HY17M28

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 HY17M28，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

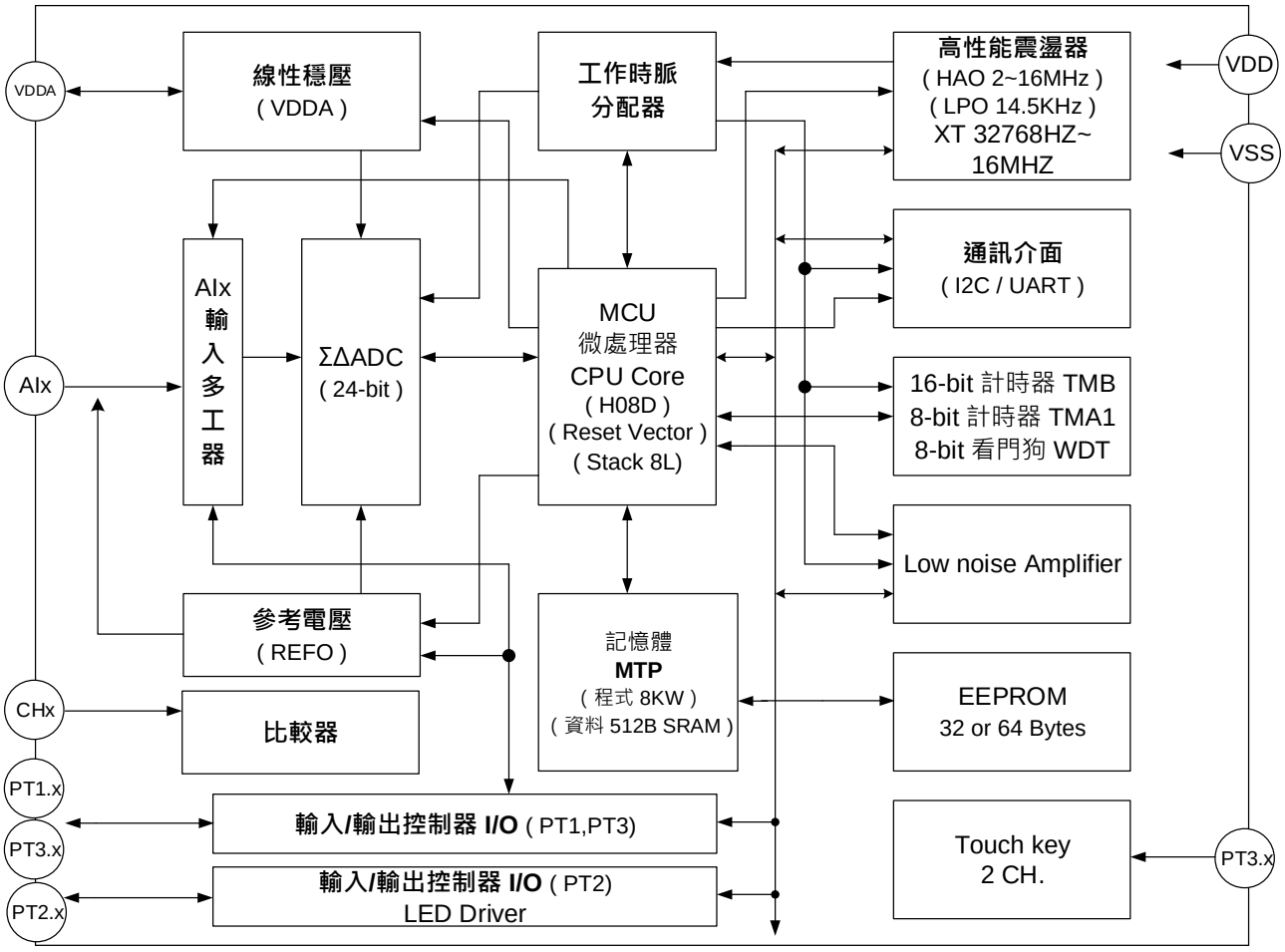
(2)類比 IP：ADC/Low noise Amplifier/CMP/Touch Key

(3)通訊 IP：Hardware UART/ Hardware I2C

(4)其他 IP：Power Management

2. 晶片說明

HY17M28 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 32768Hz~16MHz 石英震盪器或內部 2MHz~16MHz 高精度 RC 震盪器，

(04) 8K words MTP Program Memory (讀寫次數：100 cycles)及 32 bytes (讀寫次數：3,000 cycles)或 64 bytes (讀寫次數：700 cycles) EEPROM Data Memory。

(05)資料記憶體 512 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 Low noise PGA Amplifier 運算放大器。

(09)8 個 GPIO 支援定電流控制線路

(10)2 個 Touch Key 功能

(11)支援 Bootloader 線上更新(ISP)功能

(12)8-bit Timer A

(13)16-bit Timer B 模組具 PWM 波形產生功能

(14)硬體串列通訊 I2C/UART 模組

3. 數位 IP(TMA)

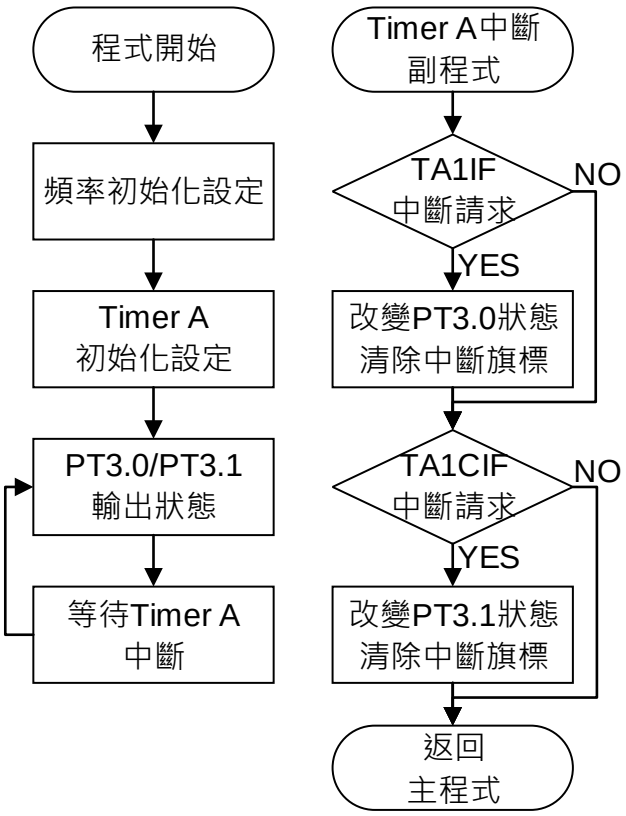
3.1. 範例名稱

TMA

3.2. 範例說明

(1) Timer A 使用方式與說明。
(2) 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。
(3) 每進一次 TA1 中斷，代表 TMA1R 加一，並改變 PT3.0 輸出狀態觀察變化速度。
(4) 每進一次 TA1C 中斷，代表 TMA1R 計數值等於 TMA1C，改變 PT3.1 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_HY17M28_4M
/*******************************************************************************
 * TMA_Demo                                                                    *
 * --------------------------------------------------------------------------- *
 * Copyright 2021 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *            HY17M28                                                          *
 *--------------------                                                         *
 *                          |                                                  *
 *                    PT2.0|->_|-|_ TMA                                        *
 *                          |                                                  *
 *                    PT2.1|->_|-|_ TMAC                                       *
 *                          |                                                  *
 *--------------------                                                         *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M28                                                            *
 * Project Name : TMA                                                          *
 * Created Date : 2022/3/1 By Cyril                                            *
 *                                                                             *
 *******************************************************************************/

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

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif
```

```
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void TMAInit(void);
void CLOCKInit(void);
void GPIOInit(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned int TMA1R_Count;
unsigned int TMA1C_Count;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    CLOCKInit();
    GPIOInit();
    TMAInit();
    TMA1R_Count=0;
    TMA1C_Count=0;

    GIE_Enable();

    while(1);

}

/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
     if(TA1IF_IsFlag())          //TA1IF=1,TMA1R++
    {
        PT3= PT3^PT30_MSK;    //PT3.0 toggle
        TMA1R_Count++;
        TA1IF_ClearFlag();
    }

     if(TA1CIF_IsFlag())            //TMA1C=TMA1R ,then TA1CIF=1
    {
        PT3= PT3^PT31_MSK;    //PT3.1 toggle
        TMA1C_Count++;
        TMA1_ClearTMA1R();    //clear TMA1R
        TA1CIF_ClearFlag();
     }
}

/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
  CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif
```

```
#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif
   CLK_DMSCKSelect(DMS_DHSCKDIV4);    //DMS_CK= HAO/1/4
}

/*---------------------------------------------------------------------------*/
/* TMA Init Function                                                         */
/*---------------------------------------------------------------------------*/
void TMAInit(void)
{
   TMA1_Open(TMAS_DMSCK,DTMA_TMACKDIV2,20); //DTMA1_CK=TMA1_CK/256/2
                                            //TA1IF=0.533(msec)
                                            //TA1CIF=TA1IF*20
   TA1CIF_ClearFlag();
   TA1CIE_Enable();
   TA1IF_ClearFlag();
   TA1IE_Enable();
}

/*---------------------------------------------------------------------------*/
/* GPIO Init Function                                                        */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
   GPIO_PT3OutputMode(PT30);
   GPIO_PT3DigitalEnable(PT30);
   GPIO_PT3OutputLow(PT30);

   GPIO_PT3OutputMode(PT31);
   GPIO_PT3DigitalEnable(PT31);
   GPIO_PT3OutputLow(PT31);
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

4. 數位 IP(TMB)

