第4章 指令系统 习题与解答

> 来源：《第四章 指令系统.ppt》（extracted/ch4，173 页）。课件原题 49 道全部收录（含全部编号例题与散例），另附 4 道考点补充题。所有执行结果均逐步验算。

一、课件原题

【寻址方式】

题1（课件 p.7，例4-1）

说明指令 `MOV AX，080AH` 的寻址方式与执行过程，执行后 AH、AL 各为多少?

解答： 源操作数 080AH 直接包含在指令中，为**立即寻址**；目的操作数 AX 为寄存器寻址。 立即数放在操作码之后，低字节在低地址：代码序列为「操作码、0AH（低字节）、08H（高字节）」。 取指后：低字节 0AH→AL，高字节 08H→AH，即 **AX=080AH（AH=08H，AL=0AH）**。 立即寻址不执行总线周期（操作数随指令一起取出），且只能作源操作数（课件 p.8）。

题2（课件 p.9，例4-2）

说明 `MOV AX，BX` 的寻址方式与执行过程。

解答： 源、目的操作数都在 CPU 寄存器中，均为**寄存器寻址**。执行时把 BX 的内容直接送 AX（CPU 内部完成），**不执行总线周期**，速度快。寄存器寻址既可用于源也可用于目的，且可同时使用（课件 p.10）。若 BX=1234H，则执行后 AX=1234H。

题3（课件 p.11，例4-3）

设 DS=3000H，说明 `MOV AX，[22A0H]` 的寻址方式，求操作数物理地址及执行后 AX 的值（已知 (322A0H)=32H，(322A1H)=48H）。

解答： EA 直接在指令中给出，为**直接寻址**，默认段 DS。 PA = DS×16 + EA = 30000H + 22A0H = **322A0H**。字操作取两个连续字节：AL←(322A0H)=32H，AH←(322A1H)=48H。 执行后 **AX=4832H**。

题4（课件 p.13，例4-5）

设 DS=2000H，SI=1000H，说明 `MOV AX，[SI]` 的寻址方式，求物理地址及执行后 AX（已知 (21000H)=A0H，(21001H)=50H）。

解答： EA 在间址寄存器 SI 中，为**寄存器间接寻址**；SI 间址默认段 DS。 PA = DS×16 + SI = 20000H + 1000H = **21000H**。 AL←(21000H)=A0H，AH←(21001H)=50H → **AX=50A0H**。

题5（课件 p.14，例4-6）

设 SS=3000H，BP=2000H，说明 `MOV AX，[BP]` 的寻址方式与默认段，求物理地址及执行后 AX（已知 (32000H)=B0H，(32001H)=62H）。

解答： **寄存器间接寻址**；以 BP 间址时默认段为**堆栈段 SS**（不是 DS）。 PA = SS×16 + BP = 30000H + 2000H = **32000H**。 AL←(32000H)=B0H，AH←(32001H)=62H → **AX=62B0H**。

题6（课件 p.15）

已知 DS=2100H，DI=2000H，指令 `MOV AX，[DI]`。求物理地址并说明执行结果。

解答： 寄存器间接寻址，DI 默认 DS。 PA = 16×DS + DI = 21000H + 2000H = **23000H**。 执行结果：将 23000H 和 23001H 两单元的内容送入 AX（低地址字节→AL，高地址字节→AH）。

题7（课件 p.16，例4-7）

按图4-6：SS=1000H，BP=2000H，COUNT=2100H，说明 `MOV AX，COUNT[BP]` 的寻址方式，求物理地址及执行后 AX（已知 (14100H)=30H，(14101H)=20H）。

解答： EA = BP + 16 位位移量 COUNT，为**基址寻址**（相对基址）；BP 默认 SS。 EA = 2000H + 2100H = 4100H；PA = SS×16 + EA = 10000H + 4100H = **14100H**。 AL←(14100H)=30H，AH←(14101H)=20H → **AX=2030H**。

题8（课件 p.17）

设 SS=2000H，BP=1000H，COUNT=2000H（16 位偏移量），求 `MOV AX，COUNT[BP]` 的物理地址与执行结果；并说明 `MOV BL，2[BX]` 的含义。

解答： PA = 16×SS + BP + COUNT = 20000H + 1000H + 2000H = **23000H**。 执行结果：23000H、23001H 两单元内容送 AX。 `MOV BL，2[BX]` 等价于 `MOV BL，[BX+2]`：基址寻址，EA=BX+2，默认 DS，把 (DS:[BX+2]) 的**字节**送 BL。

题9（课件 p.19，例4-8）

按图4-7：DS=3000H，SI=2000H，COUNT=3000H，求 `MOV AX，COUNT[SI]` 的物理地址及执行后 AX（已知 (35000H)=30H，(35001H)=20H）。

解答： EA = SI + COUNT，为**变址寻址**（相对变址），默认 DS。 EA = 2000H + 3000H = 5000H；PA = 30000H + 5000H = **35000H**。 AL←30H，AH←20H → **AX=2030H**。

题10（课件 p.20）

设 DS=3000H，SI=1000H，COUNT=2000H，求 `MOV AX，COUNT[SI]` 的物理地址与执行结果。

解答： PA = 16×DS + SI + COUNT = 30000H + 1000H + 2000H = **33000H**。 执行结果：33000H、33001H 两单元内容送 AX。

题11（课件 p.22，例4-9）

按图4-8：DS=3000H，BX=1000H，SI=2000H，求 `MOV AX，[BX+SI+250H]` 的物理地址及执行后 AX（已知 (33250H)=30H，(33251H)=20H）。

解答： EA = BX + SI + 250H，为**基址加变址寻址**（带位移即相对基址加变址）； BX 默认 DS。 EA = 1000H + 2000H + 0250H = 3250H；PA = 30000H + 3250H = **33250H**。 AL←30H，AH←20H → **AX=2030H**。

题12（课件 p.23，例4-10）

说明 `MOVSB` 的寻址方式与执行后的操作。

解答： **数据串寻址**：源串偏移在 SI（默认 DS），目标串偏移在 DI（默认 ES）。 执行后：[ES:DI]←[DS:SI]（传送一个字节），SI←SI±1，DI←DI±1（DF=0 加 1，DF=1 减 1；字节串增/减 1，字串 MOVSW 为 2）。

