HY17M24
HYCON IP User's Manual

Table of Contents

Attention:

1、 HYCON Technology Corp. reserves the right to change the content of this datasheet without further notice. For most up-to-date information, please constantly visit our website: http://www.hycontek.com .

2、 HYCON Technology Corp. is not responsible for problems caused by figures or application circuits narrated herein whose related industrial properties belong to third parties.

3、 Specifications of any HYCON Technology Corp. products detailed or contained herein stipulate the performance, characteristics, and functions of the specified products in the independent state. We does not guarantee of the performance, characteristics, and functions of the specified products as placed in the customer's products or equipment. Constant and sufficient verification and evaluation is highly advised.

4、 Please note the operating conditions of input voltage, output voltage and load current and ensure the IC internal power consumption does not exceed that of package tolerance. HYCON Technology Corp. assumes no responsibility for equipment failures that resulted from using products at values that exceed, even momentarily, rated values listed in products specifications of HYCON products specified herein.

5、 Notwithstanding this product has built-in ESD protection circuit, please do not exert excessive static electricity to protection circuit.

6、 Products specified or contained herein cannot be employed in applications which require extremely high levels of reliability, such as device or equipment affecting the human body, health/medical equipments, security systems, or any apparatus installed in aircrafts and other vehicles.

7、 Despite the fact that HYCON Technology Corp. endeavors to enhance product quality as well as reliability in every possible way, failure or malfunction of semiconductor products may happen. Hence, users are strongly recommended to comply with safety design including redundancy and fire-precaution equipments to prevent any accidents and fires that may follow.

8、 Use of the information described herein for other purposes and/or reproduction or copying without the permission of HYCON Technology Corp. is strictly prohibited.

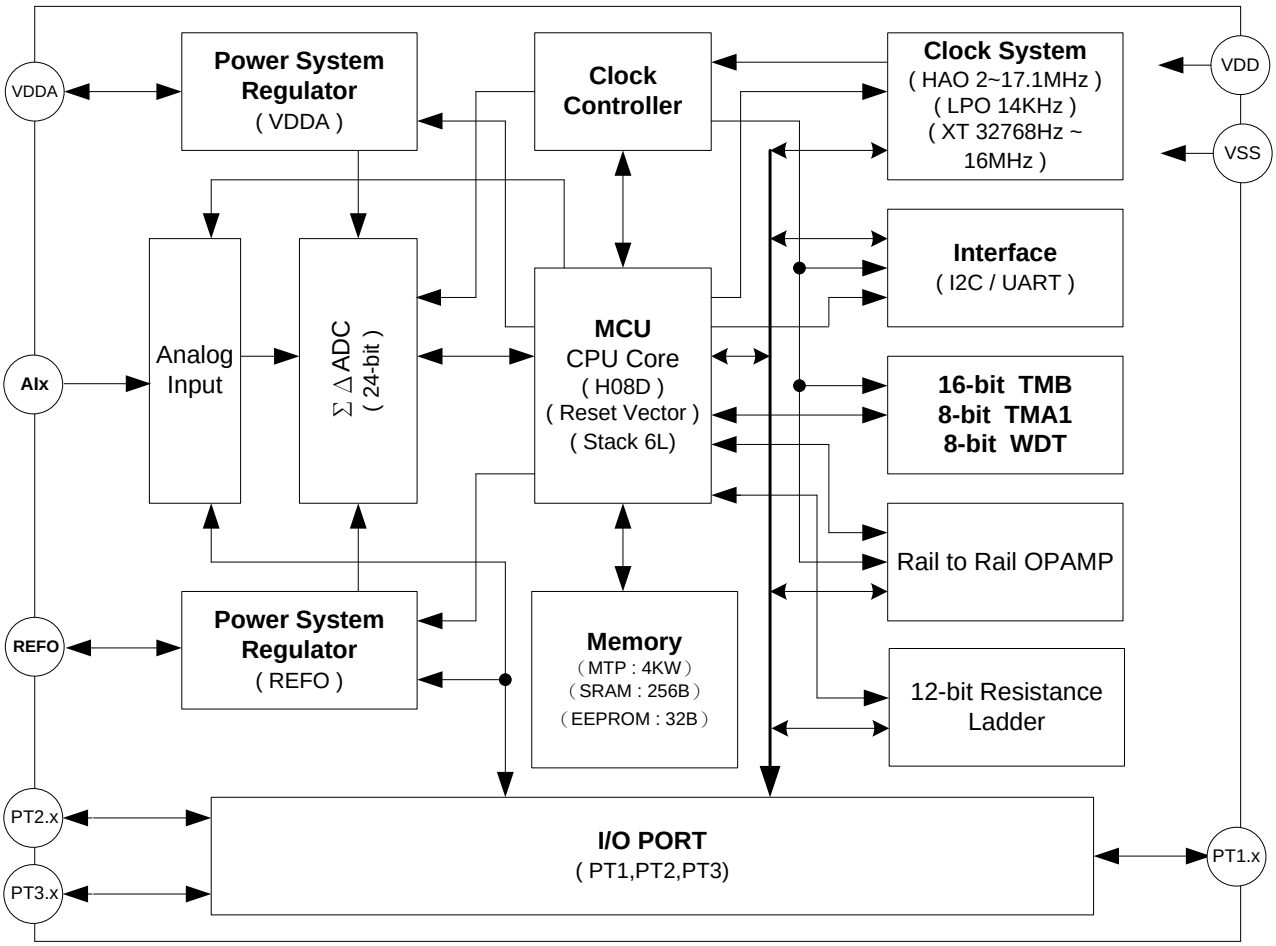
1. Document Description

HYCON IP(Intellectual Property) represents all internal IP of HYCON 8-bit MCU. This document aims at describing SOC IC internal digital, analog, communication and peripheral IP of HY17M24, of which can be grouped into four categories:

(1) Digital IP : Timer A/Timer B/WDT/PWM/GPIO
(2) Analog IP : 12 bit Resistance Ladder(DAC)/ADC/OPAMP/CMP
(3) Communication IP : Hardware UART/ Hardware I2C
(4) Peripheral IP : Power Management

2. IC Description

Basic description of each HY17M24 IP.



(01) Adopting Hycon Technology 8-bit CPU core.

(02) Voltage operation range: 1.9V~5.5V(No analog power enable condition), and temperature operation range: -40℃~85℃.

