//-----------------------------------------------------------------------------

// TP6\_BSE\_Main.c

// AUTH: FJ / Rodolphe Lajugi / Maxime CORNATON

// DATE: 5/12/21

//

// Target: C8051F02x

// Tool chain: KEIL Microvision5

//

//-----------------------------------------------------------------------------

//-----------------------------------------------------------------------------

// Cette application exécute les tâches suivantes:

// SYSCLK = Quartz externe = 22.1184 MHz (Visu sur P1.0)

// Clignotement de la LED P1.6

// Commande du clignotement de la LED avec le bouton poussoir P3.7 (INT7) et

// via le signal DECL\_EXT sur P3.6 (INT6)

// Mise en oeuvre de la base de temps Timer2 de 10ms et interruption Timer2

// Mise en oeuvre de l'UART0 à 115200 Bd, pas de parité, 8 bits, 1 stop bit

// Visu Flag INT7 sur P2.4

// Visu Flag INT6 sur P6.4

// Visu Flag INT Timer2 sur P3.5

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//-----------------------------------------------------------------------------

// Fichiers d'entête

#include "intrins.h"

#include<stdio.h>

#include<stdlib.h>

#include<c8051F020.h>

#include<c8051F020\_SFR16.h>

#include<TP5\_BSE\_Lib\_Config\_Globale.h>

#include<TP5\_BSE\_Lib\_Divers.h>

#include<TP5\_BSE\_Lib\_Timers\_UART.h>

#include<TP5\_BSE\_Lib\_INT\_EXT.h>

#include<TP5\_BSE\_Main.h>

//-----------------------------------------------------------------------------

// Déclaration des MACROS

#define LED\_ON 1

#define LED\_OFF 0

#define LED\_BLINK 0

//#define BP\_ON 0

//#define BP\_OFF 1

#define TO\_BE\_PROCESSED 1

#define PROCESSED 0

#define SET\_VISU\_INT6 P6 |= (1<<4)

#define RESET\_VISU\_INT6 P6 &= ~(1<<4)

//-----------------------------------------------------------------------------

// Déclarations Registres et Bits de l'espace SFR

sbit LED = P1^6; // LED

sbit BP =P3^7;

sbit VISU\_INT7 = P2^4;

sbit VISU\_INT\_TIMER2 = P3^5;

//sbit VISU\_INT\_TIMER4 = P3^4;

//sbit SIG\_OUT = P3^3;

//-----------------------------------------------------------------------------

// Variables globales

bit Event = PROCESSED; // Détection des évènements pour changer le clignotement de la LED

//unsigned int Event\_to\_Count = 100 ; //Valeur du comptage d'évènements

//long frequence = 0; // Fréquence calculée de SIG\_IN

//unsigned int CP\_Overflow\_Timer4; // Compteur d'overflows du Timer4

bit Flag\_Seconde = 0; // drapeau 1s écoulée

int CP\_Mesure = 0;

int Value\_mV;

int Compteur\_ramp = 0;

//-----------------------------------------------------------------------------

// MAIN Routine

//-----------------------------------------------------------------------------

void main (void) {

// unsigned char temp\_char;

// char xdata buf[20];

// char i;

// int value;

// Configurations globales

Init\_Device();

// Configurations spécifiques

Config\_INT7(); // Configuration de INT7

Config\_INT6(); // Configuration de INT6

Config\_Timer2\_TimeBase();

CFG\_Clock\_UART0();

CFG\_UART0();

Voltage\_Reference\_Init();

ADC0\_Init\_Etape2();

// ADC0\_Init\_Etape4();

DAC0\_Init();

// Fin des configurations

printf("\*\*\*\* TP5 - System OK \*\*\*\*\n\n");

EA = 1; // Validation globale des interruptions

// Boucle infinie

while(1)

{

// Code à ajouter...

if(Flag\_Seconde == 1){

Flag\_Seconde = 0;

CP\_Mesure += 1;

AD0INT = 0;

AD0BUSY = 1;

while(AD0INT == 0);

Value\_mV = ((ADC0\*2.4)/4096)\*1000;

printf("Mesure %5u -ADC0H:%#2.2bx - ADC0L:%#2.2.bx - Vin en mV: %#2.2u \n\r",CP\_Mesure,ADC0H,ADC0L,Value\_mV);

