Giao tiếp I2C

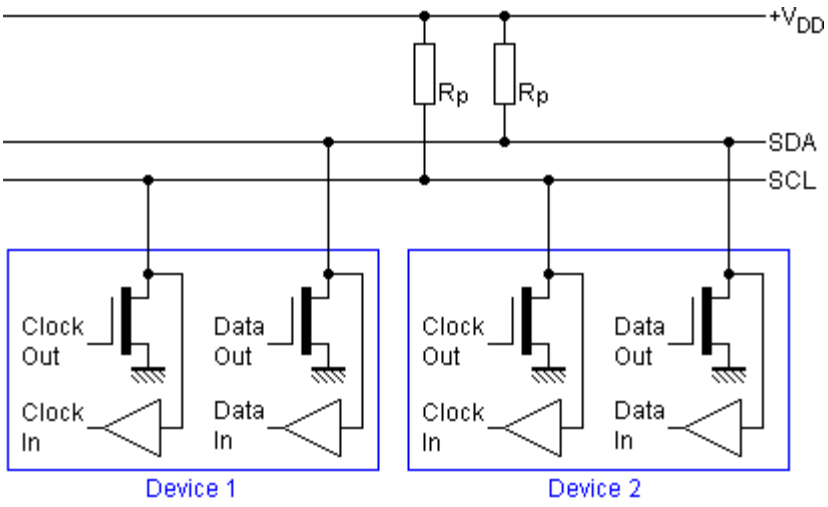
Khái niệm

I²C, viết tắt của từ tiếng Anh *"Inter-Integrated Circuit"*, là một loại bus nối tiếp được phát triển bởi hãng sản xuất linh kiện điện tử Philips. Ban đầu, loại bus này chỉ được dùng trong các linh kiện điện tử của Philips. Sau đó, do tính ưu việt và đơn giản của nó, **I²C** đã được chuẩn hóa và được dùng rộng rãi trong các mô đun truyền thông nối tiếp của vi mạch tích hợp ngày nay.

Cấu tạo và nguyên lý hoạt động

I²C sử dụng hai đường truyền tín hiệu:

- Một đường xung nhịp đồng hồ(SCL) chỉ do Master phát đi (thông thường ở 100kHz và 400kHz. Mức cao nhất là 1Mhz và 3.4MHz).
- Một đường dữ liệu(SDA) theo 2 hướng.
- Sơ đồ kết nối như hình dưới.

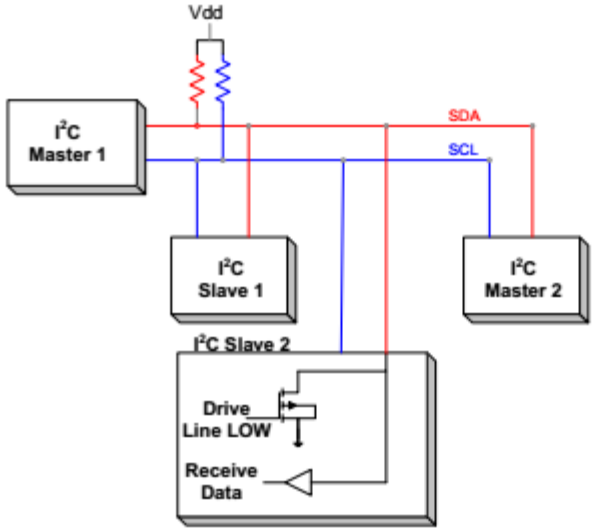


Có một lưu ý nhỏ về xung clock. Bản chất của I2C là dữ liệu trên đường SDA chỉ được ghi nhận ở sườn lên của chân CLK. Do vậy xung clock có thể không cần chính xác tốc độ

là 1MHz hay 3.4Mhz. Lợi dụng điểm này có thể sử dụng 2 chân GPIO để làm chân giao tiếp I2C mềm mà không nhất thiết cần một chân CLK tạo xung với tốc độ chính xác (có thể chỉ cần dùng delay và bật tắt mức logic, tham khảo phần code ở cuối bài :D)

SCL và SDA luôn được kéo lên nguồn bằng một điện trở kéo lên có giá trị xấp xỉ 4,7 KOhm (tùy vào từng thiết bị và chuẩn giao tiếp, có thể dao động trong khoảng 1KOhm đến 4.7 Kohm. Chú ý rằng theo cấu hình này, một thiết bị có thể ở mức logic LOW hay cao trở nhưng ko thể ở dạng HIGH => Chính trở pull up tạo ra mức logic HIGH).

