Bài 05: Lý thuyết các chuẩn giao tiếp cơ bản

Các MCU truyền nhận dữ liệu với nhau hoặc với các thiết bị thông qua tín hiệu điện áp. MCU có thể truyền nhận song song, nối tiếp các tín hiệu điện áp này thông quá các chân được cấu hình riêng biệt.

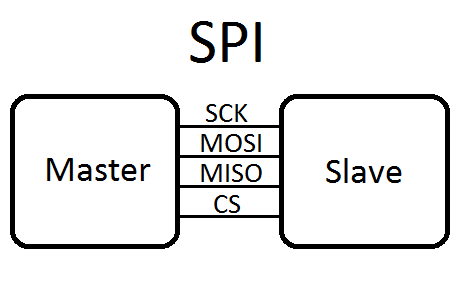
Để việc truyền nhận được dễ dàng với nhiều dòng MCU và phần cứng, các chuẩn giao tiếp được tạo ra. Vi điều khiển sẽ sử dụng các chuẩn giao tiếp khác nhau để liên lạc với nhau hoặc liên lạc với các thiết bị khác hay các phần tử khác trên mạch. Có thể kể đến như I2C, SPI, UART, ….

# 1 SPI.

SPI – Serial Peripheral Interface – hay còn gọi là giao diện ngoại vi nối tiếp, được phát triển bởi hãng Motorola.

Chuẩn đồng bộ nối truyền dữ liệu ở chế độ full - duplex (hay gọi là "song công toàn phần". Nghĩa là tại 1 thời điểm có thể xảy ra đồng thời quá trình truyền và nhận. Là giao tiếp đồng bộ, bất cứ quá trình nào cũng đều được đồng bộ với xung clock sinh ra bởi thiết bị Master

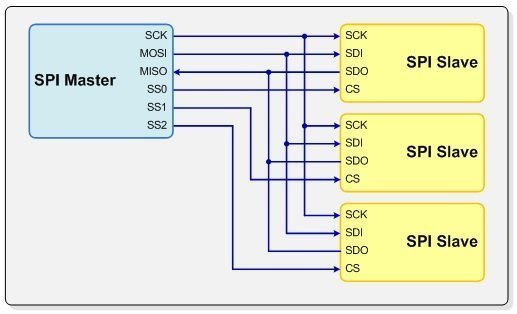
Tốc độ truyền thông cao: SPI cho phép truyền dữ liệu với tốc độ rất nhanh, thường đạt được tốc độ Mbps hoặc thậm chí hàng chục Mbps. Điều này rất hữu ích khi cần truyền dữ liệu nhanh và đáng tin cậy trong các ứng dụng như truyền thông không dây, điều khiển từ xa và truyền dữ liệu đa phương tiện.



SPI sử dụng 4 đường giao tiếp nên đôi khi được gọi là chuẩn truyền thông “ 4 dây”:

* SCK (Serial Clock): Thiết bị Master tạo xung tín hiệu SCK và cung cấp cho Slave. Xung này có chức năng giữ nhịp cho giao tiếp SPI. Mỗi nhịp trên chân SCK báo 1 bit dữ liệu đến hoặc đi → Quá trình ít bị lỗi và tốc độ truyền cao.
* MISO (Master Input Slave Output): Tín hiệu tạo bởi thiết bị Slave và nhận bởi thiết bị Master. Đường MISO phải được kết nối giữa thiết bị Master và Slave.
* MOSI (Master Output Slave Input): Tín hiệu tạo bởi thiết bị Master và nhận bởi thiết bị Slave. Đường MOSI phải được kết nối giữa thiết bị Master và Slave.
* SS (Slave Select): Chọn thiết bị Slave cụ thể để giao tiếp. Để chọn Slave giao tiếp thiết bị Master chủ động kéo đường SS tương ứng xuống mức 0 (Low). Chân này đôi khi còn được gọi là CS (Chip Select). Chân SS của vi điều khiển (Master) có thể được người dùng tạo bằng cách cấu hình 1 chân GPIO bất kỳ chế độ Output.

SPI cho phép 1 MCU chủ giao tiếp với nhiều thiết bị tớ thông qua tín hiệu chọn thiết bị SS. Các thiết bị tớ chỉ có thể có 1 chân CS để nhận tín hiệu chọn này, tuy nhiên thiết bị chủ có thể có nhiều hơn 1 chân SS để chọn từng thiết bị muốn giao tiếp.