(03) Support external 16MHz crystal oscillator or internal 16MHz RC oscillator,

(04) 4K words MTP Program Memory (Write/Erase cycle times: 100 cycles)
32 bytes EEPROM Data Memory(Write/Erase cycle times: 3,000 cycles)

(05) Data memory: 256 Byte SRAM

(06) BOR and WDT function to prevent CPU from crashing

(07) 24-bit high resolution ΣΔADC

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

(09) Built-in temperature sensor, TPS

(10) Built-in 1 OPA

(11) Built-in hardware 12-bit Resistance Ladder (DAC)

(12) 8-bit Timer A

(13) 16-bit Timer B module has PWM waveform generating function

(14) Hardware serial communication I2C/UART module

3. Digital IP (TMA)

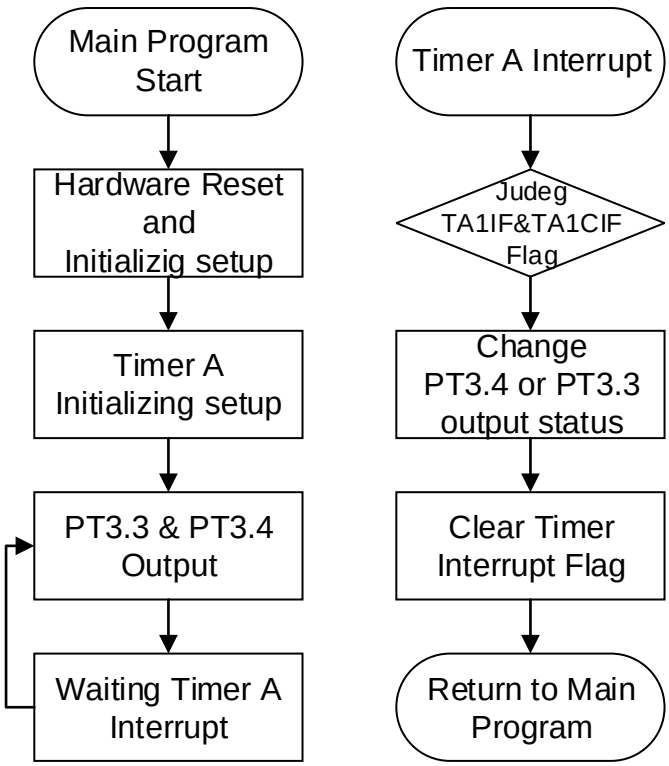
3.1. Example Name

TMA

3.2. Example Description

(1) Timer A tutorial.
(2) Enable PT1.0 Hardware Reset, and set system clock.
(3) In this example code, set Timer initializing and Timer overflow condition, enable GIE and wait Timer interrupt. Timer overflow can decide Timer interrupt frequency.
(4) Every time TA1 interrupt occurs, mean register TMA1R has been added 1, in the Timer interrupt service routine, PT3.3 will do high or low output status changed.
(5) Every time TA1C interrupt occurs, mean TMA1R count is equal to TMA1C, in the Timer interrupt service routine, PT3.4 will do high or low output status changed.

3.3. Software Flowchart



3.4. Program Description

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

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

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

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

//***** setting the timer A *********//
        TMA_Open(TMAS1_DMSCK,DTMA1_TMA1CKDIV2,4); //TMA SETTING
                                                  //TMA_Flag(us)=1/((((((HAO/DHS)/DMS))/256)/DTMA1)
                                                  //TMAC_Flag(us)=TMA_Flag*TMA1C
```

```
//***** Enable TMA INT *********//
        GIE_Enable();
        TA1CIF_ClearFlag();
        TA1CIE_Enable();
        ADCIE_Enable();
        TA1IF_ClearFlag();
        TA1IE_Enable();


        while(1);

}

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

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

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

4. Digital IP(TMB)

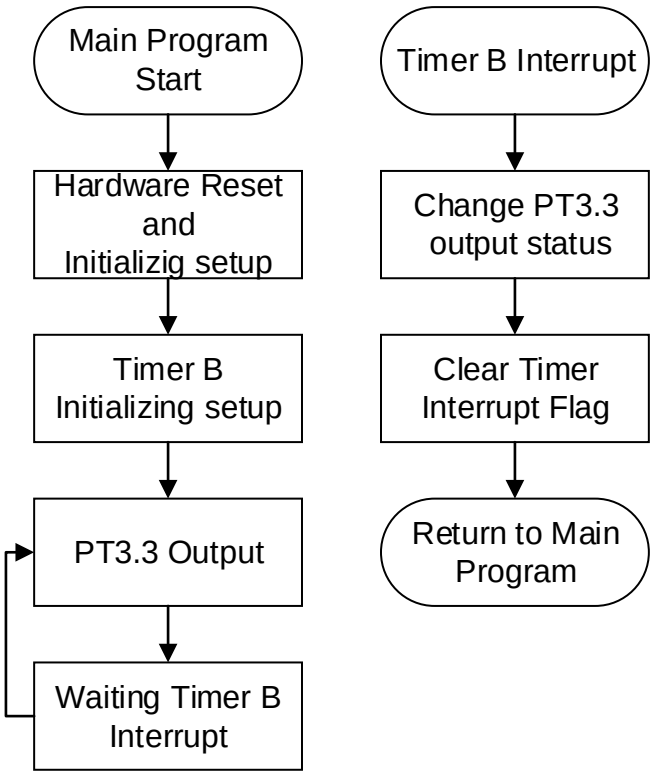
4.1. Example Name

TMB

4.2. Example Description

(1) TMB tutorial
(2) Using #define to select to compile mode_16bit, mode_17bit, mode_2_8bit, mode_8_8bit.
(3) Enable PT1.0 Hardware Reset, and set system clock.
(4) In this example code, set TMB initializing and TMB overflow condition, enable GIE and wait TMB interrupt. TMB overflow can decide TMB interrupt frequency.
(5) Every time TMB interrupt occurs, in the TMB interrupt service routine, PT3.3 will do high or low output status changed. And then return to main program.

