**CODE PHẦN CỨNG:**

**I. Các phần đã hoàn thành:**

1. Thu thập và xử lý dữ liệu cảm biến

- MAX30102 (Heart Rate và SpO2):  
Đã tích hợp thành công sensor MAX30102 để đo nhịp tim và nồng độ oxy trong máu. Phần đọc dữ liệu từ cảm biến đã hoạt động ổn định, xử lý thuật toán phát hiện nhịp tim, tính toán giá trị BPM và SpO2 chính xác phù hợp với yêu cầu.

- MLX90614 (Nhiệt độ):  
Hoàn thành việc đo nhiệt độ cơ thể không tiếp xúc sử dụng cảm biến hồng ngoại MLX90614. Dữ liệu nhiệt độ được đọc đều đặn, xử lý mượt mà và cập nhật lên màn hình với mã màu cảnh báo dựa trên ngưỡng nhiệt độ cơ thể bình thường và sốt.

- MPU6050 (Đếm bước chân):  
Phân tích dữ liệu gia tốc từ MPU6050 đã được thực hiện để đếm bước chân bằng thuật toán dựa trên sự thay đổi lớn của độ lớn vector gia tốc. Thuật toán đếm bước chân chạy ổn định, đếm chính xác số bước và được hiển thị trên màn hình.

**2. Hiển thị dữ liệu trên màn hình LCD ST7789V2 SPI Interface:**

- Các chỉ số (nhiệt độ, bước chân, thời gian) được thể hiện trên màn hình TFT ST7789V2.

- Giao diện hiển thị bao gồm các thông tin cố định (nhãn) và vùng dữ liệu cập nhật liên tục với màu sắc cảnh báo thích hợp, đảm bảo người dùng dễ dàng quan sát.

**Tuy nhiên, hiện tại còn tồn tại các vấn đề sau:**

- Thời gian hiển thị xảy ra hiện tượng nháy màn hình do việc cập nhật giao diện chưa được tối ưu.

- Giá trị nhịp tim chưa được hiển thị ổn định và chính xác, chưa đạt chuẩn như demo video đã gửi.

A black rectangular object with wires and a white board

AI-generated content may be incorrect.

**Cấu hình Chân Kết nối:**

|  |
| --- |
| // Màn hình ST7789V2 (SPI)  #define TFT\_CS 5 // Chip Select  #define TFT\_DC 16 // Data/Command  #define TFT\_RST 17 // Reset  #define TFT\_MOSI 23 // Master Out Slave In  #define TFT\_SCLK 18 // Serial Clock  #define TFT\_BL 4 // Backlight  // I2C Bus cho cảm biến  Wire.begin(21, 22); // SDA = 21, SCL = 22 |

**II. Thực hiện:**

**2.1 Cấu hình PlatformIO:**

File platformio.ini với các thư viện cần thiết:

|  |
| --- |
| [env:esp32dev]  platform = espressif32  board = esp32dev  framework = arduino  monitor\_speed = 115200  lib\_deps =  adafruit/Adafruit GFX Library  adafruit/Adafruit ST7735 and ST7789 Library  adafruit/Adafruit MLX90614 Library  adafruit/Adafruit MPU6050  WiFi  arduino-libraries/NTPClient  sparkfun/SparkFun MAX3010x Pulse and Proximity Sensor Library |

File main.cpp:

