```c
//*****************************************************************
//ECGR 5101
//Lab 08
//Naseeruddin Lodge
//Srikar Thakkallapally
//Group 12
//
//Two MSP430G2553 are connected to each other. Chip1 is connected
//to an UltraSonic Sensor and a buzzer. Chip2 is connected to
//a quad-digit 7-segment LED. Chip 1 get the value for UltraSonic
//Sensor using PWM and increases the buzzer's volume if an object
//gets closer to the UltraSonic Sensor. Seconds Chip receives the
//distance measured by UltraSonic Sensor by UART and displays that
//value in CM on the quad-digit 7-segment LED.
//*****************************************************************
#include <msp430.h>
#include <stdint.h>
#include <string.h>



//Global Variables
int milisec, distance, sensorVal, temp=0;
int US_Sensor_Reading[11];
unsigned int count=0;



//Prototyped Functions
void UART_Setup();
void TimerA_Setup();
void PWM_Setup();
void US_Sensor_Setup();
void Buzzer_Setup();
int  uartReceiveData();
void uartTransmitData(int ADCval);
void DisplayLED(int n_SingleDigit);
void Control_Dx(int n);
void Setup_Chip_one();
void Setup_Chip_two();
void Program_Chip_one();
void Program_Chip_two();



//*****************************************************************
//Main Function
//*****************************************************************
int main(void){

    //Stop watchdog timer
    WDTCTL = WDTPW | WDTHOLD;


    //Check P1.4 to see if it is connected to GND
    //or VCC. Chip1 is connected to GND and Chip2
    //is connected to VCC. Then setups up the peripherals
    //and pins accordingly.
    if(!(P1IN & BIT4)) Setup_Chip_one();
    else               Setup_Chip_two();


    while(1){

        //Check P1.4 to see if it is connected to GND
        //or VCC. Chip1 is connected to GND and Chip2
        //is connected to VCC. Then uploads the code
        //accordingly.
        if(!(P1IN & BIT4)) Program_Chip_one();
        else               Program_Chip_two();
    }

}



//*****************************************************************
//Name    : UART_Setup()
//Input   : void
//Returns : void
//
//Function to Setup UART
//*****************************************************************
void UART_Setup(){

    //Clear DCO
    DCOCTL = 0;

    //Set to 1MHz
    //MCLK = SMCLK = 1MHZ
    BCSCTL1 = CALBC1_1MHZ;
    DCOCTL = CALDCO_1MHZ;

    //P1.1 = RX = BIT1, P1.2 = TX = BIT2
    P1SEL  |= BIT1 + BIT2;
    P1SEL2 |= BIT1 + BIT2;

    //Disable USCI, reset mode
    UCA0CTL1 |= UCSWRST;

    //SMCLK
    UCA0CTL1 |= UCSSEL_2;

    //1MHz
    //Baud Rate -> 9600
    UCA0BR0 = 104;
    UCA0BR1 = 0;

    //Modulation UCBRSx = 1
    UCA0MCTL = UCBRS0;
    //Initialize USCI state machine
    UCA0CTL1 &= ~UCSWRST;

}



//*****************************************************************
//Name    : TimerA_Setup()
//Input   : Void
//Returns : Void
//
//Function to Setup Timer
//*****************************************************************
void TimerA_Setup(){

    //Resolution(Delay per TAR Count) in Seconds =
    //(DIV / Input Clock in HZ)
    //1/1MHZ = 1 * 10^-6 sec


    //CCR0 interrupt enabled
    CCTL0 = CCIE;

    //1ms at 1 MHZ
    CCR0 = 1000;

    //SMCLK, upmode
    TACTL = TASSEL_2 + MC_1;


}



//*****************************************************************
//Name    : PWM_Setup()
//Input   : Void
//Returns : Void
//
//Function to Setup PWM
//*****************************************************************
void PWM_Setup(){

    //PWM period
    TA1CCR0 = 1000;

    //CCR1 PWM Duty Cycle
    TA1CCR1 = 1;

    //CCR1 selection reset-set
    TA1CCTL1 = OUTMOD_7;

    //SMCLK submain clock,upmode
    TA1CTL = TASSEL_2 | MC_1;


}



//*****************************************************************
//Name    : Port_1(void)
//Input   : Void
//Returns : Void
//
//ISR for Echo Pin
//*****************************************************************
#pragma vector=PORT1_VECTOR
__interrupt void Port_1(void){

    //Check interrupt Status
    if(P1IFG & 0x40){

        //Check rising edge
        if(!(P1IES & 0x40)){

            //Clear timer A