4.3. Software Flowchart



4.4. Program Description

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

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

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

      GIE_Enable();
```

```
        while(1);
}

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

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

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


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


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


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

        TB1Enable();
}
```

```
/*---------------------------------------------------------------------------*/
/* CLOCK Init Function                                                       */
/*---------------------------------------------------------------------------*/
void CLOCKInit(void)
{
    CLK_CPUCKOpen(HAOM_1843KHZ_2M ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );
}

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

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

5. Digital IP(WDT)

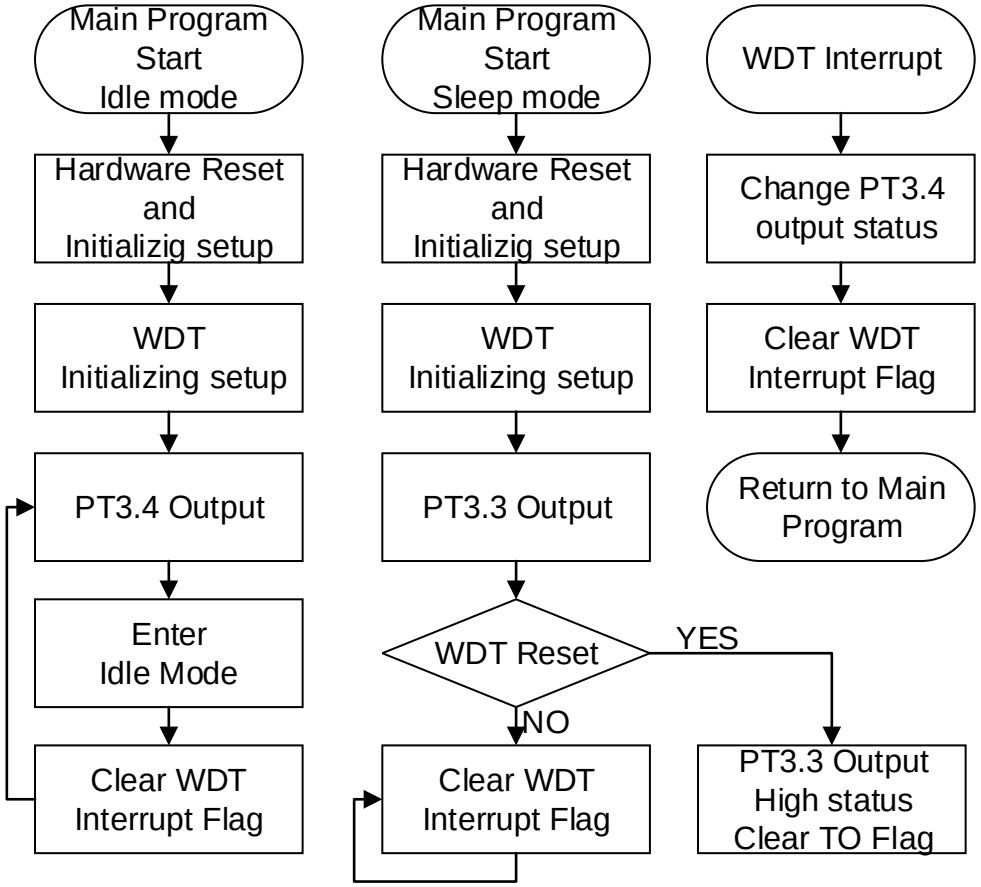
5.1. Example Name

WDT_MAIN

5.2. Example Description

(1) WDT Reset tutorial
(2) Using #define to select to compile Idle_mode, Normal_mode.
(3) Enable PT1.0 Hardware Reset, and set system clock.
(4) In this example code, set WDT initializing and WDT overflow condition, enable GIE and wait WDT interrupt. WDT overflow can decide WDT interrupt frequency.
(5) If select Idle_mode, when WDT overflow, WDT interrupt will occurs, in the WDT interrupt service routine, PT3.4 will do high or low output status changed. And then return to main program.
(6) If select Normal_mode, when WDT overflow, system will reset and set TO Flag.

5.3. Software Flowchart



5.4. Program Description

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * WDT_MAIN.c                                                                *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *
 *****************************************************************************/
/*---------------------------------------------------------------------------*/
/* Includes                                                                  */
/*---------------------------------------------------------------------------*/
#include <SFRType.h>
#include <WDT.h>
#include <CLK.h>
#include <INT.h>
#include <RST.h>
```

```
#include <GPIO.h>
/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define Normal_mode    //     WDT Reset AND TO=1
#define Idle_mode    // WDT Interrupt
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/

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

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

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

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

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

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

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

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

      WDT_Clear();    // CLEAR WDT COUNT

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

              Idle();    //    IDLE mode Enable
```

```
            __asm__("NOP");

        }

#endif

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

6. Digital IP(PWM)

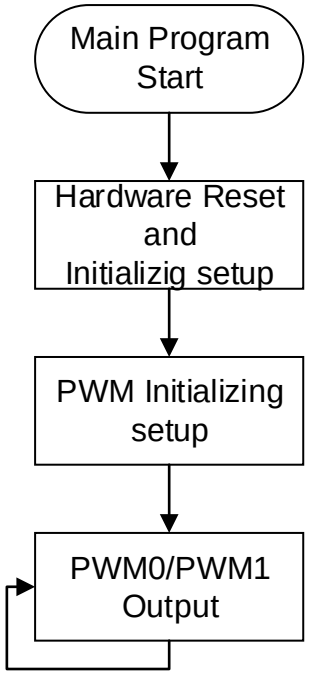
6.1. Example Name

PWM

6.2. Example Description

(1) PWM tutorial.
(2) Using #define to select to compile PWM1O,PWM2O, PWM3O, PWM4O, PWM5O, PWM6O,PWM7O.
(3) Using #define to select to compile PT16_PWM0, PT13_PWM0_1, PT32_PWM0_2, PT12_PWM1, PT20_PWM1_1, PT33_PWM1_2.
(4) Enable PT1.0 Hardware Reset, and set system clock.
(5) In this example code, set PWM initializing and PWM Duty Cycle, enable GIE
(6) PWM0(PT1.6) output enable, PWM1(PT1.2) output enable.

