 410** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

478516

S

A

A

P

A/A

R/W

DATA

Slave Address Second Byte

Slave Address First 7 bits

'11110xxx'

'0'(write)

For 10-bit Address

From Master to Slave

From Slave to Master

A = Acknowledge(SDA low)

A = No Acknowledge(SDA high)

S = START Condition

P = STOP Condition

A

A

P

DATA

DATA

S

Slave Address

'0'(write)

For 7-bit Address

R/W

A/A

A

216.

When the host receives data as shown in the figure below, the host must respond to the slave after receiving one byte of data each time, except

The last byte. In this way, the master receiver can inform the slave transmitter whether it is the last byte. Slave transmitter

SDA must be released when NACK is detected so that the host can generate a stop condition.

745602

S

A

A

P

A/A

R/W

Slave Address Second Byte

Slave Address First 7 bits

'11110xxx'

'0'(write)

For 10-bit Address

From Master to Slave

From Slave to Master

A = Acknowledge(SDA low)

A = No Acknowledge(SDA high)

S = START Condition

P = STOP Condition

A

A

P

DATA

DATA

S

Slave Address

'1'(read)

For 7-bit Address

R/W

A/A

A

Sr = RESTART Condition

DATA

Sr

217.

When the host does not want to generate a stop condition and releases the bus, a repeated start condition can be generated. Repeat start conditions and start

The conditions are the same, except that it is generated after the ACK. Working in host mode, I2C interface can use different transmission methods

Communicate with the same slave.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

388/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 411** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

The start byte transfer protocol is used by systems without dedicated I2C hardware modules. When the I2C module is used as the host,

At the beginning of each transmission, a start byte output can be generated for the required slave.

The protocol consists of seven zeros and one one, as shown in the figure below. The processor can use low-speed sampling 0 to check in the address phase

Inquire the bus. Once 0 is detected, the processor can switch from low-speed sampling to the normal rate of the host.

007882

S

SDA

Sr

SCL

1

2

7

8

9

ACK

start byte

00000001

dummy acknowledge

(HIGH)

218.

The starting byte program flow is as follows:

1. The host generates a start condition

2. The host sends the start byte (0000 0001)

3. Host sends ACK clock pulse (ACK)

4. No slave responds to ACK signal

5. The host generates a repeated start condition (RESTART)

The hardware I2C receiver does not need to respond to the start byte, because this is a reserved address, and the address will be in RESTART

After reset.

**18.3.4**

When operating in master mode, the I2C module generates a stop condition on the bus whenever the transmission is empty. If repeated

The start generation function is enabled (RESTART = 1), then the transfer direction changes from read to write or write to read.

condition. If the repeat start condition is not enabled, a start condition will be generated after the stop condition.

The following figure shows the bits of the DR register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

389/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 412** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

924941

CMD

DATA

DR

DATA: Read/Write field; data retrieved from slave is read from this

field; data to be sent to slave is written to this field.

CMD: Write-only field; this bit determines whether transfer to be

carried out is read (CMD = 1) or Write (CMD = 0)

8

7

0

219. DR

The timing diagram below describes the behavior when the Tx FIFO becomes empty when the I2C module is operating in master transmit mode.

656035

Tx FIFO loaded with data

(write data in this example)

Data availability triggers

START condition on bus

Last byte popped from

Tx FIFO

A6 A5 A4 A3 A2 A1 A0

D6 D5 D4 D3 D2 D1 D0

D6 D5 D4 D3 D2 D1 D0

D7

D7

S

W

Ack

Ack

P

Empty Tx FIFO triggers

STOP condition on bus

SDA

SCL

FIFO\_EMPTY

Ack

220.

-Tx FIFO

The following timing diagram describes the behavior of the I2C module when the Tx FIFO becomes empty when it is operating in master receive mode.

791568

Tx FIFO loaded with command

(read operation in this example)

Command availability triggers

START condition on bus

Last Command popped

from Tx FIFO

A6 A5 A4 A3 A2 A1 A0

D6 D5 D4 D3 D2 D1 D0

D6 D5 D4 D3 D2 D1 D0

D7

D7

S

R

Ack

NAck

P

Empty Tx FIFO triggers

STOP condition on bus

SDA

SCL

FIFO\_EMPTY

Ack

221.

-Tx FIFO

**18.3.5**

The I2C bus is a multi-master bus. Arbitration is one in which multiple hosts try to control the bus at the same time, but only allow it

A process that controls the bus and keeps the message from being destroyed. Once one of the hosts has controlled the bus, then until

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

390/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 413** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

When the host sends a stop condition and releases the bus to an idle state, other hosts can control the bus.

When the SCL line is high, arbitration occurs on the SDA line. If two or more hosts try to send information to the bus, at

If all other hosts generate "0", the host that first generates a "1" will lose arbitration. Lost arbitration can

Continue generating clock pulses until the end of byte transfer. If each host tries to address the same device, arbitration will continue

In the data phase.

After the loss of arbitration is detected, the I2C interface will stop generating SCL signals.

The following figure shows the bus timing of the arbitration of two hosts

828193

'0'

MSB

MSB

MSB

DATA

DATA

SDA

SCL

SDA lines up with DATA1 START condition

DATA1 loses arbitration

SDA mirrors DATA2

matching data

'1'

222.

**18.3.6**

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters

The machine generates its own clock to transmit messages. The data is only valid at the high level of the clock. Clock synchronization is via SCL signal

The line'and' connection is made. When the host changes the SCL clock to 0, the host will calculate the time when SCL is low.

The clock period starts to change the SCL clock to 1. However, if another host keeps SCL at 0, then this master

The opportunity enters the wait state until the SCL clock becomes 1.

All hosts will calculate their high time, and the shortest high time host will change SCL to 0. Next Lord

Opportunity to calculate low time, the host with the longest low time will force other hosts to enter the waiting state. This produces

A synchronized SCL clock is shown in the figure below.

975397

Wait State

Start counting HIGH period

SCL transitions HIGH

when all CLKs are in HIGH state

SCL LOW transition

Resets all CLKs to start counting

their LOW periods

CLKA

CLKB

SCL

223.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

391/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 414** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**18.4 I2C**

The I2C interface can operate in one of the following four ways:

• Slave mode

• Slave receiver mode

• Master transmitter mode

• Master receiver mode

Note: The I2C interface module can only work in master mode or slave mode, but not in both modes. So make sure to send

Bit 6 (DISSLAVE) and bit 0 (MASTER) in register CR cannot be set to 0 and 1 (or 1 and 0, respectively).

The I2C functional block diagram is as follows:

769309

APB

Bus Interface Logic

PCLK

Baud Rate

Generator

TXREG

RXREG

TX BUF

RX BUF

SPBRG

T

X Sh

ift R

eg

Register Control

Master

SC

LM

SD

AO

M

SD

AI

Detect

Logic

R

X Sh

ift Re

g

MFSM

control

logic

T

X Sh

ift R

eg

Slave

SC

LS

SD

AO

S

SD

AI

Detect

Logic

R

X Sh

ift Re

g

SFSM

control

logic

224. I2C

**18.4.1**

The following describes the program flow chart of the slave mode

1. Write 0 to bit 0 of the ENR register to disable I2C.

2. Configure the slave address by initializing the SAR register. This address is the address that the I2C interface responds to.

3. Configure the CR register to specify the address format (set bit 3 to select the 7-bit or 10-bit address format). Write 0 to deposit

Bit 6 (DISSLAVE) and write 0 to bit 0 (MASTER) in the CR register of the converter.

4. Write 1 to bit 0 in the ENR register to enable the I2C interface module.

When the I2C interface is addressed by other I2C hosts and requests data, the I2C interface works in slave transmission mode, the steps are as follows

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

392/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 415** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

under:

1. The other I2C master device initiates I2C transmission, and the send address matches the slave address in the SAR register.

2. The I2C interface responds to the sent address and recognizes that the direction of transmission is working in slave transmission mode.

3. The I2C interface generates the RD\_REQ interrupt (RAWISR bit 5 of the register) and pulls the SCL line low. Bus always

Waiting until the software responds.

If the RD\_REQ interrupt is masked (register IMR[5] = 0), it is recommended that the CPU periodically query the RAWISR register.

1. Setting RAWISR bit 5 is equivalent to generating an RD\_REQ interrupt.

2. The software must meet the requirements for I2C transmission.

3. The time interval is usually around 10 SCL clock cycles. For example, for 400kbps, the time interval is 25us.

4. If there is still data in the Tx FIFO before receiving the read request, the I2C interface will generate a TX\_ABRT interrupt

(RAWISR[6]), clear the data in Tx FIFO.

5. Software writes data to the DR register (where bit 8 is set to 0).

6. The software must first clear the RD\_REQ and TX\_ABRT interrupts of the RAWISR register (bit5, 6 respectively)

7. The I2C interface releases the SCL and sends data bytes.

8. The host device sends a repeated start condition to control the bus or sends a stop condition to release the bus.

When other host devices address the I2C interface and send data, the I2C interface works in slave receive mode, the steps are as follows:

1. The other I2C master device initiates I2C transmission, and the send address matches the slave address in the SAR register.

2. The I2C interface responds to the sent address and recognizes that the direction of transmission is working in slave receive mode.

3. The I2C interface receives the data sent by the host and stores the data in the receive buffer.

4. I2C interface generates RX\_FULL interrupt (RAWISR[2]). If RX\_FULL interrupt is masked (IMR[2] =

0), it is recommended that the software periodically query the SR register. When reading bit 3 (RFNE) of SR register is 1, it is equivalent to

RX\_FULL interrupt is generated.

5. The software obtains the received data by reading bits 7:0 in the DR register.

6. The host device sends a repeated start condition to control the bus or sends a stop condition to release the bus.

In the standard I2C protocol, all data processing is the processing of a single byte, the program writes a byte to the slave

The Tx FIFO responds to the host's read request. When a slave (sending) receives a read request from the master (master receiving)

(RD\_REQ), at least one data is put into the Tx FIFO sent from the slave. This I2C interface module can handle x

There is multiple data in the FIFO, so the next read request does not need to generate an interrupt to fetch the data. In the end, this greatly reduced

Less waiting time due to each data interruption.

This mode only exists when the I2C interface is used as a slave transmission mode. If the master sends a response from the transmitted data, the slave's

There is no data in the TX FIFO, the I2C interface will pull down the SCL line of the I2C bus until the read request interrupt (RD\_REQ) is generated

After the TX FIFO data is ready, the SCL line is released.

If the RX\_REQ interrupt is masked (ISR[5] = 0), the software can periodically query and read the RAWISR register. When read

The return of RAWISR[5] to 1 is equivalent to the generation of RX\_REQ interrupt.

The RD\_REQ interrupt is generated due to a read request and must be cleared when exiting the interrupt service routine (ISR) like an interrupt. In

The interrupt service routine (ISR) can write one or more bytes of data to the TX FIFO. Transfer these bytes to the host

During the process, if the master responds to the last byte, the slave will have to generate the RD\_REQ interrupt request again. this is

Because the host requires more data.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

393/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 416** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

If the host receives n bytes from the I2C interface, but the number of data written to the Tx FIFO by the program is greater than n, from

After the machine finishes sending the requested n bytes of data, it will clear the Tx FIFO and ignore the extra bytes.

**18.4.2**

1. Disable I2C interface by setting ENR[0] = 0

2. Configure bit 2:1 of CR register to set the speed mode of I2C operation (standard mode, fast mode). Also make sure

bit 6 (DISSLAVE) is 1, and bit 0 (MASTER) is 1.

3. Write the I2C device address to the TAR register. Setting this register can be configured as a broadcast address or start byte command.

4. Set ENR[0] to enable the I2C interface.

5. Write the transferred data and transfer direction to the DR register. If the DR is configured before enabling the I2C interface

Registers, data and commands will be lost, because the buffer is cleared when the I2C interface is disabled.

The above steps will cause the I2C interface to generate a start condition and send the address byte data to the I2C bus.

The I2C interface supports dynamic switching between reading and writing. When sending data, write data to I2C RX/TX data buffer and command register

In the low byte (DR) of the device, configure the CMD bit to 0 to generate a write operation. The next read command does not need to set DR

The low byte of the register only needs to ensure that the CMD bit is 1. If the transmit FIFO is empty, the I2C module pulls SCL low

Write the next command to the transmit FIFO.

The following flowchart is an example of a program that uses an I2C interface as a host:

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

394/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 417** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

949597

0 ENR

ѝ⾱→I2C⁑ඇ

䝽㖞CR ᇴᆈಘ ˖

1.マDISSLAVE Ѫ 1-Ӿᵪ⾱→

2. マRESTART Ѫ1-֯㜭㟽࿽࿻⁑ᔿ

3. 㖞MASTER10 Ѫ 0-7

4.マSPEED Ѫ 1-ḷ出⁑ᔿ

5.㖞MASTER Ѫ 1-ѫᵪ⁑ᔿ

TAR ㇞ TAR 㖞ⴞḷ

Ӿᵪൠ൰

マ㖞

SSHR

ᇴᆈಘマ㖞SCL

⭥ᒣઘᵏ

マ㖞

SSLR

ᇴᆈಘ 懮㖞SCL

⭥ᒣઘᵏ

Fei

IMR

ᇴᆈಘ֯㜭ᡰᴹѝ

ᯝ

RXTLR ᇴ

RX FIFO

Cheng

マ㖞TXTLR ᇴ

Note: TX FIFO

Cheng

1 ࡠ

ENR

ᇴᆈಘ֯㜭I2C᧕

ਓ

ᖰ

DR

ቭይᔭԔ ᕺᮠ

Awesome

Operating procedure

ੇ

ᱟ

TX\_EMPTYѝ

ᯝᱟ੖ӗ⭏˛

ᴹᴤཊⲴભԔਁ

Admonish

SR [5]

(MST\_ACTIV

)=0?

