1A

F1　执行机器指令、暂存少量数据和访问存储器

F2　0-255，-128~+127

F3　2

F4　57H 6FH 72H 6CH 64H

F5　I　69H

F6　F7H　FFF7H

F7　-13

F8　 99 c

F9　33H 22H 2233H 1122H

S1 ABD

S2 CD

S3 A

S4 B

S5 C

Q1

（1）与机器语言相比，汇编语言具有较好的可读性，但是功能和机器语言是一致的。

（2）与高级语言相比有以下特点：汇编语言与机器关系密切

汇编语言程序效率高

编写汇编语言源程序繁琐

汇编语言程序调试困难

Q2 略

2B

F1　实方式 保护方式　保护方式 虚拟 8086

F2　1MB 4GB

F3　8 8 8 8

F4　AH、AL、BH、BL、CH、CL、DH、DL

F5　0AH　000AH　1234000AH

F6　0DCH　00DCH　000000DCH

F7　0　FFFF0000H

S1 C

S2 BC

S3 DFJ

Q1 略

R1

MOV EAX, 89ABCh	00089ABCh
MOV AX, 1234h	00081234h
MOV AL, 98h	00081298h
MOV AH, 76h	00087698h
ADD AL, 81h	00087619h
SUB AL, 35h	000876E4h
ADD AL, AH	0008765Ah
ADC AH, AL	0008D15Ah
ADD AX, 0D2h	0008D22Ch
SUB AX, 0FFh	0008D12Dh
ADD EAX, 4567h	00091694h
SUB EAX, 7654h	0008A040h
INC AL	0008A041h
DEC AX	0008A040h
NEG EAX	FFF75FC0h

R2 00A0H

R3 b

2C

[F1] 进位标志 CF，零标志 ZF，符号标志 SF，溢出标志 OF　　//次序可变

[F2] 6，CS，DS，SS，ES、FS、GS

[F3] 0，0，1，0

[F4] 1，0，1，0

[F5]立即寻址，寄存器寻址；直接寻址，寄存器间接，寄存器相对，基址加变址，相对基址加变址　　//类似表述也可

[F6]相对基址加变址，寄存器寻址　//类似表述也可

[F7]立即寻址，基址加变址　　//类似表述也可

[F8]56H，00H

[F9]LEA　ECX, [EBX+EDX+256]　　//LEA　ECX, [EDX+EBX+256]

[F10]LEA　ECX, [ECX+2*ECX+88]

[S1] AD

[S2] D

[S3] BEF

[S4] AB

[S5]BD

[S6]AB

[S7]BEFHI

[R1]

```
MOV     AL, 89H
ADD     AL, AL              ;OF/SF/ZF/CF/AF/PF =1/0/0/1/1/1
ADD     AL, 9DH             ;OF/SF/ZF/CF/AF/PF =0/1/0/0/0/1
CMP     AL, 0BCH            ;OF/SF/ZF/CF/AF/PF =0/1/0/1/0/1
SUB     AL, AL              ;OF/SF/ZF/CF/AF/PF =0/0/1/0/0/1
DEC     AL                  ;OF/SF/ZF/CF/AF/PF =0/1/0/0/1/1
INC     AL                  ;OF/SF/ZF/CF/AF/PF =0/0/1/0/1/1
NEG     AL                  ;OF/SF/ZF/CF/AF/PF =0/0/1/0/0/1
```

[R2]

```
MOV     EDX, 87654321H
MOV     EAX, 89ABCDEFH
ADD     DL, AL              ;CF/ZF/SF/OF =1/0/0/0
SUB     DH, AH              ;CF/ZF/SF/OF =1/0/1/0
ADC     AX, DX              ;CF/ZF/SF/OF =1/0/0/0
SBB     EDX, EAX            ;CF/ZF/SF/OF =1/0/1/0
CMC                         ;CF/ZF/SF/OF =0/0/1/0
ADC     DX, AX              ;CF/ZF/SF/OF =0/0/0/0
SUB     EAX, EDX            ;CF/ZF/SF/OF =1/0/1/0
SBB     AL, DH              ;CF/ZF/SF/OF =0/0/1/0
```

