# 计算机组成原理　期末模拟题

ywy_c_asm

November 2022

**免责声明：本试题仅供参考，且不保证不会有奇怪的差错**

本试题主要参考考研真题与教材习题，以及作者本人的一些脑洞，题量和难度可能**稍大于**期末考试，知识点覆盖范围较为综合，请按需使用。

## 1　选择题

1. 下列哪个**不是**冯诺依曼机的基本特点（　　）

A. 指令和数据以二进制代码形式存储，有独立的指令存储器，取指效率高
B. 指令一般顺序执行
C. 计算机以运算器为中心
D. 输入设备和输出设备是计算机的基本部件



2. 关于中央处理器 CPU，下列说法**错误**的是（　　）

A. 在五大基本部件中，控制器和运算器构成 CPU
B. CPU 在执行乘法或除法时，结果可能需要保存在多个寄存器中
C. CPU 内寄存器之间的数据移动都是可以使用机器指令实现的
D. 寄存器 PC 和 IR 都是控制器的一部分

3. 现有一主频 10GHz 的 CPU，它的指令系统只有 5 条指令，这几条指令的 CPI 和平均使用频率如下表所示：则下列说法**正确**的是（　　）

| 指令名称 | CPI | 平均使用频率 |
|---|---|---|
| mov | 3 | 50% |
| add | 6 | 30% |
| mul | 10 | 5% |
| div | 30 | 5% |
| jmp | 6 | 10% |

I. 该 CPU 的指令周期是 $10^{-10}$s

II. 该 CPU 的平均 CPI 是 5.9

III. 若可采取某种方式将某条指令的 CPI 降为原来的一半，则改进执行最慢的 div 指令更合适

IV. 该 CPU 的平均指令执行速度约为 1695MIPS

A. I,II,III,IV　　B. 仅 II,IV　　C. 仅 II,III,IV　　D. 仅 II

4. 关于流水线 CPU，下列说法**错误**的是（　　）

A. 不同指令的不同阶段可以并行执行，这提升了时间效率

B. 若一条指令的五个执行阶段分别需要 20ns,5ns,10ns,15ns,20ns 的处理时延，由于理论上流水线 CPU 的时钟周期取决于最慢阶段的限制，至少需要提供频率为 50MHz 的时钟信号才能正常工作

C. 在顺序执行的流水线中，若前一条指令修改了通用寄存器 $R_0$，后一条指令需要使用 $R_0$ 的值，则出现先写后读（RAW）的数据相关

D. 现代 CPU 中，L1Cache 设计为指令 Cache 和数据 Cache 分立的形式，是为了解决流水线中的结构冲突

5. 小 Y 希望通过一不可靠信道向你发送某 4 位二进制串，采取基于配奇原则的 1 位纠错汉明码进行传输，假定传输过程中至多出现 1 位错，你收到 1100001（下标从高位开始），则小 Y 希望发送的数据为（　　）

A. 1001　　B. 0001　　C. 0101　　D. 0000

6. 现有一真值为-0.11010 的二进制小数以反码形式保存在某 8 位寄存器中，则该寄存器的内容为（　　）

A. 0xE8　　B. 0x94　　C. 0x97　　D. 0x9F

7. 某种基数为 2 的 16 位浮点数类型格式为，阶码字段占 6 位（包括符号位），以补码表示，尾数字段占 10 位（包括符号位），以原码表示，且不强制要求规格化，则下列说法中**错误**的是（　　）

A. 该浮点数类型可以精确表示整数 2048

B. 该浮点数类型能表示的最大的实数为 $2^{31}\times(1-2^{-9})$

C. 该浮点数类型能表示的最小的实数为 $2^{-41}$

D. 存在实数 $x$，$x$ 在该浮点数类型下的机器表示不唯一

8. 对于一个真值为-1100 的二进制整数，将其算术右移一位，则其在 5 位原码、5 位反码、5 位补码下的结果分别为（　　）

A. 1,0110　1,1001　1,1010

B. 1,1110　1,0001　1,1010

C. 0,1110　0,1001　0,1010

D. 1,1110　0,1001　0,1010

9. 关于计算机内部计算加减法的硬件器件，下列说法**正确**的是（　　）

I. 原码加减一般需要同时实现加法器和减法器，而补码加减只需要实现加法器

II. 串行进位加法器比并行进位加法器更慢，因为硬件电路更复杂，导致进位信号传播较慢

III. 运算速度：双重分组跳跃进位加法器 < 单重分组跳跃进位加法器

IV. 加法器中每一位结果和这一位的输入以及前面的进位有关

A. I,II,III,IV　　B. I,III,IV　　C. I,II,IV　　D. I,IV

10. 下列关于 RISC 的说法中，**错误**的是（　　）

A. 一般使用流水线实现，可降低指令的平均 CPI

B. 指令功能比 CISC 弱，但使用频率普遍更高，实现也更简单

C. 实现相同的程序，RISC 机花费的指令数比 CISC 机可能更多，程序规模更大

D. 由于 RISC 更加精简，故指令类型、寻址方式和通用寄存器个数均少于 CISC，易于设计

11. 某采取多重间接寻址的指令中地址码字段占 10 位，机器字长和存储字长均为 16 位，则该指令的寻址范围为（　　）个存储单元。

A. $2^{10}$　　B. $2^{16}$　　C. $2^{15}$　　D. $2^{9}$

12. 某字节寻址且支持多重中断的 CPU 具有 5 个外部中断源 A,B,C,D,E，响应优先级从高到低为 A>B>C>D>E，处理优先级从高到低为 D>C>E>A>B。CPU 内部具有堆栈指针寄存器 SP，一次单字节压栈操作为，将一个字节存入 M(SP)，并令 SP++。CPU 执行中断隐指令时要将 4 字节信息压栈，当中断返回时会出栈恢复（假设中断服务程序本身不使用堆栈）。某个时刻 CPU 正在正常工作，SP=2000，允许外部中断，所有中断源皆未屏蔽，此时若 5 个中断源同时发起中断请求，则整个过程中 SP 的最大值为（　　）

A. 2008　　B. 2012　　C. 2016　　D. 2004

13. 某采取微程序控制方式的 CPU 共有 9 条指令，平均每条指令对应 7 条微指令，用于取指的公共微程序包含 3 条微指令，采取下地址字段法确定下条微指令地址。每条微指令采取字段直接编码方式，共有 34 个微命令，构成 5 个互斥类，分别包含 8,3,12,5,6 个微命令，则 CMDR 寄存器位数至少为（　　）

A. 21　　B. 22　　C. 23　　D. 24

14. 下列关于存储器的说法中，**正确**的是（　　）

I. U 盘可以快速读写，属于随机存储器

II. 1K×8 位 SRAM 芯片总共有 13 根地址和数据引脚

III. DRAM 集成度更高，成本更低，功耗也更低，因此适合集成到 CPU 内部作为缓存

IV. 对于 DRAM 来说，分散刷新由于不存在死时间，而集中刷新和异步刷新存在存储器无法接受读写操作的死时间，因此分散刷新相对更加合适

A. 仅 IV　　B. I,II,IV　　C. II,III　　D. 都不对

15. 有一存储器采取四体交叉编址方式，支持流水线连续地址访问。若 CPU 重复连续访问一段大小为 15 个存储单元的内存区，重复 400 次，总线传输周期 20ns，存取周期 80ns，仅考虑访存带来的时间开销，则最短时间为（　　）

A. 144000ns　　B. 128040ns　　C. 128060ns　　D.129000ns

16. 某计算机采取 32 位主存地址，使用总容量为 32KB 的八路组相联 Cache，块大小 16B，采取写回策略，则该 Cache 一行的总位数至少为（　　）

A. 1176bits　　B. 1184bits　　C. 1192bits　　D.1200bits

17. 关于计算机的 I/O 系统，下列说法**正确**的是：（　　）

I. I/O 接口通过总线与 CPU 相连接
II. I/O 端口是 CPU 和外设之间的界面，控制 CPU 和外设之间的信息交换
III. I/O 接口发向 CPU 的中断请求信号 INTR 是专门为程序中断 I/O 方式服务的
IV. 程序中断方式克服了程序查询方式的低效率缺点，可以实现 CPU 与外设的完全并行工作

A. I,II　　B. I,III　　C. I,IV　　D. 仅 I

18. 关于 DMA，下列说法**错误**的是：（　　）

A. 适合用于高速外设的大批量数据传输
B. DMA 请求一般有更高的优先级
C. 由于 DMA 可直接访问主存，故在数据传输中不需要干扰 CPU 的正常工作，只需在一批数据传输完成后向 CPU 发起中断进行通知
D. DMA 是与以运算器为中心的冯诺依曼机模型相违背的

19. 下列英文缩写为总线标准名称的是（　　）

I. ISP　　II. ISA　　III. RS-232C　　IV. VESA　　V. PCI　　VI. CPI

A. I,II,III,IV,V　　B. II,III,IV,V　　C. II,III,IV,V,VI　　D. II,IV,V

20. 关于虚拟存储系统，下列说法**错误**的是：（　　）

A. 主要用于解决主存容量不足的问题
B. 作为一种缓存机制，不命中率远低于 Cache
C. 相比于不采取虚拟存储方式的计算机，CPU 访存更慢
D. 通过虚拟存储系统，辅存被映射到内存地址空间中，这使得 CPU 可通过访存指令直接与辅存交换数据

## 2 填空题

21. 6 位补码整数 1,10110 的相反数在 6 位补码表示下为 _____，在 6 位原码表示下为 _____。

22. 对于存储器而言，存取时间被定义为 _____，存取周期被定义为 _____。

23. 总线的异步通信可采取不互锁、半互锁、全互锁三种方式，其中最快的是 _____，最慢的是 _____，最可靠的是 _____。

24. 在你所学的 10 种寻址方式中，获取操作数最快的寻址方式是 _____，占用指令位数最少的寻址方式是 _____，可用于数组访问的寻址方式是 _____，可用于地址空间划分的寻址方式是 _____，可用于条件转移的寻址方式是 _____。

25. 写出 3 种 Cache 的替换策略 _____、_____、_____。

26. 现有一采取多级时序系统的 CPU，主频 20GHz，每条指令平均花费 5 个机器周期，每个机器周期平均花费 8 个时钟周期，则其平均指令执行速度为 _____MIPS。

