HY17M24

HYCON IP 使用說明書

Table of Contents

注意：

1、 本說明書中的內容，隨著產品的改進，有可能不經過預告而更改。請客戶及時到本公司網站下載更新 http://www.hycontek.com。

2、 本規格書中的圖形、應用電路等，因第三方工業所有權引發的問題，本公司不承擔其責任。

3、 本產品在單獨應用的情況下，本公司保證它的性能、典型應用和功能符合說明書中的條件。當使用在客戶的產品或設備中，以上條件我們不作保證，建議客戶做充分的評估和測試。

4、 請注意輸入電壓、輸出電壓、負載電流的使用條件，使 IC 內的功耗不超過封裝的容許功耗。對於客戶在超出說明書中規定額定值使用產品，即使是瞬間的使用，由此所造成的損失，本公司不承擔任何責任。

5、 本產品雖內置防靜電保護電路，但請不要施加超過保護電路性能的過大靜電。

6、 本規格書中的產品，未經書面許可，不可使用在要求高可靠性的電路中。例如健康醫療器械、防災器械、車輛器械、車載器械及航空器械等對人體產生影響的器械或裝置，不得作為其部件使用。

7、 本公司一直致力於提高產品的品質和可靠度，但所有的半導體產品都有一定的失效概率，這些失效概率可能會導致一些人身事故、火災事故等。當設計產品時，請充分留意冗餘設計並採用安全指標，這樣可以避免事故的發生。

8、 本規格書中內容，未經本公司許可，嚴禁用於其他目的之轉載或複製。

1. 文件說明

HYCON IP(IP 矽智財 Intellectual Property 縮寫)，代表紘康 8 位元 MCU 內部各元件的(矽智財)半導體矽晶片智慧財產權。本文件針對 HY17M24，SOC 晶片內數位、類比及通訊等其它周邊 IP 做使用說明。

主要包含以下 4 大分類:

(1)數位 IP：TimerA/TimerB/WDT/PWM/GPIO

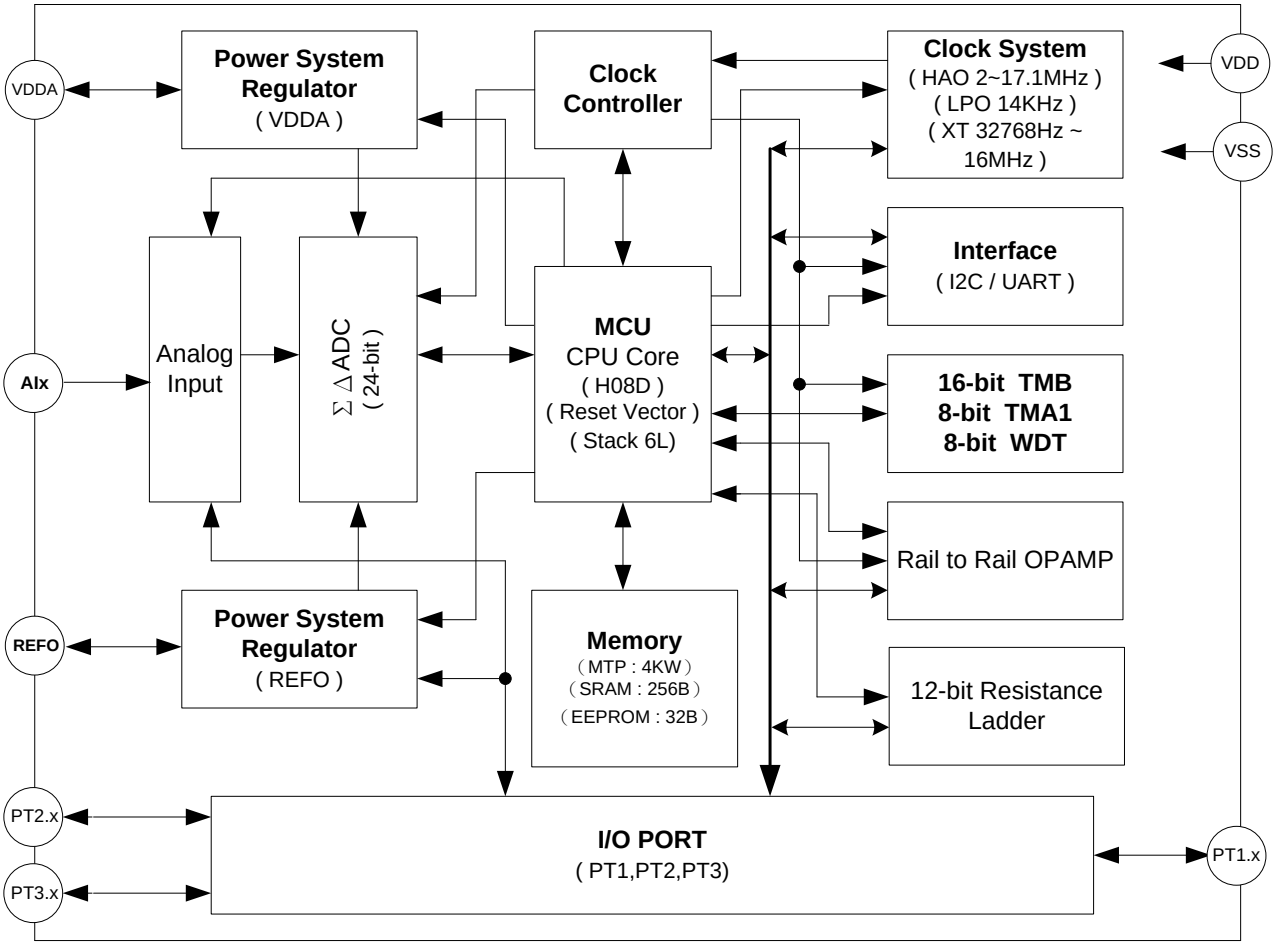
(2)類比 IP：12 bit Resistance Ladder(DAC)/ADC/OPAMP /CMP

(3)通訊 IP：Hardware UART/ Hardware I2C

(4)其他 IP：Power Management

2. 晶片說明

HY17M24 各功能 IP 基礎使用說明。



(01)採用 8-Bit RISC-Like 微控制器。

(02)電壓操作範圍 1.9V~5.5V(類比電源沒開啟情況下)，以及-40℃~85℃工作溫度範圍。

(03)支援外部 16MHz 石英震盪器或內部 16MHz 高精度 RC 震盪器，

(04) 4K words MTP Program Memory (讀寫次數：100 cycles)、32 bytes EEPROM Data Memory(讀寫次數：3,000 cycles)

(05)資料記憶體 256 Byte SRAM。

(06)擁有 BOR and WDT 功能，可防止 CPU 死機。

(07)24-bit 高精準度ΣΔADC 類比數位轉換器

(7.1) ADC Gain: x1/4, x1/2, x1,x2,x4,x8,x16.。

(7.2)內置溫度感測器 TPS。

(08)內建 1 組 OPA 運算放大器。

(09)內建硬體 12-bit Resistance Ladder(DAC)。

(10)16-bit Timer A

(11)16-bit Timer B 模組具 PWM 波形產生功能

(12)硬體串列通訊 I2C/UART 模組

3. 數位 IP(TMA)

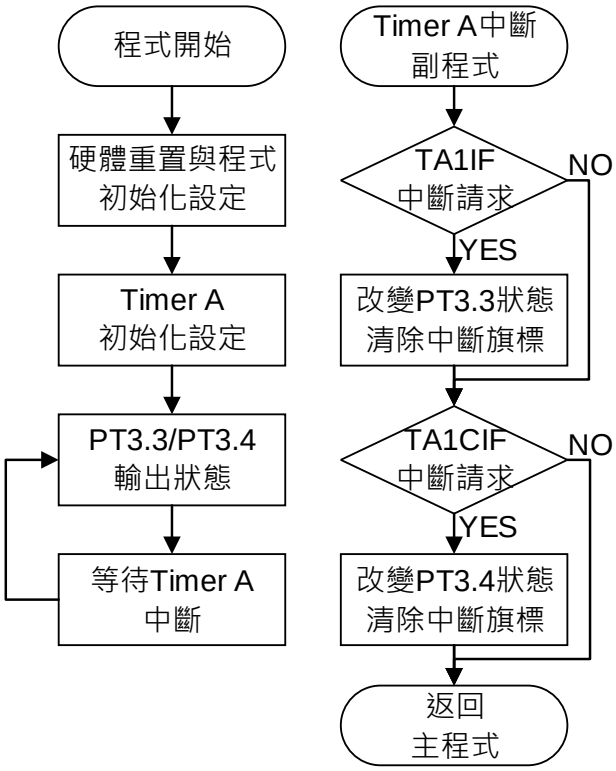
3.1. 範例名稱

TMA

3.2. 範例說明

(1) Timer A 使用方式與說明。
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 程式做好 Timer A 初始化動作與設置相關 Timer A 計數溢出值，開啟 GIE 等待 Timer A 中斷發生。Timer A 計數溢出值可決定 Timer A 進出中斷的速度。
(4) 每進一次 TA1 中斷，代表 TMA1R 加一，並改變 PT3.3 輸出狀態觀察變化速度。
(5) 每進一次 TA1C 中斷，代表 TMA1R 計數值等於 TMA1C，改變 PT3.4 輸出狀態觀察變化速度，可透過設置 TMA1C 暫存器增加 TMA 計數時間。

3.3. 軟體流程



3.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * TMA.c                                                                     *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *
 *****************************************************************************/
 /*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <GPIO.h>
#include <RST.h>
#include <TMR.h>
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char tma_flag=0,tmac_flag=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

//******PT1.0 HW Reset Function Ensable**********//
      PT10_HWResetEnable();

//***** setting CPU CLK *********//
        CLK_CPUCKOpen(HAOM_1843KHZ,OSCS_HAO,DHS_HSCKDIV1,CPUS_DHSCK);
      CLK_DMSCKSelect(DMS_DHSCKDIV2);    //DMS_CK= HAO/DHS/DMS
//***** setting GPIO *********//
        GPIO_PT3InputEnable(PT33_H);            //PT33 Digital mode
        GPIO_PT3OutputMode(PT33_H);                 //PT33 Output mode
        GPIO_PT3OutputLow(PT33_H);            //PT33 Output Low

        GPIO_PT3InputEnable(PT34_H);            //PT34 Digital mode
        GPIO_PT3OutputMode(PT34_H);                 //PT34 Output mode
        GPIO_PT3OutputLow(PT34_H);            //PT34 Output Low

//***** setting the timer A *********//
```

```
        TMA_Open(TMAS1_DMSCK,DTMA1_TMA1CKDIV2,4); //TMA SETTING
                                                  //TMA_Flag(us)=1/(((((HAO/DHS)/DMS))/256)/DTMA1)
                                                  //TMAC_Flag(us)=TMA_Flag*TMA1C
//***** Enable TMA INT *********//
        GIE_Enable();
        TA1CIF_ClearFlag();
        TA1CIE_Enable();
        ADCIE_Enable();
        TA1IF_ClearFlag();
        TA1IE_Enable();


        while(1);

}

/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
        if(TA1IF_IsFlag())          //TA1IF=1,TMA1R++
      {
                tma_flag=~tma_flag;
                if(tma_flag)
                GPIO_PT3OutputHigh(PT33_H);                 //PT33 Output High
                else
                GPIO_PT3OutputLow(PT33_H);            //PT33 Output Low
                TA1IF_ClearFlag();
      }

        if(TA1CIF_IsFlag())           //TMA1C=TMA1R ,then TA1CIF=1
      {
                tmac_flag=~tmac_flag;
                if(tmac_flag)
                GPIO_PT3OutputHigh(PT34_H);                 //PT34 Output High
                else
                GPIO_PT3OutputLow(PT34_H);            //PT34 Output Low
                TMA1_ClearTMA1();           //clear TMA1R
                TA1CIF_ClearFlag();
        }

