.model small

.data

a dw 8ac5H

b dw 9dc2H

c dw 0000h

d dw 0000h

.code start:

mov AX,@data mov DS,AX

mov CX,0000h

mov ax,a

mov bx,b

Add ax,bx

jnc GO

inc CX

GO: mov c,ax

mov d,cx

int 21h

end start

Copy code

.model small

This directive specifies the memory model for the program. In this case, it's the small model, which means that code and data are both located within the same segment.

.data

This directive marks the beginning of the data section of the program. This section contains the program's variables and constants.

a dw 8ac5H b dw 9dc2H c dw 0000h d dw 0000h

These four lines define four 16-bit variables 'a', 'b', 'c', and 'd' in the data section. The 'dw' directive specifies that each variable is a word, or 16 bits, in size. The values of 'a' and 'b' are initialized to 8ac5H and 9dc2H, respectively. The values of 'c' and 'd' are initialized to 0000h.

.code start:

This directive marks the beginning of the code section of the program. The 'start' label is defined here, which is the entry point of the program.

mov AX,@data mov DS,AX

These two instructions set up the data segment register (DS) with the address of the data section. The first instruction loads the address of the data section into the AX register, and the second instruction moves the value of AX into the DS register.

mov CX,0000h

This instruction sets the initial value of the CX register to 0. The CX register will be used later to store the carry flag from the addition.

mov ax,a mov bx,b

These two instructions move the values of 'a' and 'b' into the AX and BX registers, respectively. These registers will be used for the addition operation.

Add ax,bx

This instruction adds the values of AX and BX and stores the result in AX. If there is no carry out of the most significant bit of the addition, the program will jump to the 'GO' label. If there is a carry out, the program will increment the CX register.

jnc GO inc CX

These two instructions check whether there was a carry out of the addition operation. If there was no carry out (i.e., the result fits within 16 bits), the program jumps to the 'GO' label. If there was a carry out (i.e., the result requires more than 16 bits), the program increments the CX register.

GO: mov c,ax mov d,cx int 21h

This is the 'GO' label that the program jumps to if there was no carry out of the addition. The first instruction moves the value of AX (which is the sum of 'a' and 'b') into 'c'. The second instruction moves the value of CX into 'd' (which is either 0 or 1, depending on whether there was a carry out of the addition). The last instruction generates an interrupt 21h, which terminates the program.

end start

This directive marks the end of the program. The 'start' label is specified as the entry point of the program.