Shape, rectangle

Description automatically generated`

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**



**Giảng viên hướng dẫn: Chung Quang Khánh**

**Sinh viên thực hiện: nhóm 9 và nhóm 10**

Trương Hữu Khang 20520211

Nguyễn Linh Anh Khoa 20520219

Hà Vĩnh Kiện 20520597

Phan Duy Thông 20520789

Trương Trọng Hiếu 20520185

Lê Hoàng Thịnh 20521959

**ĐỒ ÁN THIẾT KẾ HỆ THỐNG NHÚNG**

**ĐỀ TÀI: HỆ THỐNG KIỂM SOÁT TRONG NÔNG NGHIỆP**

**LỚP CE224.N11.2**

**TP. Hồ Chí Minh, 2023**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**TP. Hồ Chí Minh, tháng 4 năm 2022**

**Bảng phân công công việc**

|  |  |  |
| --- | --- | --- |
| **MSSV** | **Họ và tên** | **Phần trăm thực hiện** |
| 20520211 | Trương Hữu Khang | 32% |
| 20520219 | Nguyễn Linh Anh Khoa | 24% |
| 20520597 | Hà Vĩnh Kiện | 24% |
| 20520789 | Phan Duy Thông | 20% |
| 20520185 | Trương Trọng Hiếu | 0% |
| 20521959 | Lê Hoàng Thịnh | 0% |

**MỤC LỤC**

[Chương 1. Tổng quan về hệ thống 5](#_Toc122954998)

[1. Giới thiệu đề tài 5](#_Toc122954999)

[2. Lý do chọn đề tài 5](#_Toc122955000)

[3. Các giao thức sử dụng 5](#_Toc122955001)

* [Giao thức I2C (Inter – Integrated Circuit) 5](#_Toc122955002)
* [Giao thức SPI (Serial Peripheral Interface) 10](#_Toc122955007)
* [Giao thức UART (Universal Asynchronous Receiver / Transmitter) 14](#_Toc122955011)

[4. Mô tả công nghệ 18](#_Toc122955014)

[5. Kết quả mong muốn 18](#_Toc122955015)

[6. Mục tiêu / Hạn chế 18](#_Toc122955016)

[Chương 2. Đề tài 19](#_Toc122955017)

[1. Cấu trúc thiết bị 19](#_Toc122955018)

[2. Đặc tính kỹ thuật 19](#_Toc122955019)

[3. Lưu đồ hoạt động 19](#_Toc122955020)

[4. Bảng mô tả 21](#_Toc122955021)

[5. Video demo 21](#_Toc122955022)

**Danh Mục Hình**

[Hình 1. Kiến trúc I2C 6](#_Toc122953884)

[Hình 2. Data frame I2C 6](#_Toc122953885)

[Hình 3. Một master với nhiều slave I2C 9](#_Toc122953886)

[Hình 4. Nhiều master với nhiều slave I2C 10](#_Toc122953887)

[Hình 5. Kiến trúc SPI 11](#_Toc122953889)

[Hình 6. Một master nhiều chân slave select với nhiều slave SPI 12](#_Toc122953890)

[Hình 7. Một master một chân slave select với nhiều slave SPI 13](#_Toc122953891)

[Hình 8. Kiến trúc UART 14](#_Toc122953893)

[Hình 9 Data frame UART 15](#_Toc122953894)

**Chương 1. Tổng quan về hệ thống**

1. **Giới thiệu đề tài**

Trong quá trình phát triển của con người, nông nghiệp đóng một vai trò rất quan trọng trong việc cung cấp lương thực, thực phẩm để xã hội phát triển. Với sự phát triển của công nghệ, năng suất sản xuất của ngành nông nghiệp ngày càng tăng, đồng thời cũng xuất hiện nhiều vấn đề cần được giải quyết. Việc kiểm soát các yếu tố của môi trường cũng ngày càng được chú trọng hơn trong nông nghiệp. Nên từ đó các nghiên cứu về việc kiểm soát môi trường trong nông nghiệp cũng xuất hiện nhiều hơn.