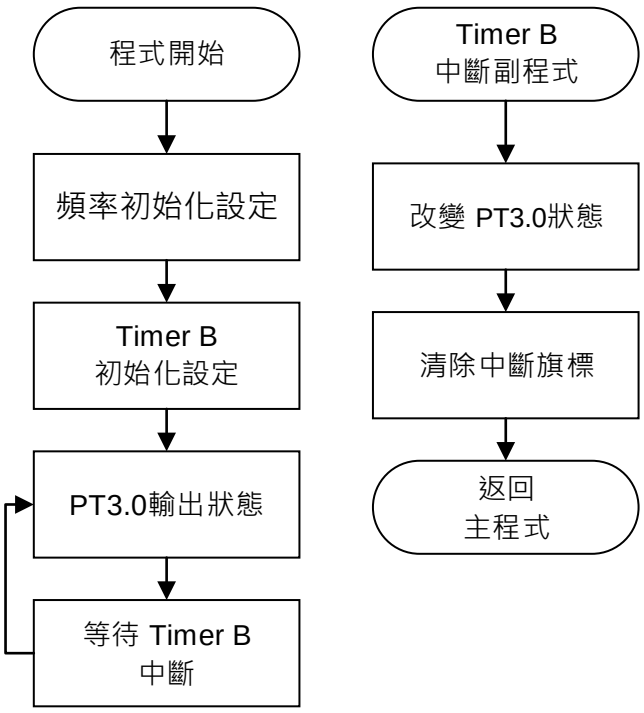
4.1. 範例名稱

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 PT3.0 輸出狀態，觀察轉換速度。

4.3. 軟體流程



4.4. 程式碼

```
#define   USE_HY17M28_2M
/*******************************************************************************
 * TMB_Demo.c                                                                  *
 * --------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *            HY17M28                                                          *
 *--------------------                                                         *
 *                          |                                                  *
 *                    PT2.0|->_|-|_ TMB                                        *
 *                          |                                                  *
 *--------------------                                                         *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M28                                                            *
 * Project Name : TMB                                                          *
 * Created Date : 2022/3/1 BY Cyril                                            *
 *                                                                             *
 *******************************************************************************/

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

/*-----------------------------------------------------------------------------*/
/* DEFINITIONS                                                                 */
/*-----------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

#define mode_16bit
//#define mode_17bit
//#define mode_2_8bit
//#define mode_8_8bit
```

```
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void Delayms(unsigned int ms);
void CLOCKInit(void);
void TMBInit(void);
void GPIOInit(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
        CLOCKInit();
        TMBInit();
        GPIOInit();

        GIE_Enable();
        while(1);
}

/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
        if(TB1IF_IsFlag())          //TB1R=TB1C ,TMBIF=1
        {
                PT3= PT3^PT30_MSK;    //PT3.0 Inverted
                TB1IF_ClearFlag();
        }
}

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


void Delayms(unsigned int ms)
{
   unsigned char t;
   for(;ms>0;ms--)              //@HAO=1.843MHz
      for(t=114;t>0;t--)
      { __asm__("NOP"); }
}
```

```
/*---------------------------------------------------------------------------*/
/* TMB Init Function                                                         */
/*---------------------------------------------------------------------------*/
void TMBInit(void)                  //select TMB mode
{
#if defined(mode_16bit)
   TMB1_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_16bit ,TB1RT_LogicH );
   TB1C0Set(0x00ff);     //Set TMB count
#endif
#if defined(mode_17bit)
   TMB1_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_17bit ,TB1RT_LogicH );
   TB1C0Set(0x02ff);     //Set TMB count
#endif
#if defined(mode_2_8bit)
   TMB1_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_2_8bit ,TB1RT_LogicH );
   TB1C0Set(0x7fff);     //Set TMB count
#endif
#if defined(mode_8_8bit)
   TMB1_Open(TMBS_HSCK ,DTMB_TMBCKDIV2 ,TB1M_8_8bit ,TB1RT_LogicH );
   TB1C0Set(0x7fff);     //Set TMB count
#endif
   TB1Enable();
}

/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}

/*---------------------------------------------------------------------------*/
/* GPIO Init Function                                                        */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
   GPIO_PT3OutputMode(PT30);
   GPIO_PT3DigitalEnable(PT30);
   GPIO_PT3OutputLow(PT30);
```

```
}

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

5. 數位 IP(WDT)

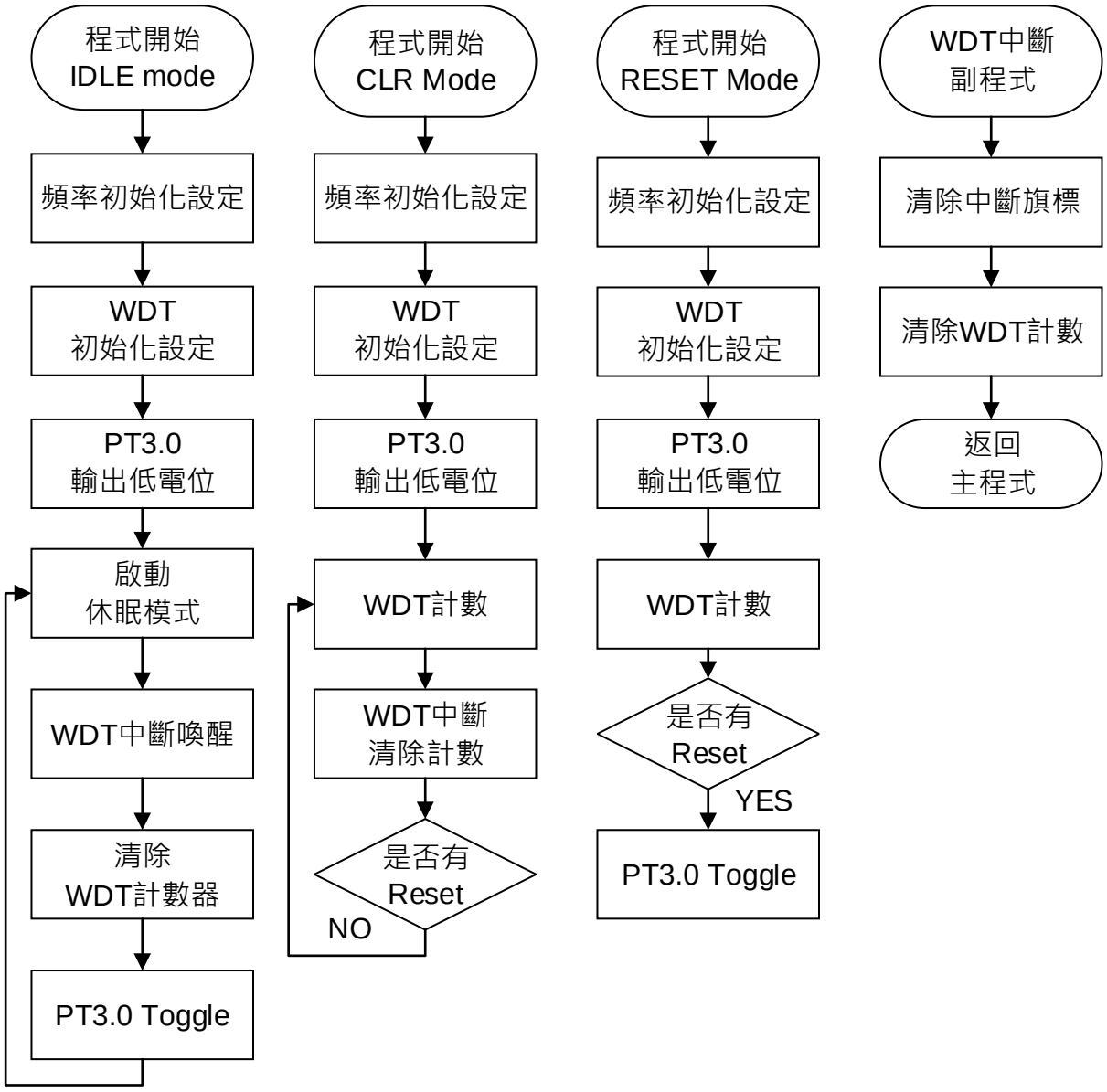
5.1. 範例名稱

WDT

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) 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 PT3.0 輸出高態，並於離開中斷前清除 WDT Count。
(6) 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。
(7) 在 CLR_Mode 下，當 WDT 計數值溢出時，系統將 Reset 並設置 TO 旗標為 1，如 TO 為”1”則改變 PT3.0 輸出狀態。
(8) 在 RESET_Mode 下，在主程序中加入清除 WDT 計數值，避免系統被重置。

5.3. 軟體流程


程式開始
IDLE mode
頻率初始化設定
WDT
初始化設定
PT3.0
輸出低電位
啟動
休眠模式
WDT中斷喚醒
清除
WDT計數器
PT3.0 Toggle
程式開始
CLR Mode
頻率初始化設定
WDT
初始化設定
PT3.0
輸出低電位
WDT計數
WDT中斷
清除計數
是否有
Reset
NO
程式開始
RESET Mode
頻率初始化設定
WDT
初始化設定
PT3.0
輸出低電位
WDT計數
是否有
Reset
YES
PT3.0 Toggle
WDT中斷
副程式
清除中斷旗標
清除WDT計數
返回
主程式

5.4. 程式碼

```
#define   USE_HY17M28_2M
/*******************************************************************************
 * WDT_Demo.c                                                                  *
 * ---------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 * Test_IDLE: IDLE mode. WDT count overflow, wake up from IDLE, WDTIF=1        *
 * Test_CLR : Normal mode. Clear WDT count, IC doesn't reset                   *
 * Test_NOP: Normal mode. Not clear WDT count,IC will reset, TO=1              *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M28                                                            *
 * Project Name : WDT                                                          *
 * Created Date : 2022/3/1 BY Cyril                                            *
 *                                                                             *
 *******************************************************************************/

/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <WDT.h>
#include <INT.h>
#include <RST.h>
#include <CLK.h>
#include <GPIO.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

#define Test_IDLE
//#define Test_CLR
//#define Test_RESET

/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
```

```
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void WDTInit(void);
void CLOCKInit(void);
void GPIOInit(void);
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    CLOCKInit();
    GPIOInit();
    WDTInit();
    GIE_Enable();


#ifdef Test_IDLE     //wake up Idle
    while(1)
    {
      WDT_WDTCKDivSelect(DWDT_WDTCKDIV64);
        Idle();
        NOP();
      PT3= PT3^PT30_MSK;    //PT3.0 toggle
     }
#endif

#ifdef Test_CLR     //clear WDT Count, won't reset
    while(1)
    {
      if(SYS_ReadWDT()!=0)    //if program reset, TO=1
      {
            SYS_ClearWDT();
            PT3= PT3^PT30_MSK;    //PT3.0 toggle
      }
      WDT_Clear();
    }
#endif

#ifdef Test_RESET     //nop, will reset
    while(1)
    {
      if(SYS_ReadWDT()!=0)    //if program reset, TO=1
      {
            SYS_ClearWDT();
            PT3= PT3^PT30_MSK;    //PT3.0 toggle
      }
      NOP();
    }
#endif

