# CHƯƠNG 3: HỢP NGỮ

### GIỚI THIỆU

Ở chương này khảo sát tập lệnh hợp ngữ của các vi điều khiển. Sau khi kết thúc chương này bạn sẽ biết:

* Biết lệnh nhị phân, lệnh hợp ngữ
* Biết trình biên dịch, trình tự lập trình
* Biết cấu trúc của 1 chương trình hợp ngữ
* Biết phân loại, tổ chức các lệnh hợp ngữ

### NGÔN NGỮ LẬP TRÌNH HỢP NGỮ

Vi điều khiển hay vi xử lý là các IC lập trình, khi bạn đã thiết kế hệ thống điều khiển có sử dụng vi xử lý hay vi điều khiển thì đó mới chỉ là **phần cứng**, muốn hệ thống vận hành thì bạn phải viết một chương trình điều khiển nạp vào bộ nhớ nội bên trong vi điều khiển hoặc bộ nhớ bên ngoài và gắn vào trong hệ thống để hệ thống vận hành và dĩ nhiên bạn phải viết đúng thì hệ thống mới vận hành đúng. Chương trình gọi là **phần mềm**.

Phần mềm và phần cứng có quan hệ với nhau, người lập trình phải hiểu rõ hoạt động của phần cứng để viết chương trình.

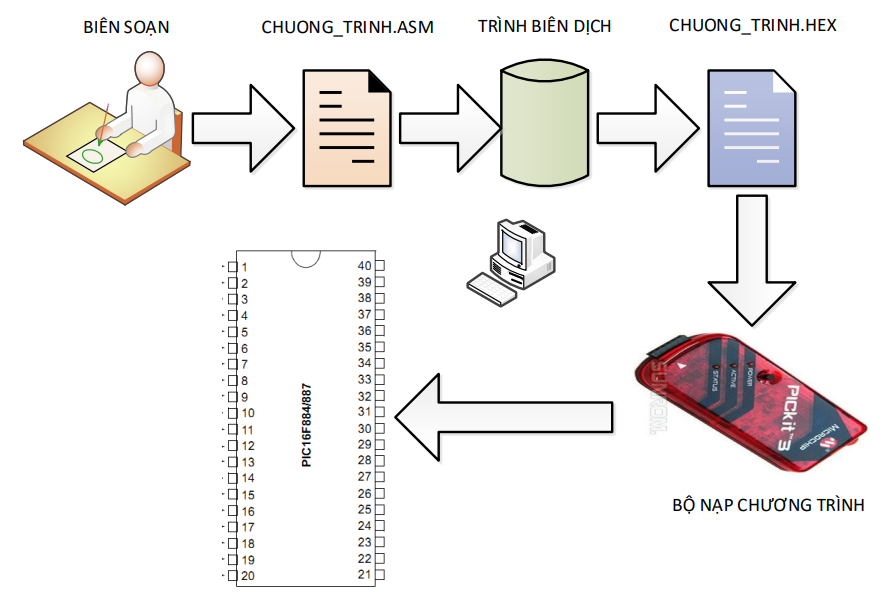
Bất kỳ vi xử lý hay vi điều khiển nào cũng đều có tập lệnh là hợp ngữ còn gọi là **ngôn ngữ Assembly**, và đó là ngôn ngữ cơ bản, đơn giản.

Các vi điều khiển khác họ với nhau thì các lệnh hợp ngữ thường khác nhau.

Khi nghiên cứu bất kỳ một vi xử lý hay một vi điều khiển nào đó thì người ta thường dùng **hợp ngữ** để lập trình cho các ứng dụng để điều khiển.

Tuy nhiên chương trình viết bằng hợp mất nhiều thời gian, chỉ phù hợp cho các ứng dụng điều khiển đơn giản, nếu muốn điều khiển các ứng dụng có nhiều phép toán phức tạp thì không nên dùng hợp ngữ, khi đó bạn nên dùng **ngôn ngữ cấp cao** hơn như **C**.

Người lập trình dùng máy tính để tiến hành biên soạn chương trình **dạng assembly** với tên mở rộng thường phổ biến là **“asm”**, sau khi biên soạn xong sẽ nhờ trình biên dịch sang file hex với phần mở rộng là **“hex”** và cuối cùng là dùng bộ nạp để nạp code vào vi điều khiển.



#### PHẦN MỀM LẬP TRÌNH HỢP NGỮ

Có nhiều cách để biên soạn chương trình bằng hợp ngữ cho vi điều khiển nói chung và vi điều khiển PIC nói riêng, khi bạn dùng vi điều khiển nào thì đều có phần mềm hổ trợ để biên soạn và kèm theo là trình biên dịch luôn.