27. 考虑一个具有 64 位 VA 以及 48 位 PA 的分页虚拟存储系统，页大小 8192B，则 VPO=_____，PPO=_____，VPN=_____，PPN=_____。

28. 有一旧式硬盘，平均寻道时间 5ms，转速 6000rpm，具有 16 个盘片，每个盘片外直径 192mm，内直径 32mm，道密度 8 道/mm，每个磁道具有 1024 个扇区，每个扇区承载 512B 数据，则其总容量为 _____GB，平均寻址时间为 _____ms，最大数据传输率为 _____。

29. 在采取水平型微指令的控制单元中，CM 的作用为 _____，CMAR 的作用为 _____，CMDR 的作用为 _____。

30. 假设 CPU 处理一次中断皆需 6$\mu$s，现一外设需要将 1000 字数据送入主存，存取周期为 20ns，外设准备数据的时间忽略不计。则在理想情况下，若使用基于周期窃取的 DMA 方式（每批数据 200 字），因传输而占用处理器的时间至多为 _____，若采取程序中断方式，则时间为 _____。

## 3 简答题

31. 下图分别为支持程序查询方式和支持程序中断方式的 I/O 接口电路，根据这两幅图，分别简述程序查询方式和程序中断方式的全过程。

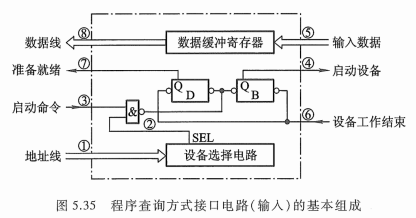


图 5.35 程序查询方式接口电路(输入)的基本组成

图 1: 支持程序查询方式的 I/O 接口电路

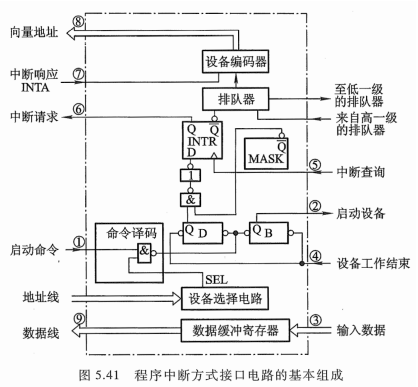


图 5.41 程序中断方式接口电路的基本组成

图 2: 支持程序中断方式的 I/O 接口电路

32. 简述提升存储器访问速度的若干思路（至少 3 条）。



# 4 计算题

33. 已知 $[x]_{补}=0,10101$，$[y]_{补}=1,01101$，使用 1 位 Booth 算法求 $[x\times y]_{补}$，写出详细过程。

34. 已知 $x=-0.01101$，$y=0.10101$，使用原码加减交替法求 $\left[\frac{x}{y}\right]_{原}$（求出商值即可），写出详细过程。

35. 设浮点数阶码取 6 位，尾数取 6 位（均包含符号位），皆为补码表示，计算 $[2^{31}\times\frac{27}{32}]+[2^{30}\times\frac{13}{16}]$，写出详细过程。



## 5 分析与设计题

36. 某 CPU 采取了如图所示的片内总线数据通路设计：

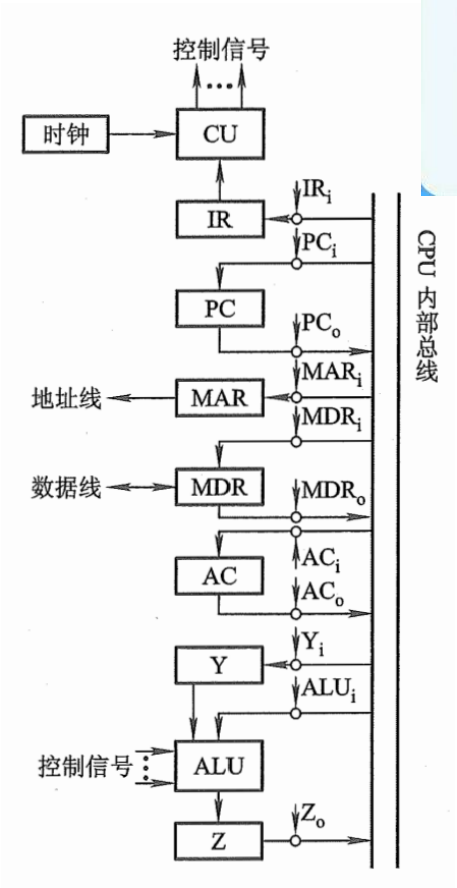



图 3: 该 CPU 的数据通路

其中，除了 CU 和 ALU 外，其余方框都表示寄存器。当 $XXX_i/XXX_o$ 信号有效时，对应的数据通路会处于开启状态，对应该寄存器 $XXX$ 的一次输入/输出。回答下列问题：

(1) 该 CPU 的指令系统中存在一条特殊加法指令 “CADD X”，其中 X 为地址码，作用为将地址为 (AC)+X 的存储单元中的数与 AC 相加，结果存放到 AC 中，即 AC←(AC)+M((AC)+X)。请写出这条指令包括取指在内的所有微操作与对应的操作信号（包括访存信号 R、W，尽管图上未画出）。

(2) 小 Y 认为图中的寄存器 Z 没有什么实际意义，反正也不是程序员能访问的通用寄存器，有没有都无所谓。你同意他的观点吗？给出理由。

(3) 简述总线型结构数据通路和非总线型结构数据通路的特点和差异。假如你是一名 CPU 设计人员，你希望采取哪种数据通路结构？给出理由。

37. 某 CPU 指令字长 16 位，具有 32 个通用寄存器，指令地址码 6 位，现在需要为其设计指令系统。要求：具有 10 条双地址码指令，11 条单地址码指令，10 条单地址码 + 单寄存器指令，6 条单寄存器指令，3 条双寄存器指令。

(1) 采取操作码扩展技术，给出你设计的指令操作码（要求：指令的操作码位于指令字的高位）。

(2) 该指令系统最多还可以设计几条零地址码指令？

38. 1978 年，Intel 公司推出了著名的 8086 CPU，该 CPU 采用了约 3 万个 3$\mu$m 技术的晶体管，可工作在最高 10MHz 的时钟频率下。尽管这些 40 多年前的技术相比于今日而言早已略显陈旧，但 8086 在计算机发展史上是具有里程碑意义的——它标志着 x86 和 IBM PC 兼容机这两种影响深远且沿用至今的架构的诞生。甚至，现代的大部分 x86 架构 CPU 为了兼容，在最初上电时需要以 8086 的方式工作（称之为实模式），而 8086 定义的的所有指令都仍可运行于现代的所有 x86 架构 CPU 上。根据题目提供的 8086 的有关数据与工作机制以及你在本课程中的所学知识，回答下列问题：

(1) 8086 具有 40 根引脚，下图是其外部引脚布局：

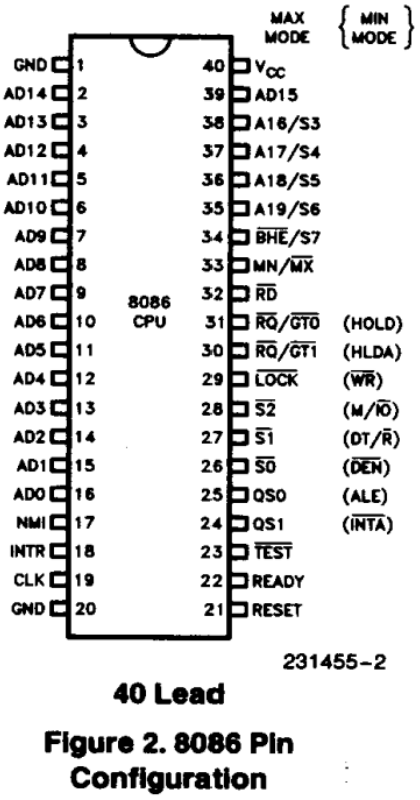



图 4: 8086 的外部引脚布局

8086 的机器字长和数据总线宽度均为 16 位，采取字节编址方式，使用 20 位地址总线，则其按字寻址的范围大小为 _____。8086 采取引脚复用技术，引脚 $AD_0 \cdots AD_{15}$ 在总线周期的不同节拍下，既可用作 16 位数据总线（$D_0 \cdots D_{15}$），也可用地址总线的低 16 位（$A_0 \cdots A_{15}$），引脚 $A_{16} \cdots A_{19}$ 为地址总线的高 4 位。在本课程中你还学过哪种使用了类似的引脚复用技术的器件？这种技术有何优缺点？

(2) 8086 的一些与总线读写操作相关的控制信号引脚如下表所示：

| 引脚（信号）名称 | 说明 |
|---|---|
| $\overline{\text{RD}}$ | 读使能信号，低电平有效 |
| $\overline{\text{WR}}$ | 写使能信号，低电平有效 |
| M/$\overline{\text{IO}}$ | 访存使能信号，高电平表示读写主存，低电平表示读写 I/O 端口 |
| $\overline{\text{BHE}}$ | 指示高 8 位数据总线 $D_8 \cdots D_{15}$ 是否有效，与奇偶分体有关 |

不难发现，8086 的访存和读写 I/O 端口都要使用同样的地址总线和数据总线，通过访存使能信号 M/$\overline{\text{IO}}$ 进行操作区分。由此可以推断出 8086 对于 I/O 端口采取什么样的编址方式？应使用什么样的指令访问 I/O 端口？这样设计有何优缺点？

(3) 8086 可以工作在多种模式下，在本题中我们考虑最简单的最小工作模式（Minimum Mode）。在该模式下，8086 的一个用于访问存储器或 I/O 端口的总线周期一般由四个节拍 $T_1$,$T_2$,$T_3$,$T_4$ 构成，这几个节拍的操作如下表所示：

| 节拍 | 操作 |
|---|---|
| $T_1$ | 将地址送上引脚 $AD_0 \cdots AD_{15}$ 和 $A_{16} \cdots A_{19}$ |
| $T_2$ | 撤销地址信号，若为写操作则将数据送上引脚 $AD_0 \cdots AD_{15}$ |
| $T_3$ | 发出控制信号，数据开始写入或读到总线上 |
| $T_4$ | 撤销信号，结束总线周期 |

假设 8086 以 10MHz 时钟频率工作，总线周期的节拍与时钟周期一致，求理想情况下 8086 的最大总线带宽。

(4) 事实上，8086 的一个总线周期可以在某些情况下具有多个节拍。8086 的 READY 引脚具有向总线周期中插入等待节拍 $T_w$ 的功能，在节拍 $T_3$ 结束时的时钟边沿，8086 会检测 READY 引脚，若为低电平，则插入一个等待节拍 $T_w$，此后每个时钟周期结束时都会检查 READY，不断插入 $T_w$，直至 READY 为高时才会结束等待，进入节拍 $T_4$ 开始结束总线周期。时序图如下所示：

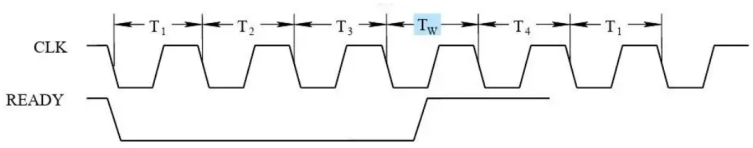



图 5: 插入等待节拍 $T_w$ 的 8086 总线周期时序图

由此可知，8086 的总线通信采取 _____ 方式，你认为 8086 这样设计的原因可能是什么?

