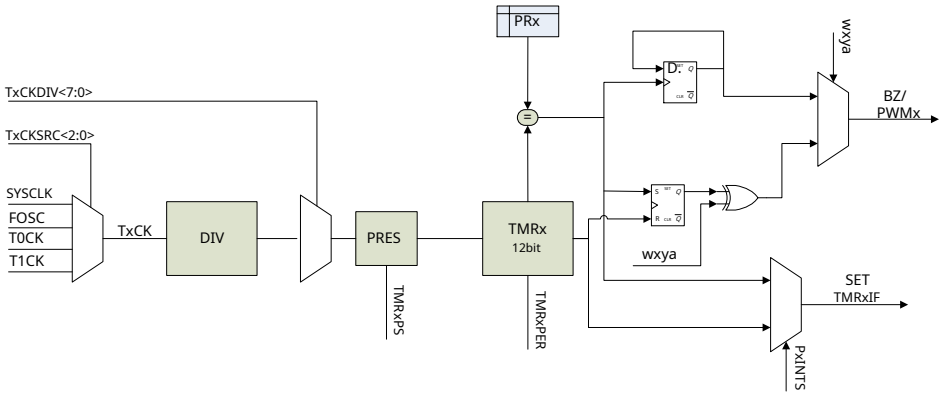
FT61F02X

PWM345 Application note

Table of contents

FT61F02x PWM345application

1. PWMxandTIMERx



picture1-1PWM3/4/5Structure diagram

Apart fromECCPprovided by the module1Road EnhancedPWMBesides, On-chip also integrates3Road time base and duty cycle are independent of each other12bit PWM, They all have the following properties:

- increment counter
- BUZZER/PWMoutput optional BUZZER/
- PWMOutput polarity selectable 8kind
- PWMresolution
- 4Two kinds of clock sources are optional
- 1~256frequency division optional
- 7bit prescaler
- Overflow interrupt or match interrupt
- Clock Modulation Output

TIMER3/4/5are all12bit-up counter that can be registered via theTMRxH:TMRxL (xcan be3/4/5)To access the count value, the software TMRxH:TMRxLA write operation will directly update the count value. whenPxEN (PWMxCR1.7)for0, work in timer mode. When the internal clock is selected as the counting source, it is a timer, and when the external clock is selected as the counting source, it is a counter.

Timer3/4/5overflow cycle

TIMERxThe maximum number of digits is12bit, by pairingPxPER[2:0]The configuration can choose different overflow period.

- existBUZZERmode, when the count value andPRWhen the registers are equal, another count clockTIMERxwill automatically clear0;
- existTIMERxoverwrite if already enabledPxPERThe value ofTMRxIFplace1, it is recommended to configure thePxPERturned on afterTMRxON.

whenTIMERxcount valueTMRxH:TMRxLIncrements to the maximum count value (byPxPERspecified), at this timeTIMERxAn overflow occurs and sets the interrupt flag (TMRxIF),andTMRxH:TMRxLregister will reset on the next increment cycle to0x00. Triggering an interrupt and/or waking up from sleep after an overflow depends on the corresponding enable/shutdown control bit (GIE, PEIEandTMRxIE). Exit on interrupt service

flagTMRxIFclear0, so as not to loop into a break.

To wake from sleep, configure to use an external clockT0CK/T1CK (PxCKSRC=010or011),otherwiseTimer1will stop Count, maintaining the count value it had before going to sleep.