题13（课件 p.24，例4-11）

说明 `IN AL，20H` 与 `OUT DX，AL` 的端口寻址方式及功能。

解答： `IN AL，20H`：**直接端口寻址**，8 位端口地址 20H 由指令直接给出（范围 00H–FFH，256 个端口）；功能：把 20H 端口的字节读入 AL。 `OUT DX，AL`：**间接端口寻址**，16 位端口地址在 DX 中（可寻址 64K 个端口）；功能：把 AL 内容输出到 DX 所指端口。

题14（课件 p.25，例4-12）

说明 `AAA` 的寻址方式与隐含操作数。

解答： **隐含寻址**：指令中不明显给出操作数。AAA 对 AL 中两个非组合 BCD 码相加的结果做十进制调整，调整结果放入 AH 和 AL，隐含操作数为 **AH 和 AL**。

题15（课件 p.26–27）

设 BX=0158H，DI=10A5H，位移量=1B57H，DS=2100H，求下列各寻址方式的有效地址和物理地址：① 直接寻址；② 寄存器间接寻址(BX)；③ BX 相对基址寻址；④ 变址寻址(DI)；⑤ DI 相对变址寻址；⑥ 基址加变址(BX+DI)；⑦ 相对基址加变址(BX+DI)。

解答：统一 PA = DS×16 + EA = 21000H + EA。① EA = 1B57H；PA = 21000H + 1B57H = **22B57H**。② EA = BX = 0158H；PA = 21000H + 0158H = **21158H**。③ EA = 0158H + 1B57H = 1CAFH；PA = 21000H + 1CAFH = **22CAFH**。④ EA = DI = 10A5H；PA = 21000H + 10A5H = **220A5H**。⑤ EA = 10A5H + 1B57H = 2BFCH；PA = 21000H + 2BFCH = **23BFCH**。⑥ EA = 0158H + 10A5H = 11FDH；PA = 21000H + 11FDH = **221FDH**。⑦ EA = 0158H + 10A5H + 1B57H = 2D54H；PA = 21000H + 2D54H = **23D54H**。（十六进制加法验算：0158+1B57：8+7=F、5+5=A、1+B=C、0+1=1 → 1CAF；10A5+1B57：5+7=C、A+5=F、0+B=B、1+1=2 → 2BFC；11FD+1B57：D+7=14 写 4 进 1、F+5+1=15 写 5 进 1、1+B+1=D、1+1=2 → 2D54。）

【数据传送指令】

题16（课件 p.34）

判断下列指令正确与否，错误的说明原因： MOV AL,BL；MOV CX,BX；MOV BX,DL；MOV DX,34H；MOV DS,1234H；MOV ES,AX；MOV CS,BX；MOV [SI],CX；MOV [DI],[SI]；MOV 2000H,AX；MOV DS:[2000H],AX。

解答：| 指令 | 判断 | 理由 | |---|---|---| | MOV AL,BL | √ | 8 位寄存器→8 位寄存器 | | MOV CX,BX | √ | 16 位→16 位 | | MOV BX,DL | × | 类型不一致（16 位←8 位） | | MOV DX,34H | √ | 立即数→16 位寄存器 | | MOV DS,1234H | × | 立即数不能直接送段寄存器 | | MOV ES,AX | √ | 通用寄存器→段寄存器（ES 可作目的） | | MOV CS,BX | × | CS 不能作目的操作数 | | MOV [SI],CX | √ | 寄存器→存储器，类型明确 | | MOV [DI],[SI] | × | 两操作数不能同为存储器 | | MOV 2000H,AX | × | 目的不能为立即数 | | MOV DS:[2000H],AX | √ | 段超越的直接寻址存储器作目的，合法 |

题17（课件 p.40–41）

将 CX 的内容压入堆栈，然后弹出栈顶至 CX 中。已知 (SS)=0200H，(SP)=0008H，(CX)=12FAH。写出两步执行后 SP、相关存储单元及 CX 的内容。

解答：栈段基址 = SS×16 = 02000H。① PUSH CX：SP←0008H-1=0007H，(02007H)←CH=12H；SP←0007H-1=0006H，(02006H)←CL=FAH。此时 **SP=0006H**，(02006H)=FAH（新栈顶），(02007H)=12H。② POP CX：CL←(02006H)=FAH，SP←0007H；CH←(02007H)=12H，SP←0008H。此时 **SP=0008H（复原），CX=12FAH（不变）**。结论：压入再弹出，数据与 SP 均恢复原状；栈向低地址生长，高字节存高地址。

题18（课件 p.42）

已知 AX=1122H，BX=3344H，SP=1010H，执行：PUSH AX / PUSH BX / POP AX / POP BX。求执行后 AX、BX、SP。

解答：PUSH AX：SP=100EH，(100EH)=22H，(100FH)=11H； PUSH BX：SP=100CH，(100CH)=44H，(100DH)=33H； POP AX：AX←字(100CH)=3344H，SP=100EH； POP BX：BX←字(100EH)=1122H，

SP=1010H。 结果：**AX=3344H，BX=1122H，SP=1010H**（利用堆栈后进先出实现两寄存器交换，SP 复原）。

题19（课件 p.45）

判断：XCHG AX,BX；XCHG BH,BL；XCHG AX,1122H；XCHG DS,AX；XCHG [SI],BP；XCHG [SI],[DI]。

解答： | 指令 | 判断 | 理由 | |---|---|---| | XCHG AX,BX | √ | 通用寄存器之间字交换 | | XCHG BH,BL | √ | 通用寄存器之间字节交换 | | XCHG AX,1122H | × | 立即数不能作操作数 | | XCHG DS,AX | × | 段寄存器不能作 XCHG 操作数 | | XCHG [SI],BP | √ | 存储器与通用寄存器交换 | | XCHG [SI],[DI] | × | 两存储单元不能直接交换 |

题20（课件 p.48）

数据段中有一张 ASCII 码转换表，首地址偏移 2000H，DS=4000H，表内容依次为 '0','1','2',...,'9','A','B',...,'F' 的 ASCII 码。欲查表中序号 9 的代码，写出指令序列并求执行后 AL。

