习题 4 参考答案

4-1

解：实现图 4-13 的逻辑关系式为：$Q=(U*(V+W)+\overline{X}*Y+\overline{Z})$，Q 为输出（I/O）参考子程序如下：

```
LOG:   MOV   C，V
       ORL   C，W
       ANL   C，U
       MOV   F0，C                    ；存中间结果
       MOV   C，Y
       ALN   C，/X
       ORL   C，F0
       ORL   C，/Z
       MOV   Q，C
       RET
```

4-2

解：参考子程序清单如下：

```
XORW:    MOV   C，  X                 ；  位变量 X 送 Cy
         ANL   C，  /Y                ；
         MOV   F0，  C                ；  暂存 F0
         MOV   C，  Y                 ；  位变量 Y 送 Cy
         ANL   C，/X                  ；
         ORL   C，F0                  ；  C←F0
         RET
```

4-3

解 一个二进制整数乘以 16，相当于左移 4 次。乘法左移要从数据的低位开始。

“小端”存储结构参考的通用子程序清单如下：

```
RLMUL:   MOV   R2，#ADDR0H           ；备份数据首地址信息
         MOV   R6，# n2              ；
RL4TIME: MOV   A，R2                 ；外层循环开始
         MOV   R0，A                 ；恢复数据首地址送入 R0
         CLR   C
         MOV   R7，#N
RLnTIME: MOV   A，@R0                ；内层循环开始
         RLC   A
         MOV   @R0，A
         INC   R0
         DJNZ  R7，RLnTIME           ；内层循环最后一条指令
         DJNZ  R6，RL4TIME           ；外层循环最后一条指令
         RET
```

则实现本题的程序为：

```
         ADDR0  EQU    40H
         N      EQU    4
```

```
          n2      EQU     4
          ORG     0000H
          SJMP    MAIN
MAIN：    MOV     SP，#5FH
          LCALL   RLMUL
          SJMP    $
RLMUL：   ……
          RET
          END
```

如果数据采用“大端”存储结构参考子程序清单如下：

```
RLMUL：   MOV     A，#ADDR0
          ADD     A，#N
          DEC     A
          MOV     R2，A                ；备份数据最低位地址信息
          MOV     R6，＃n2             ；
RL4TIME：MOV      A，R2                ；外层循环开始
          MOV     R0，A                ；恢复数据首地址送入 R0
          CLR     C
          MOV     R7，#N
RLnTIME：MOV      A，@R0               ；内层循环开始
          RLC     A
          MOV     @R0，A
          DEC     R0
          DJNZ    R7，RLnTIME          ；内层循环最后一条指令
          DJNZ    R6，RL4TIME          ；外层循环最后一条指令
          RET
```

4-4

解：

（1）参考子程序清单如下：

```
TWOADD：  MOV     R0，#ADDR0           ；被加数首址存入间址寄存器 R0 中
          MOV     R1，#ADDR1           ；加数首址存入间址寄存器 R1 中
          MOV     R7，＃n              ；数据长度存入 R7 中
          CLR     C                    ；清 Cy
AGAIN：   MOV     A，@R0               ；取被加数
          ADDC    A，@R1               ；（A）= 被加数+加数
          MOV     @R0，A               ；存此位部分和
          INC     R0                   ；指向下一个被加数与加数单元地址
          INC     R1
          DJNZ    R7，AGAIN            ；全部数据没处理完循环，否则向下执行
          MOV     F0，C                ；存进位值于 F0
          RET                          ；子程序返回
```

（2）参考子程序清单如下

```
          ADDR0       EQU     30H
          ADDR1       EQU     40H
          n           EQU     5
```

```
        ORG     0000H
        LJMP    SATRT
        ORG     0040H
START:  MOV     SP，#5FH
        MOV     30H，#9AH
        MOV     31H，#78H
        MOV     32H，#56H
        MOV     33H，#34H
        MOV     34H，#12H       ；被加数设置
        MOV     40H，#9AH
        MOV     41H，#78H
        MOV     42H，#56H
        MOV     43H，#34H       ；加数设置
        MOV     44H，#00H       ；最高位补 0
        LCALL   TWOADD
        SJMP    $               ；主程序与子程序的隔离带，调试断点。
TWOADD: ……                      ；内容省略
NOCY:   RET                     ；子程序返回指令
        END
```