}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

4. 數位 IP(TMB)

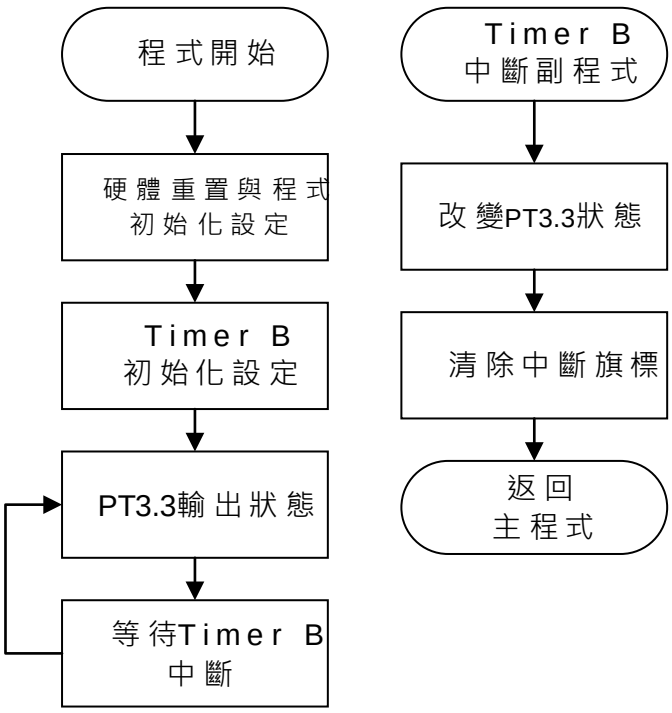
4.1. 範例名稱

TMB

4.2. 範例說明

(1) Timer B 使用方式與說明。
(2) 利用程式碼#define 來選擇要編譯執行 mode_16bit, mode_17bit, mode_2_8bit 或 mode_8_8bit
(3) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(4) 程式做好 Timer B 初始化動作與設置相關 Timer B 計數溢出值，開啟 GIE 等待 Timer B 中斷發生。Timer 計數溢出值可決定 Timer 進出中斷的速度。
(5) 每進一次 TMB 中斷，會在 TMB 中斷副程式內改變 PT3.3 輸出狀態，觀察轉換速度。

4.3. 軟體流程



4.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * TMB.c                                                                     *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *

 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <TMR.h>
#include <CLK.h>
#include <INT.h>
#include <GPIO.h>
#include <PWR.h>
#include <RST.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#define mode_16bit
//#define mode_17bit
//#define mode_2_8bit
//#define mode_8_8bit
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay_NOP(unsigned int del);
void CLOCKInit(void);
void TMBInit(void);
void GPIOInit(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char tmb_flag=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
      PT10_HWResetEnable();//PT1.0 HW Reset Function Ensable

/*******System Init************/
        CLOCKInit();
        TMBInit();
        GPIOInit();
```

```
    GIE_Enable();
    while(1);
}

/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
    if(TMBIF_IsFlag())          //TB1R=TB1C ,TMBIF=1
    {
        tmb_flag=~tmb_flag;
        if(tmb_flag)
        GPIO_PT3OutputHigh(PT33_H);
        else
        GPIO_PT3OutputLow(PT33_MSK);
        TMBIF_ClearFlag();
    }
}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
void Delay_NOP(unsigned int del)
{
  while(--del)
  NOP();
}

/*---------------------------------------------------------------------------*/
/* TMB Init Function                                                         */
/*---------------------------------------------------------------------------*/
void TMBInit(void)          //select TMB mode
{
#ifdef mode_16bit
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_16bit ,TB1RT_LogicH );
    TB1C0Set(0x00ff);       //Set TMB count
#endif


#ifdef mode_17bit
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_17bit ,TB1RT_LogicH );
    TB1C0Set(0x7fff);//Set TMB count
#endif


#ifdef mode_2_8bit
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_2_8bit ,TB1RT_LogicH );
    TB1C0Set(0x7fff);//Set TMB count
#endif


#ifdef mode_8_8bit
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_8_8bit ,TB1RT_LogicH );
    TB1C0Set(0x7fff);//Set TMB count
#endif
```

```
	TB1Enable();
}



/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{
	CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );
}

/*---------------------------------------------------------------------------*/
/* GPIO Init Function                                                        */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
	GPIO_PT3InputEnable(PT33_MSK);	//PT33 Digital mode
	GPIO_PT3OutputMode(PT33_H);		//PT33 Output mode
	GPIO_PT3OutputLow(PT33_MSK);	//PT33 Output Low
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

5. 數位 IP(WDT)

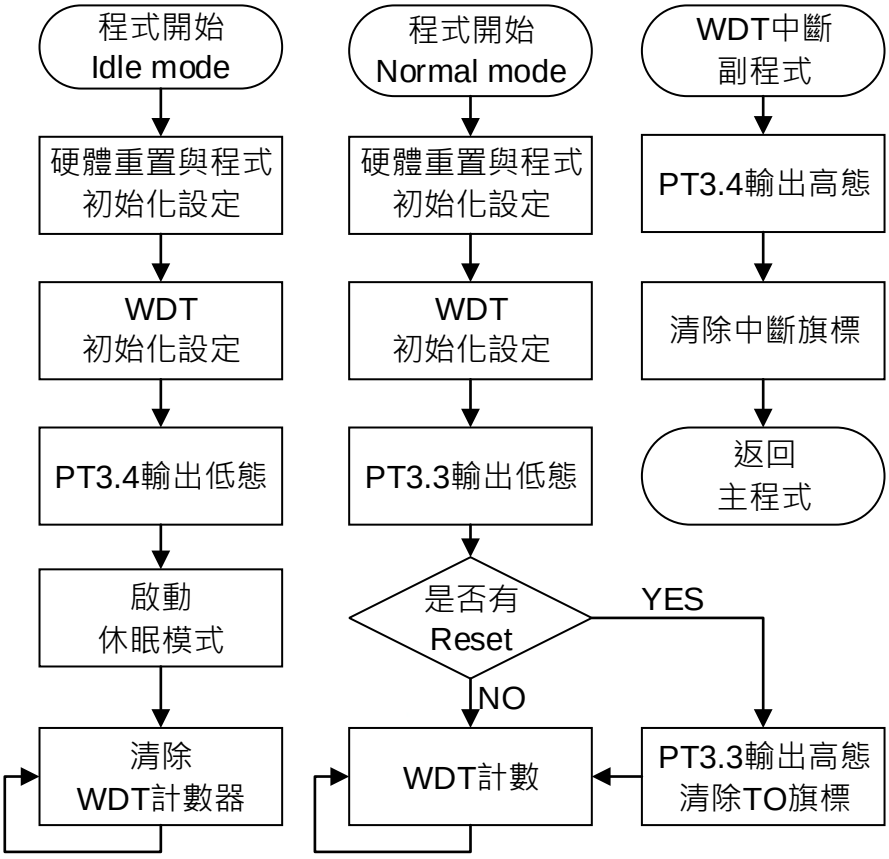
5.1. 範例名稱

WDT_MIAN

5.2. 範例說明

(1) WDT 使用方式與說明
(2) 利用程式碼#define 來選擇要編譯執行 Normal_mode 或 Idle_mode。
(3) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(4) 程式做好 WDT 初始化動作與設置 WDT 計數溢出值條件，開啟 GIE 等待 WDT 中斷發生。WDT 計數溢出值可決定 WDT 進出中斷的速度。
(5) 每進一次 WDT 中斷，會在 WDT 中斷副程式內改變 PT3.4 輸出高態，並於離開中斷前清除 WDT Count。
(6) 在 Normal_mode 下，當 WDT 計數值溢出時，系統將 Reset 並設置 TO 旗標為 1。如不要重置則必須在 WDT 計數值溢出前清除 WDT 計數值。
(7) 在 Idle_mode 下，當 WDT 計數值溢出時，系統將從 Idle 模式恢復至一般模式並進入中斷執行對應動作，於離開中斷前清除 WDT 計數值，避免系統重置。

5.3. 軟體流程


程式開始
Idle mode
硬體重置與程式
初始化設定
WDT
初始化設定
PT3.4輸出低態
啟動
休眠模式
清除
WDT計數器
程式開始
Normal mode
硬體重置與程式
初始化設定
WDT
初始化設定
PT3.3輸出低態
是否有
Reset
YES
NO
WDT計數
WDT中斷
副程式
PT3.4輸出高態
清除中斷旗標
返回
主程式
PT3.3輸出高態
清除TO旗標

5.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * WDT_MAIN.c                                                                *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *
 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <WDT.h>
#include <CLK.h>
#include <INT.h>
#include <RST.h>
#include <GPIO.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define Normal_mode    //     WDT Reset AND TO=1
#define Idle_mode    // WDT Interrupt
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

    PT10_HWResetEnable();       //PT1.0 HW Reset Function Ensable

//************Setting CPU CLK**********************//
    CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK);      //CPU CLK SETTING

//*************Setting GPIO**********************//
    GPIO_PT3InputEnable(IN33_H | IN34_H);  //PT33.PT34 Digital mode Enable
    GPIO_PT3OutputMode(TC33_H);                    //PT33 Output mode Enable
    GPIO_PT3OutputLow(PT33_MSK);                    //PT33 Output Low

    GPIO_PT3OutputMode(TC34_H);                    //PT34 Output mode Enable
    GPIO_PT3OutputLow(PT34_MSK);                        //PT34 Output Low
```

```
//************Setting WDT************************//
    WDT_Open(DWDT_WDTCKDIV8192);              //WDT Open
    WDT_Clear();   // CLEAR WDT COUNT

#ifdef Normal_mode            //At Normal mode ,if WDT overflow, system will be reset.

    if(SYS_ReadWDT())   // TO=1,mean system has reset
    {
        SYS_ClearWDT();          //CLR TO Flag
        GPIO_PT3OutputHigh(IN33_H);
    }
    while(1)
    {
        //WDT_Clear();                    // If WDT count clear,system will not reset.
        __asm__("NOP");
    }
#endif

    WDT_Clear();   // CLEAR WDT COUNT

#ifdef Idle_mode              //At Idle mode ,if WDT overflow, occurrence Interrupt.
    WDTIE_Enable();
    GIE_Enable();
    while(1)
     {
        __asm__("NOP");

          Idle();   //   IDLE mode Enable

        __asm__("NOP");

     }

#endif

}
/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt   //WDT Reset
{
      WDTIF_ClearFlag();
      SYS_ClearIDLE();       //CLR IDLE Flag
      WDT_Clear();           //CLR WDT count0.
      GPIO_PT3OutputHigh(PT34_H);
}
```

6. 數位 IP(PWM)

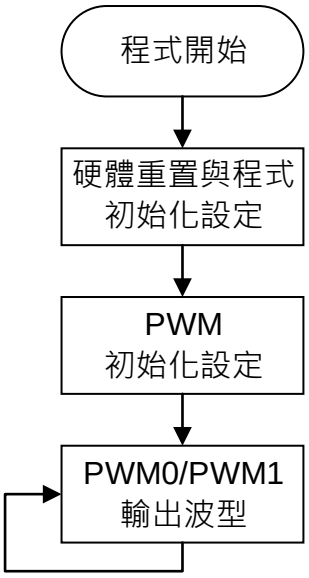
6.1. 範例名稱

PWM

6.2. 範例說明

(1) PWM 使用方式與說明。
(2) 利用程式碼#define 來選擇 PWM 模式要編譯執行 PWM1O, PWM2O, PWM3O, PWM4O, PWM5O, PWM6O 或 PWM7O。
(3) 利用程式碼#define 來規劃腳位要編譯執行 PT16_PWM0, PT13_PWM0_1, PT32_PWM0_2, PT12_PWM1, PT20_PWM1_1 或 PT33_PWM1_2。
(4) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(5) 開啟 PWM 暫存器設定與 PWM Duty 設定。
(6) 可透過上述規劃輸出腳位觀察 PWM 輸出頻率與 Duty。

6.3. 軟體流程



6.4. 程式碼