解答：

```
MOV BX，2000H   ；BX←表首偏移地址
MOV AL，09H     ；AL←表内序号 9
XLAT            ；AL←(DS:[BX+AL])，查表换码（课件误印为 XALT）
```

EA = BX+AL = 2000H+09H = 2009H；PA = 40000H+2009H = 42009H。 表中序号 0 起为 30H('0')...序号 9 为 39H('9')，故执行后 **AL=39H**。 要点：XLAT 只用 BX、AL，表长≤256 字节，不影响标志位。

题21（课件 p.50）

区分下列指令的执行结果（图中 [DATA1]=34H、[DATA1+1]=12H；BX=1100H，[1100H]=88H、[1101H]=77H）： LEA SI,DATA1；MOV SI,DATA1；MOV SI,OFFSET DATA1；MOV BX,[BX]；LEA BX,[BX]。

解答： - `LEA SI，DATA1`：取 DATA1 的**有效地址（偏移地址）**送 SI，SI=OFFSET DATA1（符号地址的偏移量，不是内容）。 - `MOV SI，DATA1`：取 DATA1 单元的**字内容**送 SI，SI=1234H（低地址 34H→低 8 位，高地址 12H→高 8 位）。 - `MOV SI，OFFSET DATA1`：OFFSET 取偏移地址，与 LEA 等价，SI=OFFSET DATA1。 - `MOV BX，[BX]`：以 BX=1100H 为 EA 取字内容，BX=7788H（88H→BL，77H→BH）。 - `LEA BX，[BX]`：取 EA=BX 的原值送 BX，**BX=1100H 不变**。 结论：LEA/OFFSET 取地址，MOV 直接变量名/间址取内容。

题22（课件 p.52）

下列指令执行后 DI、AX 的值是多少？（内存：[1200H]=34H，[1201H]=12H，[1202H]=00H，[1203H]=60H；[61234H]=33H，[61235H]=22H） LDS DI,[1200H] / MOV AX,[DI]

解答： LDS 取 4 字节指针：低地址字→DI，高地址字→DS。 DI = 字(1200H) = 1234H（34H→DL，12H→DH）；DS = 字(1202H) = 6000H（00H→低，60H→高）。 MOV AX,[DI]：PA = DS×16+DI =

60000H+1234H = 61234H；AL←(61234H)=33H，AH←(61235H)=22H。 结果：**DI=1234H，AX=2233H**（同时 DS 被改为 6000H）。

【算术运算指令】

题23（课件 p.57）

判断：ADD AL,BX；ADD CL,CH；ADD AX,[BX]；ADD [BX],28；ADD [BX],[SI]；ADD 1000H,AX；ADD [SI],BX；ADD DS,BX。

解答：| 指令 | 判断 | 理由 | |---|---|---| | ADD AL,BX | × | 类型不一致（8 位←16 位） | | ADD CL,CH | √ | 8 位+8 位 | | ADD AX,[BX] | √ | 寄存器+存储器，类型明确 | | ADD [BX],28 | × | 存储器目的操作数类型不明（须 BYTE PTR 或 WORD PTR） | | ADD [BX],[SI] | × | 不能同为存储器 | | ADD 1000H,AX | × | 目的不能为立即数 | | ADD [SI],BX | √ | 字存储器+字寄存器 | | ADD DS,BX | × | 段寄存器不能作算术运算操作数 |

题24（课件 p.58）

用 ADC 实现两个双字（32 位）相加：0123FAB5H + 0ABC212AH，写出程序段并给出每步结果。

解答：

```
MOV DX，0123H     ；被加数高字
MOV AX，0FAB5H    ；被加数低字
ADD AX，212AH     ；低字相加：FAB5H+212AH=11BDFH → AX=1BDFH，CF=1
ADC DX，0ABCH     ；高字带进位：0123H+0ABCH+1=0BE0H → DX=0BE0H，CF=0
```

验算：FAB5+212A：5+A=F、B+2=D、A+1=B、F+2=11 → 11BDF，取低 16 位 1BDF、进位 1；0123+0ABC+1=0BE0。 结果 (DX,AX) = **0BE01BDFH**。要点：多字节加法低位用 ADD、高位用 ADC 承接进位。

题25（课件 p.59–60）

有两个 4 字节数分别放在自 2000H 和 3000H 开始的存储区中（低字节在低地址），编程序段将两数相加，结果存于从 2000H 开始的单元中。并给出用符号地址 FIRST/SECOND 的等价写法。

解答：

```
MOV SI，2000H      ；SI 指向被加数（结果区）
MOV AX，[SI]       ；取被加数低字
MOV DI，3000H      ；DI 指向加数
ADD AX，[DI]       ；低字相加，产生进位 CF
MOV [SI]，AX       ；存低字结果
MOV AX，[SI+2]     ；取被加数高字
ADC AX，[DI+2]     ；高字带进位相加
MOV [SI+2]，AX     ；存高字结果
```

符号地址版（课件 p.60）：

```
MOV AX，FIRST
ADD AX，SECOND
MOV FIRST，AX
MOV AX，FIRST+2
ADC AX，SECOND+2
MOV FIRST+2，AX
```

要点：4 字节=两个字，先低字 ADD 后高字 ADC；低字节在低地址与 8086 小端一致，故按字取即可。

题26（课件 p.61）

判断：INC CL；INC [DI]；INC 2000H。错误的改正。

解答： - INC CL：√（8 位寄存器）。 - INC [DI]：×，存储器操作数类型不明；改为 `INC BYTE PTR [DI]`（或 WORD PTR）后 √。 - INC 2000H：×，立即数不能作操作数（INC 是单操作数指令，操作数须为寄存器/存储器）。 另记：INC 影响 AF、OF、PF、SF、ZF，**不影响 CF**（课件 p.62）。

题27（课件 p.70）

阅读程序：两组 50 个字节数据分别放在 FIRST 和 SECOND 开始的存储区，比较是否相同，相同 0→AL，否则 0FFH→AL。说明程序思路与关键指令作用。 MOV SI,OFFSET FIRST / MOV DI,OFFSET SECOND / MOV CX,50 / P: MOV AL,[SI] / CMP AL,[DI] / JZ NEXT / MOV AL,0FFH / JMP STOP / NEXT: INC SI / INC DI / DEC CX / JNZ P / MOV AL,0 / STOP: HLT