1. **Lý do chọn đề tài**

Nông nghiệp xanh trong lĩnh vực trồng trọt thường đòi hỏi rất khắt khe việc duy trì nhiệt độ, độ ẩm và cường độ ánh sáng cho môi trường sinh trưởng cây trồng theo từng loại cây cũng như thời gian sinh trưởng của chúng. Điều này đòi hỏi cần phải giám sát và thu thập số liệu về nhiệt độ,độ ẩm và cường độ ánh sáng tại nhiều vị trí khác nhau trong môi trường sinh trưởng cây trồng.

Biết được những thông số cần thiết sẽ cho phép nhà nông điều chỉnh hoạt động canh tác của mình một cách thích hợp để đạt tối ưu năng suất cây trồng. Như vậy, có thể thấy việc đo lường nhiệt độ - độ ẩm – cường độ ánh sáng là rất cần thiết và quan trọng mật thiết.

Với mục đích trên nhóm chúng em chọn đề tài “HỆ THỐNG KIỂM SOÁT TRONG NÔNG NGHIỆP” để hiện thực hóa ý tưởng về một hệ thống có thể đo được các thông số nhiệt độ - độ ẩm – cường độ ánh sáng với độ chính xác cao có thể áp dụng vào thực tế.

1. **Các giao thức sử dụng**

**Giao thức I2C (Inter – Integrated Circuit)**

I2C kết hợp các tính năng tốt nhất của SPI và UART. Với I2C, có thể kết nối nhiều slave với một master duy nhất (như SPI) và có thể có nhiều master điều khiển một hoặc nhiều slave. Thực sự hữu ích khi muốn có nhiều hơn một vi điều khiển ghi dữ liệu vào một thẻ nhớ duy nhất hoặc hiển thị văn bản trên một màn hình LCD.

Diagram

Description automatically generated

*Hình 1. Kiến trúc I2C*

Giống như giao tiếp UART, I2C chỉ sử dụng hai dây để truyền dữ liệu giữa các thiết bị:

* SDA (Serial Data) - đường truyền cho master và slave để gửi và nhận dữ liệu.
* SCL (Serial Clock) - đường mang tín hiệu xung nhịp.

I2C là một giao thức truyền thông nối tiếp, vì vậy dữ liệu được truyền từng bit dọc theo một đường duy nhất (đường SDA).

Giống như SPI, I2C là đồng bộ, do đó đầu ra của các bit được đồng bộ hóa với việc lấy mẫu các bit bởi một tín hiệu xung nhịp được chia sẻ giữa master và slave. Tín hiệu xung nhịp luôn được điều khiển bởi master.

**Cách hoạt động của I2C**

Với I2C, dữ liệu được truyền trong các tin nhắn. Tin nhắn được chia thành các khung dữ liệu. Mỗi tin nhắn có một khung địa chỉ chứa địa chỉ nhị phân của địa chỉ slave và một hoặc nhiều khung dữ liệu chứa dữ liệu đang được truyền. Thông điệp cũng bao gồm điều kiện khởi động và điều kiện dừng, các bit đọc / ghi và các bit ACK / NACK giữa mỗi khung dữ liệu:

Table

Description automatically generated

*Hình 2. Data frame I2C*

* Điều kiện khởi động: Đường SDA chuyển từ mức điện áp cao xuống mức điện áp thấp trước khi đường SCL chuyển từ mức cao xuống mức thấp.
* Điều kiện dừng: Đường SDA chuyển từ mức điện áp thấp sang mức điện áp cao sau khi đường SCL chuyển từ mức thấp lên mức cao.
* Khung địa chỉ: Một chuỗi 7 hoặc 10 bit duy nhất cho mỗi slave để xác định slave khi master muốn giao tiếp với nó.
* Bit Đọc / Ghi: Một bit duy nhất chỉ định master đang gửi dữ liệu đến slave (mức điện áp thấp) hay yêu cầu dữ liệu từ nó (mức điện áp cao).
* Bit ACK / NACK: Mỗi khung trong một tin nhắn được theo sau bởi một bit xác nhận / không xác nhận. Nếu một khung địa chỉ hoặc khung dữ liệu được nhận thành công, một bit ACK sẽ được trả lại cho thiết bị gửi từ thiết bị nhận.