}

/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt    //WDT Reset
```

```
{
    WDTIF_ClearFlag();
    SYS_ClearIDLE();        //CLR IDLE Flag
    WDT_Clear();            //CLR WDT count.
}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
  CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
  CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
  CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
  CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}

/*---------------------------------------------------------------------------*/
/* WDT Init Function                                                         */
/*---------------------------------------------------------------------------*/
void WDTInit(void)
{
  WDT_Open(DWDT_WDTCKDIV2048);          //WDT Open
  WDT_Clear();                          // CLEAR WDT COUNT
  WDTIF_ClearFlag();
  WDTIE_Enable();
  WDT_Enable();
}

/*---------------------------------------------------------------------------*/
/* GPIO Init Function                                                        */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
  GPIO_PT3OutputMode(PT30);
  GPIO_PT3DigitalEnable(PT30);
  GPIO_PT3OutputLow(PT30);

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

6. 數位 IP(PWM)

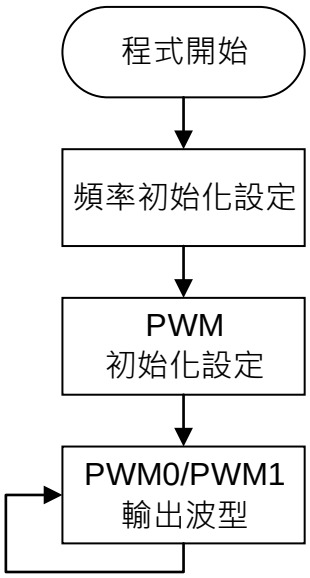
6.1. 範例名稱

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 來規劃腳位編譯執行 GPWM1_PT31, GPWM0_PT30。
(4) 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_HY17M28_2M
/*****************************************************************************
 * PWM_Demo.c                                                                *
 * ------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *    HY17M28                                                                *
 * -----------                                                               *
 *               |                                                           *
 *       PT3.0 |-->PWM0                                                      *
 *               |                                                           *
 *       PT3.1 |-->PWM1                                                      *
 *               |                                                           *
 * -----------                                                               *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M28                                                          *
 * Project Name : PWM                                                        *
 * Created Date : 2022/3/1 By Cyril                                          *
 *                                                                           *
 *****************************************************************************/

/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <TMR.h>
#include <CLK.h>
#include <INT.h>
#include <GPIO.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

////     Select PWM MODE              //
#define PWM1O
//#define PWM2O
//#define PWM3O
```

```
//#define PWM4O
//#define PWM5O
//#define PWM6O
//#define PWM7O

#define PWM1_Port    GPWM1_PT31
#define PWM0_Port    GPWM0_PT30
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void Delayms(unsigned int ms);
void CLOCKInit(void);
void PWMInit(void);

/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
        CLOCKInit();
        PWMInit();


        while(1);

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

/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}
```

```
/*---------------------------------------------------------------------------*/
/* PWM Init Function                                                         */
/*---------------------------------------------------------------------------*/
void PWMInit(void)
{
   //GPIO Init
   GPIO_GPWM0Set(PWM0_Port);
   GPIO_GPWM1Set(PWM1_Port);

//***** setting the timer B & PWM *********//
/*--------------------------------------PWM1O-----------------------------------------*/
#ifdef PWM1O
      TMB1_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
      TMB1_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
      TMB1_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
        TMB1_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
        TMB1_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
        TMB1_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
        TMB1_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(0x3FFF);      // TB1C0[15:0] is setting the PWM frequency.
                                                    // PWM Duty Cycle is always 50%
#endif
        TB1Enable();
}

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


void Delayms(unsigned int ms)
{
   unsigned char t;
   for(;ms>0;ms--)              //@HAO=1.843MHz
      for(t=114;t>0;t--)
      { __asm__("NOP"); }
}

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

7. 數位 IP(BIE)

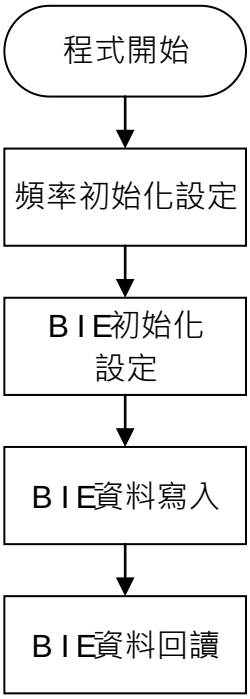
7.1. 範例名稱

Demo_BIE

7.2. 範例說明

(1) BIE 自我燒錄與讀取使用說明
(2) 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~EERD31/63。
(4) 執行 BIE 寫入功能，把存放於暫存器 EERD0~EERD31/63 之數據寫入 EEPROM。
(5) 清除 EERD0~EERD31/63 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~EERD31/63 檢視。

7.3. 軟體流程



7.4. 程式碼

```
#define   USE_HY17M28_2M
/*****************************************************************************
 * BIE_Demo                                                                  *
 * ------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M28                                                          *
 * Project Name : TMA                                                        *
 * Created Date : 2022/3/31 By Cyril                                         *
 *                                                                           *
 *****************************************************************************/

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


/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void);
void EERDWrite(void);
void EERDCLR(void);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
```

```
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    CLOCKInit();   //Operating BIE, CPU clock select HAO
    GIE_Disable();   //Disable the GIE interrupt brfore operating BIE

    BIE_EEPMode(EEMODE_64bytes);   //Select EEPROM mode
    EERDCLR();   //Clear EERD0~63

    EERDWrite();   //Write data to EERD
    BIE_DataWrite();   //Write EERD to BIE

    EERDCLR();   //Clear EERD0~63
    BIE_DataReload();   //Reload data to EERD
    NOP();

    GIE_Enable();
    while(1);

}

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

}

/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
  CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
  CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
  CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
  CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}

/*---------------------------------------------------------------------------*/
/* EEPROM Data Write Function                                                */
```

```
/*---------------------------------------------------------------------------*/
void EERDWrite(void)
{
   unsigned char eepcount=0;

   FSR0= &EERD0;
   for(eepcount=1;eepcount<=64;eepcount++)
   {
      POINC0= eepcount;
   }

}

/*---------------------------------------------------------------------------*/
/* EEPROM Data Clear Function                                                */
/*---------------------------------------------------------------------------*/
void EERDCLR(void)
{
   unsigned char eepcount;

   FSR0= &EERD0;
   for(eepcount=0;eepcount<=63;eepcount++)
   {
      POINC0=0;
   }

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

8. 類比 IP(ADC)

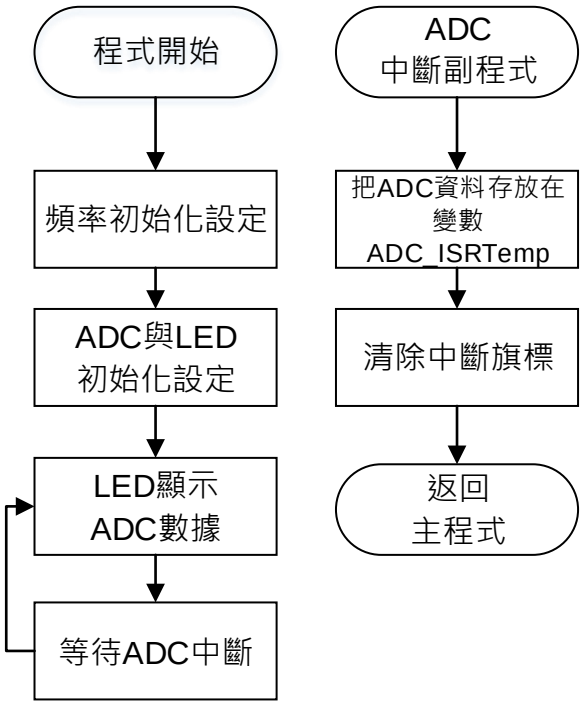
8.1. 範例名稱

ADC_LED

8.2. 範例說明

(1) 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輸出在 LED 上。

8.3. 軟體流程



8.4. 程式碼

```
#define   USE_HY17M28_2M
/*******************************************************************************
 * ------------------------------------------------------------------------- *
 * Copyright 2021 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M28                                                          *
 *******************************************************************************/

/*-----------------------------------------------------------------------------*/
/* Includes                                                                    */
/*-----------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <PWR.h>
#include <ADC.h>
#include <LED.h>
#include <TMR.h>
#include <RST.h>
#include "lcdTable.h"
/*-----------------------------------------------------------------------------*/
/* DEFINITIONS                                                                 */
/*-----------------------------------------------------------------------------*/
//#define ADC_ChopperMode

#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif
/*-----------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                         */
/*-----------------------------------------------------------------------------*/
void Delay(unsigned int del);
void CLOCKInit(void);
void LED_Display_Init(void);
void LED_Display_show(signed long DATA);
void LED_Display_h(unsigned char ledx, unsigned char On_OFF);
void ADCInit(void);
/*-----------------------------------------------------------------------------*/
/* Global CONSTANTS                                                            */
/*-----------------------------------------------------------------------------*/
```

```
unsigned char LED[8];
unsigned char LEDScan;

volatile typedef union _Flag
{
 unsigned int   _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned Reserved:15;
   };
} Flag;
Flag     Flagbits;

//signed long
//range 8388607~-8388608 (dec)
//range 0x007F FFFF ~ 0xFF80 0000 (hex)
signed long   ADC_ISRTemp;
signed long   ADCData1;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
   CLOCKInit();
   LED_Display_Init();
   ADCInit();
   GIE_Enable();

   while(1)
   {
      if(Flagbits.b_ADCdone==1)
      {
         Flagbits.b_ADCdone=0;
         LED_Display_show(ADC_ISRTemp);
         Delay(10000);
      }
   }