6.3. Software Flowchart



6.4. Program Description

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

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

////     Select PWM Output pin //
#define PT16_PWM0
//#define PT13_PWM0_1
//#define PT32_PWM0_2
#define PT12_PWM1
//#define PT20_PWM1_1
//#define PT33_PWM1_2
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void Delay_NOP(unsigned int del);
void CLOCKInit(void);
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

/*-----------------------------------PWM3O-------------------------------------*/
#ifdef PWM3O
```

```
        TMB_Open(TMBS_HSCK ,DTMB_TMBCKDIV1 ,TB1M_2_8bit ,TB1RT_LogicH );

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

        TB1_PWM0ModeSelect(PWMA0_PWM6O);            //PWM0 Output mode Select PWM6O
```

```
        TB1_PWM1ModeSelect(PWMA1_PWM6O);            //PWM1 Output mode Select PWM6O

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

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

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

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

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

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

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

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

        while(1);

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

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

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

```
#endif

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

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

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


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

7. Digital IP(BIE)

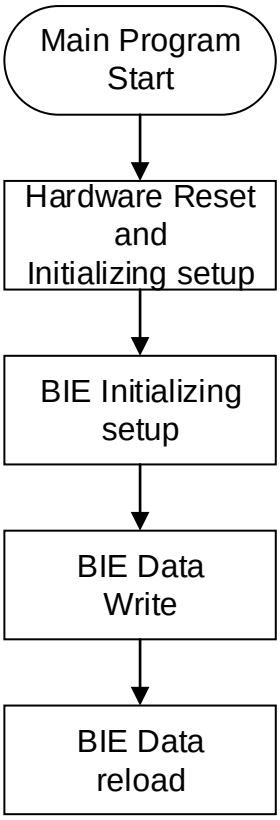
7.1. Example Name

Demo_BIE

7.2. Example Description

(1) BIE burn and read tutorial.
(2) Enable PT1.0 Hardware Reset, and set system clock.
(3) In this example code,first, enable BIE and put the data in the register EERD0~EERD31
(4) Second, burn register EERD0~EERD31 data to EEPROM.
(5) Reload data to register EERD0~EERD31.

7.3. Software Flowchart



7.4. Program Description

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

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


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

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

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

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

    BIE2_DataWriter();      //write data to EEPROM

    EEDATA_CLR();           //CLR EERD0~EERD31

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

    while(1);
```

```
}

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

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

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

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

8. Digital IP(GPIO_BZ)

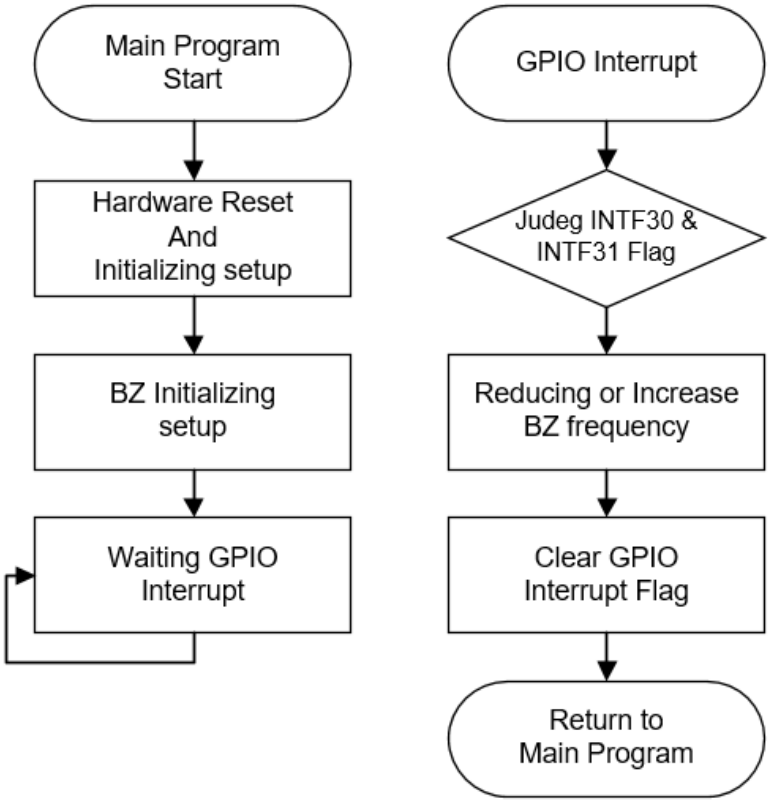
8.1. Example Name

GPIO_BZ

8.2. Example Description

(1) GPIO tutorial
(2) Using #define to select to compile BZ, BZ_1, BZ_2.
(3) Enable PT1.0 Hardware Reset, and set system clock.
(4) Setup GPIO and BZ initializing, set PT1.5 as BZ output pin, enable GIE and wait GPIO interrupt.
(5) If press button PT3.0, to do reducing of BZ frequency.
(6) If press button PT3.1, to do increase of BZ frequency.

8.3. Software Flowchart



8.4. Program Description

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

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

```
{

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

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


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

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

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

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

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

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

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

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

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

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

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

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

9. Analog IP(12 bit Resistance Ladder DAC)

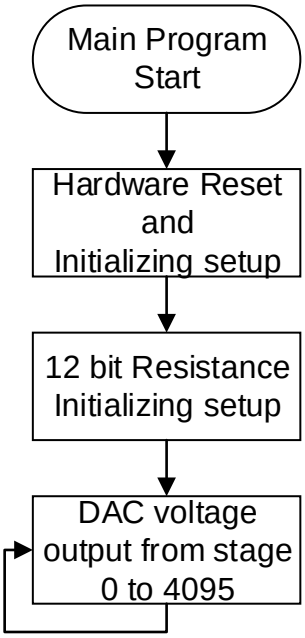
9.1. Example Name

DAC

9.2. Example Description

(1) 12 bit Resistance ladder DAC tutorial
(2) Enable PT1.0 Hardware Reset, and set system clock.
(3) In this example code, first, enable VDDA, and 12 bit Resistance Ladder positive set as VDDA, negative set as VSS. In this setting, the maximum DAC output is equal to VDDA.
(4) DAC voltage output from stage 0 to 4095, user can measure or observe DAC voltage output from IC pin DACO.

