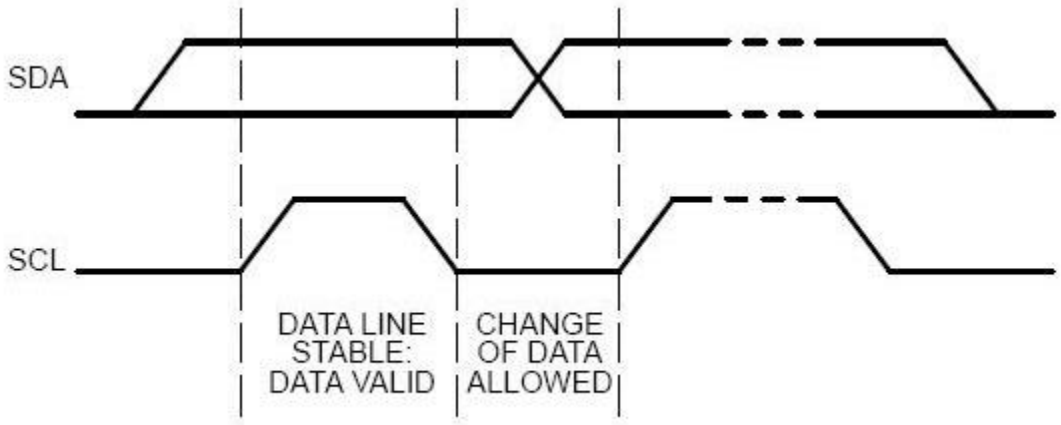
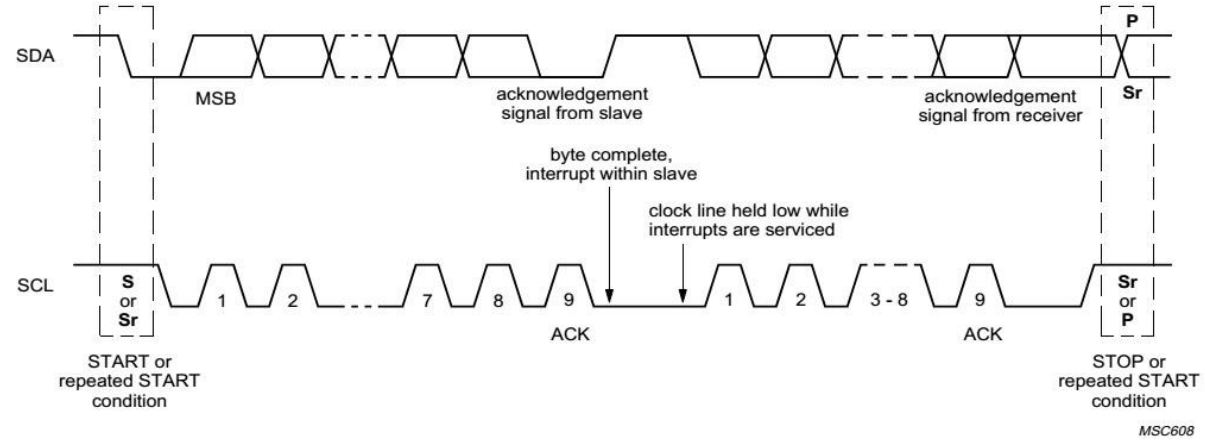
Định dạng dữ liệu truyền

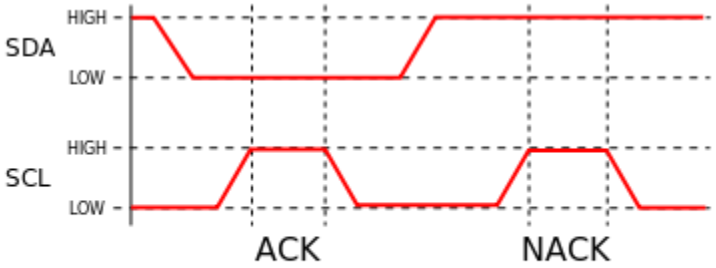
Dữ liệu được truyền trên bus I2C theo từng bit, bit dữ liệu được truyền đi tại mỗi cạnh lên của xung đồng hồ trên dây SCL, quá trình thay đổi bit dữ liệu xảy ra khi SCL đang ở mức thấp.