}

while(Compteur\_ramp != 4096){

DAC0L = Compteur\_ramp%256;

DAC0H = Compteur\_ramp>>8;

Compteur\_ramp ++;

Delay\_CLK22M\_10Micro();

}

DAC0L = 0x00;

DAC0H = 0x00;

Compteur\_ramp = 0;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Voltage\_Reference\_Init

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void Voltage\_Reference\_Init(void)

{

// Code de configuration de Vref à insérer

REF0CN |= (1<<0);

REF0CN |= (1<<1);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// ADC0\_Init\_Etape2

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ADC0\_Init\_Etape2(void)

{

// Code de configuration de l'ADC0 pour l'étape2 à insérer

AMX0CF |= (1<<1);

AMX0CF |= (1<<0);

AMX0CF &= ~(1<<3);

AMX0CF |= (1<<2);

AMX0SL |= (1<<0);

AMX0SL |= (1<<1);

AMX0SL |= (1<<2);

AMX0SL &= ~(1<<3);

ADC0CF &= ~(1<<2);

ADC0CF &= ~(1<<1);

ADC0CF &= ~(1<<0);

ADC0CF &= ~(1<<7);

ADC0CF &= ~(1<<5);

ADC0CF &= ~(1<<4);

ADC0CF &= ~(1<<3);

//ADC0CF |= (1<<6);

//ADC0CN

AD0LJST = 0;

AD0CM0 = 0;

AD0CM1 = 0;

//AD0TM = 1;

AD0EN = 1; //ACTIVATION

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// ADC0\_Init\_Etape4

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ADC0\_Init\_Etape4(void)

{

// Code de configuration de l'ADC0 pour l'étape4 à insérer

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// DAC0\_Init

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void DAC0\_Init(void)

{

// Code de configuration de l'ADC0 pour l'étape3 à insérer

DAC0CN = 0x80; //Activation

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Timer3\_Init

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void Timer3\_Init\_Fech(void)

{

// Code de configuration du Timer3

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Detection d'une action sur le Bouton poussoir P3.7

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ISR\_INT7 (void) interrupt 19 // Interruption Bouton poussoir

{

VISU\_INT7 = 1;

P3IF &= ~(1<<7); // IE3 mis à 0 - remise à zéro du pending flag de INT7 effacé

Event = TO\_BE\_PROCESSED;

VISU\_INT7 = 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Détection de fronts montants et descendants sur P3.6

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ISR\_INT6 (void) interrupt 18 // Interruption externe sur P3.6

{

SET\_VISU\_INT6;

P3IF &= ~(1<<6); // IE6 mis à 0 - remise à zéro du pending flag de INT6 effacé

P3IF ^= (1<<2); // Action sur IE6CF - Commutation Front montant / Front Descendant

Event = TO\_BE\_PROCESSED;

RESET\_VISU\_INT6;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Base de temps de 10ms

// Gestion du clignotement de la LED

// Prise en compte du changement de mode de clignotement

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ISR\_Timer2 (void) interrupt 5

{

static char CP\_Cligno;

static char CP\_Seconde = 0;

static bit STATE\_LED = LED\_BLINK;

// char Read\_Cfg\_Event;

// static unsigned int CP\_Timer4 = 0;

// static unsigned int OLD\_CP\_Timer4 = 0;

VISU\_INT\_TIMER2 = 1;

if (TF2 == 1)

{

TF2 = 0;

CP\_Seconde++;

if (CP\_Seconde >= 100)

{

CP\_Seconde = 0;

Flag\_Seconde = 1;

}

// Gestion des évènements INT6 et INT7

// pour gérer les modes de clignotement de la LED

if (Event == TO\_BE\_PROCESSED)

{

Event = PROCESSED;

STATE\_LED = !STATE\_LED;

}

// Gestion du mode de clignotement de la LED

CP\_Cligno++;

if (CP\_Cligno > 11) CP\_Cligno = 0;

if (STATE\_LED == LED\_BLINK)

{

if (CP\_Cligno < 2) LED = LED\_ON;

else LED = LED\_OFF;

}

else LED = LED\_OFF;

}

// Sécurité: si EXF2 est à 1 - RAZ de EXF2

if (EXF2 == 1)

{

EXF2 = 0;

}

VISU\_INT\_TIMER2 = 0;

}