扉0 ࡠ ENR

ᇴᆈಘᶕ⾱→I2C

⁑ඇ

ᱟ

੖

ᱟ

RX\_FULLѝᯝӗ

⭏˛

X

DR [7:0]

㧧ਆ᧕ਇⲴᮠᦞ

ᱟ

੖

225. I2C

**18.4.3 I2C**

The ABRT control bit in the ENR register allows the software to abandon the I2C bus before completing the transmission command in the TX FIFO.

In response to the ABORT request, the I2C module sends a stop condition to the I2C bus and clears the TX FIFO at the same time. in

The transfer of the operation value is allowed in the main mode.

1. Stop writing new commands to Tx FIFO (DR)

2. If working in DMA mode, set TDMAE = 0 to disable DMA transmission.

3. Set the ABRT bit of the ENR register to 1

4. Wait for TX\_ABRT interrupt

**18.5**

With **DMA**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

395/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 418** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

The I2C interface supports DMA to send and receive data. Can be turned on individually by setting the corresponding bit in the DMA register

DMA transmission or DMA reception. When the data register becomes empty when sending or the data register becomes full when receiving, DMA is generated

request. The DMA request must be responded to before the end of the current byte transfer.

With **DMA**

The DMA transmit mode can be activated by setting the TXEN bit of the DMA register. After allocating the DMA channel for I2C,

When sending data, the DMA controller loads the data from the preset storage area into the DR register.

With **DMA**

The DMA receive mode can be activated by setting the RDMAE bit in the DMA register. Allocate DMA channel for I2C

After that, each time a data byte is received, the DMA controller will transfer the data from the DR register to the preset storage area.

**18.6 I2C**

The following table lists the I2C interrupt bits and how to set and clear them. Some bits are set by hardware and cleared by software;

The other bits are set and cleared by hardware.

58.

Interrupt bit

Hardware set **/** software clear

Hardware set and clear

GEN\_CALL

√

x

START\_DET

√

x

STOP\_DET

√

x

ACTIVITY

√

x

RX\_DONE

√

x

TX\_ABRT

√

x

RD\_REQ

√

x

TX\_EMPTY

x

√

TX\_OVER

√

x

RX\_FULL

x

√

RX\_OVER

√

x

RX\_UNDER

√

x

The following figure describes the operation of interrupt bit set by hardware and cleared by software in the interrupt register

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

396/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 419** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

809039

IMR

RAWISR

pwdata[i]

i = register bit field

register\_en

(decoded from padder)

ISR

0

1

0

1

0

1

1

0

0

I2C\_EN

H/W SET

CLR\_READ\_EN

{

S/W Access

to Register

226.

**18.7 I2C**

59. I2C

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

I2C\_CR

I2C control register

0x0000007F

[Section 18.7.1](https://translate.googleusercontent.com/translate_f#420)

0x04

I2C\_TAR

I2C target address register

0x00000055

[Section 18.7.2](https://translate.googleusercontent.com/translate_f#422)

0x08

I2C\_SAR

I2C slave address register

0x00000055

[Section 18.7.3](https://translate.googleusercontent.com/translate_f#423)

0x10

I2C\_DR

I2C data command register

0x00000001

[Section 18.7.4](https://translate.googleusercontent.com/translate_f#423)

0x14

I2C\_SSHR

Standard mode I2C clock high count register

Device

0x00000190

[Section 18.7.5](https://translate.googleusercontent.com/translate_f#424)

0x18

I2C\_SSLR

Standard mode I2C clock low count register

Device

0x000001D6

[Section 18.7.6](https://translate.googleusercontent.com/translate_f#424)

0x1C

I2C\_FSHR

Fast mode I2C clock high count register

Device

0x00000036

[Section 18.7.7](https://translate.googleusercontent.com/translate_f#424)

0x20

I2C\_FSLR

Fast mode I2C clock low count register

Device

0x00000082

[Section 18.7.8](https://translate.googleusercontent.com/translate_f#425)

0x2C

I2C\_ISR

I2C interrupt status register

0x00000000

[Section 18.7.9](https://translate.googleusercontent.com/translate_f#425)

0x30

I2C\_IMR

I2C interrupt mask register

0x000008FF

[Section 18.7.10](https://translate.googleusercontent.com/translate_f#425)

0x34

I2C\_RAWISR

I2C RAW interrupt register

0x00000000

[Section 18.7.11](https://translate.googleusercontent.com/translate_f#426)

0x38

I2C\_RXTLR

I2C receive threshold

0x00000000

[Section 18.7.12](https://translate.googleusercontent.com/translate_f#427)

0x3C

I2C\_TXTLR

I2C transmission threshold

0x00000000

[Section 18.7.13](https://translate.googleusercontent.com/translate_f#428)

0x40

I2C\_ICR

I2C combination and independent interrupt clear register

0x00000000

[Section 18.7.14](https://translate.googleusercontent.com/translate_f#428)

0x44

I2C\_RX\_UNDER

I2C clear RX\_UNDER interrupt register

0x00000000

[Section 18.7.15](https://translate.googleusercontent.com/translate_f#428)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

397/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 420** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x48

I2C\_RX\_OVER

I2C clear RX\_OVER interrupt register

0x00000000

[Section 18.7.16](https://translate.googleusercontent.com/translate_f#429)

0x4C

I2C\_TX\_OVER

I2C clear TX\_OVER interrupt register

0x00000000

[Section 18.7.17](https://translate.googleusercontent.com/translate_f#429)

0x50

I2C\_RD\_REQ

I2C clears the RD\_REQ interrupt register

0x00000000

[Section 18.7.18](https://translate.googleusercontent.com/translate_f#429)

0x54

I2C\_TX\_ABRT

I2C clear TX\_ABRT interrupt register

0x00000000

[Section 18.7.19](https://translate.googleusercontent.com/translate_f#430)

0x58

I2C\_RX\_DONE

I2C clear RX\_DONE interrupt register

0x00000000

[Section 18.7.20](https://translate.googleusercontent.com/translate_f#430)

0x5C

I2C\_ACTIV

I2C clears the ACTIVITY interrupt register

0x00000000

[Section 18.7.21](https://translate.googleusercontent.com/translate_f#430)

0x60

I2C\_STOP

I2C clears the STOP\_DET interrupt register

0x00000000

[Section 18.7.22](https://translate.googleusercontent.com/translate_f#431)

0x64

I2C\_START

I2C clear START\_DET interrupt register

0x00000000

[Section 18.7.23](https://translate.googleusercontent.com/translate_f#431)

0x68

I2C\_GC

I2C clears the GEN\_CALL interrupt register

0x00000000

[Section 18.8.74](https://translate.googleusercontent.com/translate_f#431)

0x6C

I2C\_ENR

I2C enable register

0x00000000

[Section 18.7.25](https://translate.googleusercontent.com/translate_f#432)

0x70

I2C\_SR

I2C status register

0x00000006

[Section 18.7.26](https://translate.googleusercontent.com/translate_f#432)

0x74

I2C\_TXFLR

I2C transmit buffer level register

0x00000000

[Section 18.7.27](https://translate.googleusercontent.com/translate_f#433)

0x78

I2C\_RXFLR

I2C receive buffer level register

0x00000000

[Section 18.7.28](https://translate.googleusercontent.com/translate_f#433)

0x7C

I2C\_HOLD

I2C SDA hold time register

0x00000001

[Section 18.7.29](https://translate.googleusercontent.com/translate_f#434)

0x88

I2C\_DMA

I2C DMA control register

0x00000000

[Section 18.7.30](https://translate.googleusercontent.com/translate_f#434)

0x94

I2C\_SETUP

I2C SDA setup time register

0x00000064

[Section 18.7.31](https://translate.googleusercontent.com/translate_f#435)

0x98

I2C\_GCR

I2C general call ACK register

0x00000001

[Section 18.7.32](https://translate.googleusercontent.com/translate_f#435)

**18.7.1 I2C**

**I2C\_CR**

Offset address: 0x00

Reset value: 0x0000 007F

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

RESTART STOP

rw

rw

rw

r

rw

rw

rw

rw

rw

Reserved

SPEED

EMPINT STOPINT DISSLAVEREPEN MASTER10 SLAVE10

MASTER

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 11

Reserved

Always read as 0.

10

RESTART

rw

0x00

Whether to generate a RESTART signal before sending or receiving

Only in the configuration of IC\_EMPTYFIFO\_HOLD\_MASTER\_EN is "1"

Valid.

1: If the RESTART signal is "1", data is received or sent (according to CMD

Value) before a RESTART signal, regardless of whether the previous command was

Whether to change the data transmission direction. If the RESTART signal is "0", STOP

The signal will immediately follow the START signal

0: If the RESTART signal is "1", only the transmitter is changed in the previous command

The RESTART signal is generated only when the time is up. If the RESTART signal is '0',

The STOP signal will immediately follow the START signal

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

398/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 421** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Bit**

**Field**

**Type**

**Reset**

**Description**

9

STOP

rw

0x00

STOP: After sending or receiving, whether to generate a STOP signal

Only in the configuration of IC\_EMPTYFIFO\_HOLD\_MASTER\_EN is "1"

Valid.

1: A STOP signal is generated after the current byte, regardless of whether the Tx FIFO is

Whether it is empty. If the Tx FIFO is not empty, the host immediately sends a new transmission

Loss and bus arbitration signal

0: No STOP signal is generated after the current byte, regardless of whether the Tx FIFO is

Whether it is empty. The host continues the current transmission (sending or receiving data according to CMD's

value). If the Tx FIFO is empty, the host will pull SCL low to suspend the bus until

Tx FIFO receives new data

8

EMPINT

rw

0x00

This bit controls TX\_EMPTY interrupt generation, please refer to RAWISR register for details

Device.

7

STOPINT

rw

0x00

In slave mode, is a STOP interrupt generated?

1: STOP interrupt is generated when the address matches

0: STOP interrupt is generated regardless of whether the addresses match

This bit only applies to slave mode

Note: When addressing the broadcast address, if this bit is set, the slave does not generate a STOP interrupt.

The STOP interrupt is only generated when the sent address matches the slave address.

6

DISSLAVE

rw

0x01

This bit controls whether the I2C interface slave is disabled (This bit controls whether I2C

has its slave disabled)

0: slave enable

1: Slave prohibited

5

REPEN

rw

0x01

When acting as a host, this bit controls whether to send

RESTART

condition

(Determines whether RESTART conditions may be

sent when acting as a master)

0: prohibited

1: enable

When RESTART is disabled and the I2C interface is used as a host, the following functions cannot be performed

can:

Send start byte

Change the transmission direction in combined format mode

10-bit address format read operation

Replace RESTART condition to send stop condition first and then send start bar

Pieces. If the above operation is performed, bit 6 of the RAWISR register will be set

(TX\_ABRT)

4

MASTER10

r

0x01

Address format when acting as I2C host (Address mode when acting

as a master)

0: 7-bit address format

1: 10-bit address format

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

399/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 422** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

SLAVE10

rw

0x01

When acting as a slave, this bit controls the response to a 10-bit or 7-bit address (When

acting as a slave, this bit controls whether the I2C re-

sponds to 7- or 10-bit addresses)

0: 7-bit addressing address. The I2C interface ignores handling 10-bit addressing. for

7-bit addressing, only the lower 7 bits of the SAR register are compared

1: 10-bit addressing address. I2C only responds to 10-bit addressing, receiving address

10-bit comparison with SAR

2:1

SPEED

rw

0x03

These two bits control the rate mode of the I2C interface (These bits control

at which speed the I2C operates)

This setting is valid only when the I2C interface works in the host mode.

1: Standard mode (0 ∼ 100Kbps)

2: Quick mode (400Kbps)

0

MASTER

rw

0x01

This bit controls whether the host mode (This bit controls whether the I2C mas-

ter is enabled)

0: host prohibited

1: Host enable

The DISSLAVE(bit 6) and MASTER(bit 0) configurations are listed in the following table:

60. DISSLAVE(bit 6)

MASTER(bit 0)

**DISSLAVE CR[6]**

**MASTER CR[0]**

status

0

0

Slave device

0

1

Configuration error

1

0

Configuration error

1

1

Host device

**18.7.2 I2C**

**I2C\_TAR**

Offset address: 0x04

Reset value: 0x0000 0055

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

SPECIAL GC

ADDR

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 12

Reserved

Always read as 0.

11

SPECIAL

rw

0x00

This bit indicates whether the software is executing a special command (broadcast call or start

Byte command)

0: Ignore the 10th GC and use the ADDR bit normally

1: Execute special I2C command such as GC bit description

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

400/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 423** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Bit**

**Field**

**Type**

**Reset**

**Description**

10

GC

rw

0x00

If bit 11 is set, this bit indicates whether the I2C is performing a general call or initiation

Byte (If bit 11(SPECIAL) is set to 1, then this bit indicates

whether a General Call or START byte command is to be

performed by the I2C)

0: general call address. Only the write operation can be performed when sending the general call address.

The I2C interface has been working in the broadcast address mode until SPECIAL (bit 11)

Is cleared

1: Start byte command

9: 0

ADDR

rw

0x55

The target address of the main operation (This is the target address for any mas-

ter transaction)

When sending a broadcast address, these bits can be ignored.

To generate the start byte command, the CPU only needs to write these bits once.

**18.7.3 I2C**

**I2C\_SAR**

Offset address: 0x08

Reset value: 0x0000 0055

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

ADDR

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 10

Reserved

Always read as 0.