(5) 考虑 8086 的两个连向 I/O 外设的引脚 HOLD 和 HLDA，其中 HOLD 为输入信号（正常状态下应输入低电平），HLDA 为输出信号。当 CPU 在一个总线周期内进行总线操作时，若检测到 HOLD 引脚输入了高电平，则 CPU 会将 HLDA 引脚输出高电平，并在下一个总线周期时不进行任何操作，且令数据地址总线引脚 $AD_0 \cdots AD_{15}$ 和 $A_{16} \cdots A_{19}$ 处于高阻态（可简单认为 CPU 与总线断开连接）。根据所学知识，你认为这两个引脚的作用与应用场景是什么?

(6) 8086 CPU 的指令系统定义了最原始的 x86 指令集，其中相等则转移指令 “je X” 可采取相对转移的方式，该指令字长 16 位，指令格式如下：

| 0-7 位 | 8-15 位 |
|---|---|
| 0x74 | X |

其中，X 为位移量，采取 8 位补码表示，则位移量的范围为 _____。8086 在执行相对转移型指令时，会在**取指后**的 PC 的基础上加上偏移量。假设这条 “je X” 指令位于地址 0x3000 处，希望在条件成立的情况下跳转到 0x2FF0，则指令中的字节 X 应为 _____（十六进制表示）。你认为转移指令采取这种相对寻址的好处是什么? 另外，现代 x86-64 CPU 可以完全兼容 40 年前为 8086 设计的指令系统，操作码和指令格式甚至可以完全保持不变，例如这条双字节的 je 指令仍然可以被 Intel Core i9 CPU 正常执行，而这两种相差 40 年的 CPU 显然几乎不可能有相同的芯片构造与内部工艺（当然，以及二者悬殊的性能差距）。根据所学知识，解释这一现象的原理。

(7) 8086 的 INTR 引脚接受外设发来的中断请求信号，8086 会在“适当的时候”检查 INTR 引脚，若其为高电平，且标志寄存器的 IF 标志位为 1（允许中断），则 8086 响应中断，将当前的 PC 和标志寄存器入栈，将 IF 置 0，读取总线上的中断向量号，查找存储器中的中断向量表（从地址 0x00000 开始的 1KB 内存空间），跳转到相应的中断服务程序入口地址。你认为“适当的时候”应该是什么时候？若你希望使得 8086 进行多重中断，则中断服务程序应该进行什么操作？

(8) 根据第 (1) 问我们知道 8086 采取了地址/数据引脚复用，因此 8086 的外围电路需要负责进行引脚的解复用，分离出真正的地址总线 $A_0 \cdots A_{19}$ 和数据总线 $D_0 \cdots D_{15}$。顺带一提，这一解复用过程可在外部设置地址锁存器，通过 8086 的地址锁存引脚 ALE 控制，暂存 CPU 在节拍 $T_1$ 发出的地址来实现（不必担心，解复用跟本题并没有什么实际关系）。现在考虑 $A_0 \cdots A_{19}$ 和 $D_0 \cdots D_{15}$ 这 34 根解复用后的地址和数据总线，以及 8086 用于读写操作的控制信号 $\overline{\text{RD}}$、$\overline{\text{WR}}$、M/$\overline{\text{IO}}$、$\overline{\text{BHE}}$。根据第 (2) 问我们已经知道 $\overline{\text{BHE}}$ 与 8086 的奇偶分体存储有关。8086 采取奇偶分体的原因是其具有 16 位数据总线，而早期的存储器芯片或外设一般都只有 8 根数据总线，因此，8086 规定，奇地址内存单元（大小为一个字节）和偶地址内存单元应该对应不同的存储体，且偶存储体使用低 8 位数据总线 $D_0 \cdots D_7$，奇存储体使用高 8 位数据总线 $D_8 \cdots D_{15}$，$\overline{\text{BHE}}$ 信号用来控制高 8 位数据总线是否有效（即奇存储体是否工作）。8086 可以通过 $\overline{\text{BHE}}$ 以及地址总线最低位 $A_0$ 的不同组合方式来决定字或字节的读写，具体如下表所示：

| $A_0$ | $\overline{\text{BHE}}$ | 读写方式 | 偶存储体工作状态 | 奇存储体工作状态 |
|---|---|---|---|---|
| 0 | 0 | 从偶地址读写一个字 | 工作 | 工作 |
| 0 | 1 | 从偶地址读写一个字节 | 工作 | 不工作 |
| 1 | 0 | 从奇地址读一个字节 | 不工作 | 工作 |
| 1 | 1 | 无效操作 | 不工作 | 不工作 |

8086 的一种地址空间分配方案如下图所示：

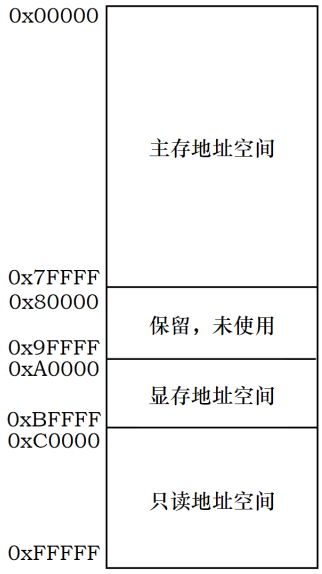



图 6: 8086 的一种地址空间分配方案

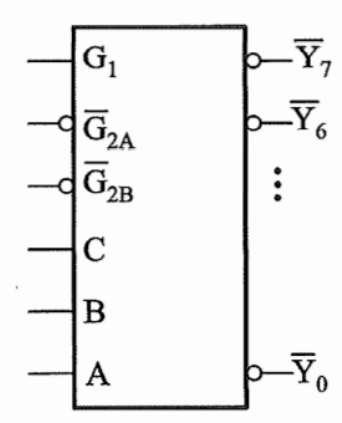



图 7: 74138 译码器芯片

其中，主存地址空间和显存地址空间都是可读写的，且显存地址空间与屏幕显示有关（事实上，在特定的显示模式下，向这个地址空间中写入像素或字符会立刻呈现在屏幕上，但本题为了简化，仅将其视为一段普通的可读写地址空间），而只读地址空间对应主板上的 BIOS ROM 芯片，存放 CPU 的初始化指令（8086 在上电复位后会自动从只读地址空间的 0xFFFF0 处开始执行指令）以及其它重要的系统设置数据。现在，给定如图所示的 74138 译码器芯片，以及若干容量为 64K×8 位、128K×8 位和 256K×8 位的 SRAM 芯片以及 EEPROM 芯片，以及门电路若干，你需要为 8086 实现**外围存储器电路**。要求：显存地址空间和主存地址空间**不得**分配在相同的存储体内，且你只需要考虑连向 8086 的地址总线 $A_0 \cdots A_{19}$ 和数据总线 $D_0 \cdots D_{15}$ 以及控制信号 $\overline{\text{RD}}$、$\overline{\text{WR}}$、M/$\overline{\text{IO}}$、$\overline{\text{BHE}}$。若给定的地址落在保留空间中，任何存储芯片都**不得**工作。给定的 SRAM 芯片具有控制引脚 $\overline{\text{RD}}$、$\overline{\text{WR}}$、$\overline{\text{CS}}$，给定的 EEPROM 芯片具有控制引脚 $\overline{\text{RD}}$、$\overline{\text{PD}}$/Progr、$\overline{\text{CS}}$。

(9)（开放型问题，可不回答）根据上述 (1)-(8) 对 8086 这一 40 年前生产的芯片的分析，你是否对《计算机组成原理》这门课程有了新的认识和感受？如果给你足够的 74 系列逻辑芯片和存储器芯片，根据《数字逻辑与数字系统设计》课程以及本课程所学知识，查阅相关资料（例如 8086 的芯片手册），你能否为 8086 设计一个完整的外围电路，构造一个具有控制外设能力的微型计算机系统？试阐述你的设计思路。

# 微信公众号：网盘计划

## 1.项目初衷

鉴于（1）哈工大各类 QQ 群内学习资料多且繁杂，而文件文字太多会导致文件被 tx 屏蔽或者降低 QQ 群信用星级；（2）校内诚信复印和纸张记忆垄断；（3）很多营销号在卖资料且售价很高；（4）学长学姐的自编材料很好，还想分享给下一届；等问题，网盘计划应运而生！哈尔滨工业大学网盘计划**旨在将窝工的各类学习资料进行归类整理，并且以网盘的形式发出来**，历时三年，现已大成，自费扫描了上百份校内复印店试题文档和各类电子教材实验报告等，归类整理了 50 多个 G 的学习资料无偿分享给大家，如果您觉得网盘计划对您有帮助的话，可通过以下方式进行打赏。