|  |
| --- |
| #include <Arduino.h>  #include <WiFi.h>  #include <WiFiUdp.h>  #include <NTPClient.h>  #include <Wire.h>  #include <Adafruit\_GFX.h>  #include <Adafruit\_ST7789.h>  #include <Adafruit\_MLX90614.h>  #include "MAX30105.h"  #include "heartRate.h"  #include <Adafruit\_MPU6050.h>  #include <Adafruit\_Sensor.h>  #define ST77XX\_GRAY 0x8410  // Chân kết nối màn ST7789V2  #define TFT\_CS 5  #define TFT\_DC 16  #define TFT\_RST 17  #define TFT\_MOSI 23  #define TFT\_SCLK 18  #define TFT\_BL 4  // Khởi tạo đối tượng  Adafruit\_ST7789 tft(TFT\_CS, TFT\_DC, TFT\_MOSI, TFT\_SCLK, TFT\_RST);  Adafruit\_MLX90614 mlx;  MAX30105 pulseSensor;  Adafruit\_MPU6050 mpu;  // Wi-Fi và NTP  const char\* ssid = "YOUR\_SSID";  const char\* password = "YOUR\_PASSWORD";  WiFiUDP ntpUDP;  NTPClient timeClient(ntpUDP, "pool.ntp.org", 25200, 60000);  // Biến tạm  int prevSecond = -1;  float prevTemp = -1000;  int prevBPM = -1;  long stepCount = 0;  // Hàm vẽ các nhãn cố định  void drawStaticElements() {  tft.fillScreen(ST77XX\_BLACK);  tft.setTextColor(ST77XX\_CYAN);  tft.setTextSize(1);  tft.setCursor(10, 50); tft.print("NHIET DO:");  tft.setCursor(130,50); tft.print("NHIP TIM:");  tft.setCursor(60, 130); tft.print("BUOC CHAN:");  // Dấu ":" cho đồng hồ  tft.setTextSize(3);  tft.setTextColor(ST77XX\_YELLOW);  tft.setCursor(80, 10); tft.print(":");  tft.setCursor(140,10); tft.print(":");  }  void setup() {  Serial.begin(115200);  Wire.begin(21, 22);  pinMode(TFT\_BL, OUTPUT);  digitalWrite(TFT\_BL, HIGH);  // Khởi màn  tft.init(240, 280);  tft.setRotation(3);  drawStaticElements();  // Kết nối Wi-Fi  WiFi.begin(ssid, password);  while (WiFi.status() != WL\_CONNECTED) delay(200);  timeClient.begin();  // Khởi MLX90614  mlx.begin();  // Khởi MAX30102  pulseSensor.begin(); pulseSensor.setup(); pulseSensor.setPulseAmplitudeRed(60);  // Khởi MPU6050  mpu.begin(); mpu.setAccelerometerRange(MPU6050\_RANGE\_8\_G);  }  void loop() {  // Cập nhật đồng hồ  timeClient.update();  int h = timeClient.getHours();  int m = timeClient.getMinutes();  int s = timeClient.getSeconds();  if (s != prevSecond) {  prevSecond = s;  tft.fillRect(10, 10, 160, 30, ST77XX\_BLACK);  tft.setTextSize(3);  tft.setTextColor(ST77XX\_YELLOW);  tft.setCursor(10, 10);  tft.printf("%02d:%02d:%02d", h, m, s);  }  // Cập nhật nhiệt độ  static unsigned long lastTemp = 0;  if (millis() - lastTemp > 2000) {  lastTemp = millis();  float temp = mlx.readObjectTempC();  if (fabs(temp - prevTemp) > 0.1) {  prevTemp = temp;  tft.fillRect(10, 65, 100, 25, ST77XX\_BLACK);  tft.setTextSize(2);  tft.setTextColor(temp > 37.5 ? ST77XX\_RED :  temp > 37.0 ? ST77XX\_YELLOW : ST77XX\_GREEN);  tft.setCursor(10, 65);  tft.printf("%.1fC", temp);  }  }  // Cập nhật nhịp tim  static unsigned long lastBeat = 0;  uint32\_t ir = pulseSensor.getIR();  pulseSensor.nextSample();  if (ir > 50000 && checkForBeat(ir)) {  long delta = millis() - lastBeat;  lastBeat = millis();  int bpm = 60000 / delta;  if (bpm >= 40 && bpm <= 200 && bpm != prevBPM) {  prevBPM = bpm;  tft.fillRect(130, 65, 100, 25, ST77XX\_BLACK);  tft.setTextSize(2);  tft.setTextColor(bpm > 100 ? ST77XX\_RED : ST77XX\_MAGENTA);  tft.setCursor(130, 65);  tft.printf("%dBPM", bpm);  }  } else if (ir < 50000 && prevBPM != 0) {  prevBPM = 0;  tft.fillRect(130, 65, 100, 25, ST77XX\_BLACK);  tft.setTextSize(2);  tft.setTextColor(ST77XX\_GRAY);  tft.setCursor(130,65);  tft.print("--BPM");  }  // Cập nhật bước chân  sensors\_event\_t a, g, temp;  mpu.getEvent(&a, &g, &temp);  float mag = sqrt(a.acceleration.x\*a.acceleration.x +  a.acceleration.y\*a.acceleration.y +  a.acceleration.z\*a.acceleration.z);  static float lastMag = 0;  static bool stepFlag = false;  if (!stepFlag && fabs(mag - lastMag) > 2.5) {  stepCount++;  stepFlag = true;  } else if (stepFlag && fabs(mag - lastMag) < 1.0) {  stepFlag = false;  }  lastMag = mag;  tft.fillRect(60, 145, 120, 25, ST77XX\_BLACK);  tft.setTextSize(2);  tft.setTextColor(ST77XX\_GREEN);  tft.setCursor(60, 145);  tft.printf("%ld", stepCount);  delay(50);  } |

