Proteus Files

Project files

Arduino Due hardware code

```
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <LiquidCrystal_I2C.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>
#define DHTPIN 6
#define DHTTYPE    DHT11  DHT_Unified
dht(DHTPIN, DHTTYPE);  uint32_t
delayMS;


LiquidCrystal_I2C lcd(0x27,16,2);
#define REPORTING_PERIOD_MS 1000
 const int trigPin = 2;
const int echoPin = 3;
MAX30100 sensor;  int a;
PulseOximeter pox;  long
duration; int distance;
uint32_t tsLastReport = 0;


void onBeatDetected()
{
    Serial.println("Beat!");
}  void setup() {
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(A0,OUTPUT);
pinMode(A1,OUTPUT);
pinMode(A2,OUTPUT);
pinMode(A3,OUTPUT);
Serial.begin(9600);
    Serial1.begin(9600);
     dht.begin();     sensor_t sensor1;
dht.temperature().getSensor(&sensor1);
dht.humidity().getSensor(&sensor1);
delayMS = sensor1.min_delay / 1000;
    lcd.init();
lcd.backlight();     if
(!pox.begin()) {
Serial.println("FAILED");
for(;;);
    } else {
        Serial.println("SUCCESS");
    }
    pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);
}  void loop()
{
pox.update();
  if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
      Serial.print("Heart rate:");
      Serial.print(pox.getHeartRate());
      Serial.print("bpm / SpO2:");
      Serial.print(pox.getSpO2());
Serial.println("%");       sensors_event_t
event;
dht.temperature().getEvent(&event);
lcd.clear();       lcd.setCursor(0,0);
lcd.print("T: ");
lcd.setCursor(2,0);
lcd.print(event.temperature);
             lcd.setCursor(9,0);
lcd.print("HB: ");
```

```
lcd.setCursor(12,0);
lcd.print(pox.getHeartRate());
while(Serial1.available())
      {         char
t=Serial1.read();
//Serial.println(t);
if(t=='F')
        {
          Serial.println("FORWARD");
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);           duration
= pulseIn(echoPin, HIGH);           distance=
duration*0.034/2;           pox.begin();
          pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);
           if(distance>15)
{
digitalWrite(A0,LOW);
digitalWrite(A1,HIGH);
digitalWrite(A2,LOW);
digitalWrite(A3,HIGH);
lcd.setCursor(0,1);
lcd.print("FORWARD");
pox.begin();
            pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);
          }
if(distance<15)           {
digitalWrite(A0,LOW);
digitalWrite(A1,LOW);
digitalWrite(A2,LOW);
digitalWrite(A3,LOW);
pox.begin();
          pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);


          }         }
if(t=='B')         {
digitalWrite(A0,HIGH);
digitalWrite(A1,LOW);
digitalWrite(A2,HIGH);
digitalWrite(A3,LOW);
lcd.setCursor(0,1);
lcd.print("REVERSE");
pox.begin();
          pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);
        }         if(t=='R')
{
digitalWrite(A0,LOW);
digitalWrite(A1,HIGH);
digitalWrite(A2,HIGH);
digitalWrite(A3,LOW);
lcd.setCursor(0,1);
lcd.print("RIGHT");
pox.begin();
          pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);
        }
if(t=='L')         {
          digitalWrite(A0,HIGH);
digitalWrite(A1,LOW);
digitalWrite(A2,LOW);
digitalWrite(A3,HIGH);
lcd.setCursor(0,1);
lcd.print("LEFT");           pox.begin();
          pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);
