HY16F3981

HYCON IP 使用說明書

Table of Contents

1. 文件說明 8

2. 晶片說明 8

3. 數位 IP(TIMER) 10

3.1. 範例名稱 10

3.2. 範例說明 10

3.3. 軟體流程 10

3.4. 程式碼 11

4. 數位 IP(WDT) 17

4.1. 範例名稱 17

4.2. 範例說明 17

4.3. 軟體流程 17

4.4. 程式說明 18

5. 數位 IP(WDT RESET) 22

5.1. 範例名稱 22

5.2. 範例說明 22

5.3. 軟體流程 22

5.4. 程式說明 23

6. 數位 IP(RTC) 28

6.1. 範例名稱 28

6.2. 範例說明 28

6.3. 軟體流程 28

6.4. 程式說明 29

7. 數位 IP(PWM) 35

7.1. 範例名稱 35

7.2. 範例說明 35

7.3. 軟體流程 35

7.4. 程式說明 36

8. 數位 IP(FLASH) 40

8.1. 範例名稱 40

8.2. 範例說明 40

8.3. 程式說明 40

9. 數位 IP(GPIO) 45

9.1. 範例名稱 45

9.2. 範例說明 45

9.3. 軟體流程 45

9.4. 程式說明 46

10. 類比 IP(12 BIT RESISTANCE LADDER DAC) 50

10.1. 範例名稱 50

10.2. 範例說明 50

10.3. 軟體流程 50

10.4. 程式說明 51

11. 類比 IP(OPA) 55

11.1. 範例名稱 55

11.2. 範例說明 55

11.3. 軟體流程 55
11.4. 程式說明 56
12. 類比 IP(ADC) 60
12.1. 範例名稱 60
12.2. 範例說明 60
12.3. 軟體流程 60
12.4. 程式說明 61
13. 類比 IP(ADC_IA) 67
13.1. 範例名稱 67
13.2. 範例說明 67
13.3. 軟體流程 67
13.4. 程式說明 68
14. 類比 IP(LCD) 74
14.1. 範例名稱 74
14.2. 範例說明 74
14.3. 軟體流程 74
14.4. 程式說明 75
15. 通訊 IP(SPI) 79
15.1. 範例名稱 79
15.2. 範例說明 79
15.3. 系統設定 79
15.4. 軟體流程 80
15.5. 程式碼 81

16. 通訊 IP(UART) 86
16.1. 範例名稱 86
16.2. 範例說明 86
16.3. 軟體流程 86
16.4. 程式說明 87
17. 通訊 IP(UART2) 95
17.1. 範例名稱 95
17.2. 範例說明 95
17.3. 軟體流程 95
17.4. 程式說明 96
18. 通訊 IP(I2C) 104
18.1. 範例名稱 104
18.2. 範例說明 104
18.3. 系統設定 104
18.4. 軟體流程 105
18.5. 程式碼 105
19. 其他 IP(POWER) 112
19.1. 範例名稱 112
19.2. 範例說明 112
19.3. 軟體流程 112
19.4. 程式說明 113
20. 其他 IP(LVD) 118
20.1. 範例名稱 118

20.2. 範例說明 118

20.3. 軟體流程 118

20.4. 程式說明 119

21. 修訂記錄 123

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。

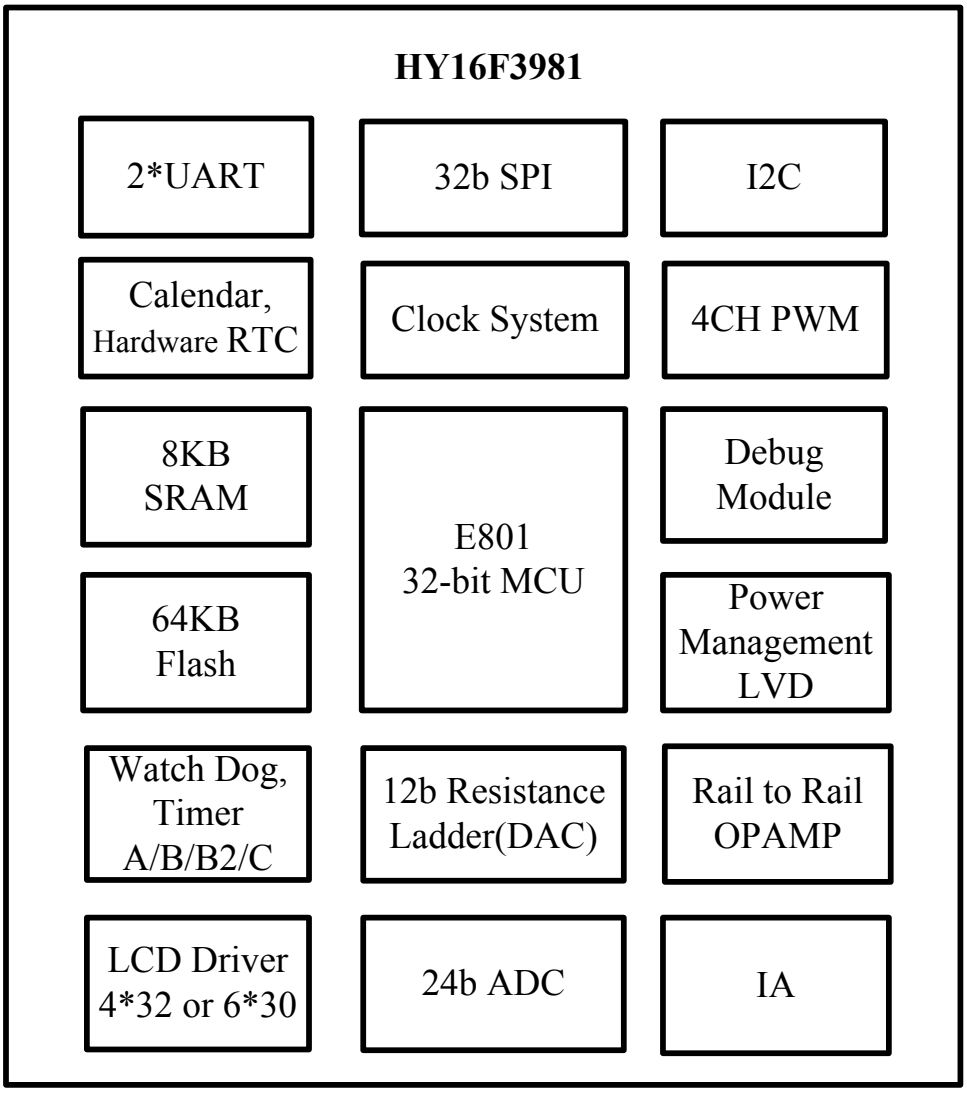
1. 文件說明

HYCON IP(IP 矽智財 Intellectual Property 縮寫)，代表紘康 32 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 HY16F3981 系列，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/timerB2/TimerC/WDT/PWM/Hardware RTC/GPIO
(2)類比 IP：12 bit Resistance Ladder(DAC)/ADC/OPAMP/LCD/IA
(3)通訊 IP：Hardware 32-bit SPI/ Hardware UART/ Hardware UART 2/ Hardware I2C
(4)其他 IP：Power Management/LVD

2. 晶片說明

HY16F3981 各功能 IP 基礎使用說明。



(01)採用晶心科技 Andes 32 位元 CPU 核心 E801 處理器。
(02)電壓操作範圍 2.2~3.6V(類比電源沒開啟情況下)，以及-40℃~85℃工作溫度範圍。
(03)支援外部 16MHz 石英震盪器或內部 16MHz 高精度 RC 震盪器，
(04)程式記憶體 64K-Byte Flash ROM
(05)資料記憶體 8K-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)內置 PGA (Programmable Gain Amplifier), 最大輸入放大倍率高達 256 倍放大。
　　(7.2)內置溫度感測器 TPS。
(08)內建 1 組 OPA 運算放大器。
(09)內建硬體 12-bit Resistance Ladder(DAC)。
(10)16-bit Timer A
(11)16-bit Timer B/B2 模組具 PWM 波形產生功能
(12)16-bit Timer C 模組具數位 Capture 功能
(13)硬體串列通訊 32-bit SPI/I2C/UART*2 模組
(14)硬體 RTC 時鐘功能模組
(15)LCD 驅動模組

※中斷副程式型態

編號	中斷副程式宣告	說明	備註
HW0	void HW0_ISR(void)	通訊 IP	UART/I2C/32-bit SPI
HW1	void HW1_ISR(void)	數位 IP	Timer A/B/C & WDT/RTC
HW2	void HW2_ISR(void)	ADC IP	SD 24-bit ADC
HW4	void HW4_ISR(void)	PT3 IP	PT3.0~PT3.7 可獨立或一起使用
HW5	void HW5_ISR(void)	PT2 IP	PT2.0~PT2.7 可獨立或一起使用
HW6	void HW6_ISR(void)	SW Int	
HW7	void HW7_ISR(void)	UART2	
HW8	void HW8_ISR(void)	TMB2	
HW9	void HW9_ISR(void)	OPA IP	Analog OPAMP

3. 數位 IP(Timer)

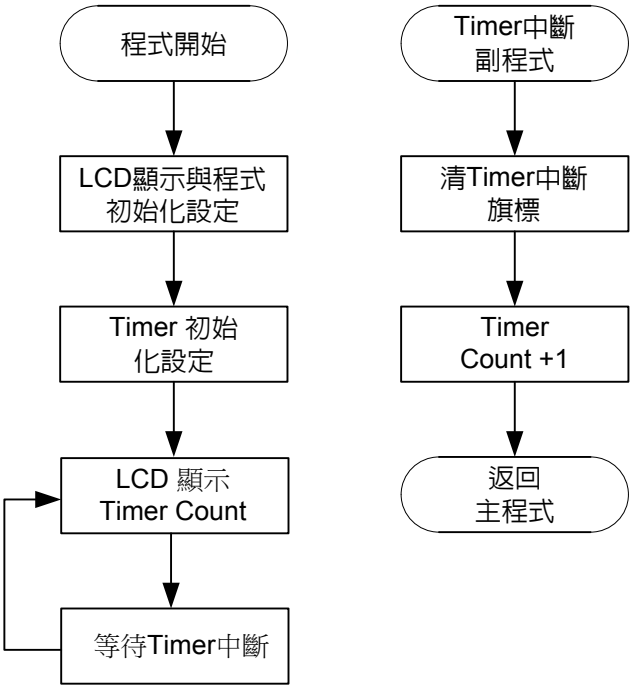
3.1. 範例名稱

HY16F3981_Timer

3.2. 範例說明

(1)Timer A/Timer B/Timer C 使用方式與說明.
(2)利用程式碼#define 來選擇要編譯執行 TMATEST 或 TMBTEST 或 TMCTEST.
(3)程式做好 Timer 初始化動作與設置相關 Timer 計數溢出值,開啟 System GIE 等待 Timer 中斷發生. Timer 計數溢出值可決定 Timer 進出中斷的速度.
(4)每進一次 Timer 中斷, 會在 Timer 中斷副程式內, 將所設定的變數 Timer Count 做+1 的動作, 並且回到主程式把 Timer Count 顯示在 LCD Display 上.

3.3. 軟體流程



3.4. 程式碼

```
/*****************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_Timer
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : User choose #define TMATEST or TMBTEST or TMCTEST.
 * This program shows Timer count ++ on LCD Display
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvTimer.h"
#include "System.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_TMC1done:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
} MCUSTATUS;
```

```
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define    TMATEST
//#define    TMBTEST
#define    TMCTEST

/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
MCUSTATUS    MCUSTATUSbits;
unsigned int TimerA_count, TimerB_count, TimerC0_count, TimerC1_count;

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

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

   TimerA_count=0;
   TimerB_count=0;
   TimerC0_count=0;
   TimerC1_count=0;
   DisplayInit();
   ClearLCDframe();

#if defined(TMATEST)
   DrvTMA_Open(15,1);                    //Timer A Overflow
                                         //15:taclk/65536/32;TMRDV=÷32
                                         //01:HS_CK
   DrvTIMER_ClearIntFlag(E_TMA);         //Clear Timer A interrupt flag
   DrvTIMER_EnableInt(E_TMA);            //Timer A interrupt enable
#endif

#if defined(TMBTEST)
   DrvTMBC_Clk_Source(1,3);              //Timer B Prescaler 1
                                         //1: HS_CK,clock source.
                                         //3: clock divider.÷8

   DrvTMB_Open(E_TMB_MODE0,E_TMB_NORMAL,0xFFFF);   //Timer B overflow 0xffff
   DrvTIMER_ClearIntFlag(E_TMB);         //Clear Timer B interrupt flag
   DrvTIMER_EnableInt(E_TMB);            //Timer B interrupt enable

#elif defined(TMCTEST)
   DrvTMBC_Clk_Source(1,1);              //Timer B Prescaler 1
                                         //1: HS_CK,clock source.
                                         //1: clock divider.÷2

   DrvTMB_Open(E_TMB_MODE0,E_TMB_NORMAL,0xFFFF);   //Timer B overflow 0xffff

   DrvCapture1_Open(2,14,1);             //Timer C0 use as Capture 1
                                         //input source selection
                                         //2:LS_CK
```

```
                                        //14:÷16384 1:Rising-edge trigger
    DrvCapture2_Open(1,1);              //Timer C1 use as Capture 2
                                        //Input source selection
                                        //1:same as Caputre1
                                        //Falling-edge trigger

    DrvTIMER_ClearIntFlag(E_TMC0);      //Clear TMC0 interrupt flag
    DrvTIMER_ClearIntFlag(E_TMC1);      //Clear TMC1 interrupt flag
    DrvTIMER_EnableInt(E_TMC0);         //Timer C0 interrupt enable
    DrvTIMER_EnableInt(E_TMC1);         //Timer C1 interrupt enable
#endif

    SYS_EnableGIE(4,0x3FF);             //Enable GIE(Global Interrupt)
    MCUSTATUSbits._byte = 0;

    while(1)                            //Wait for Interrupt
    {

      if(MCUSTATUSbits.b_TMAdone==1)
      {
        LCD_DATA_DISPLAY(TimerA_count);
        MCUSTATUSbits.b_TMAdone=0;
      }

      if(MCUSTATUSbits.b_TMBdone==1)
      {
        LCD_DATA_DISPLAY(TimerB_count);
        MCUSTATUSbits.b_TMBdone=0;
      }

      if(MCUSTATUSbits.b_TMC0done==1)
      {
        LCD_DATA_DISPLAY(TimerC0_count);
        MCUSTATUSbits.b_TMC0done=0;
      }

      if(MCUSTATUSbits.b_TMC1done==1)
      {
        LCD_DATA_DISPLAY(TimerC1_count);
        MCUSTATUSbits.b_TMC1done=0;
      }

    }
    return 0;
}


/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).            */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

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

```
/* Function Name: HW1_ISR()                                                    */
/* Description  : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).     */
/* Arguments    : None.                                                        */
/* Return Value : None.                                                        */
/* Remark       :                                                              */
/*-----------------------------------------------------------------------------*/
void HW1_ISR(void)
{
  if(DrvTIMER_GetIntFlag(E_TMA))
  {
    DrvTIMER_ClearIntFlag(E_TMA);          // Clear TMA interrupt flag
    MCUSTATUSbits.b_TMAdone=1;
    TimerA_count++;
  }

  if(DrvTIMER_GetIntFlag(E_TMB))
  {
    DrvTIMER_ClearIntFlag(E_TMB);          // Clear TMB interrupt flag
    MCUSTATUSbits.b_TMBdone=1;
    TimerB_count++;
  }

  if(DrvTIMER_GetIntFlag(E_TMC0))
  {
    DrvTIMER_ClearIntFlag(E_TMC0);         // Clear TMC0 interrupt flag
    MCUSTATUSbits.b_TMC0done=1;
    TimerC0_count++;
  }

  if(DrvTIMER_GetIntFlag(E_TMC1))
  {
    DrvTIMER_ClearIntFlag(E_TMC1);         // Clear TMC1 interrupt flag
    MCUSTATUSbits.b_TMC1done=1;
    TimerC1_count++;
  }

}
/*-----------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                    */
/* Description  : ADC interrupt Service Routine (HW2).                         */
/* Arguments    : None.                                                        */
/* Return Value : None.                                                        */
/* Remark       :                                                              */
/*-----------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*-----------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                    */
/* Description  : PT3 interrupt Service Routine (HW4).                         */
/* Arguments    : None.                                                        */
/* Return Value : None.                                                        */
/* Remark       :                                                              */
/*-----------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
```