[R3]答案并不唯一，只要满足要求就行。

[R4]直接寻址，立即寻址，寄存器寻址，寄存器间接，寄存器相对寻址

[R5]122FH，1244H，1354H，91A0H，2004H

[R6]1237H，800H，500H，2563H，5678H

[R7] （1）A=10,B=100,C=768

（2）ESI+ECX*2+5 等于 7，所以从 buff 的第七位开始取 16 位数据，高 8 位（buff 的第八个字节）是 03H，低 8 位（buff 的第九个字节）是 00H，合起来就是 0300H 即 768。

[G1] 略

2D

[F1]PUSHAD，POPA

[S1]ACD

[S2]ABC

[S3]ABC

[S4]CD

[S5]ABE

[S6]AE

[R1]

ESP=00000FFCH

ESP=00000FFAH

ESP=00000FFEH　　,EBX=43214321H

ESP=00001000H　　,EBX=43218765H

示意图略

[R2] 略

3E

[F1] 寄存器、堆栈、EAX

[F2] 寄存器、堆栈

[F3] POP、PUSH、POPA、PUSHA、POPAD、PUSHAD、CALL、RET

[S1]ABCDF

[S2]A

[S3]D

[S4]ABC

[S5]ABC

[S6]BC

[S7]CD

[R1]

(1)建立堆栈框架

(2) unsigned sum;或者定义变量 sum

(3) a1*n

(4) n-1

(5) n*(n-1)/2

(6) return sum;

[Q1] 略

[Q2] 答案并不唯一，只要满足要求就行。

3F

[F1]75H, 75H, 75H

[F2]0FF85H　　//0 可省略

MOVSX BX,AL

MOVSX EBX,AL　　//或 MOVSX EBX,BX

[F3]83838383H

MOVZX ECX,AX

MOVZX ECX,AL　　//或 XOR CH,CH

[S1]CGIJ

[S2]ABD

[S3]CD

[R1]

指令	AL 值	ZF	SF	PF
AND AL, 0FH	05H	0	0	1
OR AL, 0C3H	C7H	0	1	0
TEST AL, 88H	C7H	0	1	0
XOR AL, AL	00H	1	0	1

[R2]

指令	AL 值	CF	ZF	SF	PF
SAR AL,1	C2H	0	0	1	0
SHR AL,1	61H	0	0	0	0
ROR AL,1	B0H	1	0	1	0
RCL AL,1	61H	1	0	0	0
SHL AL,1	C2H	0	0	1	0
ROL AL,1	85H	1	0	1	0

[R3]将作为入口参数的 8 位字符代码复制为 32 位。

[Q1]可以多个答案

```
SUB   AL,AL
XOR   AL,AL
AND   AL,0
SHL   AL,8
```

[Q2]可以多个答案

```
SUB   BL,1
XOR   BL,3
SHR   BL,1
ROL   BL,7
```

[Q3]

（1）反映（无符号数）算术运算的进位或者借位；

（2）在移位操作时，过渡；

（3）函数的出入口参数。

可以有多种方法清 CF 标志：

```
CLC    //    XOR   AL,AL    //    ADD   AL,0     //    SUB   AL,0
```

[Q4]略

[Q5]略

作业 3G

[F1]-126---129　　//-128---127（如果以下一条指令为基准）

　　-126---129　　//-128---127（如果以下一条指令为基准）

[F2]段内转移，段间转移　　//可以互换

[F3]直接转移，间接转移　　//可以互换

[S1]ABCD

[S2]AC

[S3]AB

[R1]显示用户所敲键的 ASCII 码中二进制位是 1 的位数。

[R2]略

[Q1]略

作业 3H

[F1]ECX，CX

[F2]-128—127

[F3]DEC ECX/CX 和 JNZ 指令

[F4]JECXZ/JCXZ

[S1]ABD

[S2]ABCD　　//关于 ABC 可以讨论

[S3]BD

[Q1]参考：循环指令基于默认的计数器，计数器循环一次自动减 1，且同时可以通过判断 ZF 来终止循环，因此在很多情况下可以简化实现循环的代码。

但也正因此与条件转移指令方法实现循环相比，其灵活性差一些，循环条件和计数器不能随意修改。　//可以画流程图说明之

[G1]

```
SUBR1:
        PUSH   EBP
        MOV    EBP,ESP
        PUSH   EBX
        MOV    EBX, [EBP+8]
NEXT:
        MOV    AL, [EBX]
        CMP    AL,0
        JZ     OVER
        CMP    AL,'0'
        JB     CONTINUE
        CMP    AL,'9'
```

```
        JA    CONTI
        MOV   BYTE PTR [EBX],0x20
    CONTINUE:
        INC   EBX
        JMP   NEXT
    OVER:
        POP   EBX
        POP   EBP
        RET  //可以上机调试
```