解答： 思路：逐字节比较，一旦发现不等立即置 AL=0FFH 并退出；50 字节全部相等则置 AL=0。 - `CMP AL,[DI]`：做减法只置标志，不改变 AL；相等时 ZF=1。 - `JZ NEXT`：相等则跳过「置 FFH」继续比较下一字节。 - 不等时顺序执行 `MOV AL,0FFH` + `JMP STOP` 提前结束。 - `INC SI/DI` 移动指针（INC 不影响 CF，但此处用 DEC CX+JNZ 控制循环，ZF 由 DEC 产生）。 - `DEC CX` / `JNZ P`：计数 50 次；循环正常走完说明全同，`MOV AL,0`。 评分要点：CMP 不回送、ZF 判等、提前退出与正常退出两条路径、CX 计数。

题28（课件 p.71）

A=-36，B=+127，用补码加法计算 A-B，给出 SF、OF 并判断 A、B 大小。

解答： [A]补 = 11011100（DCH），[-B]补 = 10000001（81H）。 11011100 + 10000001 = 1 01011101（最高位进位自然丢失），结果 5DH。 SF = 0（结果最高位 0）；OF = 1（两负数相加得正，溢出）；CF = 1（有进位）。 带符号比较判据：SF⊕OF = 0⊕1 = 1 → **A < B**。 （真值验证：-36-127=-163，超出 8 位补码范围 -128~+127，故溢出，直接看 SF 会误判，必须用 SF⊕OF。）

题29（课件 p.72–73）

分析程序功能： LEA BX,MAX / LEA SI,BUF / MOV CL,20 / MOV AL,[SI] / NEXT: INC SI / CMP AL,[SI] / JNC GOON / XCHG [SI],AL / GOON: DEC CL / JNZ NEXT / MOV [BX],AL / HLT

解答： 功能：**在 BUF 开始的一组无符号字节数中找最大数，存入 MAX 单元**（课件 p.73）。 逐步：AL 先取首元素作为「当前最大」；每轮 INC SI 指向下一元素，CMP AL,[SI] 做无符号比较；JNC GOON（CF=0，即 AL≥[SI]）则不交换；否则 XCHG [SI],AL 把更大者换入 AL；DEC CL/JNZ 控制循环；最后 MOV [BX],AL 存最大值。 关键点：无符号比较用 CF（JNC），若带符号应改用 JGE；XCHG 保证 AL 始终保存已扫描部分的最大值。 存疑标注：课件称「20 个数」，但 CL=20 使循环比较 20 次、连同首元素共扫描 21 个字节；若严格 20 个数应置 CL=19。课件原文如此，存疑：循环次数与元素个数差 1。

题30（课件 p.77–78）

(1) 已知 AL=0FEH，BH=0AH，分别执行 MUL BH 与 IMUL BH，求 AX 及 CF、OF。 (2) 设 M1、M2 单元中各有一个 16 位无符号数，求乘积存于从 R 开始的字单元中，写程序段。

解答： (1) AL=FEH：无符号 254，带符号 -2；BH=0AH=10。 MUL BH：254×10=2540=09ECH → **AX=09ECH**；字节积高半 AH=09H≠0 → **CF=OF=1**。 IMUL BH：(-2)×10=-20，补码 FFECH → **AX=FFECH**；AH=FFH 恰为 AL=ECH（符号位 1）的符号扩展 → **CF=OF=0**。 (2)

```
MOV AX，M1      ；被乘数（隐含）
MUL M2          ；无符号字乘：(DX,AX)←AX×M2
MOV R，AX       ；积低字
MOV R+2，DX     ；积高字
```

要点：MUL/IMUL 单操作数、源不能立即数；只影响 CF、OF；CF=OF=1 表示高半部分有效（积超出低半范围）。

题31（课件 p.83）

分析下列指令执行后的结果： ① MOV AL,44H / CBW；② MOV AX,0AFDEH / CWD；③ MOV AL,86H / CBW。

解答： ① AL=44H<80H（正）→ AH=00H，**AX=0044H**。 ② AX=AFDEH≥8000H（负）→ DX=FFFFH，**AX=0AFDEH 不变，DX=FFFFH**。 ③ AL=86H≥80H（负）→ AH=FFH，**AX=FF86H**。 要点：CBW/CWD 只做

符号扩展、不影响标志位，常置于 IDIV 之前把被除数扩成双倍字长。

题32（课件 p.84）

内存中从 DATA1 开始的存储区：前两个字节为 16 位带符号被除数，第 3、4 字节为 16 位带符号除数，其后四个单元依次存商和余数。阅读并解释程序： LEA BX,DATA1 / MOV AX,[BX] / CWD / IDIV WORD PTR [BX+2] / MOV [BX+4],AX / MOV [BX+6],DX

解答： - LEA BX,DATA1：BX 取数据区首偏移（取地址不是内容）。 - MOV AX,[BX]：取 16 位被除数。 - CWD：把 AX 符号扩展到 DX，构成 32 位被除数 (DX,AX)——带符号字除法必需。 - IDIV WORD PTR [BX+2]：除以偏移 BX+2 处的 16 位带符号除数；商→AX，余数→DX（余数符号与被除数相同）。 - MOV [BX+4],AX / MOV [BX+6],DX：商、余数依次存入后续单元。 要点：WORD PTR 指明除数为字；若商超出 -32768~+32767 将产生 0 号中断。

题33（课件 p.85）

计算 (-125)×10÷300，程序如下，求运算结果： MOV AL,83H / MOV BL,10 / IMUL BL / CWD / MOV BX,300 / IDIV BX

解答： 83H 为 -125 的补码。IMUL BL：(-125)×10=-1250 → AX=FB2EH。 CWD：AX 为负 → DX=FFFFH，被除数扩为 32 位 FFFFFB2EH（=-1250）。 IDIV BX：-1250÷300 = -4 余 -50（余数符号随被除数）。 商 -4 补码 = **AX=FFFCH**；余数 -50 补码 = **DX=FFCEH**。 验算：300×(-4)+(-50) = -1200-50 = -1250 ✓。