Phần mềm khá phổ biến viết hợp ngữ cho PIC là MPLAB

MPLAB là phần mềm dùng để lập trình cho vi điều khiển PIC bằng hợp ngữ và sau này có hổ trợ lập trình bằng HI-TECH C.

#### CHƯƠNG TRÌNH HỢP NGỮ CƠ BẢN

**Chương trình** là một tập hợp các lệnh được tổ chức theo một trình tự hợp lí để giải quyết đúng các yêu cầu của người lập trình.

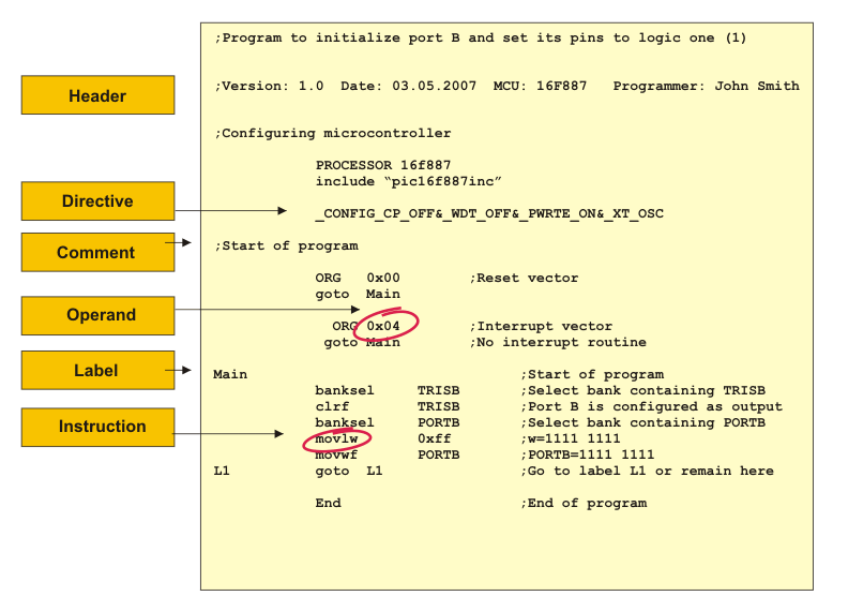
Tất cả các lệnh có thể có của một ngôn ngữ lập trình còn gọi là **tập lệnh**.

**Lệnh** của vi điều khiển là một số nhị phân 8 bit, còn gọi là mã máy hay mã đối tượng.

Nếu một vi điều khiển 8 bit ví dụ như AT80C51 có mã lệnh 8 bit thì có 256 tổ hợp khác nhau từ 0000 0000b đến 11111111b tương ứng và khi đó có thể xây dựng được 256 lệnh khác nhau.

Do mã lệnh dạng số nhị phân quá dài và khó nhớ nên các nhà lập trình đã xây dựng một ngôn ngữ lập trình hợp ngữ còn gọi là Assembly cho dễ nhớ, điều này giúp cho việc lập trình được thực hiện một cách dễ dàng và nhanh chóng cũng như đọc hiểu và gỡ rối chương trình.

Cấu trúc của một chương trình viết bằng hợp ngữ như hình dưới:



Một chương trình viết bằng hợp ngữ gồm có các thành phần như sau:

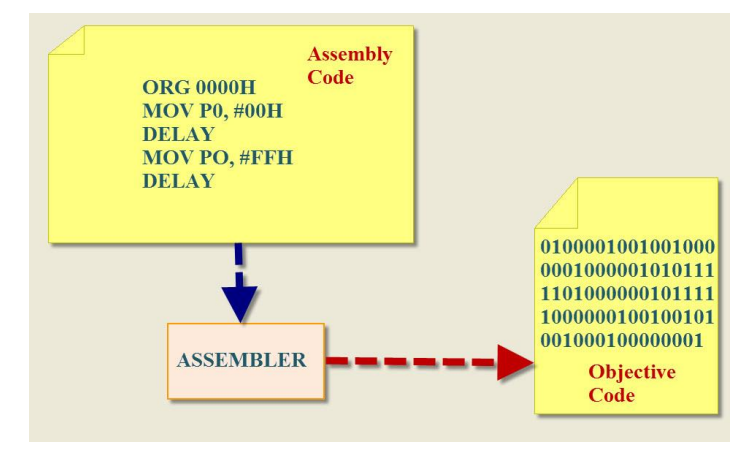
* Tiêu đề (Header) của chương trình để mô tả chương trình viết điều khiển cái gì để nhanh chóng đọc hiểu trở lại cho sau này.
* Chỉ dẫn (Directive) của chương trình để báo cho phần mềm kiểm tra lỗi và biên dịch biết để lấy thông tin khi biên dịch, chỉ dẫn có rất nhiều dạng, khai báo thư viện, định nghĩa biến, ...
* Chú thích (Comment) của chương trình dùng để giải thích rõ hơn cho một lệnh hoặc một địa chỉ hoặc một biến nào đó.
* Toán hạng hay tác tố (Operand) là các đối tượng chứa dữ liệu để lệhn thực thi, ví dụ lệnh cộng thì phải có 2 toán hạn để thực hiện phép toán.
* Nhãn (Label) thay cho các địa chỉ nơi nhảy đến trong chương trình khi thực hiện các lệnh vòng lặp hoặc khi gọi chương trình con.
* Cuối cùng là các lệnh hợp ngữ (Instruction) dùng để viết chương trình điều khiển.

#### TRÌNH BIÊN DỊCH CHO HỢP NGỮ

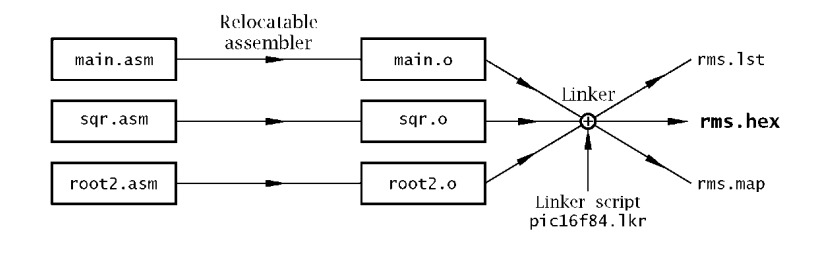
Sau khi viết xong một chương trình dùng các lệnh hợp ngữ thì phải tiến hành biên dịch các lệnh hợp ngữ thành mã máy là các số nhị phân để nạp vào bộ nhớ của vi điều khiển. Trình biên dịch tiếng Anh gọi là Assembler.

Từng chỉ dẫn, từng lệnh, từng nhãn, … được chuyển đổi thành mã nhị phân

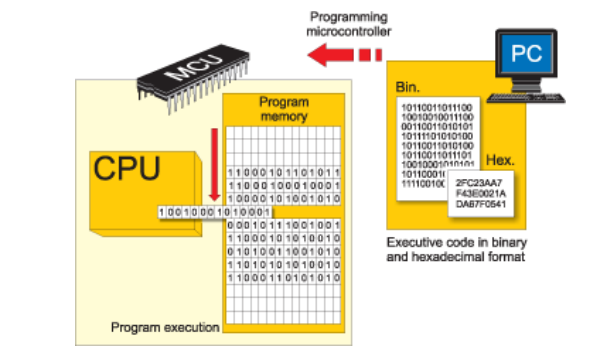
Chương trình hợp ngữ với tên mở rộng là “asm”, sau khi biên dịch thì được nhiều file cùng tên với file gốc như phần mở rộng khác nhau bao gồm 3 file cơ bản với phần mở rộng là “lst”, “hex” và “map”.



Khi bạn viết một ứng dụng điều khiển lớn và phức tạp thì bạn nên chia chương trình thành nhiều file khác nhau và khi biên dịch thì trình biên dịch sẽ biên dịch và liên kết các file đó lại với nhau để tạo ra một file hex duy nhất được thể hiện qua hình



Ngoài file hex dùng để nạp vào bộ nhớ của vi điều khiển để thực thi thì còn một dạng file thứ hai là file “bin” ở định dạng mã nhị phân



Sự khác nhau giữa file “hex” và file “bin” bởi nhiều thành phần nhưng khác nhau cơ bản là file hex chỉ biên dịch đúng các lệnh và vùng nhớ được sử dụng trong chương trình nguồn asm, không chứa mã hex của các vùng nhớ không dùng, khi nạp vào bộ nhớ của vi điều khiển sẽ nhanh.

Còn file “bin” thì biên dịch toàn bộ dung lượng bộ nhớ của vi điều khiển, ví dụ vi điều khiển có 8kword thì file bin sẽ có đầy đủ 8kword cho dù chương trình chỉ sử dụng vài trăm word và phần lớn còn lại chưa dùng đến. Điều này dẫn đến khi nạp code vào bộ nhớ mất nhiều thời gian hơn so với file hex.