}
/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
      //TMA Event
      //A21023 V01 version
        if(TA1CIF_IsFlag())
        {
               TA1CIF_ClearFlag();
               TMA1R=0;
               LEDScan=LEDScan>>1;
               if(LEDScan==0)
               {LEDScan=0x80;}

               switch(LEDScan)
               {
```

```
        case 0x01:
        {
                PT2= 0;
                TRISC2=LED[0] | 0x01;
                PT2=~(LED[0] & 0xFE);
                break;
        }
        case 0x02:
        {
                PT2= 0;
                TRISC2=LED[1] | 0x02;
                PT2=~(LED[1]& 0xFD);
                break;
        }
        case 0x04:
        {
                PT2= 0;
                TRISC2=LED[2] | 0x04;
                PT2=~(LED[2] & 0xFB);
                break;
        }
        case 0x08:
        {
                PT2= 0;
                TRISC2=LED[3] | 0x08;
                PT2=~(LED[3]& 0xF7);
                break;
        }
        case 0x10:
        {
                PT2= 0;
                TRISC2=LED[4] | 0x10;
                PT2=~(LED[4]& 0xEF);
                break;
        }
        case 0x20:
        {
                PT2= 0;
                TRISC2=LED[5] | 0x20;
                PT2=~(LED[5]& 0xDF);
                break;
        }
        case 0x40:
        {
                PT2= 0;
                TRISC2=LED[6] | 0x40;
                PT2=~(LED[6]& 0xBF);
                break;
        }
        case 0x80:
        {
                PT2= 0;
                TRISC2=LED[7] | 0x80;
                PT2=~(LED[7]& 0x7F);
                break;
        }
}
```

```
        }

    //ADC Event
    if(ADCIF_IsFlag())
    {
       ADCIF_ClearFlag();
       ADC_ISRTemp=ADCR; //ADC_GetData();
       Flagbits.b_ADCdone=1;
    }

}
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}
/*---------------------------------------------------------------------------*/
/* ADC Init Function                                                         */
/*---------------------------------------------------------------------------*/
void ADCInit(void)
{
   //==========Setup Power====================================
   PWR_VDDAOpen(LDOC_2V4);    //refer to the Datasheet for LDOC typical voltage

   //==========Setup ADC=====================================
#if defined(HAO_2MHZ)
   //HAO=2MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV2,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_6
5536);
#endif

#if defined(HAO_4MHZ)
   //HAO=4MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV4,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_6
5536);
#endif

#if defined(HAO_8MHZ)
```

```
   //HAO=8MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV8,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_6
5536);
#endif

#if defined(HAO_16MHZ)
   //HAO=15.667MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV16,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_
65536);
#endif

#if defined (ADC_ChopperMode)
   CSFON_Enable();
   ADC_ENINXCH_Enable();
   ADC_DAFM_CHOPPER();
   ADC_ENCH_Enable();
   ADC_Enable();
   ADC_CMFREnable();        //CMFR=1, Comb Filter Reset
#endif

}

/*---------------------------------------------------------------------------*/
/* LED Init Function                                                         */
/*---------------------------------------------------------------------------*/
void LED_Display_Init(void)
{
   LEDSetting(CCLevel_15mA);        //Set PT2 constant current 15mA
   LEDScan= 0;

   TMA1_Open(TMAS_DMSCK,DTMA_TMACKDIV2,2);     //Enable TMA, set LED scan rate=2
   TA1IE_Disable();       //Disable TMA1 Interrupt
   TA1CIE_Enable();       //Enable TMA1C Interrupt

}

/*---------------------------------------------------------------------------*/
/* LED_Display_show Function                                                 */
/*---------------------------------------------------------------------------*/
void LED_Display_show(signed long DATA)
{
   //display range +8388607~-8388608 (dec)
   //display range 0x7FFFFF~ 0x800000 (hex)
   ADCData1=DATA;
   if((ADCData1<0)||(ADCData1>0x80000000))    // plus-minus sign judgment
   {
          ADCData1=~ADCData1;
          ADCData1++;
        LED[7]=LED[7] | 0x40;    //UartTxBuffer[0]='-';
   }
   else
   {
        if(LED[7]==0x40)
        {LED[7]=LED[7] - 0x40;}    //UartTxBuffer[0]='+';
   }
```

```
    DATA=ADCData1;

    ADCData1= seg[DATA%10];
    LED[6]= GetDisplay(ADCData1,7);
    DATA /=10;

    ADCData1= seg[DATA%10];
    LED[5]= GetDisplay(ADCData1,6);
    DATA /=10;

    ADCData1= seg[DATA%10];
    LED[4]= GetDisplay(ADCData1,5);
    DATA /=10;

    ADCData1= seg[DATA%10];
    LED[3]= GetDisplay(ADCData1,4);
    DATA /=10;

    ADCData1= seg[DATA%10];
    LED[2]= GetDisplay(ADCData1,3);
    DATA /=10;

    ADCData1= seg[DATA%10];
    LED[1]= GetDisplay(ADCData1,2);
    DATA /=10;

    ADCData1= seg[DATA%10];
    LED[0]= GetDisplay(ADCData1,1);

}

/*---------------------------------------------------------------------------*/
/* LED_Display_h Function                                                    */
/*---------------------------------------------------------------------------*/
void LED_Display_h(unsigned char ledx, unsigned char On_OFF)
{

    unsigned char LEDScan_h, On_OFF_h ;
    LEDScan_h=ledx;
    On_OFF_h=On_OFF;

    switch(LEDScan_h)
    {
       case 0x1:
       {
          if(On_OFF_h==1)
          {LED[7]=LED[7] | 0x01;}
          if(On_OFF_h==0)
          {LED[7]=LED[7] -0x01;}
          break;
       }
       case 0x2:
       {
          if(On_OFF_h==1)
          {LED[7]=LED[7] | 0x02;}
          if(On_OFF_h==0)
          {LED[7]=LED[7] - 0x02;}
```

```
            break;
        }
        case 0x3:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x04;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x04;}
            break;
        }
        case 0x4:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x08;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x08;}
            break;
        }
        case 0x5:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x10;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x10;}
            break;
        }
        case 0x6:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x20;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x20;}
            break;
        }
        case 0x7:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x40;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x40;}
            break;
        }
    }

}

/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int del)
{
    while(--del)
    NOP();
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

9. 類比 IP(MFC)

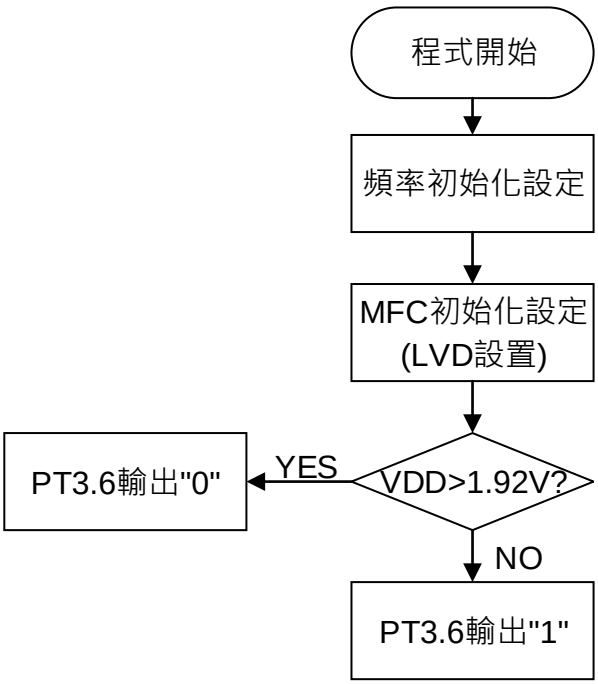
9.1. 範例名稱

MFC

9.2. 範例說明

(1) MFC 使用方式說明
(2) 設置系統工作頻率。
(3) 透過 MFC 相關設定，可以實現遲滯控制。
(4) MFC 初始設置包含, 設置類比電源 VDDA，設置 MFC 的輸入端為 RLO-1V2，設置 MFC 內建多節點電阻分壓值(RLO)以及 RLO 電壓源為 VDD。
(5) 將 MFC 作為低電壓偵測(LVD)，當 VDD 經過電阻分壓後大於 CH2，則 PT3.5 輸出”0”。反之當 VDD 經過電阻分壓後小於 CH2，則 PT3.5 輸出”1”。