9.3. Software Flowchart



9.4. Program Description

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

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

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

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

////       VDDA setup       //
```

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

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

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

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

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

}

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

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

10. Analog IP(OPA)

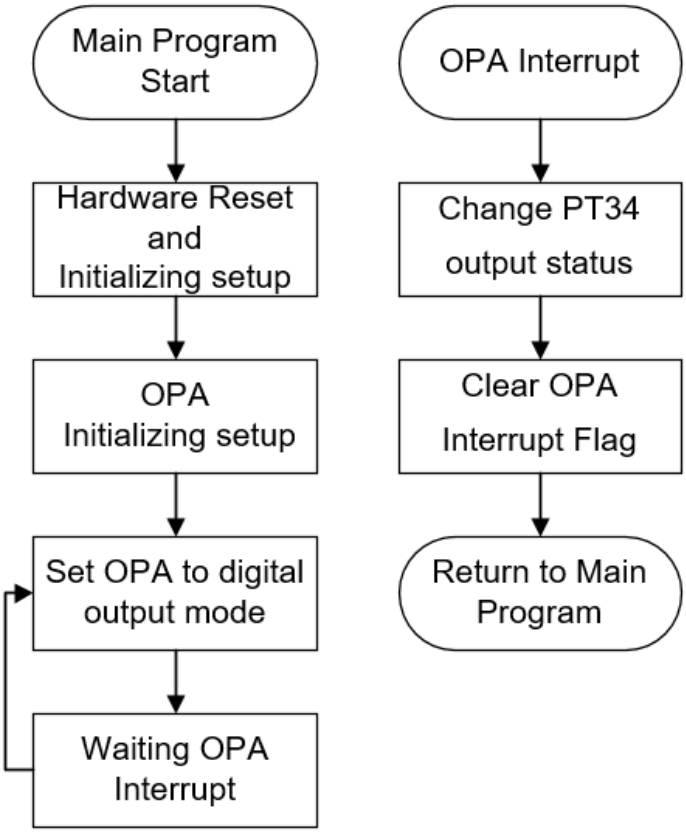
10.1. Example Name

HY17M24_OPA

10.2. Example Description

(1) OPAMP tutorial
(2) Enable PT1.0 Hardware Reset, and set system clock.
(3) Enable analog power REFO=2.0V
(4) OPAMP positive set as AI0, OPAMP negative set as REFOI.
(5) OPAMP set as comparator function. When REFOI voltage more than AI0, OPC status is 1. When REFOI voltage less than AI0, OPC status is 0.
(6) Every time, OPAMP interrupt occur, PT3.4 will do high or low output status changed according to OPC status.

10.3. Software Flowchart



10.4. Program Description

```
#define   USE_HY17M24_2M
/*****************************************************************************
 * HY17M24_OPA.c                                                             *
 * ------------------------------------------------------------------------- *
 * Copyright 2019 HYCON Technology                                           *
 * http://www.hycontek.com/                                                  *
 *                                                                           *
 * Program Description:                                                      *
 *                                                                           *
 *      HY17M24                                                                  *
 *-----------------                                                              *
 *                          |                                                *
 *                          |                                                *
 *                AI0   |-> INP                                                 *
 *                          |                                                *
 *             PT3.4 |->Output                                                *
 *                          |                                                *
 *-----------------                                                          *
 *                                                                           *
 * For External Input                                                        *
 * IC Body: HY17M24                                                          *

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

/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void SysInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char opa_flag;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
        SysInit();

        //      OPA SETTING     //
```

```
        OP1_INPSet(OP1INP_AI0_ENABLE);
        OP1_INN0Set(OP1INN0_REFOI_ENABLE);
        OPA_Out2DigOpen(OPINS_NORMAL,OPDR_NORMAL,OPOEG_LEV);       //Select digital mode

        GIE_Enable();
        while(1);
}


/*---------------------------------------------------------------------------*/
/* Interrupt Service Routines                                                */
/*---------------------------------------------------------------------------*/
void ISR(void) __interrupt
{
      unsigned char opc_status;
        if(OPCIF_IsFlag())
        {
                    OP1_OPCRead(opc_status);

                    if(opc_status)
                    GPIO_PT3OutputHigh(PT34_H);// INP>INN -> PT34 Output high
                    else
                    GPIO_PT3OutputLow(PT34_MSK);// INP<INN -> PT34 Output low

                    OPCIF_ClearFlag();              //OPCIF clear
        }

}
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/* System Init Function                                                      */
/*---------------------------------------------------------------------------*/
void SysInit(void)
{
      //PT1.0 HW Reset Function Ensable
      PT10_HWResetEnable();
      //CPU CLK SELECT
      CLK_CPUCKOpen(HAOM_1843KHZ ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );//CLK OPEN

      //Setting GPIO
      GPIO_PT3InputEnable(IN34_H);              //PT34 Digital mode
      GPIO_PT3OutputMode(TC34_H);               //PT34 Output mode
      GPIO_PT3OutputHigh(PT34_H);                     //PT34 Output high

      //REFOI ENABLE
      PWR_VDDAOpen(LDOC_2V9);                           //VDDA Enable
      REFOI_REFOint_Open(REFOS_2V0);       //REFOint Enable, REFOI=2.0V
      Delay(500);
}

/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
        for(;num>0;num--)
```

```
    __asm__("NOP");
}

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

11. Analog IP(ADC)

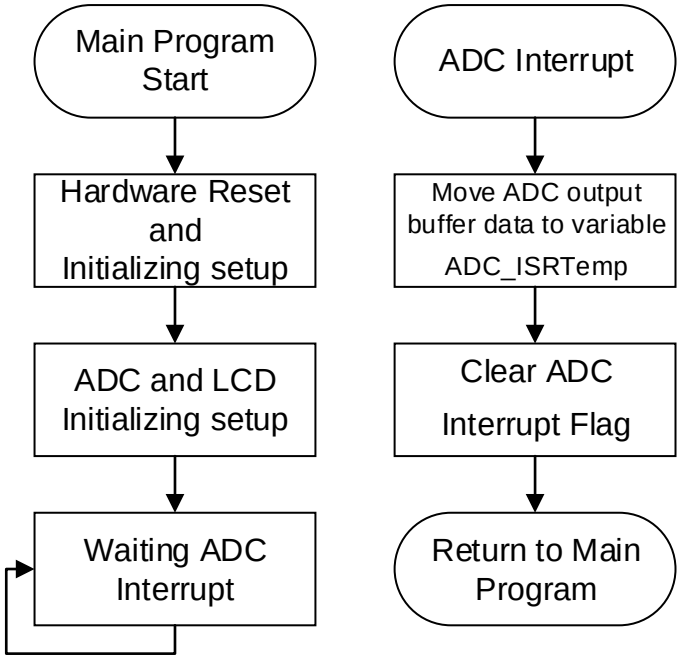
11.1. Example Name

ADC_Display

11.2. Example Description

(1) Enable PT1.0 Hardware Reset, and set system clock.
(2) ADC tutorial. Implement ADC interrupt function and capture ADC output buffer data.
(3) ADC Initializing, ADC analog power set as VDDA,ADC OSR=65536, ADC input channel set as AIO0-AIO1, ADC reference voltage set as VDDA-VSS.
(4) After ADC Initializing, Enable GIE and wait ADC interrupt. ADC OSR can decide the ADC interrupt frequency.
(5) LCD Display variable "ADC_ISRTemp"