Timer3/4/5Summary of related registers

name	state		register	address	reset value
PxPER	PWMxcycle selection 000 =4bit 100 = 9bit 001 = 5bit 101 = 10bit 010 = 6bit 110 = 11bit 011 = 8bit 111 = 12bit		PWMxCR0 [6:4] x = 3,4,5	0x10F/ 0x115/ 0x11B	RW−000
wxya	TIMERx/PWMxclock selection [1] 000 =system clock/(TxCKDIV+1) 001 = HIRC/(TxCKDIV +1) 010 = T0CK/(TxCKDIV +1) 011 = T1CK/ (TxCKDIV +1) 100 = HIRC/(TxCKDIV +1), PWM3output low level 101 = HIRC/(TxCKDIV +1), PWM3output high level 110 = HIRC/(TxCKDIV +1), PWM3According to the high pulse modulationPxCK 111 = HIRC/(TxCKDIV +1), PWM3According to low pulse modulationwxya		PWMxCR0 [3:1] x = 3,4,5		RW−000
wxya	output selection	1 = BUZZERoutput 0 = PWMoutput	PWMxCR0[0] x = 3,4,5		RW−0
wxya	TIMERx/PWMxOperating mode	1 = PWM/BUZZERmodel 0 =timer mode	PWMxCR1[7] x = 3,4,5	0x110/ 0x116/ 0x11C	RW−0
wxya	PWMxoutput polarity	1 =active low 0 =active high	PWMxCR1[6] x = 3,4,5		RW−0
TMRxPS	PWMxPrescaler [2] 000 = 1:1 100 = 1:16 001 = 1:2 101 = 1:32 010 = 1:4 110 = 1:64 011 = 1:8 111 = 1:128		PWMxCR1 [5:3] x = 3,4,5		RW−000
TMRxON	TIMERxenable bit	1 =Enable 0 =closure	PWMxCR1[2] x = 3,4,5		RW−0
T3CKDIV	TMR3The clock frequency isf_{T3CK}/ (T3CKDIV + 1)[3]		T3CKDIV[7:0]	0x111	RW−0000 0000
T4CKDIV	TMR4The clock frequency isf_{T4CK}/ (T4CKDIV + 1)[3]		T4CKDIV[7:0]	0x117	RW−0000 0000

[1]Note: Select as internalRCfast clock (HIRC)When the system clock selects slow clock, external clock or crystal clock, the internalHIRCwill be on unless in sleep mode

[2]Prescaler CounterTMRxPSCan not read and write directly, when theTMRxH/xLWhen the register is written, the prescaler counter is automatically cleared0.

[3]Note: When rightTxCKDIVWhen the register is written, the divisor divider will be automatically cleared0.

name	state	register	address	reset value
T5CKDIV	TMR5The clock frequency isf_{T5CK}/ (T5CKDIV + 1)3	T5CKDIV[7:0]	0x11D	RW−0000 0000
TMR3L	TMR3Count result register low8bit	TMR3L[7:0]	0x10C	RW−xxxx xxxx
TMR3H	TMR3Count result register high4bit	TMR3H[7:4]	0x10D	RW−xxxx
PR3H	PR3Period Register High4bit	TMR3H[3:0]		RW−1111
PR3L	PR3Period Register Low8bit	PR3L[7:0]	0x10E	RW−1111 1111
TMR4L	TMR4Count result register low8bit	TMR4L[7:0]	0x112	RW−xxxx xxxx
TMR4H	TMR4Count result register high4bit	TMR4H[7:4]	0x113	RW−xxxx
PR4H	PR4Period Register High4bit	TMR4H[3:0]		RW−1111
PR4L	PR4Period Register Low8bit	PR4L[7:0]	0x114	RW−1111 1111
TMR5L	TMR5Count result register low8bit	TMR5L[7:0]	0x118	RW−xxxx xxxx
TMR5H	TMR5Count result register high4bit	TMR5H[7:4]	0x119	RW−xxxx
PR5H	PR5Period Register High4bit	TMR5H[3:0]		RW−1111
PR5L	PR5Period Register Low8bit	PR5L[7:0]	0x11A	RW−1111 1111

surface1-1 Timer3/4/5Related User Control Registers

name	state		register	address	reset value
GIE	1 =Enable (PEIE, TMR2IEBe applicable)	global interrupt 0 =global shutdown (wake up is not affected)	INTCON[7]	0x0B 0x8B 0x10B	RW−0
PEIE	Total Peripheral Interrupt	1 =Enable (TMR2IEBe applicable) 0 =closure (no wakeup)	INTCON[6]		RW−0
P3INTS	Timer3interrupt select bit	1 = TMR3andPR3match break 0 = TMR3overflow interrupt	PWM3CR0[7]	0x10F	RW−0
TMR3IE	Timer3interrupt enable bit	1 =Enable 0 =closure (no wakeup)	PWM3CR1[1]	0x110	RW−0
TMR3IF	Timer3interrupt flag	1 =match/overflow (latch) 0 =no match/no overflow	PWM3CR1[0]		RW−0
P4INTS	Timer4interrupt select bit	1 = TMR4andPR4match break 0 = TMR4overflow interrupt	PWM4CR0[7]	0x115	RW−0
TMR4IE	Timer4interrupt enable bit	1 =Enable 0 =closure (no wakeup)	PWM4CR1[1]	0x116	RW−0
TMR4IF	Timer4interrupt flag	1 =match/overflow (latch) 0 =no match/no overflow	PWM4CR1[0]		RW−0
P5INTS	Timer5interrupt select bit	1 = TMR5andPR5match break 0 = TMR5overflow interrupt	PWM5CR0[7]	0x11B	RW−0
TMR5IE	Timer5interrupt enable bit	1 =Enable 0 =closure (no wakeup)	PWM5CR1[1]	0x11C	RW−0
TMR5IF	Timer5interrupt flag	1 =match/overflow (latch) 0 =no match/no overflow	PWM5CR1[0]		RW−0