题34（课件 p.89、91、93、95、97、99）

写出下列各 BCD 调整程序段的执行结果（含标志）： ① MOV AL,07H / MOV BL,08H / ADD AL,BL / AAA ② MOV AL,68H / MOV BL,59H / ADD AL,BL / DAA ③ MOV AX,0103H / MOV BL,04H / SUB AL,BL / AAS ④ MOV AL,83H / MOV BL,38H / SUB AL,BL / DAS ⑤ MOV AL,07H / MOV BL,09H / MUL BL / AAM ⑥ MOV AX,0703H / MOV BL,02H / AAD / DIV BL / AAM

解答： ① ADD：07+08=0FH，AF=0。AAA：低 4 位 F>9 → AL=0FH+6=15H，AH=00H+1=01H，AF=CF=1，清 AL 高 4 位 → **AL=05H，AH=01H，CF=AF=1**（7+8=15）。 ② ADD：68H+59H=C1H，低半 8+9=11H>0FH → AF=1。DAA：低 4 位=1 但 AF=1 → AL=C1H+06H=C7H、AF=1；高 4 位 C>9 → AL=C7H+60H=127H → **AL=27H，CF=1**（68+59=127）。 ③ SUB：03H-04H=FFH（借位，AF=1、CF=1），AH=01H。AAS：低 4 位 F>9 → AL=FFH-6=F9H，AH=01H-1=00H，AF=CF=1，清高 4 位 → **AL=09H，AH=00H**（13-4=9，借位体现在 AH）。 ④ SUB：83H-38H=4BH，低半 3-8 借位 → AF=1。DAS：低 4 位 B>9 → AL=4BH-6=45H、AF=1；高 4 位 4≤9 且 CF=0 → 不调 → **AL=45H，CF=0**（83-38=45）。 ⑤ MUL BL：7×9=63=3FH → AX=003FH（课件误印 MUL AL,BL，MUL 为单操作数）。AAM：AH←63/10=6，AL←63%10=3 → **AH=06H，AL=03H**；SF=0、ZF=0、PF=1（03H 含两个 1，偶校验）。 ⑥ AAD：AL←AH×10+AL=7×10+3=73=49H，AH←0。DIV BL：73÷2=36 余 1 → AL=24H、AH=01H。AAM：36→AH=03H、AL=06H → **AH=03H，AL=06H**（商 36 的非组合 BCD）。 要点：AAA/AAS 只影响 AF、CF；DAA/DAS 影响 AF、CF、SF、ZF、PF；AAM/AAD 影响 SF、ZF、PF；调整必须紧跟对应运算指令。

题35（课件 p.101–102）

(1) AX、BX 中各存一个组合 BCD 码十进制数（两位），编程序段相加，结果存 AX。 (2) 1000H 开始连续存放 4 个字节，为两个组合 BCD 数（各两字节），相加（相减）结果送回 1000H，写程序段。

解答： (1)

```
ADD AL，BL      ；低字节（个位、十位）相加
DAA             ；低字节组合 BCD 调整
MOV CL，AL      ；暂存低字节结果
MOV AL，AH
ADC AL，BH      ；高字节带进位相加
DAA             ；高字节调整
MOV AH，AL
MOV AL，CL      ；拼回 AX
```

(2)

```
MOV SI，1000H
MOV AL，[SI]
ADD AL，[SI+2]   ；两数低字节相加
DAA
MOV [SI]，AL     ；存低字节
MOV AL，[SI+1]
ADC AL，[SI+3]   ；高字节带进位相加
DAA
MOV [SI+1]，AL   ；存高字节
```

（相减时把 ADD/ADC 换为 SUB/SBB、DAA 换为 DAS 即可。） 要点：DAA 只调整 AL，故多字节 BCD 必须从低字节逐字节「运算+调整」；高字节用 ADC 承接低字节进位。

【逻辑运算与移位指令】

题36（课件 p.103–107）

求下列各指令执行后的结果： ① AL=46H，AND AL,0FH；② AX=8888H，OR AX,00FFH；③ AX=3333H，XOR AX,00FFH；④ AX=0001H，NOT AX；⑤ TEST AX,8000H 检测 D15，说明 ZF 与 D15 的关系。

解答： ① 46H=01000110，与 0FH 按位与 → 00000110 = **06H**（高 4 位屏蔽清 0，低 4 位保留）。 ② 8888H 或 00FFH → **88FFH**（低 8 位置 1，其余不变）。 ③ 3333H 异或 00FFH：AH 不变、AL 按位取反（33H→CCH）→ **33CCH**。 ④ 0001H 逐位取反 → **FFFEH**（NOT 不影响标志，符号位也取反）。 ⑤ TEST 做与运算不回送：若 D15=1，与 8000H 结果非 0 → **ZF=0**；若 D15=0，结果 0 → **ZF=1**。 共性：AND/OR/XOR/TEST 均使 CF=OF=0，SF/ZF/PF 按结果；NOT 不影响任何标志。

题37（课件 p.108）

设 AX=AAAAH，BX=BBBBH，CX=CCCCH。将 BX 的高 4 位、CX 的低 4 位、AX 的中间 8 位取反，拼成新字送 AX。写出程序并求结果。

解答：

```
AND BH，0F0H     ；取 BX 高 4 位：BH=B0H
AND CL，0FH      ；取 CX 低 4 位：CL=0CH
AND AX，0FF0H    ；保留 AX 中间 8 位：AX=0AA0H
XOR AX，0FF0H    ；中间 8 位取反：AX=0550H
OR  AH，BH       ；高 4 位拼入：AH=05H|B0H=B5H
OR  AL，CL       ；低 4 位拼入：AL=50H|0CH=5CH
```

结果 **AX=B55CH**。 要点：AND 屏蔽（清 0）、XOR 取反、OR 拼装（置位）三步位操作套路；各步互不干扰的位域才能直接 OR。

题38（课件 p.111）

求：① MOV BX,FFFCH / SAR BX,1；② MOV BL,0CH / SHL BL,1。

解答： ① FFFCH = -4 的补码。SAR 算术右移：符号位 1 保持，最低位 0 移入 CF → **BX=FFFEH（-2），CF=0**。体现「算术右移=带符号数除 2」。 ② 0CH=12。SHL 左移：最低位补 0，最高位 0 移入 CF → BL=18H=**24**，CF=0。体现「左移=乘 2」。

题39（课件 p.112）

编程序段对 AX 内容乘 10（设无溢出，结果仍为一个字），要求用移位实现。

解答： 分析：AX×10 = AX×(2+8) = AX×2 + AX×2³。

```
MOV BX，AX       ；保存原数
SAL BX，1        ；BX=原数×2
MOV CL，3        ；移位次数>1 必须用 CL
SAL AX，CL       ；AX=原数×8
ADD AX，BX       ；AX=原数×10
```

要点：8086 移位计数>1 只能 CL；SAL 与 SHL 等价；用移位+加法代替 MUL 是常考技巧（×10、×5、×9 同理）。

题40（课件 p.116）

已知 DS=7B00H，SI=1B79H，(7CB79H)=07H。执行 MOV CL,3 / SAL [SI],CL 后，(7CB79H) 单元内容和 CF 为多少?