```
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description  : PT2 interrupt Service Routine (HW5).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description  : UART2 interrupt Service Routine (HW7).                     */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description  : TMB2 interrupt Service Routine (HW8).                      */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description  : OPA interrupt Service Routine (HW9).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description  : Exception Service Routines.                                */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
```

```
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

4. 數位 IP(WDT)

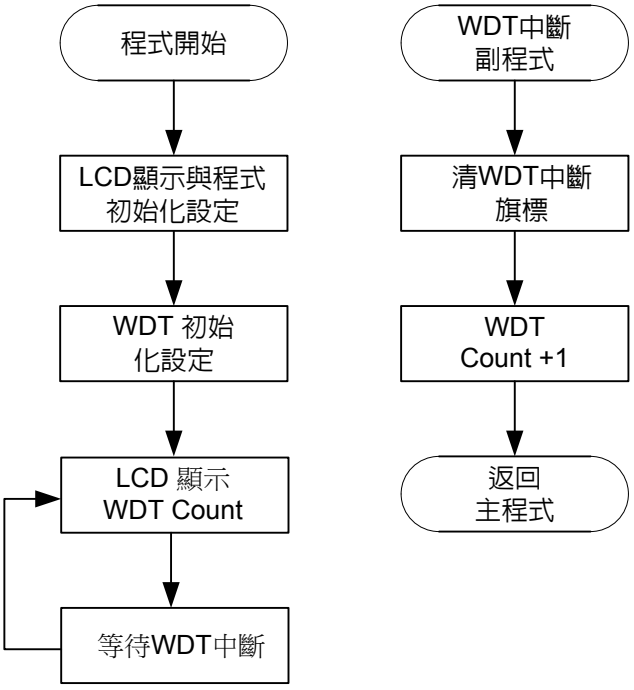
4.1. 範例名稱

HY16F3981_WDT

4.2. 範例說明

(1)WDT 使用方式與說明
(2)程式做好 WDT 初始化動作與設置 WDT 計數溢出值條件,開啟 System 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.
(3)每進一次 WDT 中斷,會在 WDT 中斷副程式內, 將所設定的變數 WDTCount 做+1 的動作, 並且回到主程式把 WDT Count 顯示在 LCD 上。

4.3. 軟體流程



4.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_WDT
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 to do WDT_count++, WDT_count display on LCD
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvTimer.h"
#include "System.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_WDTdone:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
} MCUSTATUS;

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

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

/*----------------------------------------------------------------------------*/
/* Global CONSTANTS                                                           */
/*----------------------------------------------------------------------------*/
MCUSTATUS    MCUSTATUSbits;
unsigned int WDT_count;

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

/*----------------------------------------------------------------------------*/
/* Main Function                                                              */
/*----------------------------------------------------------------------------*/
int main(void)
{
   WDT_count=0;
   DisplayInit();
   ClearLCDframe();

   DrvWDT_Open(E_IRQ,E_PRE_SCALER_D32);    //WDT IRQ open pre scaler 32
   DrvWDT_ClearWDT();                      //Clear WDT interrupt flag
   DrvTIMER_EnableInt(E_WDT);              //WDT interrupt enable
   SYS_EnableGIE(4,0x3FF);                 //Enable GIE(Global Interrupt)
   MCUSTATUSbits._byte = 0;

   while(1)                                //Wait for Interrupt
   {
      if(MCUSTATUSbits.b_WDTdone==1)
      {
         LCD_DATA_DISPLAY(WDT_count);
         MCUSTATUSbits.b_WDTdone=0;
      }
   }
   return 0;
}

/*----------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                   */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                     */
/* Return Value : None.                                                       */
/* Remark         :                                                           */
/*----------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*----------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                   */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                     */
/* Return Value : None.                                                       */
/* Remark         :                                                           */
/*----------------------------------------------------------------------------*/
void HW1_ISR(void)
```

```
{
   if(DrvTIMER_GetIntFlag (E_WDT))
   {
        WDT_count++;
      MCUSTATUSbits.b_WDTdone=1;
      DrvTIMER_ClearIntFlag(E_WDT);     //Clear WDT interrupt flag
   }
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description    : ADC interrupt Service Routine (HW2).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description    : PT3 interrupt Service Routine (HW4).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
```

```
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

5. 數位 IP(WDT Reset)

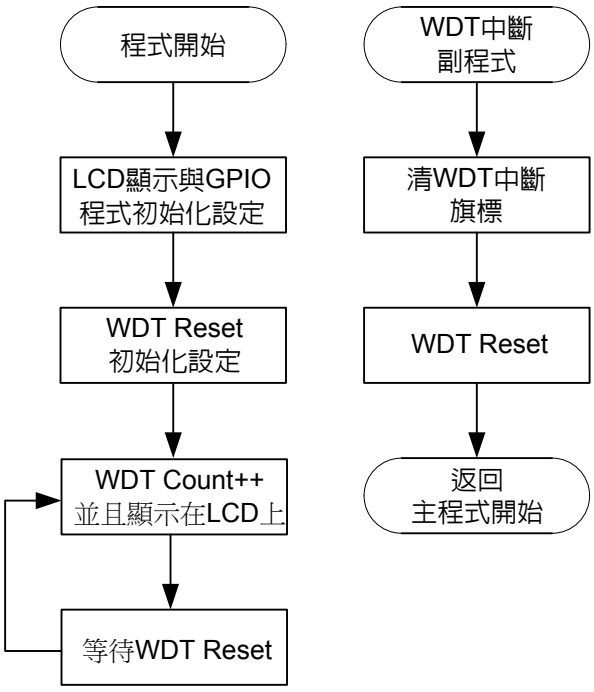
5.1. 範例名稱

HY16F3981_WDT_Reset

5.2. 範例說明

(1)WDT Reset 使用方式與說明
(2)程式做好 WDT 初始化動作與設置 WDT 計數溢出值條件,開啟 WDT Reset 與 System GIE, 主程式開始計數 WDT Count 並且顯示於 LCD 上, 等待 WDT Reset 發生.
(3)WDT Count 大約計數至 2500~3400 上下，即會發生 WDT Reset (因為每顆 IC 的高頻 HAO 內震頻率不一樣, 所以 WDT Count 計數到發生 WDT Reset 時間不一定都會相同)。
(4)可透過 PT3.0 按鍵，清除 WDT 計數暫存器, 並且重新歸零計數 WDT Count，重新計數 WDT Reset 發生時間。

5.3. 軟體流程



5.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_WDT_Reset
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 WDT Reset occur on WDT_count near to 2500~3400, it depends on the HAO frequency setting
 * After WDT_Reset, if PT3.0=Low. Set WDT_count=0, WDT re-count again, start from 0 to count
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "Display.h"
#include "System.h"
#include "DrvGPIO.h"
#include "DrvTimer.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_WDTdone:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
```

```
} MCUSTATUS;


typedef union _PTINTSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_PTINT0done:1;
      unsigned b_PTINT1done:1;
      unsigned b_PTINT2done:1;
      unsigned b_PTINT3done:1;
      unsigned b_PTINT4done:1;
      unsigned b_PTINT5done:1;
      unsigned b_PTINT6Done:1;
      unsigned b_PTINT7Done:1;
   };
} PTINTSTATUS;

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#define KEY_PORT E_PT3
#define KEYIN0 BIT0


/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
MCUSTATUS    MCUSTATUSbits;
PTINTSTATUS    PT3INTSTATUSbits;
unsigned int WDT_count;

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

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
   WDT_count=0;
   DisplayInit();
   ClearLCDframe();

   DrvWDT_Open(E_NMI,E_PRE_SCALER_D2048);              //WDT IRQ open pre scaler 2048
   DrvWDT_ClearWDT();                                  //Clear WDT_count
   DrvWDT_ResetEnable();                               //set WDNMI=1. 0x40108[6]=1b

   DrvGPIO_ClkGenerator(E_HS_CK,1);                    //Set IO sampling clock input source is HS_CK
   DrvGPIO_Open(KEY_PORT,KEYIN0,E_IO_INPUT);           //set PT3.0 INPUT
   DrvGPIO_Open(KEY_PORT,KEYIN0,E_IO_PullHigh);        //enable PT3.0 pull high R
   DrvGPIO_Open(KEY_PORT,KEYIN0,E_IO_IntEnable);       //PT3.0 interrupt enable
   DrvGPIO_IntTrigger(KEY_PORT,KEYIN0,E_N_Edge);       //PT3.0 interrupt trigger method is negative edge
   DrvGPIO_ClearIntFlag(KEY_PORT,KEYIN0);              //clear PT3 interrupt flag

   PT3INTSTATUSbits._byte = 0;
```

```
    MCUSTATUSbits._byte = 0;
    SYS_EnableGIE(4,0x3FF);                         //Enable GIE(Global Interrupt)

    while(1)
    {
      for(WDT_count=0;WDT_count<999999;WDT_count++)
      {
        LCD_DATA_DISPLAY(WDT_count);                  //WDT Reset occur on WDT_count=2500~3400, it depends on the HAO
frequency
        Delay(1000);
        if(PT3INTSTATUSbits.b_PTINT0done)           //if PT3.0 low
        {
          WDT_count=0;                                //Set WDT_count=0,
          DrvWDT_ClearWDT();                          //WDT re-count again, start from 0 to count
          PT3INTSTATUSbits.b_PTINT0done=0;
        }
      }
    }

    return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{
    if(DrvTIMER_GetIntFlag(E_WDT))
    {
      MCUSTATUSbits.b_WDTdone=1;
      DrvTIMER_ClearIntFlag(E_WDT);             //Clear WDT interrupt flag
    }
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
```

```
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description    : PT3 interrupt Service Routine (HW4).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{
   uint32_t PORT_IntFlag;

   PORT_IntFlag=DrvGPIO_GetIntFlag(KEY_PORT);
   if((PORT_IntFlag&KEYIN0)==KEYIN0)
   {
      PT3INTSTATUSbits.b_PTINT0done=1;
   }
   DrvGPIO_ClearIntFlag(KEY_PORT,KEYIN0);    //clear PT3.0 interrupt flag

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
```

```
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

6. 數位 IP(RTC)

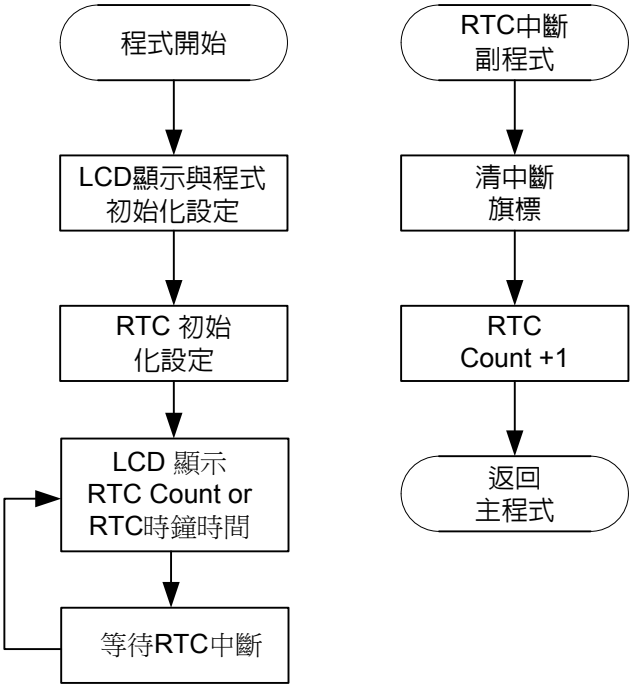
6.1. 範例名稱

HY16F3981_RTC

6.2. 範例說明

(1)RTC 使用方式與說明
(2)利用程式碼#define 來選擇要編譯執行 RTC_counter_show 或 TimerDtata_show.
(3)程式做好 RTC 初始化動作與設置 RTC 計數溢出值條件,開啟 System GIE, 等待 RTC 中斷發生.
(4)當選擇 RTC_counter_show, LCD 會顯示 RTC 進入中斷之後的每一次 RTC Count 計數+1, RTC Count 從 0 開始計數, 當選擇 TimerDtata_show, LCD 會顯示程式中所設置的時間, 每進一次 RTC 中斷, 時間會往上數 1 秒.

6.3. 軟體流程



6.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_RTC
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 display the RTC_counter or TimerDtata on LCD by #define RTC_counter_show or TimerDtata_show
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*----------------------------------------------------------------------*/
/* Includes                                                             */
/*----------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvRTC.h"
#include "DrvLCD.h"
#include "Display.h"
#include "System.h"
/*----------------------------------------------------------------------*/
/* STRUCTURES                                                           */
/*----------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_TMC1done:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
} MCUSTATUS;
```

```
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define    RTC_counter_show
#define    TimerDtata_show

/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
MCUSTATUS    MCUSTATUSbits;

unsigned int sec,min,hour,week,day,month,year ;
unsigned int RTC_counter;
unsigned int TimerDtata;
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void InitalRTC(void);
int ComputeWeek(int TempYear, int TempMonth, int TempDay);

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

   S_DRVRTC_TIME_DATA_T sCurTime;       //Setting Start
   RTC_counter=0;
   TimerDtata=0;

   DisplayInit();
   ClearLCDframe();
   InitalRTC();
   MCUSTATUSbits._byte = 0;
   SYS_EnableGIE(4,0x3FF);                  //Enable GIE(Global Interrupt)

   while(1)
   {
      if(MCUSTATUSbits.b_RTCdone==1)
      {
         DrvRTC_Read(DRVRTC_CURRENT_TIME,&sCurTime);
         TimerDtata=sCurTime.u32cSecond+sCurTime.u32cMinute*100+sCurTime.u32cHour*10000;
#if defined(TimerDtata_show)
           LCD_DATA_DISPLAY(TimerDtata);
#endif
#if defined(RTC_counter_show)
           LCD_DATA_DISPLAY(RTC_counter);
#endif

         sec=sCurTime.u32cSecond;
         min=sCurTime.u32cMinute;
         hour=sCurTime.u32cHour;
         week=sCurTime.u32cDayOfWeek;
         day=sCurTime.u32cDay;
         month=sCurTime.u32cMonth;
         year=sCurTime.u32Year;
         MCUSTATUSbits.b_RTCdone=0;
         }
```

```
    }

    return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).            */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{
    if(DrvRTC_ReadState()&&0x2)          //check PTF flag
    {
        DrvRTC_ClearState(E_DRVRTC_CLEAR_ALL);
        DrvRTC_ClearIntFlag();
        RTC_counter++;
        MCUSTATUSbits.b_RTCdone=1;
    }
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description    : ADC interrupt Service Routine (HW2).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description    : PT3 interrupt Service Routine (HW4).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
```

```
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*-------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                */
/* Description  : UART2 interrupt Service Routine (HW7).                   */
/* Arguments    : None.                                                    */
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*-------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                */
/* Description  : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments    : None.                                                    */
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*-------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                */
/* Description  : OPA interrupt Service Routine (HW9).                     */
/* Arguments    : None.                                                    */
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*-------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                  */
/* Description  : Exception Service Routines.                              */
/* Arguments    : None.                                                    */
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*-------------------------------------------------------------------------*/
/* Software Delay Subroutines                                              */
/*-------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
```

```
}


/*---------------------------------------------------------------------------*/
/* Function Name: InitalRTC()                                                */
/* Description    : RTC Initialization Subroutines.                          */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void InitalRTC()
{
   S_DRVRTC_TIME_DATA_T sCurTime;              //Setting Start
   DrvCLOCK_EnableLowOSC(E_EXTERNAL,130000);
   DrvRTC_ClkConfig(2);                        //Set 0x40308 RTCKS=10b, LSXT

   //DRVRTC_CURRENT_TIME or Alarm Time
   DrvRTC_WriteEnable();
   DrvRTC_Read(DRVRTC_CURRENT_TIME,&sCurTime);
   sCurTime.u8cClockDisplay=0;                 //DRVRTC_CLOCK_12//DRVRTC_CLOCK_24
   sCurTime.u8cAmPm=0;                         //DRVRTC_AM//DRVRTC_PM
   sCurTime.u32cSecond=40;
   sCurTime.u32cMinute=59;
   sCurTime.u32cHour=23;
   sCurTime.u32cDay=18;
   sCurTime.u32cMonth=2;
   sCurTime.u32Year=2018;
   sCurTime.u32cDayOfWeek=ComputeWeek(sCurTime.u32Year,sCurTime.u32cMonth,sCurTime.u32cDay);
   sCurTime.u8IsEnableWakeUp=0;                //WK
   DrvRTC_Write(DRVRTC_CURRENT_TIME,&sCurTime);
   DrvRTC_HourFormat(E_DRVRTC_HOUR_24);    //1:12hour 0:24hour
   DrvRTC_Enable();
   DrvRTC_PeriodicTimeEnable(E_DRVRTC_1_8_SEC);
   DrvRTC_ClearIntFlag();
   DrvRTC_EnableInt();                         //Enable RTC interrupt
}