```

```
        }
if(t=='S')         {
          digitalWrite(A0,LOW);
digitalWrite(A1,LOW);
digitalWrite(A2,LOW);
digitalWrite(A3,LOW);           pox.begin();
          pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
pox.setOnBeatDetectedCallback(onBeatDetected);


        }
}
      tsLastReport = millis();
  }

}
void configureMax30100() {
sensor.setMode(MAX30100_MODE_SPO2_HR);
  sensor.setLedsCurrent(MAX30100_LED_CURR_50MA, MAX30100_LED_CURR_27_1MA);
sensor.setLedsPulseWidth(MAX30100_SPC_PW_1600US_16BITS);
sensor.setSamplingRate(MAX30100_SAMPRATE_100HZ);   sensor.setHighresModeEnabled(true);
}
end
```

AVR Proteus Arduino Code

```
#include "dht11.h"
#include "LCD.h"
#include <TimerOne.h>
DHT dht; char c;
#define D_temp 4
#define motor1_pin1 22
#define motor1_pin2 23
#define motor2_pin1 24
#define motor2_pin2 25
#define trigpin 2
#define echopin 3

int pos = 0;  // variable to store the servo
int HBSensor = 7; //   HB i/p int HBCount =
0; //    HB start button int HBCheck = 0;
int TimeinSec = 0; int HBperMin = 0; int
HBStart = 6; int HBStartCheck = 0;

void timerIsr()
{
  if(HBStartCheck == 1)
  {
      TimeinSec = TimeinSec + 1;
      lcd.setCursor(14,0);
lcd.print(TimeinSec);
lcd.print(" ");
  }
}  void setup() {   lcd.begin(16,
2);   lcd.clear();
Serial.begin(9600);   pinMode(2,
OUTPUT);//ULTRASONIC op   pinMode(3,
INPUT); //ULTRASONIC ip
pinMode(27, INPUT_PULLUP); //F
pinMode(28, INPUT_PULLUP); //B
pinMode(29, INPUT_PULLUP); //S
pinMode(30, INPUT_PULLUP); //L
pinMode(26, INPUT_PULLUP); //R
```

```
  pinMode(motor1_pin1, OUTPUT);   /* Motor1 control pin 1 */
pinMode(motor1_pin2, OUTPUT);   /* Motor1 control pin 2 */
pinMode(motor2_pin1, OUTPUT);   /* Motor2 control pin 1 */
pinMode(motor2_pin2, OUTPUT);   /* Motor2 control pin 2 */
pinMode(HBSensor, INPUT); // HB i/p                  pinMode(HBStart,
INPUT_PULLUP); //HB start Button
  Timer1.initialize(800000);
Timer1.attachInterrupt( timerIsr );
lcd.clear();   lcd.setCursor(0, 0);
delay(250);
lcd.print("WHEELCHAIR");
delay(1000);   lcd.clear();
lcd.setCursor(9,0);
lcd.print("Time:");
lcd.setCursor(0,1);
lcd.print("HBperMin: 0");


}
//Function To Recieve Data From Bluetooth Device and move
void loop() { //DHT11 reading   dht.dht_read(D_temp);
lcd.setCursor(0, 0);   lcd.print("T:");
lcd.print(dht.temperature);   //HB Sensor Reading
  if(digitalRead(HBStart) == LOW){HBStartCheck = 1;}
if(HBStartCheck == 1)
  {
      if((digitalRead(HBSensor) == HIGH) && (HBCheck == 0))
      {
        HBCount = HBCount + 1;
HBCheck = 1;
lcd.setCursor(13,1);
//lcd.print(HBCount);
        //lcd.print(" ");
      }
      if((digitalRead(HBSensor) == LOW) && (HBCheck == 1))
      {
        HBCheck = 0;
      }
      if(TimeinSec == 10)
      {
          HBperMin = HBCount * 6;
HBStartCheck = 0;
lcd.setCursor(10,1);
lcd.print(HBperMin);           lcd.print(" ");
HBCount = 0;
          TimeinSec = 0;
      }
  }
//Ultrasonic detection   long
duration, distance;   digitalWrite(2,
LOW); //   delayMicroseconds(2);
digitalWrite(2, HIGH);
delayMicroseconds(10);
digitalWrite(2, LOW);     duration =
pulseIn(3, HIGH);   distance =
duration/58.2;   if(distance < 30) {
lcd.setCursor(0, 1);
lcd.print("Obstacle ahead ");