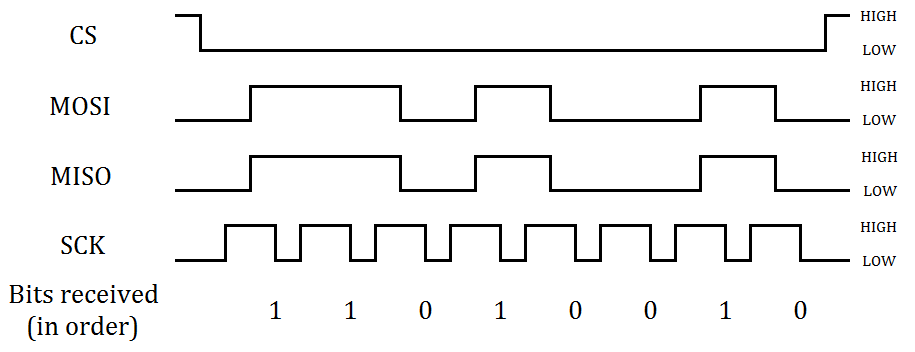


Khung truyền SPI:

Mỗi chip Master hay Slave đều có một thanh ghi dữ liệu 8 bits. Quá trình truyền nhận giữa Master và Slave xảy ra đồng thời theo chu kỳ clock ở chân CLK, một byte dữ liệu được truyền theo cả 2 hướng

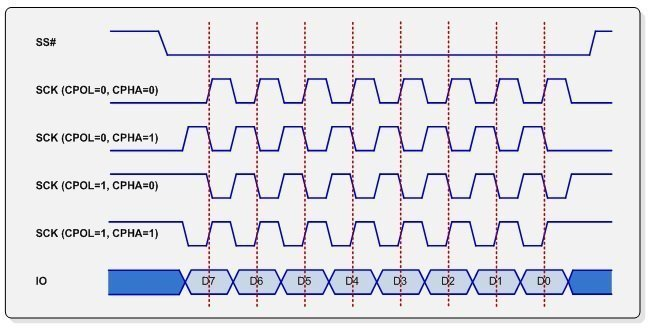
Quá trình trao đổi dữ liệu bắt đầu khi Master tạo 1 xung clock từ bộ tạo xung nhịp (Clock Generator) và kéo đường SS của Slave mà nó truyền dữ liệu xuống mức Low. Mỗi xung clock, Master sẽ gửi đi 1 bit từ thanh ghi dịch (Shift Register) của nó đến thanh ghi dịch của Slave thông qua đường MOSI. Đồng thời Slave cũng gửi lại 1 bit đến cho Master qua đường MISO.Như vậy sau 8 chu kỳ clock thì hoàn tất việc truyền và nhận 1 byte dữ liệu.

Trong giao tiếp SPI, chỉ có thể có 1 Master nhưng có thể 1 hoặc nhiều Slave cùng lúc. Ở trạng thái nghỉ, chân SS của các Slave ở mức 1, muốn giao tiếp với Slave nào thì ta chỉ việc kéo chân SS của Slave đó xuống mức 0.



Chế độ hoạt động:

SPI có 4 chế độ hoạt động phụ thuộc vào cực của xung giữ (Clock Polarity – CPOL) và pha (Phase - CPHA). CPOL dùng để chỉ trạng thái của chân SCK ở trạng thái nghỉ. Chân SCK giữ ở mức cao khi CPOL=1 hoặc mức thấp khi CPOL=0. CPHA dùng để chỉ các mà dữ liệu được lấy mẫu theo xung. Dữ liệu sẽ được lấy ở cạnh lên của SCK khi CPHA=0 hoặc cạnh xuống khi CPHA=1.