Hình 5: Quá trình truyền 1 bit dữ liệu

Mỗi byte dữ liệu được truyền có độ dài là 8 bit. Số lượng byte có thể truyền trong một lần là không hạn chế. Mỗi byte được truyền đi theo sau là một bit ACK để báo hiệu đã nhận dữ liệu. Bit có trọng số cao nhất (MSB) sẽ được truyền đi đầu tiên, các bit sẽ được truyền đi làn lượt. 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.



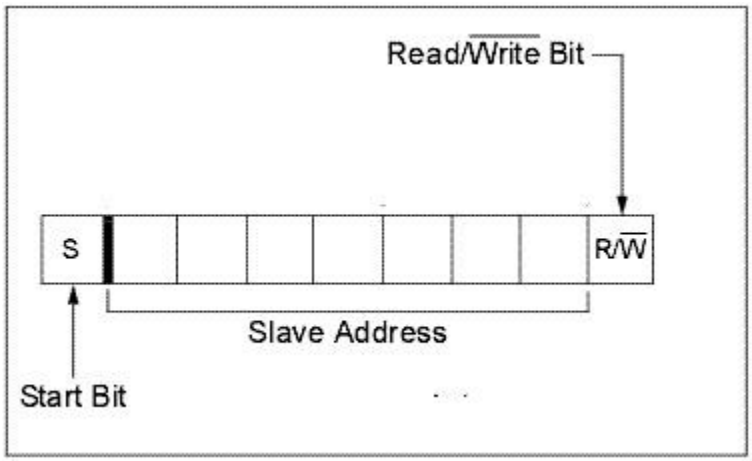


Một byte truyền đi có kèm theo bit ACK là điều kiện bắt buộc, nhằm đảm bảo cho quá trình truyền nhận được diễn ra chính xác. Khi không nhận được đúng địa chỉ hay khi muốn kết thúc quá trình giao tiếp thiết bị nhận sẽ gửi một xung Not-ACK(SDA ở mức cao) để báo cho thiết bị chủ biết, thiết bị chủ sẽ tạo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

Định dạng địa chỉ thiết bị

Mỗi thiết bị ngoại vi tham gia vào bus I2C đều có một địa chỉ duy nhất, nhằm phân biệt giữa các thiết bị với nhau. Độ dài địa chỉ là 7 bit, điều đó có nghĩa là trên một bus I2C ta có thể phân biệt tối đa 128 thiết bị. Khi thiết bị chủ muốn giao tiếp với ngoại vi nào trên

bus I2C, nó sẽ gửi 7 bit địa chỉ của thiết bị đó ra bus ngay sau xung START. Byte đầu tiên được gửi sẽ bao gồm 7 bit địa chỉ và một bít thứ 8 điều khiển hướng truyền.