**2.2 Tiến độ hiện tại:**

**Đã hoàn thành tích hợp cảm biến:**

- MAX30105: Đo nhịp tim với thuật toán phát hiện đỉnh.

- MLX90614: Đo nhiệt độ cơ thể không tiếp xúc.

- MPU6050: Đếm bước chân dựa trên độ lớn gia tốc.

- ST7789V2: Hiển thị thông tin với màu sắc phân loại.

**Đã hoàn thành giao diện hiển thị:**

- Đồng hồ thời gian thực từ NTP server.

- Hiển thị nhiệt độ với mã màu (xanh <37°C, vàng 37-37.5°C, đỏ >37.5°C).

- Counter bước chân tích lũy.

**Đang tiến hành thực hiện tính năng sau:**

- Hiển thị nhịp tim: với validation (40-200 BPM).

- Xử lý nút nhấn: Chuyển đổi giữa các màn hình khác nhau.

- Quản lý nguồn: Mạch sạc và công tắc nguồn.

- Deep Sleep Mode: Tiết kiệm pin khi không sử dụng.

- FreeRTOS Multitasking: Xử lý đa nhiệm cho hiệu suất tối ưu.

- Sử dụng module thời gian thực RTC thay vì NTP server: Để thiết bị linh hoạt hơn thay vì kết nối cố định qua 1 WIFI.

**UPDATE:**

Báo cáo chỉnh sửa giao diện màn hình LCD TFT IPS 1.69 Inch và bổ sung bảng trạng thái thiết bị

Đã chỉnh sửa lại giao diện màn hình LCD TFT IPS, sử dụng driver ST7789V2 giao tiếp SPI nhằm cải thiện trải nghiệm người dùng. Giao diện mới được thiết kế lại , thêm vào bảng trạng thái hiển thị thông tin thiết bị.

Cảm biến nhịp tim MAX30102 hiện vẫn chưa nhận được tín hiệu chính xác từ thiết bị, chưa có module RTC (Real-Time Clock) để đồng bộ thời gian và thay thế cho việc sử dụng WiFi Server.

Code platformio.ini:

|  |
| --- |
| [env:esp32dev]  platform = espressif32  board = esp32dev  framework = arduino  lib\_deps =      adafruit/Adafruit GFX Library      adafruit/Adafruit ST7735 and ST7789 Library      adafruit/Adafruit MLX90614 Library      adafruit/Adafruit MPU6050      adafruit/Adafruit BusIO      https://github.com/sparkfun/SparkFun\_MAX3010x\_Sensor\_Library.git      arduino-libraries/NTPClient  build\_flags =      -D USER\_SETUP\_LOADED=1      -D ST7789\_DRIVER=1      -D TFT\_WIDTH=240      -D TFT\_HEIGHT=280      -D TFT\_MOSI=23      -D TFT\_SCLK=18      -D TFT\_CS=5      -D TFT\_DC=16      -D TFT\_RST=17      -D TFT\_BL=4      -D LOAD\_GLCD=1      -D LOAD\_FONT2=1      -D LOAD\_FONT4=1      -D LOAD\_FONT6=1      -D LOAD\_FONT7=1      -D LOAD\_FONT8=1      -D LOAD\_GFXFF=1      -D SMOOTH\_FONT=1      -D SPI\_FREQUENCY=27000000      -D SPI\_READ\_FREQUENCY=20000000  upload\_speed = 921600  monitor\_filters = esp32\_exception\_decoder |