3I

[F1] 寄存器 堆栈

[F2] 寄存器 堆栈

[F3] 参数传递、安排局部变量、保护寄存器

[F4] 段内转移 段间转移

[F5] 直接转移，间接转移

[F6] 段内返回 段间返回

[F7] 主程序平衡 子程序平衡

[S1] A

[S2] AB

[S3] D

[Q1] 参考书例 2，例 3

[Q2]（略）//可以从堆栈作用来说明堆栈必须保持平衡。

[Q3]（略）

[R1]（略）

[R2]SUB1:功能：参数值加 7；寄存器传递入口参数。

SUB2:功能：累加求和；堆栈传递入口参数；主程序负责平衡堆栈。

SUB3:功能：参数值加 8；堆栈传递入口参数；子程序负责平衡堆栈。

运行结果可以上机调试。

作业 4J 答案

[F1]MOVSx，STOSx，SCASx， //次序可变

[F2] 8 位，16 位，32 位 //字节，字，双字

[F3]DF，CLD、STD

[F4]DS:ESI、ES:EDI

[F5]ECX //

[F6]MOVS，CMPS

[Q1] //主要考虑从时空效率和对标志影响等方面进行比较。例如功能相同，片段需要 2 条指令，LODSB 只需要一条，效率更高等。

[Q2] //主要考虑从时空效率和对标志影响等方面进行比较。例如功能相同,片段需要 2 条指令，STSOW 只需要一条，效率更高等。

[Q3]略

[G1]

```
#include  <stdio.h>
char  aeiou[] = "AEIOU";
int main()
{   int   count;
    char  buffer[128];
    printf("请输入一个字符串：");   scanf("%s",buffer);
    _asm {
        CLD
        LEA   ESI, buffer
        XOR   EDX, EDX
        SUB   EBX, EBX
   LAB1:LODSB
        OR    AL, AL
        JZ    LAB3
        CALL  ISaeiou        //判断是否元音字母
        JNZ   LAB2
        INC   EDX
   LAB2:JMP   LAB1
   LAB3:MOV   count, EDX
    }
    printf("该字符串中含元音字母数=%d\n",count);
    return  0;
    _asm  {
    ISaeiou:
        LEA   EDI, aeiou
        MOV   ECX, 5
        AND   AL,11011111B          //大小写一致化(转大写)
        REPNZ   SCASB
        RET
    }
}
```

作业 5L

[Q1] 避免时钟数多的指令、减少转移指令、减少循环执行次数、存储器地址对齐等

[Q2] 采用寄存器作为变量、采用长度较短的指令或者指令片段等方法

作业 6M

[F1]179B8；答案不唯一（2345:0006H 便是其中之一，只要逻辑地址对应的物理地址是 23456H）
[F2]64K，16
[F3]1000H，0235：FFF5H，1234：0005H
[F4]6
[F5]DS，ES
[F6]指令语句、伪指令语句、宏指令语句

[S1]CF
[S2]AD
[S3]ACD
[S4]ABCD

[Q1] 物理地址 = 段值×16 ＋ 偏移（逻辑地址左移四位加偏移地址就是物理地址）
[Q2] 主要从基址、变址可以使用的寄存器、变址是否可以带系数、位移量长度进行比较。
[Q3] 当有效地址使用 BP 时，默认使用 SS 段。因为用到 BP 时，基本与堆栈操作有关。

[R1]EDI=87655693H
    ECX=0000BD0EH
[R2]把物理地址 0FFFF0H（逻辑地址 0F000:FFF0H）开始处的 16 个字节，送到物理地址 21234H（逻辑地址 2000:1234H）开始处的区域。