Figure 2. Open Drain Drives LOW Pin Configuration



Lý do là các chân này có dạng opendrain để có thể **hoạt động ở các mức điện áp logic khác nhau.**

Việc lựa chọn trở pull up phù hợp sẽ được trình bày ở phần sau.

Các chế độ hoạt động của I²C

Dựa vào tốc độ ta chia làm 2 loại

- Chế độ chuẩn (standard mode) hoạt động ở tốc độ 100 Kbit/s.
- Chế độ tốc độ thấp (low-speed mode) hoạt động ở tốc độ 10 Kbit/s.

Nếu chia theo quan hệ chủ tớ:

- Một chủ một tớ.
- Một chủ nhiều tớ.
- Nhiều chủ nhiều tớ.

Quá trình truyền dữ liệu

- Thiết bị A (chủ) xác định đúng địa chỉ của thiết bị B (Tớ), cùng với việc xác định địa chỉ, thiết bị A sẽ quyếtđịnh đọc hay ghi vào thiết bị tớ.
- Thiết bị A gửi gữi liệu tới thiết bị B
- Thiết bị A kết thúc quá trình truyền dữ liệu.
- Khi A muốn nhận dữ liệu từ B, quá trình diễn ra tương tự, chỉ khác A sẽ nhận dữ liệu từ B.

Tần số xung nhịp đồng hồ có thể xuống 0 Hz.

Cách đánh địa chỉ

I²C sử dụng 7 bit để định địa chỉ, do đó trên một bus có thể định địa chỉ tới 112 nút, 16 địa chỉ còn lại được sử dụng vào mục đích riêng. Bit còn lại quy định việc đọc hay ghi dữ liệu (1 là write, 0 là read)

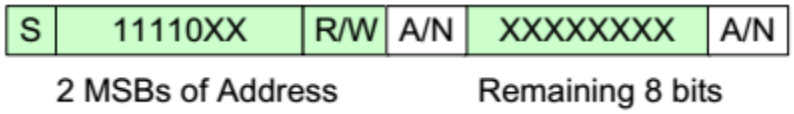
Ví dụ:

– Địa chỉ của một thiết bị là 0x20. Khi cần đọc vào thiết bị này thì thanh ghi sẽ có giá trị 0x40 (thêm bit 0) còn khi ghi thì giá trị là 0x41 (thêm vào 0).

Điểm mạnh của I²C chính là hiệu suất và sự đơn giản của nó: một khối điều khiển trung tâm có thể điều khiển cả một mạng thiết bị mà chỉ cần hai lối ra điều khiển.

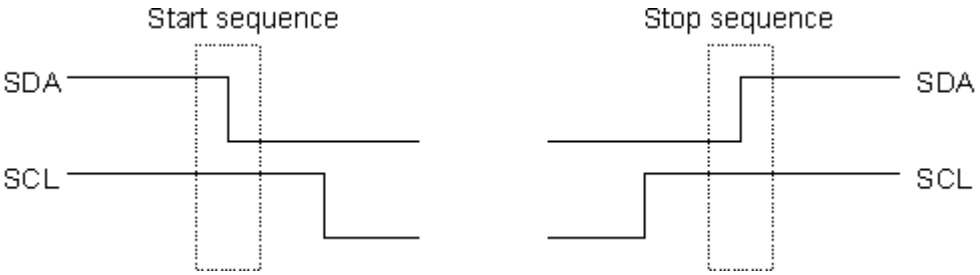
Ngoài ra I2C còn có chế độ 10bit địa chỉ:

Figure 5. 10-Bit Address



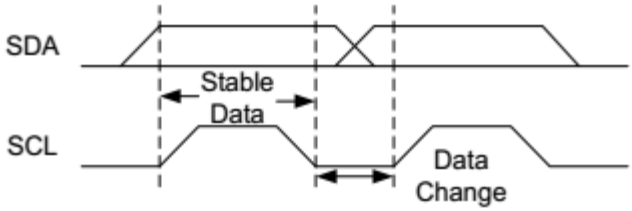
Định dạng dữ liệu truyền

Dữ liệu được truyền trên bus I2C theo từng bit, bit dữ liệu được truyền đi tại mỗi sườn lên của xung clock trên SCKL , Quá trình thay đổi bit dữ liệu xảy ra khi SCL ở mức thấp.