**Địa chỉ**

I2C không có các đường Slave Select như SPI, vì vậy cần một cách khác để cho slave biết rằng dữ liệu đang được gửi đến slave này chứ không phải slave khác. Nó thực hiện điều này bằng cách định địa chỉ. Khung địa chỉ luôn là khung đầu tiên sau bit khởi động trong một tin nhắn mới.

Master gửi địa chỉ của slave mà nó muốn giao tiếp với mọi slave được kết nối với nó. Sau đó, mỗi slave sẽ so sánh địa chỉ được gửi từ master với địa chỉ của chính nó. Nếu địa chỉ phù hợp, nó sẽ gửi lại một bit ACK điện áp thấp cho master. Nếu địa chỉ không khớp, slave không làm gì cả và đường SDA vẫn ở mức cao.

**Bit đọc / ghi**

Khung địa chỉ bao gồm một bit duy nhất ở cuối tin nhắn cho slave biết master muốn ghi dữ liệu vào nó hay nhận dữ liệu từ nó. Nếu master muốn gửi dữ liệu đến slave, bit đọc / ghi ở mức điện áp thấp. Nếu master đang yêu cầu dữ liệu từ slave, thì bit ở mức điện áp cao.

**Khung dữ liệu**

Sau khi master phát hiện bit ACK từ slave, khung dữ liệu đầu tiên đã sẵn sàng được gửi.

Khung dữ liệu luôn có độ dài 8 bit và được gửi với bit quan trọng nhất trước. Mỗi khung dữ liệu ngay sau đó là một bit ACK / NACK để xác minh rằng khung đã được nhận thành công. Bit ACK phải được nhận bởi master hoặc slave (tùy thuộc vào cái nào đang gửi dữ liệu) trước khi khung dữ liệu tiếp theo có thể được gửi.

Sau khi tất cả các khung dữ liệu đã được gửi, master có thể gửi một điều kiện dừng cho slave để tạm dừng quá trình truyền. Điều kiện dừng là sự chuyển đổi điện áp từ thấp lên cao trên đường SDA sau khi chuyển tiếp từ thấp lên cao trên đường SCL , với đường SCL vẫn ở mức cao.

**Các bước truyền dữ liệu I2C**

Master gửi điều kiện khởi động đến mọi slave được kết nối bằng cách chuyển đường SDA từ mức điện áp cao sang mức điện áp thấp trước khi chuyển đường SCL từ mức cao xuống mức thấp.

Master gửi cho mỗi slave địa chỉ 7 hoặc 10 bit của slave mà nó muốn giao tiếp, cùng với bit đọc / ghi.

Mỗi slave sẽ so sánh địa chỉ được gửi từ master với địa chỉ của chính nó. Nếu địa chỉ trùng khớp, slave sẽ trả về một bit ACK bằng cách kéo dòng SDA xuống thấp cho một bit. Nếu địa chỉ từ master không khớp với địa chỉ của slave, slave rời khỏi đường SDA cao.

Master gửi hoặc nhận khung dữ liệu.

Sau khi mỗi khung dữ liệu được chuyển, thiết bị nhận trả về một bit ACK khác cho thiết bị gửi để xác nhận đã nhận thành công khung.

Để dừng truyền dữ liệu, master gửi điều kiện dừng đến slave bằng cách chuyển đổi mức cao SCL trước khi chuyển mức cao SDA.