习题 4-5

解：参考程序清单如下：

```
        ADDR16  EQU     2000H
        RESULTH EQU     30H     ；结果高位字单元地址
        RESULTL EQU     31H     ；结果高位字单元地址
        n1      EQU     16
        ADDR0   EQU     30H     ；“和”高位字单元地址
        N       EQU     2       ；N 为被除数的字节数
        n2      EQU     4       ；n2 为除数的 2 幂次
        ORG     0000H           ；主程序起点
        LJMP    START
        ORG     0040H           ；主程序内容从这里开始
START:  MOV     SP，#5FH        ；堆栈区设置
        LCALL   MULADD          ；数组求“和”
        LCALL   RRDIV           ；数组求“平均值”
        SJMP    $               ；建立隔离
MULADD: MOV     DPTR，#ADDR16   ；数组首址存入 DPTR 中
        MOV     R7，# n1        ；数据长度存入 R7 中
        MOV     RESULTH，#0     ；结果高位字节清 0
        MOV     RESULTL，#0     ；结果低位字节清 0
AGAIN:  MOVX    A，@DPTR        ；取数组数据
        ADD     A，RESULTL      ；计算累加值
        MOV     RESULTL，A      ；存和的低位
        JNC     NOCY            ；判断有吗？
        INC     RESULTH         ；有进位处理
NOCY:   INC     DPTR            ；指向数组的下一个单元
```

```
        DJNZ    R7，AGAIN       ；全部数据没处理完循环，否则向下执行
        RET                     ；子程序返回
RRDIV：  MOV    R0，#ADDR0      ；R0 指向数的最高位地址
        MOV     R6，＃n2        ；n2 为除数的 2 幂次
        MOV     A，R0
        MOV     R2，A           ；备份数据首地址信息
RR3TIME：MOV    A，R2           ；外层循环开始
        MOV     R0，A           ；恢复数据首地址送入 R0
        CLR     C               ；清 CY，使运算结果的高位字节前 n 位为 0
        MOV     R7，＃N         ；N 为被除数的字节数
RRnTIME：MOV    A，@R0          ；内层循环上边界，取出被除数的高位
        RRC     A               ；右移此字节内容
        MOV     @R0，A          ；再存入原地址单元中
        INC     R0              ；地址指针加 1，指向数的次高位单元
        DJNZ    R7，RRnTIME     ；内层循环下边界，一次整体右移未完成，继续
        DJNZ    R6，RR3TIME     ；外层循环最后一条指令
        RET
```

求平均值直接引用【例 4-12】的子程序。

4-6 解：

分析：本题给出了参与运算数的片内 RAM 地址，3 字节无符号整数乘 1 字节无符号整数，其积不超过 4 字节。为此我们要定义 4 个字节单元用于存放积。命名为 RES0（个位）、RES1、RES2、RES3。

本程序所采用的算法：与手算竖式相同。

参考子程序如下：

```
MULJKLM：PUSH   ACC             ；保护现场
        PUSH    B
        MOV     A，M            ；取乘数到 A
        MOV     B，L            ；取被乘数个位到 B
        MUL     AB              ；（L）×（M）
        MOV     LOML，A         ；存部分积低 8 位
        MOV     HIML，B         ; 存部分积高 8 位
        MOV     A，M            ；取乘数到 A
        MOV     B，K            ；取被乘数十位到 B
        MUL     AB              ；（K）×（M）
        MOV     LOMK，A         ；存部分积低 8 位
        MOV     HIMK，B         ; 存部分积高 8 位
        MOV     A，M            ；取乘数到 A
        MOV     B，J            ；取被乘数十位到 B
        MUL     AB              ；（J）×（M）
        MOV     LOMJ，A         ；存部分积低 8 位
        MOV     HIMJ，B         ; 存部分积高 8 位
        MOV     RES0，LOML      ; 积个位存于 RES0 中
        MOV     A，HIML         ；
        ADD     A，LOMK         ；（A）=（HIML）+（LOMK）
        MOV     RES1，A         ；存积的次低位于 RES1 中
        MOV     A，HIMK         ；
        ADDC    A，LOMJ         ；（A）=（HIMK）+（Cy）+（LOMJ）
```

```
        MOV     RES2，A             ；存积的次高位于 RES2 中
        CLR     A                   ；（A）清 0，处理进位位 Cy
        ADDC    A，HIMJ             ；（A）=（HIMJ）+（Cy）
        MOV     RES3，A             ；存积的最高位于 RES3 中
        POP     B
        POP     ACC
        RET
```

主程序调用子程序时需定义子程序中的变量地址，且被乘数和乘数已放入对应单元中：

```
        J       EQU     30H
        K       EQU     31H
        L       EQU     32H             ；被乘数
        M       EQU     33H             ；乘数
        HIML    EQU     34H
        LOML    EQU     35H
        HIMK    EQU     36H
        LOMK    EQU     37H
        HIMJ    EQU     38H
        LOMJ    EQU     39H             ；部分积
        RES3    EQU     3AH
        RES2    EQU     3BH
        RES1    EQU     3CH
        RES0    EQU     3DH             ；积
        ORG     0000H
        SJMP    MAIN
        ORG     0040H
MAIN：  MOV     SP；#60H
        LCALL   MULJKLM
STOP：  SJMP    STOP
MULJKLM：PUSH   ACC
        ……                          ；子程序主体部分，略
        RET
        END
```