- **Start = HIGH to LOW on SDA when SCL is HIGH**
- **Stop = LOW TO HIGH on SDA when SCL is HIGH**
- **Other when SCL low => Data!**

Figure 6. SDA Data Change

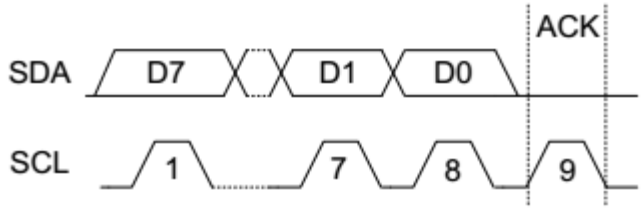


Mỗi byte dữ liệu được truyền có độ dài là 8 bits. Số lượng byte có thể truyền trong một lầ n là không hạn chế.

Mỗi byte được truyền sẽ chờ tín hiệu phản hồi là một bit ACK để báo hiệu đã

nhận dữ liệu. => **Mỗi lần I2C sẽ truyền 8bit và nhận 1bit.**

Figure 7. Eight Data Bits Followed by an ACK Bit



Bit có trọng số cao nhất (MSB) sẽ được truyền đi đầu tiên, các bít sẽ được truyền đi lần l ượt. Sau 8 xung clock trên dây SCL, 8 bit dữ liệu đã được truyền đi. Lúc này thiết bị nhận , sau khi đã nhận đủ 8 bít dữ liệu sẽ kéo SDA xuống mức thấp tạo một xung ACK ứng với xung clock thứ 9 trên dây SDA để báo hiệu đã nhận đủ 8 bit. Thiết bị truyền

khi nhận được bit ACK sẽ tiếp tục thực hiện quá trình truyền hoặc kết thúc.

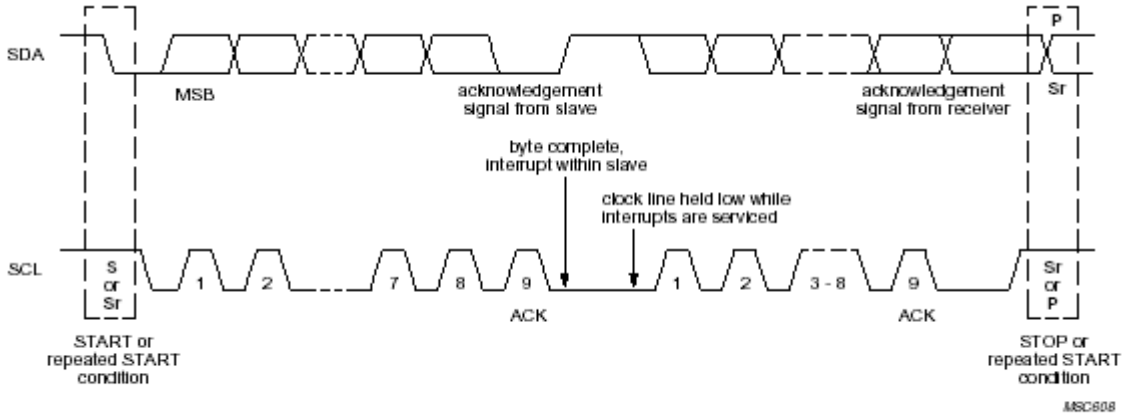


Fig.6 Data transfer on the I²C-bus.

