Assembly

**1. Cấu trúc chương trình**  
.model small  
.stack 100h  
.data  
.code  
main proc  
// code  
main endp  
end main

2. Phân loại số   
- Số nhị phân (2): Chỉ gồm 0 và 1 , kết thúc bởi b   
- Số thập phân(10) : Một dãy số thập phân kết thúc bở d hoặc D(ko cần thiết)  
- Số hex (16): Bắt đầu bằng 1 số thập phân , kết thúc bằng h hoặc H  
**3. Hàm hiển thị**   
*- Nhập 1 kí tự từ bàn phím :*   
mov ah,1  
int 21h  
-> Kết quả lưu trong al  
*- hiển thị ra 1 kí tự*  
mov ah,2  
mov bl,42h hoặc bl, “3”….  
mov dl,bl  
int 21h  
*- in ra 1 chuỗi*   
nhan db “chuoi$” đặt trên .code  
mov ax, @data  
mov ds,ax  
mov ah,9  
lea dx, nhan  
int 21h  
*- dừng chương trình*  
mov ah,4ch  
int 21h

**4. Add, Sub, Mul ,Div**

*- Add :*  
 add đích , nguồn  
-> đích = đích + nguồn  
Chú ý hàm add chỉ sử dụng vs 2 thanh ghi hoặc 1 thanh ghi với 1 biến chứ ko thể dùng add 2 biên

*- Sub :*  
sub đich , nguồn  
-> đích = đích – nguồn

*- Mul : nhân*mul gốc+ Nếu gốc 8bit -> mặc định số bị nhân là số 8bit để trong thanh ghi al  
Phép nhân được thực hiện : ax <- al \* gốc hoặc hiểu là ax <- ax \* goc  
+ Nếu gốc 16bit ->mặc định số bị nhân là 16 bit để trong thanh ghi ax  
Phép nhân được thực hiện dxax <- ax \* gốc  
Chú ý : ax là thanh ghi 16 bít, trong ax chứa 2 thanh ghi 8 bit là ah và al

*- Div : chia  
div gốc  
+ Nếu gốc là 8 bit thì số bị chia là số 16 bit không dấu để trong ax   
thương để trong al, dư để trong ah  
+ Nếu gốc 16 bit thì số bị chia là 32 bit để trong dxax , thương để trong ax, dư để trong dx***5. Vòng lặp : LOOP**

Mặc định số lần lặp đặt trong Cx , mỗi lần gọi Loop Cx = Cx – 1  
=> Ban đầu phải gán cho Cx = số bất kì nào đó đại diện số lần lặp  
Lặp tới khi cx = 0 thì dừng lại

**6. Nhập và hiển thị số**Assembly chỉ cho phép nhập 1 kí tự nên muốn nhập nhiều kí tự hay nhiều số phải dùng hàm , tổ hợp câu lệnh.  
Chú ý : Dấu : phải đặt ngay sau các nhãn và ko có khoảng trắng nếu ko sẽ sai

**7. Các lệnh cơ bản khác :**

- CMP : Hàm so sánh , áp dụng cùng các hàm nhảy

**Lệnh nhảy :**   
- JA : Nhảy khi lớn hơn -> jna : nhảy khi không lớn hơn  
- JB : Nhảy khi nhỏ hơn -> jnb : nhảy khi không nhỏ hơn  
- JC : Nhảy khi lẻ   
- JE : Nhảy khi bằng  
- JMP : Nhảy không điều kiện

Các lệnh nhảy luôn đi kèm với nhãn   
+ Nhãn là 1 định danh dạng : ‘ tennhan: ‘  
vd nhãn : hienthi:  
+ lệnh nhảy jmp hienthi : quay lại đoạn lệnh có nhãn hiển thị

**Cộng và nhân nhị phân**

- AND : Nhân dưới dạng nhị phân  
vd : mov ax ,15 : 15 = 00000000 00001111  
 and ax, 0011b = 00000000 00000011  
=> kq 1\*0=0, 1\*0 = 0, 1\*1 = 1, 1\*1 = 1 -> 00000000 00000011  
  
- OR : Cộng dươi dạng nhị phân

- CALL : gọi các hàm con  
- LEA dùng trong hàm main

**Ngăn xếp**  
- POP : lấy số ra khỏi ngăn xếp  
- PUSH : đưa 1 số vào ngăn xếp , chỉ sử dụng vs các thanh ghi 16 bit : ax, bx

- XOR : xóa một thanh ghi   
vd XOR ax, ax : xóa ax về 0

- RCL  
- ROL  
- ROR  
- SHL  
- SHR

- ADC : Dùng để đếm có nhớ tương tự cx  
 - MOV : Gán giá trị  
mov đích, nguồn   
-> gán nguồn về đích

8. Tác dụng 1 số thanh ghi  
+ dx : dùng để in ra các kí tự