9.3. 軟體流程



9.4. 程式碼

```
#define   USE_HY17M28_2M
/*******************************************************************************
 * MFC_Demo.c                                                                  *
 * --------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                             *
 * http://www.hycontek.com/                                                    *
 *                                                                             *
 * Program Description:                                                        *
 *                                                                             *
 *            HY17M28                                                          *
 *--------------------                                                         *
 *                          |                                                  *
 *                    PT3.0|->Output                                           *
 *                          |                                                  *
 *                    PT3.1|<-Input                                           *
 *                          |                                                  *
 *--------------------                                                         *
 *                                                                             *
 * For External Input                                                          *
 * IC Body: HY17M28                                                            *
 * Project Name : MFC                                                          *
 * Created Date : 2022/3/1 BY Cyril                                            *
 *                                                                             *
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <GPIO.h>
#include <CLK.h>
#include <PWR.h>
#include <CMP.h>

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void Delayms(unsigned int ms);
```

```
void CMP_LVDInit(void);
void CMP_HysteresisInit(void);
void CLOCKInit(void);
void GPIOInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
      CMP_LVDInit();
      //CMP_HysteresisInit();

        GIE_Enable();
        while(1);
}

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

}
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}

/*---------------------------------------------------------------------------*/
/* LVD Mode Init Function                                                    */
/*---------------------------------------------------------------------------*/
```

```
void CMP_LVDInit(void)
{
   CMP_RLOSet(CPRH_VDD,CPRL_OPEN,CPDA_26DIV32,CPDM_DISABLE );     //RLO=
VDD*((26+20)/(22.5+32+20))

   //Set CPDA[4:0]
   //CPDA_6DIV32, LVD=3.39V
   //CPDA_7DIV32, LVD=3.27V
   //CPDA_8DIV32, LVD=3.15V
   //CPDA_9DIV32, LVD=3.04V
   //CPDA_10DIV32, LVD=2.94V
   //CPDA_11DIV32, LVD=2.85V
   //CPDA_12DIV32, LVD=2.76V
   //CPDA_13DIV32, LVD=2.67V
   //CPDA_14DIV32, LVD=2.59V
   //CPDA_15DIV32, LVD=2.52V
   //CPDA_16DIV32, LVD=2.45V
   //CPDA_17DIV32, LVD=2.38V
   //CPDA_18DIV32, LVD=2.32V
   //CPDA_19DIV32, LVD=2.26V
   //CPDA_20DIV32, LVD=2.21V
   //CPDA_21DIV32, LVD=2.15V
   //CPDA_22DIV32, LVD=2.10V
   //CPDA_23DIV32, LVD=2.05V
   //CPDA_24DIV32, LVD=2.01V
   //CPDA_25DIV32, LVD=1.96V
   //CPDA_26DIV32, LVD=1.92V

   CMP_Open(CPPS_1V2,CPNS_RLO,CMPHS_NORMAL,CPOR_NORMAL,ENRCC_PT35);
   //LVD=1.92V (When VDD >1.92V PT36=Low, When VDD<1.92V PT36=High)
}

/*---------------------------------------------------------------------------*/
/* HysteresisInit Mode Init Function                                         */
/*---------------------------------------------------------------------------*/
void CMP_HysteresisInit(void)
{
        PWR_VDDAOpen(LDOC_2V4);    //refer to the Datasheet for LDOC typical voltage
        CMP_RLOSet(CPRH_VDDA,CPRL_SHORT,CPDA_24DIV32,0x08);
      //CPDM[4:0]=01000
      //CPDA[4:0]=11000, RLO= VDDA*(24/32) <-switch over-> CPDA[4:0]=10000, RLO= VDDA*(16/32)
      CMP_Open(CPPS_CH2,CPNS_RLO,CMPHS_NORMAL,CPOR_NORMAL,ENRCC_PT35);// When PT3.7<1.2V
PT35=Low, When PT3.7 >1.8V PT35=High,

}

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


void Delayms(unsigned int ms)
```

```
{
   unsigned char t;
   for(;ms>0;ms--)              //@HAO=1.843MHz
      for(t=114;t>0;t--)
      { __asm__("NOP"); }
}


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

10. 類比 IP(Touch Key)

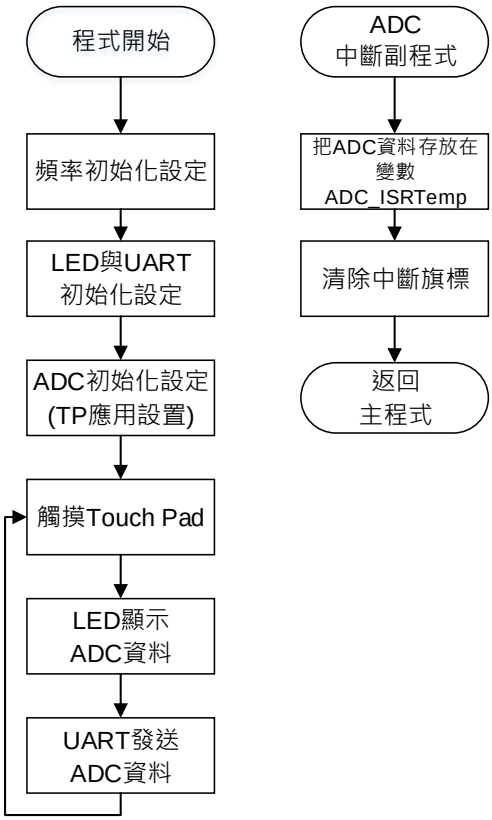
10.1. 範例名稱

TK1_UART_LED

10.2. 範例說明

(1) HY17M28 Touch Pad 搭配 UART 傳輸以及 LED 顯示的範例程式
(2) 設置系統工作頻率、UART 初始化以及 LED 初始化。
(3) ADC 初始化設置為 TP 應用通道。
(4) 程式開始執行，UART 會一直發送 ADC 資料；當觸碰 Touch Pad，LED 顯示數值加一

10.3. 軟體流程



10.4. 程式碼

```
#define   USE_HY17M28_2M
/*****************************************************************************
 * ------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M28                                                          *
 *****************************************************************************/

/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <RST.h>
#include <UART.h>
#include <PWR.h>
#include <ADC.h>
#include <TMR.h>
#include <GPIO.h>
#include <LED.h>
#include <TP.h>
#include "lcdTable.h"
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define ADC_ChopperMode

#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

#define UART_Port GTX_PT13

#define   Uart_RX_BufferSize 10
#define   Uart_TX_BufferSize 10
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int del);
void CLOCKInit(void);
void LED_Display_Init(void);
void LED_Display_show(signed long DATA);
```

```
void LED_Display_h(unsigned char ledx, unsigned char On_OFF);
void ADCInit(void);
void UARTInit(void);
void SendUART(signed long    data);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char LED[8];
unsigned char LEDScan;

volatile typedef union _Flag
{
 unsigned int    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
      unsigned Reserved:13;
   };
} Flag;

//signed long
//range 8388607~-8388608 (dec)
//range 0x007F FFFF ~ 0xFF80 0000 (hex)
signed long    ADC_ISRTemp;
signed long    ADCData1;
Flag     Flagbits;
unsigned char UartTxBuffer[Uart_TX_BufferSize]={0};
unsigned char UartRxBuffer[Uart_RX_BufferSize]={0};
unsigned char UartTxIndex,UartTxLength;
unsigned char UartRxIndex,UartRxLength;


signed long ADC_NO_TOUCH_TK1=-200000;
signed long ADC_TK1=0;
signed long ADC_TK1_Thresould=-300000;
unsigned int ADC_TK1_COUNT=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

   //Init data
   UartTxIndex=0;
   UartRxIndex=0;
   Flagbits.b_ADCdone=0;
   Flagbits.b_UART_TxDone=0;
   Flagbits.b_UART_RxDone=0;

   //PT20 LED setting
   LED_Display_Init();

   //CLOCK+ADC+UART Init
   CLOCKInit();
   ADCInit();
```

```
    UARTInit();
    Flagbits.b_UART_TxDone=1; //Avoid UART Enable error on first data

    GIE_Enable();

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

      ADC_INT_Disable();
      ADC_TK1=ADC_ISRTemp;
      if((ADC_TK1-ADC_NO_TOUCH_TK1)<ADC_TK1_Thresould)
      {
         ADC_TK1_COUNT++;
      }
      LED_Display_show(ADC_TK1_COUNT);
      Delay(10000);
      SendUART(ADC_TK1);
      ADC_INT_Enable();
    }

}
/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
      //TMA Event
        if(TA1CIF_IsFlag())
        {
                TA1CIF_ClearFlag();
                TMA1R=0;
                LEDScan=LEDScan>>1;
                if(LEDScan==0)
                {LEDScan=0x80;}

                switch(LEDScan)
                {
                        case 0x01:
                        {
                                PT2= 0;
                                TRISC2=LED[0] | 0x01;
                                PT2=~(LED[0] & 0xFE);
                                break;
                        }
                        case 0x02:
                        {
                                PT2= 0;
                                TRISC2=LED[1] | 0x02;
                                PT2=~(LED[1]& 0xFD);
                                break;
                        }
                        case 0x04:
                        {
                                PT2= 0;
```

```
                TRISC2=LED[2] | 0x04;
                PT2=~(LED[2] & 0xFB);
                break;
            }
            case 0x08:
            {
                PT2= 0;
                TRISC2=LED[3] | 0x08;
                PT2=~(LED[3]& 0xF7);
                break;
            }
            case 0x10:
            {
                PT2= 0;
                TRISC2=LED[4] | 0x10;
                PT2=~(LED[4]& 0xEF);
                break;
            }
            case 0x20:
            {
                PT2= 0;
                TRISC2=LED[5] | 0x20;
                PT2=~(LED[5]& 0xDF);
                break;
            }
            case 0x40:
            {
                PT2= 0;
                TRISC2=LED[6] | 0x40;
                PT2=~(LED[6]& 0xBF);
                break;
            }
            case 0x80:
            {
                PT2= 0;
                TRISC2=LED[7] | 0x80;
                PT2=~(LED[7]& 0x7F);
                break;
            }
        }
    }

//ADC Event
if(ADC_INT_IsFlag())
{
   ADC_INT_ClearFlag();
   ADC_ISRTemp=ADCR; //ADC_GetData();
   Flagbits.b_ADCdone=1;
}

//UART0 RX Event
if(UART0_INT_RCIsFlag())
{
   UartRxBuffer[UartRxIndex]=RC0REG;
   UartRxIndex++;
   if(UartRxIndex>=Uart_RX_BufferSize)
   {
```

```
        UartRxIndex=0;
        Flagbits.b_UART_RxDone=1;
      }
      UART0_INT_RCClearFlag();
    }

    //UART0 TX Event
    if(UART0_INT_TXIsFlag())
    {
      if(Flagbits.b_UART_TxDone==0)
      {
        TX0R=UartTxBuffer[UartTxIndex++];
        UART0_INT_TXClearFlag();
        if(UartTxIndex>=UartTxLength)
        {
          UART0_INT_TXEnable();       //TXIE=1b
          UART0_INT_RCEnable();    //RCIE=1b
          Flagbits.b_UART_TxDone=1;
          UartTxIndex=0;
        }
      }
      if(Flagbits.b_UART_TxDone==1)
      {
        UART0_INT_TXDisable();       //TXIE=0b
        UART0_INT_RCEnable();    //RCIE=1b
        UART0_INT_TXClearFlag();
      }
    }


}
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}
/*---------------------------------------------------------------------------*/
/* ADC Init Function                                                         */
/*---------------------------------------------------------------------------*/
void ADCInit(void)
```

```
{
  //==========Setup Power====================================
  PWR_BGREnable();                //bandgap Vref Enable
  PWR_VDDAOpen(LDOC_2V4);   //VDDA Enable, refer to the Datasheet for LDOC typical voltage

  //==========Setup ADC ================================
#if defined(HAO_2MHZ)
  //HAO=2MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV2,INP_TKP,INN_TKN,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_
16384);
#endif

#if defined(HAO_4MHZ)
  //HAO=4MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV4,INP_TKP,INN_TKN,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_
16384);
#endif

#if defined(HAO_8MHZ)
  //HAO=8MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV8,INP_TKP,INN_TKN,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_
16384);
#endif

#if defined(HAO_16MHZ)
  //HAO=15.667MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV16,INP_TKP,INN_TKN,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR
_16384);
#endif


  //==========Setup TK1 ======================================
  GPIO_GTKTXSet(GTKTX_PT10);   //GTKTX=01b, TX1=PT10
  TP_CfbSelect(RxCfb_10p);      //用此設定 Rx_Cfb=10p, 手沒觸碰鎖住在-217599 (TX1/RX1)
  //TP_CfbSelect(RxCfb_40p);   //必須用此設定 Rx_Cfb=40p, 手沒觸碰在-38608 (TX1/RX1)
  TP_CKDivSelect(TPCLK_HAODIV2); //TP_CK= HAO/2
  TP_MeasureChSet(MRx_Single);   //Touch Key 1 Channel
  TP_MeasureOrder(RxMx_RX1);   //RxMx=RX1(PT35)
  TP_StabilizeTime(TxSettle_3Clock);   //TXSettle=3*TX Clock time
  TP_CommonVoltageSel(TKCMS_TKBuffer);   //TKCMS= Touch key common mode buffer
  GPIO_PT3SETDA(5);
  TP_Enable();   //Enable touch key