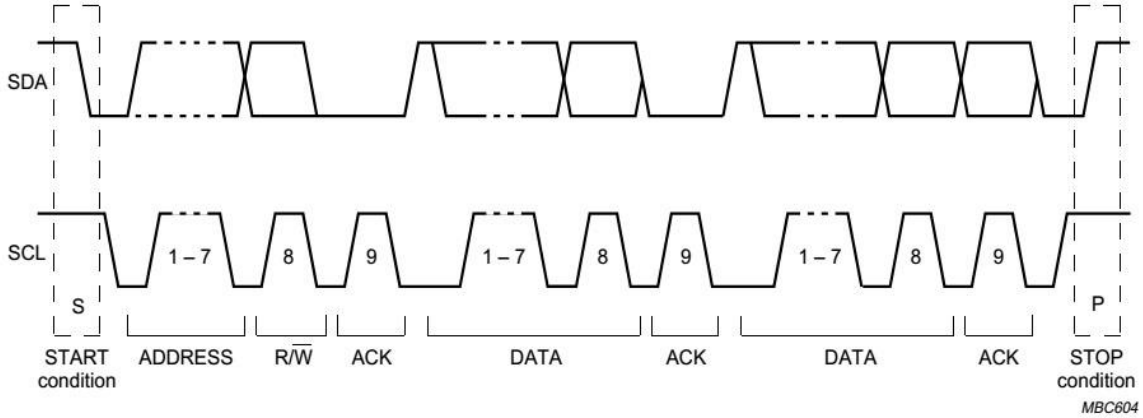


Mỗi một thiết bị ngoại vi sẽ có một địa chỉ riêng do nhà sản xuất ra nó quy định. Địa chỉ đó có thể là cố định hay thay đổi. Riêng bit điều khiển hướng sẽ quy định chiều truyền dữ liệu. Nếu bit này bằng "0" có nghĩa là byte dữ liệu tiếp theo sau sẽ được truyền từ chủ đến tớ, còn ngược lại nếu bằng "1" thì các byte theo sau byte đầu tiên sẽ là dữ liệu từ con tớ gửi đến con chủ. Việc thiết lập giá trị cho bit này do con chủ thi hành, con tớ sẽ tùy theo giá trị đó mà có sự phản hồi tương ứng đến con chủ.

Hiện nay có thể đánh địa chỉ các thiết bị trên bus I2C dưới dạng 10 bit địa chỉ. Việc thực hiện đánh dấu địa chỉ theo khung 10 bit được thực hiện nếu sau lệnh START ta gửi chuỗi 11110 (số nhị phân) ra đường SDA.

Truyền dữ liệu trên bus I2C, chế độ Master - Slave

Việc truyền dữ liệu diễn ra giữa con chủ và con tớ. Dữ liệu truyền có thể theo 2 hướng, từ chủ đến tớ hay ngược lại. Hướng truyền được quy định bởi bit thứ 8 R/W trong byte đầu tiên (byte địa chỉ) được truyền đi.



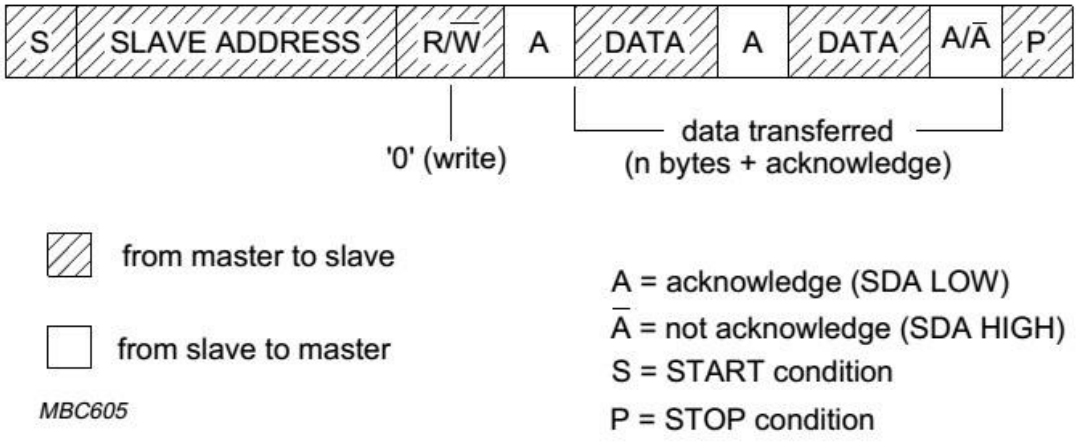
Truyền dữ liệu từ chủ đến tớ (ghi dữ liệu): Thiết bị chủ khi muốn ghi dữ liệu đến con tớ, quá trình thực hiện là:

- Thiết bị chủ tạo xung START

- Thiết bị chủ gửi địa chỉ của thiết bị tớ mà nó cần giao tiếp cùng với bit R/W= 0 ra bus và đợi xung ACK phản hồi từ con tớ

- Khi nhận được xung ACK báo đã nhận diện đúng thiết bị tớ, con chủ bắt đầu gửi dữ liệu đến con tớ theo từng byte một. Theo sau mỗi byte này đều là một xung ACK. Số lượng byte truyền là không hạn chế.