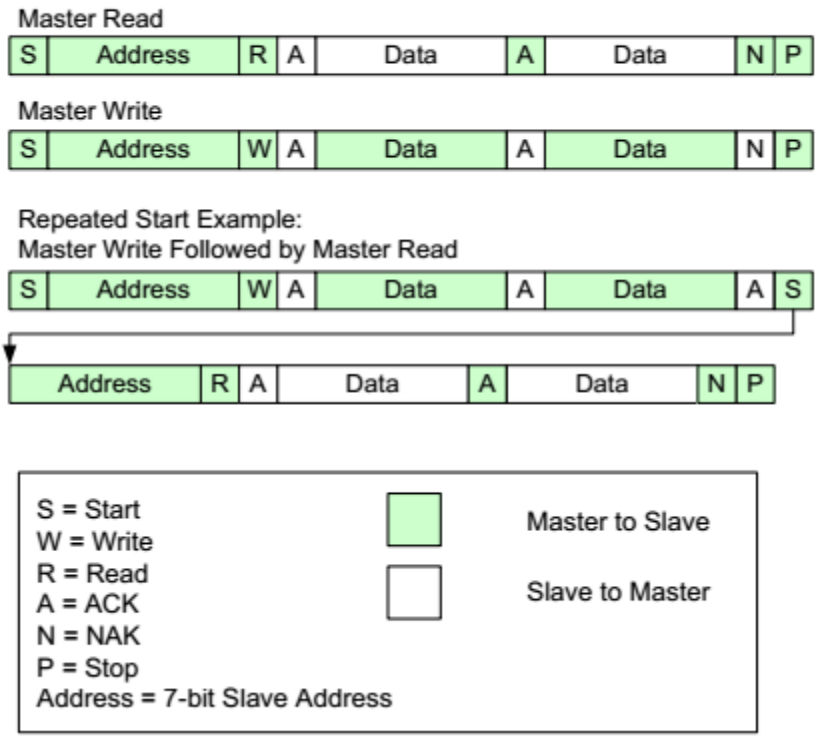
Thuật toán truyền nhận dữ liệu:

B1: Host xác định thiết bị cần giao tiếp và chế độ giao tiếp là read hay là write.

việc này được thực hiện bằng cách gửi 7bit địa chỉ thiết bị và thêm bit cuối cùng, 0 nếu read và 1 nếu write.

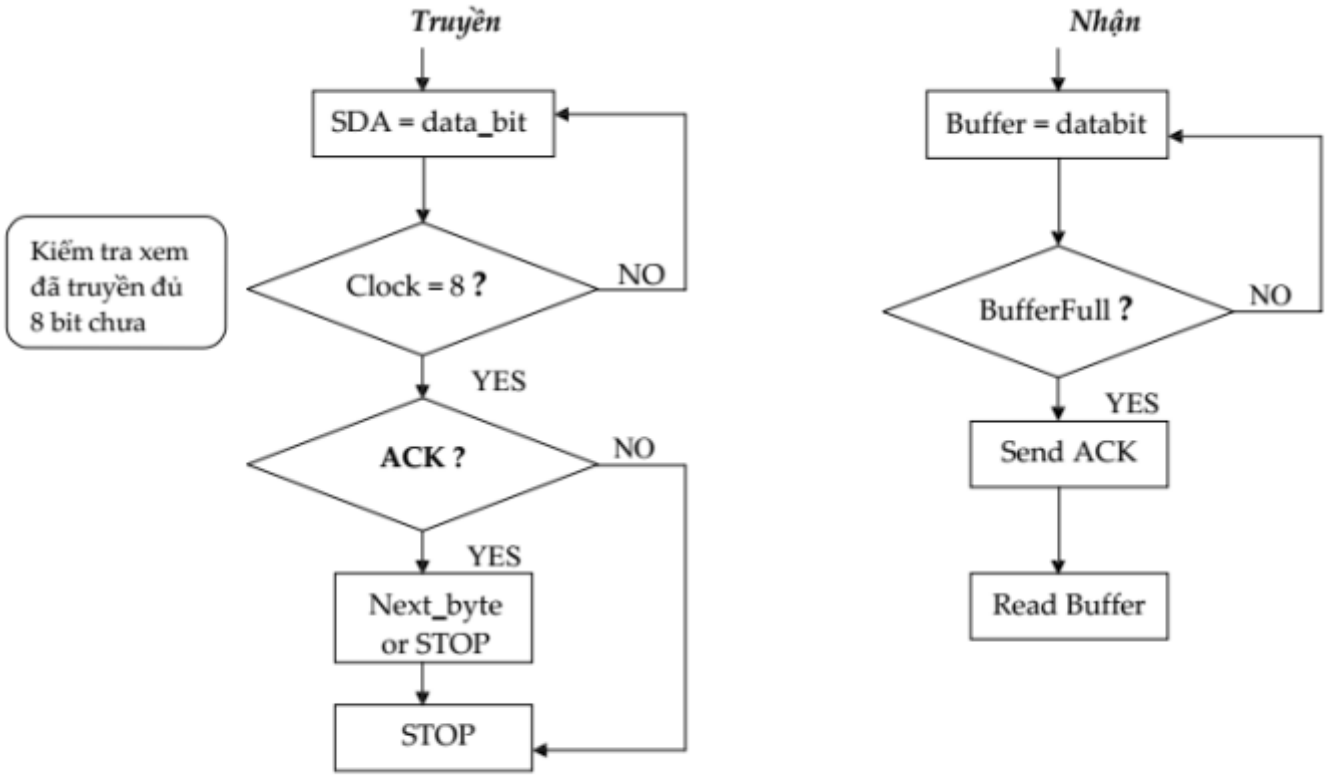
B2: Reset chế độ bằng cách thực hiện liên tiếp việc start và stop.