9: 0

ADDR

rw

0x55

When the I2C interface works in slave mode, these stored slave addresses are

Bit address format, ADDR is only valid for [6:0].

**18.7.4 I2C**

**I2C\_DR**

Offset address: 0x10

Reset value: 0x0000 0001

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

Reserved

CMD

DAT

rw

rw

rw

w

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 9

Reserved

Always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

401/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 424** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Bit**

**Field**

**Type**

**Reset**

**Description**

8

CMD

w

0x00

Control the read or write operation in master mode

1: read

0: write

When a command enters the TX FIFO, this bit is used to distinguish between read and write commands. From

In receive mode, this bit write operation is ignored. In slave transmission mode, write 0

Indicates that the data in the DR register is ready to be sent.

7:0

DAT

rw

0x01

Data to be sent or received on the I2C bus

**18.7.5**

**I2C**

**I2C\_SSHR**

Offset address: 0x14

Reset value: 0x0000 0190

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 0

CNT

rw

0x0190

SCL clock high period in I2C interface standard mode

Note: The configurable value of this register is between 6 and 65525, this is due to the I2C connection

The port uses a 16-bit counter whose value is equal to the SSHR + 10 time scale

The I2C bus is idle.

**18.7.6**

**I2C**

**I2C\_SSLR**

Offset address: 0x18

Reset value: 0x0000 01D6

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 0

CNT

rw

0x01D6

SCL clock low period in I2C interface standard mode

The minimum value is 8.

**18.7.7**

**I2C**

**I2C\_FSHR**

Offset address: 0x1C

Reset value: 0x0000 0036

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

402/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 425** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 0

CNT

rw

0x0036

SCL clock high period in I2C interface fast mode

When I2C works in standard mode, this register is read-only and the return value is 0.

The minimum value is 6.

**18.7.8**

**I2C**

**I2C\_FSLR**

Offset address: 0x20

Reset value: 0x0000 0082

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 0

CNT

rw

0x0082

SCL clock low period in I2C interface fast mode

When I2C works in standard mode, this register is read-only and the return value is 0.

The minimum value is 8.

**18.7.9 I2C**

**I2C\_ISR**

Offset address: 0x2C

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

Reserved

r

r

r

r

r

r

r

r

r

r

r

r

r

r

D

RESTART GC START

ACTIV RX\_DONE TX\_ABRTRD\_REQ TX\_EMPTY TX\_OVER RX\_FULL RX\_OVER RX\_UNDER

STOP

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 14

Reserved

Always read as 0.

13: 0

r

0x0000

For specific description, please refer to RAWISR register

**18.7.10 I2C**

**I2C\_IMR**

Offset address: 0x30

Reset value: 0x0000 08FF

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

403/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 426** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

Reserved

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

GC START

ACTIV RX\_DONE TX\_ABRTRD\_REQ TX\_EMPTY TX\_OVER RX\_FULL RX\_OVER RX\_UNDER

STOP

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 12

Reserved

Always read as 0.

11: 0

rw

0x08FF

Each bit mask corresponds to the ISR bit.

**18.7.11 I2C RAW**

**I2C\_RAWISR**

Offset address: 0x34

Reset value: 0x0000 0000

The difference between RAWISR and ISR registers is that the former will not be masked.

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

r

r

r

r

r

Reserved

GC START

ACTIV RX\_DONE TX\_ABRTRD\_REQ TX\_EMPTY TX\_OVER RX\_FULL RX\_OVER RX\_UNDER

STOP

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 12

Reserved

Always read as 0.

11

GC

r

0x00

General call

Set when a general call address is received.

Disable the I2C interface or clear it when the CPU reads the GC register. I2C will receive

The received data is stored in the receive buffer.

10

START

r

0x00

Start condition detection

Whether the I2C interface works in the master or slave, once the I2C interface is detected

This bit is set when the start or repeat start condition is reached

9

STOP

r

0x00

Stop condition detection (Stop condition detection)

The state of this bit depends on the state of STOPINT in the CR register

When STOPINT = 0

Whether the I2C interface works in the master or slave, once the I2C interface is detected

This bit is set when the stop condition is applied. In slave mode, regardless of whether the addressing is matched

Configuration will generate a STOP interrupt

When STOPINT = 1

In master mode (MASTER = 1), this bit shows whether the I2C interface sends

Birth stop condition

In slave mode (MASTER = 0), only when the slave address matches successfully

A STOP interrupt is generated.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

404/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 427** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Bit**

**Field**

**Type**

**Reset**

**Description**

8

ACTIV

r

0x00

I2C interface is activated, this bit is used to capture the active state of the I2C module

After being set, it can only be cleared by the following four methods:

Disable I2C interface

Read the ACTIV register

Read ICR register

System reset

Once set, it can only be cleared by the above method, even if I2C is idle

Status, this bit also remains high until it is cleared.

7

RX\_DONE

r

0x00

From the end of transmission (Transmit done)

When I2C is sent as a slave, if the host does not send a byte of data

If there is a response, this bit will be set.

This situation occurs in the last byte of the transmission, indicating the end of the transmission.

6

TX\_ABRT

r

0x00

Transmit abort

When the I2C interface is used as a transmitter, it cannot be set when the data in the buffer cannot be sent

Bit.

Note: Abort transmission will clear the receive and transmit buffers in the I2C interface. Delivery delay

The pulse will be in the refresh state until the TX\_ABRT register is read. Once the read operation is performed

After sending, you can receive new data on the APB bus.

5

RD\_REQ

r

0x00

Read request (Read request)

Set when I2C is a slave and other masters try to read data from the I2C interface

Bit.

The I2C interface will keep the bus waiting (SCL = 0) until the interrupt is serviced

Management. This means that the I2C interface is successfully addressed by other masters as a slave

And request to send data. The processor must respond to the interrupt and then write data to

In the DR register. This bit is cleared when the processor reads the RD\_REQ register.

4

TX\_EMPTY

r

0x00

Transmit buffer empty

The state of this bit depends on the EMPINT state in the CR register:

Set when EMPINT = 0 and the transmit buffer is empty

Set when EMPINT = 1, the transmit buffer is empty and the internal shift register ends

Bit

When the transmit buffer is not empty, it is automatically cleared by hardware.

3

TX\_OVER

r

0x00

Transmit buffer over

Set by the processor to write new data when the transmit buffer is full, resulting in an overflow.

2

RX\_FULL

r

0x00

Receive buffer not empty

Set when the receive buffer is not empty.

It is cleared by hardware when the receive buffer is empty.

1

RX\_OVER

r

0x00

Receive buffer over

Set when the receive buffer is full and new data is received. At this time, the I2C interface will sound

Yes, but new data will be lost.

0

RX\_UNDER

r

0x00

Receive buffer under (Receive buffer under)

Set when the RX FIFO is empty when the processor reads the DR register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

405/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 428** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**18.7.12 I2C**

**I2C\_RXTLR**

Offset address: 0x38

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

Reserved

TL

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 8

Reserved

Always read as 0.

7: 0

TL

r

0x00

Receive FIFO threshold level

Control RX\_FULL interrupt trigger.

**18.7.13 I2C**

**I2C\_TXTLR**

Offset address: 0x3C

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

Reserved

TL

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 8

Reserved

Always read as 0.

7: 0

TL

r

0x00

Receive FIFO threshold level

Control TX\_EMPTY interrupt trigger.

**18.7.14 I2C**

**I2C\_ICR**

Offset address: 0x40

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

ICR

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

ICR

r

0x00

Reading this register will clear all combined interrupts and independent interrupts

This bit does not clear the interrupt that the hardware can automatically clear, only the software can clear it.

Break.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

406/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 429** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**18.7.15 I2C**

**RX\_UNDER**

**I2C\_RX\_UNDER**

Offset address: 0x44

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

RX\_UNDER

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

RX\_UNDER

r

0x00

Reading this register clears the RX\_UNDER interrupt (RAWISR[0])

**18.7.16 I2C**

**RX\_OVER**

**I2C\_RX\_OVER**

Offset address: 0x48

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

RX\_OVER

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

RX\_OVER

r

0x00

Reading this register clears the RX\_OVER interrupt (RAWISR[1])

**18.7.17 I2C**

**TX\_OVER**

**I2C\_TX\_OVER**

Offset address: 0x4C

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

TX\_OVER

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

TX\_OVER

r

0x00

Reading this register clears the TX\_OVER interrupt (RAW\_ISR[3])

**18.7.18 I2C**

**RD\_REQ**

**I2C\_RD\_REQ**

Offset address: 0x50

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

407/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 430** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

RD\_REQ

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

RD\_REQ

r

0x00

Reading this register clears the RD\_REQ interrupt (RAW\_ISR[5])

**18.7.19 I2C**

**TX\_ABRT**

**I2C\_TX\_ABRT**

Offset address: 0x54

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

TX\_ABRT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

TX\_ABRT

r

0x00

Reading this register clears the TX\_ABRT interrupt (RAWISR6))

At the same time, the TX FIFO is also released from the refresh/reset state to receive the write

The data.

**18.7.20 I2C**

**RX\_DONE**

**I2C\_RX\_DONE**

Offset address: 0x58

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

RX\_DONE

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

RX\_DONE

r

0x00

Reading this register clears the RX\_DONE interrupt (RAWISR[7])

**18.7.21 I2C**

**ACTIVITY**

**I2C\_ACTIV**

Offset address: 0x5C

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

408/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 431** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

Reserved

r

ACTIV

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

ACTIV

r

0x00

If the I2C bus is inactive, read this register to clear the ACTIV interrupt

(RAWISR[8])

If I2C is still active, the ACTIV interrupt will continue to be set. When I2C

The module is disabled or this bit is cleared by hardware when the I2C bus is no longer active. can

By reading this register to get the status of ACTIV (bit 8) in RAWISR

state.

**18.7.22 I2C**

**STOP\_DET**

**I2C\_STOP**

Offset address: 0x60

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

STOP

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

STOP

r

0x00

Reading this register clears the STOP interrupt (RAWISR[9])

**18.7.23 I2C**

**START\_DET**

**I2C\_START**

Offset address: 0x64

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

START

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

START

r

0x00

Reading this register clears the START interrupt (RAWISR[10])

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

409/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 432** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**18.7.24 I2C**

**GEN\_CALL**

**I2C\_GC**

Offset address: 0x68

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

Reserved

GC

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

GC

r

0x00

Reading this register clears the GC interrupt (RAWISR[11])

**18.7.25 I2C**

**I2C\_ENR**

Offset address: 0x6C

Reset value: 0x0000 0000

ABORT ENABLE

rw

rw

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 2

Reserved

Always read as 0.

1

ABORT

rw

0x00

I2C transfer abort

0: Abort did not occur or has ended

1: The abort operation is in progress

When the I2C module is set as a host, software can interrupt I2C transmission. One

Once set, it cannot be cleared immediately. After setting, the I2C module control logic will be completed

A STOP condition is generated after the current transfer and the send buffer is cleared, aborted

TX\_ABRT interrupt is generated after operation.

The ABORT bit will be automatically cleared after the abort operation.

0

ENABLE

rw

0x00

I2C module enable (I2C mode enable)

0: Disable I2C module (transmit and receive buffers remain erased)

1: Enable I2C module

**18.7.26 I2C**

**I2C\_SR**

Offset address: 0x70

Reset value: 0x0000 0006

This register is read-only and indicates the current transfer and buffer status. The status bit does not generate an interrupt.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

410/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 433** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

Reserved

SLV\_ACTIVMST\_ACTIV RFF RFNE TFE TFNE ACTIV

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 7

Reserved

Always read as 0.

6

SLV

\_ACTIV

r

0x00

Slave FSM activity status

0: The slave state machine is in the IDLE state, so the I2C slave part is not active

1: The slave state machine is not in the IDLE state, so the I2C slave part is active

5

MST

\_ACTIV

r

0x00

Host state machine activity status (Master FSM activity status)

0: The host state machine is in the IDLE state, so the I2C host part is not active

1: The host state machine is not in the IDLE state, so the I2C host is partially active

4

RFF

r

0x00

Receive FIFO completely full

0: receive buffer is not full

1: Receive buffer is full

3

RFNE

r

0x00

Receive FIFO not empty

0: receive buffer empty

1: Receive buffer is not empty

2

TFE

r

0x01

Transmit FIFO completely empty

0: send buffer is not empty

1: send buffer empty

1

TFNF

r

0x01

Transmit FIFO not full

0: send buffer is full

1: Send buffer is not full

0

ACTIV

r

0x00

I2C activity status (I2C activity status)

The result of the OR of the MST\_ACTIV bit and the SLV\_ACTIV bit.

**18.7.27 I2C**

**I2C\_TXFLR**

Offset address: 0x74

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

Reserved

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 2

Reserved

Always read as 0.

1: 0

CNT

r

0x00

Number of valid data in transmission buffer (0 ∼ 2)

**18.7.28 I2C**

**I2C\_RXFLR**

Offset address: 0x78

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

411/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 434** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

Reserved

CNT

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 2

Reserved

Always read as 0.

1: 0

CNT

r

0x00

Number of valid data in receive buffer (0 ∼ 2)

**18.7.29 I2C SDA**

**I2C\_HOLD**

Offset address: 0x7C

Reset value: 0x0000 0001

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

TX\_HOLD

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

r

r

Reserved

RX\_HOLD

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 24

Reserved

Always read as 0.

23: 16

RX\_HOLD

r

0x00

When the I2C device is used as the receiver, the SDA hold time, the unit is APB1 system

Clock cycle

15: 0

TX\_HOLD

r

0x01

When the I2C device is used as the transmission, the SDA hold time, the unit is APB1 series

Clock cycle

**18.7.30 I2C DMA**

**I2C\_DMA**

Offset address: 0x88

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

Reserved

TXEN RXEN

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 2

Reserved

Always read as 0.

