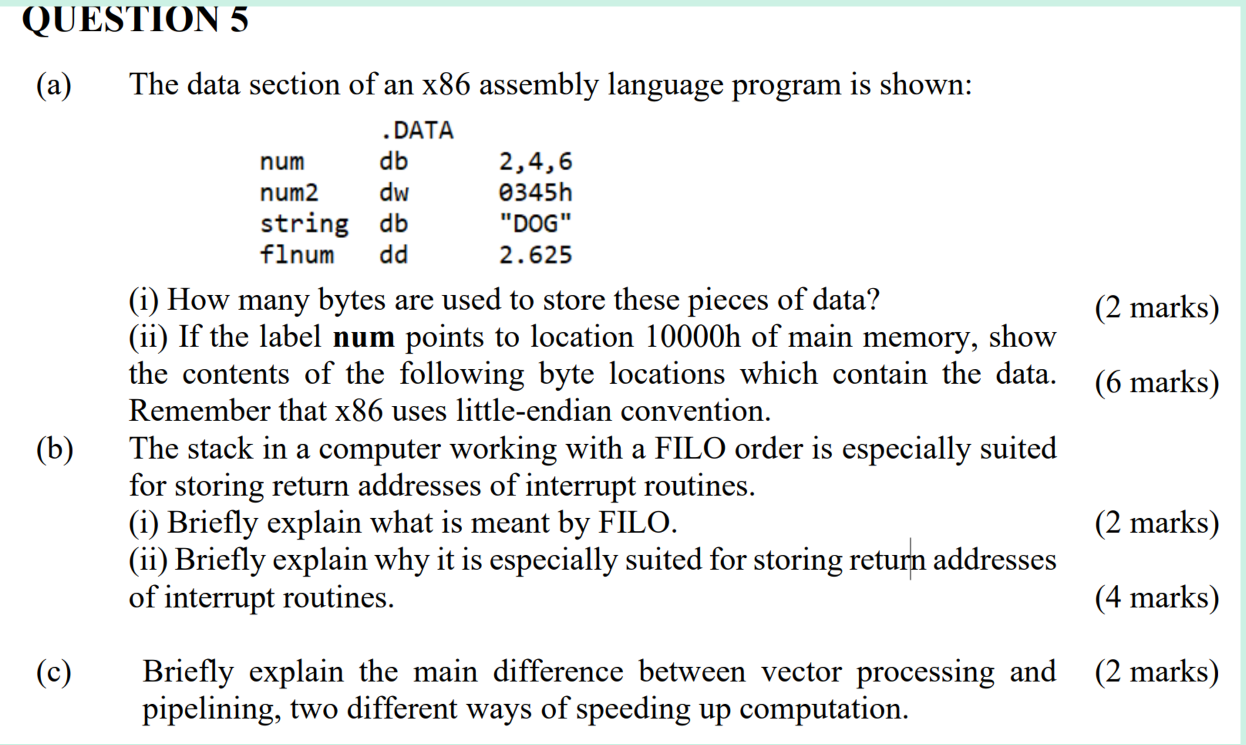
历年卷理科错题整理

x86 数据



QUESTION 5

(a) The data section of an x86 assembly language program is shown:

```
           .DATA
num        db      2,4,6
num2       dw      0345h
string     db      "DOG"
flnum      dd      2.625
```

(i) How many bytes are used to store these pieces of data? (2 marks)

(ii) If the label **num** points to location 10000h of main memory, show the contents of the following byte locations which contain the data. Remember that x86 uses little-endian convention. (6 marks)

(b) The stack in a computer working with a FILO order is especially suited for storing return addresses of interrupt routines.