**Một master với nhiều slave**

Vì I2C sử dụng định địa chỉ nên nhiều slave có thể được điều khiển từ một master duy nhất. Với địa chỉ 7 bit sẽ có 128 (27) địa chỉ duy nhất. Việc sử dụng địa chỉ 10 bit không phổ biến, nhưng nó cung cấp 1.024 (210) địa chỉ duy nhất. Để kết nối nhiều slave đến một master duy nhất, bạn có thể đấu dây như thế này, với điện trở kéo lên 4,7K Ohm kết nối đường SDA và SCL với Vcc:

Diagram, schematic

Description automatically generated

*Hình 3. Một master với nhiều slave I2C*

**Nhiều master với nhiều slave**

Nhiều master có thể được kết nối với một slave hoặc nhiều slave. Sự cố với nhiều master trong cùng một hệ thống xảy ra khi hai master cố gắng gửi hoặc nhận dữ liệu cùng một lúc qua đường SDA. Để giải quyết vấn đề này, mỗi master cần phải phát hiện xem đường SDA thấp hay cao trước khi truyền tin nhắn. Nếu đường SDA thấp, điều này có nghĩa là một master khác có quyền điều khiển bus và master đó phải đợi để gửi tin nhắn. Nếu đường SDA cao thì có thể truyền tin nhắn an toàn. Để kết nối nhiều master với nhiều slave, hãy sử dụng sơ đồ sau, với các điện trở kéo lên 4,7K Ohm kết nối các đường SDA và SCL với Vcc:

Diagram, schematic

Description automatically generated

*Hình 4. Nhiều master với nhiều slave I2C*

**Giao thức SPI (Serial Peripheral Interface)**

SPI là một giao thức giao tiếp phổ biến được sử dụng bởi nhiều thiết bị khác nhau. Ví dụ, module thẻ SD, module đầu đọc thẻ RFID và bộ phát / thu không dây 2,4 GHz đều sử dụng SPI để giao tiếp với vi điều khiển.

Lợi ích duy nhất của SPI là dữ liệu có thể được truyền mà không bị gián đoạn. Bất kỳ số lượng bit nào cũng có thể được gửi hoặc nhận trong một luồng liên tục. Với I2C và UART, dữ liệu được gửi dưới dạng gói, giới hạn ở một số bit cụ thể. Điều kiện bắt đầu và dừng xác định điểm bắt đầu và kết thúc của mỗi gói, do đó dữ liệu bị gián đoạn trong quá trình truyền.

Các thiết bị giao tiếp qua SPI có quan hệ master - slave. Master là thiết bị điều khiển (thường là vi điều khiển), còn slave (thường là cảm biến, màn hình hoặc chip nhớ) nhận lệnh từ master. Cấu hình đơn giản nhất của SPI là hệ thống một slave, một master duy nhất, nhưng một master có thể điều khiển nhiều hơn một slave.

Diagram

Description automatically generated

*Hình 5. Kiến trúc SPI*

MOSI (đầu ra master / đầu vào slave) - đường truyền cho master gửi dữ liệu đến slave.

MISO (đầu vào master / đầu ra slave) - đường cho slave gửi dữ liệu đến master.

SCLK (clock) - đường cho tín hiệu xung nhịp.

SS / CS (Slave Select / Chip Select) - đường cho master chọn slave nào để gởi tín hiệu.

**Cách hoạt động cuả SPI**

*Xung nhịp*

Tín hiệu xung nhịp đồng bộ hóa đầu ra của các bit dữ liệu từ master để lấy mẫu các bit của slave. Một bit dữ liệu được truyền trong mỗi chu kỳ xung nhịp, do đó tốc độ truyền dữ liệu được xác định bởi tần số của tín hiệu xung nhịp. Giao tiếp SPI được khởi tạo bởi master kể từ khi master cấu hình và tạo ra tín hiệu xung nhịp.

