

# 您設計產品時的好朋友！



Forum: [16-bit PIC24/dsPIC](#)

Topic: dsPIC30F4011之ADC中斷內算式對中斷時間之影響

Subject: dsPIC30F4011之ADC中斷內算式對中斷時間之影響

作者: n26051144

2018年01月10日 17:25:04

您好,我ADC中斷是以MCPWM裡設定的period作為觸發時脈(60KHz),在ADC中斷內變數ADC用來取ADCBUF的值,變數ADC1用來對ADC作計算, test則是定義成數位輸出用來測量中斷頻率,目前遇到的問題是當中斷內加上ADC1運算式時,會導致中斷頻率減少一半,只要將其反白就恢復正常,ADC1只是普通的加減乘除運算為何會讓中斷頻率減半?煩請各位先進指教指教。

以下是我的程式碼:

```
#include "C30EVM_LCD.h" // 將LCD函式的原型宣告檔案含入
#include <p30F4011.h>
#include <timer.h> // 將Timer函式的原型宣告檔案含入
#include <adc10.h> // 將adc10函式的原型宣告檔案含入
#include <pwm.h> // 將pwm函式的原型宣告檔案含入
#include <uart.h> // 將UART函式的原型宣告檔案含入
#include <outcompare.h> // 將Compare函式的原型宣告檔案含入
#include <math.h> // 將Math函式的原型宣告檔案含入
#include <spi.h> // 納入SPI函式庫函式原型名稱定義
#include <incap.h> // 將Inputcapture函式的原型宣告檔案含入
#define __dsPIC30F4011__
#define FCY 7372800 * 4 // 因為使用頻率為將外部 7.3728 MHz * 8 的模式 , 每一指令週期需 4 個 clock

// 設定MCP4921相關腳位
#define TRIS4921_SDI TRISFbits.TRISF3
#define TRIS4921_SDO TRISFbits.TRISF2
#define TRIS4921_SCK TRISFbits.TRISF6
#define TRIS4921_CS TRISBbits.TRISB2
#define MCP4921_CS LATBbits.LATB2
#define test TRISBbits.TRISB3
#define test LATBbits.LATB3
//#define TRIS4921_LDAC TRISFbits.TRISFx
//#define MCP4921_LDAC LATFbits.RFx

_FOSC(CSW_FSCM_OFF & XT_PLL16); // XT with 8xPLL oscillator, Failsafe clock off
_FWDT(WDT_OFF); // Watchdog timer disabled
_FBORPOR(PBOR_OFF & MCLR_EN); // Brown-out reset disabled, MCLR reset enabled
_FGS(CODE_PROT_OFF); // Code protect disabled

void __attribute__((__interrupt__ , auto_psv)) _ADCInterrupt(void);
void Init_Port(void);
void Init_MCPWM(void);
```

```

void Init_ADC(void) ;
void Show_ADC(void) ;
void Init_SPI(void);
void WriteSPI(unsigned int data_out);

unsigned int chanA=0x3000, spi_data, DAC_A;

signed int qq;
float ADC,ADC1,ADC2;
signed int a;

float valpha, vbeta, theta;

signed int valpha1, vbeta1;

double yat=0,yat1=0,yat2=0;
double xat=0,xat1=0,xat2=0;

double ybt=0,ybt1=0,ybt2=0;
double xbt=0,xbt1=0,xbt2=0;

const char My_String1[]="Ex 15 - MCPWM" ; // 宣告字串於 Program Memory (因為 const 宣告)
char My_String2[]="VR1: VR2: " ; // 宣告字串於 Data Memory

int main(void)
{

Init_ADC( ) ; // 將ADC進行初始化設定

OpenLCD( ) ; // 使用 OpenLCD( )對 LCD 模組作初始化設定

Init_SPI();
Init_Port(); // 4 bits Data mode
// 5 * 7 Character

Init_MCPWM( ) ;

while(1)
{
}
}

void Init_MCPWM(void)
{
/* Holds the PWM interrupt configuration value*/
unsigned int config;
/* Holds the value to be loaded into dutycycle register */
unsigned int period;
/* Holds the value to be loaded into special event compare register */