```
#define   USE_HY17M24_2M
/*******************************************************************************
 * PWM.c                                                                       *
 * --------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *    HY17M24                                                                  *
 * -----------                                                                 *
 *              |                                                              *
 *       PT1.6 |-->PWM0                                                        *
 *              |                                                              *
 *       PT1.2 |-->PWM1                                                        *
 *              |                                                              *
 * -----------                                                                 *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M24                                                            *

 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <TMR.h>
#include <CLK.h>
#include <INT.h>
#include <RST.h>
#include <GPIO.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
////     Select PWM MODE              //
#define PWM1O
//#define PWM2O
//#define PWM3O
//#define PWM4O
//#define PWM5O
//#define PWM6O
//#define PWM7O

////     Select PWM Output pin //
#define PT16_PWM0
//#define PT13_PWM0_1
//#define PT32_PWM0_2
#define PT12_PWM1
//#define PT20_PWM1_1
//#define PT33_PWM1_2
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
```

```
void Delay_NOP(unsigned int del);
void CLOCKInit(void);
void TMBInit(void);
void GPIOInit(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char tmb_flag=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

    //PT1.0 HW Reset Function Ensable
    PT10_HWResetEnable();

//***** setting cpu clk ********//
    CLK_CPUCKOpen(HAOM_1843KHZ,OSCS_HAO,DHS_HSCKDIV1,CPUS_DHSCK);

//***** setting GPIO    *********//
    GPIOInit();

//***** setting the timer B & PWM *********//
/*--------------------------------------PWM1O-----------------------------------------*/
#ifdef PWM1O
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_16bit ,TB1RT_LogicH );

    TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
    TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

    TB1_PWM0ModeSelect(PWMA0_PWM1O);           //PWM0 Output mode Select PWM1O
    TB1_PWM1ModeSelect(PWMA1_PWM1O);           //PWM1 Output mode Select PWM1O

    TB1_PWMO0(PWMO0_OUTPUT);                   //PWM0 Enable
    TB1_PWMO1(PWMO1_OUTPUT);                   //PWM1 Enable

    TB1C0Set(0x0800);    // TB1C0[15:0] is setting the PWM frequency.
    TB1C1Set(0x0400);    // TB1C1[15:0] is setting the PWM Duty Cycle.
#endif

/*--------------------------------------PWM2O-----------------------------------------*/
#ifdef PWM2O
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_16bit ,TB1RT_LogicH );

    TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
    TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

    TB1_PWM0ModeSelect(PWMA0_PWM2O);           //PWM0 Output mode Select PWM2O
    TB1_PWM1ModeSelect(PWMA1_PWM2O);           //PWM1 Output mode Select PWM2O

    TB1_PWMO0(PWMO0_OUTPUT);                   //PWM0 Enable
    TB1_PWMO1(PWMO1_OUTPUT);                   //PWM1 Enable

    TB1C0Set(0x0833);    //TB1C0[15:0] is setting the PWM frequency.
    TB1C2Set(0x01a4);    // TB1C2[15:0] is setting the PWM Duty Cycle.
#endif
```

```
/*---------------------------------------PWM3O-----------------------------------------*/
#ifdef PWM3O
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_2_8bit ,TB1RT_LogicH );

    TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
    TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

    TB1_PWM0ModeSelect(PWMA0_PWM3O);           //PWM0 Output mode Select PWM3O
    TB1_PWM1ModeSelect(PWMA1_PWM3O);           //PWM1 Output mode Select PWM3O

    TB1_PWMO0(PWMO0_OUTPUT);                   //PWM0 Enable
    TB1_PWMO1(PWMO1_OUTPUT);                   //PWM1 Enable

    TB1C0Set(0x0029);     // TB1C0L[7:0] is setting the PWM frequency.
    TB1C1Set(0x0016);     // TB1C0L[7:0] is setting the PWM Duty Cycle.
#endif

/*---------------------------------------PWM4O-----------------------------------------*/
#ifdef PWM4O
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_2_8bit ,TB1RT_LogicH );

    TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
    TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

    TB1_PWM0ModeSelect(PWMA0_PWM4O);           //PWM0 Output mode Select PWM4O
    TB1_PWM1ModeSelect(PWMA1_PWM4O);           //PWM1 Output mode Select PWM4O

    TB1_PWMO0(PWMO0_OUTPUT);                   //PWM0 Enable
    TB1_PWMO1(PWMO1_OUTPUT);                   //PWM1 Enable

    TB1C0Set(0xD100);     // TB1C0H[15:8] is setting the PWM frequency.
    TB1C2Set(0x0088);     // TB1C2L[7:0] is setting the PWM Duty Cycle.
#endif

/*---------------------------------------PWM5O-----------------------------------------*/
#ifdef PWM5O
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_8_8bit ,TB1RT_LogicH );

    TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
    TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

    TB1_PWM0ModeSelect(PWMA0_PWM5O);           //PWM0 Output mode Select PWM5O
    TB1_PWM1ModeSelect(PWMA1_PWM5O);           //PWM1 Output mode Select PWM5O

    TB1_PWMO0(PWMO0_OUTPUT);                   //PWM0 Enable
    TB1_PWMO1(PWMO1_OUTPUT);                   //PWM1 Enable

    TB1C0Set(0x00C8);     // TB1C0L[7:0] is setting the PWM frequency.
    TB1C1Set(0x0064);     // TB1C1L[7:0] is setting the PWM Duty Cycle.
    TB1C2Set(0x0080);     //       TB1C2L[7:0] is setting the PWM Duty Cycle fine tuning.
#endif

/*---------------------------------------PWM6O-----------------------------------------*/
#ifdef PWM6O
    TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_17bit ,TB1RT_LogicH );
```

```
        TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
        TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

        TB1_PWM0ModeSelect(PWMA0_PWM6O);            //PWM0 Output mode Select PWM6O
        TB1_PWM1ModeSelect(PWMA1_PWM6O);            //PWM1 Output mode Select PWM6O

        TB1_PWMO0(PWMO0_OUTPUT);                    //PWM0 Enable
        TB1_PWMO1(PWMO1_OUTPUT);                    //PWM1 Enable

        TB1C0Set(0x0002);     // TB1C0L[7:0] is setting the PWM frequency.
        TB1C1Set(0x0001);     // TB1C1L[7:0] is setting the PWM Duty Cycle.
        TB1C2Set(0x0080);     //       TB1C2L[7:0] is setting the PWM Duty Cycle
                                                    // TB1C0 > TB1C2 > TB1C1
#endif

/*-------------------------------------PWM7O-------------------------------------------*/
#ifdef PWM7O
        TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_16bit ,TB1RT_LogicH );

        TB1_PWM0_PHASE(PA0IV_NORMAL);          //PWM0 Output Phase Normal
        TB1_PWM1_PHASE(PA1IV_INVER);           //PWM1 Output Phase Normal

        TB1_PWM0ModeSelect(PWMA0_PWM7O);            //PWM0 Output mode Select PWM7O
        TB1_PWM1ModeSelect(PWMA1_PWM7O);            //PWM1 Output mode Select PWM7O

        TB1_PWMO0(PWMO0_OUTPUT);                    //PWM0 Enable
        TB1_PWMO1(PWMO1_OUTPUT);                    //PWM1 Enable

        TB1C0Set(0x1FFF);     // TB1C0[15:0] is setting the PWM frequency.
                                                    // PWM Duty Cycle is always 50%
#endif

        TB1_ClearTMB1();      //TMB COUNT CLR
        TB1Enable();

        while(1);

}
/*-------------------------------------------------------------------------*/
/* Subroutine Function                                                     */
/*-------------------------------------------------------------------------*/
void GPIOInit(void)
{
#ifdef PT16_PWM0
        GPIO_PT1InputEnable(PT16_H);                //PT16 Digital mode
        GPIO_PT1OutputMode(PT16_H);                 //PT16 Output mode
        GPIO_PT1OutputLow(PT16_MSK);        //PT16 Output Low
        GPIO_PM16Sel(PM16_PWM0);                    //PT16 PWM0 Enable
#endif

#ifdef PT13_PWM0_1
        GPIO_PT1InputEnable(PT13_H);                //PT13 Digital mode
        GPIO_PT1OutputMode(PT13_H);                 //PT13 Output mode
        GPIO_PT1OutputLow(PT13_MSK);        //PT13 Output Low
        GPIO_PM13Sel(PM13_PWM0_1);                  //PT13 PWM0_1 Enable
#endif
```

```
#ifdef PT32_PWM0_2
        GPIO_PT3InputEnable(PT32_H);                    //PT32 Digital mode
        GPIO_PT3OutputMode(PT32_H);                 //PT32 Output mode
        GPIO_PT3OutputLow(PT32_MSK);        //PT32 Output Low
        GPIO_PM32Sel(PM32_PWM0_2);          //PT32 PWM0_2 Enable
#endif

#ifdef PT12_PWM1
        GPIO_PT1InputEnable(PT12_H);                    //PT12 Digital mode
        GPIO_PT1OutputMode(PT12_H);                 //PT12 Output mode
        GPIO_PT1OutputLow(PT12_MSK);        //PT12 Output Low
        GPIO_PM12Sel(PM12_PWM1);                    //PT12 PWM1 Enable
#endif

#ifdef PT20_PWM1_1
        GPIO_PT2InputEnable(PT20_H);                    //PT20 Digital mode
        GPIO_PT2OutputMode(PT20_H);                 //PT20 Output mode
        GPIO_PT2OutputLow(PT20_MSK);        //PT20 Output Low
        GPIO_PM20Sel(PM20_PWM1_1);                  //PT20 PWM1_1 Enable
#endif

#ifdef PT33_PWM1_2
        GPIO_PT3InputEnable(PT33_H);                    //PT33 Digital mode
        GPIO_PT3OutputMode(PT33_H);                 //PT33 Output mode
        GPIO_PT3OutputLow(PT33_MSK);        //PT33 Output Low
        GPIO_PM33Sel(PM33_PWM1_2);          //PT33 PWM1_2 Enable
#endif


}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

7. 數位 IP(BIE)

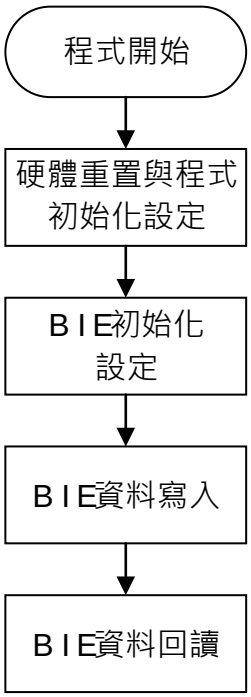
7.1. 範例名稱

Demo_BIE

7.2. 範例說明

(1) BIE 自我燒錄與讀取使用說明
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 開啟 BIE 功能，將欲寫入之數據存放於暫存器 EERD0~EERD31。
(4) 執行 BIE 寫入功能，把存放於暫存器 EERD0~EERD31 之數據寫入 EEPROM。
(5) 清除 EERD0~EERD31 資料以便回讀時確認是否正確。
(6) 執行 BIE 讀取功能，回讀的資料可透過暫存器 EERD0~EERD31 檢視。