            TACTL|=TACLR;
            milisec = 0;

            //Set to Falling edge
            P1IES |= 0x40;
        }
        else{

            //ECHO length
            sensorVal = milisec*1000 + TAR;
        }

        //Clear flag
        P1IFG &= ~0x40;
    }
}



//*****************************************************************
//Name    : Timer_A(void)
//Input   : Void
//Returns : Void
//
//ISR for Timer
//*****************************************************************
#pragma vector=TIMER0_A0_VECTOR
__interrupt void Timer_A(void){

    milisec++;
}
```

```c
//*****************************************************************
//Name    : US_Sensor_Setup()
//Input   : void
//Returns : void
//
//Function to Setup all pins related to UltraSonic Sensor
//*****************************************************************
void US_Sensor_Setup(){

    //Disable interupt
    P1IE &= ~0x01;

    //Trigger to P1.5
    P1DIR |= 0x20;

    //Generate pulse from Trigger
    P1OUT |= 0x20;

    //Generate pulse from Trigger for 10us
    __delay_cycles(10);

    //Stop pulse from Trigger
    P1OUT &= ~0x20;

    //Echo to P1.6
    P1DIR &= ~0x40;

    //Clear Flag
    P1IFG = 0x00;

    //Enable interrupt for ECHO pin
    P1IE |= 0x40;

    //Set ECHO PIN to rising edge
    P1IES &= ~0x40;
}



//*****************************************************************
//Name    : Buzzer_Setup()
//Input   : void
//Returns : void
//
//Function to Setup all pins related to UltraSonic Sensor
//*****************************************************************
void Buzzer_Setup(){

    //Set Direction of buzzer as output
    P2DIR |= BIT2;

    //Set select Pin for P2.1
    P2SEL |= BIT2;
}



//*****************************************************************
//Name    : uartReceiveData()
//Input   : void
//Returns : int
//
//Checks if USCI_A0 RX has been received and returns it
//*****************************************************************
int uartReceiveData(){

    //Check if USCI_A0 RX has been received
    while (!(IFG2 & UCA0RXIFG));

    return UCA0RXBUF;
}



//*****************************************************************
//Name    : uartTransmitData(int ADCval)
//Input   : int
//Returns : void
//
//Checks if USCI_A0 TX buffer is ready and transmits the data
//*****************************************************************
void uartTransmitData(int ADCval){

    //Check if USCI_A0 TX buffer is ready
    while(!(IFG2 & UCA0TXIFG));

    UCA0TXBUF = ADCval;
}



//*****************************************************************
//Name    : DisplayLED(char n_SingleDigit)
//Input   : char
//Returns : void
//
//Used to display a number.
//char 0-9 is given as an input and the corresponding
//light turns on using a switch statement
//Case 10 is used to turn off LED and turn off All the LED digits
//*****************************************************************
void DisplayLED(int n_SingleDigit){

    switch(n_SingleDigit){

        //P2OUT = 0xhgfedcba
        //0x00111111
```

```
case 0:
    P2OUT = 0xC0;
    break;

//0x00000110
case 1:
    P2OUT = 0xF9;
    break;

//0x01011011
case 2:
    P2OUT = 0xA4;
    break;

//0x01001111
case 3:
    P2OUT = 0xB0;
    break;

//0x01100110
case 4:
    P2OUT = 0x99;
    break;

//0x01101101
case 5:
    P2OUT = 0x92;
    break;

//0x01111101
case 6:
    P2OUT = 0x82;
    break;

//0x00000111
case 7:
    P2OUT = 0xF8;
    break;

//0x01111111
case 8:
    P2OUT = 0x80;
    break;

//0x01110111
case 9:
    P2OUT = 0x90;
    break;

case 10:
    P1OUT = 0x00;
    P2OUT = 0xFF;
    break;
    }

}



//*****************************************************************
//Name    : Control_Dx(int n)
//Input   : int
//Returns : void
//
//This function is responsible for displaying the numbers.
//It gets an input of the reading of the potentiometer.
//Depending on the number of digits required, the digits are turned on.
//Then the Display LED function is used to turn on the numbers, one
//number at a time.
//If the ADC number is 357. 7 is displayed on D1, 5 on D2 and 3 on D3.