Bất kỳ giao thức giao tiếp nào mà các thiết bị chia sẻ tín hiệu xung nhịp thì đều được gọi là đồng bộ. SPI là một giao thức giao tiếp đồng bộ. Ngoài ra còn có các phương thức không đồng bộ không sử dụng tín hiệu xung nhịp. Ví dụ, trong giao tiếp UART, cả hai bên đều được đặt thành tốc độ truyền được cấu hình sẵn để chỉ ra tốc độ và thời gian truyền dữ liệu.

Tín hiệu xung nhịp trong SPI có thể được sửa bằng cách sử dụng các thuộc tính của phân cực xung nhịp và pha xung nhịp. Hai thuộc tính này làm việc cùng nhau để xác định khi nào các bit được xuất ra và khi được lấy mẫu. Phân cực xung nhịp có thể được thiết lập bởi master để cho phép các bit được xuất ra và lấy mẫu trên cạnh lên hoặc xuống của chu kỳ xung nhịp. Pha xung nhịp có thể được đặt để đầu ra và lấy mẫu xảy ra trên cạnh đầu tiên hoặc cạnh thứ hai của chu kỳ xung nhịp, bất kể nó đang tăng hay giảm.

*Slave Select*

Master có thể chọn slave mà nó muốn giao tiếp bằng cách đặt đường CS / SS của slave ở mức điện áp thấp. Ở trạng thái idle, không truyền tải, dòng slave select được giữ ở mức điện áp cao. Nhiều chân CS / SS có thể có sẵn trên thiết bị master cho phép đấu dây song song nhiều slave. Nếu chỉ có một chân CS / SS, nhiều slave có thể được kết nối với master bằng cách nối chuỗi.

*Nhiều slave*

SPI có thể thiết lập để hoạt động với một master và một slave duy nhất, và nó có thể được thiết lập với nhiều slave do một master duy nhất điều khiển. Có hai cách để kết nối nhiều slave với master. Nếu master có nhiều chân slave select, các slave có thể được nối dây song song như thế này:

Diagram, schematic

Description automatically generated

#### Hình 6. Một master nhiều chân slave select với nhiều slave SPI

Nếu chỉ có một chân slave select, các slave có thể được nối chuỗi như sau:

Diagram, schematic

Description automatically generated

*Hình 7. Một master một chân slave select với nhiều slave SPI*

*MOSI và MISO*

Master gửi dữ liệu đến slave từng bit, nối tiếp qua đường MOSI. Slave nhận dữ liệu được gửi từ master tại chân MOSI. Dữ liệu được gửi từ master đến slave thường được gửi với bit quan trọng nhất trước.

Slave cũng có thể gửi dữ liệu trở lại master thông qua đường MISO nối tiếp. Dữ liệu được gửi từ slave trở lại master thường được gửi với bit ít quan trọng nhất trước.

**Các bước truyền dữ liệu SPI**

1. Master ra tín hiệu xung nhịp.
2. Master chuyển chân SS / CS sang trạng thái điện áp thấp, điều này sẽ kích hoạt slave.
3. Master gửi dữ liệu từng bit một tới slave dọc theo đường MOSI. Slave đọc các bit khi nó nhận được.
4. Nếu cần phản hồi, slave sẽ trả lại dữ liệu từng bit một cho master dọc theo đường MISO. Master đọc các bit khi nó nhận được.

**Giao thức UART (Universal Asynchronous Receiver / Transmitter)**

UART hay bộ thu-phát không đồng bộ đa năng là một trong những hình thức giao tiếp kỹ thuật số giữa thiết bị với thiết bị đơn giản và lâu đời nhất. Bạn có thể tìm thấy các thiết bị UART trong một phần của mạch tích hợp (IC) hoặc dưới dạng các thành phần riêng lẻ. Các UART giao tiếp giữa hai nút riêng biệt bằng cách sử dụng một cặp dẫn và một nối đất chung.