Figure 9. Complete Transaction



B3: Gửi địa chỉ thanh ghi cần truy nhập của thiết bị cũng như chế độ read hay write.

B4: Gửi hoặc nhận 1byte dữ liệu. Sau khi truyền 1byte dữ liệu, bên nhận đc dữ liệu sẽ gửi lại 1bit ACK để xác nhận đã nhận được dữ liệu và tiếp tục truyền hoặc bit NACK để báo nhận đc dữ liệu nhưng kết thúc quá trình truyên.



Hình 1.8. Lưu đồ thuật toán quá trình truyêng nhận dữ liệu

Chú ý: ASK là bit do slave truyền chứ ko phải do master truyền

Một byte truyền đi có kèm theo bit ACK là điều kiên bắt buộc, nhằm đảm bảo cho quá trình truyền nhận được diễn ra chính xác. Khi không nhận được đúng địa chỉ hay khi muốn kết thúc quá trình giao tiếp, thiết bị nhận sẽ gửi một xung Not-ACK (SDA ở mức cao) để báo cho thiết bị chủ biết, thiết bị chủ sẽ tạo xung STOP để kết thúc hay lặp lại mộ t xung START để bắt đầu quá trình mới.

Chọn trở Pullup I2C – I2C Bus pullup resistor Calculation

Do I2C sử dụng đầu ra dạng Opendrain/Open colector nên có thể sử dụng với nhiều dạng điện áp khác nhau. Để làm được điều này ta cần sử dụng trở pullup để kéo lên

mức điện áp phù hợp. Giá trị của trở pullup tương đối quan trọng, nếu chọn không phù hợp có thể dẫn đến việc mất mát tín hiệu.