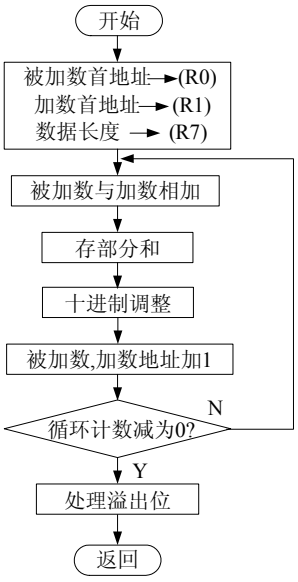

图 04-1 习题5-7程序流程

与【例 4-18】的程序相比，本程序显然没有通用性。本程序只适用于 3 字节和 2 字节被乘数与单字节乘数的乘法运算，而两个单字节数的运算，当然直接用乘指令了。本程序的另一个缺点是，RAM 空间占用量巨大。

4-7 解：

（1）子程序流程如图 04-1 所示。

（2）参考子程序如下：

```
BCDADD：MOV     R0，#ADDR0          ；被加数首址存入间址寄存器 R0 中
        MOV     R1，#ADDR1          ；加数首址存入间址寄存器 R1 中
        MOV     R7，#DATANUM        ；数据长度存入 R7 中
        CLR     C                   ；清 Cy
AGAIN：  MOV     A，@R0              ；取被加数
        ADDC    A，@R1              ；（A）= 被加数+加数
        DA      A                   ；十进制调整
        MOV     @R0，A              ；存此位部分和
```

```
        INC     R0                  ；指向下一个被加数与加数（内存单元地址）
        INC     R1
        DJNZ    R7，AGAIN           ；全部数据没处理完循环，否则向下执行
        MOV     00H，C              ；进位值处理
NOCY：  RET                         ；子程序返回指令
```

位地址单元用 00H，是片内 RAM 的 20H 单元的第 0 位。

4-8

解：参考子程序如下：

```
CVERT1：  ADD     A，#90H           ；入口 A 中为半字节十六进制数，高 4 位为 0
          DA      A
          ADDC    A，#40H
          DA      A                 ；出口：（A）= 转换好的 ASCII 码值
          RET
```

用此算法编写的程序，指令精练，也不用转移判断，执行的快，但算法不易想到。而教材【例 4-21】的算法直观，直接根据两种代码的数量关系得到算法，一般人都能想到，一样能完成任务。因此，不要为想不到高级算法而苦恼，首先要有一个解决问题意识，并解决它。不会的东西可以学，知识积少成多。好的算法来自对要解决的问题的深刻理解之上。如果片面地强调好的算法，而忽视对问题基本属性的研究，解决问题的能力总是提高不了的。

4-9 解：修改后的子程序如下：

```
CVERT2：  CJNE    A，#3AH，NOEQU    ；入口 A 中为 ASCII 值
          SJMP    COMM              ；不可能情况，视为出错，直接返回
NOEQU：   JC      SMALL             ；ASCII 值<3AH 减 30H
          SUBB    A，#37H           ；ASCII>39H 减 37H
          SJMP    COMM
SMALL：                             ；CLR     C
          SUBB    A，#2FH           ；SUBB A，#30H
COMM：    RET                       ；出口：（A）= 转换好的 ASCII 码值
```

删除和修改的指令已注释，并将文字加重。

习题 4-10 训练题

解：

参考程序如下：

```
        D       EQU     P1.0        ；P1.0 为输出，P1.1~P1.7 为（开关）输入
        ORG     0000H               ；主程序起点
        LJMP    START
        ORG     0040H               ；主程序从这里开始
START：  MOV     SP，#5FH           ；堆栈区设置
        CLR     D                   ；先灭灯，绿色环保
        MOV     A，P1
        ANL     A，#FEH             ；本程序可以扩展到 7 个开关
        MOV     C，P                ；得到上电后开关状态组合后的奇偶性
        CLR     01H                 ；作为上次检测的奇偶性，存于 01H
        CLR     F0                  ；F0 与 D 同步，为输出状态
START1：MOV     C，F0
```

```
                MOV     D，C                ；
                MOV     A，P1               ；循环检测开关状态组合后的奇偶性
                ANL     A，#0FEH
                JB      P，PEQU1`
                JNB     01H，START1         ；开关状态组合后的奇偶性没变继续检测
                SJMP    CHENG               ；开关状态组合后的奇偶性变化则转移
    PEQU1：     JB      01H，START1         ；开关状态组合后的奇偶性没变继续检测
    CHENG：     CPL     D                   ；开关状态组合后的奇偶性变化
                CPL     F0                  ；与 D 保持同步
                SJMP    START1              ；继续循环检测
                END
```