解答：先求操作数物理地址：PA = DS×16+SI = 7B000H+1B79H = 7CB79H，与给定单元一致。07H=00000111，左移 3 次：第 1 次：00001110，移出 0 → CF=0；第 2 次：00011100，移出 0 → CF=0；第 3 次：00111000，移出 0 → CF=0。结果 **(7CB79H)=38H，CF=0**（CF 只保留最后一次移出位）。

题41（课件 p.117–118）

BUF 单元开始连续存放 8 个组合 BCD 码，要求转换成 ASCII 码存放在 RES 开始的缓冲区，高位 BCD 转换的 ASCII 放地址较高的单元。写出完整程序并说明 45H 的转换过程。

解答：

```
MOV SI，OFFSET BUF   ；源指针
MOV DI，OFFSET RES   ；目标指针
MOV BL，8            ；8 个组合 BCD 字节
TRA：MOV AL，[SI]     ；取一个组合 BCD
MOV BH，AL           ；备份（留高 4 位）
AND AL，0FH          ；取低 4 位
OR  AL，30H          ；低 4 位→ASCII
MOV [DI]，AL         ；存低位的 ASCII（低地址）
INC DI
MOV CL，4            ；移位次数 4 装 CL
SHR BH，CL           ；高 4 位移到低 4 位
OR  BH，30H          ；高 4 位→ASCII
MOV [DI]，BH         ；存高位的 ASCII（高地址）
INC SI
INC DI
DEC BL
JNZ TRA              ；8 个字节循环
HLT
```

45H 的转换：低半 5 → AND 0FH 得 05H → OR 30H 得 35H('5')；高半 4 → SHR 4 位得 04H → OR 30H 得 34H('4')。RES 中先 35H 后 34H，即高位数字的 ASCII 在高地址，符合题意。要点：一字节拆两字符用「AND 取低 + SHR 取高」，OR 30H 完成 BCD→ASCII；CL 保护移位次数。

题42（课件 p.119–120）

分析程序功能：MOV SI,1000H / MOV DI,3000H / MOV CX,4 / BBB: MOV AL,[SI] / MOV BL,AL / AND AL,0FH / OR AL,30H / MOV [DI],AL / INC DI / MOV AL,BL / PUSH CX / MOV CL,4 / SHR AL,CL / OR AL,30H / MOV [DI],AL / INC DI / INC SI / POP CX / DEC CX / JNZ BBB / HLT

解答：功能：**将 1000H 开始存放的 4 个组合 BCD 码转换为 8 个 ASCII 码，存放在 3000H 开始的单元中**（课件 p.120），每个 BCD 字节先存低位数字的 ASCII、再存高位数字的 ASCII。关键点：- 外层计数用 CX=4，而 SHR 需要 CL=4 作移位计数，二者冲突，故用 `PUSH CX` 保存串计数、`POP CX` 恢复——这是本程序的考点。- AND

0FH 取低半、SHR 4 取高半，OR 30H 转 ASCII。 - 例：(1000H)=12H → (3000H)=32H('2')、(3001H)=31H('1')。 - DEC CX/JNZ 控制 4 轮；SI 每轮 +1、DI 每轮 +2。

【串操作指令】

题43（课件 p.129–130）

将 DATA1 开始的 100 个字节传送到 DATA2 开始的存储区，分别用①一般传送指令、②串传送指令（循环）、③重复前缀三种方法实现。

解答： ① 一般传送：

```
MOV SI，OFFSET DATA1
MOV DI，OFFSET DATA2
MOV CX，64H          ; 100=64H
PL1：MOV AL，[SI]
MOV [DI]，AL
INC SI
INC DI
DEC CX
JNZ PL1
DONE：HLT
```

② 串传送+循环：

```
MOV SI，OFFSET DATA1
MOV DI，OFFSET DATA2
MOV CX，64H
CLD                  ; DF=0，地址递增
PL1：MOVSB            ; 每次搬 1 字节并自动改 SI/DI
DEC CX
JNZ PL1
DONE：HLT
```

③ 重复前缀：

```
MOV SI，OFFSET DATA1
MOV DI，OFFSET DATA2
MOV CX，64H
CLD
REP MOVSB            ; CX≠0 重复，自动 CX-1，直到 CX=0
HLT
```

要点：串指令隐含 DS:SI→ES:DI（须保证 ES 指向目标段）；CLD 定方向；REP 把「DEC CX+JNZ」硬件化。

题44（课件 p.131）

比较存储器中两串数（50 字节）是否相同，完全相同 AL=0，否则 AL=0FFH。阅读程序并解释 REPZ CMPSB 与 JZ 的配合： MOV SI,OFFSET DATA1 / MOV DI,OFFSET DATA2 / CLD / MOV CX,50 / REPZ CMPSB / JZ EQQ / MOV AL,0FFH / JMP STOP / EQQ: MOV AL,0 / STOP: HLT

解答： `REPZ CMPSB`：每轮做 [DS:SI]-[ES:DI] 置标志、SI/DI+1、CX-1；**CX≠0 且 ZF=1（相等）才继续**，一旦不等（ZF=0）或 CX=0 退出。 退出后两种情况：ZF=1 说明 50 字节全比完且都相等 → `JZ EQQ` 置 AL=0；ZF=0 说明中途发现不等 → 置 AL=0FFH。 要点：REPE/REPZ 用于「找不同」，REPNE/REPNZ 用于「找相同」；退出后必须再用 JZ/JNZ 区分「比完」与「提前退出」。