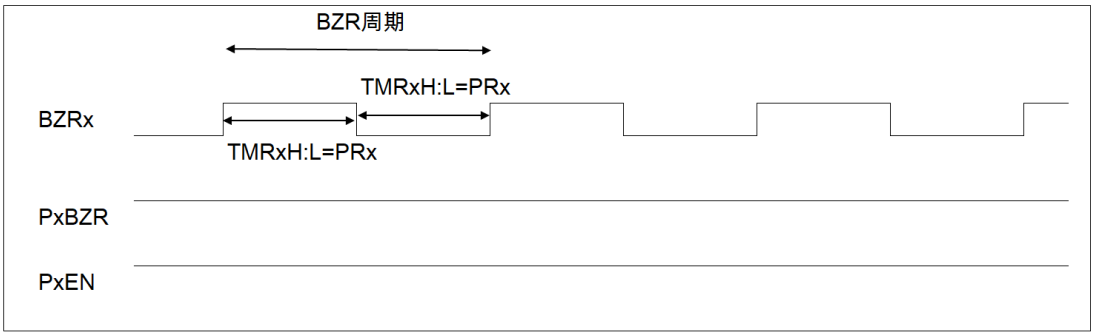
surface1-2 Timer3/4/5Interrupt Enable and Status Bits

1.1. TMRxH/Lreading and writing

TIMERxis an asynchronous clock, which is read by software in the running state12bitThe count value may be read when the low8bitafter,TIMERxAn overflow occurs and the count value is reset. At this time, if it is read high again4bitthen0.

For write operations, it is recommended to firstTIMERxstop (TMRxON=0), and then write the target value intoTMRxH/TMRxL.

1.2. BUZZEROperating mode



picture1-2 50%duty cycleBUZZERsquare wave

BUZZERcycle

formula1.1 $T_{BUZ} = 2 * 2^{TMRxPS} * PRx * T_{wxya}$

Notice:

1.work atBUZZERmode,TIMERxautomatically work on12bitmode, withPxPERThe value of

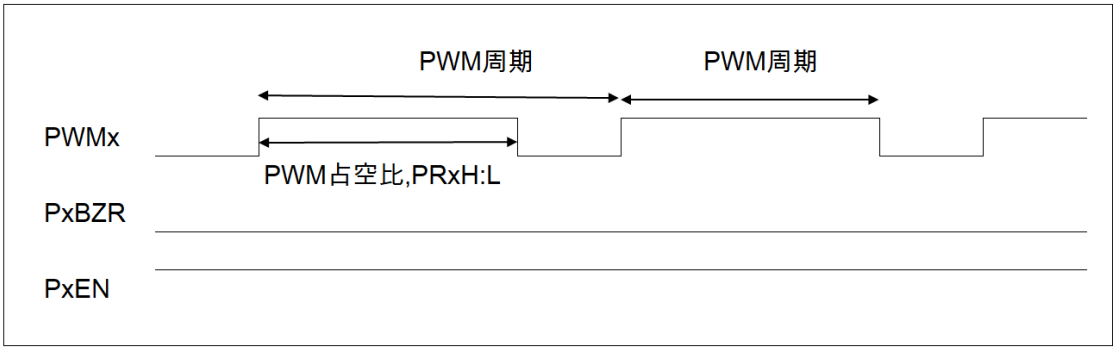
2.whenTMRxH:LequalPRxhour,TMRxH:Lwill automatically clear0

3. BUZZERmode:

- ifPRx=0x000,butwxyapin fixed output0 12bitof
- TIMERxwhen overflowTMRxIFwill be set1

4.sameTIMERThe same mode, through the configuration (wxya,wxya,TMRxON,TMRxIE,PEIE)and"PxCKSRC = 010or 011" ,BUZZERCan work in sleep mode

1.3. PWMOperating mode



picture1-3 PWMxWorking mode (forward output)

Notice:

If you want to use bothECCPofPWM (P1Apin outputPWMmodulation) andPWM3/4/5, the registerCCP1CONof P1M<1:0>to be configured as00 (Single output mode =P1Amodulation,P1B/P1C/P1Dfor the port). which is,ECCPpriority ratio PWM3/ PWM4/PWM5high.