这样做系统的可靠性有所增加。例如，开关可能长时间没人切换，但偶然的干扰可能会使灯的状态发生变化，但如果不增加周期输出，这个干扰是不能立即解除的。由于此任务的扫描周期很短，所以周期输出，能将干扰立即排除，有利于系统可靠性的提高。

训练题 4-11 解：参考 C51 程序如下：

```
#include <intrins.h>
#include <REGSTC51.h>
#define uchar unsigned char
#define uint unsigned int
#define D P1_0
uchar idata w_command,count;
static bdata uchar sclkdata;
sbit sclkdata7 = sclkdata^7;
uchar code *point_l;                // 指针
bit   D_state                       // 灯输出状态
bit JO_JY(uchar P1_state)           // 判断一个 8 位数的奇偶性函数
{
    uchar i,add_num = 0;
    bit P1_jo = 0;
    sclkdata = P1_state;
    for (i=0;i<8;i++)
    {
        if(sclkdata7)
            add_num++;
        sclkdata <<= 1;
    }
    if (add_num%2 !=0)
        P1_jo = 1;
    return P1_jo;
}
main()
{
    uchar data P1_work;
    bit JO_result,T_save;
START:
```

```
    D = 0;                          // 先灭灯，绿色环保
    D_state   = 0;
    P1_work = P1;
    P1_work &= 0xfe;                // 本程序可以扩展到 7 个开关
    JO_result = JO_JY(P1_work);
    T_save = JO_result;             // 得到上电后开关状态组合后的奇偶性
START1:
    D = D_state;                    // 每循环一次，对灯的控制位输出
    P1_work = P1;
    P1_work &= 0xfe;
    JO_result = JO_JY(P1_work);     // 循环检测开关状态组合后的奇偶性
    if (T_save != JO_result)        // 开关状态组合后的奇偶性变化
    {
        D =~D;                      // 灯输出状态取反
        D_state = D;                // 存新的灯输出状态
        T_save = JO_result;         // 存入新开关状态组合后的奇偶性
    }
    goto      START1;               // 继续循环检测
}
```

习题 4-12 训练题解：

方法是： 用 DPH 分时指定源、目的地址高 8 位，由于，源、目的地址是同时变化的，所以改变 DPL 的值，源、目的地址将同时改变。这就是本题能用一个 DPTR 实现其题目要求的条件。

参考子程序如下：

```
            ORG       0000H
            SJMP    MAIN
MAIN:       MOV     SP,#5FH
            LCALL   MXRAMD
            SJMP    $
MXRAMD:     MOV     R7,#20H             ;循环计数值
            MOV     DPL, #00H           ;源、目的数据首址
LOOP1:      MOV     DPH,#20H            ;源数据首址，循环体头
            MOVX    A,@DPTR             ;取数
            MOV     DPH,#30H            ;目的数据首址
            MOVX    @DPTR,A             ;完成一个数的转移
            INC     DPL                 ;指针加 1，指向下一源单元
            DJNZ    R7,LOOP1            ;循环体尾
            RET
```

习题 4-13 训练题解：

参考子程序如下：

```
MULADD：    MOV     R0，#ADDR0          ；数组首址存入间址寄存器 R0 中
            MOV     R7，# n             ；数据长度存入 R7 中
            MOV     B，#00H             ；和高位清零
            CLR     A                   ；清累加器
AGAIN：     ADD     A，@R0              ；（A）= 数组和
```

```
        JNC     NOCY                ；判断有进位吗？
        INC     B                   ；有进位处理
NOCY：  INC     R0                  ；指向数组的下一个单元
        DJNZ    R7，AGAIN           ；全部数据没处理完循环，否则向下执行
        RET                         ；子程序返回指令
```

区别：数据在片内和在片外，使用的数据指针不同。

习题 4-14 训练题 略。

习题 4-15 训练题 略。

习题 4-16 训练题 解：

在【例 4-19】程序 MAIN： MOV SP，#5FH 指令后，先加上除数值的判断部分，正常进入计算，不正常则对 B 赋值，说明问题类型，返回。

增加的程序部分如下：

```
MAIN：  MOV     SP，#5FH
        MOV     R1，#50H            ；设除数的首地址为 50H
        MOV     B，#00H             ；00H 表示除数值正常
        MOV     A，ADDR2            ；得到除数最高位地址
        MOV     R1，A
        MOV     R7，#n
LOOP：  MOV     A，@R1
        JNZ     MAIN1
        INC     R1
        DJNZ    R7，LOOP
        MOV     B，#0FFH            ；除数为 0
        RET
MAIN1：  ……                         ；接【例 4-19】程序 MAIN 部分
```

刘焕成 2010 年 8 月 2 日再次校稿。
2011 年 03 月 26 日再次校稿。
2011 年 05 月 08 日校稿 3。
2011 年 11 月 01 日校稿。