/*---------------------------------------------------------------------------*/
/* Compute Week Subroutines                                                  */
/*---------------------------------------------------------------------------*/
int ComputeWeek(int TempYear, int TempMonth, int TempDay)
{
   int TempWeek;
   if (TempMonth >= 3)
   {
      TempMonth = TempMonth - 2;
   }
   else
   {
      TempMonth = TempMonth + 10;
      TempYear--;
   }

   TempWeek = TempYear + (int)(TempYear / 4) - (int)(TempYear / 100) +(int)(TempYear / 400) + (int)(2.6 * TempMonth - 0.2) +
TempDay;
   TempWeek = TempWeek - 7*(int)(TempWeek / 7);
   return (TempWeek);
}
/*---------------------------------------------------------------------------*/
```

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

7. 數位 IP(PWM)

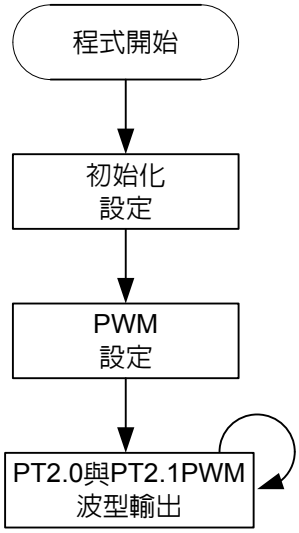
7.1. 範例名稱

HY16F3981_PWM

7.2. 範例說明

(1) PWM 使用方式與說明
(2) 程式初始先設置 HAO 工作頻率與 TimerB 計數溢出值條件
(3) 開啟 PWM 暫存器設定與 PWM Duty 設定,再將要執行 PWM 輸出的 I/O 設置為輸出模式.
(4) 可在 PT2.0 與 PT2.1 觀察到 PWM 波型, PT2.0 為 PWM0, PT2.1 為 PWM1.

7.3. 軟體流程



7.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_PWM
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : PT2.0 output PWM0(Mode:PWMA), PT2.1 output PWM1(Mode:PWMA)
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvTimer.h"
#include "System.h"
#include "DrvGPIO.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/


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


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


/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
```

```
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
   DrvCLOCK_SelectIHOSC(1);                          // Select HAO 4MHz
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);
   SYS_DisableGIE();                                 //Disable GIE

   DrvPWM0_Open(0,1,2);                              //PWM0 Enable Port 2.0 =PWMO0, Port 2.1 =PWMO1(Set PWM0=PT2.0)
   DrvPWM1_Open(1,1,2);                              //PWM1 Enable Port 2.0 =PWMO0, Port 2.1 =PWMO1(Set PWM1=PT2.1)
   DrvPWM_CountCondition(0x7fff,0x3fff);             //PWM0 Duty 0X7FFF, PWM0=PT2.0
                                                     //PWM1 Duty 0X3FFF, PWM1=PT2.1
   DrvGPIO_Open(E_PT2,0x01l0x02,E_IO_OUTPUT);        //PT2.0, PT2.1 Set Output
   DrvTMBC_Clk_Source(0,0);                          //TMB Clock Enable/1
   DrvTMB_Open(E_TMB_MODE0,E_TMB_NORMAL,0xffff);   //TMB Overflow 0XFFFF
                                                     //PWM Period 0XFFFF
   while(1);
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description   : PT3 interrupt Service Routine (HW4).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
```

```
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description  : PT2 interrupt Service Routine (HW5).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description  : UART2 interrupt Service Routine (HW7).                     */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description  : TMB2 interrupt Service Routine (HW8).                      */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description  : OPA interrupt Service Routine (HW9).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description  : Exception Service Routines.                                */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
```

```
    asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
    asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

8. 數位 IP(Flash)

8.1. 範例名稱

HY16F3981_Flash

8.2. 範例說明

(1) Flash 自我燒錄與讀取使用說明
(2) 第一段的程式範例先執行一整個 Page 的自我燒錄並讀回做驗證, 如果寫入與讀回的內容不相同, LCD 會顯示”1”並且程式卡在 While(1).
(3) 第二段的程式範例執行一個 Word 的自我燒錄並讀回做驗證, 如果寫入與讀回的內容不相同, 則設計程式卡在 While(1).
注意1 : 在執行Flash燒錄與讀取程式指令之前, 必須先執行SYS_DisableGIE(); 關閉全域使能中斷, 這可以避免程式運行異常的行為發生.
注意2 : 執行Flash燒錄指令, 必須確保晶片工作電壓VDD3V高於2.7V, 如果晶片電壓VDD3V低於2.7V, 則可能會發生燒錄錯誤行為.

8.3. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ------------------------------------------------------------------------
 * Project Name : HY16F3981_Flash
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : ROM_BurnPage on Flash 0x9F000 and then read out to check.
 * DrvFlash_Burn_Word on Flash 0x98014. 0x98010, 0x98020 and then read out to check
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*------------------------------------------------------------------------*/
```

```
/* Includes                                                                    */
/*-----------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvFlash.h"
#include "system.h"
/*-----------------------------------------------------------------------------*/
/* STRUCTURES                                                                  */
/*-----------------------------------------------------------------------------*/


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



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


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

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

   int index,error;
   unsigned int BufferTx[32];
   unsigned int BufferRx[32];

   DisplayInit();
   ClearLCDframe();
   Delay(10000);
   DisplayHYcon();
   Delay(10000);

   SYS_DisableGIE();                    //before execute Flash burn/read function, user have to do that SYS_DisableGIE();

   for(index=0;index<32;index++)
   {
      BufferTx[index]=index;            //to set BufferTx[0]=0x0....BufferTx[31]=0x1f
      BufferRx[index]=0xFFFFFFFF;       //to set BufferRx[0]=0xFFFFFFFF....BufferRx[31]=0xFFFFFFFF
   }

   ROM_BurnPage(FLASH_KEY_A,0xF000,0x2000,BufferTx);
   ReadPage(0xF000,BufferRx);           //Read BufferRx to make sure BurnPage data

   for(index=0;index<32;index++)
   {
      BufferTx[index]=31-index;         //to set BufferTx[0]=0x1f....BufferTx[31]=0x0
```

```
    BufferRx[index]=0xFFFFFFFF;       //to set BufferRx[0]=0xFFFFFFFF....BufferRx[31]=0xFFFFFFFF
  }

  ROM_BurnPage(FLASH_KEY_A,0xF000,0x2000,BufferTx);
  ReadPage(0xF000,BufferRx);          //Read BufferRx to make sure BurnPage data

  //verify the ROM_BurnPage function, if fail to show "1" on the LCD Display
  for(index=0;index<32;index++)
  {
    if(BufferTx[index]!=BufferRx[index])
    {
      LCD_DATA_DISPLAY(1);
      while(1);
    }
  }


  error=DrvFlash_Burn_Word(0x8014,0x2000,0x12345678);
  if(error==0)                        //if Error=0, it means BurnWord success
  ReadPage(0x8014,BufferRx);

  error=DrvFlash_Burn_Word(0x8010,0x2000,0x12345678);
  if(error==0)                        //if Error=0, it means BurnWord success
  ReadPage(0x8010,BufferRx);

  error=DrvFlash_Burn_Word(0x8020,0x2000,0x12345678);
  if(error==0)                        //if Error=0, it means BurnWord success
  ReadPage(0x8020,BufferRx);

  while(1);

  return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
```

```
/* Arguments       : None.                                                    */
/* Return Value : None.                                                     */
/* Remark          :                                                          */
/*--------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*--------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                 */
/* Description    : PT3 interrupt Service Routine (HW4).                    */
/* Arguments       : None.                                                  */
/* Return Value : None.                                                     */
/* Remark          :                                                        */
/*--------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*--------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                 */
/* Description    : PT2 interrupt Service Routine (HW5).                    */
/* Arguments       : None.                                                  */
/* Return Value : None.                                                     */
/* Remark          :                                                        */
/*--------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*--------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                 */
/* Description    : UART2 interrupt Service Routine (HW7).                  */
/* Arguments       : None.                                                  */
/* Return Value : None.                                                     */
/* Remark          :                                                        */
/*--------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*--------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                 */
/* Description    : TMB2 interrupt Service Routine (HW8).                   */
/* Arguments       : None.                                                  */
/* Return Value : None.                                                     */
/* Remark          :                                                        */
/*--------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*--------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                 */
/* Description    : OPA interrupt Service Routine (HW9).                    */
/* Arguments       : None.                                                  */
/* Return Value : None.                                                     */
/* Remark          :                                                        */
/*--------------------------------------------------------------------------*/
```

```
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

9. 數位 IP(GPIO)

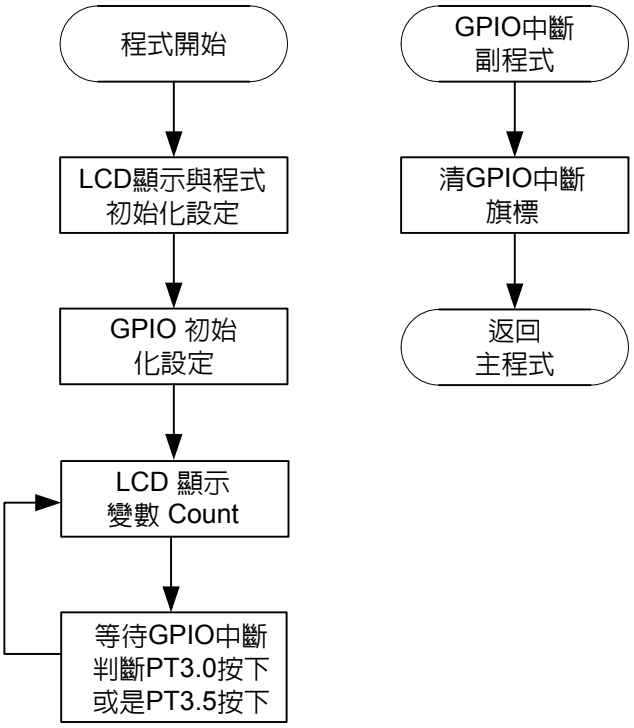
9.1. 範例名稱

HY16F3981_GPIO

9.2. 範例說明

(1) GPIO 使用範例程式.
(2)程式初始化 GPIO 相關設置與 LCD 設置.
(3)每按一次 PT3.0,變數 count+1, LCD 顯示 count 數值
(4)每按一次 PT3.5,變數 count-1, LCD 顯示 count 數值

9.3. 軟體流程



9.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_GPIO
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : if PT3.0 low to do count++, if PT3.5 low to do count--
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvGPIO.h"
#include "System.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _PTINTSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_PTINT0done:1;
      unsigned b_PTINT1done:1;
      unsigned b_PTINT2done:1;
      unsigned b_PTINT3done:1;
      unsigned b_PTINT4done:1;
      unsigned b_PTINT5done:1;
      unsigned b_PTINT6Done:1;
      unsigned b_PTINT7Done:1;
   };
} PTINTSTATUS;

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

```
/* DEFINITIONS                                                                */
/*----------------------------------------------------------------------------*/
#define KEY_PORT E_PT3
#define KEYIN0 BIT0
#define KEYIN1 BIT5

/*----------------------------------------------------------------------------*/
/* Global CONSTANTS                                                           */
/*----------------------------------------------------------------------------*/
PTINTSTATUS    PT3INTSTATUSbits;

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

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

   int count=0;

   DisplayInit();
   ClearLCDframe();
   LCD_DATA_DISPLAY(count);

   DrvGPIO_ClkGenerator(E_HS_CK,1);                    //Set IO sampling clock input source is HS_CK
   DrvGPIO_Open(KEY_PORT,KEYIN1|KEYIN0,E_IO_INPUT);          //set PT3.0/PT3.5 INPUT
   DrvGPIO_Open(KEY_PORT,KEYIN1|KEYIN0,E_IO_PullHigh);     //enable PT3.0/PT3.5 pull high R
   DrvGPIO_Open(KEY_PORT,KEYIN1|KEYIN0,E_IO_IntEnable);   //PT3.0/PT3.5 interrupt enable
   DrvGPIO_IntTrigger(KEY_PORT,KEYIN1|KEYIN0,E_N_Edge);   //PT3.0/PT3.5 interrupt trigger method is negative edge
   DrvGPIO_ClearIntFlag(KEY_PORT,KEYIN1|KEYIN0);           //clear PT3 interrupt flag
   PT3INTSTATUSbits._byte = 0;
   SYS_EnableGIE(4,0x3FF);                             //Enable GIE(Global Interrupt)

   while(1)
   {
      if(PT3INTSTATUSbits.b_PTINT0done)                //if PT3.0 low
      {
         LCD_DATA_DISPLAY(count++);
         PT3INTSTATUSbits.b_PTINT0done=0;
      }
      if(PT3INTSTATUSbits.b_PTINT5done)                //if PT3.5 low
      {
         LCD_DATA_DISPLAY(count--);
         PT3INTSTATUSbits.b_PTINT5done=0;
      }
   }

}

/*----------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                   */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                     */
/* Return Value : None.                                                       */
/* Remark         :                                                           */
```

```
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description   : PT3 interrupt Service Routine (HW4).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{
  uint32_t PORT_IntFlag;

  PORT_IntFlag=DrvGPIO_GetIntFlag(KEY_PORT);
  if((PORT_IntFlag&KEYIN0)==KEYIN0)
  {
    PT3INTSTATUSbits.b_PTINT0done=1;
  }
    if((PORT_IntFlag&KEYIN1)==KEYIN1)
  {
    PT3INTSTATUSbits.b_PTINT5done=1;
  }
  DrvGPIO_ClearIntFlag(KEY_PORT,KEYIN1|KEYIN0);          //clear PT3.0/PT3.5 interrupt flag

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description   : PT2 interrupt Service Routine (HW5).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{
```

```
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

10. 類比 IP(12 bit Resistance Ladder DAC)

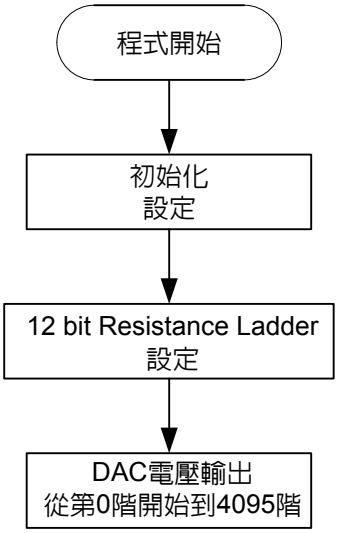
10.1. 範例名稱

HY16F3981_DAC

10.2. 範例說明

(1) 12 bit Resistance ladder DAC 使用說明
(2) 程式先將 VDDA 做開啟動作，設置 VDDA 為 12 bit Resistance Ladder 的正端，負端設置為 VSS. 在此設置下, DAC 能夠輸出的最高電壓為 VDDA
(3) DAC 的輸出由第 0 階開始輸出到 4095 階，可以由 IC 腳位 DAO 量到 DAC 輸出電壓

10.3. 軟體流程



10.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_DAC
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 12bit DAC output voltage from stage 0 to 4095
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvDAC.h"
#include "DrvPMU.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/


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


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


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

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
```

```
/*---------------------------------------------------------------------------*/
int main(void)
{
  unsigned char i;

  //Set Power system
  DrvPMU_VDDA_LDO_Ctrl(E_LDO);                    //LDO ON
  DrvPMU_VDDA_Voltage(E_VDDA2_4);                 //VDDA=2.4

  //Set DAC
  DrvDAC_Open(E_DAC_PVDDA,E_DAC_NVSSA,0);   //DAC_Vrefp= VDDA, DAC_Vrefn= VSSA, DAO=0
  DrvDAC_DALH(1);                                 //Set DALH=1b
  DrvDAC_EnableOutput();                          //Enable 12-bit resistance ladder DAC output
  DrvDAC_Enable();                                //Enable 12-bit resistance ladder DAC function

  while(1)
  {
    for(i=0;i<4096;i++)
    {
      DrvDAC_DABIT(i);
      Delay(1000);                                //Delay for reference
        //User can use meter to check the pin14(DAO) to pin2(VSS) to check the DAC output voltage
    }
  }
  return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{
```

```
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description    : PT3 interrupt Service Routine (HW4).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
```

