#include <SoftwareSerial.h>

#include <Wire.h>

#include <HCSR04.h>

#include "MAX30100\_PulseOximeter.h"

//~~~~~~ESP Serial~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

SoftwareSerial mySerial(9, 10); // RX, TX

//~~~~~~~EYE~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

const int eye\_blink = 3;

String eb;

//~~~~~~GLUCOSE LEVEL~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

const int glucose\_level = 13;

String gl;

int glucoseData = 0;

const int glucoseDigitalOutput = 8;

bool glucoseState = false;

//~~~~~~~MAX30100~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

#define REPORTING\_PERIOD\_MS 1000

PulseOximeter pox;

uint32\_t tsLastReport = 0;

int HeartBeatValueInt, OxygenPercentInt;

String hb, oxy;

double high\_pulse\_value, low\_pulse\_value;

void onBeatDetected() {

Serial.println("Beat!");

}

//~~~~~~~~~~~~~~~Flex Sensor~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

String ub, in\_ub = "Urine Bag Empty";

int flexSensorData, flexSensorData\_prev1, flexSensorData\_prev2, flexSensorData\_prev3;

const int flexAnalogInput = A0;

const int flexDigitalOutput = 15;

#define FLEX\_PERIOD\_MS 1000

uint32\_t flexLastReport = 0;

//~~~~~~~~~~~~~~Robot~~~~~~~~~~~~~

#define MOTOR1\_PINA 4

#define MOTOR1\_PINB 5

#define MOTOR2\_PINA 6

#define MOTOR2\_PINB 7

int LS = 16; // left sensor

int RS = 17; // right sensor

String voice;

bool enableLF = false;

UltraSonicDistanceSensor distanceSensor(11, 12);

//~~~~~~~~~SETUP~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void setup() {

Serial.begin(9600);

mySerial.begin(9600);

pinMode(MOTOR1\_PINA, OUTPUT);

pinMode(MOTOR1\_PINB, OUTPUT);

pinMode(MOTOR2\_PINA, OUTPUT);

pinMode(MOTOR2\_PINB, OUTPUT);

pinMode(glucose\_level, INPUT);

pinMode(eye\_blink, INPUT);

pinMode(flexDigitalOutput, OUTPUT);

pinMode(glucoseDigitalOutput, OUTPUT);

digitalWrite(glucoseDigitalOutput, HIGH);

digitalWrite(flexDigitalOutput, HIGH);

delay(10);

Serial.print("Initializing pulse oximeter..");

if (!pox.begin()) {

Serial.println("FAILED");

for (;;);

} else {

Serial.println("SUCCESS");

}

pox.setOnBeatDetectedCallback(onBeatDetected);

}

String getGlucoseLevel() {

// Serial.println("getGlucoseLevel");

glucoseData = digitalRead(glucose\_level);

if (glucoseData == HIGH && glucoseState == false) {

digitalWrite(glucoseDigitalOutput, LOW); delay(10);

glucoseState = true;

}

if (glucoseData == LOW && glucoseState == true) {

digitalWrite(glucoseDigitalOutput, HIGH); delay(10);

glucoseState = false;

}

if (glucoseData == HIGH) {

Serial.println("Glucose level Low");

return "Low";

} else {

Serial.println("Glucose level Normal");

return "Normal";

}

}

String getEyeBlink() {

// Serial.println("getEyeBlink");

if (digitalRead(eye\_blink)) {

Serial.println("Eye Open");

return "Open ";

} else {

Serial.println("Eye Close");

return "Close";

}

}

String getUrineBagStatus() {

// Serial.println("getUrineBagStatus");

if (millis() - flexLastReport > FLEX\_PERIOD\_MS) {

flexSensorData\_prev3 == flexSensorData\_prev2;

flexSensorData\_prev2 == flexSensorData\_prev1;

flexSensorData\_prev1 == flexSensorData;

flexSensorData = analogRead(flexAnalogInput);

Serial.print("Flex Data:");

Serial.println(flexSensorData);

if (flexSensorData > 700 && flexSensorData\_prev1 > 700 && flexSensorData\_prev2 > 700 && flexSensorData\_prev3 > 700) {

digitalWrite(flexDigitalOutput, HIGH); delay(10);