7.3. 軟體流程



7.4. 程式碼

```
#define   USE_HY17M24_2M
/*******************************************************************************
 * Demo_BIE.c                                                                  *
 * ---------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                              *
 * http://www.hycontek.com/                                                   *
 *                                                                              *
 * Program Description:                                                       *
 *                                                                              *
 * For External Input                                                         *
 * IC Body: HY17M24                                                             *

 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <BIE.h>
#include <RST.h>


/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void EEDATA_Write(void);
void EEDATA_CLR(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    unsigned char i;
    //PT1.0 HW Reset Function Enable
    PT10_HWResetEnable();

    BIE2_Enable();                  //BIE2 ENABLE
    EEDATA_Write();

    BIE2_DataWriter();      //write data to EEPROM

    EEDATA_CLR();           //CLR EERD0~EERD31

    BIE2_DataRead();        //Reload EERD0~EERD31 from Information 2

    while(1);
```

```
}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* EPROM DATA WRITE                                                          */
/*---------------------------------------------------------------------------*/
void EEDATA_Write(void)              // EERD0~EERD31 is EEPROM data register
{
    unsigned char i=0x00;
    EERD0 = i++;    EERD1 = i++;    EERD2 = i++;    EERD3 = i++;
    EERD4 = i++;    EERD5 = i++;    EERD6 = i++;    EERD7 = i++;
    EERD8 = i++;    EERD9 = i++;    EERD10 = i++;   EERD11 = i++;
    EERD12 = i++;   EERD13 = i++;   EERD14 = i++;   EERD15 = i++;
    EERD16 = i++;   EERD17 = i++;   EERD18 = i++;   EERD19 = i++;
    EERD20 = i++;   EERD21 = i++;   EERD22 = i++;   EERD23 = i++;
    EERD24 = i++;   EERD25 = i++;   EERD26 = i++;   EERD27 = i++;
    EERD28 = i++;   EERD29 = i++;   EERD30 = i++;   EERD31 = i++;
}

/*---------------------------------------------------------------------------*/
/* EPROM DATA CLR                                                            */
/*---------------------------------------------------------------------------*/
void EEDATA_CLR(void)                //CLR EERD0~EERD31 data
{
    EERD0 = 0;EERD1 = 0;EERD2 = 0;EERD3 = 0;
    EERD4 = 0;EERD5 = 0;EERD6 = 0;EERD7 = 0;
    EERD8 = 0;EERD9 = 0;EERD10 = 0;     EERD11 = 0;
    EERD12 = 0;     EERD13 = 0;     EERD14 = 0;     EERD15 = 0;
    EERD16 = 0;     EERD17 = 0;     EERD18 = 0;     EERD19 = 0;
    EERD20 = 0;     EERD21 = 0;     EERD22 = 0;     EERD23 = 0;
    EERD24 = 0;     EERD25 = 0;     EERD26 = 0;     EERD27 = 0;
    EERD28 = 0;     EERD29 = 0;     EERD30 = 0;     EERD31 = 0;
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

8. 數位 IP(GPIO_BZ)

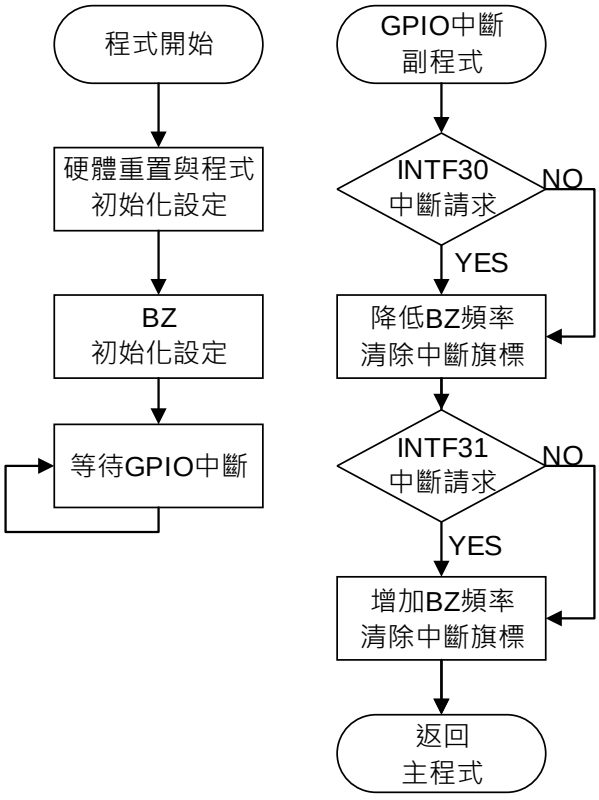
8.1. 範例名稱

GPIO_BZ

8.2. 範例說明

(1) GPIO 使用範例程式.
(2) 利用程式碼#define 來選擇要編譯執行 BZ, BZ_1 或 BZ_2
(3) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(4) 程式做好 BZ 初始化動作，設置 PT1.5 作為輸出，設置 PT3.1/PT3.0 作為按鈕輸入，開啟 GIE 等待 GPIO 中斷發生。
(5) 每進一次 PT3.0 中斷，會在中斷副程式內降低 BZ 輸出頻率，可透過 BZ 輸出腳位觀察。
(6) 每進一次 PT3.1 中斷，會在中斷副程式內增加 BZ 輸出頻率，可透過 BZ 輸出腳位觀察。

8.3. 軟體流程



8.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * GPIO_BZ.c                                                                 *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *       HY17M24                                                             *
 * -----------------                                                         *
 *                            |                                              *
 *                            |                                              *
 *             PT1.5     |-> BZ                                             *
 *                            |                                              *
 *                            |                                              *
 * -----------------                                                         *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *
 *                                                                           *
 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <PWR.h>
#include <GPIO.h>
#include <RST.h>
#include <CLK.h>
#include <WDT.h>

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#define BZ
//#define BZ_1
//#define BZ_2
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void GPIOInit(void);
void BZInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char cks=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
```

```
void main(void)
{

      //PT1.0 HW Reset Function Enable
      PT10_HWResetEnable();
////    CPU CLK Init              //
       CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );

       GPIOInit();
       BZInit();
       GIE_Enable();
       while(1);
}


/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
       if(INTF30_IsFlag())
       {
              Delay(2000);              //debounce
              while(!(GPIO_PT3GET(PT30_H)));  //wait PT30=0

              if(++cks >3) cks = 3;
       }
       if(INTF31_IsFlag())
       {
              Delay(2000);              //debounce
              while(!(GPIO_PT3GET(PT31_H)));  //wait PT31=0
              if(--cks >3) cks = 0;
       }
       switch(cks)
       {
              case 0:
                     BZ_BZCKSelect(DBZ_DZCKDIV2);        //Output BZ_CK = BZ_CLK / 2
              break;

              case 1:
                     BZ_BZCKSelect(DBZ_DZCKDIV4);        //Output BZ_CK = BZ_CLK / 4
              break;

              case 2:
                     BZ_BZCKSelect(DBZ_DZCKDIV8);        //Output BZ_CK = BZ_CLK / 8
              break;

              case 3:
                     BZ_BZCKSelect(DBZ_DZCKDIV16);              //Output BZ_CK = BZ_CLK / 16
              break;
       }
       INTF30_ClearFlag();
       INTF31_ClearFlag();

}
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
```

```
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/*            GPIO Init function                                             */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
    GPIO_PT3InputEnable(IN30_H | IN31_H);          //PT30.PT31 Digital mode
    GPIO_PT3InputEnable(IN31);
    GPIO_PT3InputMode(TC30_L | TC30_L);            //PT30.PT31 Input mode
    GPIO_PT3SETPU(PU30_H | PU31_H);                    //PT30.PT31 Pull-up    resistors Enable
    INTG30_Edgefall();          //PT30 INT triggering conditions 1->0
    INTE30_Enable();            //PT30 INT Enable
    INTG31_Edgefall();          //PT31 INT triggering conditions 1->0
    INTE31_Enable();            //PT31 INT Enable
}
/*---------------------------------------------------------------------------*/
/*            BZ Init function                                               */
/*---------------------------------------------------------------------------*/
void BZInit(void)
{
    CLK_DMSCKSelect(DMS_DHSCKDIV2); //DMS_CK = DHS_CK /2
    BZ_CLKSelect(BZS_LSCK);                        //BZ_CLK = LS_CK = DMS_CK / 8
    BZ_BZCKSelect(DBZ_DZCKDIV2);        //Output BZ_CK = BZ_CLK / 2

    //      BZ -> PT15 //
#ifdef BZ
    GPIO_PT1InputEnable(IN15_H);        //PT15 Digital mode
    GPIO_PT1OutputMode(TC15_H);         //PT15 Output mode
    GPIO_PM15Sel(PM15_BZ);                  //PT15 BZ mode
#endif

    //      BZ_1 -> PT21    //
#ifdef BZ_1
    GPIO_PT2InputEnable(IN21_H);        //PT21 Digital mode
    GPIO_PT2OutputMode(TC21_H);         //PT21 Output mode
    GPIO_PM21Sel(PM21_BZ_1);                //PT21 BZ_1 mode
#endif

    //      BZ_2 -> PT34    //
#ifdef BZ_2
    GPIO_PT3InputEnable(IN34_H);        //PT34 Digital mode
    GPIO_PT3OutputMode(TC34_H);         //PT34 Output mode
    GPIO_PM34Sel(PM34_BZ_2);                //PT34 BZ_2 mode
#endif
    BZ_Enable();

}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
   __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
```

```
/* End Of File                                                                 */
/*-----------------------------------------------------------------------------*/
```

9. 類比 IP(12 bit Resistance Ladder DAC)

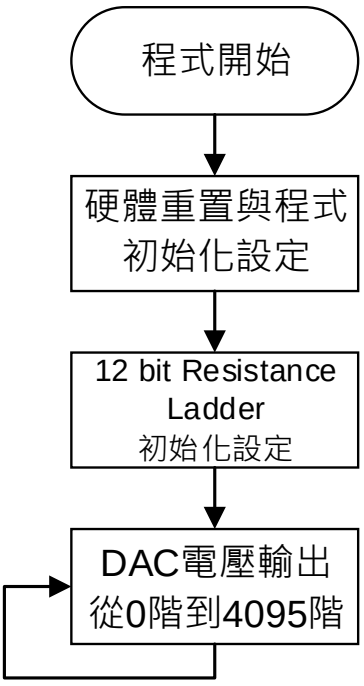
9.1. 範例名稱

DAC

9.2. 範例說明

(1) 12 bit Resistance ladder DAC 使用說明
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 程式先將 VDDA 做開啟動作，設置 VDDA 為 12 bit Resistance Ladder 的正端, 負端設置為 VSS。在此設置下， DAC 能夠輸出的最高電壓為 VDDA。
(4) DAC 的輸出由第 0 階開始輸出到 4095 階，可以由 IC 腳位 DACO 量到 DAC 輸出電壓。

9.3. 軟體流程



9.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * DAC.c                                                                     *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *         HY17M24                                                           *
 * ------------------                                                        *
 *                            |                                              *
 *                            |                                              *
 *                PT1.4|-> DACO                                              *
 *                            |                                              *
 *                            |                                              *
 *-------------------                                                        *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *

 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <PWR.h>
#include <GPIO.h>
#include <RST.h>
#include <ADC.h>
#include <CLK.h>
#include <ladder.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned int i,dabit=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

////     CPU CLK Init                //
      CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );
```

```
////      VDDA setup      //
      PWR_BGREnable();                                  //bandgap Vref Enable
      PWR_VDDAOpen(LDOC_2V4);        //VDDA Enable
      PWR_LDOMode(LDOM_VDD);
      Delay(500);      //Delay 10ms

////    ACMint ENABLE  //
//      REFOI_ACMint_Open(SELVIN_1V2);

////    REFOint ENABLE        //
//      REFOI_REFOint_Open(REFOS_1V2);

////    DAC Init              //
      DAC_Open(LDOC_2V9,DAPS_VDDA,DANS_VSS, dabit);

        while(1)
        {
            for(dabit=0;dabit< 4096;dabit++)
            {
                DACBitH = dabit >> 8;                    //      DABIT[11:0]
                DACBitL = dabit & 0xff;
                Delay(100);
            }
        }

}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
        for(;num>0;num--)
      __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