Diagram

Description automatically generated  
*Hình 8. Kiến trúc UART*

**Hướng dẫn giao tiếp UART**

Vì nó là thiết lập phổ quát nên chúng ta có thể định cấu hình UART để hoạt động với nhiều loại giao thức nối tiếp khác nhau. UART đã được điều chỉnh thành các đơn vị chip đơn vào đầu những năm 1970, bắt đầu với Western Digital’s WD1402A.

*Trong một sơ đồ giao tiếp UART:*

Chân Tx (truyền) của một chip kết nối trực tiếp với chân Rx (nhận) của chip kia và ngược lại. Thông thường, quá trình truyền sẽ diễn ra ở 3.3V hoặc 5V. UART là một giao thức một master, một slave, trong đó một thiết bị được thiết lập để giao tiếp với duy nhất một thiết bị khác.

Dữ liệu truyền đến và đi từ UART song song với thiết bị điều khiển (ví dụ: CPU).

Khi gửi trên chân Tx, UART đầu tiên sẽ dịch thông tin song song này thành nối tiếp và truyền đến thiết bị nhận.

UART thứ hai nhận dữ liệu này trên chân Rx của nó và biến đổi nó trở lại thành song song để giao tiếp với thiết bị điều khiển của nó.

UART truyền dữ liệu nối tiếp, theo một trong ba chế độ:

* Full duplex: Giao tiếp đồng thời đến và đi từ mỗi master và slave
* Half duplex: Dữ liệu đi theo một hướng tại một thời điểm
* Simplex: Chỉ giao tiếp một chiều

Dữ liệu truyền qua UART được tổ chức thành các gói. Mỗi gói chứa 1 bit bắt đầu, 5 đến 9 bit dữ liệu (tùy thuộc vào UART), một bit chẵn lẻ tùy chọn và 1 hoặc 2 bit dừng.

Diagram

Description automatically generated

*Hình 9 Data frame UART*

*Bit bắt đầu*

Đường truyền dữ liệu UART thường được giữ ở mức điện áp cao khi không truyền dữ liệu. Để bắt đầu truyền dữ liệu, UART truyền sẽ kéo đường truyền từ mức cao xuống mức thấp trong một chu kỳ clock. Khi UART nhận phát hiện sự chuyển đổi điện áp cao xuống thấp, nó bắt đầu đọc các bit trong khung dữ liệu ở tần số của tốc độ truyền.

*Khung dữ liệu*

Khung dữ liệu chứa dữ liệu thực tế được chuyển. Nó có thể dài từ 5 bit đến 8 bit nếu sử dụng bit chẵn lẻ. Nếu không sử dụng bit chẵn lẻ, khung dữ liệu có thể dài 9 bit. Trong hầu hết các trường hợp, dữ liệu được gửi với bit ít quan trọng nhất trước tiên.

*Bit chẵn lẻ*

Bit chẵn lẻ là một cách để UART nhận cho biết liệu có bất kỳ dữ liệu nào đã thay đổi trong quá trình truyền hay không. Bit có thể bị thay đổi bởi bức xạ điện từ, tốc độ truyền không khớp hoặc truyền dữ liệu khoảng cách xa. Sau khi UART nhận đọc khung dữ liệu, nó sẽ đếm số bit có giá trị là 1 và kiểm tra xem tổng số là số chẵn hay lẻ. Nếu bit chẵn lẻ là 0 (tính chẵn), thì tổng các bit 1 trong khung dữ liệu phải là một số chẵn. Nếu bit chẵn lẻ là 1 (tính lẻ), các bit 1 trong khung dữ liệu sẽ tổng thành một số lẻ. Khi bit chẵn lẻ khớp với dữ liệu, UART sẽ biết rằng quá trình truyền không có lỗi. Nhưng nếu bit chẵn lẻ là 0 và tổng là số lẻ; hoặc bit chẵn lẻ là 1 và tổng số là chẵn, UART sẽ biết rằng các bit trong khung dữ liệu đã thay đổi.

