作业参考答案

1.2 汇编源程序是指用汇编语言编写的程序，而汇编程序特指将汇编源程序汇编成目标文件的编译程序。

1.4

（1）1000011,43　　（2）100010， 22

（3）1111 1101，FE　　（4）111 1011，7B

1.6 （1） BE　　（2）3D　（3） 89　（4） B41

1.7 （1）0100 0000　（2）[10011000]原=[1110 1000]补

2.2

（1）数据寄存器：Ax，Bx，Cx，Dx； AX：作为累加器，是算术运算的主要寄存器。在乘除等指令中存放操作数，在 I/O 指令中使用它与外部设备传送信息。BX：当通用寄存器使用，在计算存储器地址时，作基址寄存器使用。CX：当通用寄存器使用，此外常用来保存计数值，当计数器使用。

DX：当通用寄存器使用，一般在作双字长运算时把 DX 和 AX 组合使用，对某些 I/O 操作，DX 用来存放 I/O 的端口地址。

（2）地址寄存器：Sp，Bp，Si，Di

（3）段寄存器：Cs，Ds，Es，Ss；段寄存器的作用是专用于存储器寻址，用来直接或间接地存放段地址。

（4）专用寄存器：Ip，Flags；Ip 寄存器专门存放下一条指令的地址，Flags 标志寄存器，又称程序状态寄存器。它是存放条件码标志、控制标志和系统标志的寄存器。

2.3

1234:2002=12340+2002=14342

1430:0042=14300+0042=14342

FF00:0FFF=FF000+0FFF=FFFFF

2.4

85=55H,69=45H,-69=BBH，

8 位二进制补码运算:

(1)85+69 =55H+45H=9AH=154, CF=0,OF=1

(2)85+(-69)=55H+BBH=10H=16,CF=1,OF=0

(3)85-(-69)=55H-BBH=9AH=154,CF=1,OF=1

(4)85-(69)=55H-45H=10H=16,CF=0,OF=0

16 位二进制补码运算:

85=0055H,69=0045H,-69=0FFBBH，

(1)85+69 =0055H+0045H=009AH=154, CF=0,OF=0

(2)85+(-69)=0055H+0FFBBH=0010H=16,CF=1,OF=0

(3)85-(-69)=0055H-0FFBBH=009AH=154,CF=1,OF=0

(4)85-(69)=0055H-0045H=0010H=16,CF=0,OF=0

2.5

00010+0000=00010H

00010+FFFF=1000FH

2.6

20000-0000H=20000H

20000-FFF0H=10010H

2.7

52506=42510+FFF6=52500+0006,基地址最大为 5250H，最小为 4251.

段地址取值范围：0000—FFFF。即 65536 个。

3.1

(1)用编辑程序 EDIT 建立 .ASM 源文件

(2)用汇编程序 MASM 把.ASM 文件原文件汇编成.OBJ 文件

(3)用连接程序 LINK 将.OBJ 文件转换成.EXE 文件

(4)在 DOS 下直接运行.EXE 文件或在 DEBUG 下调试该.EXE 文件

3.3

程序的结束：mov ah,4ch

int 21h

程序的结束和指出执行的起点：End start

3.4 源程序的文件扩展名为*.asm，改扩展名不可以执行。

3.7

- D 显示内存内容
- E 修改内存单元内容
- T 跟踪命令
- G 运行命令
- A 汇编命令
- R 查看或修改寄存器内容

3.8

(1) U

(2) 1234H

(3) 12440H

(4) R BX

(5) A [0104]

3.10

(1)将调用功能的功能号存入 AH 寄存器。

(2)如必要，设置该调用功能的入口参数。

(3)执行 INT 21H 指令。

(4)如必要，按规定取得出口参数（返回参数）。

3.12

```
mov ax,145B
mov   ds,ax
mov ah,09
mov dx,0
int 21h
```

4.1

例如 DS=6542H，指令 mov ax,DS:[123A]； 123A 为有效地址，6542H 为段地址，65420H+0123AH=6665A 即物理地址.