10. 類比 IP(OPA)

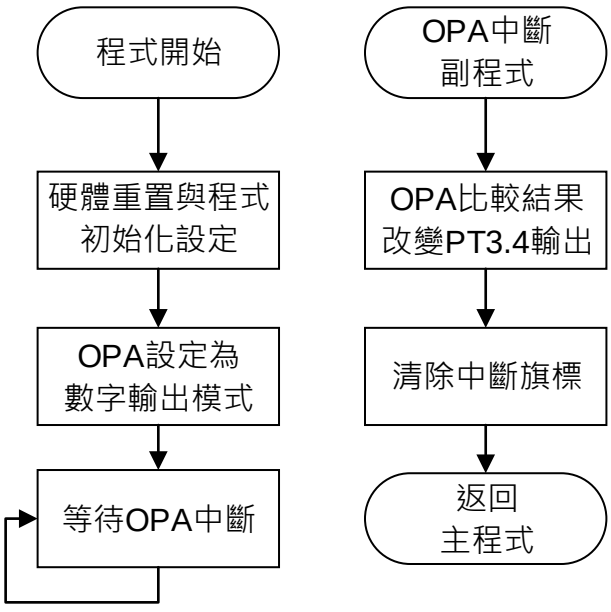
10.1. 範例名稱

HY17M24_OPA

10.2. 範例說明

(1) OPAMP 使用與設定方式。
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 開啟 VDDA 電壓，設置 REFO=2.0V。
(4) 程式選擇 AI0 為 OPAMP 的正端，OPAMP 的負端選擇 REFOI。
(5) 將 OPAMP 當作比較器使用，當 REFO 電壓大於 AI0，OPC 狀態為 0，當 REFO 電壓小於 AI0，OPC 狀態為 1。
(6) 每當 OPAMP 進入中斷一次，根據 OPC 狀態改變 PT3.4 輸出值。

10.3. 軟體流程



10.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * DAC.c                                                                     *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *         HY17M24                                                           *
 * ------------------                                                        *
 *                            |                                              *
 *                            |                                              *
 *                PT1.4|-> DACO                                              *
 *                            |                                              *
 *                            |                                              *
 *-------------------                                                        *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *

 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <PWR.h>
#include <GPIO.h>
#include <RST.h>
#include <ADC.h>
#include <CLK.h>
#include <ladder.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned int i,dabit=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

////     CPU CLK Init                //
      CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );
```

```
////      VDDA setup      //
      PWR_BGREnable();                                  //bandgap Vref Enable
      PWR_VDDAOpen(LDOC_2V4);        //VDDA Enable
      PWR_LDOMode(LDOM_VDD);
      Delay(500);      //Delay 10ms

////    ACMint ENABLE  //
//      REFOI_ACMint_Open(SELVIN_1V2);

////    REFOint ENABLE        //
//      REFOI_REFOint_Open(REFOS_1V2);

////    DAC Init              //
     DAC_Open(LDOC_2V9,DAPS_VDDA,DANS_VSS, dabit);

       while(1)
       {
          for(dabit=0;dabit< 4096;dabit++)
          {
               DACBitH = dabit >> 8;                    //      DABIT[11:0]
               DACBitL = dabit & 0xff;
               Delay(100);
          }
       }

}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
       for(;num>0;num--)
     __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

11. 類比 IP(ADC)

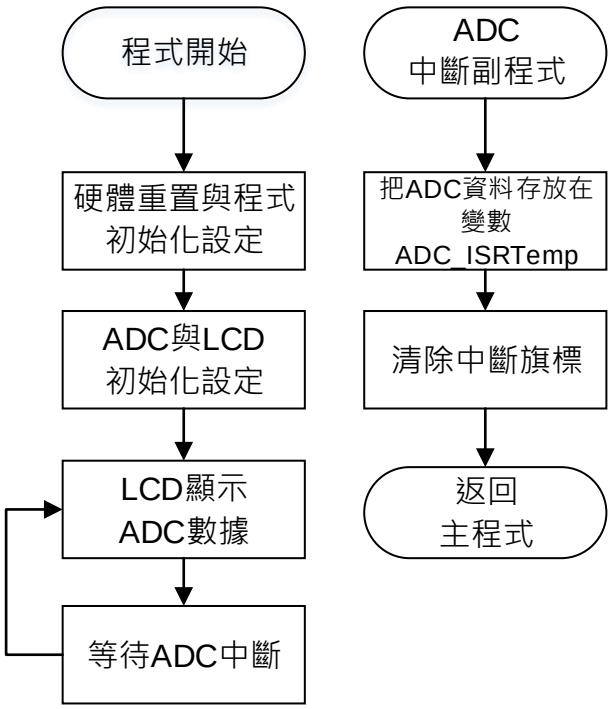
11.1. 範例名稱

ADC_Display

11.2. 範例說明

(1) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(2) 透過 ADC 相關設定，可以實現 ADC 進入中斷抓取 ADC 輸出暫存器資料。
(3) ADC 初始設置包含，設置 ADC 的類比電源為 VDDA，設置 ADC 的取樣頻率 OSR 為 65536，設置 ADC 的輸入端設置為 AIO0-AIO1，設置 ADC 的參考電壓設置為 VDDA 對 VSS。
(4) ADC 初始設置完成後，開啟 GIE 等待 ADC 中斷發生。ADC OSR 設置可決定 ADC 進出中斷的速度。進入 ADC 中斷抓取的暫存器資料透過 I2C 與 HY2613 LCD Driver 通訊，將 ADC 輸出資料顯示在 LCD 上。