PWMxcycle

PWMxcycle byTIMERxThe prescaler setting register of theTMRxPSandPxPERdecided together.

formula1.2 $PWMperiod = 2^{TMRxPS} * 2^{Nbit} * T_{wxya}$

- NbitforPxPERThe number of timer digits set.

PWMxduty cycle

PWMThe duty cycle is set by the registerPRxH:LDecide.PRxH:Ltotal12bit, the software needs to be written twice separately. software writingPRxH:LThe value of will take effect immediately and will directly affect the currentPWMxduty cycle is recommended at start-upTIMERxBefore writing the target value intoPRx.

formula1.3 $T_{pwm} = 2^{TMRxPS} * (PRx) * T_{wxya}$

PWMxconfiguration

1.will be relatedTRISLocation1,prohibitPWMxpin output driver

2.WritePWMxCR0register value, setPWMxPeriod, interrupt generation method and selection of clock source

3.WritePWMxCR1Register value, configured asPWMmode, and the prescaler

4.loadPRxregister, setPWMduty cycle

5.configure and startTIMERx:

- WillPWMxCR1registerTMRXIFInterrupt flag bit cleared
- WillPWMxCR1registerTMRxONLocation1start upTIMERx

6.start overPWMcycle after enablingPWMoutput:

- waitTIMERxoverflow (PWMxCR1registerTMRxIFLocation1)
- will be relatedTRISbit clear enablePWMxpin output driver

7.sameTIMERThe same mode, through the configuration (wxya,wxya,TMRxON,TMRxIE,PEIE)and"PxCKSRC = 010or 011" ,PWMCan work in sleep mode

Note: ifPWMThe clock is set to the system clock (PxCKSRC=000), then any change in the system clock frequency will result inPWMFrequency changes.