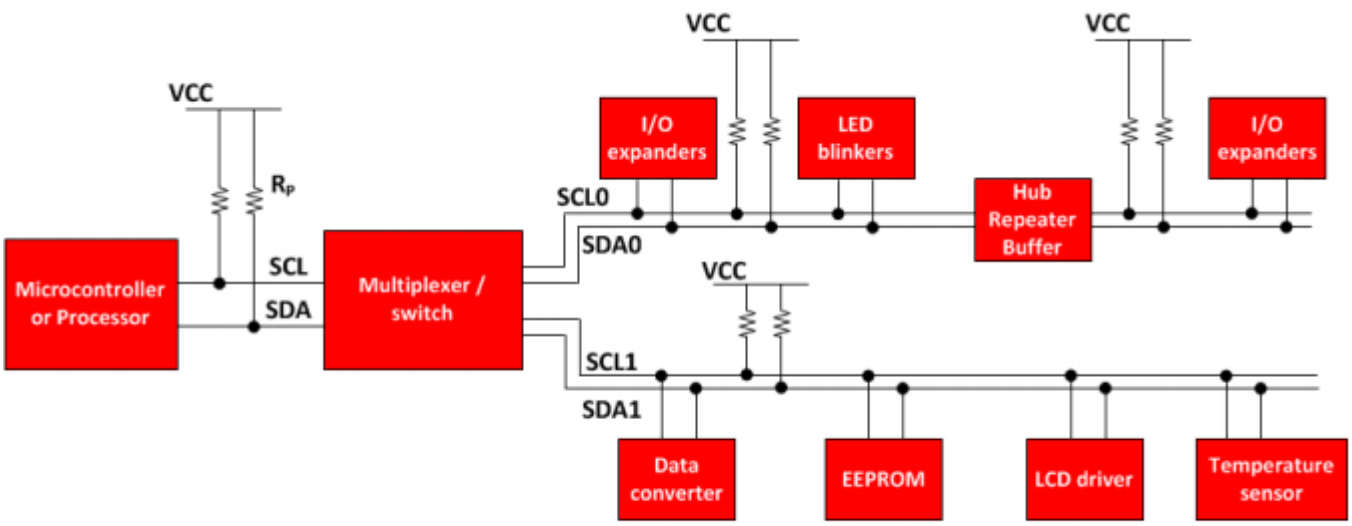


Figure 1. Application Example Showing I2C Communication Between the Different IC's on a System and With Pullup Resistors on I2C Bus

Giá trị trở pull up phù hợp cần đảm bảo 2 yếu tố:

- Thỏa mãn phù hợp mức logic
- Đảm bảo rise time của tín hiệu.

Để IC nhận ra đúng mức logic thì giá trị điện áp tại chân phải lớn hơn V_{OL}. Ta có công thức tính trở min như sau:

$$R_P(min) = \frac{(V_{CC} - V_{OL}(max))}{I_{OL}}$$

Ngoài ra, do đặc thù của I2C có thêm phần rise time. Nếu giá trị trở quá lớn sẽ dẫn đến việc rise time cao.

Xem kết nối như một mạch RC (C là tụ kháng sinh) thì ta có điện áp theo thời gian tính theo công thức sau:

$$V(t) = V_{CC} \times \left(1 - e^{\frac{-t}{RC}}\right)$$

Thông thường, VIH và VIL thường tính lần lượt bằng 0.7*VCC và 0.3*VCC nên ta có:

For $V_{IH} = 0.7 \times V_{CC}$:

$$V_{IH} = 0.7 \times V_{CC} = V_{CC} \times \left(1 - e^{\frac{-t1}{Rp \times C_b}}\right)$$

For $V_{IL} = 0.3 \times V_{CC}$:

$$V_{IL} = 0.3 \times V_{CC} = V_{CC} \times \left(1 - e^{\frac{-t2}{Rp \times C_b}}\right)$$

The rise time for the I2C bus can be written as:

$$t_r = t_2 - t_1 = 0.8473 \times R_p \times C_b$$

Từ đó ta có R_{MAX} là

$$R_p(max) = \frac{t_r}{(0.8473 \times C_b)}$$

các giá trị trên thường cho theo bảng trong datasheet

Table 1. Parametrics from I2C specifications

Parameter		Standard Mode (Max)	Fast Mode (Max)	Fast Mode Plus (Max)	Unit
t_r	Rise time of both SDA and SCL signals	1000	300	120	ns
C_b	Capacitive load for each bus line	400	400	550	pF
V_{OL}	Low-level output voltage (at 3 mA current sink, V_{CC} > 2 V)	0.4	0.4	0.4	V
	Low-level output voltage (at 2 mA current sink, $V_{CC} \leq 2$ V)	–	0.2 × V_{CC}	0.2 × V_{CC}	V

Ví dụ về tính trở pull up

4 Example

For Fast-mode I2C communication with the following parameters, calculate the pullup resistor value.

C_b = 200 pF, V_{CC} = 3.3 V

Solution:

Taking the values from Table 1:

$$R_P(\max) = \frac{t_r}{(0.8473 \times C_b)} = \frac{\left(300 \times 10^{-9}\right)}{\left(0.8473 \times 200 \times 10^{-12}\right)} = 1.77\ \text{k}\Omega \tag{7}$$

$$R_P(\min) = \frac{V_{CC} - V_{OL}(\max)}{I_{OL}} = \frac{(3.3 - 0.4)}{\left(3 \times 10^{-3}\right)} = 966.667\ \Omega \tag{8}$$

Therefore, we can select any available resistor value between 966.667 Ω and 1.77 kΩ. The value of the pullup resistor can be selected based on the trade-off for the power consumption and speed.