1

TXEN

rw

0x00

Transmit DMA enable

0: Send DMA prohibited

1: Send DMA enable

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

412/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 435** |

UM\_MM32F003\_q\_Ver1.19

I2C

(I2C)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

RXEN

rw

0x00

Receive DMA enable

0: receive DMA prohibited

1: Receive DMA enable

**18.7.31 I2C SDA**

**I2C\_SETUP**

Offset address: 0x94

Reset value: 0x0000 0064

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

CNT

Reserved

r

r

r

r

r

r

r

r

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 8

Reserved

Always read as 0.

7: 0

CNT

rw

0x64

SDA setup time (SDA setup)

If required, the recommended delay time is 1000nS, and the APB1 clock frequency is

At 10MHZ, it is recommended to set this register to 11. The minimum value of this register is 2.

**18.7.32 I2C**

**ACK**

**I2C\_GCR**

Offset address: 0x98

Reset value: 0x0000 0001

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

Reserved

GC

**Bit**

**Field**

**Type**

**Reset**

**Description**

15: 1

Reserved

Always read as 0.

0

GC

rw

0x01

ACK (ACK general call)

1: respond to ACK after receiving a broadcast call

0: No response or interruption after receiving the broadcast call

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

413/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 436** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

19 Use **(UART)**

Universal Asynchronous Transceiver (UART)

**19.1 UART**

Universal asynchronous transceiver (UART) provides a flexible method and uses industry standard NRZ asynchronous serial data format

Full-duplex data exchange between external devices. UART uses fractional baud rate generator to provide a wide range of baud rates

select. It supports synchronous one-way communication and half-duplex single-line communication, as well as modem (CTS/RTS) operation.

Using the DMA method of multi-buffer configuration, high-speed data communication can be achieved.

**19.2 UART**

• Support RS-232S protocol in asynchronous mode, in line with industry standard 16550

• Support DMA request

• Full-duplex asynchronous operation

• Fractional baud rate generator system

• Programmable baud rate for sending and receiving

• Separate transmit and receive buffer registers

• Built-in one byte send and receive buffer

• Send and receive data low order first

• A start bit starts, followed by a data bit, the output data length can be 5, 6, 7, 8 bits, and finally

Stop bit. In addition, you can choose whether to add parity bit, parity bit after the data bit and before the stop bit.

• The 9th bit can be configured for synchronization frame

• Support hardware odd or even parity generation and detection

• Line disconnection generation and detection

• Support hardware automatic flow control

• Support the following interrupt sources:

**– The** sender BUFFER is empty

**– The** data at the receiving end is valid

**–** Receive buffer buffer overflow

**–** Frame error

**–** Parity error

**–** Receive disconnected frames

**–** Send shift register complete

**-** sending at The Disconnection Frame IS Completed

**–** Receive sync frame

**19.3 UART**

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

414/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 437** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

Any UART bidirectional communication requires at least two pins: receive data input (RX) and transmit data output (TX).

RX: Receive data serial input. Recover data by oversampling technology to distinguish data and noise.

TX: Transmit data output. When the transmitter is disabled, the output pin returns to its I/O port configuration. When the transmitter is activated

When alive and not sending data, the TX pin is high.

• The bus should be idle before sending or receiving

• a start bit

• One data word (5, 6, 7 or 8 bits) with the least significant bit first

• 0.5, 1, 1.5, 2 stop bits, which indicates the end of the data frame

• Use fractional baud rate generator-16-bit integer and 4 decimal places.

The following pins are required in hardware flow control mode:

• nCTS: Clear to send. If it is high, block the next data transmission at the end of the current data transmission.

• nRTS: Send request. If it is low, it means that the UART is ready to receive data.

710870

APB

Bus Interface Logic

pclk

Baud Rate

Generator

TXREG

RXREG

TXBUFFER

(1 Byte)

RXBUFFER

(1 Byte)

spbrg

TX Shift Register

S

P STOP

Register Control

TX

Transmitter

RX Shift Register

Receiver

Data

Recovery

RX

bclk16

bclk16

rxbuffer\_full

rxoerr

rxferr

rxperr

rxbrk

char[1

:0

]

rxen

txen

spb

pen

psel

spb

pen

psel

char[1

:0

]

txbuffer\_empty

pclk

pclk

227. UART

**19.3.1 UART**

The word length can be selected from 5 to 8 bits by programming the CHAR bits in the UART\_CCR register. During the start bit, TX

The pin is at a low level and is at a high level during the stop bit.

The idle symbol is regarded as a complete data frame consisting entirely of '1', followed by the beginning of the next frame containing the data

Bit (the number of '1' also includes the number of stop bits).

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

415/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 438** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

The disconnect symbol is regarded as receiving all '0's in one frame period (including the stop bit period, also '0'). At the end of the disconnected frame,

The transmitter inserts 1 or 2 stop bits ('1') to acknowledge the start bit.

Transmission and reception are driven by a common baud rate generator. When the enable bits of the transmitter and receiver are set respectively, the

Don't generate a clock for it.

849357

Worry

ս

ڌ→

ս

лњ

Worry

ս

ਟ㜭Ⲵ

ཷڦỰ

傼ս

ᮠᦞᑗ

**.. 8** ս ᆇ䮯 **.. 1** њ ڌ → ս

лањᮠᦞᑗ

ᰦ䫏

ս **0**

ս **1**

ս **2**

ս **3**

ս **4**

ս **5**

ս **6**

ս **7**

stop

Bit

Idle frame

Break frame

Start

Bit

Start

Bit

228. UART

**19.3.2**

The transmitter sends data words of 5 to 8 bits according to the state of CHAR bits. When the transmit enable bit (TXEN) is set, send

The data sent to the shift register is output on the TX pin.

During UART transmission, the least significant bit of the data is shifted out on the TX pin first. In this mode, UART\_TDR

The register contains a buffer between the internal bus and the transmit shift register.

There is a low-level start bit before each character; the number of stop bits that follow is configurable.

The TXEN bit cannot be reset during data transmission, otherwise the data on the TX pin will be destroyed because the baud rate counter stops counting. positive

The current data during transmission will be lost.

The number of stop bits sent with each character can be programmed through the SPB bit.

A break frame is a 10-bit low level followed by a stop bit; or an 11-bit low level followed by a stop bit. Received disconnect frame will be set

The RXBRK\_INTF bit of the bit interrupt status register.

1. Activate the UART by setting the UARTEN bit in the UART\_GCR register.

2. Program the CHAR bit of UART\_CCR to define the word length.

3. The number of SPB programming stop bits in UART\_CCR.

4. Set the TXEN bit in UART\_GCR.

5. Use the UART\_BRR register to select the required baud rate.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

416/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 439** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

6. Write the data to be sent into the UART\_TDR register (this action clears the TX\_INTF bit). In only one buffer

In the case of a data converter, repeat step 6 for each data to be sent.

Clearing the TX\_INTF bit is always done by writing to the data register. The TX\_INTF bit is set by hardware, it

show:

• Data has been transferred from the TDR to the shift register, and data transmission has started

• The TDR register is cleared

• The next data can be written into the UART\_TDR register without overwriting the previous data.

If the TXIEN bit is set, this flag will generate an interrupt. If the UART is sending data at this time, check the UART\_TDR

The write operation of the register stores the data into the TDR register, and copies the data into the shift register at the end of the current transfer

Device.

If the UART is not sending data at this time and is in an idle state, the write operation to the UART\_TDR register is directly

Put the data into the shift register, the data transfer starts, and the TX\_INTF bit is set immediately. At the same time UART\_CSR

TXBUF\_EMPTY will also be set. When a frame is sent (after the stop bit is sent), there is no UART\_TDR

Write new data (TDR register is empty), TXC will be set, indicating that all transfers have been completed.

838422

Write

UART\_TDR

Baud rate

clock

TX

TX\_INTF bit

TXDONE bit

Note: This timing diagram shows two consecutive transmissions.

Word 1

Transmit Shift Reg.

Start Bit

Bit 0

WORD 1

Bit 1

Bit 7/8

Bit 0

Word 2

WORD 1

Start Bit

WORD 2

Transmit Shift Reg.

WORD 2

Stop Bit

229.

Set BRK to send a disconnect symbol. If you set BRK=1, after completing the current data transmission, it will be on the TX line

Send a disconnect symbol on. The software must set BRK when the sending of the break character is completed (when the stop bit of the break symbol)

= 0. The UART inserts a logic '1' at the end of the last disconnected frame to ensure that it can identify the start bit of the next frame.

**19.3.3**

During UART reception, the least significant bit of the data is first shifted in from the RX pin. In this mode, UART\_RDR sends

The buffer contained in the memory is located between the internal bus and the receive shift register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

417/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 440** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

Configuration steps:

1. Set UARTEN in the UART\_GCR register to '1' to activate the UART.

2. Program the CHAR bit of UART\_CCR to define the word length.

3. The number of SPB programming stop bits in UART\_CCR.

4. Use the UART\_BRR register to select the required baud rate.

5. Set the RXEN bit of UART\_GCR. Activate the receiver so that it starts looking for the start bit.

When a character is received,

• The RX\_INTF bit is set. It indicates that the contents of the shift register are transferred to RDR. In other words, the data has been

Received and can be read (including error flags related to it).

• If the RXIEN bit is set, an interrupt is generated.

• If a frame error or overflow error is detected during reception, the error flag will be set.

• Software reads the UART\_RDR register. The RX\_INTF bit must be cleared before the reception of the next character ends.

When receiving data, the RXEN bit should not be reset. If the RXEN bit is cleared during reception, the reception of the current byte is lost.

When a disconnect frame is received, the UART will set the RXBRK\_INTF interrupt.

If another character is received before UART\_RDR is not read, an overflow error occurs.

When an overflow error occurs:

• The RXOERR\_INTF bit is set.

• RDR content will not be lost. Reading the UART\_RDR register can still get the previous data.

• The previous contents of the shift register will be overwritten. All data received subsequently will be lost.

• If the RXOERREN bit is set, an interrupt is generated.

A frame error was detected when the stop bit was not received and recognized at the expected time. When a frame error is detected:

• The RXFERR\_INTF bit is set by hardware.

• Invalid data will not be transferred from the shift register to the UART\_RDR register.

• If the RXFERREN bit is set, an interrupt is generated.

**19.3.4 9**

If the B8EN control bit of the UART\_CCR register is enabled, the UART enables the transmission and reception of 9-bit data.

Send and receive 9-bit data. Note: After B8EN is enabled, the parity enable bit PEN has no effect.

When sending data, you need to set B8TXD before writing data to the transmit register UART\_TDR. B8TXD

The values ​​​​of of MSB and UART\_TDR as the transmission data are transmitted at the same time. If B8TOG is set, if B8TXD

When it is the same as B8POL, it means that the data is used as an address frame or a synchronization frame, and B8TXD will automatically flip after transmission.

In the following data transmission process, there is no need to set B8TXD to invalid level.

When receiving data, the highest bit of the received data can be read from register bit B8RXD. If the received B8RXD is

When B8POL is the same, the RXB8\_INTF bit of the interrupt status register UART\_ISR will be set.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

418/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 441** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**19.3.5**

Multiprocessor communication can be achieved through UART (connect several UARTs in a network). For example, a UART device

The backup can be the master, its TX output is connected to the RX input of other UART slave devices;

The TX output is logically connected together and connected to the RX input of the master device.

In a multiprocessor configuration, we usually want only the addressed receiver to be activated to receive subsequent data.

In this way, the extra UART service overhead caused by the participation of unaddressed receivers can be reduced.

Devices that are not addressed can enable their silent function to be placed in silent mode. In silent mode:

• No reception status bit will be set.

• All receive interrupts are disabled.

• The RWU bit in the UART\_CCR register is set. RWU can be controlled automatically by hardware or under certain conditions

Software write.

According to the state of the WAKE bit in the UART\_CCR register, the UART can enter or exit the silent mode in two ways

formula.

• If the WAKE bit is reset: a free bus test is performed.

• If the WAKE bit is set: address mark detection is performed.

**(WAKE=0)**

When the RWU bit is written to 1, the UART enters silent mode. When an idle frame is detected, it is woken up. Then RWU

Cleared by hardware, the interrupt status flag RX\_INTF will not be set. RWU can also be written to 0 by software.

**(address mark)**

**(WAKE=1)**

In this mode, if the MSB is B8POL, the byte is considered as an address, otherwise it is considered as data. in a

In the address byte, the address of the target receiver is compared with its own address by the receiver, and the address and mask bit of the receiver are

The programming is in the UART\_RXADDR and UART\_RXMASK registers.

If the received byte does not match its programmed address, the UART enters silent mode. At this point, the hardware sets RWU

Bit. Receiving this byte will neither set the interrupt status flag RX\_INTF nor generate an interrupt or issue a DMA request, because

The UART is already in silent mode.

When the received byte matches the programmed address in the receiver, the UART exits the silent mode. Then the RWU bit is cleared,

The subsequent bytes are received normally. When receiving this matching address byte, the interrupt status flag RX\_INTF will be set because

The RWU bit has been cleared.

**19.3.6**

The single-wire half-sided mode is selected by setting the HDSEL bit in the UART\_SCR register. In this mode, UART\_SCR

The SCEN bit of the register must remain clear.

The UART can be configured to follow the single-wire half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are inside the chip

interconnection. Use the control bit "HALF DUPLEX SEL" (HDSEL bit in UART\_SCR) to select half duplex and full

Duplex communication.

