Tên Project: Hệ thống Giám sát và Thu thập Dữ liệu Môi trường qua LoRa

Mục tiêu: Xây dựng một hệ thống không dây để thu thập dữ liệu nhiệt độ, độ ẩm và phát hiện người qua lại từ các khu vực khác nhau (Node A0, A1) và tập trung về một điểm trung tâm (Node T0), sau đó truyền dữ liệu về máy tính (PC) để giám sát hoặc xử lý thêm.

Các Thành phần chính:

1. Node Thu thập dữ liệu (A0, A1):
   * Vi điều khiển: Arduino Nano
   * Module LoRa: RA02 (SX1278) + Anten
   * Cảm biến nhiệt độ và độ ẩm: DHT11 hoặc DHT22 (DHT22 cho độ chính xác cao hơn)
   * Cảm biến chuyển động: PIR HC-SR501
   * Nguồn: Pin hoặc adapter 5V
2. Node LoRa Trung tâm (T0):
   * Vi điều khiển: Arduino Nano
   * Module LoRa: RA02 (SX1278) + Anten
   * Kết nối với PC: Cáp USB (sử dụng cổng Serial)
   * Nguồn: Kết nối qua USB từ PC hoặc adapter 5V
3. Máy tính (PC):
   * Phần mềm giám sát: Một ứng dụng viết bằng Python hoặc ngôn ngữ khác để đọc dữ liệu từ cổng Serial của T0 và hiển thị, lưu trữ, hoặc xử lý.

Sơ đồ Hoạt động:

1. Các Node A0, A1:
   * Đọc dữ liệu từ cảm biến DHT (nhiệt độ, độ ẩm) và cảm biến PIR (trạng thái chuyển động).
   * Đóng gói dữ liệu thu thập được thành một gói tin (packet) có cấu trúc rõ ràng (ví dụ: bao gồm ID Node, nhiệt độ, độ ẩm, trạng thái PIR).
   * Sử dụng module LoRa RA02 để gửi gói tin này đến Node T0. Các node A0 và A1 sẽ hoạt động độc lập và gửi dữ liệu định kỳ hoặc khi có sự kiện (ví dụ: phát hiện chuyển động).
2. Node Trung tâm T0:
   * Sử dụng module LoRa RA02 để lắng nghe và nhận các gói tin từ các Node A0 và A1.
   * Giải mã gói tin nhận được để trích xuất dữ liệu từ mỗi node.
   * Định dạng lại dữ liệu (ví dụ: chuỗi văn bản) để dễ dàng gửi qua cổng Serial.
   * Truyền dữ liệu đã xử lý đến PC thông qua kết nối USB-Serial tích hợp trên Arduino Nano.
3. Máy tính (PC):
   * Chạy phần mềm giám sát.
   * Kết nối với cổng Serial mà Arduino Nano (T0) đang sử dụng.
   * Đọc dữ liệu được gửi từ T0 qua cổng Serial.
   * Hiển thị dữ liệu trên giao diện người dùng, lưu vào file, cơ sở dữ liệu, hoặc thực hiện các phân tích khác tùy theo yêu cầu.

Cấu trúc Gói tin LoRa (ví dụ):

Một gói tin có thể là một chuỗi ký tự được phân tách bằng dấu phẩy hoặc ký tự đặc biệt, ví dụ:

A0,25.5,60.2,1 (Node A0, Nhiệt độ 25.5°C, Độ ẩm 60.2%, Có chuyển động)

A1,24.8,62.1,0 (Node A1, Nhiệt độ 24.8°C, Độ ẩm 62.1%, Không có chuyển động)

Các Bước Triển khai:

1. Kết nối phần cứng:
   * Nối module LoRa RA02 với Arduino Nano cho cả 3 node (A0, A1, T0). Cần chú ý đến các chân SPI (SCK, MISO, MOSI, SS) và các chân DIO0, RST. Thường sử dụng thư viện LoRa của Sandeep Mistry, các chân kết nối phổ biến là:
     + LoRa DIO0 -> Arduino D2 (hoặc chân ngắt khác)
     + LoRa RST -> Arduino D9
     + LoRa NSS -> Arduino D10
     + LoRa SCK -> Arduino D13
     + LoRa MISO -> Arduino D12
     + LoRa MOSI -> Arduino D11
     + LoRa VCC -> Arduino 3.3V (Kiểm tra kỹ module RA02 của bạn, một số có thể hoạt động với 5V nhưng 3.3V là an toàn hơn và đúng chuẩn LoRa)
     + LoRa GND -> Arduino GND
   * Nối cảm biến DHT11/DHT22 với Arduino Nano trên các Node A0, A1. Cần chân VCC (5V), GND, và chân Data (thường nối vào một chân Digital có hỗ trợ pull-up hoặc sử dụng điện trở pull-up ngoài nếu cần, ví dụ D3).
   * Nối cảm biến PIR HC-SR501 với Arduino Nano trên các Node A0, A1. Cần chân VCC (5V), GND, và chân Output (nối vào một chân Digital, ví dụ D4).
2. Lập trình Arduino:
   * Cài đặt thư viện: Cài đặt thư viện LoRa của Sandeep Mistry và thư viện DHT sensor library của Adafruit (cần cả Adafruit Unified Sensor nữa).
   * Code cho Node A0, A1:
     + Khởi tạo LoRa với tần số phù hợp (ví dụ: 433MHz).
     + Khởi tạo cảm biến DHT.
     + Cấu hình chân PIR là INPUT.
     + Trong vòng lặp loop(), đọc dữ liệu từ DHT và PIR.
     + Đóng gói dữ liệu thành chuỗi.
     + Sử dụng LoRa.beginPacket(), LoRa.print(), LoRa.endPacket() để gửi gói tin.
     + Thêm độ trễ giữa các lần gửi để tiết kiệm năng lượng và tránh gây nhiễu. Có thể gửi theo lịch trình hoặc khi có sự thay đổi trạng thái PIR.
   * Code cho Node T0:
     + Khởi tạo LoRa với cùng tần số.
     + Khởi tạo Serial communication với PC (Serial.begin(tốc độ\_baud)).
     + Trong vòng lặp loop(), sử dụng LoRa.parsePacket() để kiểm tra xem có gói tin LoRa nào đến không.
     + Nếu có gói tin, đọc dữ liệu bằng LoRa.read().
     + Giải mã chuỗi dữ liệu nhận được.
     + Định dạng lại và gửi dữ liệu qua Serial đến PC bằng Serial.print() hoặc Serial.println().
3. Phát triển phần mềm PC:
   * Sử dụng ngôn ngữ như Python với thư viện pyserial.
   * Kết nối đến cổng Serial tương ứng với Arduino Nano (T0).
   * Đọc dữ liệu từ cổng Serial.
   * Phân tích chuỗi dữ liệu nhận được.
   * Hiển thị dữ liệu (console, giao diện đồ họa) hoặc lưu trữ.

Những điểm cần lưu ý và mở rộng:

* Tần số LoRa: Chọn tần số phù hợp với quy định của quốc gia bạn (ví dụ: 433MHz, 868MHz, 915MHz).
* Anten: Sử dụng anten phù hợp và chất lượng tốt để đạt được khoảng cách truyền tối ưu.
* Quản lý năng lượng: Đối với các node A0, A1 chạy bằng pin, cần tối ưu hóa việc sử dụng năng lượng (ví dụ: đưa Arduino vào chế độ ngủ sâu và chỉ thức dậy để đọc cảm biến và gửi dữ liệu).
* Giao thức truyền tin: Với 3 node, bạn có thể sử dụng giao tiếp peer-to-peer đơn giản. Với nhiều node hơn, cân nhắc triển khai một giao thức đơn giản hơn LoRaWAN, ví dụ: mỗi node có một ID duy nhất, và gói tin chứa ID của node gửi. Node T0 sẽ xử lý dữ liệu dựa trên ID này.
* Xử lý va chạm (Collision): Nếu nhiều node gửi dữ liệu cùng lúc, có thể xảy ra va chạm. Với một số ít node và tần suất gửi thấp, vấn đề này có thể không nghiêm trọng. Với hệ thống lớn hơn, cần các kỹ thuật truy cập kênh (ví dụ: CSMA/CA đơn giản hoặc triển khai LoRaWAN Class A).
* Phản hồi từ T0 (tùy chọn): Node T0 có thể gửi gói tin phản hồi về cho các node A0, A1 để xác nhận đã nhận được dữ liệu.
* Giao diện PC: Có thể phát triển một giao diện đồ họa đơn giản trên PC sử dụng thư viện như Tkinter, PyQt (Python) để hiển thị dữ liệu trực quan hơn.
* Lưu trữ dữ liệu: Lưu trữ dữ liệu vào file CSV hoặc cơ sở dữ liệu (SQLite đơn giản, MySQL, PostgreSQL) trên PC.
* Đồng bộ thời gian (tùy chọn): Nếu cần dữ liệu có dấu thời gian chính xác, các node có thể đồng bộ thời gian với T0.