```
/* Arguments        : None.                                                    */
/* Return Value : None.                                                    */
/* Remark             :                                                       */
/*-------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*-------------------------------------------------------------------------*/
/* Software Delay Subroutines                                              */
/*-------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*-------------------------------------------------------------------------*/
/* End Of File                                                             */
/*-------------------------------------------------------------------------*/
```

11. 類比 IP(OPA)

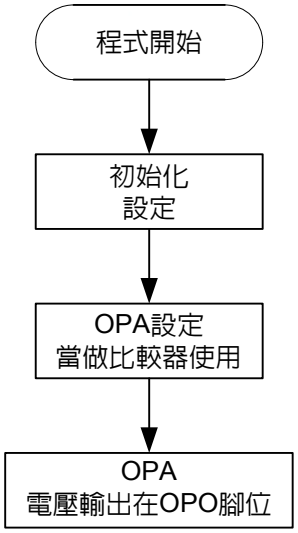
11.1. 範例名稱

HY16F3981_OPA

11.2. 範例說明

(1) OPAMP 使用與設定方式
(2)將類比電壓 REFO=1.2V 打開
(3)程式選擇 REFO 為 OPAMP 的正端, OPAMP 的負端選擇 AIO5.
(4)將 OPAMP 當作比較器使用，當 REFO 電壓大於 AIO5, OPO 腳位 PT3.7 會輸出 High，當 REFO 電壓小於 AIO5, OPO 腳位 PT3.7 會輸出 Low.
(5)每當 OPAMP 進入中斷一次, PT2.0 腳位會產生 High or Low 的狀態變化

11.3. 軟體流程



11.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_Style
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 OPA to do voltage compare, if REFO>AIO5, PT3.7=High, IF REFO<AIO5, PT3.7=Low
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------*/
/* Includes                                                            */
/*---------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "System.h"
#include "DrvPMU.h"
#include "DrvOP.h"
#include "DrvGPIO.h"
/*---------------------------------------------------------------------*/
/* STRUCTURES                                                          */
/*---------------------------------------------------------------------*/


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


/*---------------------------------------------------------------------*/
/* Global CONSTANTS                                                    */
/*---------------------------------------------------------------------*/
unsigned char i;


/*---------------------------------------------------------------------*/
/* Function PROTOTYPES                                                 */
/*---------------------------------------------------------------------*/
```

```
void Delay(unsigned int num);

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
  i=0;

  DrvPMU_VDDA_LDO_Ctrl(E_LDO);          //LDO ON
  DrvPMU_VDDA_Voltage(E_VDDA2_4);       //VDDA=2.4
  DrvPMU_REFO_Enable();                 //REFO ON

  DrvGPIO_Open(2,0x01,E_IO_OUTPUT);    //PT2.0 Output

  DrvOP_Open();
  DrvOP_PInput(0x08);                   //OPA positive reference input selection REFO
  DrvOP_NInput(0x02);                   //OPA negative reference input selection AIO5
  DrvOP_OPOoutEnable();                 //OPA Out Enable PT3.7. If REFO>AIO5, PT3.7=High, IF REFO<AIO5, PT3.7=Low
  DrvOP_OPDEN(1);                       //OPDEN=1b

  DrvOP_EnableInt();
  SYS_EnableGIE(4,0x3FF);               //Enable GIE(Global Interrupt)

  while(1)
  {
    //If REFO>AIO5, enter HW9_ISR()
  }

  return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description  : I2C/UART/SPI interrupt Service Routine (HW0).              */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description  : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).   */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description  : ADC interrupt Service Routine (HW2).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
```

```
/* Remark            :                                                    */
/*---------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                            */
/* Description    : PT3 interrupt Service Routine (HW4).               */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark            :                                                 */
/*---------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                            */
/* Description    : PT2 interrupt Service Routine (HW5).               */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark            :                                                 */
/*---------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                            */
/* Description    : UART2 interrupt Service Routine (HW7).             */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark            :                                                 */
/*---------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                            */
/* Description    : TMB2 interrupt Service Routine (HW8).              */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark            :                                                 */
/*---------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                            */
/* Description    : OPA interrupt Service Routine (HW9).               */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark            :                                                 */
/*---------------------------------------------------------------------*/
void HW9_ISR(void)
{
```

```
    if(DrvOP_ReadIntFlag())
    {
      if(i==ENABLE)
      {
        DrvGPIO_ClrPortBits(E_PT2,0);          //PT2.0 Output Low
        i=DISABLE;
      }
      else
      {
        DrvGPIO_SetPortBits(E_PT2,0);          //PT2.0 Output High
        i=ENABLE;
      }
        DrvOP_ClearIntFlag();                  //Clear OPA interrupt flag
    }
}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
    asm("nop");    //procedure define by customer.
    asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
    asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

12. 類比 IP(ADC)

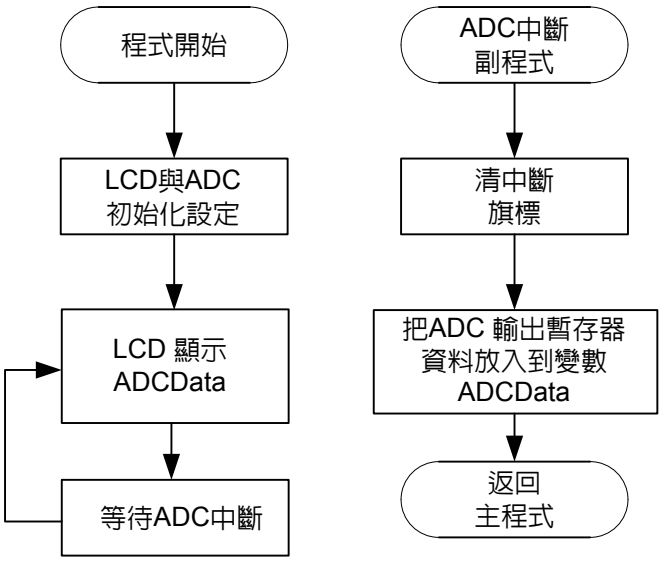
12.1. 範例名稱

HY16F3981_ADC

12.2. 範例說明

(1)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.
(2)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 32768, ADC 資料的輸出頻率為 30Hz(sps), 設置 ADC 的輸入端設置為 AIO2-AIO3, 設置 ADC 的參考電壓設置為 VDDA 對 VSS.
(3) ADC 初始設置完成後, 開啟 System 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 ADC 輸出暫存器資料顯示在 LCD 上.

12.3. 軟體流程



12.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_ADC
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : Showing high 16 bit ADCData on LCD display
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "System.h"
#include "DrvADC.h"
#include "DrvPMU.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_TMC1done:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
} MCUSTATUS;
```

```
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define HAO_2MHZ
#define HAO_4MHZ
//#define HAO_10MHZ
//#define HAO_16MHZ
//#define ADC_Chopper

/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
MCUSTATUS    MCUSTATUSbits;
int ADCData;
int ADCData_Array[2];
int ADCData_Chopper;
char i;
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void InitalADC(void);
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
   InitalADC();
   DisplayInit();
   ClearLCDframe();
   Delay(10000);
   DisplayHYcon();
   Delay(10000);
   SYS_EnableGIE(4,0x3FF);                  //Enable GIE(Global Interrupt)

   MCUSTATUSbits._byte = 0;

   while(1)
   {

#if defined(ADC_Chopper)
      for(i=0;i<=1;i++)
      {
         switch (i)
         {
            case 0:
            while(MCUSTATUSbits.b_ADCdone==0);
            MCUSTATUSbits.b_ADCdone=0;
            DrvADC_SetADCInputChannel(ADC_Input_AIO2,ADC_Input_AIO3);
            DrvADC_CombFilter(DISABLE);
            DrvADC_CombFilter(ENABLE);
            MCUSTATUSbits.b_ADCdone=0;
            while(MCUSTATUSbits.b_ADCdone==0);
            ADCData_Array[0]=ADCData>>16;
            MCUSTATUSbits.b_ADCdone=0;
            break;

            case 1:
            DrvADC_SetADCInputChannel(ADC_Input_AIO3,ADC_Input_AIO2);
```

```
            DrvADC_CombFilter(DISABLE);
            DrvADC_CombFilter(ENABLE);
            MCUSTATUSbits.b_ADCdone=0;
            while(MCUSTATUSbits.b_ADCdone==0);
            ADCData_Array[1]=ADCData>>16;
            ADCData_Chopper=(ADCData_Array[0]-ADCData_Array[1])>>1;
            LCD_DATA_DISPLAY(ADCData_Chopper);
            MCUSTATUSbits.b_ADCdone=0;
            break;
          }
        }
#else   //NO ADC_Chopper mode

        if(MCUSTATUSbits.b_ADCdone)
        {
          LCD_DATA_DISPLAY(ADCData>>16);      //16bits give up.
          MCUSTATUSbits.b_ADCdone=0;
        }
#endif

    }
    return 0;
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).            */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description    : ADC interrupt Service Routine (HW2).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{
    if(DrvADC_ReadIntFlag())                //Read ADC Flag
    {
        ADCData=DrvADC_GetConversionData();
        MCUSTATUSbits.b_ADCdone=1;
    }
```

```
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description    : PT3 interrupt Service Routine (HW4).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
```

```
/* Arguments       : None.                                                    */
/* Return Value : None.                                                    */
/* Remark          :                                                       */
/*-------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");   //procedure define by customer.
   asm("nop");
}


/*-------------------------------------------------------------------------*/
/* Function Name: InitalADC()                                              */
/* Description    : ADC Initialization Subroutines.                        */
/* Arguments       : None.                                                 */
/* Return Value : None.                                                    */
/* Remark          :                                                       */
/*-------------------------------------------------------------------------*/
void InitalADC(void)
{
   //Set ADC Clock
#if defined(HAO_2MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(0);                   //Select internal 2MHZ=HS_CK
   DrvADC_ClkEnable(2);                       //Setting ADC CLOCK ADCK=HS_CK/4
#endif
#if defined(HAO_4MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(1);                   //Select internal 4MHZ=HS_CK
   DrvADC_ClkEnable(2);                       //Setting ADC CLOCK ADCK=HS_CK/4
#endif
#if defined(HAO_10MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(2);                   //Select internal 10MHZ=HS_CK
   DrvADC_ClkEnable(3);                       //Setting ADC CLOCK ADCK=HS_CK/8
#endif
#if defined(HAO_16MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(3);                   //Select internal 16MHZ=HS_CK
   DrvADC_ClkEnable(4);                       //Setting ADC CLOCK ADCK=HS_CK/16
#endif

   //Set VDDA voltage
   DrvPMU_VDDA_LDO_Ctrl(E_LDO);
   DrvPMU_VDDA_Voltage(E_VDDA2_4);
   DrvPMU_BandgapEnable();
#if defined(HAO_2MHZ)
   Delay(0x18000);
#endif
#if defined(HAO_4MHZ)
   Delay(0x30000);
#endif
#if defined(HAO_10MHZ)
   Delay(0x60000);
#endif
#if defined(HAO_16MHZ)
   Delay(0xC0000);
#endif
   DrvPMU_AnalogGround(ENABLE);               //ADC analog ground source selection.
```

```
                                        //1 : Enable buffer and use internal source(need to work with ADC)

    //Set ADC input pin
    DrvADC_SetADCInputChannel(ADC_Input_AIO2,ADC_Input_AIO3); //Set the ADC positive/negative input voltage source.
    DrvADC_InputSwitch(OPEN);                  //ADC signal input (positive and negative) short(VISHR) control.
    DrvADC_RefInputShort(OPEN);                //Set the ADC reference input (positive and negative) short(VRSHR) control.
    DrvADC_ADGain(0);                          //ADC Gain=1
    DrvADC_DCoffset(0);                        //DC offset input voltage selection (VREF=REFP-REFN)
    DrvADC_RefVoltage(0,0);                    //Set the ADC reference voltage. VDDA-VSSA
    DrvADC_FullRefRange(1);                    //Set the ADC full reference range select.
                                               //0: Full reference range input
                                               //1: 1/2 reference range input
    DrvADC_OSR(0);                             //0 : OSR=32768

    //Set ADC interrupt
    DrvADC_EnableInt();
    DrvADC_Enable();
    DrvADC_CombFilter(ENABLE);                 //Enable comb filter

}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
    asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

13. 類比 IP(ADC_IA)

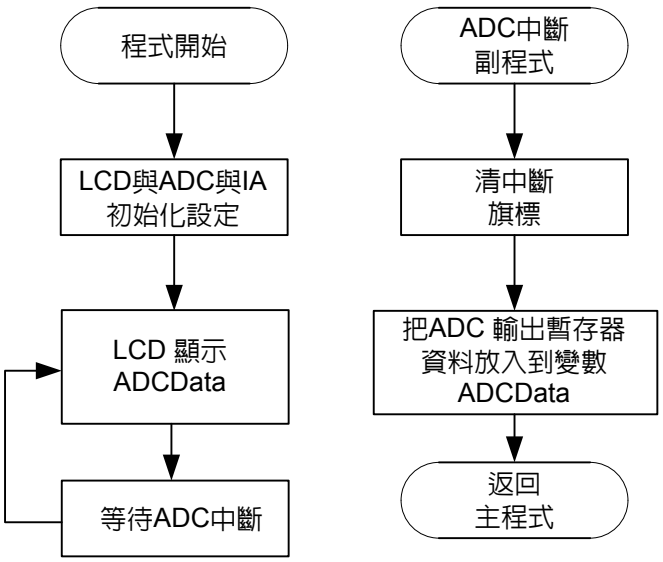
13.1. 範例名稱

HY16F3981_ADC_IA

13.2. 範例說明

(1)透過 ADC 與前級 IA 儀表放大器相關設定,可以實現 ADC 進入中斷抓取 ADC 輸出暫存器資料.
(2)ADC 與前級 IA 儀表放大器的初始設置包含, 設置 ADC 的類比電源為 VDDA, 設置 ADC 的取樣頻率 OSR 為 32768, ADC 資料的輸出頻率為 30Hz(sps), 設置 IA 的輸入端設置為 AIO0-AIO1, 設置 ADC 的參考電壓設置為 VDDA 對 VSS.
(3) ADC 與前級 IA 儀表放大器的初始設置完成後, 開啟 System 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 ADC 輸出暫存器資料顯示在 LCD 上.
(4)程式中開啟 REFO 電壓當做 BIAS 參考電壓, REFO 電壓為 1.2V. 硬體設置 : REFO 與 AIO1 短路. 外部輸入訊號到 AIO0-AIO1 由 IA 做訊號量測.
IA 可量測輸入電壓範圍 :+/- 1080mV

13.3. 軟體流程



13.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_ADC_IA
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : Showing high 16 bit ADCData on LCD display
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "System.h"
#include "DrvADC.h"
#include "DrvPMU.h"
#include "DrvIA.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_TMC1done:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
} MCUSTATUS;
```

```
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define HAO_2MHZ
#define HAO_4MHZ
//#define HAO_10MHZ
//#define HAO_16MHZ
#define SW_IA_ADC_Chopper
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
MCUSTATUS    MCUSTATUSbits;
int ADCData;
int ADCData_Array[2];
int ADCData_Chopper;
char i;
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void InitalADC(void);
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
   InitalADC();
   DisplayInit();
   ClearLCDframe();
   Delay(10000);
   DisplayHYcon();
   Delay(10000);
   SYS_EnableGIE(4,0x3FF);                  //Enable GIE(Global Interrupt)

   MCUSTATUSbits._byte = 0;

   while(1)
   {
#if defined(SW_IA_ADC_Chopper)
      for(i=0;i<=1;i++)
      {
         switch (i)
         {
            case 0:
            while(MCUSTATUSbits.b_ADCdone==0);
            MCUSTATUSbits.b_ADCdone=0;
            DrvADC_SetADCInputChannel(IA_Input_AIO0,IA_Input_AIO1);
            DrvADC_CombFilter(DISABLE);
            DrvADC_CombFilter(ENABLE);
            MCUSTATUSbits.b_ADCdone=0;
            while(MCUSTATUSbits.b_ADCdone==0);
            ADCData_Array[0]=ADCData>>16;
            MCUSTATUSbits.b_ADCdone=0;
            break;

            case 1:
            DrvADC_SetADCInputChannel(IA_Input_AIO1,IA_Input_AIO0);
            DrvADC_CombFilter(DISABLE);
```

```
            DrvADC_CombFilter(ENABLE);
            MCUSTATUSbits.b_ADCdone=0;
            while(MCUSTATUSbits.b_ADCdone==0);
            ADCData_Array[1]=ADCData>>16;
            ADCData_Chopper=(ADCData_Array[0]-ADCData_Array[1])>>1;
            LCD_DATA_DISPLAY(ADCData_Chopper);
            MCUSTATUSbits.b_ADCdone=0;
            break;
          }
        }
#else   //HW IA_ADC_Chopper mode

        if(MCUSTATUSbits.b_ADCdone)
        {
          LCD_DATA_DISPLAY(ADCData>>16);     //16bits give up.
          MCUSTATUSbits.b_ADCdone=0;
        }
#endif
    }
    return 0;
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{
    if(DrvADC_ReadIntFlag())                    //Read ADC Flag
    {
        ADCData=DrvADC_GetConversionData();
        MCUSTATUSbits.b_ADCdone=1;
    }
}
/*---------------------------------------------------------------------------*/
```

```
/* Function Name: HW4_ISR()                                                  */
/* Description  : PT3 interrupt Service Routine (HW4).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description  : PT2 interrupt Service Routine (HW5).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description  : UART2 interrupt Service Routine (HW7).                     */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description  : TMB2 interrupt Service Routine (HW8).                      */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description  : OPA interrupt Service Routine (HW9).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description  : Exception Service Routines.                                */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
```

```
/* Remark            :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}


/*---------------------------------------------------------------------------*/
/* Function Name: InitalADC()                                                */
/* Description    : ADC Initialization Subroutines.                          */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark            :                                                       */
/*---------------------------------------------------------------------------*/
void InitalADC(void)
{
   //Set ADC Clock
#if defined(HAO_2MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(0);                   //Select internal 2MHZ=HS_CK
   DrvADC_ClkEnable(2);                       //Setting ADC CLOCK ADCK=HS_CK/4
#endif
#if defined(HAO_4MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(1);                   //Select internal 4MHZ=HS_CK
   DrvADC_ClkEnable(2);                       //Setting ADC CLOCK ADCK=HS_CK/4
#endif
#if defined(HAO_10MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(2);                   //Select internal 10MHZ=HS_CK
   DrvADC_ClkEnable(3);                       //Setting ADC CLOCK ADCK=HS_CK/8
#endif
#if defined(HAO_16MHZ)
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);     //Select HSRC
   DrvCLOCK_SelectIHOSC(3);                   //Select internal 16MHZ=HS_CK
   DrvADC_ClkEnable(4);                       //Setting ADC CLOCK ADCK=HS_CK/16
#endif

   //Set VDDA voltage
   DrvPMU_VDDA_LDO_Ctrl(E_LDO);
   DrvPMU_VDDA_Voltage(E_VDDA2_4);
   DrvPMU_BandgapEnable();
   DrvPMU_REFO_Enable();
   Delay(0x1000);
   DrvPMU_AnalogGround(ENABLE);               //ADC analog ground source selection.
                                              //1 : Enable buffer and use internal source(need to work with ADC)

   //Set IA
   DrvIA_SetIAInputChannel(IA_Input_AIO0,IA_Input_AIO1);    //ADC positive=AIO0, negative=AIO1
   DrvIA_IAGain(IA_IAGain_4);                 //IA Gain=4
#if defined(SW_IA_ADC_Chopper)
   DrvIA_IACHM(IA_IACHM_NoChopper);           //IA Chopper Off
#else    //HW IA_ADC_Chopper mode
   DrvIA_IACHM(IA_IACHM_Both);                //IA Chopper On
#endif
   DrvIA_IAIS(0);                             //Input control=open
   DrvIA_ENIA(0);                             //Disable IA
```

```
    DrvIA_ENIA(1);                                  //Enable IA

    //Set ADC input pin
    DrvADC_SetADCInputChannel(OP_OP,OP_ON);    //Set the ADC positive/negative input voltage source.
    DrvADC_InputSwitch(OPEN);                  //ADC signal input (positive and negative) short(VISHR) control.
    DrvADC_RefInputShort(OPEN);                //Set the ADC reference input (positive and negative) short(VRSHR) control.
    DrvADC_ADGain(0);                            //ADC Gain=1
    DrvADC_DCoffset(0);                        //DC offset input voltage selection (VREF=REFP-REFN)
    DrvADC_RefVoltage(0,0);                    //Set the ADC reference voltage. VDDA-VSSA
    DrvADC_FullRefRange(1);                    //Set the ADC full reference range select.
                                                 //0: Full reference range input
                                                 //1: 1/2 reference range input
    DrvADC_OSR(0);                               //0 : OSR=32768

    //Set ADC interrupt
    DrvADC_EnableInt();
    DrvADC_Enable();
    DrvADC_CombFilter(ENABLE);                 //Enable comb filter

}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
    asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

14. 類比 IP(LCD)

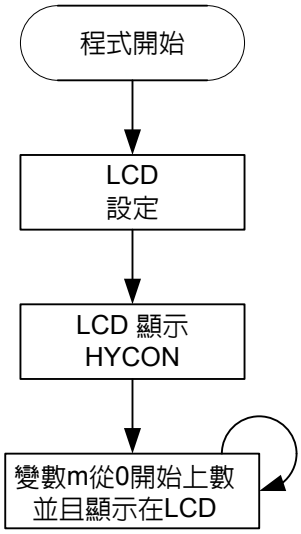
14.1. 範例名稱

HY16F3981_LCD

14.2. 範例說明

(1)LCD 使用方式說明
(2)開啟 LCD 並設定 VLCD 電壓及 Duty
(3)將部分 I/O 設定為 LCD Mode
(4)於 LCD 面板上顯示 HYCON 字樣
(5)HYCON 顯示後,會開始由 0 慢慢往上數, 數至 50000

14.3. 軟體流程



14.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_LCD
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 to do m++, and then LCD_DATA_DISPLAY(m)
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------*/
/* Includes                                                            */
/*---------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"

/*---------------------------------------------------------------------*/
/* STRUCTURES                                                          */
/*---------------------------------------------------------------------*/


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


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

unsigned int m;

/*---------------------------------------------------------------------*/
/* Function PROTOTYPES                                                 */
/*---------------------------------------------------------------------*/
void Delay(unsigned int num);
```