## 2.网盘计划进度（密码 1920）

网盘计划全文


微信公众号二维码

哈工大PPT模板（密码1920）

***腾讯自动屏蔽以上链接，请用浏览器扫一扫***

**（关注公众号“网盘计划”回复课程名称即可获取全部资源!!!）**

# 答案解析

ywy_c_asm

November 2022

## 1 选择题

1. 下列哪个**不是**冯诺依曼机的基本特点（ A ）

A. 指令和数据以二进制代码形式存储，有独立的指令存储器，取指效率高
B. 指令一般顺序执行
C. 计算机以运算器为中心
D. 输入设备和输出设备是计算机的基本部件

**解析**：若指令与数据分开存储，即设置独立的指令存储器，那就不是冯诺依曼结构了，违背了指令与数据以同等方式存储的原则，那叫哈佛结构。单片机等嵌入式系统经常这么干。

2. 关于中央处理器 CPU，下列说法**错误**的是（ C ）

A. 在五大基本部件中，控制器和运算器构成 CPU
B. CPU 在执行乘法或除法时，结果可能需要保存在多个寄存器中
C. CPU 内寄存器之间的数据移动都是可以使用机器指令实现的
D. 寄存器 PC 和 IR 都是控制器的一部分

**解析**：注意寄存器和通用寄存器的区别，不是所有 CPU 内部的寄存器都能用机器指令访问。

3. 现有一主频 10GHz 的 CPU，它的指令系统只有 5 条指令，这几条指令的 CPI 和平均使用频率如下表所示：则下列说法**正确**的是（ B ）

| 指令名称 | CPI | 平均使用频率 |
|---|---|---|
| mov | 3 | 50% |
| add | 6 | 30% |
| mul | 10 | 5% |
| div | 30 | 5% |
| jmp | 6 | 10% |

I. 该 CPU 的指令周期是 $10^{-10}$s
II. 该 CPU 的平均 CPI 是 5.9
III. 若可采取某种方式将某条指令的 CPI 降为原来的一半，则改进执行最慢的 div 指令更合适
IV. 该 CPU 的平均指令执行速度约为 1695MIPS

A. I,II,III,IV　　B. 仅 II,IV　　C. 仅 II,III,IV　　D. 仅 II

**解析**：指令周期可不是时钟周期，I 不对。CPU 的平均 CPI=3×50%+6×30%+10×5%+30×5%+6×10%=5.9，平均指令执行速度 $=\frac{10\text{GHz}}{5.9}$=1695MIPS，II 和 IV 正确。而 add 指令在平均 CPI 中占 6×30%=1.8，div 指令在平均 CPI 中占 30×5%=1.5，因此改进 add 指令更合适，III 不对。

4. 关于流水线 CPU，下列说法**错误**的是（ B ）

A. 不同指令的不同阶段可以并行执行，这提升了时间效率

B. 若一条指令的五个执行阶段分别需要 20ns,5ns,10ns,15ns,20ns 的处理时延，由于理论上流水线 CPU 的时钟周期取决于最慢阶段的限制，至少需要提供频率为 50MHz 的时钟信号才能正常工作

C. 在顺序执行的流水线中，若前一条指令修改了通用寄存器 $R_0$，后一条指令需要使用 $R_0$ 的值，则出现先写后读（RAW）的数据相关

D. 现代 CPU 中，L1Cache 设计为指令 Cache 和数据 Cache 分立的形式，是为了解决流水线中的结构冲突

**解析**：B 本来说的没问题，但它说时钟频率至少 50MHz，应该是至多 50MHz。

5. 小 Y 希望通过一不可靠信道向你发送某 4 位二进制串，采取基于配奇原则的 1 位纠错汉明码进行传输，假定传输过程中至多出现 1 位错，你收到 1100001（下标从高位开始），则小 Y 希望发送的数据为（ A ）

A. 1001　　B. 0001　　C. 0101　　D. 0000

**解析**：$P_0 = C_1 \oplus C_3 \oplus C_5 \oplus C_7 = 1 \oplus 0 \oplus 0 \oplus 1 = 0$，$P_1 = C_2 \oplus C_3 \oplus C_6 \oplus C_7 = 1 \oplus 0 \oplus 0 \oplus 1 = 0$，$P_2 = C_4 \oplus C_5 \oplus C_6 \oplus C_7 = 0 \oplus 0 \oplus 0 \oplus 1 = 1$，而这是配奇原则，因此 $P_0$ 和 $P_1$ 有问题，出错下标为 $\overline{P_2P_1P_0} = 011 = 3$，纠正反转 $C_3$ 得 1110001，因此原数据为 1001。

6. 现有一真值为-0.11010 的二进制小数以反码形式保存在某 8 位寄存器中，则该寄存器的内容为（ C ）

A. 0xE8　　B. 0x94　　C. 0x97　　D. 0x9F

**解析**：负数反码需要符号位为 1，绝对值取反，因此 $[-0.1101000]_{\text{反}} = 1.0010111$=0x97，注意真值要在低位补 0 补成 8 位。

7. 某种基数为 2 的 16 位浮点数类型格式为，阶码字段占 6 位（包括符号位），以补码表示，尾数字段占 10 位（包括符号位），以原码表示，且不强制要求规格化，则下列说法中**错误**的是（ C ）

A. 该浮点数类型可以精确表示整数 2048

B. 该浮点数类型能表示的最大的实数为 $2^{31} \times (1 - 2^{-9})$

C. 该浮点数类型能表示的最小的实数为 $2^{-41}$

D. 存在实数 $x$，$x$ 在该浮点数类型下的机器表示不唯一

**解析**：该浮点数无法精确表示 2047，却可以精确表示 2048，因为 2048 只需将尾数设为 0.1，阶码设为 12 即可，A 正确。C 肯定不对，最小的实数肯定是负的，这说的是最小的正实数。

8. 对于一个真值为-1100 的二进制整数，将其算术右移一位，则其在 5 位原码、5 位反码、5 位补码下的结果分别为（ A ）

A. 1,0110　1,1001　1,1010

B. 1,1110　1,0001　1,1010

C. 0,1110　0,1001　0,1010

D. 1,1110 0,1001 0,1010

**解析**：注意算术移位并不改变符号位，原码和反码仅移绝对值，负数反码和补码补 1。

9. 关于计算机内部计算加减法的硬件器件，下列说法**正确**的是（ D ）

I. 原码加减一般需要同时实现加法器和减法器，而补码加减只需要实现加法器
II. 串行进位加法器比并行进位加法器更慢，因为硬件电路更复杂，导致进位信号传播较慢
III. 运算速度：双重分组跳跃进位加法器 < 单重分组跳跃进位加法器
IV. 加法器中每一位结果和这一位的输入以及前面的进位有关

A. I,II,III,IV B. I,III,IV C. I,II,IV D. I,IV

**解析**：II 不对，并行进位加法器因为要同时产生所有位得进位，硬件电路更复杂。III 不对，肯定是双重分组更快。

10. 下列关于 RISC 的说法中，**错误**的是（ D ）

A. 一般使用流水线实现，可降低指令的平均 CPI
B. 指令功能比 CISC 弱，但使用频率普遍更高，实现也更简单
C. 实现相同的程序，RISC 机花费的指令数比 CISC 机可能更多，程序规模更大
D. 由于 RISC 更加精简，故指令类型、寻址方式和通用寄存器个数均少于 CISC，易于设计

**解析**：D 不对，RISC 机的通用寄存器更多，为了减少访存次数。

11. 某采取多重间接寻址的指令中地址码字段占 10 位，机器字长和存储字长均为 16 位，则该指令的寻址范围为（ C ）个存储单元。

A. $2^{10}$ B. $2^{16}$ C. $2^{15}$ D. $2^{9}$

**解析**：注意间接寻址分为单重间接寻址和多重间接寻址，而后者还需要 1 位指示是否为真正的操作数地址。详见唐书 7.3.2。

12. 某字节寻址且支持多重中断的 CPU 具有 5 个外部中断源 A,B,C,D,E，响应优先级从高到低为 A>B>C>D>E，处理优先级从高到低为 D>C>E>A>B。CPU 内部具有堆栈指针寄存器 SP，一次单字节压栈操作为，将一个字节存入 M(SP)，并令 SP++。CPU 执行中断隐指令时要将 4 字节信息压栈，当中断返回时会出栈恢复（假设中断服务程序本身不使用堆栈）。某个时刻 CPU 正在正常工作，SP=2000，允许外部中断，所有中断源皆未屏蔽，此时若 5 个中断源同时发起中断请求，则整个过程中 SP 的最大值为（ B ）

A. 2008 B. 2012 C. 2016 D. 2004

**解析**：本题的意思其实就是让你分析出多重中断最多能嵌套多少层，因为嵌套一层就会使 SP+=4，返回时会使 SP-=4。首先，CPU 先根据响应优先级响应 A，然后 A 的服务程序根据处理优先级将 A 和 B 屏蔽，开中断，然后 CPU 根据响应优先级响应 C（在 A 服务程序的基础上嵌套），将 C,E,A,B 屏蔽，开中断，CPU 继续响应 D（在 C 服务程序基础上嵌套），将所有中断源屏蔽，D 服务程序执行完后恢复屏蔽字，返回到 C 服务程序，C 服务程序执行完后恢复屏蔽字，返回 A 服务程序，此时 E 未屏蔽，响应 E（在 A 服务程序上嵌套），E 服务程序执行完后恢复屏蔽字，返回 A 服务程序，A 服务程序执行完后恢复屏蔽字，返回原程序，此时 B 未屏蔽，响应 B，然后返回。因此当 A-C-D 三重嵌套时 SP 达到最大值 2000+3×4=2012。

13. 某采取微程序控制方式的 CPU 共有 9 条指令，平均每条指令对应 7 条微指令，用于取指的公共微程序包含 3 条微指令，采取下地址字段法确定下条微指令地址。每条微指令采取字段直接编码方式，共有 34 个微命令，构成 5 个互斥类，分别包含 8,3,12,5,6 个微命令，则 CMDR 寄存器位数至少为（ C ）

A. 21　　B. 22　　C. 23　　D. 24

**解析**：首先，控存容量为 9×7+3=66 条微指令，因此下地址字段需 7 位，而这 5 个字段中都必须添加一个表示不发微命令的状态，因此要表示 9,4,13,6,7 个状态，分别花费 4,2,4,3,3 位，则微指令亦即 CMDR 位数至少为 7+4+2+4+3+3=23。

14. 下列关于存储器的说法中，**正确**的是（ D ）

I. U 盘可以快速读写，属于随机存储器

II. 1K×8 位 SRAM 芯片总共有 13 根地址和数据引脚

III. DRAM 集成度更高，成本更低，功耗也更低，因此适合集成到 CPU 内部作为缓存

IV. 对于 DRAM 来说，分散刷新由于不存在死时间，而集中刷新和异步刷新存在存储器无法接受读写操作的死时间，因此分散刷新相对更加合适

A. 仅 IV　　B. I,II,IV　　C. II,III　　D. 都不对

**解析**：I 不对，U 盘是 Flash，和 EEPROM 一样都属于只读存储器。II 不对，应该是 10+8=18 根。III 不对，DRAM 慢，应用 SRAM 作 Cache。IV 不对，分散刷新强行拉慢了所有存取周期，不如异步刷新更合适。一个都不对，真是离谱。