When HDSEL is 1,

• RX is no longer used

• When there is no data transmission, TX is always released. Therefore, it appears as a standard in idle state or receiving state

Quasi-I/O port. This means that when the I/O is not driven by the UART, it must be configured as a floating input (or open-drain output)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

419/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 442** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

high).

Otherwise, the communication is similar to the normal UART mode. Software to manage online conflicts (for example, by using a

Central arbiter). In particular, sending is never hindered by hardware. When the TXEN bit is set, as soon as the data is written to

On the data register, the transmission continues.

**19.3.7**

Set the SCEN bit of the UART\_SCR register to select the smart card mode.

The interface conforms to the ISO7816-3 standard and supports the smart card asynchronous protocol. The UART should be set to:

• 8-bit data bit plus parity bit: CHAR=111, PEN=1 in the UART\_CCR register

• 1.5 stop bits when sending and receiving: namely SPB1=1, SPB0=1 of UART\_CCR register

The example shown in the figure below illustrates the signal on the data line in the presence of check errors and no check errors.

123456

No parity error

0

1

2

3

4

5

6

7

P

Parity error

0

1

2

3

4

5

6

7

P

S

S

Protection time

Protection time

When a parity error occurs,

The signal line is pulled down by the receiver

230. UART

When connected to a smart card, the TX of the UART drives a bidirectional line of the smart card. In order to do this, RX must

TX is connected to the same I/O port. During the transmission of the start bit and data byte, the transmitter output enable bit TXEN is

Set up, it is released during the transmission of the stop bit (weak pull-up), so if a check error is found, the receiver can

According to the line pulled down. If TXEN is not used, TX is pulled high during the stop bit: in this case, as long as TX is configured

Set to open drain, the receiver can also drive this line.

Smart card is a single-wire half-duplex communication protocol

• Sending data from the transmit shift register is delayed by a minimum of 1/2 baud clock. During normal operation, a

A full transmit shift register will start shifting out data at the next baud clock edge. In smart card mode, this

The transmission is delayed by 1/2 baud clock.

• If a parity error is detected during the reception of a data frame set to 0.5 or 1.5 stop bits, the

After receiving the frame (that is, when the stop bit ends), the transmit line is pulled down for one baud clock cycle. This is telling the smart card to issue

The data sent to the UART was not received correctly. This NACK signal (pull the transmission line down for one baud clock cycle)

A framing error will occur at the sending end (the sending end is configured to 1.5 stop bits). The application can be based on the agreement

Resend the data. If the NACK control bit is set, the receiver will give a NACK when a check error occurs

Signal; otherwise NACK will not be sent.

• The setting of the TXC flag can be delayed by programming the protection time register. In normal operation, when sending the shift mail

When the memory becomes empty and no new transmission request occurs, TXC is set. In smart card mode, empty send shift

The register will trigger the guard time counter to start counting up until the value in the guard time register. TXC in this section

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

420/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 443** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

Time is forced to be lowered. When the guard time counter reaches the value in the guard time register, TXC is set high.

• The revocation of the logo is not affected by the smart card mode.

• If the transmitter detects a frame error (receives the receiver's NACK signal), the transmitter's receive function module will not

Detect NACK as the start bit. According to the ISO protocol, the duration of the received NACK can be 1 or 2

Baud clock cycle.

• On the receiver side, if a check error is detected and NACK is sent, the receiver will not send NACK

Detected as the start bit.

Note: 1. The disconnect symbol has no meaning in smart card mode. A 00h data with frame error will be treated as data without

Is a disconnect symbol.

2. When switching the TXEN bit back and forth, no IDLE frame is sent. The ISO protocol does not define IDLE frames.

The figure below details how the UART samples the NACK signal. In this example, the UART is sending data, and

It is configured with 1.5 stop bits. In order to check the integrity of the data and the NACK signal, the receive function block of the UART is activated

live.

123456

Bit 7

Parity bit

1.5 stop bits

1 bit time

1.5 bit time

On the 8th, 9th,

10th position sampling

On the 16th, 17th,

18th position sampling

On the 8th, 9th,

10th position sampling

On the 8th, 9th,

10th position sampling

0.5 bit time

231. UART

**19.3.8**

Set the BRR and FRA registers, you can set the corresponding baud rate, refer to the following formula:

*f baudrate* =

*f PCLK*

16 × *UARTDIV*

*UARTDIV* = *BRR* +

*FRA*

16

Get:

*f baudrate* =

*f PCLK*

16 × *BRR* + *FRA*

The minimum value of the BRR register is 4.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

421/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 444** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**19.3.9**

Since there is no separate clock for asynchronous operation, the receiver needs a synchronous method to the receiver. To be able to receive the pin

'RX' gets the correct character data, UART has a detection circuit. UART uses 16 times the data baud rate'bclk16'

The clock is used to sample the data of the RX pin. Each data has 16 clock samples, taking the middle of the 7, 8, 9 falling edge

Sampled value.

576531

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

1

2

3

1

Samples

Baud CLK for all but start bit

Start bit

Bin0

RX

Baud CLK

x16 CLK

232. RX

**19.3.10**

Parity control (generate a parity bit when transmitting, and perform parity check when receiving) can be registered by setting UART\_CCR

The PEN bit on the device is activated. If parity error occurs, invalid data will not be transferred from the shift register to UART\_RDR

register.

Even parity: The parity bit makes the data in a frame and the number of '1's in the parity bit even.

For example: data = 00110101, there are 4 '1's, if even parity is selected (PSEL = 1 in UART\_CCR),

The check digit will be '0'.

Odd parity: This parity bit makes the data in a frame and the number of '1's in the parity bit odd.

For example: data = 00110101, there are 4 '1's, if odd parity is selected (PSEL = 0 in UART\_CCR),

The check digit will be '1'.

Transmission mode: If the PEN bit of UART\_CCR is set, the MSB bit of the data written into the data register is verified

Transmitted after bit replacement (if you select even parity even '1's, if you select odd parity odd '1s'). If parity

The verification fails, the RXPERR\_INTF flag in the UART\_ISR register is set to '1', and if RXPERREN

If it is set in advance, an interrupt is generated.

**19.3.11**

Use nCTS input and nRTS output to control the serial data flow between 2 devices. The following figure shows that in this mode

How to connect 2 devices.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

422/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 445** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

686568

TX circuit

UART 1

RX circuit

RX circuit

UART 2

TX circuit

TX

nCTS

RX

nRTS

RX

nRTS

TX

nCTS

233.

UART

By setting AUTOFLOWEN in UART\_GCR, RTS and CTS flow control can be enabled.

**RTS**

If RTS flow control is enabled, nRTS becomes effective as long as the UART receiver is ready to receive new data (connect

Low level). When data arrives in the receive register, nRTS is released, which indicates that you want to stop at the end of the current frame

Stop data transmission. The following figure is an example of communication that enables RTS flow control.

941810

RX

nRTS

Start

Bit

Start

Bit

Stop

Bit

Stop

Bit

ldle

RXNE

Data 1 read

Data 2 can now be transmitted

RXNE

Data 1

Data 2

234. RTS

**CTS**

If CTS flow control is enabled, the transmitter checks the nCTS input before sending the next frame. If nCTS is valid (pulled

Low level), the next data is sent (assuming that the data is ready to be sent), otherwise the next frame of data is not sent

go with. If nCTS becomes invalid during the transmission, the transmission will stop after the current transmission is completed. The following figure is a CTS flow control

Examples of communications that are enabled.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

423/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 446** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

771455

CTS

CTS

Data 3

empty

Data 3

Data 2

Data 1

Data 2

empty

TDR

TX

nCTS

Transmit data register

Stop

Bit

Start

Bit

Start

Bit

Stop

Bit

ldle

Writing data 3 in TDR

Transmission of Data 3 is

delayed until nCTS = 0

235. CTS

**19.3.12**

With **DMA**

UART can use DMA for communication.

With **DMA**

When using DMA for transmission, first configure the address of the UART\_TDR register on the DMA control register to

DMA transfer destination address, configure the memory address as the DMA transfer source address, and configure the amount of data transferred.

DMA mode is activated by setting the DMAMODE bit of the UART\_GCR register. When the TXEN bit is set to '1'

At this time, DMA transfers data from the designated SRAM area to the UART\_TDR register.

With **DMA**

When receiving using DMA, first configure the address of the UART\_RDR register on the DMA control register to

The source address of DMA transfer, configure the memory address as the destination address of DMA transfer, and configure the amount of data transferred.

DMA mode is activated by setting the DMAMODE bit of the UART\_GCR register. When the RXEN bit is enabled, every

When a byte is received, the DMA transfers the data from the UART\_RDR register to the designated SRAM area.

**19.4 UART**

61. UART

Interrupt event

Interrupted state

Enable bit

Send buffer empty

TX\_INTF

TXIEN

Received valid data

RX\_INTF

RXIEN

Send shift register complete

TXC\_INTF

TXC\_EN

Receive overflow error

RXOERR\_INTF

RXOERREN

Parity error

RXPERR\_INTF

RXPERREN

Frame error

RXFERR\_INTF

RXFERREN

Receive disconnected frame

RXBRK\_INTF

RXBRKEN

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

424/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 447** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

Interrupt event

Interrupted state

Enable bit

Send disconnect frame

TXBRK\_INTF

TXBRK\_EN

Receive sync frame

RXB8\_INTF

RXB8\_EN

If the corresponding interrupt enable control bits are set, these settings can generate their corresponding interrupts.

**19.5 UART**

62. UART

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

UART\_TDR

UART transmit data register

0x00000000

[Section 19.5.1](https://translate.googleusercontent.com/translate_f#447)

0x04

UART\_RDR

UART receive data register

0x00000000

[Section 19.5.2](https://translate.googleusercontent.com/translate_f#447)

0x08

UART\_CSR

UART current status register

0x00000009

[Section 19.5.3](https://translate.googleusercontent.com/translate_f#448)

0x0C

UART\_ISR

UART interrupt status register

0x00000000

[Section 19.5.4](https://translate.googleusercontent.com/translate_f#449)

0x10

UART\_IER

UART interrupt enable register

0x00000000

[Section 19.5.5](https://translate.googleusercontent.com/translate_f#450)

0x14

UART\_ICR

UART interrupt clear register

0x00000000

[Section 19.5.6](https://translate.googleusercontent.com/translate_f#451)

0x18

UART\_GCR

UART global control register

0x00000000

[Section 19.5.7](https://translate.googleusercontent.com/translate_f#452)

0x1C

UART\_CCR

UART general control register

0x00000030

[Section 19.5.8](https://translate.googleusercontent.com/translate_f#453)

0x20

UART\_BRR

UART baud rate register

0x00000001

[Section 19.5.9](https://translate.googleusercontent.com/translate_f#454)

0x24

UART\_FRA

UART Fractional Baud Rate Register

0x00000000

[Section 19.5.10](https://translate.googleusercontent.com/translate_f#455)

0x28

UART\_RXADDR

UART receive address register

0x00000000

[Section 19.5.11](https://translate.googleusercontent.com/translate_f#455)

0x2C

UART\_RXMASK

UART receive mask register

0x000000FF

[Section 19.5.12](https://translate.googleusercontent.com/translate_f#456)

0x30

UART\_SCR

UART SCR register

0x00000000

[Section 19.5.13](https://translate.googleusercontent.com/translate_f#456)

**19.5.1 UART**

**UART\_TDR**

Offset address: 0x00

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

Reserved

TXREG

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 8

Reserved

Always read as 0.

7: 0

TXREG

rw

0x00

Transmit data register (Transmit data register)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

425/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 448** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**19.5.2 UART**

**UART\_RDR**

Offset address: 0x04

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

r

r

r

r

r

r

r

r

Reserved

Reserved

RXREG

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 8

Reserved

Always read as 0.

7: 0

RXREG

r

0x00

Receive data register (Receive data register)

This register is read-only

**19.5.3 UART**

**UART\_CSR**

Offset address: 0x08

Reset value: 0x0000 0009

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

TXBUF\_

EMPTY

TXFULL

RXAVL

TXC

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 4

Reserved

Always read as 0.

3

TXBUF\_

EMPTY

r

0x01

Transmit buffer empty flag bit

1: Send buffer is empty

0: Send buffer is not empty

2

TXFULL

r

0x00

Transmit buffer full flag bit

1: Send buffer is full

0: Send buffer is not full

1

RXAVL

r

0x00

Receive valid data flag bit (Receive valid data flag bit)

This bit is set when the receive buffer receives a complete byte of data.

1: Receive buffer received a complete and valid byte data

0: Receive buffer is empty

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

426/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 449** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**Bit**

**Field**

**Type**

**Reset**

**Description**

0

TXC

r

0x01

Transmit complete flag bit

1: Both the transmit buffer and the transmit shift register are empty

0: Send is not empty

**19.5.4 UART**

**UART\_ISR**

Offset address: 0x0C

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

r

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

RX\_

INTF

RX

BPK\_

INTF

RXP

ERR\_

INTF

RXO

ERR\_

INTF

TX\_

INTF

TX

BRK\_

INTF

RXB8

\_INTF

TXC\_

INTF

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 9

Reserved

Always read as 0.

8

RXB8\_ INTF

r

0x00

UART synchronization frame interrupt flag. In 9-bit communication mode, when receiving

When the ninth bit of the data is the same as the register CCR.B8POL, RXB8\_INT

position. This bit can be used as an interrupt request signal

1: Received sync frame

0: No sync frame received

7

TXBRK\_

INTF

r

0x00

UART disconnect frame transmission complete interrupt flag bit.