#ifdef ADC_ChopperMode
  CSFON_Enable();
  ADC_ENINXCH_Enable();
  ADC_DAFM_CHOPPER();
  ADC_ENCH_Enable();
#endif
  ADC_Enable();
  ADC_CMFREnable();       //CMFR=1, Comb Filter Reset
}
```

```
/*---------------------------------------------------------------------------*/
/* LED_Display_show Function                                                 */
/*---------------------------------------------------------------------------*/
void LED_Display_show(signed long DATA)
{
   //display range +8388607~-8388608 (dec)
   //display range 0x7FFFFF~ 0x800000 (hex)
   ADCData1=DATA;
   if((ADCData1<0)||(ADCData1>0x80000000))    // plus-minus sign judgment
   {
           ADCData1=~ADCData1;
           ADCData1++;
         LED[7]=LED[7] | 0x40;    //UartTxBuffer[0]='-';
   }
   else
   {
         if(LED[7]==0x40)
         {LED[7]=LED[7] - 0x40;}    //UartTxBuffer[0]='+';
   }
   DATA=ADCData1;

   ADCData1= seg[DATA%10];
   LED[6]= GetDisplay(ADCData1,7);
   DATA /=10;

   ADCData1= seg[DATA%10];
   LED[5]= GetDisplay(ADCData1,6);
   DATA /=10;

   ADCData1= seg[DATA%10];
   LED[4]= GetDisplay(ADCData1,5);
   DATA /=10;

   ADCData1= seg[DATA%10];
   LED[3]= GetDisplay(ADCData1,4);
   DATA /=10;

   ADCData1= seg[DATA%10];
   LED[2]= GetDisplay(ADCData1,3);
   DATA /=10;

   ADCData1= seg[DATA%10];
   LED[1]= GetDisplay(ADCData1,2);
   DATA /=10;

   ADCData1= seg[DATA%10];
   LED[0]= GetDisplay(ADCData1,1);

}

/*---------------------------------------------------------------------------*/
/* LED Init Function                                                         */
/*---------------------------------------------------------------------------*/
void LED_Display_Init(void)
{
   LEDSetting(CCLevel_15mA);         //Set PT2 constant current 15mA
```

```
    LEDScan= 0;

    TMA1_Open(TMAS_DMSCK,DTMA_TMACKDIV2,2);     //Enable TMA, set LED scan rate=2
    TA1IE_Disable();       //Disable TMA1 Interrupt
    TA1CIE_Enable();       //Enable TMA1C Interrupt

}
/*---------------------------------------------------------------------------*/
/* LED_Display_h Function                                                    */
/*---------------------------------------------------------------------------*/
void LED_Display_h(unsigned char ledx, unsigned char On_OFF)
{

    unsigned char LEDScan_h, On_OFF_h ;
    LEDScan_h=ledx;
    On_OFF_h=On_OFF;

    switch(LEDScan_h)
    {
      case 0x1:
      {
        if(On_OFF_h==1)
        {LED[7]=LED[7] | 0x01;}
        if(On_OFF_h==0)
        {LED[7]=LED[7] -0x01;}
        break;
      }
      case 0x2:
      {
        if(On_OFF_h==1)
        {LED[7]=LED[7] | 0x02;}
        if(On_OFF_h==0)
        {LED[7]=LED[7] - 0x02;}
        break;
      }
      case 0x3:
      {
        if(On_OFF_h==1)
        {LED[7]=LED[7] | 0x04;}
        if(On_OFF_h==0)
        {LED[7]=LED[7] - 0x04;}
        break;
      }
      case 0x4:
      {
        if(On_OFF_h==1)
        {LED[7]=LED[7] | 0x08;}
        if(On_OFF_h==0)
        {LED[7]=LED[7] - 0x08;}
        break;
      }
      case 0x5:
      {
        if(On_OFF_h==1)
        {LED[7]=LED[7] | 0x10;}
        if(On_OFF_h==0)
        {LED[7]=LED[7] - 0x10;}
```

```
            break;
        }
        case 0x6:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x20;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x20;}
            break;
        }
        case 0x7:
        {
            if(On_OFF_h==1)
            {LED[7]=LED[7] | 0x40;}
            if(On_OFF_h==0)
            {LED[7]=LED[7] - 0x40;}
            break;
        }
    }

}

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

/*---------------------------------------------------------------------------*/
/* UART Init Function                                                        */
/*---------------------------------------------------------------------------*/
void UARTInit(void)
{
    //GPIO Init
    GPIO_GTXSet(UART_Port);    //TX PT1.3 Rx PT1.4

#ifndef UART_ABD
        //BRGR[15:0]=((CPUCK/Baudrate)/4)-1
    #if defined(HAO_2MHZ)
        //((1843000/9600)/4)-1 =47, HAO=1.843M, baudrate=9600
        UART0_Open(47 ,8 ,PARITY_None);
    #endif
    #if defined(HAO_4MHZ)
        //((4147000/9600)/4)-1 =107, HAO=4.147M, baudrate=9600
        UART0_Open(107 ,8 ,PARITY_None);
    #endif
    #if defined(HAO_8MHZ)
        //((8755000/9600)/4)-1 =227, HAO=8.755M, baudrate=9600
        UART0_Open(227 ,8 ,PARITY_None);
    #endif
    #if defined(HAO_16MHZ)
        //((15667000/9600)/4)-1 =407, HAO=15.667M, baudrate=9600
        UART0_Open(407 ,8 ,PARITY_None);
    #endif
```

```
#else
   UART0_Open(0 ,8 ,PARITY_None);
   UART0_WUEDisable();                    // Wake-up disable
   UART0_ABDEnable();                     //Enable Auto Baudrate
   while((BA0CN & 0x01) == 1 );          //Wait for master send 055H
   UART0_INT_RCClearFlag();               //Clear RC interrupt flag

#endif
   TXIE_Enable();
   RCIE_Enable();
}

/*---------------------------------------------------------------------------*/
/* Function Name: void SendUART(int data)                                    */
/* Description    : UART send data.                                          */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void SendUART(signed long    data)
{
   //display range +8388607~-8388608 (dec)
   ADCData1=data;
   if((ADCData1<0)||(ADCData1>0x80000000))    // plus-minus sign judgment
   {
          ADCData1=~ADCData1;
          ADCData1++;
        UartTxBuffer[0]='-';
   }
   else
   {
      UartTxBuffer[0]='+';
   }
   UartTxBuffer[7]=ADCData1%10 | '0';    //unit
   ADCData1=ADCData1/10;
   UartTxBuffer[6]=ADCData1%10 | '0';    //unit
   ADCData1=ADCData1/10;
   UartTxBuffer[5]=ADCData1%10 | '0';    //ten
   ADCData1=ADCData1/10;
   UartTxBuffer[4]=ADCData1%10 | '0';    //hundred
   ADCData1=ADCData1/10;
   UartTxBuffer[3]=ADCData1%10 | '0';    //thousand
   ADCData1=ADCData1/10;
   UartTxBuffer[2]=ADCData1%10 | '0';    //ten thousand
   ADCData1=ADCData1/10;
   UartTxBuffer[1]=ADCData1%10 | '0';    //hundred thousand
   ADCData1=ADCData1/10;
   UartTxBuffer[8]='\r';
   UartTxBuffer[9]='\n';
   Flagbits.b_UART_TxDone=0;
   UartTxLength=10;
   UartTxIndex=0;
   TXIE_Enable();       //Enable UART Tx Interrupt;
   RCIE_Enable();     //Enable UART Rx Interrupt
   while(!Flagbits.b_UART_TxDone);    //If    Flagbits.b_UART_TxDone=DISABLE, stop at here
}
```

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

11. 通訊 IP(UART)

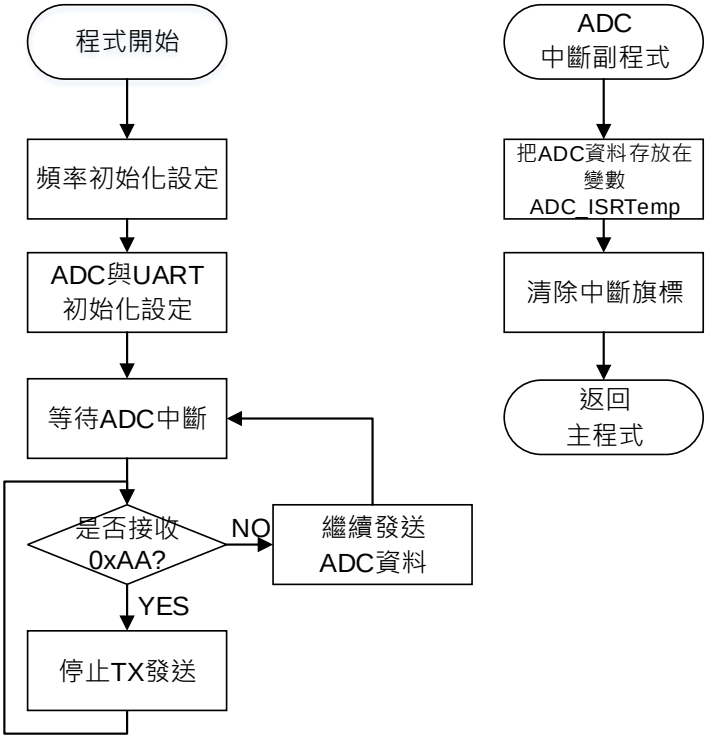
11.1. 範例名稱

UART

11.2. 範例說明

(5) HY17M28 使用 UART 做 TX 與 RX 傳輸範例程式
(6) 設置系統工作頻率以及 ADC 初始化設定。
(7) 開啟 UART 功能，設置 UART 相關設定，字串傳送速率 9600。
(8) 程式開始執行，UART 會一直送 ADC 資料，直到收到'0xAA'後會 TX 停止傳送。當收到字串不是"0xAA"， 則會繼續發送 ADC 資料。

11.3. 軟體流程



11.4. 程式碼