Code main.cpp:

|  |
| --- |
| #include <Arduino.h>  #include <WiFi.h>  #include <WiFiUdp.h>  #include <NTPClient.h>  #include <Wire.h>  #include <Adafruit\_GFX.h>  #include <Adafruit\_ST7789.h>  #include <Adafruit\_MLX90614.h>  #include "MAX30105.h"  #include "heartRate.h"  #include <Adafruit\_MPU6050.h>  #include <Adafruit\_Sensor.h>  #define TFT\_CS 5  #define TFT\_DC 16  #define TFT\_RST 17  #define TFT\_MOSI 23  #define TFT\_SCLK 18  #define TFT\_BL 4  #define ST77XX\_GRAY 0x8410  #define ST77XX\_ORANGE 0xFD20  Adafruit\_ST7789 tft(TFT\_CS, TFT\_DC, TFT\_MOSI, TFT\_SCLK, TFT\_RST);  Adafruit\_MLX90614 mlx;  MAX30105 pulseSensor;  Adafruit\_MPU6050 mpu;  const char\* ssid = "PTIT.HCM\_SV";  const char\* password = "";  WiFiUDP ntpUDP;  NTPClient timeClient(ntpUDP, "pool.ntp.org", 25200, 60000); // GMT+7 offset  int prevSecond = -1;  float prevTemp = -1000;  int prevBPM = -1;  long stepCount = 0;  const byte RATE\_SIZE = 4;  byte rates[RATE\_SIZE];  byte rateSpot = 0;  long lastBeat = 0;  float beatsPerMinute = 0;  int beatAvg = 0;  unsigned long lastHeartRateCheck = 0;  unsigned long lastTempCheck = 0;  unsigned long lastStepCheck = 0;  unsigned long lastDisplayUpdate = 0;  const unsigned long HEART\_RATE\_INTERVAL = 20;  const unsigned long TEMP\_INTERVAL = 2000;  const unsigned long STEP\_INTERVAL = 50;  const unsigned long DISPLAY\_UPDATE\_INTERVAL = 100;  float lastMagnitude = 0;  bool stepFlag = false;  unsigned long lastStepTime = 0;  void drawStaticUI() {    tft.fillScreen(ST77XX\_BLACK);    tft.fillRect(0, 0, 240, 30, ST77XX\_BLUE);    tft.setTextColor(ST77XX\_WHITE);    tft.setTextSize(2);    String title = "HEALTH MONITOR";    int16\_t x1, y1;    uint16\_t w, h;    tft.getTextBounds(title, 0, 0, &x1, &y1, &w, &h);    tft.setCursor((240 - w) / 2, 8);    tft.print(title);    tft.setTextColor(ST77XX\_CYAN);    tft.setTextSize(1);    tft.setCursor(10, 45);    tft.print("THOI GIAN:");    tft.drawRect(5, 40, 230, 35, ST77XX\_CYAN);    tft.setCursor(10, 90);    tft.print("NHIP TIM:");    tft.drawRect(5, 85, 110, 50, ST77XX\_CYAN);    tft.setCursor(125, 90);    tft.print("NHIET DO:");    tft.drawRect(120, 85, 115, 50, ST77XX\_CYAN);    tft.setCursor(10, 150);    tft.print("BUOC CHAN:");    tft.drawRect(5, 145, 230, 50, ST77XX\_CYAN);    tft.setCursor(10, 210);    tft.print("TRANG THAI:");    tft.drawRect(5, 205, 230, 70, ST77XX\_CYAN);  }  void displayTime() {    timeClient.update();    int h = timeClient.getHours();    int m = timeClient.getMinutes();    int s = timeClient.getSeconds();    if (s != prevSecond) {      prevSecond = s;      tft.fillRect(10, 50, 220, 20, ST77XX\_BLACK);      tft.setTextColor(ST77XX\_YELLOW);      tft.setTextSize(3);      tft.setCursor(30, 50);      tft.printf("%02d:%02d:%02d", h, m, s);    }  }  void displayHeartRate() {    tft.fillRect(10, 100, 100, 30, ST77XX\_BLACK);    tft.setTextSize(2);    if (beatAvg > 0) {      tft.setTextColor(beatAvg > 100 ? ST77XX\_RED : ST77XX\_GREEN);      tft.setCursor(15, 105);      tft.printf("%d", beatAvg);      tft.setCursor(15, 120);      tft.print("BPM");    } else {      tft.setTextColor(ST77XX\_GRAY);      