1: Shift register disconnection frame data transmission is completed

0: The shift register is empty or is being shifted to send

Note: Disconnect frames cannot be sent continuously.

6

RXBRK\_

INTF

r

0x00

UART receive frame break inter-bit flag (Receive frame break inter-

rupt flag bit)

After the abnormal stop bit, the RX pin receives 10 or more within a period of time

It is a low level of 10 bits.

1: Detect disconnected frames

0: No disconnect frame

5

Reserved

Always read as 0.

4

RXPERR\_

INTF

r

0x00

Parity error interrupt flag bit (Parity error interrupt flag bit)

1: Parity error detected

0: No parity error

3

RXOERR\_

INTF

r

0x00

Receive overflow error interrupt flag (Receive overflow error interrupt flag

flag bit)

Set only when autoflowen=0.

1: Receive overflow error

0: No overflow error

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

427/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 450** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**Bit**

**Field**

**Type**

**Reset**

**Description**

2

TXC\_INTF

r

0x00

The UART sends the shift register completion interrupt flag.

1: Shift register data transmission is completed

0: The shift register is empty or is being shifted to send

Note: This flag is related to the protection time

1

RX\_INTF

r

0x00

Receive valid data interrupt flag

bit)

This bit is set when the receive buffer receives a complete byte of data.

1: Receive buffer valid byte data

0: Receive buffer is empty

0

TX\_INTF

r

0x00

Transmit buffer empty interrupt flag (Transmit buffer empty interrupt flag

flag bit)

1: Send buffer empty

0: Send buffer is not empty

**19.5.5 UART**

**UART\_IER**

Offset address: 0x10

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

rw

rw

rw

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

RXIEN

RX

BPK

EN

RXP

ERR

EN

RXO

ERR

EN

TXIEN

TXBRK

\_IEN

RXB8

\_IEN

TXC\_

IEN

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 9

Reserved

Always read as 0.

8

RXB8\_IEN

rw

0x00

UART synchronization frame interrupt enable control bit.

1: Enable receive sync frame interrupt

0: Disable the reception of synchronization frame interrupt

7

TXBRK\_IEN

rw

0x00

UART disconnect frame transmission complete interrupt enable control bit.

1: Enable interrupt for sending disconnected frames

0: Disable the interruption of sending disconnected frames

6

RXBRKEN

rw

0x00

UART receive frame break interrupt enable bit (Receive frame break inter-

rupt enable bit)

1: Interrupt enable

0: Interrupt disabled

5

Reserved

Always read as 0.

4

RXPERREN

rw

0x00

Parity error interrupt enable bit (Parity error interrupt enable bit)

1: Interrupt enable

0: Interrupt disabled

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

428/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 451** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**Bit**

**Field**

**Type**

**Reset**

**Description**

3

RXOERREN

rw

0x00

Receive overflow error interrupt enable bit (Receive overflow error interrupt

enable bit)

1: Interrupt enable

0: Interrupt disabled

2

TXC\_IEN

rw

0x00

UART transmit shift register complete interrupt enable control bit.

1: Shift register data transmission is completed

0: The shift register is empty or is being shifted to send

1

RXIEN

rw

0x00

Receive buffer interrupt enable bit (Receive buffer interrupt enable bit)

1: Interrupt enable

0: Interrupt disabled

0

TXIEN

rw

0x00

Transmit buffer empty interrupt enable bit (Transmit buffer empty interrupt

enable bit)

1: Interrupt enable

0: Interrupt disabled

**19.5.6 UART**

**UART\_ICR**

Offset address: 0x14

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

w

w

w

w

w

w

w

w

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

RXICLR

RX

BPK

CLR

RXP

ERR

CLR

RXO

ERR

CLR

TXICLR

TXBRK

\_CLR

RXB8

\_CLR

TXC\_

CLR

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 9

Reserved

Always read as 0.

8

RXB8\_CLR

w

0x00

The UART sync frame interrupt flag clears the control bit.

1: Clear the received synchronization frame interrupt flag

0: No action

7

TXBRK\_CLR

w

0x00

The UART disconnect frame transmission completion interrupt flag clears the control bit.

1: Clear the interruption flag of disconnected frame transmission completion

0: No action

6

RXBRKCLR

w

0x00

UART Receive frame break inter-clear bit (Receive frame break inter-

rupt clear bit)

1: Interrupt clear

0: Interrupt is not cleared

5

Reserved

Always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

429/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 452** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**Bit**

**Field**

**Type**

**Reset**

**Description**

4

RXPERRCLR w

0x00

Parity error interrupt clear bit (Parity error interrupt clear bit)

1: Interrupt clear

0: Interrupt is not cleared

3

RXOERRCLR w

0x00

Receive overflow error interrupt clear bit (Receive overflow error interrupt

clear bit)

1: Interrupt clear

0: Interrupt is not cleared

2

TXC\_CLR

w

0x00

UART transmit shift register complete interrupt enable control bit.

1: Clear the shift register data transmission completion interrupt flag

0: No action

1

RXICLR

w

0x00

Receive interrupt clear bit (Receive interrupt clear bit)

1: Interrupt clear

0: Interrupt is not cleared

0

TXICLR

w

0x00

Transmit buffer empty interrupt clear bit

clear bit)

1: Interrupt clear

0: Interrupt is not cleared

**19.5.7 UART**

**UART\_GCR**

Offset address: 0x18

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

rw

rw

rw

rw

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

DMA

MODE

AUTO

FLOW

EN

TXEN RXEN

UARTEN

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 5

Reserved

Always read as 0.

4

TXEN

rw

0x00

Send enable bit (Enable transmit)

1: Send enable

0: transmission prohibited. Can clear TX BUFFER

3

RXEN

rw

0x00

Receive enable bit (Enable receive)

1: Receive enable

0: Reception prohibited. RX BUFFER can be cleared.

2

AUTO

FLOWEN

rw

0x00

Automatic flow control enable bit (Automatic flow control enable bit)

1: Automatic flow control is enabled

0: Automatic flow control is disabled

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

430/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 453** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**Bit**

**Field**

**Type**

**Reset**

**Description**

1

DMAMODE

rw

0x00

DMA mode selection bit

1: Select DMA mode

0: select normal mode

0

UARTEN

rw

0x00

UART module selection bit (UART mode selection bit)

1: UART module is enabled

0: UART module disabled

**19.5.8 UART**

Use **UART\_CCR**

Offset address: 0x1C

Reset value: 0x0000 0030

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

Reserved

CHAR

BRK SPB0 PSEL PEN

WAKE RWU B8EN

SPB1

B8RXD

B8TXD

B8POL

B8TOG

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 14

Reserved

Always read as 0.

13

WAKE

rw

0x00

Wake up method. This bit determines how to wake up the UART

1: Wake up from address mark

0: Idle bus wakeup

Note: Before putting the UART into silent mode, the UART must have received

One data byte. Otherwise, in silent mode, it cannot be detected by the idle bus

wake.

12

RWU

rw

0x00

Receive wake up. This bit is used to decide whether to put the UART into silent mode. The

Bits can be set or cleared by software. When the wake-up sequence comes, the hardware will also

It is automatically cleared.

1: The receiver is in silent mode

0: receiver is in normal operating mode

When setting the address tag to wake up, if the receiving buffer is not empty, the software cannot be used

modify.

11

B8EN

rw

0x00

The ninth bit of the UART sync frame enables control bits. Check enable bit after this bit is enabled

PEN does not work.

1: Enable the ninth bit of the synchronization frame

0: Disable the ninth bit of the synchronization frame

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

431/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 454** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

**Bit**

**Field**

**Type**

**Reset**

**Description**

10

B8TOG

rw

0x00

The UART synchronization frame sends the ninth bit of automatic flip control bit.

1: Enable the ninth bit auto flip

0: Disable the ninth digit from flipping automatically

Note: When B8TXD and B8POL are the same, after configuring the registers

The second data transferred begins to flip, and the first data defaults to the address bit.

9

B8POL

rw

0x00

The ninth polarity control bit of the UART synchronization frame.

1: The ninth bit of the sync frame is active high

0: The ninth bit of the sync frame is active low

8

B8TXD

rw

0x00

The ninth bit of UART synchronization frame transmission data.

1: The ninth bit of the sync frame is high

0: The ninth bit of the synchronization frame is low

7

B8RXD

rw

0x00

The ninth bit of UART synchronization frame received data. Read only.

1: The ninth bit of the received sync frame is high

0: The ninth bit of the received sync frame is low

6

SPB1

rw

0x00

Stop bit selection bit, combined with SPB0 to set the number of stop bits.

5: 4

CHAR

rw

0x03

UART width bit

00: 5 digits

01: 6 digits

10: 7 digits

11: 8 bits

3

BRK

rw

0x00

UART transmit frame break (UART transmit frame break)

1: Serial forced output logic '0' (disconnect frame)

0: Disconnect is prohibited

2

SPB0

rw

0x00

Stop bit selection (Stop bit selection)

Set the number of transmission stop bits.

SPB1, SPB0: 00, 1 stop bit

SPB1, SPB0: 01, 2 stop bits (with 5 data bits, SPB is not used

Set, the stop bit is forced to 1 bit)

SPB1, SPB0: 10, 0.5 stop bits

SPB1, SPB0: 11, 1.5 stop bits

1

PSEL

rw

0x00

Parity selection bit

When parity is enabled, this bit is used to select whether to use even parity or odd parity.

1: Even parity

0: odd parity

0

PEN

rw

0x00

Parity enable bit

1: Send and receive enable check

0: Disable verification

**19.5.9 UART**

**UART\_BRR**

Offset address: 0x20

Reset value: 0x0000 0001

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

432/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 455** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

DIV\_Mantissa

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

DIV\_Mantissa rw

0x01

Integer part of UARTDIV

These 16 bits define the integral of the UART divider division factor (UARTDIV)

Several parts.

DIV\_Mantissa has a minimum value of 4

**19.5.10 UART**

**UART\_FRA**

Offset address: 0x24

Reset value: 0x0000 0000

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

Reserved

DIV\_Fraction

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 4

Reserved

Always read as 0.

3: 0

DIV\_Fraction

rw

0x00

Fractional part of UARTDIV

These 4 bits define the fraction of the UART divider division factor (UARTDIV)

section.

**19.5.11 UART**

**UART\_RXADDR**

Offset address: 0x28

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

433/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 456** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

RXADDR

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 8

Reserved

Always read as 0.

7: 0

RXADDR

rw

0x00

UART synchronization frame data local matching address. If RXMASK = 0xFF

When the received synchronization frame data is the same as the matching address of the machine, it is generated

RXB8\_INTF.

Address 0 is a broadcast address, and will respond when received.

**19.5.12 UART**

**UART\_RXMASK**

Offset address: 0x2C

Reset value: 0x0000 00FF

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

rw

rw

rw

rw

rw

rw

rw

rw

Reserved

RXMASK

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 8

Reserved

Always read as 0.

7: 0

RXMASK

rw

0xFF

When the data bits are all "0", the synchronization frame interrupt is generated when any data is received.

begging.

If the data bit is "1", when the corresponding bits of RDR and RXADDR match, the

Generate synchronization frame interrupt request.

**19.5.13 UART SCR**

**UART\_SCR**

Offset address: 0x30

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

434/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 457** |

UM\_MM32F003\_q\_Ver1.19

Use (UART)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

rw

r

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

Reserved

Reserved

SCEN

SCAE

N

NACK

SCFCNT

HDSE

L

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 13

Reserved

Reserved, always read as 0.

12

HDSEL

rw

0x00

Single line half duplex mode selection.

1: enable half-duplex mode

0: Disable half-duplex mode

11: 4

SCFCNT

rw

0x00

ISO7816 protection counter. When sending data within the protection counter period

The low level of is prohibited as the start bit of the next data.

0: 16 baud rate counting time

15..1 is 15..1 baud rate count time

3

Reserved

Reserved, always read as 0.

2

NACK

r

0x00

Main receive frame response bit

1

SCAEN

rw

0x00

ISO7816 verifies the automatic answer bit.

1: enable auto answer

0: Disable auto answer

0

SCEN

rw

0x00

ISO7816 enables control bits.

1: Enable ISO7816 function

0: Disable ISO7816 function

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

435/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 458** |

UM\_MM32F003\_q\_Ver1.19

(HWDIV)

20

**(HWDIV)**

Hardware division (HWDIV)

**20.1**

Hardware division is very useful in some high-performance applications, and can automatically perform signed or unsigned 32-bit integer division

Method calculation.

**20.2**

• Signed or unsigned integer division

• 32-bit divisor and dividend, output 32-bit quotient and remainder

• 8 HCLK cycles completed

• If the divisor is zero, an overflow interrupt flag will be generated

• Write divisor to automatically perform division

• When reading the quotient and remainder registers, it automatically waits for the operation to finish without checking the status bit

**20.3**

The hardware division unit includes four 32-bit data registers, which are the dividend, divisor, quotient, and remainder.

Sign or unsigned 32-bit division. You can choose signed division through the hardware division control register USIGN

Or unsigned division.

Each time the divisor register is written, the division operation is automatically triggered. After the operation is completed, the result is written to the quotient and remainder.

Memory. If the quotient register, remainder register or status register is read before the end, the read operation will be suspended until

The operation result is returned only after the end.

If the divisor is zero, an overflow interrupt flag will be generated.

**20.4**

63.

