

```

#include<htc.h>

#include<stdio.h>

#include<htc.h>

#include<math.h>

#define timer_prescale 16

#define Fosc 8e6

#define pwm_frq 1.22e3

#define PID0 0x00

#define PID1 0x01

#define PID2 0x02

#define brk 0x00

#define idle 0xFF

#define sync 0x55

#define PID 0xAA

#define period 0x65

#define ondutycycle 100

#define offdutycycle 0

typedef unsigned char ubit08 ;

typedef unsigned short ubit16 ;

typedef float ubit32;

int PID_1(void);

ubit08 PID_Recd;


mc_conf()

{

//PORT initialisation

CM1CON0 = 0x00;    //All comparators are off

CM2CON0 = 0x00;

ADCON0 = 0x00;    //A/D converter off

ADCON1 = 0x00;

```

```

APFCON0 =0X00;    //alternate pin functions off

APFCON1 =0X00;

ANSELA  = 0x00;    //Analog functionalities off

ANSELC  = 0x00;    //Analog functionalities off

OPTION_REG = 0x00;

TRISA = 0x0B;    //hex 0X1B RA0- I/P, RA1-I/p, RA2-O/P, RA3-I/P, RA4-I/P, RA5-O/P(EN).

TRISC = 0x05;    //hex 0X25 RC0- I/P, RC1-O/P, RC2-O/P, RC3-O/P, RC4-O/P(TX), RC5-
O/P(PWM).

WPUA = 1;

WPUC = 1;

PORTA = 0;        //Setting PORTA pins to 0 value

PORTC = 0;        //Setting PORTC pins to 0 value

// RA4=1;

// uc oscillator frequency settings

OSCCON = 0x72;    //hex 0x71 configuring to 8MHz internal oscillator

OSCTUNE = 0x00;


//UART settings

TRISC4 = 0;    //configuring RC4 & RC5 as uart inputs

// TRISC5 = 1;

RXDTSEL=1;    // Rx is on RA1

TXSTA = 0x2E;    //hex 0x24,8 bit transmission, Transmit Enabled, Asynchronous mode, BRGH=1

RCSTA = 0x90;    //hex 0x90, enable serial port, 8 bit reception, enable reciver

BAUDCON = 0x08; //hex 0x08, BRG16 = 1

SPBRGH = 0;

SPBRG  = 103;    //(ubit08)x;

TXIE = 0;    //TX Interrupt disable

RCIE = 0;    //RX Interrupt disable

GIE  = 0;

PEIE = 0;

```

```
// pwm settings
```

```
TMR2ON=1;
```

```
PR2=0x65;
```

```
C1TSEL1=0;
```

```
C1TSEL0=0;
```

```
C2TSEL1=0;
```

```
C2TSEL0=0;
```

```
TMR2IF=0;
```

```
T2CKPS1=1;
```

```
T2CKPS0=0;
```

```
C3TSEL1=0;
```

```
C3TSEL0=0;
```

```
C4TSEL1=0;
```

```
C4TSEL0=0;
```

```
CCP1M0 = 0;
```

```
CCP1M1 = 0;
```

```
CCP1M2 = 1;
```

```
CCP1M3 = 1;
```

```
CCP2M0 = 0;
```

```
CCP2M1 = 0;
```

```
CCP2M2 = 1;
```

```
CCP2M3 = 1;
```

```
CCP3M0 = 1;
```

```
CCP3M1 = 1;
```

```
CCP3M2 = 1;
```

```
CCP3M3 = 1;
```

```
CCP4M0 = 1;
```

```
CCP4M1 = 1;
```

```

CCP4M2 = 1;
CCP4M3 = 1;
}

```

```

set_pwm1(ubit08 dc1,ubit08 dc2, ubit08 dc3, ubit08 dc4)

```

```

{
ubit08 next_state;
ubit16 x1,x2,x3,x4;
ubit32 pulse_width,d1,d2,d3,d4 ;
d1= ((dc1*Fosc)/(pwm_frq*100*timer_prescale));
d2= ((dc2*Fosc)/(pwm_frq*100*timer_prescale));
d3= ((dc3*Fosc)/(pwm_frq*100*timer_prescale));
d4= ((dc4*Fosc)/(pwm_frq*100*timer_prescale));
x1=round(d1);
x2=round(d2);
x3=round(d3);
x4=round(d4);
DC1B0= x1 ;
x1 = x1>>1;
DC1B1= x1 ;
x1 = x1>>1;
CCPR1L= x1 ;
DC2B0= x2 ;
x2 = x2>>1;
DC2B1= x2 ;
x2 = x2>>1;
CCPR2L= x2 ;
DC3B0= x3 ;
x3 = x3>>1;

```

```
DC3B1= x3 ;  
x3 = x3>>1;  
CCPR3L=x3;  
DC4B0= x4 ;  
x4 = x4>>1;  
DC4B1= x4 ;  
x4 = x4>>1;  
CCPR4L=x4;  
  
}
```

```
void main(void)  
{  
    ubit08 next_state;  
    mc_conf();  
    set_pwm1(60, 30, 50, 20);  
    //set_pwm1(80);  
}
```