C:/Users/User/MPLABXProjects/2PHpwmmain.c

```c
/*
 * File:2PhasePWMv1.1.c
 * Author: vidura
 * This is the firmware for the 2PhasePWMv1.1 module
 * Created on 21 de abril de 2019, 14:22
 */

#include <stdint.h>
#include <stdio.h>
#include<stdlib.h>
#include <pic.h>
#include <xc.h>
//configuration bits
#define _XTAL_FREQ 1000000

#pragma config FOSC=EC , WDTE=OFF , PWRTE=ON , MCLRE=OFF , CP=ON, CPD=OFF
#pragma config BOREN=ON, IESO=OFF , FCMEN=OFF



unsigned int Read_ADC_Value(void)
{
    unsigned int ADCValue;

    ADCON0bits.GO_nDONE = 1;          // start conversion
    while (ADCON0bits.GO_nDONE);      // wait for conversion to finish
    ADCValue = ADRESH;
    ADCValue >>= 3;                   // get the 8 msbs of the result and trunkate to 5 bits
    return (ADCValue);                // return the 5 bit result in a single variable
}






void main(void)
{
    //unsigned int adresult = 0;
        // analog input config
        TRISA = 0b00000101;
        TRISB = 0b01110000;
        TRISC = 0b11011111;

        ANSEL0 = 0b11010101;
        ANSEL1 = 0b00000000;


        //setup comparators
         CM1CON0 = 0b10011010;  // C1 on, outp internal, INV POL, hi speed, +in RA0,-in RC2
         CM2CON0 = 0b10011011;  // C2 on, outp internal, INV POL, hi speed, +in RC0,-in RC3
        // setup 2Phase module
         PWMCON0 = 0b10111111;  // autorestart, blanking ph 1+2 ,sync master, Out1+2 enable
         PWMCON1 = 0b00011111;  // complemmentary drive disabled,deadtime155ns
         PWMCLK = 0b00011111;   // prescaler Fosc/1,  periode 32 pwmclks
         PWMPH1 = 0b00100000;   // Comp1 shtdwn enabled, start 1pwmclk.
         PWMPH2 = 0b01010000;   // Comp2 shtdwn enabled, start16pwmclk(180°)
         TRISCbits.TRISC1 = 0;
         TRISCbits.TRISC4 = 0;  //enable output PH1+2

         ADCON1bits.ADCS = 0b111;  // dedicated RC intosc for ADC
         ADCON0bits.ADFM = 0;      // left justified
         ADCON0bits.VCFG = 0;      // Vdd volatge reverence
         ADCON0bits.CHS = 0b0010;
         ADCON0bits.ADON=1;            //turn adc ON


         do
         {          if(PORTCbits.RC7 == 1)               // check dipswitch RC7
         {
                        //PWMPH1bits.PH = Read_ADC_Value();    // and assign corresponding register
                         PWMCON1bits.CMDLY = Read_ADC_Value();
                         __delay_us(10);
         }                                                     // for  loading the  AD Result
                    else
                         PWMPH2bits.PH = Read_ADC_Value();
                         __delay_us(10);
                         PWMCON1bits.COMOD1 = PORTCbits.RC6;   // load values for dipswitches on RC6, RB6,RB5
                         __delay_us(10);
                         PWMCLKbits.PWMP1 = PORTBbits.RB6;
                         __delay_us(10);
                         PWMCLKbits.PWMP0 = PORTBbits.RB5;
                         __delay_us(10);

         }
         while(1)              //repeate for ever
         ;
}
```