- Kết thúc quá trình truyền, con chủ sau khi truyền byte cuối sẽ tạo xung STOP báo hiệu kết thúc.



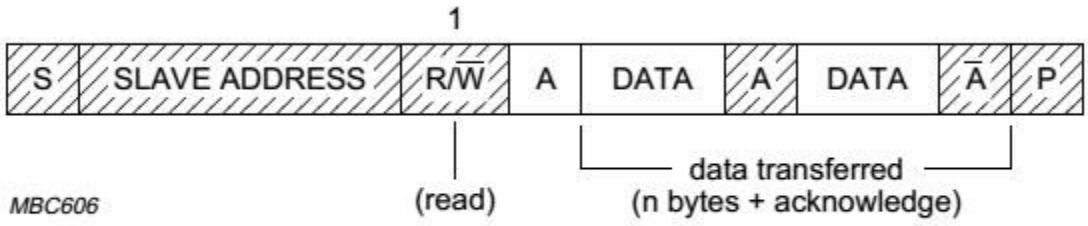
Truyền dữ liệu từ tớ đến chủ (đọc dữ liệu): Thiết bị chủ muốn đọc dữ liệu từ thiết bị tớ, quá trình thực hiện như sau:

- Khi bus rỗi, thiết bị chủ tạo xung START, báo hiệu bắt đầu giao tiếp

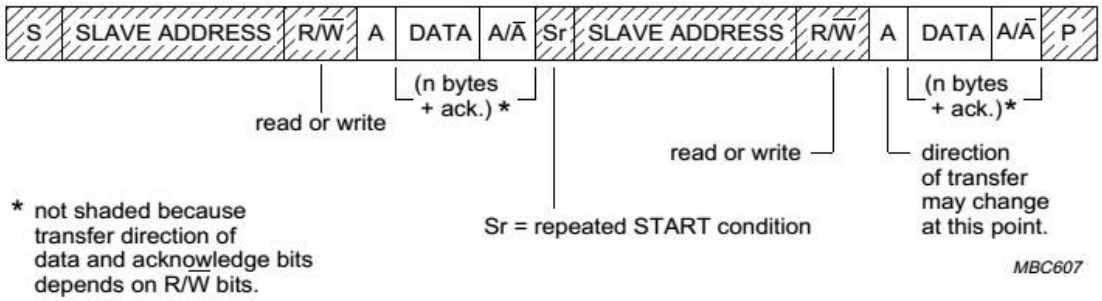
- Thiết bị chủ gửi địa chỉ của thiết bị tớ càn giao tiếp cùng với bit R/W = 1 và đợi xung ACK từ phía thiết bị tớ

- Sau xung ACK từ con tớ, thiết bị tớ sẽ gửi từng byte ra bus, thiết bị chủ sẽ nhận dữ liệu và trả về xung ACK. số lượng byte không hạn chế

- Khi muốn kết thúc quá trình giao tiếp, thiết bị chủ gửi xung Not ACK và tạo xung STOP để kết thúc.



Quá trình kết hợp ghi và đọc dữ liệu: giữa hai xung START và STOP, thiết bị chủ có thể thực hiện việc đọc hay ghi nhiều lần, với một hay nhiều thiết bị. Để thực hiện việc đó, sau một quá trình ghi hay đọc, thiết bị chủ lặp lại một xung START và lại gửi lại địa chỉ của thiết bị tớ và bắt đầu một quá trình mới.



Chế độ giao tiếp MasterSlave là chế độ cơ bản trong một bus I2C, toàn bộ bus được quản lý bởi một master duy nhát. Trong chế độ này sẽ không xảy ra tình trạng xung đột bus hay mất đồng bộ xung clock vì chỉ có một con chủ (master) duy nhất có thể tạo xung clock.