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

HWDIV\_DVDR

Dividend register

0x00000000

[Section 20.4.1](https://translate.googleusercontent.com/translate_f#458)

0x04

HWDIV\_DVSR

Divisor register

0x00000001

[Section 20.4.2](https://translate.googleusercontent.com/translate_f#459)

0x08

HWDIV\_QUOTR

Quotient register

0x00000000

[Section 20.4.3](https://translate.googleusercontent.com/translate_f#459)

0x0C

HWDIV\_RMDR

Remainder register

0x00000000

[Section 20.4.4](https://translate.googleusercontent.com/translate_f#460)

0x10

HWDIV\_SR

HWDIV status register

0x00000000

[Section 20.4.5](https://translate.googleusercontent.com/translate_f#460)

0x14

HWDIV\_CR

HWDIV control register

0x00000001

[Section 20.4.6](https://translate.googleusercontent.com/translate_f#460)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

436/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 459** |

UM\_MM32F003\_q\_Ver1.19

(HWDIV)

**20.4.1**

**(HWDIV\_DVDR)**

Offset address: 0x00

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

DIVIDEND

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

DIVIDEND

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

DIVIDEND

rw

0x0000

0000

Dividend register bit (Dividend data)

**20.4.2**

**(HWDIV\_DVSR)**

Offset address: 0x04

Reset value: 0x0000 0001

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

DIVISOR

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

DIVISOR

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

DIVISOR

rw

0x0000

0001

Divisor data

After writing this register, the division operation is automatically triggered.

**20.4.3**

**(HWDIV\_QUOTR)**

Offset address: 0x08

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

QUOTIENT

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

QUOTIENT

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

437/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 460** |

UM\_MM32F003\_q\_Ver1.19

(HWDIV)

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

QUOTIENT

r

0x0000

0000

Quotient data

**20.4.4**

**(HWDIV\_RMDR)**

Offset address: 0x0C

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

REMAINDER

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

REMAINDER

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 0

REMAINDER

r

0x0000

0000

Remainder data (Remainder data)

**20.4.5 HWDIV**

**(HWDIV\_SR)**

Offset address: 0x10

Reset value: 0x0000 0000

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

r

OVF

Res.

Res.

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 1

Reserved

Reserved, always read as 0.

0

OVF

r

0x00

Overflow status flag bit (Overflow)

Automatically clear before the next division operation

1: The current divisor is zero.

0: The current operation divisor is not zero.

**20.4.6 HWDIV**

**(HWDIV\_CR)**

Offset address: 0x14

Reset value: 0x0000 0001

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

438/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 461** |

UM\_MM32F003\_q\_Ver1.19

(HWDIV)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

Res.

Res.

OVFE USIGN

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 2

Reserved

Reserved, always read as 0.

1

OVFE

rw

0x00

Overflow interrupt enable (Overflow interrupt enable)

1: Divide by zero overflow interrupt enable

0: division by zero overflow interrupt is not enabled

0

USIGN

rw

0x01

Unsigned enable

1: Unsigned division

0: signed division

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

439/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 462** |

UM\_MM32F003\_q\_Ver1.19

(SYSCFG)

twenty one

**(SYSCFG)**

System Configuration Controller (SYSCFG)

The chip has a set of system configuration registers. The main functions of these registers are as follows:

• The remapping part is from the DMA trigger source of TIM16 and TIM17 and ADC to other different DMA channels.

• Manage external interrupts connected to the GPIO port.

• Remap memory to the code start area.

• External interrupt pin configuration

**21.1 SYSCFG**

64. SYSCFG

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

SYSCFG\_CFGR

SYSCFG configuration register

0x0000000X

[Section 21.1.1](https://translate.googleusercontent.com/translate_f#462)

0x08

SYSCFG\_EXTICR1

External interrupt configuration register 1

0x00000000

[Section 21.1.2](https://translate.googleusercontent.com/translate_f#463)

0x0C

SYSCFG\_EXTICR2

External interrupt configuration register 2

0x00000000

[Section 21.1.3](https://translate.googleusercontent.com/translate_f#464)

0x10

SYSCFG\_EXTICR3

External Interrupt Configuration Register 3

0x00000000

[Section 21.1.4](https://translate.googleusercontent.com/translate_f#464)

0x14

SYSCFG\_EXTICR4

External interrupt configuration register 4

0x00000000

[Section 21.1.5](https://translate.googleusercontent.com/translate_f#465)

**21.1.1 SYSCFG**

**SYSCFG\_CFGR**

This register is used to configure the memory start area mapping and DMA request remapping. Has two configurable memory start

0x0000 0000 The control bits of the address storage area type. These two control bits can be configured by software to shield the selection of BOOT.

After reset, these two control bits are configured for the actual BOOT mode.

Offset address: 0x00

Reset value: 0x0000 000X (X is the actual BOOT mode selection control bit)

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

rw

rw

rw

rw

rw

Reserved

MEM\_ MODE

Reserved

ADC

\_DMA

\_RMP

TIM16

\_DMA

\_RMP

TIM17

\_DMA

\_RMP

Reserved

Reserved

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 13

Reserved

Always read as 0.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

440/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 463** |

UM\_MM32F003\_q\_Ver1.19

(SYSCFG)

**Bit**

**Field**

**Type**

**Reset**

**Description**

12

TIM17\_DMA

\_RMP

rw

0x00

TIM17 DMA request remap-bit (TIM17 DMA request remap-bit

ping bit)

This bit is set and cleared by software. It controls the TIM17 DMA channel request

Remapping.

0: No remapping (TIM17\_CH1 and TIM17\_UP DMA request mapping

On DMA channel 1)

1: remapping (TIM17\_CH1 and TIM17\_UP DMA request mapping in

DMA channel 2)

11

TIM16\_DMA

\_RMP

rw

0x00

TIM16 DMA request remap bit (TIM16 DMA request remap-

ping bit)

This bit is set and cleared by software. It controls the TIM16 DMA channel request

Remapping.

0: No remapping (TIM16\_CH1 and TIM16\_UP DMA request mapping

On DMA channel 3)

1: remapping (TIM16\_CH1 and TIM16\_UP DMA request mapping in

DMA channel 4)

10 :9

Reserved

Always read as 0.

8

ADC\_DMA

\_RMP

rw

0x00

ADC DMA request remapping bit (ADC DMA request remapping bit

bit)

This bit is set and cleared by software. It controls what the ADC DMA channel requests

Remapping

0: No remapping (ADC DMA request is mapped on DMA channel 1)

1: Remapping (ADC DMA request is mapped on DMA channel 2)

7: 2

Reserved

Always read as 0.

1: 0

MEM\_MODE

rw

0x00

Memory selection bit (Memory selection bit) is set and cleared by software

These bits. It controls the internal memory mapping to address 0x0000 0000. when

The value of these bits after reset is determined by the pin configuration value of BOOT0 and the nBOOT1 bit

The value determines.

x0: The main flash memory is mapped to 0x0000 0000

01: The system flash memory is mapped to 0x0000 0000

11: Embedded RAM mapped to 0x0000 0000

**21.1.2**

**1 SYSCFG\_EXTICR1**

Offset address: 0x08

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

441/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 464** |

UM\_MM32F003\_q\_Ver1.19

(SYSCFG)

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

EXTI0

EXTI3

EXTI2

EXTI1

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

EXTIx

rw

0x00

EXTIx configuration (x = 0…3) (EXTI x configuration)

These bits can be used for software reading and writing. Input for selecting EXTIx external interrupt

source.

0000: PA[x] pin

0001: PB[x] pin

0010: PC[x] pin

0011: PD[x] pin

**21.1.3**

**2 SYSCFG\_EXTICR2**

Offset address: 0x0C

Reset value: 0x0000 0000

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

EXTI4

EXTI7

EXTI6

EXTI5

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

EXTIx

rw

0x00

EXTIx configuration (x = 4…7) (EXTI x configuration)

These bits can be used for software reading and writing. Input for selecting EXTIx external interrupt

source.

0000: PA[x] pin

0001: PB[x] pin

0010: PC[x] pin

0011: PD[x] pin

**21.1.4**

**3 SYSCFG\_EXTICR3**

Offset address: 0x10

Reset value: 0x0000 0000

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

442/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 465** |

UM\_MM32F003\_q\_Ver1.19

(SYSCFG)

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

EXTI8

EXTI11

EXTI10

EXTI9

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

EXTIx

rw

0x00

EXTIx configuration (x = 8...11) (EXTI x configuration)

These bits can be used for software reading and writing. Input for selecting EXTIx external interrupt

source.

0000: PA[x] pin

0001: PB[x] pin

0010: PC[x] pin

0011: PD[x] pin

**21.1.5**

**4 SYSCFG\_EXTICR4**

Offset address: 0x14

Reset value: 0x0000 0000

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

rw

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

16

27

28

29

30

31

26

1

2

3

4

5

6

7

8

9

0

11

12

13

14

15

10

Reserved

EXTI12

EXTI15

EXTI14

EXTI13

**Bit**

**Field**

**Type**

**Reset**

**Description**

31: 16

Reserved

Always read as 0.

15: 0

EXTIx

rw

0x00

EXTIx configuration (x = 12...15) (EXTI x configuration)

These bits can be used for software reading and writing. Input for selecting EXTIx external interrupt

source.

0000: PA[x] pin

0001: PB[x] pin

0010: PC[x] pin

0011: PD[x] pin

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

443/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 466** |

UM\_MM32F003\_q\_Ver1.19

(DEVICE)

twenty two

**(Device)**

Device Electronic Signature (Device)

The electronic signature is stored in the system storage area of ​​​​​the flash memory module and can be read by JTAG, SWD or CPU.

The chip identification information it contains is written at the factory, and user firmware or external devices can read the electronic signature to

Automatically match microcontrollers with different configurations.

**22.1**

**22.1.1**

**(96**

**)**

The unique identification of the product is very suitable:

• Used as a serial number (such as a USB character serial number or other terminal applications).

• Used as a password, when writing the flash memory, use this unique identifier in combination with the software encryption and decryption algorithm to improve the code in

Flash memory security.

• Used to activate the bootstrapping process with safety mechanism.

The reference number provided by the 96-digit product unique identification is for any series of microcontrollers, in any case

only. Under no circumstances can the user modify this identity.

This 96-bit product unique identifier can be read in units of bytes (8 bits) according to the usage not used by the user.

It can be read in half words (16 bits) or full words (32 bits).

**22.2 CRS**

65.

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

UID1

Unique identification code

0xXXXXXXXX

[Section 22.2.1](https://translate.googleusercontent.com/translate_f#466)

0x02

UID2

Unique identification code

0xXXXXXXXX

[Section 22.2.2](https://translate.googleusercontent.com/translate_f#467)

0x04

UID3

Unique identification code

0xXXXXXXXX

[Section 22.2.3](https://translate.googleusercontent.com/translate_f#467)

0x08

UID4

Unique identification code

0xXXXXXXXX

[Section 22.2.4](https://translate.googleusercontent.com/translate_f#467)

**22.2.1**

**(UID1)**

Base address: 0x1FFF F7E8

Address offset: 0x00

Read only, its value is written at the factory

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

444/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 467** |

UM\_MM32F003\_q\_Ver1.19

(DEVICE)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

U\_ID

**Bit**

**Field**

**Type**

**Reset**

**Description**

15 :0

U\_ID

r

U\_ID: 15:0 unique ID bits (15:0 unique ID bits)

The value of this field is also reserved for other future functions.

**22.2.2**

**(UID2)**

Address offset: 0x02

Read only, its value is written at the factory

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

U\_ID

**Bit**

**Field**

**Type**

**Reset**

**Description**

15 :0

U\_ID

r

U\_ID: unique identification mark 31: 16 bits (31: 16 unique ID bits)

The value of this field is also reserved for other future functions.

**22.2.3**

**(UID3)**

Address offset: 0x04

Read only, its value is written at the factory

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

U\_ID

U\_ID

**Bit**

**Field**

**Type**

**Reset**

**Description**

31 :0

U\_ID

r

U\_ID: unique identification mark 63: 32 bits (63: 32 unique ID bits)

The value of this field is also reserved for other future functions.

**22.2.4**

**(UID4)**

Address offset: 0x08

Read only, its value is written at the factory

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

445/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 468** |

UM\_MM32F003\_q\_Ver1.19

(DEVICE)

15

1

2

3

4

5

6

7

8

9

10

11

12

13

14

0

31

17

18

19

20

twenty one

twenty two

twenty three

twenty four

25

26

27

28

29

30

16

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

U\_ID

U\_ID

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:0

U\_ID

r

U\_ID: unique identification mark 95: 64 bits (95: 64 unique ID bits)

The value of this field is also reserved for other future functions.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

446/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 469** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

twenty three

**(DBG)**

Debugging support (DBG)

**23.1**

This series of cores contains hardware debugging modules to support complex debugging operations. The hardware debugging module allows the kernel to fetch

Make a breakpoint) or access data (data breakpoint) to stop. When the kernel is stopped, the internal state of the kernel and the external state of the system

The status can be queried. After completing the query, the kernel and peripherals can be restored and the program will continue to execute.

When the series of microcontrollers are connected to the debugger and start debugging, the debugger will use the hardware debugging module of the kernel to adjust

Trial operation.

stand by:

• Serial debug interface

871928

Cortex-M

Core

Bus matrix

Bridge

NVIC

DWT

FPB

TPIU

DCode

interface

SWJ-DP

Debug AP

JTDO/

TRACESWO

JNTRST

Cortex-M0 debug support

CC debug support

JTMS/

SWDIO

JTDI

JTCK/

SWCLK

AHB-AP

Data

Internal Private

Peripheral Bus(PPB)

ITM

External Private

Peripheral Bus (PPB)

System

interface

DBGMCU

Trace Port

TRACESWO

TRACECK

TRACED[3:0]

236. MM32

CPU

The CPU core provides integrated on-chip debugging functions. It consists of the following parts:

• SW-DP: serial debug port

• AHP-AP: AHB access terminal

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

447/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 470** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

• ITM: execution tracking unit

• FPB: Flash instruction breakpoint

• DWT: data trigger

• TPUI: Tracking unit interface

**23.2**

The different packages of the chip microcontroller have different effective pin numbers. Therefore, some pin-related functions may be

Vary.