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

   DisplayInit();
   ClearLCDframe();
   DisplayHYcon();
   Delay(500000);
   for(m=0;m<50000;m++)
   {
       LCD_DATA_DISPLAY(m);
     Delay(65535);
   }
   return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description   : PT3 interrupt Service Routine (HW4).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
```

```
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
```

```
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

15. 通訊 IP(SPI)

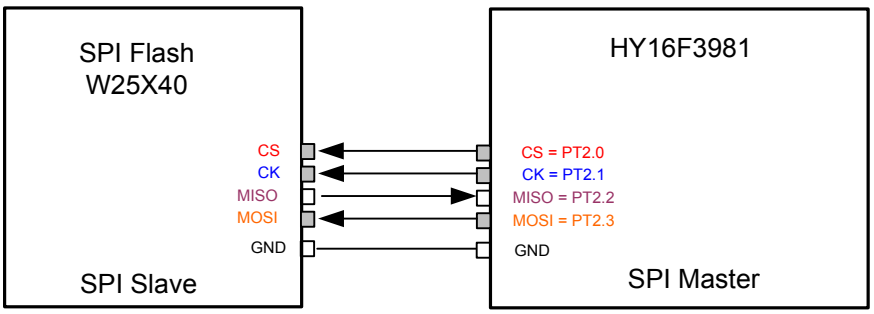
15.1. 範例名稱

HY16F3981_SPI

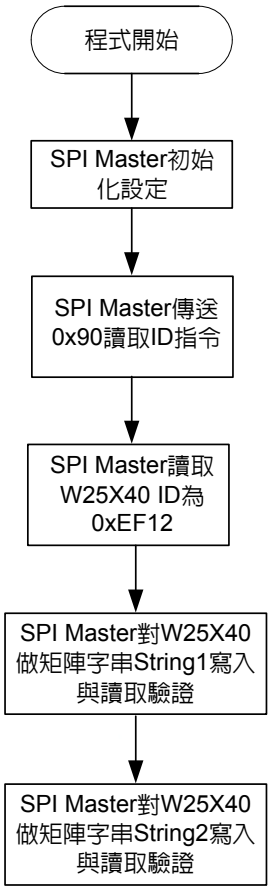
15.2. 範例說明

(1) HY16F3981 工作在三線式 SPI Master 來對 SPI Flash 25X40(Winbond)做寫入與讀取控制範例. 因為 SPI Master 為三線式，所以 CS(Chip Select)需要使用 GPIO 來做控制，設定 PT2.0=CS 功能。在完成 SPI 所有控制腳位的初始化之後，即可開始進行 SPI 控制動作
(2) 程式第一段測試為, SPI Flash Manufacturer and Device Identification 讀取確認,如果正確,變數 Flash_ID 會等於 0xEF12
(3) 程式第二段測試為, 矩陣字串 String1 寫入與讀取驗證測試, 如果寫入與讀取內容相同, LCD 會顯示 0, 如果不相同, LCD 會顯示 1
(3) 程式第三段測試為, 矩陣字串 String2 寫入與讀取驗證測試, 如果寫入與讀取內容相同, LCD 會顯示 0, 如果不相同, LCD 會顯示 123

15.3. 系統設定



15.4. 軟體流程


程式開始
SPI Master初始化設定
SPI Master傳送0x90讀取ID指令
SPI Master讀取W25X40 ID為0xEF12
SPI Master對W25X40做矩陣字串String1寫入與讀取驗證
SPI Master對W25X40做矩陣字串String2寫入與讀取驗證

15.5. 程式碼

說明 ： 在此僅貼上 main.c 程式內容做參考.

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_SPI
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 use SPI interface to communicate with SPI Flash(W25X40)
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------*/
/* Includes                                                            */
/*---------------------------------------------------------------------*/
#include "DrvREG32.h"
#include "SpecialMacro.h"
#include "stdint.h"
#include "HY16F3981.h"
#include "Display.h"
#include "System.h"
#include "DrvGPIO.h"
#include "my define.h"
#include "SPI_Flash.h"

/*---------------------------------------------------------------------*/
/* STRUCTURES                                                          */
/*---------------------------------------------------------------------*/


/*---------------------------------------------------------------------*/
/* DEFINITIONS                                                         */
/*---------------------------------------------------------------------*/
#define FlashAddress 0x000000

/*---------------------------------------------------------------------*/
/* Global CONSTANTS                                                    */
/*---------------------------------------------------------------------*/
unsigned char Flash_Write_Buffer[256];
unsigned char Flash_Read_Buffer[256];
unsigned short Flash_ID;
```

```
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);

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

   const unsigned char String1[]={
   "Hycon Technology was established in July, 2007 in Taipei, Taiwan. We dedicate ourselves to develop high precision and low drift analog
signal related processing ICs." };
   const unsigned char String2[]={
   "The identity stands for the vision and values of Hycon Technology- To be the ideal solution partner in the area of analog circuit." };
   unsigned short index_d,Size;

   InitalSPI();
   DisplayInit();
   ClearLCDframe();
   Delay(10000);
   DisplayHYcon();
   Delay(10000);

   Flash_ID=SpiFlash_ReadMidDid();                          //If correct, Flash_ID=0xEF12

   //Test1 : 16F3981 write String1 to Flash, and Read out
   Size=sizeof(String1);
   SpiFlash_ChipErase();                                    //Erase DATA
   Delay(256);
   SpiFlash_ReadData(Flash_Read_Buffer,FlashAddress,Size);          //Read DATA
   Delay(256);

   for( index_d=0;index_d<Size;index_d++)
   {
      Flash_Write_Buffer[index_d]=String1[index_d];
   }
   SpiFlash_PageProgram(Flash_Write_Buffer,FlashAddress,Size);    //Write DATA
   Delay(256);
   SpiFlash_ReadData(Flash_Read_Buffer,FlashAddress,Size);          //Read DATA
   Delay(256);

   for( index_d=0;index_d<Size;index_d++)
   {
      if(Flash_Read_Buffer[index_d]!=String1[index_d])
      {
         LCD_DATA_DISPLAY(1);                                   //If error, LCD Display "1"
         while(1);
      }
      else
         LCD_DATA_DISPLAY(0);                                   //If correct, LCD Display "0"
      }

   //Test2 : 16F3981 write String2 to Flash, and Read out
   Size=sizeof(String2);
   SpiFlash_SectorErase(0x00);                              //Erase DATA
   Delay(256);
   SpiFlash_ReadData(Flash_Read_Buffer,FlashAddress,Size);          //Read DATA
```

```
    Delay(256);

    for( index_d=0;index_d<Size;index_d++)
    {
       Flash_Write_Buffer[index_d]=String2[index_d];
    }
    SpiFlash_PageProgram(Flash_Write_Buffer,FlashAddress,Size);    //Write DATA
    Delay(256);
    SpiFlash_ReadData(Flash_Read_Buffer,FlashAddress,Size);          //Read DATA
    Delay(256);

    for( index_d=0;index_d<Size;index_d++)
    {
       if(Flash_Read_Buffer[index_d]!=String2[index_d])
       {
          LCD_DATA_DISPLAY(1);                                          //If error, LCD Display "1"
          while(1);
       }
       else
          LCD_DATA_DISPLAY(123);                                        //If correct, LCD Display "123"
       }

    while(1);

    return 0;

}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description   : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark           :                                                        */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{
```

```
}
/*---------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                            */
/* Description  : PT3 interrupt Service Routine (HW4).                 */
/* Arguments    : None.                                                */
/* Return Value : None.                                                */
/* Remark       :                                                      */
/*---------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                            */
/* Description  : PT2 interrupt Service Routine (HW5).                 */
/* Arguments    : None.                                                */
/* Return Value : None.                                                */
/* Remark       :                                                      */
/*---------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                            */
/* Description  : UART2 interrupt Service Routine (HW7).               */
/* Arguments    : None.                                                */
/* Return Value : None.                                                */
/* Remark       :                                                      */
/*---------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                            */
/* Description  : TMB2 interrupt Service Routine (HW8).                */
/* Arguments    : None.                                                */
/* Return Value : None.                                                */
/* Remark       :                                                      */
/*---------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                            */
/* Description  : OPA interrupt Service Routine (HW9).                 */
/* Arguments    : None.                                                */
/* Return Value : None.                                                */
/* Remark       :                                                      */
/*---------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                              */
```