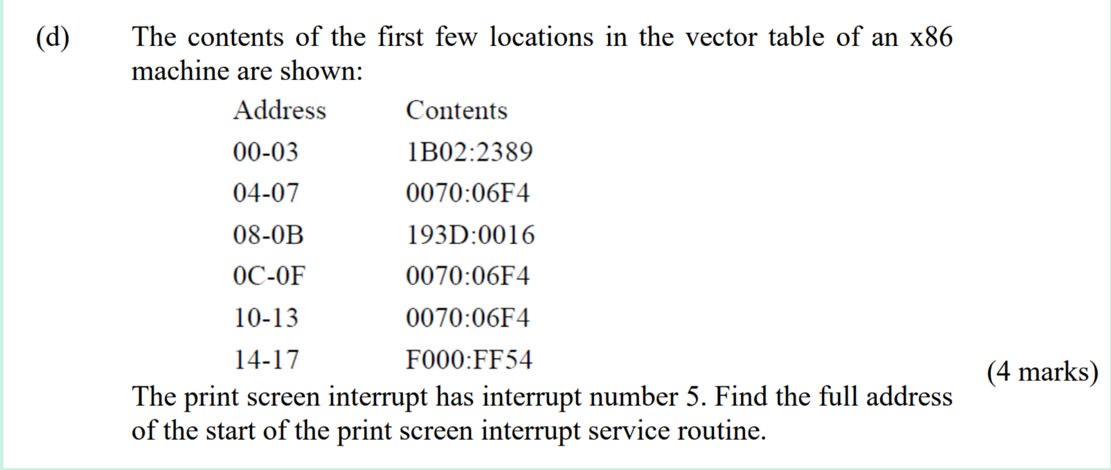
(i) Briefly explain what is meant by FILO. (2 marks)

(ii) Briefly explain why it is especially suited for storing return addresses of interrupt routines. (4 marks)

(c) Briefly explain the main difference between vector processing and pipelining, two different ways of speeding up computation. (2 marks)

24 Question5

- a
  - 1
    - db 每个数字占一个字节
    - dw 两个字节
    - dd 四个字节
    - 字符串也是每个一字节
  - 2
    - 十六进制才关注大端小端，其他的按顺序放，字母和数字都转成 ascll 码
  - 细分主题 2



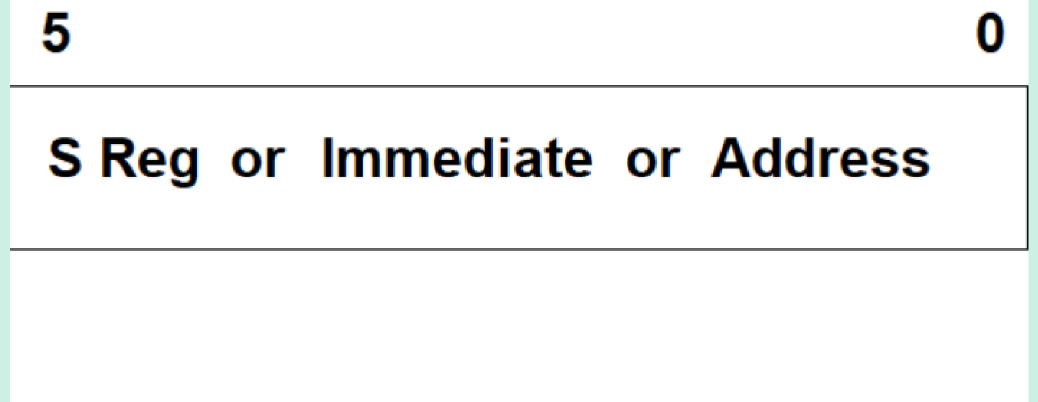
(d) The contents of the first few locations in the vector table of an x86 machine are shown:

Address	Contents
00-03	1B02:2389
04-07	0070:06F4
08-0B	193D:0016
0C-0F	0070:06F4
10-13	0070:06F4
14-17	F000:FF54

The print screen interrupt has interrupt number 5. Find the full address of the start of the print screen interrupt service routine. (4 marks)