15. 有一存储器采取四体交叉编址方式，支持流水线连续地址访问。若 CPU 重复连续访问一段大小为 15 个存储单元的内存区，重复 400 次，总线传输周期 20ns，存取周期 80ns，仅考虑访存带来的时间开销，则最短时间为（ B ）

A. 144000ns　　B. 128040ns　　C. 128060ns　　D.129000ns

**解析**：注意，15 并不是 4 的倍数，应该在一轮访问结束后空一个间隔再进行下一轮访问，比如你是 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 这样访问，结束后下一个总线传输周期你不能立刻访问 0，必须隔至少 4 个周期才能访问。因此你每一轮实际上要空一个周期（充当一次冗余的地址访问），进行 16 次访问（当然，最后一轮只需要把最后的 15 个访问完即可）。因此总共访问的地址数是 16×400-1=6399，流水线下总时间为 80+(6399-1)×20=128040ns。

16. 某计算机采取 32 位主存地址，使用总容量为 32KB 的八路组相联 Cache，块大小 16B，采取写回策略，则该 Cache 一行的总位数至少为（ D ）

A. 1176bits　　B. 1184bits　　C. 1192bits　　D.1200bits

**解析**：块大小 16B=$2^4$，则块内地址 4 位，一共 $\frac{32\text{KB}}{16\text{B}}=2^{11}$ 块，每组 8 块，则共有 $\frac{2^{11}}{8}=2^8$ 组，组索引 8 位，因此标记字段 =32-4-8=20 位，而 Cache 内每块除了 20 位标记以及 16×8=128 位数据还有 1 位有效位和 1 位脏位，一块在 Cache 内占用 20+1+1+128=150 位，一行（组）共 150×8=1200 位。

17. 关于计算机的 I/O 系统，下列说法**正确**的是：（ D ）

I. I/O 接口通过总线与 CPU 相连接

II. I/O 端口是 CPU 和外设之间的界面，控制 CPU 和外设之间的信息交换

III. I/O 接口发向 CPU 的中断请求信号 INTR 是专门为程序中断 I/O 方式服务的

IV. 程序中断方式克服了程序查询方式的低效率缺点，可以实现 CPU 与外设的完全并行工作

A. I,II　　B. I,III　　C. I,IV　　D. 仅 I

**解析**：II 不对，那是 I/O 接口而不是 I/O 端口。III 不对，不能说专门为程序中断方式，因为 DMA 方式也要发起中断请求。IV 不对，不能说完全并行工作，因为数据的传输仍需 CPU 主动进行。

18. 关于 DMA，下列说法**错误**的是：（ C ）

A. 适合用于高速外设的大批量数据传输

B. DMA 请求一般有更高的优先级

C. 由于 DMA 可直接访问主存，故在数据传输中不需要干扰 CPU 的正常工作，只需在一批数据传输完成后向 CPU 发起中断进行通知

D. DMA 是与以运算器为中心的冯诺依曼机模型相违背的

**解析**：C 不对，DMA 在数据传输过程中需要和 CPU 竞争总线，可能会使得 CPU 访存延迟一个周期。D 是对的，所谓的“现代计算机已经改为以存储器为中心”的一个重要特征就是 I/O 设备可以通过 DMA 不经运算器（CPU）直接与存储器交互。

19. 下列英文缩写为总线标准名称的是（ B ）

I. ISP　　II. ISA　　III. RS-232C　　IV. VESA　　V. PCI　　VI. CPI

A. I,II,III,IV,V　　B. II,III,IV,V　　C. II,III,IV,V,VI　　D. II,IV,V

**解析**：emmm 并没有什么可解析的……

20. 关于虚拟存储系统，下列说法**错误**的是：（ D ）

A. 主要用于解决主存容量不足的问题

B. 作为一种缓存机制，不命中率远低于 Cache

C. 相比于不采取虚拟存储方式的计算机，CPU 访存更慢

D. 通过虚拟存储系统，辅存被映射到内存地址空间中，这使得 CPU 可通过访存指令直接与辅存交换数据

**解析**：C 是对的，因为多了地址翻译的过程。D 不对，无论如何 CPU 都不能直接访问辅存，虚拟存储系统中需要通过缺页中断将辅存中的页调入主存，然后 CPU 才能访问这段原来在辅存中的地址空间。

## 2 填空题

21. 6 位补码整数 1,10110 的相反数在 6 位补码表示下为 0,01010，在 6 位原码表示下为 0,01010。

22. 对于存储器而言，存取时间被定义为 启动一次存取操作（读或写）到完成该操作所需的全部时间，存取周期被定义为 存储器进行两次独立的读写操作之间所需的最小时间间隔。

23. 总线的异步通信可采取不互锁、半互锁、全互锁三种方式，其中最快的是 不互锁，最慢的是 全互锁，

最可靠的是 全互锁 。

24. 在你所学的 10 种寻址方式中，获取操作数最快的寻址方式是 直接寻址 ，占用指令位数最少的寻址方式是 隐含寻址（堆栈寻址也行） ，可用于数组访问的寻址方式是 变址寻址 ，可用于地址空间划分的寻址方式是 基址寻址 ，可用于条件转移的寻址方式是 相对寻址 。

25. 写出 3 种 Cache 的替换策略 先进先出 FIFO 、 最近最少使用 LRU 、 随机替换法 。

26. 现有一采取多级时序系统的 CPU，主频 20GHz，每条指令平均花费 5 个机器周期，每个机器周期平均花费 8 个时钟周期，则其平均指令执行速度为 500 MIPS。

27. 考虑一个具有 64 位 VA 以及 48 位 PA 的分页虚拟存储系统，页大小 8192B，则 VPO= 13 ，PPO= 13 ，VPN= 51 ，PPN= 0,01010 。

**解析**：VA：虚拟地址，PA：物理地址，VPO/PPO：页内地址，VPN：虚拟页号，PPN：物理页号，VPN=VA-VPO，PPN=PA-PPO。

28. 有一旧式硬盘，平均寻道时间 5ms，转速 6000rpm，具有 16 个盘片，每个盘片外直径 192mm，内直径 32mm，道密度 8 道/mm，每个磁道具有 1024 个扇区，每个扇区承载 512B 数据，则其总容量为 5 GB，平均寻址时间为 10 ms，最大数据传输率为 50MB/s 。

**解析**：磁道数 =8 道/mm$\times\frac{192\text{mm}-32\text{mm}}{2}$=640 道，总容量 =512B/扇区 ×1024 扇区/道 ×640 道 ×16 片 =$5\times2^{30}$B=5GB，磁盘转一圈需要 $\frac{60\text{s}}{6000\text{rpm}}$=10ms，平均寻址时间 = 平均寻道时间 + 平均旋转等待时间 =5ms+0.5×10ms=10ms，数据传输率 =$\frac{1024\times512\text{B}}{10\text{ms}}$=50MB/s。

29. 在采取水平型微指令的控制单元中，CM 的作用为 控制存储器，用来存放全部微指令 ，CMAR 的作用为 控存地址寄存器，用来存放欲读出的微指令地址 ，CMDR 的作用为
控存数据寄存器，用来存放从控存中读出的微指令 。

30. 假设 CPU 处理一次中断皆需 6$\mu$s，现一外设需要将 1000 字数据送入主存，存取周期为 20ns，外设准备数据的时间忽略不计。则在理想情况下，若使用基于周期窃取的 DMA 方式（每批数据 200 字），因传输而占用处理器的时间至多为 50$\mu$s ，若采取程序中断方式，则时间为 6ms 。

**解析**：DMA 方式下要传 5 批，每批传送结束后需要引发中断，每个字在总线上传输时至多占用 CPU 一个存取周期，因此时间至多为 5×6$\mu$s+20ns×1000=50$\mu$s，程序中断方式下每个字都要引发中断，总时间为 6$\mu$s×1000=6ms。本题改编自唐书例 5.3。

# 3 简答题

31. 下图分别为支持程序查询方式和支持程序中断方式的 I/O 接口电路，根据这两幅图，分别简述程序查询方式和程序中断方式的全过程。

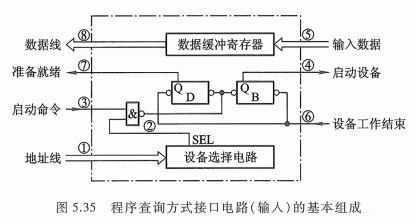


图 5.35 程序查询方式接口电路(输入)的基本组成

图 1: 支持程序查询方式的 I/O 接口电路

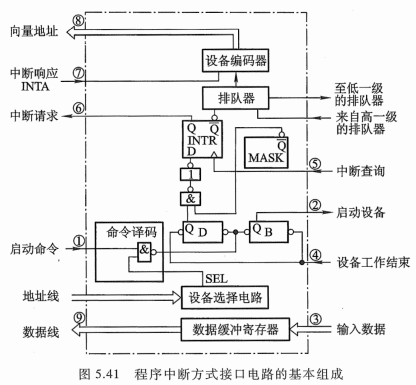


图 5.41 程序中断方式接口电路的基本组成

图 2: 支持程序中断方式的 I/O 接口电路

**解析**:

程序查询方式：CPU 先发送设备地址进行设备选择，然后向 I/O 接口发送启动命令，令 D=0，B=1，设备启动并开始准备数据。此时 CPU 一直踏步等待设备工作完成，不断查询 D 是否被置为 1。当设备工作完成后，将数据送入数据缓冲寄存器 DBR，发送结束信号，令 D=1，B=0。此时 CPU 检测到 D=1，则从 DBR 中取出数据。

程序中断方式：CPU 先发送设备地址进行设备选择，然后向 I/O 接口发送启动命令，令 D=0，B=1，设备启动并开始准备数据。此时 CPU 进行其它工作。当设备工作完成后，将数据送入数据缓冲寄存器 DBR，发送结束信号，令 D=1，B=0。此时若中断屏蔽触发器 MASK 为 0，说明该 I/O 接口中断未屏蔽，则发出中断请求信号 INTR，多个 INTR 需要通过硬件排队电路选择出响应优先级最高的 INTR，通过编码器生成对应的中断向量发送给 CPU，CPU 在当前指令执行结束后检测到 INTR 信号，则响应该中断，回复中断响应信号 INTA，并执行中断隐指令。