```
/* Description    : Exception Service Routines.                                    */
/* Arguments      : None.                                                            */
/* Return Value : None.                                                            */
/* Remark           :                                                                 */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                      */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                                     */
/*---------------------------------------------------------------------------*/
```

16. 通訊 IP(UART)

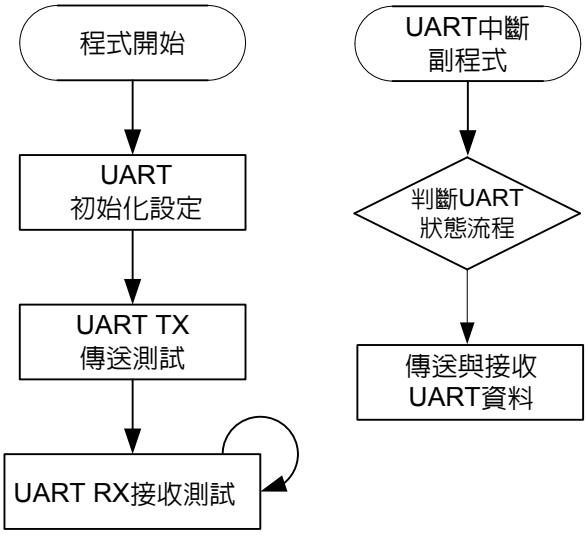
16.1. 範例名稱

HY16F3981_UART

16.2. 範例說明

(1)HY16F3981 使用 UART 做 TX 與 RX 傳輸範例程式
(2)程式開頭可以透過//#define PORT9 來選擇 UART Port 是要工作在 PT2 port 或著 PT9 port. 如果是 Port2 則 TX=PT2.0, RX=PT2.1. 如果是 Port9 則 TX=PT9.4, RX=PT9.5.
(3)程式開頭可以透過#define HAO_2MHZ/ HAO_4MHZ/ HAO_10MHZ/ HAO_16MHZ 來選擇晶片工作頻率
(4)程式開始執行, UART 會一直送字串’ ABCDEF’，直到收到’ abcedf’ 字串, 才會送出自串’ abcedf’ 並且 TX 停止傳送. 當收到不是字串’ abcedf’，則會再開始送’ ABCDEF’ 字串.

16.3. 軟體流程



16.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_UART
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : using #define HAO_2MHZ/ HAO_4MHZ/ HAO_10MHZ/ HAO_16MHZ to select HAO frequency.
 * UART TX continue transmit ‘ABCEDF’ until RX receiving ’ abcedf’ to stop transmit. If RX receiving not equal to ’ abcedf’ ,
UART TX start to transmit ‘ABCEDF’
 *
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvUART.h"
#include "DrvGPIO.h"
#include "System.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_TMC1done:1;
      unsigned b_RTCdone:1;
      unsigned b_UART_TxDone:1;
      unsigned b_UART_RxDone:1;
   };
```

```
} MCUSTATUS;

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define PORT9
#define HSRC    //Internal HAO
//#define HSXT    //External 4MHz

#define HAO_2MHZ
//#define HAO_4MHZ
//#define HAO_10MHZ
//#define HAO_16MHZ

#if defined(PORT9)
#define    UART_PORT E_PT9
#define    UART_TXD    BIT4
#define    UART_RXD    BIT5
#else
#define    UART_PORT E_PT2
#define    UART_TXD    BIT0
#define    UART_RXD    BIT1
#endif

#define    Uart_RX_BufferSize 6
#define    Uart_TX_BufferSize 8

/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char UartRxBuffer[Uart_RX_BufferSize]={0},UartTxBuffer[Uart_TX_BufferSize]={0};
unsigned char UartTxIndex,UartTxLength,UartRxIndex,UartRxLength;
MCUSTATUS    MCUSTATUSbits;

unsigned char UartRxBuffer_Command[Uart_RX_BufferSize]=
{
0x61,0x62,0x63,0x64,0x65,0x66 //ASCII    KEYWORD = abcdef
};
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void InitalUart(void);
void InitalClock(void);
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
   unsigned char Stop_To_Send_UART=DISABLE;
   InitalClock();
   InitalUart();
   MCUSTATUSbits._byte = 0;
   MCUSTATUSbits.b_UART_TxDone=ENABLE;
   SYS_EnableGIE(4,0x3FF);    //Enable GIE(Global Interrupt)

   UartTxIndex=0;
   UartRxIndex=0;
```

```
  while(1)
  {

    //UART RX
    if(MCUSTATUSbits.b_UART_RxDone==ENABLE)
    {

      if(
        (UartRxBuffer[5]==UartRxBuffer_Command[5] && UartRxBuffer[4]==UartRxBuffer_Command[4]) &&
        (UartRxBuffer[3]==UartRxBuffer_Command[3] && UartRxBuffer[2]==UartRxBuffer_Command[2]) &&
        (UartRxBuffer[1]==UartRxBuffer_Command[1] && UartRxBuffer[0]==UartRxBuffer_Command[0])
        )
    {
        //if UART receive == 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
        //send out 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f) and stop to send out
        Stop_To_Send_UART=ENABLE;    //UART stop to send out ABCDEF
        for(UartTxLength=0;UartTxLength<=Uart_TX_BufferSize;UartTxLength++)
        {
          UartTxBuffer[UartTxLength]=UartRxBuffer[UartTxLength];
          if(UartTxLength==Uart_RX_BufferSize)
          {
            UartRxIndex=0;
            MCUSTATUSbits.b_UART_TxDone=DISABLE;
            DrvUART_EnableInt(ENABLE,DISABLE);      //Enable UART Tx Interrupt, Disable UART Rx Interrup
            while(!MCUSTATUSbits.b_UART_TxDone); //If    MCUSTATUSbits.b_UART_TxDone=DISABLE, stop at here
          }
        }
    }
    else
    {
      //if UART receive != 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
      //User defined at here
      Stop_To_Send_UART=DISABLE;      //UART start to send out ABCDEF
    }
      UartRxIndex=0; //When finished the data reception. Set UartRxIndex=0
      MCUSTATUSbits.b_UART_RxDone=DISABLE;
    }

    //UART TX
    if(MCUSTATUSbits.b_UART_TxDone==ENABLE && Stop_To_Send_UART==DISABLE )
    {
      UartTxBuffer[7]='\r';
      UartTxBuffer[6]='\n';
      UartTxBuffer[5]=0x46;    //F
      UartTxBuffer[4]=0x45;    //E
      UartTxBuffer[3]=0x44;    //D
      UartTxBuffer[2]=0x43;    //C
      UartTxBuffer[1]=0x42;    //B
      UartTxBuffer[0]=0x41;    //A
      MCUSTATUSbits.b_UART_TxDone=DISABLE;
      UartTxLength=8;
      UartTxIndex=0;
      DrvUART_EnableInt(ENABLE,ENABLE);          //Enable UART Tx Interrupt;Enable UART Rx Interrupt
      while(!MCUSTATUSbits.b_UART_TxDone);    //If    MCUSTATUSbits.b_UART_TxDone=DISABLE, stop at here
    }

  }

}
```

```
/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).            */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{
  if(DrvUART_GetRxFlag())
  {
    UartRxBuffer[UartRxIndex]=DrvUART_Read();
    UartRxIndex++;
    if(UartRxIndex>=Uart_RX_BufferSize)
    {
      UartRxIndex=0;
      MCUSTATUSbits.b_UART_RxDone=ENABLE;
    }
      DrvUART_ClrRxFlag();
  }

  if(DrvUART_GetTxFlag())
  {
    if(MCUSTATUSbits.b_UART_TxDone==DISABLE)
    {
      DrvUART_Write(UartTxBuffer[UartTxIndex++]);
      DrvUART_ClrTxFlag();
      if(UartTxIndex>=UartTxLength)
      {
        DrvUART_EnableInt(ENABLE,ENABLE);   //ENABLE UART Tx Interrupt, ENABLE UART Rx Interrupt
        MCUSTATUSbits.b_UART_TxDone=ENABLE;
        UartTxIndex=0;
      }
    }
    if(MCUSTATUSbits.b_UART_TxDone==ENABLE)
    {
      DrvUART_EnableInt(DISABLE,ENABLE);   //DISABLE UART Tx Interrupt, ENABLE UART Rx Interrupt
      DrvUART_ClrTxFlag();
    }
  }

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description    : ADC interrupt Service Routine (HW2).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
```

```
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description  : PT3 interrupt Service Routine (HW4).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description  : PT2 interrupt Service Routine (HW5).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description  : UART2 interrupt Service Routine (HW7).                     */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description  : TMB2 interrupt Service Routine (HW8).                      */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description  : OPA interrupt Service Routine (HW9).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{
```

```
}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}

/*---------------------------------------------------------------------------*/
/* Function Name: InitalClock()                                              */
/* Description    : CLOCK Initial Subroutines.                               */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void InitalClock(void)
{

#if defined(HSRC)
   #if defined(HAO_2MHZ)
   //Clock INIT 2MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(0);                        //Select internal 2MHZ
   DrvCLOCK_SelectMCUClock(0,0);                   //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(0);                       //Calibration 1.843MHz
   #endif
   #if defined(HAO_4MHZ)
   //Clock INIT 4MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(1);                        //Select internal 4MHZ
   DrvCLOCK_SelectMCUClock(0,0);                   //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(1);                       //Calibration 4.147MHz
   #endif
   #if defined(HAO_10MHZ)
   //Clock INIT 10MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(2);                        //Select internal 10MHZ
   DrvCLOCK_SelectMCUClock(0,0);                   //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(2);                       //Calibration 9.216MHz
   #endif
   #if defined(HAO_16MHZ)
   //Clock INIT 16MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(3);                        //Select internal 16MHZ
   DrvCLOCK_SelectMCUClock(0,0);                   //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(3);                       //Calibration 15.667MHz
   #endif
#endif

#if defined(HSXT)
   DrvCLOCK_EnableHighOSC (E_EXTERNAL,50);
#endif
}
```

```
/*---------------------------------------------------------------------------*/
/* Function Name: InitalUart()                                               */
/* Description    : UART Initial Subroutines.                                */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void InitalUart(void)
{

#if defined(HSRC)
   DrvUART_ClkEnable(1,0);                  //Choose the internal HAO as clock source
   #if defined(PORT9)
      #if defined(HAO_2MHZ)
      DrvUART_Open(1843,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,7);
      #endif
      #if defined(HAO_4MHZ)
      DrvUART_Open(4147,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,7);
      #endif
      #if defined(HAO_10MHZ)
      DrvUART_Open(9216,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,7);
      #endif
      #if defined(HAO_16MHZ)
      DrvUART_Open(15667,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,7);
   #endif
   #else //PORT2
      #if defined(HAO_2MHZ)
      DrvUART_Open(1843,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
      #endif
      #if defined(HAO_4MHZ)
      DrvUART_Open(4147,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
      #endif
      #if defined(HAO_10MHZ)
      DrvUART_Open(9216,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
      #endif
      #if defined(HAO_16MHZ)
      DrvUART_Open(15667,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
      #endif
   #endif
#endif

   //1843 : oscillator frequency 2MHz Unit After Calibration HAO = 1843kHz
   //4147 : oscillator frequency 4MHz Unit After Calibration HAO = 4147kHz
   //9216 : oscillator frequency 10MHz Unit After Calibration HAO = 9216kHz
   //15667 : oscillator frequency 16MHz Unit After Calibration HAO = 15667kHz
   //None parity
   //8 data bits.
   //7 : Port 9.4 =TX, Port 9.5 =RX
   //2 : Port 2.0 =TX, Port 2.1 =RX

#if defined(HSXT)
   DrvUART_ClkEnable(0,0);                  //Choose the external 4MHz OSC as clock source
   DrvUART_Open(4000,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
#endif

#if defined(PORT9)
   DrvGPIO_LCDIOOpen(UART_PORT, UART_TXD, E_IO_OUTPUT);
   DrvGPIO_LCDIOOpen(UART_PORT, UART_RXD, E_IO_INPUT);
```

```
#else
   DrvGPIO_Open(UART_PORT,UART_TXD,E_IO_OUTPUT);
   DrvGPIO_Open(UART_PORT,UART_RXD,E_IO_INPUT);
   DrvGPIO_Open(UART_PORT,UART_RXD|UART_TXD,E_IO_PullHigh);
#endif

   DrvGPIO_ClkGenerator(E_HS_CK,1);
   DrvUART_EnableInt(ENABLE,ENABLE); //Enable UART Tx Interrupt, Enable UART Rx Interrupt
   DrvUART_Disable_AutoBaudrate();
   DrvUART_Close();
   DrvUART_Enable();
}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

17. 通訊 IP(UART2)

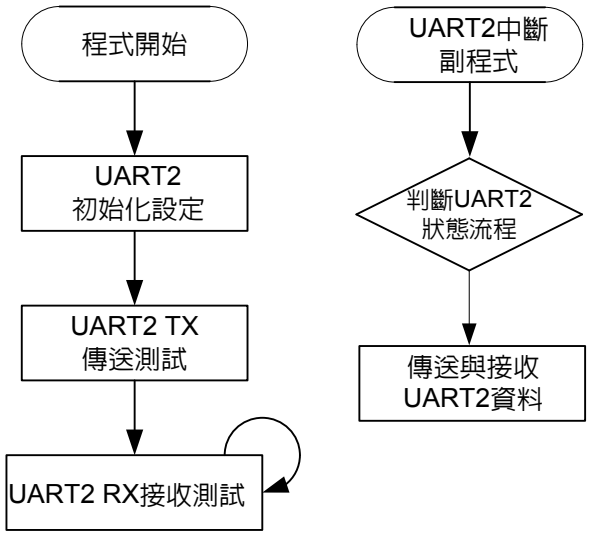
17.1. 範例名稱

HY16F3981_UART2

17.2. 範例說明

(1)HY16F3981 使用 UART2 做 TX 與 RX 傳輸範例程式
(2)程式開頭可以透過//#define PORT9 來選擇 UART2 Port 是要工作在 PT2 port 或著 PT9 port. 如果是 Port2 則 TX=PT2.2, RX=PT2.3. 如果是 Port9 則 TX=PT9.2, RX=PT9.3.
(3)程式開頭可以透過#define HAO_2MHZ/ HAO_4MHZ/ HAO_10MHZ/ HAO_16MHZ 來選擇晶片工作頻率
(4)程式開始執行, UART2 會一直送字串’ ABCDEF’，直到收到’ abcedf’ 字串, 才會送出自串’ abcedf’ 並且 TX 停止傳送. 當收到不是字串’ abcedf’，則會再開始送’ ABCDEF’ 字串.

17.3. 軟體流程



17.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_UART2
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : using #define HAO_2MHZ/ HAO_4MHZ/ HAO_10MHZ/ HAO_16MHZ to select HAO frequency.
 * UART TX continue transmit ‘ABCEDF’ until RX receiving ’ abcedf’ to stop transmit. If RX receiving not equal to ’ abcedf’ ,
UART TX start to transmit ‘ABCEDF’
 *
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "System.h"
#include "DrvGPIO.h"
#include "DrvUART.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
volatile typedef union _MCUSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_ADCdone:1;
      unsigned b_TMAdone:1;
      unsigned b_TMBdone:1;
      unsigned b_TMC0done:1;
      unsigned b_TMC1done:1;
      unsigned b_RTCdone:1;
      unsigned b_UART2_TxDone:1;
      unsigned b_UART2_RxDone:1;
```

```
    };
} MCUSTATUS;


/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define PORT9
#define HSRC    //Internal HAO
//#define HSXT    //External 4MHz

#define HAO_2MHZ
//#define HAO_4MHZ
//#define HAO_10MHZ
//#define HAO_16MHZ


#if defined(PORT9)
#define    UART2_PORT E_PT9
#define    UART2_TXD    BIT2
#define    UART2_RXD    BIT3
#else
#define    UART2_PORT E_PT2
#define    UART2_TXD    BIT2
#define    UART2_RXD    BIT3
#endif

#define    Uart2_RX_BufferSize 6
#define    Uart2_TX_BufferSize 8


/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char Uart2RxBuffer[Uart2_RX_BufferSize]={0},Uart2TxBuffer[Uart2_TX_BufferSize]={0};
unsigned char Uart2TxIndex,Uart2TxLength,Uart2RxIndex,Uart2RxLength;
MCUSTATUS    MCUSTATUSbits;

unsigned char Uart2RxBuffer_Command[Uart2_RX_BufferSize]=
{
0x61,0x62,0x63,0x64,0x65,0x66 //ASCII    KEYWORD = abcdef
};
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);
void InitalUart2(void);
void InitalClock(void);
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{

    unsigned char Stop_To_Send_UART2=DISABLE;
    InitalClock();
    InitalUart2();
    MCUSTATUSbits._byte = 0;
    MCUSTATUSbits.b_UART2_TxDone=ENABLE;
```

```
SYS_EnableGIE(4,0x3FF);    //Enable GIE(Global Interrupt)

Uart2TxIndex=0;
Uart2RxIndex=0;

while(1)
{

  //UART2 RX
  if(MCUSTATUSbits.b_UART2_RxDone==ENABLE)
  {

    if(
      (Uart2RxBuffer[5]==Uart2RxBuffer_Command[5] && Uart2RxBuffer[4]==Uart2RxBuffer_Command[4]) &&
      (Uart2RxBuffer[3]==Uart2RxBuffer_Command[3] && Uart2RxBuffer[2]==Uart2RxBuffer_Command[2]) &&
      (Uart2RxBuffer[1]==Uart2RxBuffer_Command[1] && Uart2RxBuffer[0]==Uart2RxBuffer_Command[0])
      )
  {
      //if UART2 receive == 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
      //send out 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f) and stop to send out
      Stop_To_Send_UART2=ENABLE;    //UART stop to send out ABCDEF
      for(Uart2TxLength=0;Uart2TxLength<=Uart2_TX_BufferSize;Uart2TxLength++)
      {
        Uart2TxBuffer[Uart2TxLength]=Uart2RxBuffer[Uart2TxLength];
        if(Uart2TxLength==Uart2_RX_BufferSize)
        {
          Uart2RxIndex=0;
          MCUSTATUSbits.b_UART2_TxDone=DISABLE;
          DrvUART2_EnableInt(ENABLE,DISABLE);     //Enable UART2 Tx Interrupt, Disable UART2 Rx Interrup
          while(!MCUSTATUSbits.b_UART2_TxDone); //If    MCUSTATUSbits.b_UART2_TxDone=DISABLE, stop at here
        }
      }
  }
  else
  {
    //if UART2 receive != 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
    //User defined at here
    Stop_To_Send_UART2=DISABLE;     //UART2 start to send out ABCDEF
  }
    Uart2RxIndex=0; //When finished the data reception. Set Uart2RxIndex=0
    MCUSTATUSbits.b_UART2_RxDone=DISABLE;
  }

  //UART2 TX
  if(MCUSTATUSbits.b_UART2_TxDone==ENABLE && Stop_To_Send_UART2==DISABLE )
  {
    Uart2TxBuffer[7]='\r';
    Uart2TxBuffer[6]='\n';
    Uart2TxBuffer[5]=0x46;    //F
    Uart2TxBuffer[4]=0x45;    //E
    Uart2TxBuffer[3]=0x44;    //D
    Uart2TxBuffer[2]=0x43;    //C
    Uart2TxBuffer[1]=0x42;    //B
    Uart2TxBuffer[0]=0x41;    //A
    MCUSTATUSbits.b_UART2_TxDone=DISABLE;
    Uart2TxLength=8;
    Uart2TxIndex=0;
    DrvUART2_EnableInt(ENABLE,ENABLE);         //Enable UART2 Tx Interrupt;Enable UART2 Rx Interrupt
    while(!MCUSTATUSbits.b_UART2_TxDone);    //If    MCUSTATUSbits.b_UART2_TxDone=DISABLE, stop at here
```

```
      }

   }

}

/*---------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                            */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).      */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark         :                                                    */
/*---------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                            */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1). */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark         :                                                    */
/*---------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                            */
/* Description    : ADC interrupt Service Routine (HW2).               */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark         :                                                    */
/*---------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                            */
/* Description    : PT3 interrupt Service Routine (HW4).               */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark         :                                                    */
/*---------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                            */
/* Description    : PT2 interrupt Service Routine (HW5).               */
/* Arguments      : None.                                              */
/* Return Value : None.                                                */
/* Remark         :                                                    */
/*---------------------------------------------------------------------*/
void HW5_ISR(void)
{
```

```
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

  if(DrvUART2_GetRxFlag())
  {
    Uart2RxBuffer[Uart2RxIndex]=DrvUART2_Read();
    Uart2RxIndex++;
    if(Uart2RxIndex>=Uart2_RX_BufferSize)
    {
      Uart2RxIndex=0;
      MCUSTATUSbits.b_UART2_RxDone=ENABLE;
    }
      DrvUART2_ClrRxFlag();
  }

  if(DrvUART2_GetTxFlag())
  {
    if(MCUSTATUSbits.b_UART2_TxDone==DISABLE)
    {
      DrvUART2_Write(Uart2TxBuffer[Uart2TxIndex++]);
      DrvUART2_ClrTxFlag();
      if(Uart2TxIndex>=Uart2TxLength)
      {
        DrvUART2_EnableInt(ENABLE,ENABLE);   //ENABLE UART2 Tx Interrupt, ENABLE UART2 Rx Interrupt
        MCUSTATUSbits.b_UART2_TxDone=ENABLE;
        Uart2TxIndex=0;
      }
    }
      if(MCUSTATUSbits.b_UART2_TxDone==ENABLE)
      {
        DrvUART2_EnableInt(DISABLE,ENABLE);   //DISABLE UART2 Tx Interrupt, ENABLE UART2 Rx Interrupt
        DrvUART2_ClrTxFlag();
      }

  }


}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

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