4.2

(1)立即数寻址

(2)寄存器间接寻址

(3)变址寻址（或寄存器间接寻址）

(4)寄存器寻址

(5)基址变址寻址

(6)相对基址寻址（或寄存器相对寻址）

(7)直接寻址

(8)相对基址变址寻址

4.3

(1)操作数在指令中无 EA

(2)直接寻址：EA=4524H，物理地址=DS：4524

(3)使用 BX 的寄存器寻址：无 EA

(4)使用 BX 的间接寻址：EA=463DH，物理地址=DS：463D

(5)使用 BP 的寄存器相对寻址：MOV AX，[BP+4524]，EA=2006+4524，物理地址=SS：EA

(6)基址变址寻址： MOV AX，[BX+SI]，EA= BX+SI，物理地址=DS：EA

(7)相对基址变址寻址：MOV AX，[4524+BX+SI]，EA=4524+BX+SI，

物理地址=DS：EA

4.4

(1)直接寻址

MOV　AX，ARRAY+8 或 MOV　AX，[ARRAY+8]

(2)使用 BX 的间接寻址

LEA　BX，ARRAY+8 ；MOV　AX，[BX]

(3)使用 BX 的寄存器相对寻址

LEA　BX，ARRAY　；MOV　AX，[BX+8]

(4)基址变址寻址

LEA　BX，ARRAY ；　MOV　SI，8 ；　MOV　AX，[BX+SI]

5.1

(1) MOV　　AX，[BX]

(2) MOV　　DS，BX

(3) MOV　　ES，AX

(4) MOV　　AL，DL

(5) PUSH　　AX

(6) ADD　　[BX]，DI

(7) LEA　　BX，V

(8) MOV　　DX,OFFSET V

(9) MOV　　WORD PTR [SI]，AX

(10) MUL　　BX

(11) DIV　　BX

(12) MOV　　BYTE PTR [SI]，2

(13) MOV　　AX，[BX+SI]

(14) SHR　　AX，1

(15) CMP　　AX，6

(16) MOV　　[FFFE]，AX

(17) MOV　　AX，[BX+4]

(18) JMP　　FAR PTR PRO

5.2

处理器对两个操作数进行运算时，按照无符号数求得结果，并相应设置进位标志 CF；同时，根据是否超出有符号数的范围设置溢出标志 OF。应该利用哪个标志，则由程序员来决定。也就是说，如果将参加运算的操作数认为是无符号数，就应该关心进位；认为是有符号数，则要注意是否溢出。

5.3

	SF、ZF、CF、OF			
(1)1234H+7450H，	1	0	0	1
(2)5678H+7450H，	1	0	0	1
(3)9804H+7450H，	0	0	1	0
(4)E0A0H+7450H，	0	0	1	0

5.5

JB　L1

JMP　L2

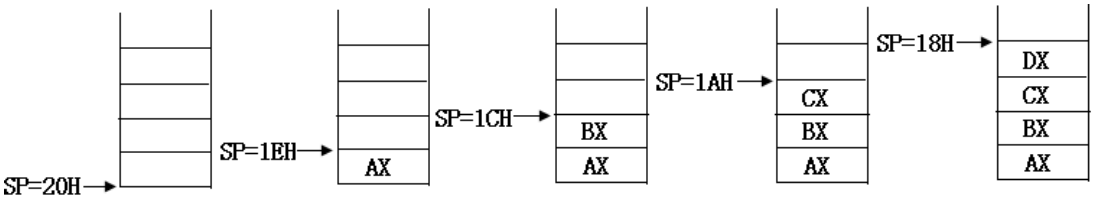
5.7

14 00 1E 00 28 00 20 00 30 00 FA FF 61 62 63 64 $

(BX)=1E00H，(DI)=0000H，(CX)=2000H，(DX)=0064H

5.8

```
code segment
        assume cs:code
start:
          Push AX
          push BX
          push CX
          Push DX
          Pop　SI
          Pop　DI
          Pop　BP
          Pop　BX
Code ends
        End start
```