题45（课件 p.132）

在某字串（100 字节）中查找是否存在字符 '$'，存在则 BX=其偏移地址，否则 BX=0。阅读程序并解释 REPNZ SCASB 与 DEC DI 的作用： MOV DI,OFFSET DATA1 / MOV AL,'$' / MOV CX,64H / CLD / REPNZ SCASB / JZ FOUND / MOV BX,0 / JMP STOP / FOUND: DEC DI / MOV BX,DI / STOP: HLT

解答： `REPNZ SCASB`：每轮 AL-[ES:DI] 置标志、DI+1、CX-1；**CX≠0 且 ZF=0（不等）才继续**，找到 '$'（ZF=1）或 CX=0 退出。 `JZ FOUND`：ZF=1 即找到。注意 SCASB 比较后 DI 已自动 +1 指向下一元素，故 `DEC DI` 回退一步才是 '$' 所在偏移，再送 BX。 未找到（CX=0 退出，ZF=0）→ BX=0。 要点：SCAS 的关键字在 AL/AX、串在 ES:DI；搜索类前缀用 REPNE/REPNZ；命中后指针回退 1（字串回退 2）。

题46（课件 p.133）

将 20 个 0AAAAH 送入 3000H 开始的地址单元，程序如下，说明各指令作用： CLD / LEA DI,[3000H] / MOV CX,14H / MOV AX,0AAAAH / REP STOSW

解答： - CLD：DF=0，DI 递增。 - LEA DI,[3000H]：DI=3000H（取有效地址；等价 MOV DI,3000H）。 - MOV CX,14H：20=14H，字个数。 - MOV AX,0AAAAH：待存的字。 - REP STOSW：每轮 [ES:DI]←AX、DI+2、CX-1，重复 20 次，把 3000H 起 40 字节填为 AAAAH。 存疑标注：课件程序未设置 ES，而 STOSW 目标段固定为 ES；实际运行须先使 ES 指向 3000H 所在段（如 ES=DS）。课件原文如此，存疑：缺 ES 初始化。

题47（课件 p.134）

将某数据区 100 个数中的正数、负数分开，分别送同一数据段的另外两个数据区。阅读程序并说明思路： MOV SI,OFFSET DATA1 / MOV DI,OFFSET DATA2 / MOV BX,OFFSET DATA3 / MOV CX,64H / CLD / LP: LODSB / TEST AL,80H / JNZ MIVS / STOSB / JMP NEXT / MIVS: XCHG BX,DI / STOSB / XCHG BX,DI / NEXT: DEC CX / JNZ LP / HLT

解答： 思路：LODSB 从 DS:SI 逐字节取数（SI 自动 +1）； `TEST AL,80H` 检测符号位 D7：ZF=0（D7=1，负数）转 MIVS；正数直接 STOSB 存入 ES:DI（DATA2 区，DI 自动 +1）。 负数分支： `XCHG BX,DI` 把负数区指针 DATA3 换入 DI，STOSB 存入后再 `XCHG BX,DI` 换回——用交换指令在两个目标指针间切换，避免另设指针修改指

令。 DEC CX/JNZ 循环 100 次。 要点：TEST 判位不回送；LODSB/STOSB 自动改指针；课件说「同一数据段」，故须 ES=DS（课件未显式设置，存疑：缺 ES 初始化）。

【控制转移指令】

题48（课件 p.151）

DATA1 开始的存储区放了 100 个 8 位无符号数，找出最大数存 MAX 单元。阅读程序并说明 JAE 的作用与循环次数： MOV BX,OFFSET DATA1 / MOV AL,[BX] / MOV CX,63 / P1: INC BX / CMP AL,[BX] / JAE NEXT / MOV AL,[BX] / NEXT: DEC CX / JNZ P1 / MOV MAX,AL / HLT

解答： AL 存「当前最大」，初值为第 1 个数；每轮 INC BX 指向下一数，CMP AL,[BX] 无符号比较；`JAE NEXT`（CF=0，AL≥[BX]）则不更新；否则 MOV AL,[BX] 更新最大值；DEC CX/JNZ 循环；最后存 MAX。 循环次数：应比较其余 99 个数，故 CX 初值应为 99=**63H**。课件写作「MOV CX,63」，若按十进制 63 则只比较 64 个数，与「100 个数」不符。课件原文如此，存疑：63 应为 63H。 要点：无符号比较用 JAE/JB 族（看 CF）；求最大套路=取首值+逐个比较+条件更新。

题49（课件 p.154，例4-59）

某数据段中有 10 个字节的数组 BUF，要求找出 '$' 字符，然后将 '$' 前的所有数据相加，结果放 AX。阅读程序并说明两段循环的作用： MOV SI,OFFSET BUF / DEC SI / MOV CX,10 / LL1: INC SI / CMP BYTE PTR [SI],'$' / LOOPNZ LL1 / SUB SI,OFFSET BUF / JZ LL3 / MOV CX,SI / MOV SI,OFFSET BUF / MOV AX,0 / LL2: ADD AL,[SI] / ADC AH,0 / INC SI / LOOP LL2 / LL3: HLT

解答： 第一段（定位）：SI 先减 1，循环内 INC SI 使首轮指向 BUF[0]；`CMP BYTE PTR [SI],'$'` 逐字节比较；`LOOPNZ LL1` 在 CX≠0 且 ZF=0（不是 '$'）时继续，找到 '$' 或扫完 10 字节退出。退出时 SI 指向 '$'（或 BUF+10）。 `SUB SI,OFFSET BUF`：得 '$' 的下标 k，即 '$' 前的字节数；JZ LL3 处理 k=0（'$' 在首字节，前面无数据）。 第二段（求和）：CX=k 作计数，SI 回到 BUF 首；`ADD AL,[SI]` 累加低字节、`ADC AH,0` 把进位加进高字节，实现 16 位和；LOOP 循环 k 次；结果在 AX。 边界说明：若未找到 '$'，k=10，程序把 10 字节全部求和；若 '$' 在首字节（k=0）直接跳 LL3，课件未先清 AX（存疑：应补 MOV AX,0 再跳）。 要点：LOOPNZ=「不等则循环」适合搜索；用地址差求元素个数；字节累加用 ADC AH,0 扩展成字和。