```
#define   USE_HY17M28_2M
/*****************************************************************************
 * UART_Demo.c                                                               *
 * ------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *            HY17M28                                                        *
 * ------------------                                                        *
 *                    |                                                      *
 *                PT3.0|->TX                                                 *
 *                PT3.1|<-RX                                                 *
 *                    |                                                      *
 *-------------------                                                        *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M28                                                          *
 * Project Name : UART                                                       *
 * Created Date : 2022/3/2 BY Cyril                                          *
 *                                                                           *
 *****************************************************************************/

/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <RST.h>
#include <UART.h>
#include <PWR.h>
#include <ADC.h>
#include <GPIO.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

#define UART_Port GTX_PT13

#define   Uart_RX_BufferSize 1
```

```
#define    Uart_TX_BufferSize 10

//#define UART_ABD
//#define ADC_ChopperMode
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void Delayms(unsigned int ms);
void SysInit(void);
void CLOCKInit(void);
void UART0Init(void);
void ADCInit(void);
void SendUART(signed long    data);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
volatile typedef union _Flag
{
  unsigned int    _byte;
    struct
    {
       unsigned b_ADCdone:1;
       unsigned b_UART_TxDone:1;
       unsigned b_UART_RxDone:1;
       unsigned Reserved:13;
    };
} Flag;
//signed long
//range 8388607~-8388608 (dec)
//range 0x007F FFFF ~ 0xFF80 0000 (hex)
signed long    ADC_ISRTemp;
signed long    ADC_ISRTemp_Buffer[8];
signed long    ADCData1;
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=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    //ADC_UART Demo
    UartTxIndex= 0;
    UartRxIndex= 0;
    Flagbits.b_ADCdone= 0;
    Flagbits.b_UART_TxDone= 0;
    Flagbits.b_UART_RxDone= 0;

    CLOCKInit();
    ADCInit();
    UART0Init();
    Flagbits.b_UART_TxDone=1;
    GIE_Enable();
```

```
    while(1)
    {

        if(Flagbits.b_UART_RxDone==1)
      {
            if( UartRxBuffer[0]==0xAA)          //If receive 0xAA, stop to send UART
            {
                  STOP_TO_SEND_UART= 1;
                  Flagbits.b_UART_RxDone= 0;
            }
            else
                  STOP_TO_SEND_UART=0;
      }


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

      if(Flagbits.b_UART_TxDone==1 && STOP_TO_SEND_UART==0)
      {
            ADC_INT_Disable();
            SendUART(ADC_ISRTemp);
            ADC_INT_Enable();
      }
    }

}

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

    NOP();

    //ADC Event
    if(ADC_INT_IsFlag())
    {
      ADC_INT_ClearFlag();
      ADC_ISRTemp=ADCR; //ADC_GetData();
      Flagbits.b_ADCdone=1;
    }

    //UART0 RX Event
    if(UART0_INT_RCIsFlag())
    {
      UartRxBuffer[UartRxIndex]=RC0REG;
      UartRxIndex++;
      if(UartRxIndex>=Uart_RX_BufferSize)
      {
        UartRxIndex=0;
        Flagbits.b_UART_RxDone=1;
      }
      UART0_INT_RCClearFlag();
```

```
  }

  //UART0 TX Event
  if(UART0_INT_TXIsFlag())
  {
    if(Flagbits.b_UART_TxDone==0)
    {
      TX0R=UartTxBuffer[UartTxIndex++];
      UART0_INT_TXClearFlag();
      if(UartTxIndex>=UartTxLength)
      {
        UART0_INT_TXEnable();      //TXIE=1b
        UART0_INT_RCEnable();   //RCIE=1b
        Flagbits.b_UART_TxDone=1;
        UartTxIndex=0;
      }
    }
    if(Flagbits.b_UART_TxDone==1)
    {
      UART0_INT_TXDisable();      //TXIE=0b
      UART0_INT_RCEnable();   //RCIE=1b
      UART0_INT_TXClearFlag();
    }
  }

}

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

/*---------------------------------------------------------------------------*/
/* UART Init Function                                                        */
/*---------------------------------------------------------------------------*/
void UART0Init(void)
{
  //GPIO Init
  GPIO_GTXSet(UART_Port);   //TX PT1.3 Rx PT1.4

#ifndef UART_ABD
    //BRGR[15:0]=((CPUCK/Baudrate)/4)-1
  #if defined(HAO_2MHZ)
    //((1843000/9600)/4)-1 =47, HAO=1.843M, baudrate=9600
    UART0_Open(47 ,8 ,PARITY_None);
  #endif
  #if defined(HAO_4MHZ)
    //((4147000/9600)/4)-1 =107, HAO=4.147M, baudrate=9600
    UART0_Open(107 ,8 ,PARITY_None);
  #endif
  #if defined(HAO_8MHZ)
    //((8755000/9600)/4)-1 =227, HAO=8.755M, baudrate=9600
    UART0_Open(227 ,8 ,PARITY_None);
  #endif
  #if defined(HAO_16MHZ)
    //((15667000/9600)/4)-1 =407, HAO=15.667M, baudrate=9600
    UART0_Open(407 ,8 ,PARITY_None);
  #endif
```

```
#else
   UART0_Open(0 ,8 ,PARITY_None);
   UART0_WUEDisable();                    // Wake-up disable
   UART0_ABDEnable();                     //Enable Auto Baudrate
   while((BA0CN & 0x01) == 1 );          //Wait for master send 055H
   UART0_INT_RCClearFlag();               //Clear RC interrupt flag

#endif
   TXIE_Enable();
   RCIE_Enable();
}
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}

/*---------------------------------------------------------------------------*/
/* ADC Init Function                                                         */
/*---------------------------------------------------------------------------*/
void ADCInit(void)
{
    //==========Setup Power====================================
   PWR_BGREnable();                  //bandgap Vref Enable
   PWR_VDDAOpen(LDOC_2V4);   //VDDA Enable, refer to the Datasheet for LDOC typical voltage
   PWR_VCMSSelect(VCMS_ACMint);
   PWR_ACMintOpen(SELVIN_V12);
   //==========Setup ADC Clock================================
#if defined(HAO_2MHZ)
   //HAO=2MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV2,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_6
5536);
#endif

#if defined(HAO_4MHZ)
   //HAO=4MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV4,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_6
5536);
```

```
#endif

#if defined(HAO_8MHZ)
   //HAO=8MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV8,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_6
5536);
#endif

#if defined(HAO_16MHZ)
   //HAO=15.667MHz, ADC_CK=1MHz

ADC_Open(DADC_DHSCKDIV16,INP_AI0,INN_AI1,VRH_VDDA,VRL_VSS,ADGN_1,VREGN_DIV2,DCSET_P0,OSR_
65536);
#endif


#if defined (ADC_ChopperMode)
   CSFON_Enable();
   ADC_ENINXCH_Enable();
   ADC_DAFM_CHOPPER();
   ADC_ENCH_Enable();
   ADC_Enable();
   ADC_CMFREnable();        //CMFR=1, Comb Filter Reset
#endif

}
/*---------------------------------------------------------------------------*/
/* Function Name: void SendUART(int data)                                    */
/* Description    : UART send data.                                          */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void SendUART(signed long    data)
{

   //display range +8388607~-8388608 (dec)
   ADCData1=data;
   if((ADCData1<0)||(ADCData1>0x80000000))    // plus-minus sign judgment
   {
           ADCData1=~ADCData1;
           ADCData1++;
         UartTxBuffer[0]='-';
   }
   else
   {
      UartTxBuffer[0]='+';
   }
   UartTxBuffer[7]=ADCData1%10 | '0';    //unit
   ADCData1=ADCData1/10;
   UartTxBuffer[6]=ADCData1%10 | '0';    //unit
   ADCData1=ADCData1/10;
   UartTxBuffer[5]=ADCData1%10 | '0';    //ten
   ADCData1=ADCData1/10;
   UartTxBuffer[4]=ADCData1%10 | '0';    //hundred
   ADCData1=ADCData1/10;
```

```
    UartTxBuffer[3]=ADCData1%10 | '0';    //thousand
    ADCData1=ADCData1/10;
    UartTxBuffer[2]=ADCData1%10 | '0';    //ten thousand
    ADCData1=ADCData1/10;
    UartTxBuffer[1]=ADCData1%10 | '0';    //hundred thousand
    ADCData1=ADCData1/10;
    UartTxBuffer[8]='\r';
    UartTxBuffer[9]='\n';
    Flagbits.b_UART_TxDone=0;
    UartTxLength=10;
    UartTxIndex=0;
    TXIE_Enable();       //Enable UART Tx Interrupt;
    RCIE_Enable();     //Enable UART Rx Interrupt
    while(!Flagbits.b_UART_TxDone);    //If    Flagbits.b_UART_TxDone=DISABLE, stop at here
}

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


void Delayms(unsigned int ms)
{
    unsigned char t;
    for(;ms>0;ms--)             //@HAO=1.843MHz
       for(t=114;t>0;t--)
       { __asm__("NOP"); }
}


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

12. 通訊 IP(I2C)

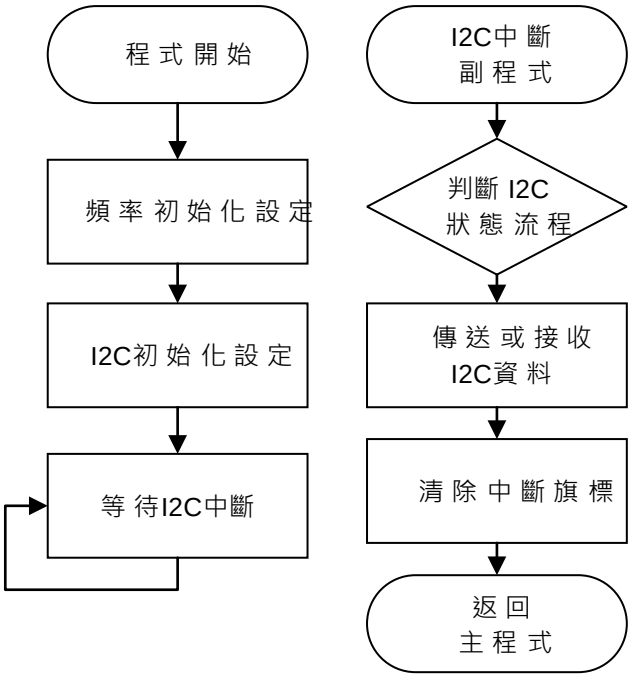
12.1. 範例名稱

I2C_Master_Demo

12.2. 範例說明

(1) HY17M28 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 HY17M28 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。