Chế độ Multi Master

Trên bus I2C có thể có nhiều hơn một con chủ điều khiển bus. Khi đó bus I2C sẽ hoạt động ở chế độ Multi Master.

Cấu hình chuẩn giao tiếp I2C trong trình biên dịch CCS

1.#USE I2C (***options***)

Cú pháp:	#USE I2C (***options***)
Các	***Options*** có thể là các lệnh sau

phần tử:	**MASTER**	Thiết lập ở chế độ master (master mode)
	MULTI_MASTER	Thiết lập ở chế độ multi_master (multi_master mode)
	SLAVE	Thiết lập ở chế độ slave mode
	SCL=pin	Chỉ định chân SCL (pin là bit địa chỉ)
	SDA=pin	Chỉ định chân SDA (pin là bit địa chỉ)
	ADDRESS=nn	Chỉ định địa chỉ slave mode
	FAST	Sử dụng tốc độ truyền nhanh (Fast)
	FAST=nnnnnn	Thiết lập tốc độ là nnnnnn hz
	SLOW	Sử dụng tốc độ truyền chậm.
	RESTART_WDT	Khởi động WDT trong khi chờ thực thị I2C_READ
	FORCE_HW	Bắt buộc sử dụng phần cứng khi giao tiếp (Khối module phần cứng I2C)
	FORCE_SW	Bắt buộc sử dụng phần mềm khi giao tiếp (Tự tạo mã giả lập bằng phần mềm do CCS tự sinh code)
	NOFLOAT_HIGH	Does not allow signals to float high, signals are driven from low to high
	SMBUS	Bus được sử dụng không phải là I2C bus, nhưng gần giống.
	STREAM=id	Gán một tên nhận dạng đến cổng I2C này. Tên nhận dạng này có thể sử dụng trong các hàm giống i2c_read hoặc i2c_write.
	NO_STRETCH	Do not allow clock streaching
	MASK=nn	Thiết lập một mặt nạ địa chỉ cho các phần có hỗ trợ Ví dụ: #use I2C(slave,sda=PIN_C4,scl=PIN_C3, address=0xa0, FORCE_HW, mask=0xDF)
	I2C1	Thay vì sử dụng SCL= và SDA= đây thiết lập chân

		giao tiếp tương ứng với module I2C thứ nhất.
	I2C2	Thay vì sử dụng SCL= và SDA= đây thiết lập chân giao tiếp tương ứng với module I2C thứ hai
	NOINIT	Không thực hiện khởi tạo khối ngoại vi I2C, chỉ khởi tạo khi khi sử dụng lệnh I2C_INIT() khi chạy chương trình.
	Có một vài chíp cho phép	
	DATA_HOLD	No ACK is sent until I2C_READ is called for data bytes (slave only)
	ADDRESS_HOLD	No ACK is sent until I2C_read is called for the address byte (slave only)
	SDA_HOLD	Min of 300ns holdtime on SDA a from SCL goes low
Mục đích:	CCS cung cấp hỗ trợ giao tiếp I2C dựa trên phần cứng và trên phần mềm (đối với các vi điều khiển không hỗ trợ module giao tiếp i2C bằng phần cứng), Tham khảo datasheet đối với dòng vi điều khiển mà bạn muốn lập trình, không phải tất cả Pic đều hổ trợ module phần cứng đối với giao tiếp I2C. Thư viện I2C chứa đựng các hàm để thực hiện giao tiếp. #USE I2C ảnh hưởng đến các hàm I2C_START, I2C_STOP, I2C_READ, I2C_WRITE and I2C_POLL đến khi #USE I2C tiếp theo được khai báo. Các hàm được tạo bằng phần mềm ngoại trừ khi bạn khai báo FORCE_HW.	
Ví dụ:	#use I2C(master, sda=PIN_B0, scl=PIN_B1) #use I2C(slave,sda=PIN_C4,scl=PIN_C3, address=0xa0, FORCE_HW)	

	#use I2C(master, scl=PIN_B0, sda=PIN_B1, fast=450000) //Thiết lập tốc độ là 450 KBSP