11.3. Software Flowchart



11.4. Program Description

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

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

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
#define I2CBufferSize    11             // HY17M24 max BufferSize 127
#define I2C_WRITE    0x00           // I2C WRITE command
#define I2C_READ     0x01           // I2C READ command
#define I2C_Delay    1000000       //HAO=2MHz, wait 5s
volatile typedef union _Flag
{
  unsigned int    _byte;
    struct
    {
       unsigned b_ADCdone:1;
```

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

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

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

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

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

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

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

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

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

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

		case 0x30:

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

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

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

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

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

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

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

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

        default:
                I2C_Ctrl(0,0,0,0);    //Clear all I2C flag
                I2C_EndFlag=1;
                break;
}
        I2C_I2CINT_CLEAR();          //CLR I2C INT Flag
        I2C_I2CER_CLEAR();           //CLR I2C error Flag
        I2CIF_ClearFlag();                          //CLR I2CIF
}
if(I2CERIF_IsFlag())    //Get I2C Error Interrupt Flag
{
      I2C_EndFlag=1;
        I2C_I2CINT_CLEAR();          //CLR I2C INT Flag
        I2C_I2CER_CLEAR();           //CLR I2C error Flag
```

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


}

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

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

}

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

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

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

}

/*---------------------------------------------------------------------------*/
/*                          Send ADC Data to Display                         */
/*---------------------------------------------------------------------------*/
void SendDisplay(signed long    data)
{
        unsigned int num;
      //display range +8388607~-8388608 (dec)
        ADCData1=data;
        if((ADCData1<0)||(ADCData1>0x80000000))    // plus-minus sign judgment
        {
                ADCData1=~ADCData1;
                ADCData1++;
                DisplayBuffer[6]=S_Minus;                    // "-"
        }
        else
        {
```

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

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

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

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

        I2CIE_Enable();
        I2CERIE_Enable();
}

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

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

12. Analog IP (CMP)

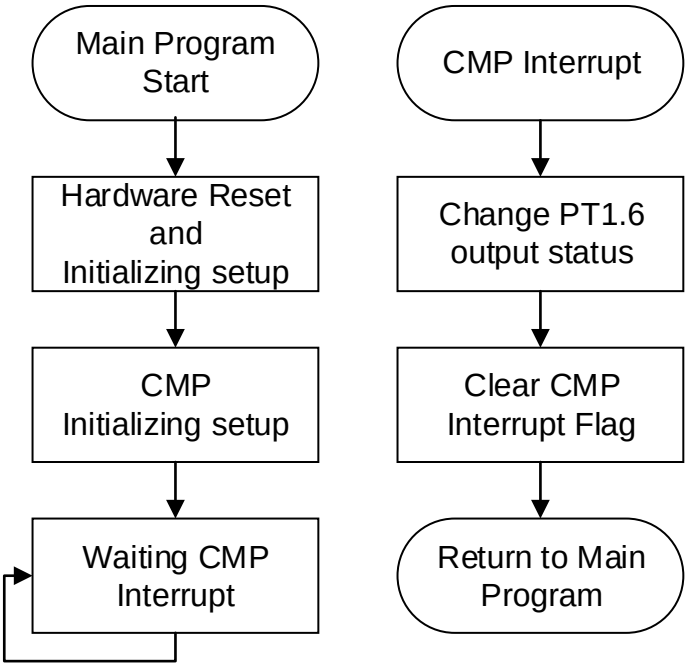
12.1. Example Name

HY17M24_CMP

12.2. Example Description

(1) CMP tutorial.
(2) Enable PT1.0 Hardware Reset, and set system clock.
(3) CMP set as Hysteresis comparator.
(4) CMP Initializing, compare RLO with CH2(PT3.7) voltage. When CH2(PT3.7) is more than RLO, CMPO(PT1.6) output high. When RLO is more than CH2(PT3.7), CMPO(PT1.6) output low.
(5) Control register CPDM and register CPDA can make CMP as Hysteresis Comparator. When CH2(PT3.7) is more than RLO, CPOB=CPDA=0,that makes RLO voltage change. When RLO is more than CH2(PT3.7), CPOB=CPDA=1, that makes RLO voltage change.

12.3. Software Flowchart



12.4. Program Description

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

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

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

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

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

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

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


    GIE_Enable();
    while(1);
}


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

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

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

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

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

}

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

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

13. Communication IP(UART)

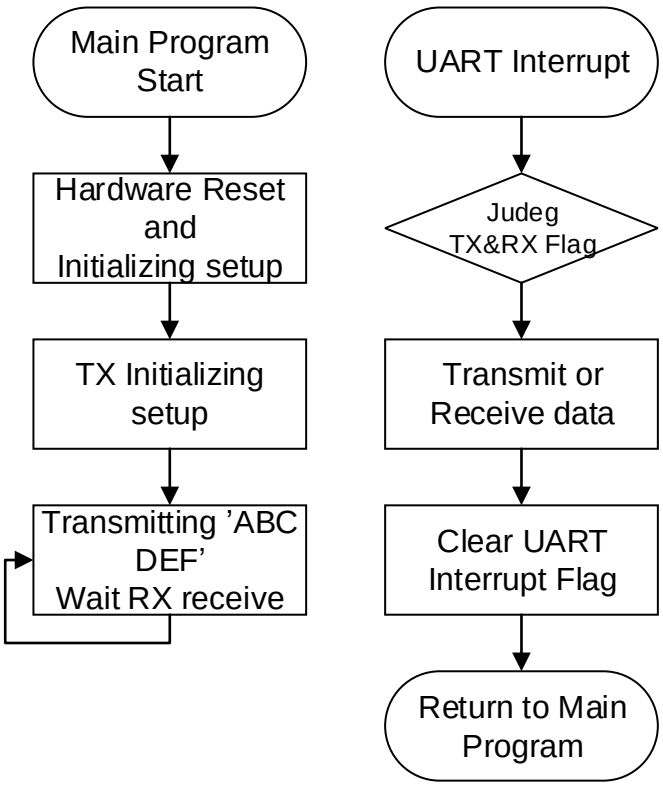
13.1. Example Name

HY17M24_UART

13.2. Example Description

(1) HY17M24 UART TX and RX tutorial.
(2) Enable PT1.0 Hardware Reset, and set system clock.
(3) UART Initializing, UART Baud Rate=9600.
(4) 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'

13.3. Software Flowchart



13.4. Program Description

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

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

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

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

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

#define   Uart_RX_BufferSize 6
```

