```c
/*************************************/
/*  File: gestion_Pwm.c              */
/*  Name: gestion_Pwm                 */
/*                                   */
/*  Author: Jérémy C                 */
/*  Programmation: language C        */
/*                                   */
/*  Project: RGB lamp                */
/*  Date: 14/07/2013                 */
/*  Microconrtoller: PIC16F1829      */
/*  Version: v1                      */
/*  Last modification:               */
/*************************************/

#include <xc.h>
#define _XTAL_FREQ 32000000          // Used by the XC8 delay_ms(x) macro

//config bits that are part-specific for the PIC16F1829
#pragma config FOSC=INTOSC, WDTE=OFF, PWRTE=OFF, MCLRE=ON, CP=OFF, CPD=OFF,
BOREN=ON, CLKOUTEN=OFF, IESO=OFF, FCMEN=OFF
#pragma config WRT=OFF, PLLEN=ON, STVREN=OFF, LVP=ON

#define DOWN        0
#define UP          1
#define SWITCH      PORTAbits.RA1
#define LED1        PORTAbits.RA2
#define LED2        PORTCbits.RC0

static volatile short int PWM1, PWM2, PWM3;
static volatile unsigned int tempo_led;
static volatile bit flag_led, flag_un;

void init_pic();
void init_PWM();
void gestion_PWM(short int PWM1, short int PWM2, short int PWM3);

void main(void)
{
    init_pic();
    init_PWM();
    gestion_PWM(1020,10,10);

    while (1)
    {
        if(flag_led)            // All the 10ms
        {
            flag_led = 0;

            if(flag_un)
            {
                LED1 = 1;       // set of LED1, reset of LED2
                LED2 = 0;
                flag_un = 0;
            }
            else
            {
                LED1 = 0;       // set of LED2, reset of LED1
                LED2 = 1;
                flag_un = 1;
            }
        }
    }
}

void gestion_PWM(short int PWM1, short int PWM2, short int PWM3)
{
    CCP1CONbits.DC1B = PWM1;  // LSBits PWM1
    CCPR1L = (PWM1>>2) ;      // MsBits PWM1
    CCP2CONbits.DC2B = PWM2;  // LSBits PWM2
    CCPR2L = (PWM2>>2) ;      // MsBits PWM2
    CCP4CONbits.DC4B = PWM3;  // LSBits PWM3
    CCPR4L = (PWM3>>2) ;      // MsBits PWM3
}

void init_pic()
{
    OSCCON = 0b11110000;     //32MHz clock speed
    OSCSTAT = 0b11110001;

    ANSELA = 0b00000001;     // Potar in analog (RA0)
    ANSELB = 0;
    ANSELC = 0;

    TRISA = 0b00100011;      // Potar, switch, IR Receiver in Input
    TRISB = 0;
    TRISC = 0;

    LATA = 0;
    LATB = 0;
    LATC = 0;

    INTCON = 0b11111000;     // Interrupt control register,
GIE=PEIE=TMR0IE=INTE=IOCIE=1
    OPTION_REG = 0b00000111; //1:256 prescaler
    TMR0 = 99;               // 255-99= 156, 156 * 32µs = 5ms

    PIE1 = 0b11000011;
    PIE2 = 0;
    PIE3 = 0b00001010;
    PIE4 = 0;

    CCP2SEL = 0;             // Selection of CCP2 module on RC3

    flag_led = 0;            // Initialisation of variables
    flag_un = 0;
    LED2 = 0;
}

void init_PWM()
{
 /*
 * PWM registers configuration
 * Fosc = 32000000 Hz
 * Fpwm = 1.95kHz
 * Duty Cycle = 0 %
 * Resolution is 10 bits
 * Prescaler is 16
 * Enable Timer2/4/6 by setting the TMR2ON bit of the TxCON register.
 */

CCPTMRS = 0b10110100 ;   // Selection of Timer2/4/6 for each PWM
```

```
T2CON = 0b00000110 ;     // 1:16 Prescaler
T4CON = 0b00000110 ;
T6CON = 0b00000110 ;

PR2 = 0b11111111 ;      // Set the PWM period by loading the PR2 register.
PR4 = 0b11111111 ;      // Set the PWM period by loading the PR4 register.
PR6 = 0b11111111 ;      // Set the PWM period by loading the PR6 register.

CCPR1L = 0x00 ;    // Configure the duty cycle CCP1
CCPR2L = 0x00 ;    // Configure the duty cycle CCP2
CCPR4L = 0x00 ;    // Configure the duty cycle CCP4

CCP1CON |= 0b00001100 ;  // Configure the CCP module for the PWM mode
CCP2CON |= 0b00001100 ;  // by loading the CCPxCON register
CCP4CON |= 0b00001100 ;  // with the appropriate values.

TMR2IF = 0; // Clear the TMR2IF interrupt flag bit of the PR2 register.
TMR4IF = 0; // Clear the TMR2IF interrupt flag bit of the PR4 register.
TMR6IF = 0; // Clear the TMR2IF interrupt flag bit of the PR6 register.
}

void interrupt prog_it()
{
    if(TMR0IF)  // Multiplexage, comptage de 5ms
    {
    T0IF = 0;
    TMR0 = 99;
    tempo_led++;

        if (tempo_led >= 2) // 10ms
        {
        flag_led = 1;
        tempo_led = 0;
        }
    }

    if(TMR2IF)  // PWM1
    {
    TMR2IF = 0;
    }

    if(TMR4IF)  // PWM2
    {
    TMR4IF = 0;
    }

    if(TMR6IF)  // PWM3
    {
    TMR6IF = 0;
}
}
```