二、考点补充题

题50【补充•考点：循环移位计算】

设 AL=9CH，分别求下列指令执行后 AL 与 CF（各小题独立，③④ 设执行前 CF=1）： ① ROL AL,1；② ROR AL,1；③ RCL AL,1；④ RCR AL,1；⑤ MOV CL,4 / ROL AL,CL。

解答： 9CH = 10011100B。 ① ROL（左环，CF 只记录移出位）：最高位 1 移出环回最低位 → 00111001 = **39H，CF=1**。 ② ROR（右环）：最低位 0 移出环回最高位 → 01001110 = **4EH，CF=0**。 ③ RCL（CF 入环，9 位左环）：新 AL = (AL<<1)|原CF = 00111000|1 = 00111011 = **3BH**；新 CF = 原最高位 = **1**。 ④ RCR（CF 入

环，9 位右环）：新 AL = (原CF<<7)|(AL>>1) = 10000000|01001110 = 11001110 = **CEH**；新 CF = 原最低位 = **0**。 ⑤ 左环 4 位=高低半字节交换：1001 1100 → 1100 1001 = **C9H**；CF=最后一次移出位=原 D4=**1**。 要点：ROL/ROR 的 CF 不入环，RCL/RCR 的 CF 入环；多位移位次数>1 必须 CL（8086）。

题51【补充•考点：指令执行后标志位计算】

逐小题给出执行后操作数与 CF、OF、SF、ZF、AF、PF： ① MOV AL,7FH / ADD AL,1；② MOV AL,80H / SUB AL,1；③ 设 CF=1，MOV AL,0FFH / INC AL；④ MOV AL,0 / NEG AL，再 MOV AL,5 / NEG AL；⑤ MOV AX,8000H / NEG AX。

解答： ① 7FH+1=80H：正+正得负 → OF=1；无进位 CF=0；SF=1；ZF=0；低半 F+1=10H 半进位 AF=1；80H 含 1 个 1（奇）PF=0。→ AL=80H，CF=0、OF=1、SF=1、ZF=0、AF=1、PF=0。 ② 80H-1=7FH：负-正得正 → OF=1；无借位 CF=0；SF=0；ZF=0；低半 0-1 借位 AF=1；7FH 含 7 个 1（奇）PF=0。→ AL=7FH，CF=0、OF=1、SF=0、ZF=0、AF=1、PF=0。 ③ FFH+1=00H：ZF=1、SF=0、OF=0、PF=1（0 个 1 为偶）、AF=1；**INC 不影响 CF，CF 保持 1**。→ AL=00H，CF=1、OF=0、SF=0、ZF=1、AF=1、PF=1。 ④ NEG 0：0-0=0，**CF=0**，ZF=1、SF=0、OF=0；NEG 5：0-5=FBH（-5 补码），操作数非 0 → **CF=1**，SF=1、ZF=0、OF=0。 ⑤ NEG 8000H：0-8000H 仍为 8000H（-32768 无对应正数），**OF=1**；操作数非 0 → CF=1；SF=1、ZF=0。→ AX=8000H，CF=1、OF=1。 要点：OF 看「符号意外」（正+正=负、负-正=正等）；INC/DEC 不动 CF；NEG 的 CF「非 0 即 1」；80H/8000H 取补值不变但 OF=1。

题52【补充•考点：默认段与段超越的物理地址计算】

设 DS=2000H，SS=3000H，ES=4000H，BX=1000H，BP=1000H，SI=0200H。求下列指令源操作数的物理地址： ① MOV AX,[BP+SI]；② MOV AX,DS:[BP+SI]；③ MOV AX,[BX+SI]；④ MOV AX,ES:[BX]；⑤ MOV AX,[BP]。并说明①为何不用 DS。

解答： ① 含 BP → 默认 SS：EA=1000H+0200H=1200H，PA=30000H+1200H=**31200H**。 ② 段超越 DS：PA=20000H+1200H=**21200H**。 ③ BX+SI → 默认 DS：EA=1200H，PA=20000H+1200H=**21200H**。 ④ 超越 ES：EA=BX=1000H，PA=40000H+1000H=**41000H**。 ⑤ BP → 默认 SS：PA=30000H+1000H=**31000H**。 说明：8086 约定凡用 BP 作基址/间址即访问堆栈段（便于子程序用 BP 指向栈内参数与局部变量），故默认 SS；要访问数据段必须加 DS: 前缀。

题53【补充•考点：串操作方向标志 DF 分析】

设 DS=ES=2000H，源串 10 字节位于偏移 0100H–0109H，目标区偏移 0200H–0209H。 ① CLD 后 SI=0100H、DI=0200H、CX=10，执行 REP MOVSB，求首/末次传送的物理地址及结束后 SI、DI、CX； ② STD 后 SI=0109H、DI=0209H、CX=10，执行 REP MOVSB，求首/末次传送物理地址及结束后 SI、DI、CX； ③ 若②中误置 SI=0100H、DI=0200H 会怎样？ ④ 若目标区改为 0105H–010EH（与源串重叠），应选哪个方向？为什么？

解答： ① DF=0 递增：首次 (20100H)→(20200H)，末次 (20109H)→(20209H)；结束后 SI=0100H+10=**010AH**，DI=**020AH**，CX=**0**。 ② DF=1 递减：首次 (20109H)→(20209H)，末次

(20100H)→(20200H)；结束后 SI=0109H-10=**00FFH**，DI=**01FFH**，CX=**0**。 ③ STD 却从首地址开始：将把 0100H、00FFH、00FEH… 的内容搬往 0200H、01FFH…，搬错区域（越出源串），故 DF=1 时指针必须预置到串**末**元素。 ④ 目标在源之上且重叠：应 **STD 反向（从高地址往低地址搬）**。若正向搬，源串高地址部分会先被覆盖再被读取，数据出错；反向搬时每次读取的源字节尚未被写过，结果正确。（若目标在源之下重叠则应 CLD 正向。） 要点：DF 只定方向、不改段约定；重叠串搬运方向是综合题常考点。