**23.2.1 SWD**

The 2 ordinary I/O ports of this chip can be used as SW-DP interface pins. These pins are present in all packages.

66. SWJ

**SWJ-DP** port pin name

**SW** debug interface

Pin assignment

Types of

Debugging function

SW debug interface

input Output

Serial data input/output

PA13

SWCLK

enter

Serial clock

PA14

**23.2.2 SWD**

It is necessary to ensure that the input pins of SWD are not floating, because they are directly connected to the D flip-flop to control the adjustment

Test mode. Special attention must be paid to the SWCLK pin because they are directly connected to the clock terminals of some D flip-flops.

To avoid any uncontrolled I/O levels, the chip embeds internal pull-ups and pull-downs on the SWD input pin.

• SWDIO: internal pull-up

• SWCLK: input with pull-down

The software can use these I/O ports as ordinary I/O ports.

**23.3 ID**

There are multiple ID codes inside the chip.

**23.3.1**

**Id**

The microcontroller contains an MCU ID code. This ID defines the part number and silicon version of the MCU. It is DBG\_MCU

It is an integral part of and mapped to the external APB bus. SW debug port (2 pins) or through user code

Both can access this code.

DBGMCU\_IDCODE

Address: 0x40013400 only supports 32-bit access

Read only = 0xXXXXXXXX, where X is a bit with uncertain content

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

448/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 471** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

31

30

29

28

27

26

25

twenty four

twenty three

twenty two

twenty one

20

19

18

17

16

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

DEV\_ID

DEV\_ID

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

r

Bit 31: 0

DEV\_ID: Device identification code (Device identifier)

**23.3.2 Cortex JEDEC-106 ID**

The CPU has a JEDEC-106 ID code. It is located at the address mapped to the internal APB bus at 0xE00FF000\_

0xE00FFFFF in the 4KB ROM table.

The following table is the ID codes of the series:

68. ID

**ID** name

chip

DEV\_ID

0xCC4460B1

CPU TAP SW ID

0x0BB11477

**23.4 SW**

**23.4.1 SW**

This synchronous serial protocol uses 2 pins:

• SWCLK: clock signal from the host to the target

• SWDIO: bidirectional data signal

The protocol allows reading and writing of two register sets (DPACC and APACC register sets).

Data bits are transmitted in LSB.

Since SWDIO is a bidirectional port, this pin needs to be pulled up (100K resistor is recommended).

Every time the SWDIO direction changes according to the protocol, a conversion time needs to be inserted. During this period the host and target are not driven

This signal line. The default value of the conversion time is 1 bit, but it can be adjusted by configuring the SWCLK frequency.

**23.4.2 SW**

Each sequence consists of 3 stages:

• Host sends packet request (8 bits)

• Target sends confirmation response (3 digits)

• Host or target send data (33 bits)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

449/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 472** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

69.

8

Bit

name

description

0

Start

Must be 1

1

APnDP

0: access DP 1: access AP

2

RnW

0: write request 1: read request

4:3

A (3:2)

Address of DP or AP register

5

Parity

Check digit of the previous bit

6

Stop

0

7

Park

Cannot be driven by the host, the target will always read as 1 due to the pull-up

For detailed descriptions of the DPACC and APACC register descriptions, please refer to the CPU Technical Reference Manual.

The packet request is always followed by a (1 bit by default) conversion time, at which time neither the host nor the target drives the line.

70.

3

Bit

name

description

0..2

ACK

001: failed

010: Wait

100: Success

When ACK is a failure or wait, or an ACK that replies to a read operation, there is a transition time after this ACK.

71.

33

Bit

name

description

0..31

WDATA/RDATA written or read data

32

Parity

Parity of 32-bit data

There is a conversion time after the data transfer operation of the read operation.

**23.4.3 SW-DP**

**Reset idle states ID code**

The SW-DP state machine has an internal ID code to identify the SW-DP, which complies with the JEP-106 standard.

The SW-DP state machine does not work until the debugger reads this ID code.

• The SW-DP state machine will be in the RESET state. After power-on reset, or after the DP is switched to SWD, there may be more than

50 cycles of high level.

• When the state machine is in the RESET state, if there is at least 2 cycles of low level, the state machine will switch to IDLE state

state.

• When the state machine is in the RESET state, it must first enter the IDLE state and perform a read DP-SW ID send

The operation of the memory. Otherwise, the debugger can only get a failed ACK response when performing other transfers.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

450/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 473** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

**23.4.4 DP**

**AP**

**/**

• The read operation of DP is not transitive: the debugger will get the data directly (if ACK = successful), or wait (eg

If ACK = wait).

• The read operation of the AP is transitive. This means that the result of the previous read operation can only be obtained in the next operation. Such as

If the next operation is not access to the AP, you must read the DP-RDBUFF register to get the last read operation

the result of.

• The READOK flag in the DP-CTRL/STAT register will be changed after each AP read operation and RDBUFF read operation

Update to inform the debugger whether the AP's read operation is successful.

• SW-DP has a write buffer (both DP and AP have a write buffer), which allows other transmissions to continue

Subject to write operations. If the write buffer is full, the debugger will get a waiting ACK response. Read the IDCODE register,

Reading the CTRL/STAT register and writing the ABORT register are still accepted when the write buffer is full.

• Due to the asynchronous nature of SWCLK and HCLK, it is necessary to insert 2 additional after the write operation (after the parity bit)

SWCLK cycle to ensure that the internal write operation is completed correctly. These two extra clock cycles are required when the line is low

Insert (in IDLE state). This operation step is especially important when writing a CTRL/STAT register to make a power-on request

It is important, otherwise the next operation (an operation that is only valid after the kernel is powered on) will be executed immediately, which will lead to failure.

**23.4.5 SW-DP**

When APnDP=0, the following registers can be accessed.

**A** ( **3** : **2** )

Read **/** write

**SELECT** register

CTRLSEL bit

register

description

00

read

IDCODE

Fixed at 0x1BA0 1477 (used to identify SW-DP).

00

write

ABORT

01

Read/write

0

DP-

CTRL/STAT

Request a system or debug power-up operation;

Configure the operation mode of AP access;

Control comparison and verification operations;

Read some status bits (overflow, power-on response).

01

Read/write

1

WIRE

CON-

TROL

Configure serial communication physical layer protocol (such as conversion time length, etc.).

10

read

READ

RE-

SEND

Allows data to be recovered from an erroneous debug transmission without repeating the original

AP transmission.

10

write

SELECT

Select the current access port and valid 4-word register window.

11

Read/write

READ

BUFFER

Because the access of the AP is transitive (the result of the current AP read operation will be in

It comes out at the next AP transmission), so this register is very necessary. Send this

The memory will capture the data result of the last read operation from the AP, so you can get

Data without having to start a new AP transmission.

**23.4.6 SW-AP**

When APnDP=1, you can access

The access address of the AP register consists of the following two parts:

• The value of A[3:2]

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

451/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 474** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

• The current value of the DP SELECT register

**23.5 MCU**

**MCUDBG**

The MCU debugging module assists the debugger to provide the following functions:

• Low power mode

• Provide timer and watchdog clock control at breakpoint

• Control of tracking pin assignment

**23.5.1**

Use WFI and WFE to enter a low-power mode. MCU supports multiple low-power modes, which can turn off the CPU

Clock, or reduce CPU power consumption. The kernel does not allow FCLK or HCLK to be turned off during debugging. These clocks are

Debugging operations are necessary, so they must work during debugging. The MCU uses a special method that allows

The user debugs the code in low power mode.

In order to achieve this function, the debugger must first set some configuration registers to change the characteristics of the low-power mode.

• In sleep mode, the debugger must first set the DBG\_SLEEP bit in the DBGMCU\_CR register. This will be

HCLK provides the same clock as FCLK (system clock configured by code).

• In stop mode, the debugger must first set the DBG\_STOP bit. This will activate the internal oscillator in shutdown mode

Provide clocks for FCLK and HCLK.

**23.5.2**

When a breakpoint occurs, it is necessary to select the working mode of the counter according to the different uses of the timer and watchdog:

• When a breakpoint occurs, the counter continues to count. This is often used when outputting a PWM to control a motor.

• When a breakpoint occurs, the counter stops counting. This is necessary for the watchdog counter.

**23.5.3**

**MCU**

This register allows the MCU to be configured in the debug state. include:

• Support low power consumption mode

• Support timer and watchdog counter

• Assign tracking pins

The DBGMCU\_CR register is mapped to the external APB bus, the base address is 0x40013404. Register by PORE-

SET Asynchronous reset (not reset by system reset). When the core is in the reset state, the debugger can write to this register.

If the debugger does not support these features, user software can still write to these registers.

**23.6 DBG**

73. DBG

**Offset**

**Acronym**

**Register Name**

**Reset**

**Section**

0x00

DBG

DBG control register

0x00000000

[Section 23.6.1](https://translate.googleusercontent.com/translate_f#474)

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

452/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 475** |

UM\_MM32F003\_q\_Ver1.19

(DBG)

**23.6.1 DBG**

**(DBG\_CR)**

Address: 0x40013404 only supports 32-bit access

POR reset: 0x0000 0000 (not reset by system reset)

15

14

13

12

11

10

9

8

7

6

5

4

3

2

1

0

31

30

29

28

27

26

25

twenty four

twenty three

twenty two

twenty one

20

19

18

17

16

w

w

w

w

w

w

w

w

w

Reserved

Reserved

DBG\_TIMx\_STOP

DBG\_

WWDG

\_STOP

DBG\_

IWDG

\_STOP

Reserved

DBG\_

STAN

DBY

DBG\_

STOP

DBG\_

SLEEP

**Bit**

**Field**

**Type**

**Reset**

**Description**

31:14

Reserved

Always read as 0.

13:10

DBG\_TIMx\_

STOP

w

0x00

The counter stops working when the kernel enters the debug state x = 4..1 (TIMx

counter stopped when core is halted)

0: The counter of the selected timer still works normally

1: The counter of the selected timer stops working

9

DBG\_WWDG\_

STOP

w

0x00

When the kernel enters the debugging state, the debug window watchdog stops working (Debug

window watchdog stopped when core is halted)

0: Window watchdog counter is still working

1: The window watchdog counter stops working

8

DBG\_IWDG\_

STOP

w

0x00

When the kernel enters the debugging state, the watchdog stops working (Debug indepen-

dent watchdog stopped when core is halted)

0: Watchdog counter is still working

1: The watchdog counter stops working

7:3

Reserved

Always read as 0.

2

DBG\_STAN

DBY

w

0x00

Debug Standby mode

0: (FCLK off, HCLK off) The entire digital circuit is powered off. Soft

From the point of view of the software, exiting STANDBY mode is the same as resetting (except

Some status bits indicate that the microcontroller has just exited from the STANDBY state)

1: (FCLK on, HCLK on) the digital circuit part is not powered off, FCLK and

The HCLK clock is clocked by the internal RL oscillator. In addition, the microcontroller

Exiting STANDBY mode by generating a system reset is the same as resetting.

1

DBG\_STOP

w

0x00

Debug Stop mode

0

DBG\_SLEEP

w

0x00

Debug sleep mode

0: (FCLK on, HCLK off) In sleep mode, FCLK is

The configured system clock is provided, and HCLK is turned off. Since sleep mode does not

Reset the configured clock system, so when exiting from sleep mode, the software does not

The clock system needs to be reconfigured.

1: (FCLK on, HCLK on) In sleep mode, FCLK and HCLK

The clocks are provided by the previously configured system clock.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

453/ [454](https://translate.googleusercontent.com/translate_f#476)

|  |
| --- |
| **Page 476** |

UM\_MM32F003\_q\_Ver1.19

twenty four

Modify record

74.

period

version

content

2019/07/23

Rev1.19

SPI\_EXTCTL is only valid when DW8\_32 bit is '0'

2019/07/16

Rev1.18

The minimum value of the UART BRR register is 4

2019/07/03

Rev1.17

Modify the capture/compare register 5 description error in the advanced timer,

Change CCMR5 to CCMR3

2019/05/09

Rev1.16

Modify the port mode configuration

The brake and dead zone register BDTR in the advanced timer

Generator setting bit DTG correction

2019/04/16

Rev1.15

Modify adc calculation formula

2019/04/04

Rev1.14

Modify ADC

2019/01/10

Rev1.13

Modify PWR description

2018/12/22

Rev1.12

TIM14 modify the picture 565511, delete some redundant content.

TIMX\_16bit Function description part of the input capture section will be

"CC1S = 01 in the TIMx\_CCR1 register" was modified to

"CC1S = 01 in the TIMx\_CCMR1 register".

TIM1/8 replaces "capture" in the text with "capture".

TIM1/8

The function description part of the input capture section will be

"CC1S = 01 in the TIMx\_CCR1 register" modified

It is "CC1S=01 in the TIMx\_CCMR1 register".

TIM1/8 TIMx timer and externally triggered synchronization part will be wrong

The writing "IMx\_CR1" was changed to "TIMx\_CR1".

DIV\_Mantiss cannot be 0 in UART\_BRR.

2018/12/20

Rev1.11

UART\_CSR is changed to UART current status register.

[www.mm32mcu.com](https://translate.google.com/translate?hl=en&prev=_t&sl=auto&tl=en&u=http://www.mindmotion.com.cn/)

454/ [454](https://translate.googleusercontent.com/translate_f#476)