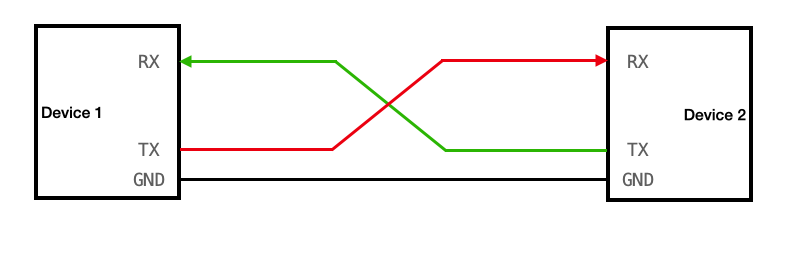


* Mode 0 (mặc định) – xung nhịp của đồng hồ ở mức thấp (CPOL = 0) và dữ liệu được lấy mẫu khi chuyển từ thấp sang cao (cạnh lên) (CPHA = 0).
* Mode 1 - xung nhịp của đồng hồ ở mức thấp (CPOL = 0) và dữ liệu được lấy mẫu khi chuyển từ cao sang thấp (cạnh xuống) (CPHA = 1).
* Mode 2 - xung nhịp của đồng hồ ở mức cao (CPOL = 1) và dữ liệu được lấy mẫu khi chuyển từ cao sang thấp (cạnh lên) (CPHA = 0).
* Mode 3 - xung nhịp của đồng hồ ở mức cao (CPOL = 1) và dữ liệu được lấy mẫu khi chuyển từ thấp sang cao (cạnh xuông) (CPHA = 1).

# 2 UART.

UART (Universal Asynchronous Receiver-Transmitter – Bộ truyền nhận dữ liệu không đồng bộ) là một giao thức truyền thông phần cứng dùng giao tiếp nối tiếp không đồng bộ và có thể cấu hình được tốc độ

Giao thức UART là một giao thức đơn giản và phổ biến, bao gồm hai đường truyền dữ liệu độc lập là TX (truyền) và RX (nhận). Dữ liệu được truyền và nhận qua các đường truyền này dưới dạng các khung dữ liệu (data frame) có cấu trúc chuẩn, với một bit bắt đầu (start bit), một số bit dữ liệu (data bits), một bit kiểm tra chẵn lẻ (parity bit) và một hoặc nhiều bit dừng (stop bit).



Thông thường, tốc độ truyền của UART được đặt ở một số chuẩn, chẳng hạn như 9600, 19200, 38400, 57600, 115200 baud và các tốc độ khác. Tốc độ truyền này định nghĩa số lượng bit được truyền qua mỗi giây. Các tốc độ truyền khác nhau thường được sử dụng tùy thuộc vào ứng dụng và hệ thống sử dụng.

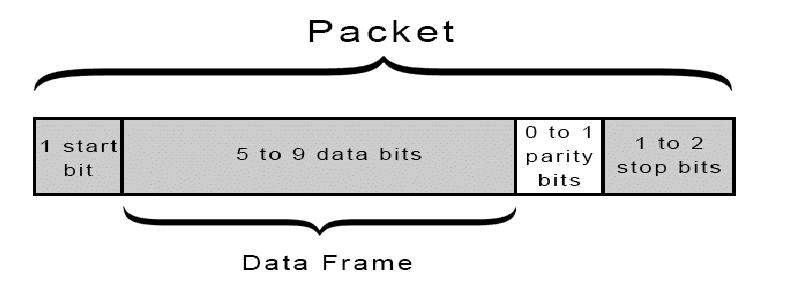
Uart truyền dữ liệu nối tiếp, theo 1 trong 3 chế độ:

* Simplex: Chỉ tiến hành giao tiếp một chiều
* Half duplex: Dữ liệu sẽ đi theo một hướng tại 1 thời điểm
* Full duplex: Thực hiện giao tiếp đồng thời đến và đi từ mỗi master và slave