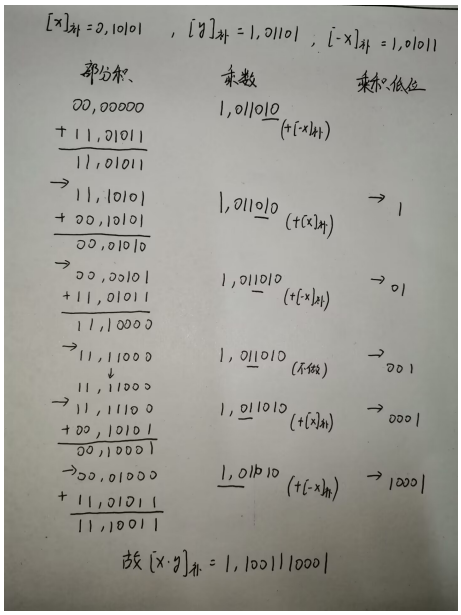
32. 简述提升存储器访问速度的若干思路（至少 3 条）。

**解析**: 1. 在高速工作的 CPU 和速度较慢的主存之间使用高速缓存 Cache，解决 CPU 和主存速度不匹配问题。2. 将存储器设计为多体系统（例如高位交叉或低位交叉），使得不同存储体可以并行工作，提升时间效率。3. 增大总线带宽（或存储字长），使得一次访存能取出更多的数据。4. 使用速度更快的高性能存储芯片，例如 DDR-SDRAM。

# 4 计算题

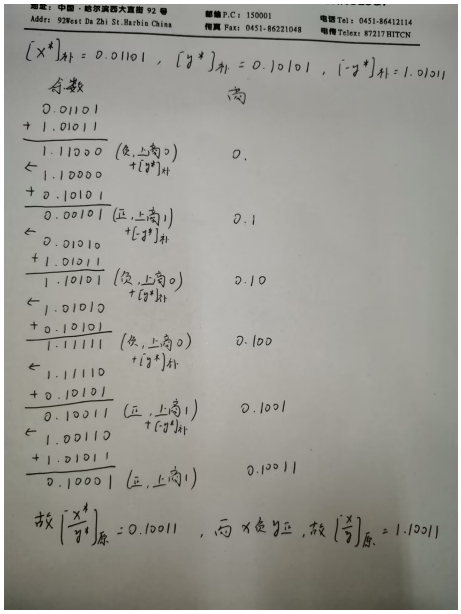
33. 已知 $[x]_{补}=0,10101$，$[y]_{补}=1,01101$，使用 1 位 Booth 算法求 $[x\times y]_{补}$，写出详细过程。

**解析**:




34. 已知 $x=-0.01101$，$y=0.10101$，使用原码加减交替法求 $[\frac{x}{y}]_{原}$（求出商值即可），写出详细过程。

**解析**:



35. 设浮点数阶码取 6 位，尾数取 6 位（均包含符号位），皆为补码表示，计算 $[2^{31}\times\frac{27}{32}]+[2^{30}\times\frac{13}{16}]$，写出详细过程。

**解析**：

A=$2^{00,11111}\times$00.11011，B=$2^{00,11110}\times$00.11010

对阶：B=$2^{00,11111}\times$00.01101

尾数相加：A+B=$2^{00,11111}\times$(00.11011+00.01101)=$2^{00,11111}\times$01.01000

右规：A+B=$2^{01,00000}\times$0.10100

溢出判断：由于阶码双符号位为 01，阶码正溢出，故浮点加法上溢！

# 5 分析与设计题

36. 某 CPU 采取了如图所示的片内总线数据通路设计：

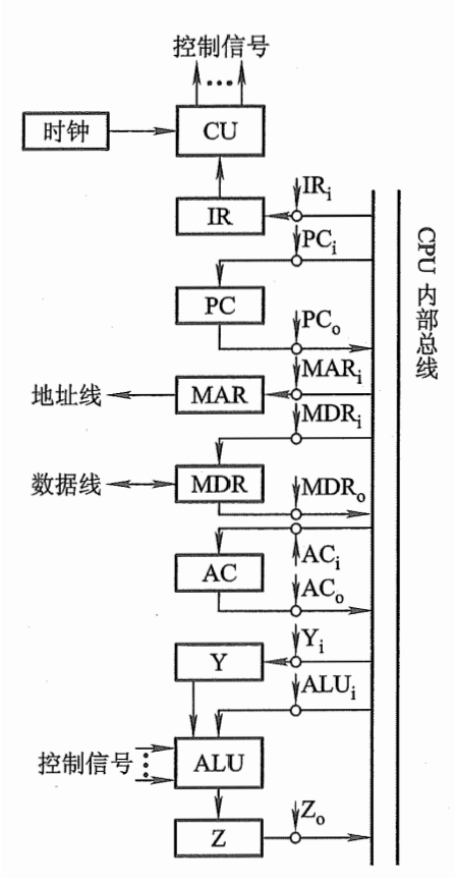



图 3: 该 CPU 的数据通路

其中，除了 CU 和 ALU 外，其余方框都表示寄存器。当 $XXX_i/XXX_o$ 信号有效时，对应的数据通路会处于开启状态，对应该寄存器 $XXX$ 的一次输入/输出。回答下列问题：

(1) 该 CPU 的指令系统中存在一条特殊加法指令 “CADD X”，其中 X 为地址码，作用为将地址为 (AC)+X 的存储单元中的数与 AC 相加，结果存放到 AC 中，即 AC←(AC)+M((AC)+X)。请写出这条指令包括取指在内的所有微操作与对应的操作信号（包括访存信号 R、W，尽管图上未画出）。

**解析**：

取指周期：

1. $PC_o, MAR_i$ 有效，PC→Bus→MAR
2. 1→R，发送读命令
3. M(MAR)→MDR
4. $MDR_o, IR_i$ 有效，MDR→Bus→IR，此时 CU 开始进行译码
5. (PC)+1→PC（这里一般不需要 ALU）

执行周期：

1. $AC_o, Y_i$ 有效，AC→Bus→Y
2. $MDR_o, ALU_i$ 有效，Ad(MDR)→Bus→ALU 输入端（注意：MDR 在此充当 IR 使用，取地址码，唐书 9.2.2 就这么搞的）
3. CU 向 ALU 发加法信号，Y+Ad(MDR)→Z
4. $Z_o, MAR_i$ 有效，Z→Bus→MAR
5. 1→R，发送读命令
6. M(MAR)→MDR
7. $AC_o, Y_i$ 有效，AC→Bus→Y（这一步其实可省略）
8. $MDR_o, ALU_i$ 有效，MDR→Bus→ALU 输入端
9. CU 向 ALU 发加法信号，Y+MDR→Z
10. $Z_o, AC_i$ 有效，Z→Bus→AC

(2) 小 Y 认为图中的寄存器 Z 没有什么实际意义，反正也不是程序员能访问的通用寄存器，有没有都无所谓。你同意他的观点吗？给出理由。

**解析**：Z 不可以省略，因为 ALU 是组合逻辑器件，且总线上同一时刻只能有一个数据信号，此时总线上是 ALU 的输入端信号，若无 Z，ALU 会立刻将结果送上总线，造成数据冲突。

(3) 简述总线型结构数据通路和非总线型结构数据通路的特点和差异。假如你是一名 CPU 设计人员，你希望采取哪种数据通路结构？给出理由。

**解析**：总线型结构数据通路结构简单，易于设计，耗费更少的内部连线，但容易产生数据冲突，为避免冲突需要为指令周期增加更多的节拍，使得数据有秩序地被传递，这样减慢了指令的执行速度。非总线型数据通路则不存在冲突的问题，多个数据传递操作可以并行进行，指令执行速度快，但设计复杂，内部连线更多。考虑到摩尔定律能够为芯片提供足够多的晶体管器件，现代处理器设计更倾向于以更复杂规模更大的电路设计换取更高的性能，因此非总线型数据通路更合适。

37. 某 CPU 指令字长 16 位，具有 32 个通用寄存器，指令地址码 6 位，现在需要为其设计指令系统。要求：具有 10 条双地址码指令，11 条单地址码指令，10 条单地址码 + 单寄存器指令，6 条单寄存器指令，3 条双寄存器指令。

(1) 采取操作码扩展技术，给出你设计的指令操作码（要求：指令的操作码位于指令字的高位）。

(2) 该指令系统最多还可以设计几条零地址码指令?

**解析**:

从短到长分配操作码：双地址码指令操作码 4 位，单地址码 + 单寄存器指令操作码 5 位，双寄存器指令操作码 6 位，单地址码指令操作码 10 位，单寄存器指令操作码 11 位。

4 位操作码指令 10 条，可用操作码 $2^4 = 16$ 个，分配 0000~1001。还剩 6 个（1010~1111），剩下的指令必须以它们为高 4 位。

5 位操作码指令 10 条，可用操作码 $6\times2^{5-4}=12$ 个，分配 10100~11101，还剩 2 个（11110,11111），剩下的指令必须以它们为高 5 位。

6 位操作码指令 3 条，可用操作码 $2\times2^{6-5}=4$ 个，分配 111100~111110，还剩 1 个（111111），剩下的指令必须以它为高 6 位。

10 位操作码指令 11 条，可用操作码 $1\times2^{10-6}=16$ 个，分配 1111110000~1111111010，还剩 5 个，剩下的指令必须以它们为高 10 位。

11 位操作码指令 6 条，可用操作码 $5\times2^{11-10}=10$ 个，分配 11111110110~1111111011，还剩 4 个，剩下的指令必须以它们为高 11 位。

因此，零地址码指令还可以设计 $4\times2^{16-11}=128$ 条。

38. 1978 年，Intel 公司推出了著名的 8086 CPU，该 CPU 采用了约 3 万个 3$\mu$m 技术的晶体管，可工作在最高 10MHz 的时钟频率下。尽管这些 40 多年前的技术相比于今日而言早已略显陈旧，但 8086 在计算机发展史上是具有里程碑意义的——它标志着 x86 和 IBM PC 兼容机这两种影响深远且沿用至今的架构的诞生。甚至，现代的大部分 x86 架构 CPU 为了兼容，在最初上电时需要以 8086 的方式工作（称之为实模式），而 8086 定义的的所有指令都仍可运行于现代的所有 x86 架构 CPU 上。根据题目提供的 8086 的有关数据与工作机制以及你在本课程中的所学知识，回答下列问题：