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

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

   UartTxIndex=0;
   UartRxIndex=0;;

   while(1)
   {


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

         if(
            (UartRxBuffer[5]==UartRxBuffer_Command[5] && UartRxBuffer[4]==UartRxBuffer_Command[4]) &&
            (UartRxBuffer[3]==UartRxBuffer_Command[3] && UartRxBuffer[2]==UartRxBuffer_Command[2]) &&
            (UartRxBuffer[1]==UartRxBuffer_Command[1] && UartRxBuffer[0]==UartRxBuffer_Command[0])
            )
         {
            //if UART receive == 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f)
            //send out 0x61(a)0x62(b)0x63(c)0x64(d)0x65(e)0x66(f) and stop to send out
```

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

         UartRxIndex=0;
         Flagbits.b_UART_RxDone=0;

      }


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

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

   NOP();

   //UART RX Event
   if(UART_INT_RCIsFlag())
```

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

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

}

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

/*---------------------------------------------------------------------------*/
/* UART Init Function                                                        */
/*---------------------------------------------------------------------------*/
void UARTInit(void)
{
#ifdef UART_ABD
  UART_Open(0 ,8 ,PARITY_None ,0);
  UART_WUEDisable();                    // Wake-up disable
  UART_ABDEnable();                     //Enable Auto Baudrate
  while((BA0CN & 0x01) == 1 );          //Wait for master send 055H
  UART_INT_RCClearFlag();               //Clear RC interrupt flag
#else
  UART_Open(48 ,8 ,PARITY_EVEN ,0);     //HAO=2MHz,baud=9600
```

```
#endif
   TXIE_Enable();
   RCIE_Enable();

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

   CLK_CPUCKOpen(HAOM_1843KHZ,OSCS_HAO ,DHS_HSCKDIV1 ,CPUS_DHSCK );

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

14. Communication IP(I2C)

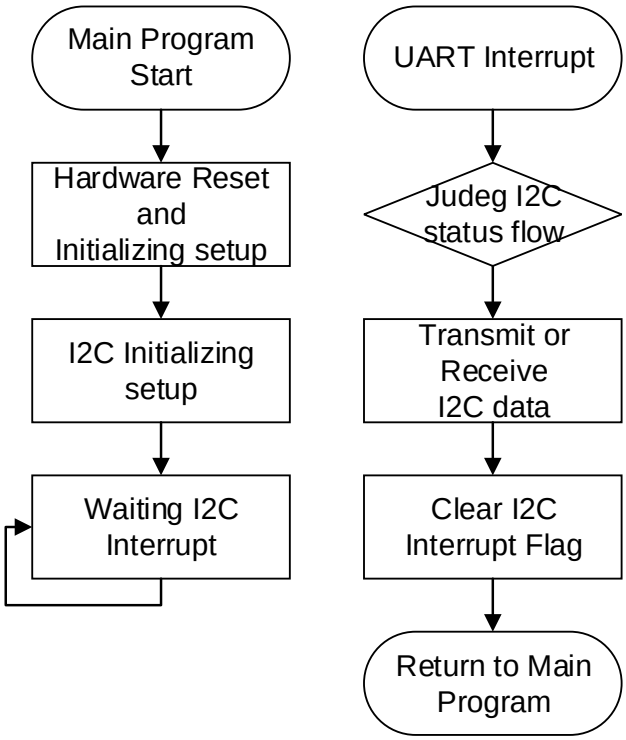
14.1. Example Name

I2C_E2PROM

14.2. Example Description

(1) HY17M24 I2C Master mode tutorial.
(2) I2C Master communicate with I2C EEPROM 24C02, I2C Master single write & read and multiple write & read example.
(3) In this example, first, I2C Master write 0x01 to EEPROM WORD ADDRESS 0x00, and then I2C Master read EEPROM WORD ADDRESS 0x00.
(4) Second, I2C Master write a sequence of 4 bytes data to EEPROM WORD ADDRESS 0x01, and then I2C Master read a sequence of 2/5/6 bytes data from EEPROM WORD ADDRESS 0x00.
(5) If finish this example correctly, EEPROM address 0x00 data is equal to 0xAA, EEPROM address 0x01 data is equal to 0x02, EEPROM address 0x02 data is equal to 0x03, EEPROM address 0x03 data is equal to 0x04, EEPROM address 0x04 data is equal to 0x05.

14.3. Software Flowchart



14.4. Program Description

Explanation : Paste main.c at here for reference only. No shows I2C Master initializing code below.

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

 * For External Input                                          *
 * IC Body: HY17M24                                               *

 *****************************************************************************/
/*--------------------------------------------------------------------------*/
/* Includes                                                      */
/*--------------------------------------------------------------------------*/
#include <SFRType.h>
#include <INT.h>
#include <CLK.h>
#include <RST.h>
#include <PWR.h>
#include <I2C.h>
#include <GPIO.h>
#include "EEPROM_24Cxx.h"
/*--------------------------------------------------------------------------*/
/* DEFINITIONS                                                      */
/*--------------------------------------------------------------------------*/
#define   I2CBufferSize   10          // HY17M24 max BufferSize 127
#define I2C_WRITE   0x00          // I2C WRITE command
#define I2C_READ    0x01          // I2C READ command
```

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

    GIE_Enable();

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

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

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

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

    EEPROM_ReadArray(I2C_Read_Buffer, 0x00, 6); //sequential I2C_READ data from 0x00 to 0x05
    //read data:0xaa,0x01,0x02,0x03,0x04,0xff
    Delay(1000);                                    //Delay for EEPROM 24C02 I2C_WRITE Cycle Time
    while(1);
}
```

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

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

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

        case 0x30:

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

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

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

```
                break;

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

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

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

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

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

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

}
```