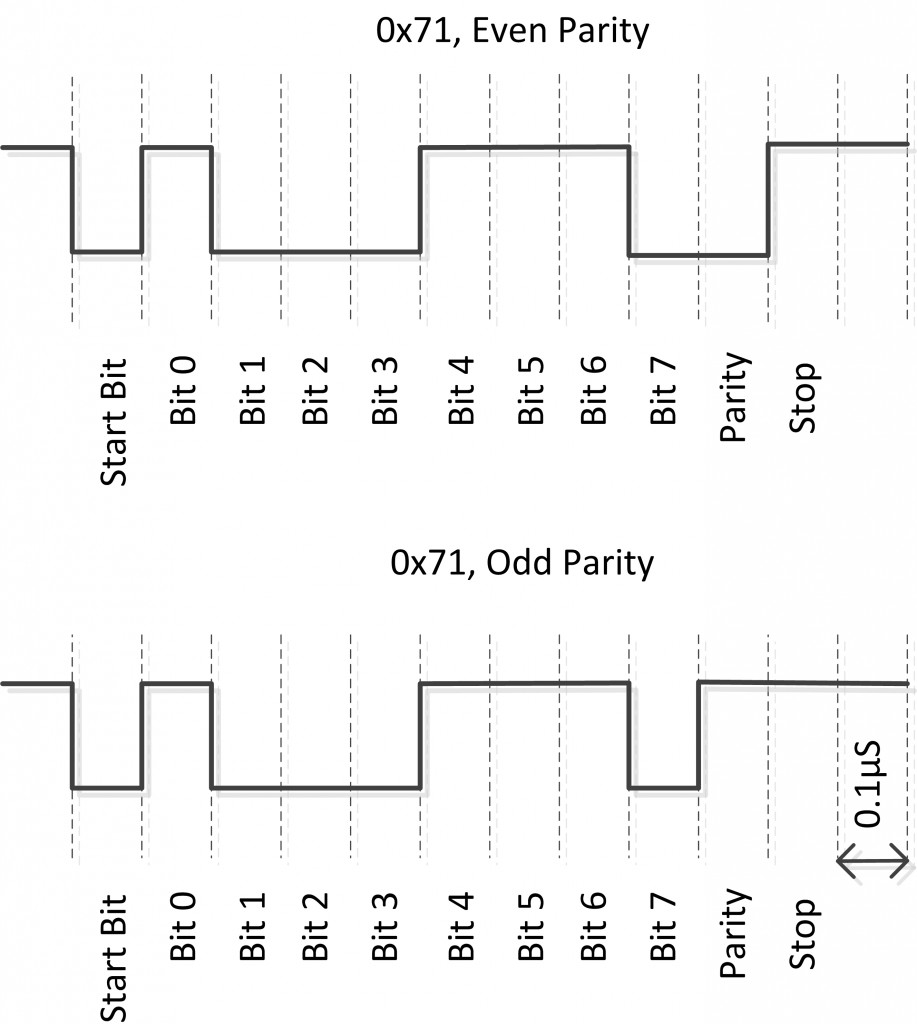
Chân Tx (truyền) của một chip sẽ kết nối trực tiếp với chân Rx (nhận) của chip khác và ngược lại. Quá trình truyền dữ liệu thường sẽ diễn ra ở 3.3V hoặc 5V. Uart là một giao thức giao tiếp giữa một master và một slave. Trong đó 1 thiết bị được thiết lập để tiến hành giao tiếp với chỉ duy nhất 1 thiết bị khác.

Dữ liệu truyền đến và đi từ Uart song song với thiết bị điều khiển. Khi tín hiệu gửi trên chân Tx (truyền), bộ giao tiếp Uart đầu tiên sẽ dịch thông tin song song này thành dạng nối tiếp và sau đó truyền tới thiết bị nhận. Chân Rx (nhận) của Uart thứ 2 sẽ biến đổi nó trở lại thành dạng song song để giao tiếp với các thiết bị điều khiển.

Dữ liệu truyền qua Uart sẽ đóng thành các gói (packet). Mỗi gói dữ liệu chứa 1 bit bắt đầu, 5 – 9 bit dữ liệu (tùy thuộc vào bộ Uart), 1 bit chẵn lẻ tùy chọn và 1 bit hoặc 2 bit dừng.



Quá trình truyền dữ liệu Uart sẽ diễn ra dưới dạng các gói dữ liệu này, bắt đầu bằng 1 bit bắt đầu, đường mức cao được kéo dần xuống thấp. Sau bit bắt đầu là 5 – 9 bit dữ liệu truyền trong khung dữ liệu của gói, theo sau là bit chẵn lẻ tùy chọn để nhằm xác minh việc truyền dữ liệu thích hợp. Sau cùng, 1 hoặc nhiều bit dừng sẽ được truyền ở nơi đường đặt tại mức cao. Vậy là sẽ kết thúc việc truyền đi một gói dữ liệu