Chính vì thê file hex thường được sử dụng rất nhiều đến nỗi người dùng không biết có sự tồn tại của file bin.

### LỆNH HỢP NGỮ CỦA VI ĐIỀU KHIỂN PIC 16F887

#### GIỚI THIỆU

Phần trên đã trình bày về ngôn ngữ lập trình Assembly do con người tạo ra, khi sử dụng ngôn ngữ Assembly để viết thì người lập trình vi điều khiển phải học hết tất cả các lệnh và viết đúng theo qui ước về cú pháp, trình tự sắp xếp dữ liệu để chương trình biên dịch có thể biên dịch đúng.

Ở phần này sẽ trình bày chi tiết về tập lệnh của vi điều khiển giúp bạn hiểu rõ từng lệnh để bạn có thể lập trình được.

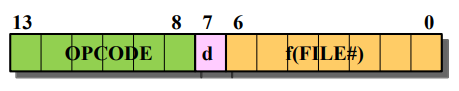
Tập lệnh của PIC 16 được chia ra làm 3 nhóm lệnh:

* Lệnh xử lý bit
* Lệnh xử lý byte
* Lệnh xử lý hằng số và điều khiển

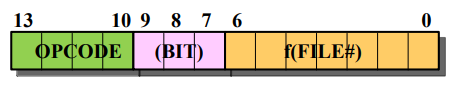
Mỗi lệnh của PIC là một từ dữ liệu 14 bit được chia ra làm 2 nhóm gồm mã lệnh hoạt động (**opcode**: operation code) và tác tố (**operand**).

Mã lệnh Opcode cho biết loại lệnh mà CPU phải thực hiện

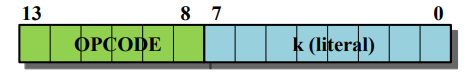
**Lệnh xử lý bit**

****

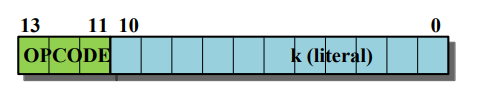
**Lệnh xử lý byte**

****

**Lệnh xử lý hằng số**

****

**Lệnh xử lý điều khiển**

****

Tác tố operand là dữ liệu mà lệnh sẽ xử lý.

|  |  |  |
| --- | --- | --- |
| **STT** | **TÁC TỐ** | **MÔ TẢ** |
| 1 | f | Register file address (0×00 to 0×7F): là địa chỉ 7 bit của file thanh ghi 8 bit, của 1 bank. |
| 2 | W | Working register: Thanh ghi làm việc hay thanh ghi A. |
| 3 | b | Bit address within an 8-bit file register: là địa chỉ của 1 bit nằm trong thanh ghi file 8 bit |
| 4 | k | Literal field, constant data or label: Là hằng số hoặc địa chỉ của nhãn |
| 5 | d | Destination select: lựa chọn nơi lưu dữ liệu: d=0 thì lưu vào W, d=1 thì lưu vào f, mặc nhiên không ghi d trong lệnh thì tương ứng d=1. |
| 6 | PC | Program counter: bộ đếm chương trình. |
| 7 | TO | Time-out bit: bit báo thời gian đã hết. |
| 8 | PD | Power -down bit: bit báo CPU đang làm việc ở chế độ ngủ. |

**Với lệnh xử lý byte**: thì 'f' đại diện cho file thanh ghi và 'd' đại diện cho hướng lưu dữ liệu: nếu 'd' bằng 0 thì dữ liệu sau khi xử lý lưu vào thanh ghi 'W', nếu 'd' bằng 1 thì dữ liệu sau khi xử lý lưu vào thanh ghi 'f'.

**Với lệnh xử lý bit**: thì 'b' đại diện cho bit nằm trong file thanh ghi 'f''.

**Với lệnh xử lý hằng số hoặc điều khiển**: thì 'k' đại diện cho hằng số 8 bit hoặc địa chỉ 11 bit.

Hầu hết các lệnh thực hiện mất 1 chu kỳ lệnh, ngoại trừ lệnh kiểm tra điều kiện đúng sai hoặc lệnh làm thay đổi giá trị của thanh ghi PC thì thực hiện mất 2 chu kỳ máy.

#### KHẢO SÁT TẬP LỆNH TÓM TẮT VI ĐIỀU KHIỂN PIC 16F887

#### TẬP LỆNH CHI TIẾT