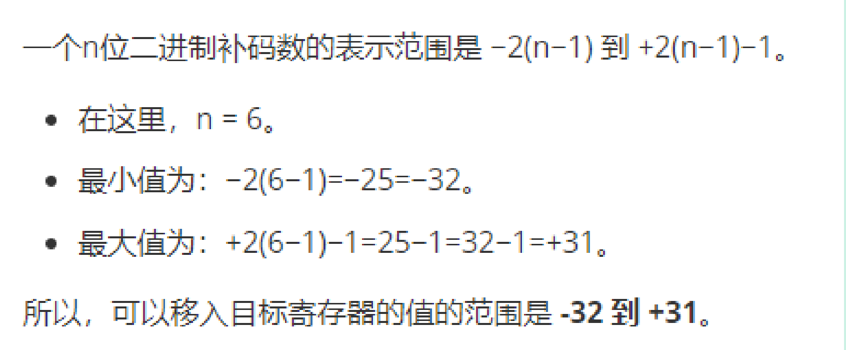
full address of the start

- 从零开始数，所以 '5' 对应第六个
- 然后就是cs*16+ip



What is the range of values that can be moved to the destination register if immediate addressing mode is used?

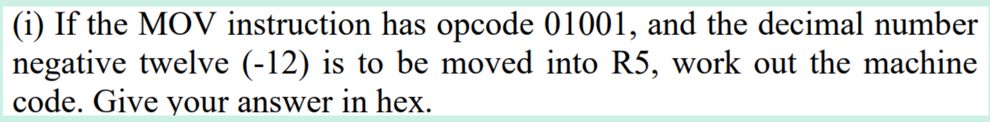
- 如果采用立即寻址，范围就是 2^n 从正到负



一个n位二进制补码数的表示范围是 −2(n−1) 到 +2(n−1)−1。

- 在这里，n = 6。
- 最小值为：-2(6−1)=-25=-32。
- 最大值为：+2(6−1)−1=25−1=32−1=+31。

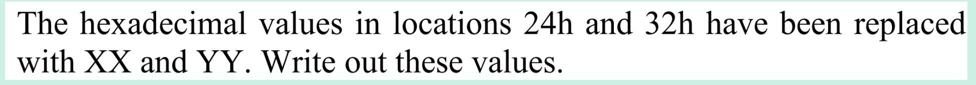
所以，可以移入目标寄存器的值的范围是 **-32 到 +31**。



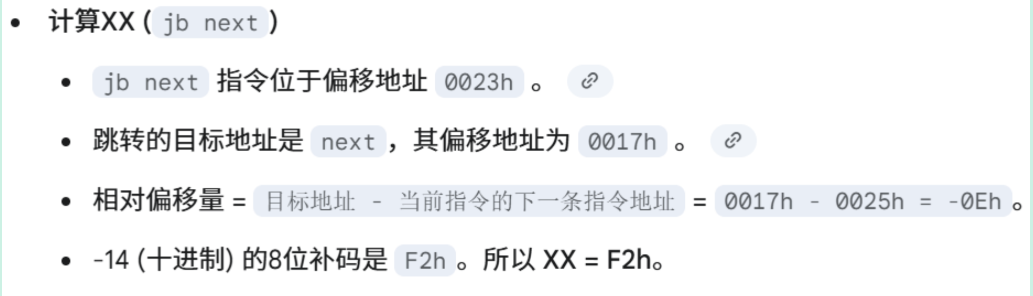
(i) If the MOV instruction has opcode 01001, and the decimal number negative twelve (-12) is to be moved into R5, work out the machine code. Give your answer in hex.

- 如果有负数，直接转换成 2'complement ,再转16进制

- 地址偏移计算



The hexadecimal values in locations 24h and 32h have been replaced with XX and YY. Write out these values.