```
/* Function Name: HW9_ISR()                                                    */
/* Description  : OPA interrupt Service Routine (HW9).                         */
/* Arguments    : None.                                                        */
/* Return Value : None.                                                        */
/* Remark       :                                                              */
/*-----------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*-----------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                      */
/* Description  : Exception Service Routines.                                  */
/* Arguments    : None.                                                        */
/* Return Value : None.                                                        */
/* Remark       :                                                              */
/*-----------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");   //procedure define by customer.
   asm("nop");
}

/*-----------------------------------------------------------------------------*/
/* Function Name: InitalClock()                                                */
/* Description  : CLOCK Initial Subroutines.                                   */
/* Arguments    : None.                                                        */
/* Return Value : None.                                                        */
/* Remark       :                                                              */
/*-----------------------------------------------------------------------------*/
void InitalClock(void)
{

#if defined(HSRC)
   #if defined(HAO_2MHZ)
   //Clock INIT 2MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(0);                       //Select internal 2MHZ
   DrvCLOCK_SelectMCUClock(0,0);                  //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(0);                           //Calibration 1.843MHz
   #endif
   #if defined(HAO_4MHZ)
   //Clock INIT 4MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(1);                       //Select internal 4MHZ
   DrvCLOCK_SelectMCUClock(0,0);                  //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(1);                           //Calibration 4.147MHz
   #endif
   #if defined(HAO_10MHZ)
   //Clock INIT 10MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(2);                       //Select internal 10MHZ
   DrvCLOCK_SelectMCUClock(0,0);                  //CPU CLOCK IS 'hs_ck/1'
   DrvCLOCK_CalibrateHAO(2);                           //Calibration 9.216MHz
   #endif
   #if defined(HAO_16MHZ)
   //Clock INIT 16MHZ
   DrvCLOCK_EnableHighOSC(E_INTERNAL,10);          //Select HSRC
   DrvCLOCK_SelectIHOSC(3);                       //Select internal 16MHZ
```

```
    DrvCLOCK_SelectMCUClock(0,0);                    //CPU CLOCK IS 'hs_ck/1'
    DrvCLOCK_CalibrateHAO(3);                        //Calibration 15.667MHz
    #endif
#endif

#if defined(HSXT)
    DrvCLOCK_EnableHighOSC (E_EXTERNAL,50);
#endif
}

/*---------------------------------------------------------------------------*/
/* Function Name: InitalUart2()                                              */
/* Description    : UART2 Initial Subroutines.                               */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void InitalUart2(void)
{

#if defined(HSRC)
    DrvUART2_ClkEnable(1,0);                 //Choose the internal HAO as clock source
    #if defined(PORT9)
        #if defined(HAO_2MHZ)
        DrvUART2_Open(1843,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,6);
        #endif
        #if defined(HAO_4MHZ)
        DrvUART2_Open(4147,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,6);
        #endif
        #if defined(HAO_10MHZ)
        DrvUART2_Open(9216,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,6);
        #endif
        #if defined(HAO_16MHZ)
        DrvUART2_Open(15667,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,6);
    #endif
    #else //PORT2
        #if defined(HAO_2MHZ)
        DrvUART2_Open(1843,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
        #endif
        #if defined(HAO_4MHZ)
        DrvUART2_Open(4147,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
        #endif
        #if defined(HAO_10MHZ)
        DrvUART2_Open(9216,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
        #endif
        #if defined(HAO_16MHZ)
        DrvUART2_Open(15667,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
        #endif
    #endif
#endif
    //1843 : oscillator frequency 4MHz Unit After Calibration HAO = 1843kHz
    //4147 : oscillator frequency 4MHz Unit After Calibration HAO = 4147kHz
    //9216 : oscillator frequency 4MHz Unit After Calibration HAO = 9216kHz
    //15667 : oscillator frequency 4MHz Unit After Calibration HAO = 15667kHz
    //None parity
    //8 data bits.
    //6 : Port 9.2 =TX, Port 9.3 =RX
    //2 : Port 2.2 =TX, Port 2.3 =RX
```

```
#if defined(HSXT)
   DrvUART2_ClkEnable(0,0);                    //Choose the external OSC as clock source
   DrvUART2_Open(4000,B9600,DRVUART_PARITY_NONE,DRVUART_DATABITS_8,DRVUART_STOPBITS_1,2);
#endif

#if defined(PORT9)
   DrvGPIO_LCDIOOpen(UART2_PORT, UART2_TXD, E_IO_OUTPUT);
   DrvGPIO_LCDIOOpen(UART2_PORT, UART2_RXD, E_IO_INPUT);

#else
   DrvGPIO_Open(UART2_PORT,UART2_TXD,E_IO_OUTPUT);
   DrvGPIO_Open(UART2_PORT,UART2_RXD,E_IO_INPUT);
   DrvGPIO_Open(UART2_PORT,UART2_RXD|UART2_TXD,E_IO_PullHigh);
#endif

   DrvUART2_EnableInt(ENABLE,ENABLE); //ENABLE UART2 Tx Interrupt, ENABLE UART2 Rx Interrupt
   DrvUART2_Disable_AutoBaudrate();
   DrvUART2_Close();
   DrvUART2_Enable();
}


/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

18. 通訊 IP(I2C)

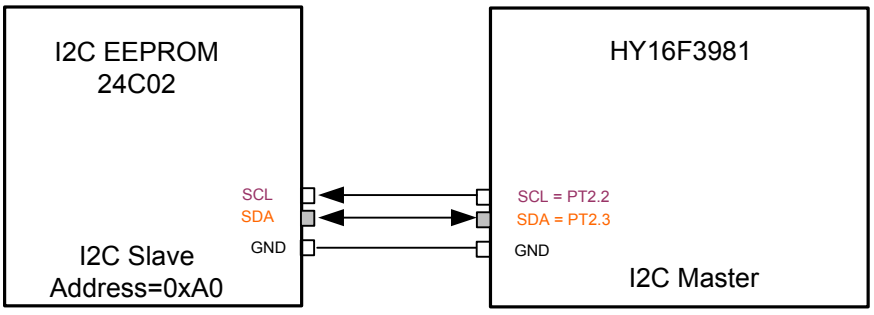
18.1. 範例名稱

HY16F3981_I2C

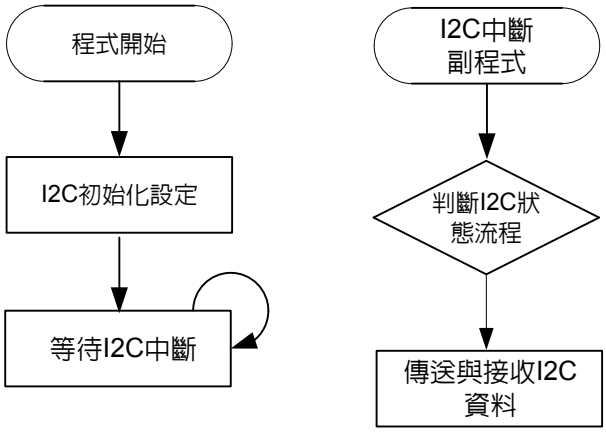
18.2. 範例說明

(1)HY16F3981 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 HY16F3981 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 0x01 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 連續做 5 bytes 的資料讀取直到 0x04，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 0x01，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。

18.3. 系統設定



18.4. 軟體流程



18.5. 程式碼

說明 ： 在此僅貼上 main.c 程式內容做參考.

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ------------------------------------------------------------------------
 * Project Name : HY16F3981_I2C
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    : HY16F3981 use I2C interface to communicate with I2C EEPROM(24Cxx)
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *
 * HY16F3981
 * ---------
 *               |                        --------
 *       PT2.3 | SDA <--> SDA |EEPROM|
 *       PT2.2 | SCK ---> SCK --------
 *         GND   |
 *               |
```

```
 * ---------
 ******************************************************************************/
/*---------------------------------------------------------------------*/
/* Includes                                                            */
/*---------------------------------------------------------------------*/
#include "DrvI2C.h"
#include "System.h"
#include "EEPROM_24Cxx.h"
#include "my define.h"
#include "HY16F3981.h"

/*---------------------------------------------------------------------*/
/* STRUCTURES                                                          */
/*---------------------------------------------------------------------*/


/*---------------------------------------------------------------------*/
/* DEFINITIONS                                                         */
/*---------------------------------------------------------------------*/
#define   I2C_PORT E_PT2
#define   SCL_PIN 2
#define   SDA_PIN 3
#define   SCL_BIT BIT2
#define   SDA_BIT BIT3
#define   I2CBufferSize   256
#define I2C_WRITE     0x00   // I2C WRITE command
#define I2C_READ      0x01   // I2C READ command

/*---------------------------------------------------------------------*/
/* Global CONSTANTS                                                    */
/*---------------------------------------------------------------------*/
unsigned char I2C_Read_Buffer[I2CBufferSize];;
unsigned char I2C_RW;
unsigned char I2C_TARGET;         // Target I2C slave address
unsigned char I2C_Sendbuf[I2CBufferSize];
unsigned char I2C_Recbuf[I2CBufferSize];
unsigned char I2C_EndFlag;
unsigned int I2C_DataTxLen,I2C_DataTxIndex,I2C_DataRxLen,I2C_DataRxIndex;

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

/*---------------------------------------------------------------------*/
/* Main Function                                                       */
/*---------------------------------------------------------------------*/
int main(void)
{
   unsigned int i;
   unsigned char EEPROM_WriteData[4] = {0x02,0x03,0x04,0x05};

   for(i=0;i<=I2CBufferSize;i++)
   {
      I2C_Read_Buffer[i]=0;          //Initial I2C data buffer=0
   }

   DrvI2C_SetIOPin(5);               //Setting io pin, 5: SCL=PT2.2;SDA=PT2.3
   DrvI2C_Open(0x63);                 //Enable I2C function and set I2C baud rate=5kHz
```

```
    //Default CPU clock is 2MHz, Data Baud Rate : (I2CLK)/[4x(CRG+1)]= 2000000/[4*(99+1)]=5kHz
    DrvI2C_EnableInt(2);                //Enable I2C interrupt and error interrupt

    SYS_EnableGIE(4,0x3FF);             //Enable GIE(Global Interrupt)

    // I2C single I2C_WRITE & I2C_READ
    EEPROM_ByteWrite(0x00,0x01); //Single Byte I2C_WRITE word address 0x00, and I2C_WRITE data 0x01
    Delay(1000);                        //Delay for EEPROM 24C02 I2C_WRITE Cycle Time 5ms
    I2C_Read_Buffer[0]=EEPROM_ByteRead(0x00);   //Single Byte I2C_READ word address 0x00, the I2C_READ value is 0x01
    Delay(1000);                        //Delay for EEPROM 24C02 I2C_WRITE Cycle Time 5ms

    // I2C sequential I2C_WRITE & I2C_READ
    EEPROM_WriteArray(0x01,EEPROM_WriteData,4);   //I2C_WRITE array data to the word address from 0x01 to 0x04
    Delay(1000);                        //Delay for EEPROM 24C02 I2C_WRITE Cycle Time 5ms
    EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 5);   //sequential I2C_READ data from 0x00 to 0x04
    Delay(1000);                        //Delay for EEPROM 24C02 I2C_WRITE Cycle Time 5ms

    while(1);

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).            */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{
    unsigned char I2C_Status,I2C_IntFlag;

    I2C_IntFlag=DrvI2C_ReadIntFlag();
    if((I2C_IntFlag == E_DRVI2C_INT)||(I2C_IntFlag == E_DRVI2C_INT_ALL))    //Get I2C Interrupt Flag
    {
        I2C_Status=DrvI2C_GetStatusFlag();          //Get I2C Status Flag
        switch(I2C_Status)
        {
            case 0x90:    //MACTFlag+RWFlag
                {             //START has been transmitted
                    DrvI2C_WriteData(I2C_TARGET);    //Send Slave Address & R/W Bit
                    DrvI2C_Ctrl(0,0,0,0);                 //Clear all I2C flag
                    break;
                };
            case 0x84:    //MACTFlag+ACKFlag
                {             //Slave A + W has been transmitted. ACK has been received.
                    DrvI2C_WriteData(I2C_Sendbuf[I2C_DataTxIndex++]);    //Send Data to Slave
                    DrvI2C_Ctrl(0,0,0,0);           //Clear all I2C flag
                    break;
                };
            case 0x80: //MACTFlag
                {           //Slave A + W has been transmitted. ACK has been received.
                    DrvI2C_WriteData(I2C_Sendbuf[I2C_DataTxIndex++]);    //Send Data to Slave
                    DrvI2C_Ctrl(0,0,0,0);           //Clear all I2C flag
                    break;
                };
            case 0x30:
                {
                    DrvI2C_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)
            {
                DrvI2C_WriteData(I2C_Sendbuf[I2C_DataTxIndex++]);    //Send Data to Slave
                DrvI2C_Ctrl(0,0,0,0);        // Clear all I2C flag
            }
            else
            {
                if(I2C_RW == I2C_WRITE)
                {
                    DrvI2C_Ctrl(0,1,0,0);    //I2C as master sends STOP signal
                    I2C_EndFlag=1;
                }
                else if(I2C_RW == I2C_READ)
                    DrvI2C_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
            DrvI2C_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.
            DrvI2C_WriteData(I2C_TARGET | I2C_READ);    //Send Slave Address & R/W Bit
            DrvI2C_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)
            {
                DrvI2C_Ctrl(0,0,0,1);    //Set ACK bit
            }else{
                DrvI2C_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_Recbuf[I2C_DataRxIndex++]=DrvI2C_ReadData();
            if((I2C_DataRxLen-1)>I2C_DataRxIndex)
            {
                DrvI2C_Ctrl(0,0,0,1);    //Set ACK bit
            }
            else
            {
                DrvI2C_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_Recbuf[I2C_DataRxIndex++]=DrvI2C_ReadData();
              DrvI2C_Ctrl(0,1,0,0);   //I2C as master sends STOP signal
              I2C_EndFlag=1;
              break;
            };
        default:
            {
              DrvI2C_Ctrl(0,0,0,0);   //Clear all I2C flag
              I2C_EndFlag=1;
              break;
            };
      }
      DrvI2C_ClearIRQ();
      DrvI2C_ClearEIRQ();                 //Clear EIRQFlag
      DrvI2C_ClearIntFlag(0);             //Clear I2C Interrupt Flag(I2CIF)
    }
    if((I2C_IntFlag == E_DRVI2C_ERROR_INT)||(I2C_IntFlag == E_DRVI2C_INT_ALL))   //Get I2C Error Interrupt Flag
    {
      I2C_EndFlag=1;
      DrvI2C_ClearIRQ();
      DrvI2C_ClearEIRQ();                 //Clear EIRQFlag
      DrvI2C_ClearIntFlag(1);             //Clear I2C Interrupt Flag(I2CEIF)
      DrvI2C_Ctrl(0,0,0,0);               //Clear all I2C flag
    }
    SYS_EnableGIE(4,0x3FF);               //Enable GIE(Global Interrupt)
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description   : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description   : ADC interrupt Service Routine (HW2).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description   : PT3 interrupt Service Routine (HW4).                      */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark          :                                                         */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{
```