12.3. 軟體流程


程 式 開 始
頻 率 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

12.4. 程式碼

說明 ： 在此僅貼上 I2C_Master_Demo.C 程式內容做參考.

```
#define   USE_HY17M28_2M
/*******************************************************************************
 * I2C_E2PROM.c                                                                *
 * -------------------------------------------------------------------------- *
 * Copyright 2022 HYCON Technology                                            *
 * http://www.hycontek.com/                                                   *
 *                                                                            *
 * Program Description:                                                       *
 *                                                                            *
 *           HY17M28                    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: HY17M28                                                             *

 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <PWR.h>
#include <I2C.h>
#include <GPIO.h>
#include "EEPROM_24Cxx.h"
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
```

```
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif

#define I2C_PORT GSCL_PT16

#define   I2CBufferSize   10
#define I2C_WRITE   0x00          // I2C WRITE command
#define I2C_READ    0x01          // I2C READ command

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void Delayms(unsigned int ms);
void I2CInit(void);
void CLOCKInit(void);
/*---------------------------------------------------------------------------*/
/* 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];
volatile unsigned char I2C_EndFlag;
unsigned char EEPROM_WriteData[5] = {0x02,0x03,0x04,0x05,0x8A};
unsigned int I2C_DataTxLen,I2C_DataTxIndex,I2C_DataRxLen,I2C_DataRxIndex;
unsigned char i,temp;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    CLOCKInit();
    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
    Delayms(10);                              //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

    // I2C sequential I2C_WRITE & I2C_READ
```

```
	EEPROM_WriteArray(0x01,EEPROM_WriteData,5); //I2C_WRITE array data to the word address from 0x01 to
0x04
	Delayms(10);                                    //Delay for EEPROM 24C02 I2C_WRITE Cycle Time
	EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 2); //sequential I2C_READ data from 0x00 to 0x01
	EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 5); //sequential I2C_READ data from 0x00 to 0x04
	EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 6); //sequential I2C_READ data from 0x00 to 0x05
	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                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

}

/*---------------------------------------------------------------------------*/
/* I2C Init Function                                                         */
/*---------------------------------------------------------------------------*/
void I2CInit(void)
{
   //GPIO Init
   GPIO_GSCLSet(I2C_PORT);

#if defined(HAO_2MHZ)
   I2C_Open(40);   //Default CPU clock is 2MHz, Data Baud Rate : (I2CLK)/[4x(CRG+1)]= 2000000/[4*(4+1)]=100kHz
#endif
#if defined(HAO_4MHZ)
   I2C_Open(9);   //Default CPU clock is 1MHz, Data Baud Rate : (I2CLK)/[4x(CRG+1)]= 4000000/[4*(9+1)]=100kHz
#endif
#if defined(HAO_8MHZ)
   I2C_Open(19);   //Default CPU clock is 8MHz, Data Baud Rate : (I2CLK)/[4x(CRG+1)]= 8000000/[4*(19+1)]=100kHz
#endif
#if defined(HAO_16MHZ)
   I2C_Open(39);   //Default CPU clock is 16MHz, Data Baud Rate : (I2CLK)/[4x(CRG+1)]= 16000000/[4*(39+1)]=100kHz
#endif
}
```

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


void Delayms(unsigned int ms)
{
   unsigned char t;
   for(;ms>0;ms--)              //@HAO=1.843MHz
      for(t=114;t>0;t--)
      { __asm__("NOP"); }
}

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

13. 其他 IP(Power)

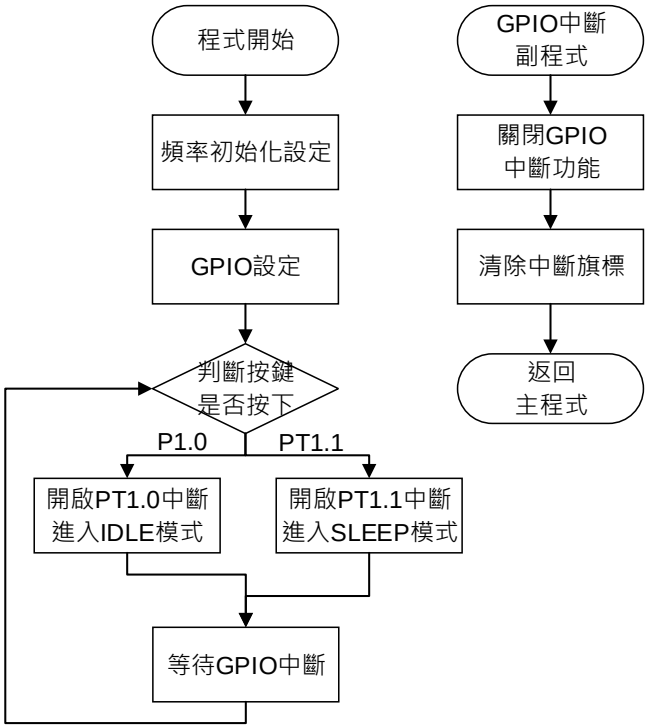
13.1. 範例名稱

Demo_PWR

13.2. 範例說明

(1) HY17M28 進入 Sleep 與 Idle Mode 範例程式
(2) 設置系統工作頻率。
(3) 關閉 BOR2 與 VDDA 電壓
(4) 電後按下 PT1.0 進入 Sleep Mode，再按下 PT1.0 喚醒。
(5) 電後按下 PT1.1 進入 Idle Mode，再按下 PT1.1 喚醒。
(6) 此範例程式搭配 HY17M28 硬體開發板，可測量到 Sleep 耗電流約為 0.25uA, Idle 耗電流約為 1uA.

13.3. 軟體流程



13.4. 程式碼

```
#define   USE_HY17M28_2M
/*****************************************************************************
 * ------------------------------------------------------------------------- *
 * Copyright 2021 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M28                                                          *
 *****************************************************************************/

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

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#ifdef USE_HY17M28_2M
#define HAO_2MHZ
#endif
#ifdef USE_HY17M28_4M
#define HAO_4MHZ
#endif
#ifdef USE_HY17M28_8M
#define HAO_8MHZ
#endif
#ifdef USE_HY17M28_16M
#define HAO_16MHZ
#endif
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int del);
void GPIOInit(void);
void CLOCKInit(void);
```

```
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
volatile typedef union _Flag
{
   unsigned int   _byte;
   struct
   {
      unsigned b_PT1INT0done:1;
      unsigned b_PT1INT1done:1;
      unsigned b_PT1INT2done:1;
      unsigned b_PT1INT3done:1;
      unsigned b_PT1INT4done:1;
      unsigned b_PT1INT5done:1;
      unsigned b_PT1INT6done:1;
      unsigned b_PT2INT0done:1;
      unsigned b_PT2INT1done:1;
      unsigned b_PT2INT2done:1;
      unsigned b_PT2INT3done:1;
      unsigned b_PT2INT4done:1;
      unsigned b_PT2INT5done:1;
      unsigned b_PT2INT6done:1;
      unsigned b_PT2INT7done:1;
      unsigned b_PT3INT0done:1;    //PT31~PT37 not design in this demo
   };
} Flag;
Flag     Flagbits;
unsigned int counterA;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{

   counterA=0;
   CLOCKInit();
   GPIOInit();
   Flagbits.b_PT1INT0done=0;
   Flagbits.b_PT1INT1done=0;
   GIE_Enable();

   while(1)
   {

      if(Flagbits.b_PT1INT0done==1)   //PT1.0
      {
```

```
        Flagbits.b_PT1INT0done=0;
        CSFON_Enable();
        ENBOR2_Disable();
        PWR_BGRDisable();
        CLK_IDLE_HAOSet(IDLEM_HAOAuto);   //OSC switch to HAO ,when the GPIO interrupt wake up
        Idle();                           //IDLE mode
        counterA++;
      }
      if(Flagbits.b_PT1INT1done==1)   //PT1.1
      {
        Flagbits.b_PT1INT1done=0;
        CSFON_Enable();
        ENBOR2_Disable();
        PWR_BGRDisable();
        CLK_OSCSelect(OSCS_LPO);
        CLK_HAODisable();
        Sleep();                          //Sleep mode
        counterA--;
      }

    }

}
/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
    NOP();
    //GPIO Event

    if(E0IF_IsFlag())
    {
      Flagbits.b_PT1INT0done=1;
      E0IF_ClearFlag();
    }

    if(E1IF_IsFlag())
    {
      Flagbits.b_PT1INT1done=1;
      E1IF_ClearFlag();
    }
}
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
```

```
void Delay(unsigned int del)
{
   while(--del)
   NOP();
}
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{//Please refer to the Datasheet for HAOM center frequency
#if defined(HAO_2MHZ)
   CLK_CPUCKOpen(HAOM_2MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_4MHZ)
   CLK_CPUCKOpen(HAOM_4MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_8MHZ)
   CLK_CPUCKOpen(HAOM_8MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif

#if defined(HAO_16MHZ)
   CLK_CPUCKOpen(HAOM_16MHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_HSCK );
#endif
}
/*---------------------------------------------------------------------------*/
/* GPIO Init Function                                                        */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
   //GPIO initial on PT1.0 & PT1.1 (INPUT)
   GPIO_PT1DigitalEnable(PT10);    //PT1.0 digital input enable
   GPIO_PT1DigitalEnable(PT11);    //PT1.1 digital input enable
   GPIO_PT1SETPU(PT10);    //PT1.0 input high
   GPIO_PT1SETPU(PT11);    //PT1.1 input high
   INTEG0Sel(INTEG0_EDGEFALL);    //PT1.0. INTEG0[1:0]=00b, falling edge
   INTEG1Sel(INTEG1_EDGEFALL);    //PT1.0. INTEG0[1:0]=00b, falling edge
   E0IE_Enable();    //PT1.0 E0IE enable
   E1IE_Enable();    //PT1.1 E1IE enable
   E0IF_ClearFlag();    //PT1.0 E0IF=0b
   E1IF_ClearFlag();    //PT1.1 E1IF=0b
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

14. 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	2022/12/30	初版發行