*Bit dừng*

Để báo hiệu sự kết thúc của gói dữ liệu, UART gửi sẽ điều khiển đường truyền dữ liệu từ điện áp thấp đến điện áp cao trong ít nhất khoảng 2 bit.

*Có thể tóm tắt lại như sau.*

Quá trình truyền dữ liệu diễn ra dưới dạng các gói dữ liệu, bắt đầu bằng một bit bắt đầu, đường mức cao được kéo xuống đất.

Sau bit bắt đầu, năm đến chín bit dữ liệu truyền trong khung dữ liệu của gói, theo sau là bit chẵn lẻ tùy chọn để xác minh việc truyền dữ liệu thích hợp.

Cuối cùng, một hoặc nhiều bit dừng được truyền ở nơi đường đặt ở mức cao.

Như vậy là kết thúc một gói.

UART là giao thức không đồng bộ, do đó không có đường clock nào điều chỉnh tốc độ truyền dữ liệu. Người dùng phải đặt cả hai thiết bị để giao tiếp ở cùng tốc độ. Tốc độ này được gọi là tốc độ truyền, được biểu thị bằng bit trên giây hoặc bps. Tốc độ truyền thay đổi đáng kể, từ 9600 baud đến 115200 và hơn nữa. Tốc độ truyền giữa UART truyền và nhận chỉ có thể chênh lệch khoảng 10% trước khi thời gian của các bit bị lệch quá xa.

Mặc dù UART là giao thức cũ và chỉ có thể giao tiếp giữa một master và slave duy nhất, nhưng nó dễ thiết lập và cực kỳ linh hoạt. Do đó, bạn có thể gặp nó khi làm việc với các dự án vi điều khiển. UART có thể là một phần của hệ thống mà bạn sử dụng hàng ngày, mà có thể bạn không nhận ra.

**Bảng tóm tắt đặc tính của ba loại giao thức:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Đặc điểm** | **I2C** | **SPI** | **UART** |
| Sơ đồ kết nối | Kết nối khá đơn giảnDiagram, schematic  Description automatically generated | Kết nối khá phức tạpDiagram, schematic  Description automatically generated | Kết nối đơn giản  Diagram, schematic  Description automatically generated |
| Số lượng dây | 2dây | 4dây  (Số lượng dây tăng khi số thiết bị tăng) | 2 dây  (1 dây truyền & 1dây  nhận dữ liệu) |
| Chế độ truyền | Half duplex (Nhiều master & nhiều slave) | Full duplex  (Một master & nhiều slave) | Full duplex  (Không phân biệt  master – slave) |
| Tốc độ truyền | Truyền đồng bộ  (Hỗ trợ tốc độ 100kbps, 400kbps,3.4Mbps, 1Mbps) | Truyền đồng bộ  (Tốc độ khoảng 10Mbps đến 20Mbps) | Truyền không đồng bộ  (Tốc độ tự đặt, tối đa khoảng 460kbps) |
| Khoảng cách | 1 – 10m | Khoảng 10m | 12m trên lý thuyết |
| Số thiết bị | Lên đến 127 thiết bị Giao tiếp bằng địa chỉ | Số lượng hạn chế  Giao tiếp bằng chân chọn chip | 2 thiết bị  Giao tiếp 1 - 1 |

1. **Mô tả công nghệ**

Hệ thống có khả năng:

* + - Đo nhiệt độ của không khí, độ ẩm đất và cường độ ánh sáng, nồng độ các chất trong không khí..
    - Hiển thị các thông số nhiệt độ, độ ẩm, cường độ ánh sáng, thành phần các chất trong không khí lên màn hình OLED.