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

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

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

}

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

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

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

15. Peripheral IP(Power)

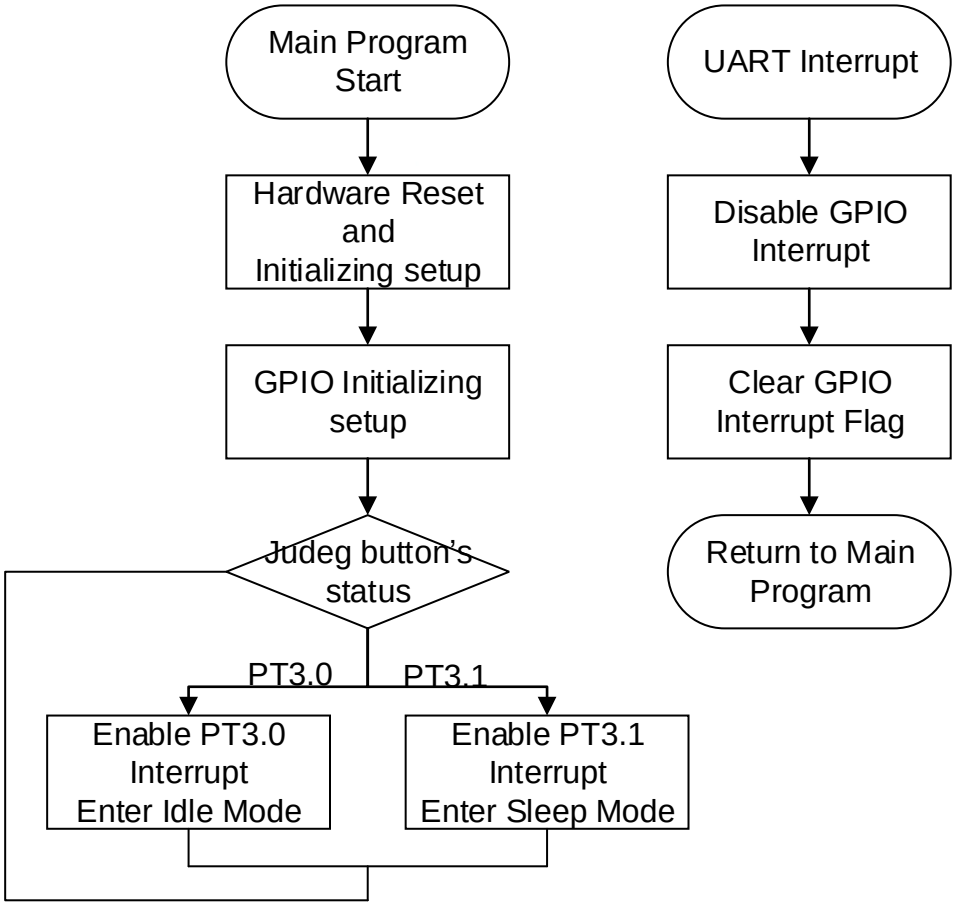
15.1. Example Name

Demo_PWR

15.2. Example Description

(1) HY17M24 Sleep and Idle mode tutorial.
(2) If press button PT3.1, enter Sleep mode.
(3) If press button PT3.0, enter Idle mode.
(4) If this example code work on HY17M24 Development board, user can measure Sleep mode current about 0.25uA, Idle mode current about 1uA.

15.3. Software Flowchart



15.4. Program Description

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

/*---------------------------------------------------------------------------*/
/* DEFINITIONS                                                               */
/*---------------------------------------------------------------------------*/
//#define Sleep_mode
#define Idle_mode
/*---------------------------------------------------------------------------*/
/* Function PROTOTYPES                                                       */
/*---------------------------------------------------------------------------*/
void GPIOInit(void);
void Delay(unsigned int num);
/*---------------------------------------------------------------------------*/
/* Global CONSTANTS                                                          */
/*---------------------------------------------------------------------------*/
unsigned char cks=0;
/*---------------------------------------------------------------------------*/
/* Main Function                                                             */
/*---------------------------------------------------------------------------*/
void main(void)
{
    //PT1.0 HW Reset Function Enable
    PT10_HWResetEnable();
////   CPU CLK Init              //
    CLK_OSCSelect(OSCS_LPO);
```

```
    CLK_HAODisable();
    CLK_CPUCKSelect(CPUS_DHSCK);

     GPIOInit();

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

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

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

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


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

     if(INTF31_IsFlag())
     {
          Delay(20);          //debounce
          while(GPIO_PT3GET(PT31_H) == 0);      //wait PT31=0
          INTE31_Disable();
     }
     INTF30_ClearFlag();
     INTF31_ClearFlag();
}
```

```
/*---------------------------------------------------------------------------*/
/* Subroutine Function                                                       */
/*---------------------------------------------------------------------------*/
/*---------------------------------------------------------------------------*/
/*            GPIO Init function                                             */
/*---------------------------------------------------------------------------*/
void GPIOInit(void)
{
    GPIO_PT3InputEnable(IN30_H | IN31_H);          //PT3.0/3.1 Digital mode
    GPIO_PT3SETPU(PU30_H | PU31_H);
    GPIO_PT3InputMode(TC30_H | TC31_H);                //PT3.0/3.1 Input mode
    INTG30_Edgefall();          //PT30 INT triggering conditions 1->0
    INTE30_Disable();           //PT30 INT Disable
   INTG31_Edgefall();       //PT31 INT triggering conditions 1->0
    INTE31_Disable();           //PT31 INT Disable
}
/*---------------------------------------------------------------------------*/
/* Software Delay Subroutines                                                */
/*---------------------------------------------------------------------------*/
void Delay(unsigned int num)
{
    for(;num>0;num--)
   __asm__("NOP");
}

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

16. Revisions

The following describes the major changes made to the document, excluding the punctuation and font changes.

Version	Page	Date	Revision Summary
V01	All	2020/05/04	First edition
V02	All	2021/07/30	-The 12 bit Resistance ladder DAC : Modify DAC_Open() -The all demo code: Remove CCOPT_Enable()
V03	7 All	2021/09/15	- Modify the minimum operating voltage of the digital circuit -The all demo code: Change the parameter name of HAOM HAOM_1843KHZ, HAOM_4147KHZ, HAOM_8755KHZ, HAOM_17510KHZ
V04	7	2023/02/22	1. Modify the Write/Erase cycle times of MTP/EEPROM. Before the modification, the number of MTP the Write/Erase cycle times is 1K times After the modification, the number of MTP the Write/Erase cycle times is 100 times Before the modification, the number of EEPROM the Write/Erase cycle times is 30K times After the modification, the number of EEPROM the Write/Erase cycle times is 3K times