(1) 8086 具有 40 根引脚，下图是其外部引脚布局：

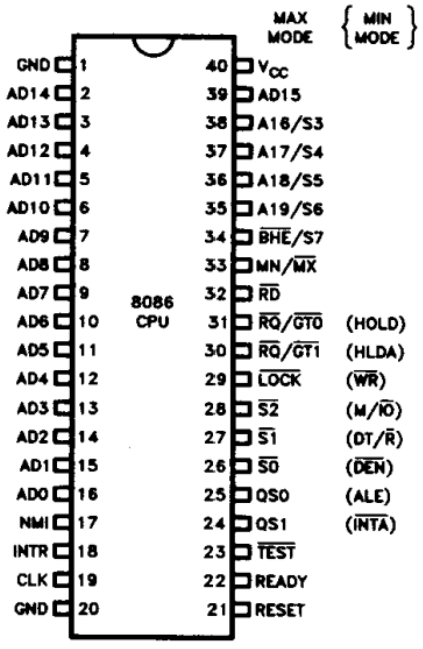



**Figure 2. 8086 Pin Configuration**

图 4: 8086 的外部引脚布局

8086 的机器字长和数据总线宽度均为 16 位，采取字节编址方式，使用 20 位地址总线，则其按字寻址的范围大小为__512K__。8086 采取引脚复用技术，引脚 $AD_0 \cdots AD_{15}$ 在总线周期的不同节拍下，既可用作 16 位数据总线（$D_0 \cdots D_{15}$），也可用地址总线的低 16 位（$A_0 \cdots A_{15}$），引脚 $A_{16} \cdots A_{19}$ 为地址总线的高 4 位。在本课程中你还学过哪种使用了类似的引脚复用技术的器件？这种技术有何优缺点？

**解析**：每个字 16 位即 2 字节，按字寻址范围 $\frac{2^{20}}{2}$=512K。DRAM 芯片也采取了引脚复用技术（行地址和列地址在不同时间通过相同的地址引脚送入）。引脚复用技术能够减少芯片外部的引脚数（尤其是对于像 8086 这样的古老的双列直插芯片而言……），而这会带来时间上的牺牲——8086 不得不使用至少两个节拍将地址和数据送上总线（当然 DRAM 也不得不使用更多的时间接收行列地址），如果地址和数据引脚本身就是独立的，那么地址和数据的发送可以同时在一个节拍内完成。

(2) 8086 的一些与总线读写操作相关的控制信号引脚如下表所示：

| 引脚（信号）名称 | 说明 |
|---|---|
| $\overline{\text{RD}}$ | 读使能信号，低电平有效 |
| $\overline{\text{WR}}$ | 写使能信号，低电平有效 |
| M/$\overline{\text{IO}}$ | 访存使能信号，高电平表示读写主存，低电平表示读写 I/O 端口 |
| $\overline{\text{BHE}}$ | 指示高 8 位数据总线 $D_8 \cdots D_{15}$ 是否有效，与奇偶分体有关 |

不难发现，8086 的访存和读写 I/O 端口都要使用同样的地址总线和数据总线，通过访存使能信号 M/$\overline{\text{IO}}$ 进行操作区分。由此可以推断出 8086 对于 I/O 端口采取什么样的编址方式？应使用什么样的指令访问 I/O 端口？这样设计有何优缺点？

**解析**：既然具有专门的控制信号区分访存和 I/O 操作，因此 I/O 端口一定采取独立编址方式，和存储器地址是独立的，具有专门的 I/O 指令（不使用访存指令读写 I/O 端口）。这样设计的优点是不占用额外的内存地址空间（要知道，8086 的地址空间本身就小得可怜了……），相较于统一编址方式的缺点是需要更多的指令（统一编址只需要有访存指令）以及控制信号线。

(3) 8086 可以工作在多种模式下，在本题中我们考虑最简单的最小工作模式（Minimum Mode）。在该模式下，8086 的一个用于访问存储器或 I/O 端口的总线周期一般由四个节拍 $T_1$,$T_2$,$T_3$,$T_4$ 构成，这几个节拍的操作如下表所示：

| 节拍 | 操作 |
|---|---|
| $T_1$ | 将地址送上引脚 $AD_0 \cdots AD_{15}$ 和 $A_{16} \cdots A_{19}$ |
| $T_2$ | 撤销地址信号，若为写操作则将数据送上引脚 $AD_0 \cdots AD_{15}$ |
| $T_3$ | 发出控制信号，数据开始写入或读到总线上 |
| $T_4$ | 撤销信号，结束总线周期 |

假设 8086 以 10MHz 时钟频率工作，总线周期的节拍与时钟周期一致，求理想情况下 8086 的最大总线带宽。

**解析**：一个总线周期 =4 个时钟周期，传送 16 位数据，因此最大总线带宽为 $\frac{10\text{MHz}}{4} \times 2\text{B}=5\text{MBps}$。

(4) 事实上，8086 的一个总线周期可以在某些情况下具有多个节拍。8086 的 READY 引脚具有向总线周期中插入等待节拍 $T_w$ 的功能，在节拍 $T_3$ 结束时的时钟边沿，8086 会检测 READY 引脚，若为低电平，则插入一个等待节拍 $T_w$，此后每个时钟周期结束时都会检查 READY，不断插入 $T_w$，直至 READY 为高时才会结束等待，进入节拍 $T_4$ 开始结束总线周期。时序图如下所示：

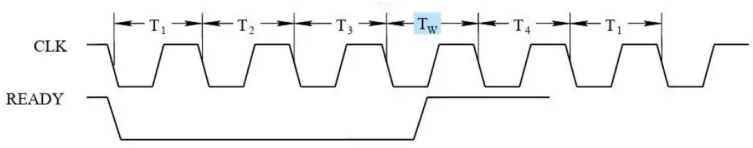



图 5: 插入等待节拍 $T_w$ 的 8086 总线周期时序图

由此可知，8086 的总线通信采取__半同步通信__方式，你认为 8086 这样设计的原因可能是什么？

**解析**：根据所学知识可知，半同步通信结合了同步通信和异步通信的特点，既由统一时钟信号控制，又增加了等待的能力，可以解决同步通信中双方速度不一致的问题。8086 在通过总线对 I/O 端口等进行读写操作时，外设可以通过 READY 引脚（对应唐书上的 WAIT 信号）告知 8086 自己是否已经完成了相应操作，若未完成则不能让 8086 结束总线周期，需要令其等待，则将 READY 信号置为低电平。这样，8086 即可通过总线控制低速外设了。当然，这个和我们所学的 3 种 I/O 方式的不同之处在于，它能够支持 I/O 端口速度很低甚至存储器速度很低的情况（那个年代的存储器速度真的也都是很慢的……）。

(5) 考虑 8086 的两个连向 I/O 外设的引脚 HOLD 和 HLDA，其中 HOLD 为输入信号（正常状态下应输入低电平），HLDA 为输出信号。当 CPU 在一个总线周期内进行总线操作时，若检测到 HOLD 引脚输入了高电平，则 CPU 会将 HLDA 引脚输出高电平，并在下一个总线周期时不进行任何操作，且令数据地址总线引脚 $AD_0 \cdots AD_{15}$ 和 $A_{16} \cdots A_{19}$ 处于高阻态（可简单认为 CPU 与总线断开连接）。根据所学知识，你认为这两个引脚的作用与应用场景是什么？

**解析**：这个主要用于 DMA 的总线占用，对应唐书上“DMA 接口向 CPU 发出占用总线的请求，CPU 发回响应信号表示允许将总线控制权交给 DMA”，这会使得 8086 暂时断开与总线的连接，使得 DMA 接口能够通过公共的总线进行直接内存访问。

(6) 8086 CPU 的指令系统定义了最原始的 x86 指令集，其中相等则转移指令“je X”可采取相对转移的方式，该指令字长 16 位，指令格式如下：

| 0-7 位 | 8-15 位 |
|---|---|
| 0x74 | X |

其中，X 为位移量，采取 8 位补码表示，则位移量的范围为__-128~+127__。8086 在执行相对转移型指令时，会在**取指后**的 PC 的基础上加上偏移量。假设这条“je X”指令位于地址 0x3000 处，希望在条件成立的情况下跳转到 0x2FF0，则指令中的字节 X 应为__0xEE__（十六进制表示）。你认为转移指令采取这种相对寻址的好处是什么？另外，现代 x86-64 CPU 可以完全兼容 40 年前为 8086 设计的指令系统，操作码和指令格式甚至可以完全保持不变，例如这条双字节的 je 指令仍然可以被 Intel Core i9 CPU 正常执行，而这两种相差 40 年的 CPU 显然几乎不可能有相同的芯片构造与内部工艺（当然，以及二者悬殊的性能差距）。根据所学知识，解释这一现象的原理。

**解析**：

取指后 PC=0x3002，则位移量 X=0x2FF0-0x3002=-0x12=0xEE。

转移使用相对寻址的优点是与程序的具体位置无关，如果使用绝对转移（转移的目的地址固定），则这意味着程序必须被加载到固定的地址上（当然，csapp 告诉我们，操作系统可以使用虚拟地址空间解决这个问题），而相对转移只与程序的局部结构有关，更加灵活。

8086 CPU 与 Core i9 CPU 显然是两种硬件实现上几乎完全不同的 CPU，但后者能够完全兼容前者的指令系统，根据唐书第一章内容，硬件实现是计算机组成的问题，指令系统是计算机体系结构的问题，相同的体系结构可以有不同的组成，一种机器的体系结构能够维持很多年，而机器的组成会随着技术发展产生较大变化。

(7) 8086 的 INTR 引脚接受外设发来的中断请求信号，8086 会在“适当的时候”检查 INTR 引脚，若其为高电平，且标志寄存器的 IF 标志位为 1（允许中断），则 8086 响应中断，将当前的 PC 和标志寄存器入栈，将 IF 置 0，读取总线上的中断向量号，查找存储器中的中断向量表（从地址 0x00000 开始的 1KB 内存空间），跳转到相应的中断服务程序入口地址。你认为“适当的时候”应该是什么时候？若你希望使得 8086 进行多重中断，则中断服务程序应该进行什么操作？