```

```

unsigned int sptime;
/* Holds PWM configuration value */
unsigned int config1;
/* Holds the value be loaded into PWMCON1 register */
unsigned int config2;
/* Holds the value to configure the special event trigger postscale and dutycycle */
unsigned int config3;
/* The value of 'dutycyclereg' determines the duty cycle register(PDCx) to be written
*/
unsigned int dutycyclereg;
unsigned int dutycycle;
unsigned char updatedisable;
/* Configure pwm interrupt enable/disable and set interrupt priorities */
config = (PWM_INT_DIS & PWM_FLTA_DIS_INT & PWM_INT_PR1
          & PWM_FLTA_INT_PRO);
ConfigIntMCPWM( config );
/* Configure PWM to generate square wave of 50% duty cycle */
//dutycyclereg = 1;
//dutycycle = 0x7FFF;
//updatedisable = 0;
//SetDCMCPWM(dutycyclereg,dutycycle,updatedisable);
//PDC1=0x0;
//PDC2=0x0;
//PDC3=0x0;
period = 491.52;
sptime = 0x00;
config1 = (PWM_EN & PWM_IDLE_STOP & PWM_OP_SCALE1
          & PWM_IPCLK_SCALE1 & PWM_MOD_FREE);
config2 = (PWM_MOD1_COMP & PWM_MOD2_COMP & PWM_MOD3_COMP & //PWM_MOD2_IND
//PWM1,2,3啟動互補
          PWM_PEN3H & PWM_PEN2H & PWM_PEN1H &
          PWM_PEN3L & PWM_PEN2L & PWM_PEN1L);
config3 = (PWM_SEVOPS1 & PWM_OSYNC_PWM & PWM_UEN);
OpenMCPWM(period,sptime,config1,config2,config3);
DTCON1=0x0508; //deadtime control
}

void Init_ADC(void)
{

unsigned int Channel, PinConfig, Scanselct, Adcon3_reg, Adcon2_reg, Adcon1_reg;
//TRISB = 0xFFFF; //RB0~RB8作數位輸入用,給類比訊號輸入
ADCON1bits.ADON = 0; /* turn off ADC */

PinConfig = ENABLE_ANO_ANA; // Select port pins as analog inputs ADPCFG<15:0>

Adcon1_reg = ADC_MODULE_ON & // Turn on A/D module (ADON)
ADC_IDLE_STOP & // ADC turned off during idle (ADSIDL)
ADC_FORMAT_INTG & // Output in integer format (FORM)

```

```

ADC_CLK_MPWM &          // Conversion trigger by MCPWM (SSRC)
ADC_SAMPLE_INDIVIDUAL & // Sample channels individually (SIMSAM)
ADC_AUTO_SAMPLING_ON;   // Sample trigger automatically (ASAM)

Adcon2_reg = ADC_VREF_AVDD_AVSS & // Voltage reference : +AVdd, -AVss (VCFG)
ADC_SCAN_OFF &           // Scan off (CSCNA)
ADC_ALT_BUF_OFF &       // Use fixed buffer (BUFM)
ADC_ALT_INPUT_OFF &     // Does not alternate between MUX A & MUX B (ALTS)
ADC_CONVERT_CHO &      // Convert only channel 0 (CHPS)
ADC_SAMPLES_PER_INT_1;  // 1 sample between interrupt (SMPI)

Adcon3_reg = ADC_SAMPLE_TIME_10 & // Auto-Sample time (SAMC)
ADC_CONV_CLK_SYSTEM &           // Use system clock (ADRC)
ADC_CONV_CLK_4Tcy;             // Conversion clock = 4 Tcy (ADCS)
                                // ADCS = 2*(154ns)/(1/Fcy)-1 = 3.5416

Scanselct = SCAN_NONE;         // ADC scan no channel (ADCSSL)

OpenADC10(Adcon1_reg, Adcon2_reg, Adcon3_reg, PinConfig, Scanselct);

Channel = ADC_CHO_POS_SAMPLEA_ANO & // CHO Pos. : ANO, Neg. : Nominal Vref- Defined in
ADCON2
        ADC_CHO_NEG_SAMPLEA_NVREF ;// (ADCHS)
SetChanADC10(Channel);
//ConfigIntADC10(ADC_INT_DISABLE);
IFS0bits.ADIF = 0;           //清除AD中斷旗標
IEC0bits.ADIE = 1;         //致能AD中斷旗標
ADC0N1bits.ADON = 1;       //開啟ADC

}

void __attribute__((__interrupt__, auto_psv)) _ADCInterrupt(void)
{
    test =! test;
    int i;
    a=a+1;
    IFS0bits.ADIF = 0;
    ADC = (ADCBUF0);        // get ADC value
    ADC1 = ((ADC-430)/430)*155;
    if(a==3)
    {
        PDC1 = ADC;
        a=0;
    }
}

void Init_Port(void)
{

```

```
ADPCFG = 0xfffe;
TRISB = TRISB & 0xfff3;
//TRIS4921_LDAC=0; // LDAC 腳位設為輸出
//test=0;
TRIS4921_SDI=1; // SDI 腳位設為輸入
TRIS4921_SDO=0; // SDO 腳位設為輸出
TRIS4921_SCK=0; // SCK 腳位設為輸出
ADPCFGbits.PCFG2=1; //設定RB2為數位輸出入腳位
TRIS4921_CS=0; // Chip Select 腳位設為輸出
MCP4921_CS=1; // CS初始化為1
//MCP4921_LDAC=1; // LDAC初始化為1
}
```