```
}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
```

```
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

19. 其他 IP(Power)

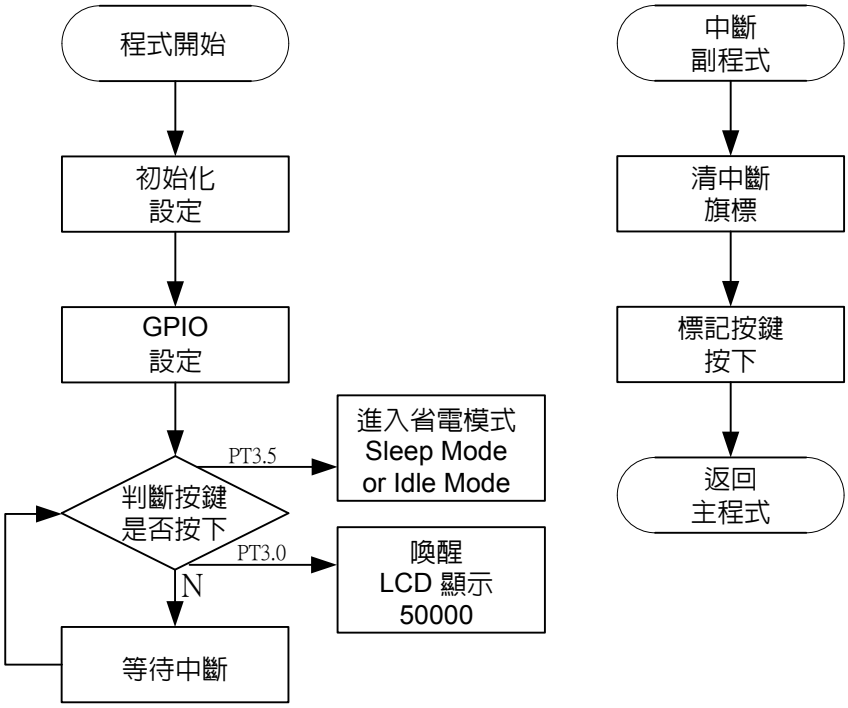
19.1. 範例名稱

HY16F3981_Power

19.2. 範例說明

(1) HY16F3981 進入 Sleep 與 Idle Mode 範例程式
(2)硬體連接部分，各腳位不浮接. 例如:腳位 EDIO 與 ECK 接地.
(3)上電後按下 PT3.0 喚醒並且 LCD 顯示 50000。
(4)上電後按下 PT3.5 進入 Idle Mode 或 Sleep Mode，進入 Idle Mode or Sleep Mode 是藉由程式開頭的#define IDLE_MODE 或#define SLEEP_MODE 來決定。
(5)此範例程式搭配 HY16F3981 硬體開發板，可測量到 Sleep 耗電流約為 2.5uA, Idle 耗電流約為 3.5uA.

19.3. 軟體流程



19.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_Power
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    :
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvPMU.h"
#include "System.h"
#include "DrvGPIO.h"

/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/
typedef union _PTINTSTATUS
{
   char    _byte;
   struct
   {
      unsigned b_PTINT0done:1;
      unsigned b_PTINT1done:1;
      unsigned b_PTINT2done:1;
      unsigned b_PTINT3done:1;
      unsigned b_PTINT4done:1;
      unsigned b_PTINT5done:1;
      unsigned b_PTINT6Done:1;
      unsigned b_PTINT7Done:1;
   };
} PTINTSTATUS;
```

```
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#define KEY_PORT E_PT3
#define KEYIN0 BIT0
#define KEYIN1 BIT5

//#define IDLE_MODE
#define SLEEP_MODE
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
PTINTSTATUS    PT3INTSTATUSbits;
unsigned int PT30_IDF;
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num);

/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
int main(void)
{
   DisplayInit();
   ClearLCDframe();
   DrvGPIO_ClkGenerator(E_HS_CK,1);                          //Set IO sampling clock input source is E_HS_CK
   DrvGPIO_Open(KEY_PORT,KEYIN1|KEYIN0,E_IO_INPUT);          //set PT3.0/PT3.5 INPUT
   DrvGPIO_Open(KEY_PORT,KEYIN1|KEYIN0,E_IO_PullHigh);       //enable PT3.0/PT3.5 pull high R
   DrvGPIO_Open(KEY_PORT,KEYIN1|KEYIN0,E_IO_IntEnable);      //PT3.0/PT3.5 interrupt enable
   DrvGPIO_IntTrigger(KEY_PORT,KEYIN1|KEYIN0,E_N_Edge);      //PT3.0/PT3.5 interrupt trigger method is negative edge
   DrvGPIO_ClearIntFlag(KEY_PORT,KEYIN1|KEYIN0);             //clear PT3 interrupt flag
   PT3INTSTATUSbits._byte = 0;
   SYS_EnableGIE(4,0x3FF);                                   //Enable GIE(Global Interrupt)

   while(1)
   {
      if(PT3INTSTATUSbits.b_PTINT0done)                      //if PT3.0 low, wake up
      {
         LCD_DATA_DISPLAY(50000);
         PT3INTSTATUSbits.b_PTINT0done=0;
      }

      if(PT3INTSTATUSbits.b_PTINT5done)                      //if PT3.5 low, Enter Sleep or Idle Mode
      {

         PT30_IDF=DrvGPIO_PortIDIF(E_PT3) & 0x01;            //check wake up pin, PT3.0 IDF status
         if(PT30_IDF==0x01)
         {
         //Enter Sleep or Idle Mode setting
         DrvLCD_VLCDMode(E_VLCD_DISABLE);
         while((inw(0x41B00)&(1<<IDF))==0);                  //Wait LCD Idle, IDF=20
         DrvPMU_LDO_LowPower(1);                             //SET low power mode
         DrvCLOCK_SelectMCUClock(E_LS_CK,1);                 //SET CPUCKL=LPO/2
         DrvCLOCK_CloseIHOSC();                              //Close HAO
         #if defined(IDLE_MODE)
         SYS_LowPower(1);                                    //Idle Mode, around 3.5uA
         #endif
         #if defined(SLEEP_MODE)
```

```
            SYS_LowPower(0);                                    //sleep mode, around 2.5uA
            #endif
            }
            //If wake up from idle mode, user can enable internal HAO first to do some application
            DrvPMU_LDO_LowPower(0);                             //SET normal power mode
            DrvCLOCK_EnableHighOSC(E_INTERNAL,1);
            DrvCLOCK_SelectMCUClock(E_HS_CK,0);                 //SET CPUCKL=HAO/1
            DrvGPIO_ClkGenerator(E_HS_CK,1);                    //Set IO sampling clock input source is E_HS_CK
            DisplayInit();
            PT3INTSTATUSbits.b_PTINT5done=0;
        }
    }

    return 0;
}

/*---------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                  */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).            */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                  */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description    : ADC interrupt Service Routine (HW2).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description    : PT3 interrupt Service Routine (HW4).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{
```

```
    uint32_t PORT_IntFlag;

    PORT_IntFlag=DrvGPIO_GetIntFlag(KEY_PORT);
    if((PORT_IntFlag&KEYIN0)==KEYIN0)
    {
       PT3INTSTATUSbits.b_PTINT0done=1;
    }
       if((PORT_IntFlag&KEYIN1)==KEYIN1)
    {
       PT3INTSTATUSbits.b_PTINT5done=1;
    }
    DrvGPIO_ClearIntFlag(KEY_PORT,KEYIN1|KEYIN0);          //clear PT3.0/PT3.5 interrupt flag

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description    : PT2 interrupt Service Routine (HW5).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description    : UART2 interrupt Service Routine (HW7).                   */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description    : TMB2 interrupt Service Routine (HW8).                    */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description    : OPA interrupt Service Routine (HW9).                     */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void HW9_ISR(void)
{

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

```
/* Function Name: tlb_exception_handler()                                    */
/* Description    : Exception Service Routines.                              */
/* Arguments      : None.                                                    */
/* Return Value : None.                                                      */
/* Remark         :                                                          */
/*---------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*---------------------------------------------------------------------------*/
/* End Of File                                                               */
/*---------------------------------------------------------------------------*/
```

20. 其他 IP(LVD)

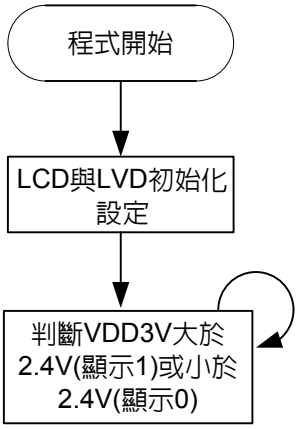
20.1. 範例名稱

HY16F3981_LVD

20.2. 範例說明

(1) HY16F3981 LVD 低電壓偵測範例程式
(2) LCD Display 初始化設置與 LVD 初始化設定, 開啟與設置 band gap 電壓當作參考電壓源.
(3)設置 LVD 低電壓為 2.4V, 當 VDD3V 電壓小於 2.4V 則 LCD Display 會顯示”0”, 如果 VDD3V 電壓高於 2.4V 則 LCD Display 會顯示”1”

20.3. 軟體流程



20.4. 程式說明

```
/*******************************************************************************
 *
 * Copyright (c) 2016-2026 HYCON Technology, Inc.
 * All rights reserved.
 * HYCON Technology <www.hycontek.com>
 *
 * HYCON reserves the right to amend this code without notice at any time.
 * HYCON assumes no responsibility for any errors appeared in the code,
 * and HYCON disclaims any express or implied warranty, relating to sale
 * and/or use of this code including liability or warranties relating
 * to fitness for a particular purpose, or infringement of any patent,
 * copyright or other intellectual property right.
 *
 * ----------------------------------------------------------------------
 * Project Name : HY16F3981_LVD
 * IDE tooling    : AndeSight C/C++ IDE, version: 2.1.1 Build ID : 201608241332
 * Device Ver.    : HY16F_RDSp3_DeviceV0.2 crt0.o for HY16F3981 MCU.
 * Library Ver. : 0.2
 * MCU Device     :
 * Description    :
 * Created Date : 2018/3/9
 * Created by     : Robert.Wang
 *
 * Program Description:
 * --------------------
 *******************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include "DrvCLOCK.h"
#include "Display.h"
#include "my define.h"
#include "HY16F3981.h"
#include "DrvPMU.h"
/*---------------------------------------------------------------------------*/
/* STRUCTURES                                                                */
/*---------------------------------------------------------------------------*/


/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#define VOLTAGE_High    1
#define VOLTAGE_LOW     0

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


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

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

```
/* Main Function                                                              */
/*----------------------------------------------------------------------------*/
int main(void)
{
   unsigned char LVDO_JUDGEMENT;

   DisplayInit();
   ClearLCDframe();
   Delay(10000);
   DisplayHYcon();

   //Set bandgap voltage
   DrvPMU_VDDA_LDO_Ctrl(E_LDO);
   DrvPMU_VDDA_Voltage(E_VDDA2_4);
   DrvPMU_BandgapEnable();

   //Set LVD
   DrvPMU_SetLVDS(LVD_24V);
   DrvPMU_SetLVD12(LVD_V12_BGR);
   DrvPMU_SetLVDVS(LVD_VDD3V);
   DrvPMU_EnableENLVD();

   while(1)
   {
      LVDO_JUDGEMENT=DrvPMU_GetLVDO();
      if(LVDO_JUDGEMENT==VOLTAGE_High)
      {
          LCD_DATA_DISPLAY(VOLTAGE_High);    //VDD3V>LVDS
      }
      if(LVDO_JUDGEMENT==VOLTAGE_LOW)
      {
          LCD_DATA_DISPLAY(VOLTAGE_LOW);     //VDD3V<LVDS
      }
   }

}

/*----------------------------------------------------------------------------*/
/* Function Name: HW0_ISR()                                                   */
/* Description    : I2C/UART/SPI interrupt Service Routine (HW0).             */
/* Arguments      : None.                                                     */
/* Return Value : None.                                                       */
/* Remark         :                                                           */
/*----------------------------------------------------------------------------*/
void HW0_ISR(void)
{

}
/*----------------------------------------------------------------------------*/
/* Function Name: HW1_ISR()                                                   */
/* Description    : WDT & RTC & Timer A/B/C interrupt Service Routine (HW1).  */
/* Arguments      : None.                                                     */
/* Return Value : None.                                                       */
/* Remark         :                                                           */
/*----------------------------------------------------------------------------*/
void HW1_ISR(void)
{

}
```

```
/*---------------------------------------------------------------------------*/
/* Function Name: HW2_ISR()                                                  */
/* Description  : ADC interrupt Service Routine (HW2).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW2_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW4_ISR()                                                  */
/* Description  : PT3 interrupt Service Routine (HW4).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW4_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW5_ISR()                                                  */
/* Description  : PT2 interrupt Service Routine (HW5).                       */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW5_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW7_ISR()                                                  */
/* Description  : UART2 interrupt Service Routine (HW7).                     */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW7_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW8_ISR()                                                  */
/* Description  : TMB2 interrupt Service Routine (HW8).                      */
/* Arguments    : None.                                                      */
/* Return Value : None.                                                      */
/* Remark       :                                                            */
/*---------------------------------------------------------------------------*/
void HW8_ISR(void)
{

}
/*---------------------------------------------------------------------------*/
/* Function Name: HW9_ISR()                                                  */
/* Description  : OPA interrupt Service Routine (HW9).                       */
/* Arguments    : None.                                                      */
```

```
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void HW9_ISR(void)
{

}
/*-------------------------------------------------------------------------*/
/* Function Name: tlb_exception_handler()                                  */
/* Description    : Exception Service Routines.                            */
/* Arguments      : None.                                                  */
/* Return Value : None.                                                    */
/* Remark       :                                                          */
/*-------------------------------------------------------------------------*/
void tlb_exception_handler()
{
   asm("nop");    //procedure define by customer.
   asm("nop");
}
/*-------------------------------------------------------------------------*/
/* Software Delay Subroutines                                              */
/*-------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
   for(;num>0;num--)
   asm("NOP");
}
/*-------------------------------------------------------------------------*/
/* End Of File                                                             */
/*-------------------------------------------------------------------------*/
```

21. 修訂記錄

以下描述本檔差異較大的地方，而標點符號與字形的改變不在此描述範圍。

日期	文件版次	頁次	摘要
2017/4/20	V01	All	初版發行
2018/7/13	V02	All	- HY16F3981_ADC : Add HAO 2MHz/4MHz/10MHz/16MHz clock setting. Add #define ADC_Chopper setting - HY16F3981_GPIO : Modify the GPIO initialization(GPIO setting-->Clear GPIO Flag-->judgement GPIO INT) - HY16F3981_I2C : Modify the I2C_EndFlag judgement - HY16F3981_UART : Modify the UART initialization, avoid to occur the first byte error issue. Modify the demo code application. - HY16F3981_UART2 : Add the UART2 demo code - HY16F3981_LVD : Add the LVD demo code - HY16F3981_Power : Modify the GPIO initialization(GPIO setting-->Clear GPIO Flag-->judgement GPIO INT) Added the DrvGPIO_PortIDIF judgement before enter sleep or idle mode - HY16F3981_ADC_IA : Add HAO 2MHz/4MHz/10MHz/16MHz clock setting and modify the IA initialization(Using IA_IACHM_Both) Add #define SW_IA_ADC_Chopper setting