# 3 I2C.

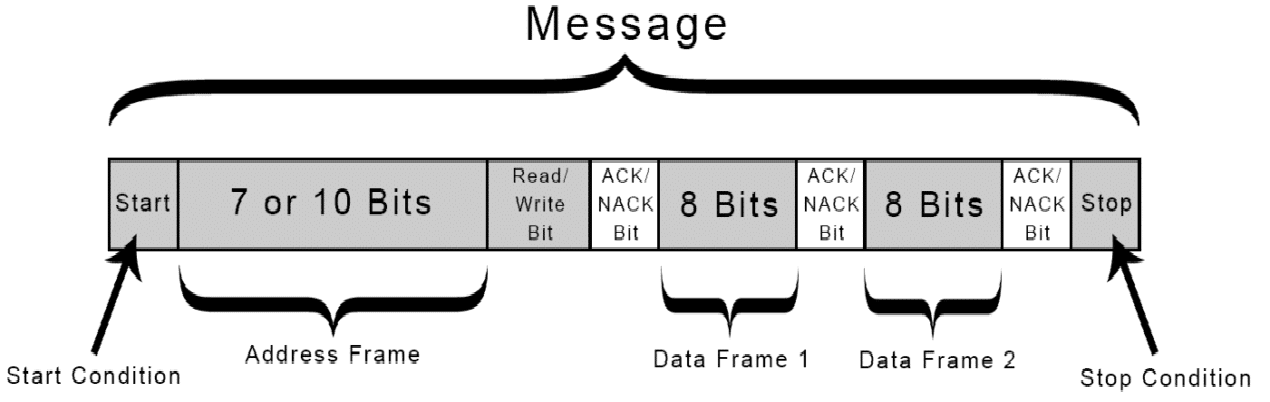
I2C kết hợp các tính năng tốt nhất của SPI và UART. Giống như giao tiếp UART, I2C chỉ sử dụng hai dây để truyền dữ liệu giữa các thiết bị:

* 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.

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:



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.

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

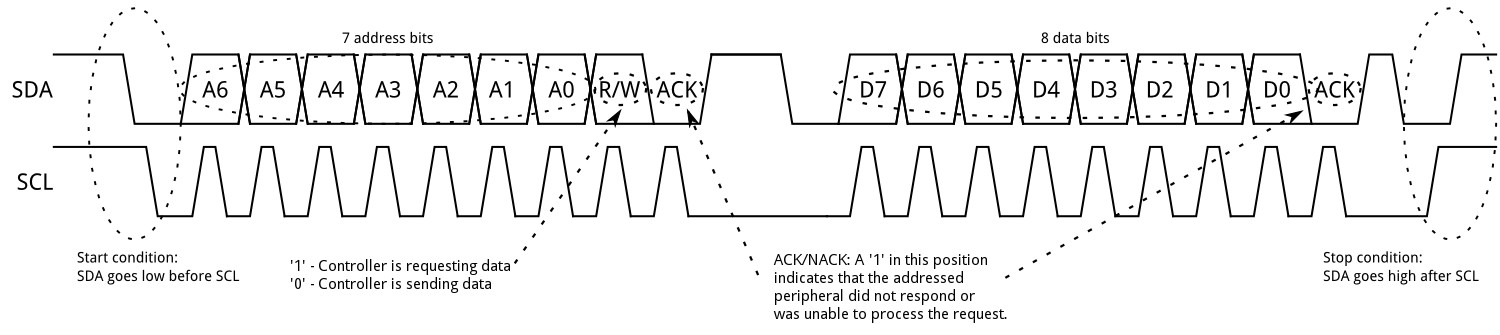
Sau khi master phát hiện bit ACK từ slave, khung dữ liệu đầu tiên đã sẵn sàng được gửi.

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**

1. 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.
2. Master gửi cho mỗi slave địa chỉ 7 hoặc 10 bit của slave mà nó muốn giao tiếp, cùng với bit đọc / ghi.
3. 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.
4. Master gửi hoặc nhận khung dữ liệu.
5. 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.
6. 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 (2 mũ 7) địa chỉ duy nhất. Việc sử dụng địa chỉ 10 bit không phổ biến, nhưng nó cung cấp 1.024 (2 mũ 10) địa chỉ duy nhất.

**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.