//DisplayLED(10) is used to turn off everything
//*****************************************************************
void Control_Dx(int n){

    int SingleDigit;

    //if n = 271, 2 is displayed on D1
    //then D1 is turned off and D2 is turned on
    //and 7 is displayed. Then D3 is turned on
    //and 1 is displayed

    if(n <= 9){

        //D4 - 0x00100000

        P1OUT = 0x01;
        SingleDigit = n;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

    }

    else if((n>=10) && (n<=99)){

        //D4 - 0x00100000
        //D3 - 0x00010000

        DisplayLED(10);

        P1OUT = 0x01;
        SingleDigit = n % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);

        P1OUT = 0x20;
        SingleDigit = n / 10 % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);
    }

    else if((n>=100) && (n<=999)){

        //D4 - 0x00100000
        //D3 - 0x00010000
        //D2 - 0x00000100

        P1OUT = 0x01;
        SingleDigit = n % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);

        P1OUT = 0x20;
        SingleDigit = n / 10 % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);

        P1OUT = 0x40;
        SingleDigit = n / 100 % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);
    }

    else if(n >= 1000){

        //D4 - 0x00100000
        //D3 - 0x00010000
        //D2 - 0x00000100
        //D1 - 0x00000010

        P1OUT = 0x01;
        SingleDigit = n % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);

        P1OUT = 0x20;
        SingleDigit = n / 10 % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);
        P1OUT = 0x40;
        SingleDigit = n / 100 % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);

        P1OUT = 0x80;
        SingleDigit = n / 1000 % 10;
        DisplayLED(SingleDigit);
        __delay_cycles(10000);

        DisplayLED(10);
    }


}



//******************************************************************
//Name    : Setup_Chip_one()
//Input   : void
//Returns : void
//
//Sets up all the ports and pins used by Chip 1
//******************************************************************
void Setup_Chip_one(){

    //Function to Setup UART
    UART_Setup();

    P1IFG  = 0x00;

    //Buzzer Setup
    Buzzer_Setup();

    //Timer Setup
    TimerA_Setup();

    //PWM Setup
    PWM_Setup();

    //Enable Interrupts
    _enable_interrupts();


}



//******************************************************************
//Name    : void Setup_Chip_two()
//Input   : void
//Returns : void
```

```
//
//Sets up all the ports and pins used by Chip 2
//*****************************************************************
void Setup_Chip_two(){

    //Set XIN and XOUT to GPIO
    P2SEL = 0;
    P2SEL2 = 0;

    //P1.0,P1.5,P1.6,P1.7 -> D1,D2,D3,D4
    P1DIR = 0xE1;

    //Set P2.0 - P2.5 to output
    //P2.0 - P2.7 -> abcdefgh
    P2DIR = 0xFF;

    //Function to setup UART
    UART_Setup();

}



//*****************************************************************
//Name    : Program_Chip_one()
//Input   : void
//Returns : void
//
//Program to be uploaded to Chip 2 This program is placed into an
//infinite while loop so it keeps occurring forever
//*****************************************************************
void Program_Chip_one(){

    unsigned int j,k;

    //Sets up UltraSonic Sensor to read distance
    US_Sensor_Setup();

    //Delay for 30ms
    //If no object it found, ECHO times out
    __delay_cycles(30000);

    //Converting ECHO value to CM
    distance = sensorVal/58;


    //Get 10 Ultrasonic Readings in an array
    if(count <= 11){

        US_Sensor_Reading[count] = distance;
        count++;
    }

    //Sort the array with UltraSonic Sensor readings in
    //ascending order and their median will be the
```

```c
    //distance in CM
    else if(count == 12){

        for(k=0; k<11; k++){

            for (j = 0; j+1<11-k; j++){


                if (US_Sensor_Reading[j] > US_Sensor_Reading[j + 1]){

                    temp = US_Sensor_Reading[j];
                    US_Sensor_Reading[j] = US_Sensor_Reading[j + 1];
                    US_Sensor_Reading[j + 1] = temp;
                }
            }
        }

        temp = US_Sensor_Reading[6];

        //Five levels of sound for UltraSonic Sensor
        if(temp>=0 && temp<=80) TA1CCR1 = 250;
        else if(temp>=81 && temp<=160) TA1CCR1 = 100;
        else if(temp>=161 && temp<=240) TA1CCR1 = 50;
        else if(temp>=241 && temp<=320) TA1CCR1 = 15;
        else if(temp>=321) TA1CCR1 = 5;

        count = 0;
    }


    //Send UltraSonic Sensor readings to Chip 2
    uartTransmitData(temp/2);

}



//*******************************************************************
//Name    : Program_Chip_two()
//Input   : void
//Returns : void
//
//Program to be uploaded to Chip 2 This program is placed into an
//infinite while loop so it keeps occurring forever
//*******************************************************************
void Program_Chip_two(){

    //Display ADC Value from Chip 1 on the LED
    Control_Dx(uartReceiveData()*2);

}
```