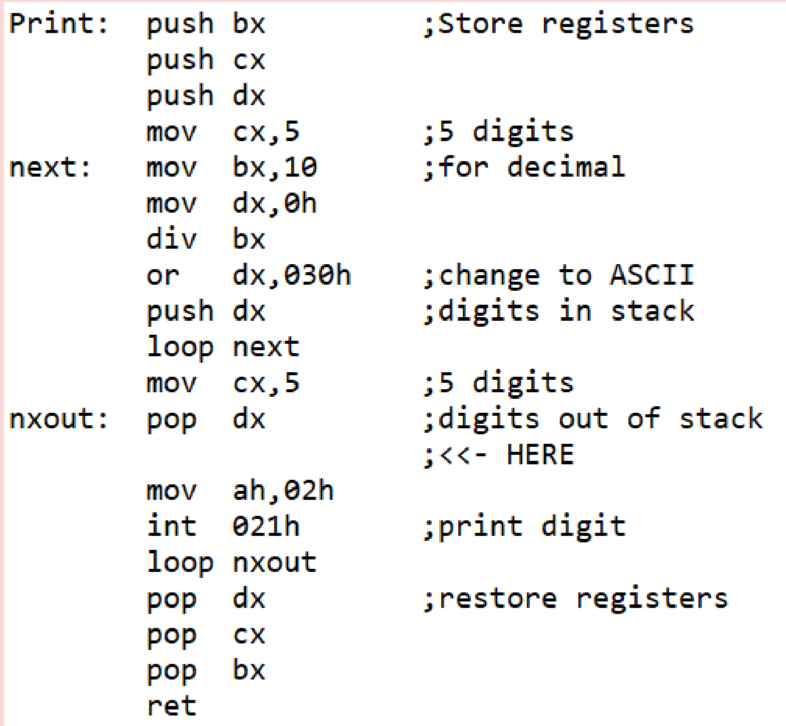


- 计算XX (`jb next`)
  - `jb next` 指令位于偏移地址 `0023h` 。
  - 跳转的目标地址是 `next` ，其偏移地址为 `0017h` 。
  - 相对偏移量 = `目标地址 - 当前指令的下一条指令地址` = `0017h - 0025h = -0Eh` 。
  - -14 (十进制) 的8位补码是 `F2h` 。所以 XX = F2h。

- 先找到起始地址的起始位置，然后加二，然后终点地址减去起点
- 得到结果如果是负数，转换成2 th complement

x86 汇编

- div bx

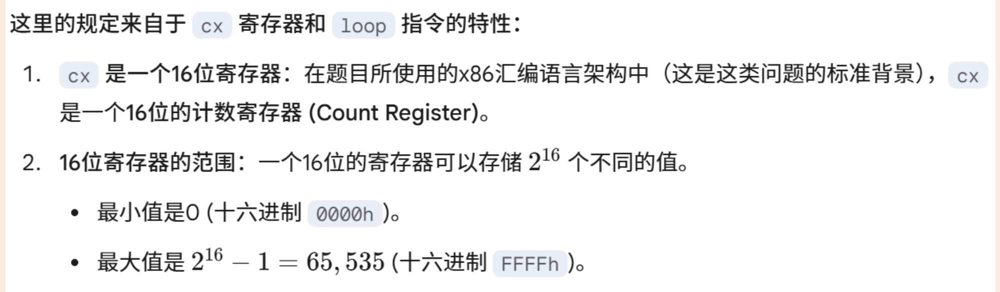


```
Print:  push bx           ;Store registers
        push cx
        push dx
        mov  cx,5         ;5 digits
next:   mov  bx,10        ;for decimal
        mov  dx,0h
        div  bx
        or   dx,030h      ;change to ASCII
        push dx           ;digits in stack
        loop next
        mov  cx,5         ;5 digits
nxout:  pop  dx           ;digits out of stack
                          ;<<- HERE
        mov  ah,02h
        int  021h         ;print digit
        loop nxout
        pop  dx           ;restore registers
        pop  cx
        pop  bx
        ret
```

- 商存在 ax,余数存在 dx
- 循环移位
  - ror 最低位移到最高位
  - rol 最高位移到最低位
  - rcl 最高位到 cf cf 到最低位
- 打印 16 进制的时候 10 与 'A'有 07h 的差距