tft.setCursor(25, 105);      tft.print("--");      tft.setCursor(15, 120);      tft.print("BPM");    }  }  void displayTemperature(float temp) {    tft.fillRect(120, 100, 115, 40, ST77XX\_BLACK);    tft.setTextSize(2);    if (temp > -100) {      uint16\_t color = (temp > 37.5) ? ST77XX\_RED : (temp > 37.0) ? ST77XX\_ORANGE : ST77XX\_GREEN;      tft.setTextColor(color);      tft.setCursor(130, 105);      tft.printf("%.0f", temp);      tft.setCursor(170, 105);      tft.print("C");    } else {      tft.setTextColor(ST77XX\_GRAY);      tft.setCursor(140, 105);      tft.print("--");      tft.setCursor(170, 105);      tft.print("C");    }  }  void displaySteps() {    tft.fillRect(10, 160, 230, 40, ST77XX\_BLACK);    tft.setTextColor(ST77XX\_GREEN);    tft.setTextSize(4);    String stepsStr = String(stepCount);    int steps\_pixel = stepsStr.length() \* 6 \* 4;    int x = 60;    int y = 160;    tft.setCursor(x, y);    tft.print(stepsStr);    tft.setTextSize(3);    tft.setCursor(x + steps\_pixel + 4, y + 7); /    tft.print("buoc");  }  void displayStatus(String status, uint16\_t color = ST77XX\_WHITE) {    tft.fillRect(10, 220, 220, 50, ST77XX\_BLACK);    tft.setTextColor(color);    tft.setTextSize(1);    tft.setCursor(15, 230);    tft.print(status);  }  void setup() {    Serial.begin(115200);    Wire.begin(21, 22);    Wire.setClock(100000);    pinMode(TFT\_BL, OUTPUT);    digitalWrite(TFT\_BL, HIGH);    tft.init(240, 280);    tft.setRotation(0);    drawStaticUI();    displayStatus("Khoi tao cam bien...", ST77XX\_YELLOW);    WiFi.begin(ssid, password);    displayStatus("Ket noi WiFi...", ST77XX\_YELLOW);    int wifiTimeout = 0;    while (WiFi.status() != WL\_CONNECTED && wifiTimeout < 20) {      delay(500);      Serial.print(".");      wifiTimeout++;    }    if (WiFi.status() == WL\_CONNECTED) {      Serial.println("\nWiFi connected!");      timeClient.begin();      displayStatus("WiFi ket noi thanh cong", ST77XX\_GREEN);    } else {      displayStatus("Loi WiFi - che do offline", ST77XX\_RED);    }    delay(1000);    if (!mlx.begin()) {      Serial.println("MLX90614 not found!");      displayStatus("Loi: MLX90614 khong tim thay", ST77XX\_RED);      delay(2000);    } else {      Serial.println("MLX90614 initialized");    }    if (!pulseSensor.begin(Wire, I2C\_SPEED\_STANDARD)) {      Serial.println("MAX30105 not found!");      displayStatus("Loi: MAX30105 khong tim thay", ST77XX\_RED);      delay(2000);    } else {      Serial.println("MAX30105 initialized");      pulseSensor.setup();      pulseSensor.setPulseAmplitudeRed(0x1F);      pulseSensor.setPulseAmplitudeIR(0x1F);      pulseSensor.setPulseAmplitudeGreen(0);      pulseSensor.setup(60, 4, 2, 100, 411, 4096);    }    if (!mpu.begin()) {      Serial.println("MPU6050 not found!");      displayStatus("Loi: MPU6050 khong tim thay", ST77XX\_RED);      delay(2000);    } else {      Serial.println("MPU6050 initialized");      mpu.setAccelerometerRange(MPU6050\_RANGE\_8\_G);      mpu.setGyroRange(MPU6050\_RANGE\_500\_DEG);      mpu.setFilterBandwidth(MPU6050\_BAND\_21\_HZ);    }    displayStatus("Tat ca cam bien san sang!", ST77XX\_GREEN);    