2.Application example

```
/*file name:TEST_61F02x_PWM345.C
* Features:     FT61F02x-EnhancedPWMDemo
*IC:            FT61F023 SOP16
* Crystal:      16M/2T
* illustrate:   This program is used to demonstratePWM345Features
*               In the demo program inPWM345output2kHz, 1kHz, 500Hz The
*               duty cycle is50%signal of
*
*                FT61F023 SOP16
*                 ------------------------
* VDD--------|1(VDD) (VSS)16|--------GND
* NC----------|2(PA7)      (PA0)15|---------NC
* NC----------|3(PA6)      (PA1)14|---------NC
* NC----------|4(PA5)      (PA2)13|---------NC
* PWM4-----|5(PC3)      (PA3)12|---------NC
* PWM5-----|6(PC2)      (PC0)11|---------NC
* NC----------|7(PA4)      (PC1)10|---------NC
* NC----------|8(PC5)       (PC4)09|----PWM3
*                 ------------------------
*/
# include "SYSCFG.h"
//********************Macro definition ***************************
//PWMPin input and output control
#define   PWM3Dir      TRISC4
#define   PWM4Dir      TRISC3
#define   PWM5Dir      TRISC2
/*-------------------------------------------------- -
* Function name:interrupt
* Features:   interrupt handling
* enter:    none
* output:   none
  ----------------------------------------------------
----- * / void interrupt ISR(void)
{
    NOP();
}
/*-------------------------------------------------- -
* Function name:POWER_INITIAL
* Features:   Power-on system initialization
* enter:    none
* output:   none
  ----------------------------------------------------
--------- */ void POWER_INITIAL (void)
```

```
{
    OSCCON = 0B01110001;        //IRCF=111=16MHz/2=8MHz,0.125μs//Temporarily
    INTCON = 0;                 disable all interrupts
    PORTA = 0B00000000;
    TRISA = 0B00000000;         //PAinput Output1-enter0-output
    PORTC = 0B00000000;
    TRISC = 0B00000000;         //PCinput Output1-enter0-output

    WPUA = 0B00000000;          //ban allPApull up
    WPUC = 0B00000000;          //ban allPCpull up
    OPTION = 0B00001000;        //Bit3=1, WDT MODE, PS=000=WDT RATE 1:1
    MSCKCON = 0B00000000;
    //Bit6->0,prohibitPA4,PC5Regulated output
    //Bit5->0,TIMER2the clock isFosc //Bit4->0,
    prohibitLVR

    CMCON0 = 0B00000111;        //turn off the comparator,Cxfor numbersIOmouth
}
/*---------------------------------------------- -
* Function name:PWM_INITIAL
* Features:PWM3,4,5initialization
* set upPWM3period =2 TMRXPS*2 PXPER*[(T3CKDIV+1)/PWMclock source]
*                  =2 0*2 8*[(30+1)/16000000]=496μs
*      PWM4period =2 TMRXPS*2 PXPER*[(T4CKDIV+1)/PWMclock source]
*                  =2 0*2 8*[(62+1)/16000000]=1.008ms
*      PWM5period =2 TMRXPS*2 PXPER*[(T5CKDIV+1)/PWMclock source]
*                  =2 0*2 8*[(124+1)/16000000]=2ms
  ----------------------------------------------------
----- */ void PWM_INITIAL (void)
{
    PWM3Dir = 1;                //PWM3outputPINTemporarily in input mode //
    PWM4Dir = 1;                PWM4outputPINTemporarily in input mode //
    PWM5Dir = 1;                PWM5outputPINTemporarily in input mode
    //-------------PWM3----------------
    PWM3CR0 = 0B00110010;
    //Bit7:       disable interrupt
    //Bit[6:4]:Period bit selection011-8bit
    //Bit[3:1]:clock selection001-internalRCfast clock/(
    T3CKDIV+1) //Bit0: PWMoutput

    PWM3CR1 = 0B10000000;
    //Bit7:     1-TMR3forPWM/BUZZERmodel
    //Bit6:     0-PWM3active high
    //Bit[5:3]:    000-PWM3The prescaler is set to1:1
```

```
    //Bit2:       0-temporarily closedTMR3
    //Bit1:       0-prohibitTMR3to interrupt
    //Bit0:       0-TMR3Interrupt flag bit read only

    TMR3H=0;
    T3CKDIV = 30;                   //assignmentT3CKDIV
    PR3L = 128;
    //-------------PWM4----------------
    PWM4CR0 = 0B00110010;
    //Bit7:       disable interrupt
    //Bit[6:4]:   Period bit selection011-8bit
    //Bit[3:1]:   clock selection001-internalRCfast clock/(T4CKDIV+1)
    //Bit0:       PWMoutput

    PWM4CR1 = 0B10000000;
    //Bit7:       1-TMR4forPWM/BUZZERmodel
    //Bit6:       0-PWM4active high
    //Bit[5:3]:   000-PWM4The prescaler is set to1:1
    //Bit2:       0-temporarily closedTMR4
    //Bit1:       0-prohibitTMR4to interrupt
    //Bit0:       0-TMR4Interrupt flag bit read only

    TMR4H=0;
    T4CKDIV = 62;                   //assignmentT4CKDIV
    PR4L = 128;
    //------------PWM5--------------
    PWM5CR0 = 0B00110010;
    //Bit7:       disable interrupt
    //Bit[6:4]:Period bit selection011-8bit
    //Bit[3:1]:clock selection001-internalRCfast clock/(
    T5CKDIV+1) //Bit0: PWMoutput

    PWM5CR1 = 0B10000000;
    //Bit7:       1-TMR5forPWM/BUZZERmodel
    //Bit6:       0-PWM5active high
    //Bit[5:3]:   000-PWM5The prescaler is set to1:1
    //Bit2:       0-temporarily closedTMR5
    //Bit1:       0-prohibitTMR5to interrupt
    //Bit0:       0-TMR5Interrupt flag bit read only

    TMR5H=0;
    T5CKDIV = 124;                  //assignmentT5CKDIV
    PR5L = 128;
}
```

```
/*---------------------------------------------- -
* Function name:main
* Features:  main function
* enter:  none
* output:  none
  ----------------------------------------------------
----- */ void main(void)
{
    POWER_INITIAL();
    PWM_INITIAL();

    TMR3ON=1;
    TMR4ON=1;
    TMR5ON=1;
    PWM3Dir = 0;          //PWM3 PINSet to output mode to allowPWMoutput //
    PWM4Dir = 0;          PWM4 PINSet to output mode to allowPWMoutput //
    PWM5Dir = 0;          PWM5 PINSet to output mode to allowPWMoutput
    while(1)
    {
        NOP();
    }
}
```

Contact information

Fremont Micro Devices Corporation

5−8, 10/F, Changhong Building Ke-Ji
Nan 12 Road, Nanshan District,
Shenzhen, Guangdong, PRC 518057

Tel: (+86 755) 8611 7811
Fax: (+86 755) 8611 7810

Fremont Micro Devices (HK) Limited

16, 16/F, Block B, Veristong Industrial Centre, 34−36 Au Pui
Wan Street, Fotan, Shatin, Hong Kong SAR

Tel: (+852) 2781 1186
Fax: (+852) 2781 1144

http://www.fremontmicro.com