5.11

```
MOV　DX,0
MOV　BX，AX
SHL　AX，1
ADC　DX,0
SHL　DX,1
SHL　AX，1
 ADC　DX,0
 ADD　AX，BX
 ADC　DX,0
```

5.12

(1)

```
MOV  AX，DATA
MOV  DS，AX
MOV  AL，‘*’
LEA  DI，STR1
MOV  CX，STR1-BUFF
CLD
REP  STOSB
```

(2)

```
MOV  AX，DATA
MOV  DS，AX
MOV  ES，AX
CLD
LEA  SI，BUFF
LEA  DI，STR1
MOV  CX，STR1-BUFF
REP  MOVSB
```

(3)

```
MOV  AX，DATA
MOV  DS，AX
MOV  ES，AX
STD
LEA  SI，STR1-1
LEA  DI，LEN-1
MOV  CX，STR1-BUFF
REP  MOVSB
```

(4)

```
MOV  AX，DATA
MOV  DS，AX
MOV  ES，AX
CLD
LEA  SI，BUFF
LEA  DI，STR1
MOV  CX，STR1-BUFF
REPE  CMPSB
```

(5)

```
MOV  AX，DATA
MOV  ES，AX
MOV  BX，0
CLD
MOV  AL，‘$’
LEA  SI，BUFF
MOV  CX，STR1-BUFF
NEXT: REPNE  SCASB
JCXZ  NO-FOUND
INC  BX
JMP  NEXT
```

5.13

L1，L1，L5

6.1

```
MOV  AL，ARRAY+2          （AL）=（ 09    ）H
ADD  AL，DA2+1            （AL）=（ 4A    ）H
MOV  AX，DA2-DA1          （AX）=（ 000c  ）H
MOV  BL，LEN              （BL）=（  09   ）H
```

MOV　AX, DA3　　（AX）=（ 0102 ）H

MOV　BX, TYPE　DA4　　（BX）=（ 0001 ）H

MOV　BX, OFFSET　DA4　　（BX）=（ 0102 ）H

MOV　CX, SIZE　DA4　　（CX）=（ 0004 ）H

MOV　DX, LENGTH　DA4　　（DX）=（ 0004 ）H

MOV　BX, WORD PTR　DA4　　（BX）=（ 0201 ）H

MOV　BL, LEN　AND　0FH　　（BL）=（ 09 ）H

MOV　BL, LEN　GT　5　　（BL）=（ ff ）H

MOV　AX, LEN　MOD 5　　（AX）=（ 0004 ）H

6.2

变量是为指令提供的操作数，标号是为指令提供标识，都是为了在指令中引用。它们最主要的属性有：偏移属性，段属性，类型属性。例如：

MOV　BX，OFFSET　VAL ；取偏移属性

MOV　BX，SEG　VAL　；取段属性

MOV　BX，TYPE　VAL　；取类型属性

6.3

指令只能出现在代码段，定义数据的伪指令通常在数据段，伪指令在代码段两端也可，但不能在指令之间。

6.6

MOV　AL，ARRAY+2　　（AL）=（ 09 ）H

ADD　AL，DA2+1　　（AL）=（ 4A ）H

MOV　AX，DA2-DA1　　（AX）=（ 0008 ）H

MOV　AX，DA1+1　　（AX）=（ 0900 ）H

MOV　BL，LEN　　（BL）=（ 10 ）H

6.7

```
data segment
    Array db ‘inspire a generation!’
    Data1 df  0fedcbah
    Data2 db 10101010B
    Data3 db 100 dup(0)
    dw 500 dup(?)
data ends
```

6.11

（1）0020h　　（2）0008h

7.1

（1）jl　l1

（2）xchg al,cl

7.6

```
code segment
        assume cs:code
main    proc    far
    push ds
    sub ax,ax
    push ax
    mov ax,1
a:
    mov cx,ax
b:
    mov bx, ax
    sub ax,ax
    mov ah,2
    mov dl, '*'
    int 21h
    mov ax, bx
    loop    b
    cmp ax,6
    jz exit
    inc ax
    mov bx, ax
    sub ax,ax
    mov ah,2
    mov dl,13
    int 21h
    mov ah,2
    mov dl,10
    int 21h
    mov ax, bx
    jmp a
exit:
    ret
code ends
end main
```