





#CODE: (CLASS PROGRAM )

// CONFIG

#pragma config FOSC = EXTRC // Oscillator Selection bits (RC oscillator)

#pragma config WDTE = OFF // Watchdog Timer Enable bit (WDT disabled)

#pragma config PWRTE = OFF // Power-up Timer Enable bit (PWRT disabled)

#pragma config BOREN = OFF // Brown-out Reset Enable bit (BOR disabled)

#pragma config LVP = OFF // Low-Voltage (Single-Supply) In-Circuit Serial Programming Enable bit (RB3 is digital I/O, HV on MCLR must be used for programming)

#pragma config CPD = OFF // Data EEPROM Memory Code Protection bit (Data EEPROM code protection off)

#pragma config WRT = OFF // Flash Program Memory Write Enable bits (Write protection off; all program memory may be written to by EECON control)

#pragma config CP = OFF // Flash Program Memory Code Protection bit (Code protection off)

#include<xc.h>

#define \_XTAL\_FREQ 6000000

// Declare PWM duty cycle high and low byte variables

unsigned char pwm10\_high, pwm10\_low;

unsigned char pwm50\_high, pwm50\_low;

unsigned char pwm80\_high, pwm80\_low;

// Function prototypes

void pwm\_init(void);

void pwm\_dutycycle\_update(void);

void main()

{

pwm\_init(); // Initialize PWM settings

while (1)

{

pwm\_dutycycle\_update(); // Update PWM duty cycle continuously

}

}

// Define PWM duty cycle high and low byte values

void pwm\_init(void)

{

TRISC = 0xFB; // Configure RC2 as output (1111 1011)

CCP1CON = 0x0C; // Set PWM mode for CCP1

T2CON = 0x06; // Enable Timer2 with prescaler 1:16

PR2 = 0x5E; // Set PWM period

// Initialize PWM duty cycle values

// Additional configurations can be added based on requirements

pwm10\_high = 0x09;

pwm10\_low = 0x02;

pwm50\_high = 0x2F;

pwm50\_low = 0x00;

pwm80\_high = 0x4B;

pwm80\_low = 0x00;

}

// Function to update PWM duty cycle

void pwm\_dutycycle\_update(void)

{

// Set 10% duty cycle

CCPR1L = pwm10\_high; // High byte (0x09)

CCP1CON = (CCP1CON & 0xCF) | (pwm10\_low << 4); // Low byte (0x02)

\_\_delay\_ms(3000); // Delay for 3 seconds

// Set 50% duty cycle

CCPR1L = pwm50\_high; // High byte (0x2F)

CCP1CON = (CCP1CON & 0xCF) | (pwm50\_low << 4); // Low byte (0x00)

\_\_delay\_ms(3000); // Delay for 3 seconds

// Set 80% duty cycle

CCPR1L = pwm80\_high; // High byte (0x4B)

CCP1CON = (CCP1CON & 0xCF) | (pwm80\_low << 4); // Low byte (0x00)

\_\_delay\_ms(3000); // Delay for 3 seconds

}

CLASS5: (TASK PROGRAM)

// CONFIG

#pragma config FOSC = EXTRC // Oscillator Selection bits (RC oscillator)

#pragma config WDTE = OFF // Watchdog Timer Enable bit (WDT disabled)

#pragma config PWRTE = OFF // Power-up Timer Enable bit (PWRT disabled)

#pragma config BOREN = OFF // Brown-out Reset Enable bit (BOR disabled)

#pragma config LVP = OFF // Low-Voltage (Single-Supply) In-Circuit Serial Programming Enable bit (RB3 is digital I/O, HV on MCLR must be used for programming)

#pragma config CPD = OFF // Data EEPROM Memory Code Protection bit (Data EEPROM code protection off)

#pragma config WRT = OFF // Flash Program Memory Write Enable bits (Write protection off; all program memory may be written to by EECON control)

#pragma config CP = OFF // Flash Program Memory Code Protection bit (Code protection off)

#include<xc.h>

#define \_XTAL\_FREQ 6000000

// Declare PWM duty cycle high and low byte variables

unsigned char pwm20\_high, pwm20\_low;

unsigned char pwm60\_high, pwm60\_low;

unsigned char pwm90\_high, pwm90\_low;

// Function prototypes

void pwm\_init(void);

void pwm\_dutycycle\_update(void);

void main()

{

pwm\_init(); // Initialize PWM

while (1)

{

pwm\_dutycycle\_update(); // Update PWM duty cycle continuously

}

}

// Define PWM duty cycle high and low byte values

void pwm\_init(void)

{

TRISC = 0xFB; // Configure RC2 as output (1111 1011)

CCP1CON = 0x0C; // Set PWM mode for CCP1

T2CON = 0x06; // Enable Timer2 with prescaler 1:16

PR2 = 0x2E; // Set PWM period

// Initialize PWM duty cycle values

pwm20\_high = 0x09; //0000 1001

pwm20\_low = 0x02; //0000 0010

pwm60\_high = 0x1C; //0001 1100

pwm60\_low = 0x01; //0000 0001

pwm90\_high = 0x2A; //0010 1010

pwm90\_low = 0x01; //0000 0001

}

// Function to update PWM duty cycle

void pwm\_dutycycle\_update(void)

{

// Set 20% duty cycle

CCPR1L = pwm20\_high; // High byte (0x09)

CCP1CON = (CCP1CON & 0xCF) | (pwm20\_low << 4); // Low byte (0x02)

\_\_delay\_ms(3000); // Delay for 3 seconds

// Set 60% duty cycle

CCPR1L = pwm60\_high; // High byte (0x1C)

CCP1CON = (CCP1CON & 0xCF) | (pwm60\_low << 4); // Low byte (0x01)

\_\_delay\_ms(3000); // Delay for 3 seconds

// Set 90% duty cycle

CCPR1L = pwm90\_high; // High byte (0x2A)

CCP1CON = (CCP1CON & 0xCF) | (pwm90\_low << 4); // Low byte (0x01)

\_\_delay\_ms(3000); // Delay for 3 seconds

}