Serial.println("~~~~~~~~~~~~~~~~Urine Bag Empty");

in\_ub = "Empty";

}

if (flexSensorData < 700 && flexSensorData\_prev1 < 700 && flexSensorData\_prev2 < 700 && flexSensorData\_prev3 < 700) {

digitalWrite(flexDigitalOutput, LOW); delay(10);

Serial.println("~~~~~~~~~~~~~~~~Urine Bag Full");

in\_ub = "Full";

}

flexLastReport = millis();

}

return in\_ub;

}

void Send2NodeMCU() {

mySerial.print(hb);

mySerial.print(",");

mySerial.print(oxy);

mySerial.print(",");

mySerial.print(gl);

mySerial.print(",");

mySerial.print(eb);

mySerial.print(",");

mySerial.print(ub);

mySerial.println(";");

}

void loop() {

pox.update();

serialEvent();

if (distanceSensor.measureDistanceCm() < 20) {

halt();

}

if (enableLF == false) {

if (millis() - tsLastReport > REPORTING\_PERIOD\_MS) {

// Serial.println("Read Oximeter Data");

HeartBeatValueInt = pox.getHeartRate();

OxygenPercentInt = pox.getSpO2();

hb = String(HeartBeatValueInt);

oxy = String(OxygenPercentInt);

tsLastReport = millis();

Serial.print("Heart Beat:"); Serial.print(hb);

Serial.print(", Oxygen level:"); Serial.println(oxy);

gl = getGlucoseLevel();

eb = getEyeBlink();

ub = getUrineBagStatus();

Send2NodeMCU();

}

}else{

lineFollower();

}

}

void serialEvent() {

while (mySerial.available()) {

delay(10);

char c = mySerial.read();

if (c == '#') {

break;

}

voice += c;

}

if (voice.length() > 0) {

if (voice == "forward" && enableLF == false)

forward();

else if (voice == "reverse" && enableLF == false)

reverse();

else if (voice == "left" && enableLF == false)

left();

else if (voice == "right" && enableLF == false)

right();

else if (voice == "stop" && enableLF == false)

halt();

else if (voice == "line" && enableLF == false)

enableLF = true;

else if (voice == "manual" && enableLF == true)

enableLF = false;

voice = "";

}

}

void forward() {

digitalWrite(MOTOR1\_PINA, HIGH);

digitalWrite(MOTOR1\_PINB, LOW);

digitalWrite(MOTOR2\_PINA, LOW);

digitalWrite(MOTOR2\_PINB, HIGH);

Serial.println("Robo goes Forward");

}

void reverse() {

digitalWrite(MOTOR1\_PINA, LOW);

digitalWrite(MOTOR1\_PINB, HIGH);

digitalWrite(MOTOR2\_PINA, HIGH);

digitalWrite(MOTOR2\_PINB, LOW);

Serial.println("Robo goes Reverse");

}

void left() {

digitalWrite(MOTOR1\_PINA, HIGH);

digitalWrite(MOTOR1\_PINB, LOW);

digitalWrite(MOTOR2\_PINA, HIGH);

digitalWrite(MOTOR2\_PINB, LOW);

Serial.println("Robo turns Left");

}

void right() {

digitalWrite(MOTOR1\_PINA, LOW);

digitalWrite(MOTOR1\_PINB, HIGH);

digitalWrite(MOTOR2\_PINA, LOW);

digitalWrite(MOTOR2\_PINB, HIGH);

Serial.println("Robo turns Right");

}

void halt() {

digitalWrite(MOTOR1\_PINA, LOW);

digitalWrite(MOTOR1\_PINB, LOW);

digitalWrite(MOTOR2\_PINA, LOW);

digitalWrite(MOTOR2\_PINB, LOW);

// Serial.println("Robo Stopped");

}

void lineFollower() {

if (digitalRead(LS) && digitalRead(RS)) // Move Forward

{

forward();

}

if (!(digitalRead(LS)) && digitalRead(RS)) // Turn right

{

right();

}

if (digitalRead(LS) && !(digitalRead(RS))) // turn left

{

left();

}

if (!(digitalRead(LS)) && !(digitalRead(RS))) // stop

{

halt();

}

}