11.3. 軟體流程



11.4. 程式碼

```
#define   USE_HY17M24_2M
/*******************************************************************************
 * ADC_Display.c                                                               *
 * --------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *       HY17M24                                                               *
 *------------------                                                           *
 *                          |                                                  *
 *                  AI0   |->INP                                               *
 *                  AI1   |->INN                                               *
 *                          |                                                  *
 *                          |                                                  *
 *             PT3.0   |->SCL                                                  *
 *             PT3.1 |->SDA                                                    *
 *                          |                                                  *
 *------------------                                                           *
 *                                                                             *
 *                                                                             *
 *                                                                             *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M24                                                            *

 *******************************************************************************/
/*-----------------------------------------------------------------------------*/
/* Includes                                                                    */
/*-----------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <RST.h>
#include <I2C.h>
#include <PWR.h>
#include <ADC.h>
#include "seg7.h"
#include "HY2613.h"

/*-----------------------------------------------------------------------------*/
/* DEFINITIONS                                                                 */
/*-----------------------------------------------------------------------------*/
#define I2CBufferSize    11             // HY17M24 max BufferSize 127
#define I2C_WRITE    0x00           // I2C WRITE command
#define I2C_READ     0x01           // I2C READ command
#define I2C_Delay    1000000       //HAO=2MHz, wait 5s
volatile typedef union _Flag
{
  unsigned int    _byte;
    struct
```

```
  {
    unsigned b_ADCdone:1;
    unsigned b_TMAdone:1;
    unsigned b_UART_TxDone:1;
    unsigned b_UART_RxDone:1;
    unsigned Reserved:12;
  };
} Flag;

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void SysInit(void);
void I2CInit(void);
void Delay(unsigned int num);
void ADCInit(void);
void SendDisplay(signed long   data);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
signed long   ADC_ISRTemp;
signed long   ADC_ISRTemp_Buffer[8];
signed long   ADCData1;
Flag     Flagbits;
unsigned char I2C_RW;
unsigned char I2C_TARGET;          // Target I2C slave address
unsigned char I2C_Sendbuf[I2CBufferSize];
unsigned char I2C_Recbuf[I2CBufferSize];
unsigned char I2C_EndFlag;
unsigned int I2C_DataTxLen,I2C_DataTxIndex,I2C_DataRxLen,I2C_DataRxIndex;
unsigned char DisplayBuffer[9];
unsigned char i;

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
        SysInit();
        I2CInit();
        ADCInit();
        Flagbits.b_UART_TxDone=1;
        GIE_Enable();
        ClearLCDframe();                //CLR Display

        while(1)
        {
                Flagbits.b_ADCdone=0;
        while(Flagbits.b_ADCdone==0);   //until get ADC_ISRTemp
        Flagbits.b_ADCdone=0;

        ADC_INT_Disable();
        SendDisplay(ADC_ISRTemp);
        ADC_INT_Enable();
        }
}
```

```
/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
    if(ADC_INT_IsFlag())
    {
        ADC_INT_ClearFlag();
        ADC_ISRTemp=ADCR; //ADC_GetData();
        Flagbits.b_ADCdone=1;
    }

    if(I2CIF_IsFlag() &&I2C_EndFlag==0)   //Get I2C Interrupt Flag
    {
    switch(I2C_STAState())
    {
        case 0x90:   //MACTFlag+RWFlag
            //START has been transmitted
            I2C_SendData(I2C_TARGET);   //Send Slave Address & R/W Bit
            I2C_Ctrl(0,0,0,0);              //Clear all I2C flag
            break;

        case 0x84:   //MACTFlag+ACKFlag
            //Slave A + W has been transmitted. ACK has been received.
            I2C_SendData(I2C_Sendbuf[I2C_DataTxIndex++]);   //Send Data to Slave
            I2C_Ctrl(0,0,0,0);          //Clear all I2C flag
            break;

        case 0x80: //MACTFlag
            //Slave A + W has been transmitted. ACK has been received.
                I2C_SendData(I2C_Sendbuf[I2C_DataTxIndex++]);   //Send Data to Slave
            I2C_Ctrl(0,0,0,0);          //Clear all I2C flag
            break;

        case 0x30:

            I2C_Ctrl(0,0,0,0);          //Clear all I2C flag
            I2C_EndFlag=1;
            break;

        case 0x8C:   //MACTFlag+DFFlag+ACKFlag
            //DATA has been transmitted and ACK has been received
            if(I2C_DataTxIndex<I2C_DataTxLen)
            {
              I2C_SendData(I2C_Sendbuf[I2C_DataTxIndex++]);   //Send Data to Slave
              I2C_Ctrl(0,0,0,0);       // Clear all I2C flag
            }
            else
            {
                if(I2C_RW == I2C_WRITE)
                {
                    I2C_Ctrl(0,1,0,0);   //I2C as master sends STOP signal
                    I2C_EndFlag=1;
                }
                else if(I2C_RW == I2C_READ)
                I2C_Ctrl(1,0,0,0);   //I2C as master sends START signal
                I2C_DataTxIndex=0;
```

```
            }
            break;

        case 0x88:    //MACTFlag+DFFlag
            //DATA has been transmitted and NACK has been received
            I2C_Ctrl(0,1,0,0);       //I2C as master sends STOP signal
            I2C_DataTxIndex=0;
            I2C_EndFlag=1;
            break;

        case 0xB0:
            //A repeated START has been transmitted.
            I2C_SendData(I2C_TARGET | I2C_READ);    //Send Slave Address & R/W Bit
            I2C_Ctrl(0,0,0,0);       //Clear all I2C flag
            break;

        case 0x94:    //MACTFlag+RWFlag+ACKFlag
            //Slave A + R has been transmitted. ACK has been received.
            if(I2C_DataRxLen>1)
            {
               I2C_Ctrl(0,0,0,1);    //Set ACK bit
            }else{
               I2C_Ctrl(0,0,0,0);    //Clear all I2C flag
            }
            break;

        case 0x9C:    //MACTFlag+RWFlag+DFFlag+ACKFlag
            //Data byte has been received. ACK has been transmitted.
            I2C_ReceiveDATA(I2C_Recbuf[I2C_DataRxIndex++]);
            //I2C_Recbuf[I2C_DataRxIndex++]=I2C_ReceiveDATA();
            if((I2C_DataRxLen-1)>I2C_DataRxIndex)
            {
               I2C_Ctrl(0,0,0,1);    //Set ACK bit
            }
            else
            {
               I2C_Ctrl(0,0,0,0);    //Clear all I2C flag
            }
            break;

        case 0x98:    //MACTFlag+RWFlag+DFFlag
            //Data byte has been received. NACK has been transmitted.
            I2C_ReceiveDATA(I2C_Recbuf[I2C_DataRxIndex++]);
            // I2C_Recbuf[I2C_DataRxIndex++]=I2C_ReceiveDATA();
            I2C_Ctrl(0,1,0,0);    //I2C as master sends STOP signal
            I2C_EndFlag=1;
            break;

        default:
            I2C_Ctrl(0,0,0,0);    //Clear all I2C flag
            I2C_EndFlag=1;
            break;
    }
        I2C_I2CINT_CLEAR();          //CLR I2C INT Flag
        I2C_I2CER_CLEAR();           //CLR I2C error Flag
        I2CIF_ClearFlag();                      //CLR I2CIF
}
```

```
        if(I2CERIF_IsFlag())    //Get I2C Error Interrupt Flag
        {
              I2C_EndFlag=1;
                I2C_I2CINT_CLEAR();          //CLR I2C INT Flag
                I2C_I2CER_CLEAR();           //CLR I2C error Flag
                I2CERIF_ClearFlag();         //CLR I2CERIF
              I2C_Ctrl(0,0,0,0);             //Clear all I2C flag
        }


}

/*---------------------------------------------------------------------------*/
/* System Init Function                                                      */
/*---------------------------------------------------------------------------*/
void SysInit(void)
{
        //PT1.0 HW Reset Function Ensable
        PT10_HWResetEnable();

        //CPU CLK SELECT
        CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );

}

/*---------------------------------------------------------------------------*/
/* ADC Init Function                                                         */
/*---------------------------------------------------------------------------*/
void ADCInit(void)
{
    //==========Setup Power=====================================
   PWR_BGREnable();                             //bandgap Vref Enable
   PWR_VDDAOpen(LDOC_2V4);   //VDDA Enable
   PWR_LDOMode(LDOM_VDD);
   Delay(500);       //Delay 10ms

     //==========Setup ADC=======================================
  ADC_Open(DADC_DHSCKDIV2,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,
                                    ADGN_1,VREGN_DIV1,DCSET_P0,OSR_65536,VCMS_REFOint);

   //==========Setup ADC INT===================================
   ADC_INT_ClearFlag();          //Clear ADIF
   ADC_INT_Enable();     //ADC INT Enable
   ADC_CMFREnable();         //CMFR=1, Comb Filter Reset

}

/*---------------------------------------------------------------------------*/
/*                          Send ADC Data to Display                         */
/*---------------------------------------------------------------------------*/
void SendDisplay(signed long    data)
{
        unsigned int num;
      //display range +8388607~-8388608 (dec)
        ADCData1=data;
        if((ADCData1<0)||(ADCData1>0x80000000))    // plus-minus sign judgment
        {
```

```
            ADCData1=~ADCData1;
            ADCData1++;
            DisplayBuffer[6]=S_Minus;                    // "-"
        }
        else
        {
            DisplayBuffer[6]=0;
        }
        DisplayBuffer[0]=seg[(ADCData1 /1000000)];    //ten thousand
        ADCData1=ADCData1%1000000;
        DisplayBuffer[1]=seg[(ADCData1 /100000)] ;    //thousand
        ADCData1=ADCData1%100000;
        DisplayBuffer[2]=seg[(ADCData1 /10000)] ;    //hundred
        ADCData1=ADCData1%10000;
        DisplayBuffer[3]=seg[(ADCData1 /1000)] ;    //ten
        ADCData1=ADCData1%1000;
        DisplayBuffer[4]=seg[(ADCData1 /100)] ;    //unit
        ADCData1=ADCData1%100;
        DisplayBuffer[5]=seg[(ADCData1 /10)] ;    //unit
//      ADCData1=ADCData1%10;

        DisplayBuffer[7]=0x00;
        DisplayBuffer[8]=0x00;
        RAM2LCD(DisplayBuffer,9);
        for(num=10000;num>0;num--)          __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* I2C Init Function                                                         */
/*---------------------------------------------------------------------------*/
void I2CInit(void)
{
        I2C_Open(50);             //I2C Open

        I2C_SetIOPin(2); // SCL-> PT3.0    SDA->PT3.1

        I2CIE_Enable();
        I2CERIE_Enable();
}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
        for(;num>0;num--)
      __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

12. 類比 IP(CMP)

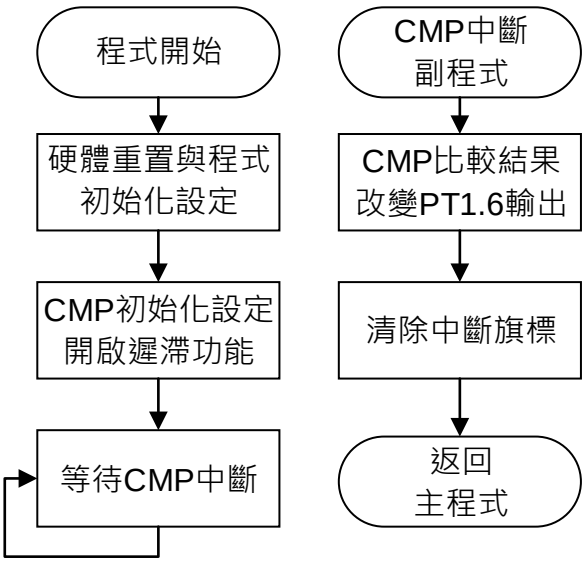
12.1. 範例名稱

HY17M24_CMP

12.2. 範例說明

(1) CMP 使用方式說明
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 透過 CMP 相關設定，可以實現遲滯控制。
(4) CMP 初始設置包含, 設置類比電源 VDDA，設置 CMP 的輸入端為 RLO-CH2(PT3.7)，設置 CMP 內建多節點電阻分壓值(RLO)。
(5) 透過遲滯控制器 CPDM 暫存器與節點控制器 CPDA 暫存器控制遲滯切換區間。當 PT3.7 高於 RLO，PT1.6 輸出”1”，則 CPOB= 0= CPDA　改變節點電壓(RLO)；當 PT3.7 低於 RLO，PT1.6 輸出”0”，則 CPOB=1= CPDA 改變節點電壓。節點選擇器在兩節點之間切換，實現遲滯功能。

12.3. 軟體流程



12.4. 程式碼

```
#define   USE_HY17M24_2M
/*******************************************************************************
 * HY17M24_CMP.c                                                               *
 * ---------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *            HY17M24                                                          *
 *--------------------                                                         *
 *                              |                                              *
 *                   PT1.6|->Output                                           *
 *                              |                                              *
 *                   PT3.7|->Input                                            *
 *                              |                                              *
 *--------------------                                                         *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M24                                                            *

 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <PWR.h>
#include <GPIO.h>
#include <RST.h>
#include <ADC.h>
#include <CLK.h>
#include <CMP.h>

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define hysteresis_mode
#define LVD_mode

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void SysInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
```

```
    SysInit();
#ifdef hysteresis_mode
    CMP_Open(CPPS_CH2,CPNS_RLO,CMPHS_NORMAL,CPOR_NORMAL,ENRCC_PT16);
    GPIO_PT3SETDA(DA37_H);              //PT37 CH2 INPUT
    CMP_RLOSet(CPRH_VDDA,CPRL_OPEN,CPDA_5DIV32,CPDM2_ENABLE);    // Lower than 0.81V &  Higher
0.97V
    PWR_VDDAOpen(LDOC_2V9);
#endif

#ifdef LVD_mode
    CMP_Open(CPPS_1V2,CPNS_RLO,CMPHS_NORMAL,CPOR_NORMAL,ENRCC_PT16);
    CMP_RLOSet(CPRH_VDD,CPRL_OPEN,CPDA_12DIV32,CPDM_DISABLE );    // VDD=3.3V, LVD=2.8
#endif


    GIE_Enable();
    while(1);
}


/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* System Init Function                                                      */
/*---------------------------------------------------------------------------*/
void SysInit(void)
{
    //PT1.0 HW Reset Function Disable
   PT10_HWResetEnable();

    //CPU CLK SELECT
    CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );

    //PT16 OUTPUT
    GPIO_PT1OutputMode(TC16_H);         //PT16 Output mode
    GPIO_PT1InputEnable(IN16_H);        //PT16 Digital mode

}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
   __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

13. 通訊 IP(UART)

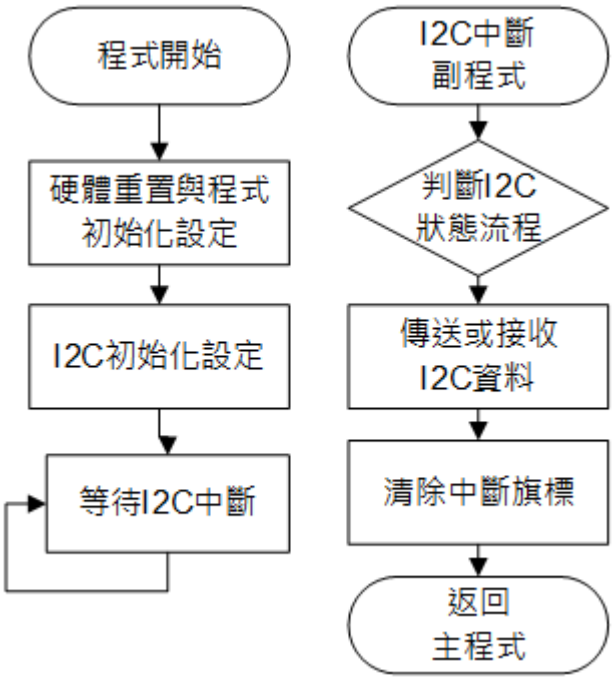
13.1. 範例名稱

HY17M24_UART

13.2. 範例說明

(1) HY17M24 使用 UART 做 TX 與 RX 傳輸範例程式
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 開啟 UART 功能，設置 UART 相關設定，字串傳送速率 9600。
(4) 程式開始執行，UART 會一直送字串”ABCDEF”，直到收到’abcedf’字串後會送出字串”abcedf”並且 TX 停止傳送。當收到字串不是”abcedf”， 則會繼續發送”ABCDEF”字串。