**解析**：“适当的时候”显然就是一条指令执行结束的时刻。若要实现多重中断，中断服务程序需要在开始时通过处理优先级设置中断屏蔽字，并将 IF 置 1（开中断）。

(8) 根据第 (1) 问我们知道 8086 采取了地址/数据引脚复用，因此 8086 的外围电路需要负责进行引脚的解复用，分离出真正的地址总线 $A_0 \cdots A_{19}$ 和数据总线 $D_0 \cdots D_{15}$。顺带一提，这一解复用过程可在外部设置地址锁存器，通过 8086 的地址锁存引脚 ALE 控制，暂存 CPU 在节拍 $T_1$ 发出的地址来实现（不必担心，解复用跟本题并没有什么实际关系）。现在考虑 $A_0 \cdots A_{19}$ 和 $D_0 \cdots D_{15}$ 这 34 根解复用后的地址和数据总线，以及 8086 用于读写操作的控制信号 $\overline{RD}$、$\overline{WR}$、M/$\overline{IO}$、$\overline{BHE}$。根据第 (2) 问我们已经知道 $\overline{BHE}$ 与 8086 的奇偶分体存储有关。8086 采取奇偶分体的原因是其具有 16 位数据总线，而早期的存储器芯片或外设一般都只有 8 根数据总线，因此，8086 规定，奇地址内存单元（大小为一个字节）和偶地址内存单元应该对应不同的存储体，且偶存储体使用低 8 位数据总线 $D_0 \cdots D_7$，奇存储体使用高 8 位数据总线 $D_8 \cdots D_{15}$，$\overline{BHE}$ 信号用来控制高 8 位数据总线是否有效（即奇存储体是否工作）。8086 可以通过 $\overline{BHE}$ 以及地址总线最低位 $A_0$ 的不同组合方式来决定字或字节的读写，具体如下表所示：

| $A_0$ | $\overline{BHE}$ | 读写方式 | 偶存储体工作状态 | 奇存储体工作状态 |
|---|---|---|---|---|
| 0 | 0 | 从偶地址读写一个字 | 工作 | 工作 |
| 0 | 1 | 从偶地址读写一个字节 | 工作 | 不工作 |
| 1 | 0 | 从奇地址读一个字节 | 不工作 | 工作 |
| 1 | 1 | 无效操作 | 不工作 | 不工作 |

8086 的一种地址空间分配方案如下图所示：

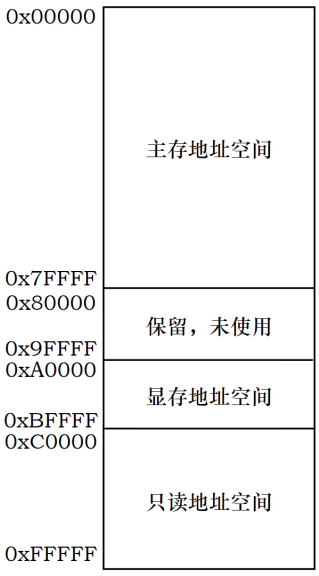



图 6: 8086 的一种地址空间分配方案

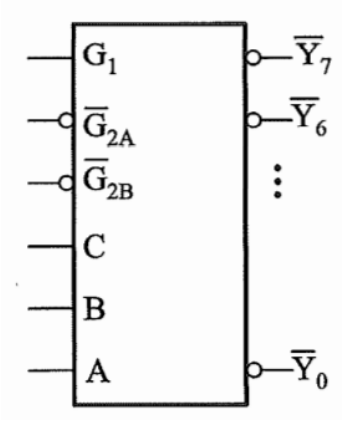
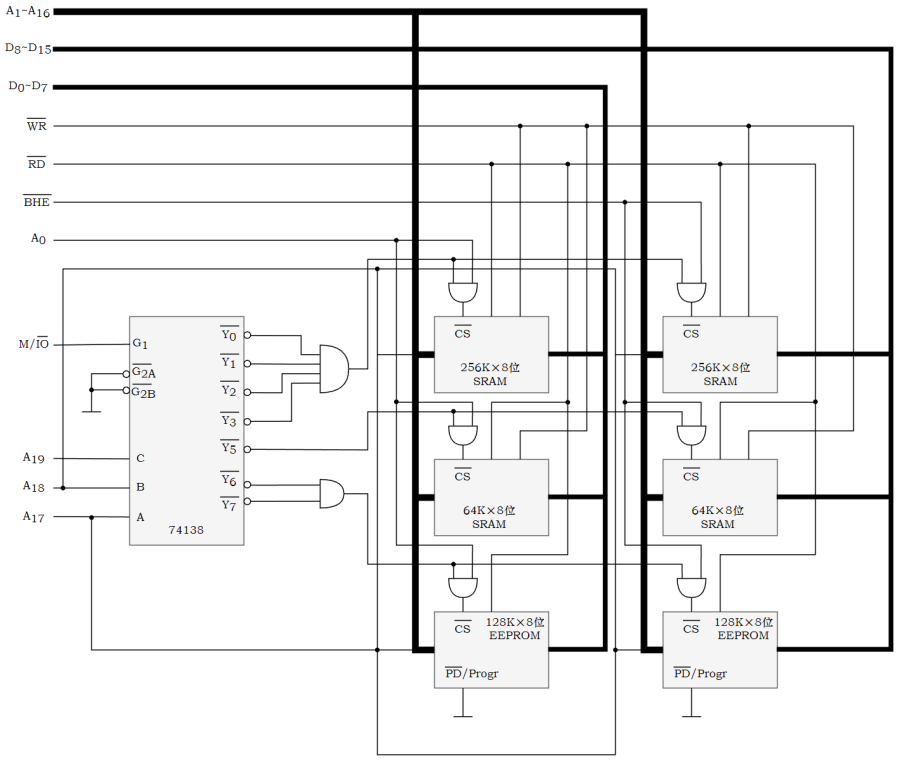



图 7: 74138 译码器芯片

其中，主存地址空间和显存地址空间都是可读写的，且显存地址空间与屏幕显示有关（事实上，在特定的显示模式下，向这个地址空间中写入像素或字符会立刻呈现在屏幕上，但本题为了简化，仅将其视为一段普通的可读写地址空间），而只读地址空间对应主板上的 BIOS ROM 芯片，存放 CPU 的初始化指令（8086 在上电复位后会自动从只读地址空间的 0xFFFF0 处开始执行指令）以及其它重要的系统设置数据。现在，给定如图所示的 74138 译码器芯片，以及若干容量为 64K×8 位、128K×8 位和 256K×8 位的 SRAM 芯片以及 EEPROM 芯片，以及门电路若干，你需要为 8086 实现**外围存储器电路**。要求：显存地址空间和主存地址空间**不得**分配在相同的存储体内，且你只需要考虑连向 8086 的地址总线 $A_0 \cdots A_{19}$ 和数据总线 $D_0 \cdots D_{15}$ 以及控制信号 $\overline{RD}$、$\overline{WR}$、M/$\overline{IO}$、$\overline{BHE}$。若给定的地址落在保留空间中，任何存储芯片都**不得**工作。给定的 SRAM 芯片具有控制引脚 $\overline{RD}$、$\overline{WR}$、$\overline{CS}$，给定的 EEPROM 芯片具有控制引脚 $\overline{RD}$、$\overline{PD}$/Progr、$\overline{CS}$。

**解析**：（套路都是讲过好多遍的，这里懒得赘述了……唐书例 4.3 就是 8086 奇偶分体访存的例子）



(9)（开放型问题，可不回答）根据上述 (1)-(8) 对 8086 这一 40 年前生产的芯片的分析，你是否对《计算机组成原理》这门课程有了新的认识和感受？如果给你足够的 74 系列逻辑芯片和存储器芯片，根据《数字逻辑与数字系统设计》课程以及本课程所学知识，查阅相关资料（例如 8086 的芯片手册），你能否为 8086 设计一个完整的外围电路，构造一个具有控制外设能力的微型计算机系统？试阐述你的设计思路。

你可能会对我在这套“模拟题”里整这么一道绕来绕去且大杂烩的巨型题的行为感到疑惑，的确，就考试而言，不可能给你出这样一道题，顶多，可以让你使用你的“所学知识”，对各种经过变形的抽象模型进行所谓的分析。这个题其实是想通过一个十分贴切（因为唐书很多地方很大程度上就是在参照着 8086 的机制去讲，毕竟 8086 在那个年代的影响实在是太深远了，但出于所谓的“不结合具体机型”的考虑，该书试图对此尽力隐藏）的具体案例将同学们在本课程中所学的诸多知识串联起来，让大家明白，原来书上讲的看着不怎么真的各种奇奇怪怪的概念与机制，在现实中真的存在，而且全都被囊括在一个巴掌大的芯片里了——它们全部都是有联系的，并且它在**物理上**竟然是这样实现的。一直以来，我个人始终认为《计算机组成原理》应该和《数字逻辑与数字系统设计》合并成同一门课，无论考研如何安排。它们应当共同揭示软件系统如何以物理形式运行在现实世界之上。构造一个基于 8086 的软硬件微机系统的尝试就是一个很好的例子，csapp 课程已经教会我们如何使用 x86 汇编语言编程，数字逻辑课程已经教会我们如何构造逻辑电路（而且甚至也已经进行过芯片连线实验），计组课程已经教会我们旧式计算机系统结构的各种原理与机制，我们其实应该已经具备了这样的能力，不是吗？亲眼目睹写出的汇编语言代码驱动出真正的电平信号，或许比一切任何形式的 PCIRMARMDRCU 之流更加有说服力——起码能让你充分坚信你正在学**真实的**玩意。

针对这个第 (9) 问可以给感兴趣的同学一些指引，可以去查阅 8284A、8255A、8259A 这三种古老芯片的信息，它们是 Intel 专门为 8086 的外围电路设计的，分别对应时钟、I/O 和中断的功能（有趣的是，据说从 2021 级开始设置的计组实验就是关于它们的）。关于整个硬件系统的更详细的设计思路，可以参考一些外校的《微机原理与接口技术》课程实验，这些课程很多本身就是专门研究 8086 及其衍生硬件的。