延迟计算题

- back 要比 nop 少一次
- CPU 的赫兹表示 一秒多少时钟周期

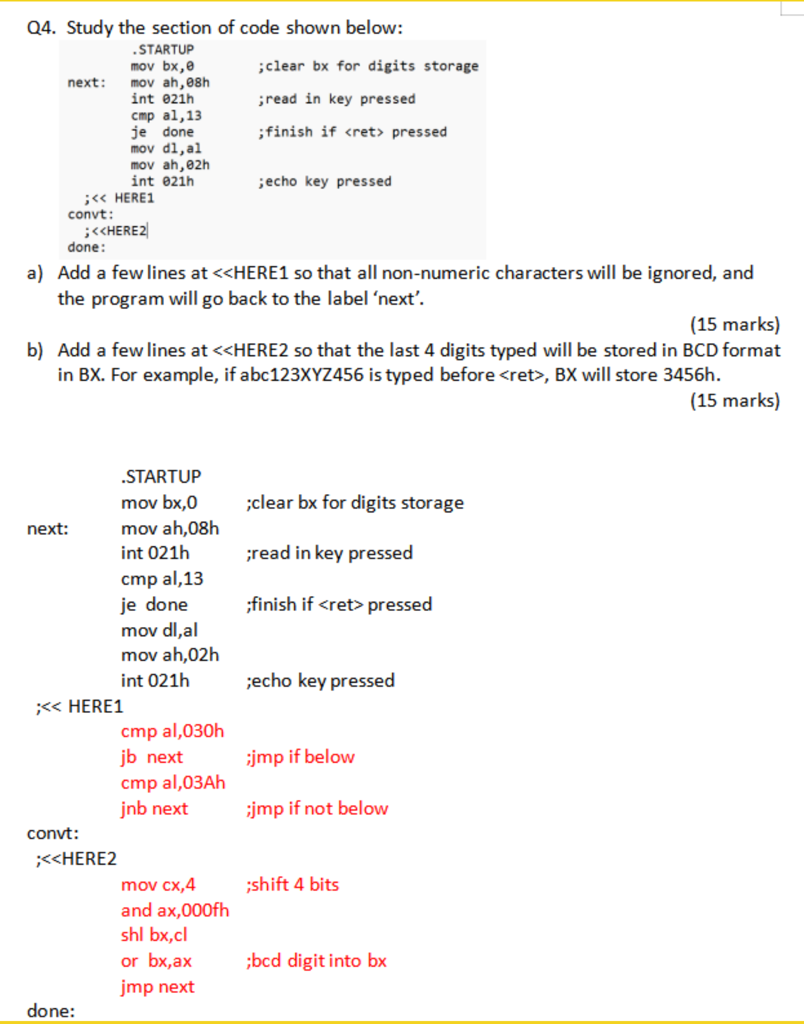


这里的规定来自于 `cx` 寄存器和 `loop` 指令的特性：

1. `cx` 是一个16位寄存器：在题目所使用的x86汇编语言架构中（这是这类问题的标准背景），`cx` 是一个16位的计数寄存器 (Count Register)。
2. 16位寄存器的范围：一个16位的寄存器可以存储 2^{16} 个不同的值。
   - 最小值是0 (十六进制 `0000h`)。
   - 最大值是 $2^{16} - 1 = 65,535$ (十六进制 `FFFFh`)。

一般一个数占一 Byte， 表示十六进制，0000-FFFF

最终



Q4. Study the section of code shown below:

```
        .STARTUP
        mov bx,0        ;clear bx for digits storage
next:   mov ah,08h
        int 021h        ;read in key pressed
        cmp al,13
        je  done        ;finish if <ret> pressed
        mov dl,al
        mov ah,02h
        int 021h        ;echo key pressed
  ;<< HERE1
convt:
  ;<<HERE2
done:
```

a) Add a few lines at <<HERE1 so that all non-numeric characters will be ignored, and the program will go back to the label 'next'.

(15 marks)

b) Add a few lines at <<HERE2 so that the last 4 digits typed will be stored in BCD format in BX. For example, if abc123XYZ456 is typed before <ret>, BX will store 3456h.

(15 marks)

```
        .STARTUP
        mov bx,0        ;clear bx for digits storage
next:   mov ah,08h
        int 021h        ;read in key pressed
        cmp al,13
        je  done        ;finish if <ret> pressed
        mov dl,al
        mov ah,02h
        int 021h        ;echo key pressed
 ;<< HERE1
        cmp al,030h
        jb  next        ;jmp if below
        cmp al,03Ah
        jnb next        ;jmp if not below
convt:
 ;<<HERE2
        mov cx,4        ;shift 4 bits
        and ax,000fh
        shl bx,cl
        or  bx,ax       ;bcd digit into bx
        jmp next
done:
```

最终