13.3. 軟體流程



13.4. 程式碼

```
#define   USE_HY17M24_2M
/*******************************************************************************
 * HY17M24_UART.c                                                              *
 * ---------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *               HY17M24                                                       *
 * ----------------------                                                      *
 *                                  |                                          *
 *                       PT1.5|->TX                                           *
 *                       PT1.6|->RX                                           *
 *                                  |                                          *
 *----------------------                                                       *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M24                                                            *

 *******************************************************************************/
/*-------------------------------------------------------------------------*/
/* Includes                                                                */
/*-------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <RST.h>
#include <GPIO.h>
#include <UART.h>

/*-------------------------------------------------------------------------*/
/* DEFINITIONS                                                             */
/*-------------------------------------------------------------------------*/
#define UART_ABD

volatile typedef union _Flag
{
 unsigned int   _byte;
  struct
  {
    unsigned b_ADCdone:1;
    unsigned b_TMAdone:1;
    unsigned b_UART_TxDone:1;
    unsigned b_UART_RxDone:1;
    unsigned Reserved:12;
  };
} Flag;

//        0        1        2        3
//       TX   PT1.5   PT2.0   PT3.6   PT3.4
//       RX   PT1.6   PT2.1   PT3.7   PT3.5
```

```
#define   Uart_RX_BufferSize 6
#define   Uart_TX_BufferSize 10
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void SysInit(void);
void Delay_NOP(unsigned int del);
void CLOCKInit(void);
void UARTInit(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
Flag     Flagbits;
unsigned char UartTxBuffer[Uart_TX_BufferSize]={0};
unsigned char UartRxBuffer[Uart_RX_BufferSize]={0};
unsigned char UartTxIndex,UartTxLength;
unsigned char UartRxIndex,UartRxLength;
unsigned char STOP_TO_SEND_UART;
//STOP_TO_SEND_UART command
unsigned char UartRxBuffer_Command[Uart_RX_BufferSize]=
{
0x61,0x62,0x63,0x64,0x65,0x66 //ASCII    KEYWORD = abcdef
};
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
   //PT1.0 HW Reset Function Enable
   PT10_HWResetEnable();

   //UART Demo
   STOP_TO_SEND_UART=0;
   Flagbits.b_UART_TxDone=0;
   Flagbits.b_UART_RxDone=0;
   CLOCKInit();
   UARTInit();
   Flagbits.b_UART_TxDone=1;
   GIE_Enable();

   UartTxIndex=0;
   UartRxIndex=0;;

   while(1)
   {


      //UART RX
      if(Flagbits.b_UART_RxDone==1)
      {

         if(
            (UartRxBuffer[5]==UartRxBuffer_Command[5] && UartRxBuffer[4]==UartRxBuffer_Command[4]) &&
            (UartRxBuffer[3]==UartRxBuffer_Command[3] && UartRxBuffer[2]==UartRxBuffer_Command[2]) &&
            (UartRxBuffer[1]==UartRxBuffer_Command[1] && UartRxBuffer[0]==UartRxBuffer_Command[0])
            )
         {
```

```
            //if UART receive == 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
            //send out 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f) and stop to send out
            STOP_TO_SEND_UART=1;
            for(UartTxLength=0;UartTxLength<=Uart_TX_BufferSize;UartTxLength++)
            {
               UartTxBuffer[UartTxLength]=UartRxBuffer[UartTxLength];
               if(UartTxLength==Uart_RX_BufferSize)
               {
                  UartRxIndex=0;
                  Flagbits.b_UART_TxDone=0;
                  UART_INT_TXEnable();       //TXIE=1b
                  UART_INT_RCDisable();   //RCIE=0b
                  while(!Flagbits.b_UART_TxDone);
               }
            }
         }
         else
         {
         //if UART receive != 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
         //User defined at here
         STOP_TO_SEND_UART=0;
         }

         UartRxIndex=0;
         Flagbits.b_UART_RxDone=0;

      }


      if(Flagbits.b_UART_TxDone==1 && STOP_TO_SEND_UART==0)
      {
         UartTxBuffer[7]='\r';
         UartTxBuffer[6]='\n';
         UartTxBuffer[5]=0x46;   //F
         UartTxBuffer[4]=0x45;   //E
         UartTxBuffer[3]=0x44;   //D
         UartTxBuffer[2]=0x43;   //C
         UartTxBuffer[1]=0x42;   //B
         UartTxBuffer[0]=0x41;   //A
         Flagbits.b_UART_TxDone=0;
         UartTxLength=8;
         UartTxIndex=0;
         UART_INT_TXEnable();       //TXIE=1b
         UART_INT_RCEnable();    //RCIE=1b
         while(!Flagbits.b_UART_TxDone);   //If    Flagbits.b_UART_TxDone=DISABLE, stop at here
         Delay_NOP(3000);
      }
   }
}

/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{

   NOP();
```

```
    //UART RX Event
    if(UART_INT_RCIsFlag())
    {
      UartRxBuffer[UartRxIndex]=RC0REG;
      UartRxIndex++;
      if(UartRxIndex>=Uart_RX_BufferSize)
      {
        UartRxIndex=0;
        Flagbits.b_UART_RxDone=1;
      }
      UART_INT_RCClearFlag();
    }

    //UART TX Event
    if(UART_INT_TXIsFlag())
    {
      if(Flagbits.b_UART_TxDone==0)
      {
        TX0R=UartTxBuffer[UartTxIndex++];
        UART_INT_TXClearFlag();
        if(UartTxIndex>=UartTxLength)
        {
          UART_INT_TXEnable();      //TXIE=1b
          UART_INT_RCEnable();   //RCIE=1b
          Flagbits.b_UART_TxDone=1;
          UartTxIndex=0;
        }
      }
        if(Flagbits.b_UART_TxDone==1)
        {
          UART_INT_TXDisable();      //TXIE=0b
          UART_INT_RCEnable();   //RCIE=1b
          UART_INT_TXClearFlag();
        }
      }

}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
void Delay_NOP(unsigned int del)
{
    while(--del)
    NOP();
}

/*---------------------------------------------------------------------------*/
/* UART Init Function                                                        */
/*---------------------------------------------------------------------------*/
void UARTInit(void)
{
#ifdef UART_ABD
    UART_Open(0 ,8 ,PARITY_None ,0);
    UART_WUEDisable();                  // Wake-up disable
    UART_ABDEnable();                   //Enable Auto Baudrate
```

```
   while((BA0CN & 0x01) == 1 );          //Wait for master send 055H
   UART_INT_RCClearFlag();               //Clear RC interrupt flag
#else
   UART_Open(48 ,8 ,PARITY_EVEN ,0);      //HAO=2MHz,baud=9600
#endif
   TXIE_Enable();
   RCIE_Enable();

}
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{

   CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );

}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

14. 通訊 IP(I2C)

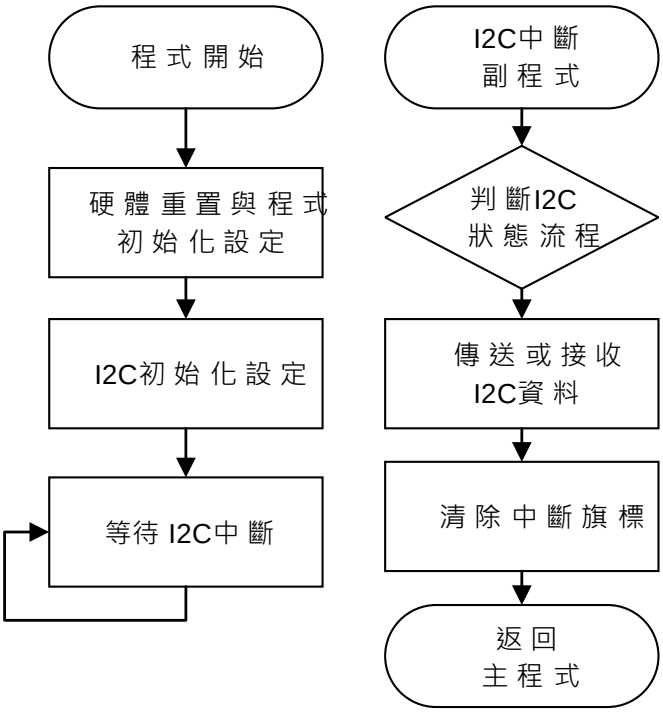
14.1. 範例名稱

I2C_E2PROM

14.2. 範例說明

(1) HY17M24 工作在 I2C Master Mode, 對 I2C EEPROM 24C02 做寫入與讀取控制範例.
(2) 範例程式內容包含 HY17M24 之 I2C Master 設置初始化過程，並且對一個 I2C EEPROM(24C02)裝置做單次的資料寫入和讀取與連續的資料寫入讀取範例
(3) 程式第一段功能為做單次性的資料寫入和讀取在 EEPROM 裝置，首先由 I2C Master 寫入資料 0xAA 在 EEPROM 的 WORD ADDRESS 0x00，然後再下 Read 指令讀回 EEPROM 的 WORD ADDRESS 0x00 位置，如果程式正確無誤，則會由 I2C Master 端讀回到 0x01 的數值資料。
(4) 程式第二段功能為做連續性的資料寫入和讀取在 EEPROM 裝置，從 EEPROM 的 WORD ADDRESS 0x01 開始寫入 4bytes 資料直到 WORD ADDRESS 0x04，再依序從 EEPROM 的 WORD ADDRESS 0x00 連續讀取，分別為 2、5、6 筆資料，觀察資料是否已經正確被寫入。
(5) 在本文範例程式裡面， 當執行完程式第一段和第二段的正確的執行結果為 EEPROM Address 0x00 資料等於 0xAA，Address 0x01 資料等於 0x02，Address 0x02 資料等於 0x03，Address 0x03 資料等於 0x04，Address 0x04 資料等於 0x05。

14.3. 軟體流程


程式開始
硬體重置與程式初始化設定
I2C初始化設定
等待I2C中斷
I2C中斷副程式
判斷I2C狀態流程
傳送或接收I2C資料
清除中斷旗標
返回主程式

14.4. 程式碼

說明 ： 在此僅貼上 I2C_E2PROM.C 程式內容做參考.

```
#define   USE_HY17M24_2M
/*******************************************************************************
 * I2C_E2PROM.c                                                              *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *           HY17M24                    24C02A                               *
 *---------------------              ------------------                              *
 *                                     |              |                      *
 *                                     |              |                      *
 *                     VDD |----->    | VDD                                  *
 *                   PT3.0 |----->    | SCL                                  *
 *                   PT3.1 |----->    | SDA                                  *
 *                     VSS |----->    | VSS                                  *
 *                                    |               |                      *
 *                                    |               |                      *
 *------------------                   ----------------                              *
 *
*
 * EEPROM Address
*
 * addr[0x00]=0xaa                                                           *
 * addr[0x01]=0x02                                                           *
 * addr[0x02]=0x03                                                           *
 * addr[0x03]=0x04                                                           *
 * addr[0x04]=0x05                                                           *

 * For External Input                                                        *
 * IC Body: HY17M24                                                          *

 *******************************************************************************/
/*--------------------------------------------------------------------------*/
/* Includes                                                                 */
/*--------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <RST.h>
#include <PWR.h>
#include <I2C.h>
#include <GPIO.h>
#include "EEPROM_24Cxx.h"
/*--------------------------------------------------------------------------*/
/* DEFINITIONS                                                              */
/*--------------------------------------------------------------------------*/
#define   I2CBufferSize   10          // HY17M24 max BufferSize 127
#define I2C_WRITE   0x00           // I2C WRITE command
```

```
#define I2C_READ     0x01          // I2C READ command

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void SysInit(void);
void I2CInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char I2C_Read_Buffer[I2CBufferSize];
unsigned char I2C_RW;
unsigned char I2C_TARGET;          // Target I2C slave address
unsigned char I2C_Sendbuf[I2CBufferSize];
unsigned char I2C_Recbuf[I2CBufferSize];
unsigned char I2C_EndFlag;
unsigned char EEPROM_WriteData[4] = {0x02,0x03,0x04,0x05};
unsigned int I2C_DataTxLen,I2C_DataTxIndex,I2C_DataRxLen,I2C_DataRxIndex;
unsigned char i,temp;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    SysInit();
    I2CInit();
    for(i=0;i<I2CBufferSize;i++)
    {
        I2C_Read_Buffer[i]=0x00;      //CLR I2C_Read_Buffer[]
    }

    GIE_Enable();

    // I2C single I2C_WRITE & I2C_READ
    EEPROM_ByteWrite(0x00,0xaa);                  //Single Byte I2C_WRITE word address 0x00, and I2C_WRITE
data 0x01
    Delay(1000);                                  //Delay for EEPROM 24C02 I2C_WRITE Cycle Time
    I2C_Read_Buffer[0]=EEPROM_ByteRead(0x00);    //Single Byte I2C_READ word address 0x00, the I2C_READ
value is 0x01
    Delay(1000);                                 //Delay for EEPROM 24C02 I2C_WRITE Cycle Time

    // I2C sequential I2C_WRITE & I2C_READ
    EEPROM_WriteArray(0x01,EEPROM_WriteData,4); //I2C_WRITE array data to the word address from 0x01 to
0x04
    Delay(1000);                                 //Delay for EEPROM 24C02 I2C_WRITE Cycle Time

    EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 2); //sequential I2C_READ data from 0x00 to 0x01
    //read data:0xaa,0x01
    Delay(1000);                                 //Delay for EEPROM 24C02 I2C_WRITE Cycle Time

    EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 5); //sequential I2C_READ data from 0x00 to 0x04
    //read data:0xaa,0x01,0x02,0x03,0x04
    Delay(1000);                                 //Delay for EEPROM 24C02 I2C_WRITE Cycle Time

    EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 6); //sequential I2C_READ data from 0x00 to 0x05
    //read data:0xaa,0x01,0x02,0x03,0x04,0xff
```

```
        Delay(1000);                                    //Delay for EEPROM 24C02 I2C_WRITE Cycle Time
        while(1);
}
/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
        if(I2CIF_IsFlag() &&I2C_EndFlag==0)    //Get I2C Interrupt Flag
        {
        switch(I2C_STAState())
        {
                case 0x90:    //MACTFlag+RWFlag
                              //START has been transmitted
                        I2C_SendData(I2C_TARGET);    //Send Slave Address & R/W Bit
                        I2C_Ctrl(0,0,0,0);                //Clear all I2C flag
                        break;

                case 0x84:    //MACTFlag+ACKFlag
                              //Slave A + W has been transmitted. ACK has been received.
                        I2C_SendData(I2C_Sendbuf[I2C_DataTxIndex++]);    //Send Data to Slave
                        I2C_Ctrl(0,0,0,0);          //Clear all I2C flag
                        break;

                case 0x80: //MACTFlag
                             //Slave A + W has been transmitted. ACK has been received.
                             I2C_SendData(I2C_Sendbuf[I2C_DataTxIndex++]);    //Send Data to Slave
                        I2C_Ctrl(0,0,0,0);          //Clear all I2C flag
                        break;

                case 0x30:

                        I2C_Ctrl(0,0,0,0);          //Clear all I2C flag
                        I2C_EndFlag=1;
                        break;

                case 0x8C:    //MACTFlag+DFFlag+ACKFlag
                             //DATA has been transmitted and ACK has been received
                        if(I2C_DataTxIndex<I2C_DataTxLen)
                        {
                          I2C_SendData(I2C_Sendbuf[I2C_DataTxIndex++]);    //Send Data to Slave
                          I2C_Ctrl(0,0,0,0);       // Clear all I2C flag
                        }
                        else
                        {
                                if(I2C_RW == I2C_WRITE)
                                {
                                     I2C_Ctrl(0,1,0,0);    //I2C as master sends STOP signal
                                     I2C_EndFlag=1;
                                }
                                else if(I2C_RW == I2C_READ)
                                I2C_Ctrl(1,0,0,0);    //I2C as master sends START signal
                                I2C_DataTxIndex=0;
                        }
                        break;

                case 0x88:    //MACTFlag+DFFlag
```

```
                //DATA has been transmitted and NACK has been received
            I2C_Ctrl(0,1,0,0);      //I2C as master sends STOP signal
            I2C_DataTxIndex=0;
            I2C_EndFlag=1;
            break;

        case 0xB0:
                //A repeated START has been transmitted.
            I2C_SendData(I2C_TARGET | I2C_READ);   //Send Slave Address & R/W Bit
            I2C_Ctrl(0,0,0,0);      //Clear all I2C flag
            break;

        case 0x94:   //MACTFlag+RWFlag+ACKFlag
                //Slave A + R has been transmitted. ACK has been received.
            if(I2C_DataRxLen>1)
            {
              I2C_Ctrl(0,0,0,1);   //Set ACK bit
            }else{
              I2C_Ctrl(0,0,0,0);   //Clear all I2C flag
            }
            break;

        case 0x9C:   //MACTFlag+RWFlag+DFFlag+ACKFlag
              //Data byte has been received. ACK has been transmitted.
            I2C_ReceiveDATA(I2C_Recbuf[I2C_DataRxIndex++]);
            if((I2C_DataRxLen-1)>I2C_DataRxIndex)
            {
              I2C_Ctrl(0,0,0,1);   //Set ACK bit
            }
            else
            {
              I2C_Ctrl(0,0,0,0);   //Clear all I2C flag
            }
            break;

        case 0x98:   //MACTFlag+RWFlag+DFFlag
              //Data byte has been received. NACK has been transmitted.
            I2C_ReceiveDATA(I2C_Recbuf[I2C_DataRxIndex++]);
            I2C_Ctrl(0,1,0,0);   //I2C as master sends STOP signal
            I2C_EndFlag=1;
            break;

        default:
            I2C_Ctrl(0,0,0,0);   //Clear all I2C flag
            I2C_EndFlag=1;
            break;
    }
      I2C_I2CINT_CLEAR();       //CLR I2C INT Flag
      I2C_I2CER_CLEAR();        //CLR I2C error Flag
      I2CIF_ClearFlag();                  //CLR I2CIF
}

if(I2CERIF_IsFlag())   //Get I2C Error Interrupt Flag
{
    I2C_EndFlag=1;
      I2C_I2CINT_CLEAR();       //CLR I2C INT Flag
      I2C_I2CER_CLEAR();        //CLR I2C error Flag
```

```
                I2CERIF_ClearFlag();            //CLR I2CERIF
               I2C_Ctrl(0,0,0,0);               //Clear all I2C flag
        }

}
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* System Init Function                                                      */
/*---------------------------------------------------------------------------*/
void SysInit(void)
{
        //PT1.0 HW Reset Function Ensable
       PT10_HWResetEnable();

        //CPU CLK SELECT
        CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );

}

/*---------------------------------------------------------------------------*/
/* I2C Init Function                                                         */
/*---------------------------------------------------------------------------*/
void I2CInit(void)
{
        I2C_Open(50);     //I2C Open
        I2C_SetIOPin(2); // SCL-> PT3.0    SDA->PT3.1
}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
        for(;num>0;num--)
       __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

15. 其他 IP(Power)

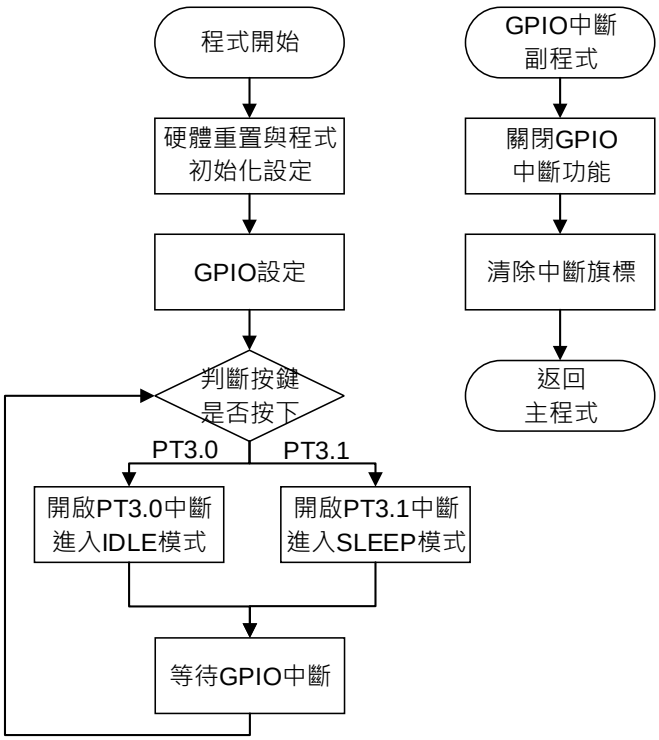
15.1. 範例名稱

Demo_PWR

15.2. 範例說明

(1) HY17M24 進入 Sleep 與 Idle Mode 範例程式
(2) 開啟 PT1.0 硬體重置功能及設置系統工作頻率。
(3) 關閉 BOR2 與 VDDA 電壓
(4) 電後按下 PT3.0 進入 Sleep Mode，再按下 PT3.0 喚醒。
(5) 電後按下 PT3.1 進入 Idle Mode，再按下 PT3.1 喚醒。
(6) 此範例程式搭配 HY17M24 硬體開發板，可測量到 Sleep 耗電流約為 0.25uA, Idle 耗電流約為 1uA.

15.3. 軟體流程



15.4. 程式碼

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * Demo_PWR.c                                                                *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *            HY17M24                                                        *
 *--------------------                                                       *
 *                              |                                            *
 *                              |                                            *
 *                PT3.0 |->button1                                          *
 *                PT3.1 |->button2                                          *
 *                              |                                            *
 *                              |                                            *
 *-------------------                                                        *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *
 *                                                                           *
 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <PWR.h>
#include <GPIO.h>
#include <RST.h>
#include <CLK.h>

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define Sleep_mode
#define Idle_mode
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void GPIOInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
```

```
/* Global CONSTANTS                                                    */
/*---------------------------------------------------------------------*/
unsigned char cks=0;
/*---------------------------------------------------------------------*/
/* Main Function                                                       */
/*---------------------------------------------------------------------*/
void main(void)
{
    //PT1.0 HW Reset Function Enable
    PT10_HWResetEnable();
////   CPU CLK Init            //
     CLK_OSCSelect(OSCS_LPO);
    CLK_HAODisable();
    CLK_CPUCKSelect(CPUS_DHSCK);

     GPIOInit();

     GIE_Enable();
     while(1)
     {
         if(GPIO_PT3GET(PT30_H) == 0)
         {
            Delay(20);        //debounce
          while(GPIO_PT3GET(PT30_H) == 0);      //wait PT30=0
            INTE30_Enable();      //PT30 INT Enable
            ENBOR2_Disable();
            PWR_BGRDisable();

            NOP();
            Idle();   //   IDLE mode Enable
            NOP();
         }

         if(GPIO_PT3GET(PT31_H) == 0)
         {
            Delay(20);        //debounce
          while(GPIO_PT3GET(PT31_H) == 0);      //wait PT31=0
            INTE31_Enable();      //PT31 INT Enable
            ENBOR2_Disable();
            PWR_BGRDisable();

            NOP();
            Sleep();   //   Sleep mode Enable
            NOP();
         }
     }
```

```
}


/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
    if(INTF30_IsFlag())
    {
        Delay(20);          //debounce
        while(GPIO_PT3GET(PT30_H) == 0);    //wait PT30=0
        INTE30_Disable();
    }

    if(INTF31_IsFlag())
    {
        Delay(20);          //debounce
        while(GPIO_PT3GET(PT31_H) == 0);    //wait PT31=0
        INTE31_Disable();
    }
    INTF30_ClearFlag();
    INTF31_ClearFlag();
}
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/*          GPIO Init function                                               */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
    GPIO_PT3InputEnable(IN30_H | IN31_H);           //PT3.0/3.1 Digital mode
    GPIO_PT3SETPU(PU30_H | PU31_H);
    GPIO_PT3InputMode(TC30_H | TC31_H);             //PT3.0/3.1 Input mode
    INTG30_Edgefall();          //PT30 INT triggering conditions 1->0
    INTE30_Disable();           //PT30 INT Disable
   INTG31_Edgefall();       //PT31 INT triggering conditions 1->0
    INTE31_Disable();           //PT31 INT Disable
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
```

```
    __asm__("NOP");
}

/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

16. 修訂記錄

以下描述本文件差異較大的地方，而標點符號與字形的改變不在此描述範圍。

文件版次	頁次	日期	摘要
V01	All	2020/04/28	初版發行
V02	All	2021/07/30	修正 DAC 範例程式 移除各範例程式 CCOPT_Enable()
V03	All	2021/09/15	更改 HAOM 參數名稱 HAOM_1843KHZ、HAOM_4147KHZ、HAOM_8755KHZ、HAOM_17510KHZ
V04	8	2023/02/22	1. 修改 MTP/EEPROM 的燒錄次數 修改前 MTP 燒錄次數 1K 次 修改後 MTP 燒錄次數 100 次 修改前 EEPROM 燒錄次數 30K 次 修改後 EEPROM 燒錄次數 3K 次