delay(2000);    displayStatus("Dat ngon tay len cam bien...", ST77XX\_CYAN);    Serial.println("Setup complete. Starting main loop...");  }  void loop() {    unsigned long currentTime = millis();    if (WiFi.status() == WL\_CONNECTED) {      displayTime();    }    if (currentTime - lastHeartRateCheck >= HEART\_RATE\_INTERVAL) {      lastHeartRateCheck = currentTime;      long irValue = pulseSensor.getIR();      if (irValue > 50000) {        if (checkForBeat(irValue)) {          long delta = currentTime - lastBeat;          lastBeat = currentTime;          beatsPerMinute = 60000.0 / delta;          if (beatsPerMinute >= 40 && beatsPerMinute <= 200) {            rates[rateSpot++] = (byte)beatsPerMinute;            rateSpot %= RATE\_SIZE;            long total = 0;            for (byte i = 0; i < RATE\_SIZE; i++) {              total += rates[i];            }            beatAvg = total / RATE\_SIZE;            if (beatAvg != prevBPM) {              prevBPM = beatAvg;              displayHeartRate();              displayStatus("Dang do nhip tim...", ST77XX\_GREEN);            }          }        }      } else {        if (prevBPM != 0) {          prevBPM = 0;          beatAvg = 0;          displayHeartRate();          displayStatus("Dat ngon tay len cam bien...", ST77XX\_CYAN);        }      }    }    if (currentTime - lastTempCheck >= TEMP\_INTERVAL) {      lastTempCheck = currentTime;      float temp = mlx.readObjectTempC();      if (abs(temp - prevTemp) > 0.1) {        prevTemp = temp;        displayTemperature(temp);        if (temp > 37.5) {          displayStatus("Canh bao: Sot cao!", ST77XX\_RED);        } else if (temp > 37.0) {          displayStatus("Nhiet do co the sot nhe", ST77XX\_ORANGE);        }      }    }    if (currentTime - lastStepCheck >= STEP\_INTERVAL) {      lastStepCheck = currentTime;      sensors\_event\_t a, g, temp;      mpu.getEvent(&a, &g, &temp);      float magnitude = sqrt(a.acceleration.x \* a.acceleration.x +                             a.acceleration.y \* a.acceleration.y +                             a.acceleration.z \* a.acceleration.z);      if (!stepFlag && abs(magnitude - lastMagnitude) > 3.0 &&          (currentTime - lastStepTime) > 400) {        stepCount++;        stepFlag = true;        lastStepTime = currentTime;        displaySteps();      } else if (stepFlag && abs(magnitude - lastMagnitude) < 1.5) {        stepFlag = false;      }      lastMagnitude = magnitude;    }    if (currentTime - lastDisplayUpdate >= DISPLAY\_UPDATE\_INTERVAL \* 20) {      lastDisplayUpdate = currentTime;      if (beatAvg > 0 && prevTemp > 30) {        String healthStatus = "Binh thuong";        uint16\_t statusColor = ST77XX\_GREEN;        if (beatAvg > 100 || prevTemp > 37.5) {          healthStatus = "Can theo doi";          statusColor = ST77XX\_ORANGE;        }        if (beatAvg > 120 || prevTemp > 38.0) {          healthStatus = "Can kham bac si!";          statusColor = ST77XX\_RED;        }        displayStatus(healthStatus, statusColor);      }    }    delay(1);  } |

Hình ảnh thực tế hiện tại:

A close up of a device

AI-generated content may be incorrect.