Serial.println("Obstacle ahead ");
    // Stop the motors
digitalWrite(motor1_pin1, LOW);
digitalWrite(motor1_pin2, LOW);
digitalWrite(motor2_pin1, LOW);
```

```
digitalWrite(motor2_pin2, LOW);
delay(400);     lcd.clear();
  }
else   {
    while (Serial.available()) //Function To Recieve Data From Bluetooth Device and move
    {
      char c = (char)Serial.read();
if (c == 'f')
      {
        Serial.print("FORWARD");
lcd.setCursor(0, 1);         lcd.print("FORWARD
");         digitalWrite(motor1_pin1, HIGH);
digitalWrite(motor1_pin2, LOW);
digitalWrite(motor2_pin1, HIGH);
digitalWrite(motor2_pin2, LOW);

}
      if (c == 'b')
      {
        Serial.print("REVERSE");
lcd.setCursor(0, 1);         lcd.print("REVERSE
");         digitalWrite(motor1_pin1, LOW);
digitalWrite(motor1_pin2, HIGH);
digitalWrite(motor2_pin1, LOW);
digitalWrite(motor2_pin2, HIGH);
      }
      if (c == 's')
      {
        Serial.print("STOP");
lcd.setCursor(0, 1);         lcd.print("STOP
");         digitalWrite(motor1_pin1, LOW);
digitalWrite(motor1_pin2, LOW);
digitalWrite(motor2_pin1, LOW);
digitalWrite(motor2_pin2, LOW);
      }
      if (c== 'l')
       {
        Serial.print("LEFT");
        lcd.setCursor(0,             1);
lcd.print("LEFT                      ");
digitalWrite(motor1_pin1,         HIGH);
digitalWrite(motor1_pin2,          LOW);
digitalWrite(motor2_pin1,          LOW);
digitalWrite(motor2_pin2, HIGH);
      }       if
(c== 'r')
       {
        Serial.print("RIGHT");
lcd.setCursor(0,                     1);
lcd.print("RIGHT                     ");
digitalWrite(motor1_pin1,          LOW);
digitalWrite(motor1_pin2,         HIGH);
digitalWrite(motor2_pin1,         HIGH);
digitalWrite(motor2_pin2, LOW);
      }
delay(1000);     }
    if (digitalRead(27) == 0)
{
      Serial.print("FORWARD");
lcd.setCursor(0,                   1);
lcd.print("FORWARD                ");
digitalWrite(motor1_pin1,       HIGH);
digitalWrite(motor1_pin2,        LOW);
```

```
digitalWrite(motor2_pin1,       HIGH);
digitalWrite(motor2_pin2, LOW);

}
    if (digitalRead(28) == 0)
{
      Serial.print("REVERSE");
lcd.setCursor(0,                   1);
lcd.print("REVERSE                 ");
digitalWrite(motor1_pin1,        LOW);
digitalWrite(motor1_pin2,       HIGH);
digitalWrite(motor2_pin1,        LOW);
digitalWrite(motor2_pin2, HIGH);
    }
    if (digitalRead(29) == 0)
{
      Serial.print("STOP");
lcd.setCursor(0,                   1);
lcd.print("STOP                    ");
digitalWrite(motor1_pin1,       LOW);
digitalWrite(motor1_pin2,       LOW);
digitalWrite(motor2_pin1,       LOW);
digitalWrite(motor2_pin2, LOW);
    }
    if (digitalRead(30) == 0)
{
      Serial.print("LEFT");
lcd.setCursor(0,                   1);
lcd.print("LEFT                    ");
digitalWrite(motor1_pin1,       HIGH);
digitalWrite(motor1_pin2,        LOW);
digitalWrite(motor2_pin1,        LOW);
digitalWrite(motor2_pin2, HIGH);
    }
    if (digitalRead(26) == 0)
{
      Serial.print("RIGHT");
lcd.setCursor(0, 1);
lcd.print("RIGHT      ");
digitalWrite(motor1_pin1, LOW);
digitalWrite(motor1_pin2,
HIGH);
digitalWrite(motor2_pin1,
HIGH);
digitalWrite(motor2_pin2, LOW);
    }   }
delay(1000);
}
```