1. **Kết quả mong muốn**
   * + Biết được cách lập trình cho vi điều khiển, các thiết bị ngoại vi (sensor) và các linh kiện khác trong mạch.
     + Tạo ra một mô hình đo các thông số nhiệt độ - độ ẩm – cường độ ánh sáng có thể hoạt động tốt, từ đó có thể chế tạo được cửa tự động phục vụ thực tế.
     + Đọc và xử lý tín hiệu từ các sensor.
     + Biết được cách lập trình đa luồng, sử dụng Real-time OS và cách định thời các Task
2. **Mục tiêu / Hạn chế**

Mục tiêu:

* + - Hệ thống hoạt động được với thời gian thực.
    - Hoạt động đúng với thiết kế, ít bị xảy ra lỗi.

Hạn chế:

* + - Chưa có khả năng tự sửa lỗi

**Chương 2. Đề tài**

1. **Cấu trúc thiết bị**

|  |  |
| --- | --- |
| **Thiết bị** | **Loại sử dụng** |
| Bộ điều khiển | Arduino Nano 33 BLE |
| Hiển thị thông tin | Màn hình OLED SSD1306 0.96-inch |
| Đo độ ẩm đất | Cảm biến độ ẩm đất v1.2 |
| Đo cường độ ánh sáng | Cảm biến cường độ ánh sáng GY-30 BH1750 |
| Đo độ ẩm và nhiệt độ môi trường xung quanh | Mạch cảm biến nhiệt độ và độ ẩm không khí AHT20 |
| Đo nồng độ các chất trong không khí | MQ3 |

1. **Đặc tính kỹ thuật**

|  |  |
| --- | --- |
| **Thông số** | **Giá trị** |
| Giới hạn đo nhiệt độ | -40 ° C ~ + 80 ° C |
| Sai số đo nhiệt độ | 2% với ° C |
| Giới hạn đo độ ẩm | 0% ~ 100% RH |
| Sai số đo độ ẩm | 2% với RH |
| Giới hạn đo cường độ ánh sáng | 0 – 65535 lux |
| Sai số đo cường độ ánh sáng | 4% với lux |
| Giới hạn đo nồng độ các khí | 0.04 – 4 mg/L |

1. **Lưu đồ hoạt động**

Diagram

Description automatically generated

Chart, box and whisker chart

Description automatically generatedHình 10. High Level Block Diagram

Hình 11. Data Flow Diagram

1. **Bảng mô tả**

|  |  |  |
| --- | --- | --- |
| **STT** | **Đặc điểm** | **Miêu tả** |
| 1 | Đặc điểm sản phẩm | Tiến trình: đo các thông số, tín hiệu xử lý, hiển thị trên OLED |
| Chế độ: thủ công |
| Hạn chế: phạm vi nhỏ |
| 2 | Đặc điểm kỹ thuật | Đầu vào: Đất, không khí |
| Đầu ra: màn hình OLED |
| 3 | Đặc điểm kỹ thuật phần cứng | VĐK: Arduino Nano 33 BLE |
| Cảm biến: GY-30 BH1750VI, AHT20, MQ3 |
| Thiết bị xuất: màn hình OLED |
| 4 | Đặc điểm kỹ thuật phần mềm | Functions: đọc tín hiệu cảm biến, giải mã tín hiệu, xuất dữ liệu lên màn hình hiển thị. |
| Control algorithm: sử dụng biến cờ để phát tín hiệu việc hoàn thành các quy trình để hiện thi các chức năng |
| 5 | Đặc điểm kiểm tra kỹ thuật | Platform: Platformio, Arduino IDE |
| Quá trình kiểm thử: Kiểm tra thành phần, dữ liệu, kiểm tra hiển thị |

1. **Video demo**

[**https://drive.google.com/file/d/14qqzgrvZ7ue\_lG8LSLMi7wTnYfE44sfV/view?usp=share\_link**](https://drive.google.com/file/d/14qqzgrvZ7ue_lG8LSLMi7wTnYfE44sfV/view?usp=share_link)