



厦门大学《计算机组成原理》考卷解析

信息学院

软件工程系

软件工程专业

试卷类型：样卷 2

by maoli

一、 选择题（请从 A、B、C、D 中选择唯一的一个正确答案，15 小题，每小题 2 分，共 30 分；在答题纸填写答案时请写上每小题的对应编号）

1. 下列关于冯·诺依曼计算机基本思想的叙述中，错误的是：

- A. 程序的功能都通过 CPU 执行指令实现
- B. 指令和数据都用二进制表示，形式上无差别
- C. 指令按地址访问，数据都在指令中直接给出
- D. 程序执行前，指令和数据需预先存放在存储器中

冯诺依曼计算机的特点是：1. 计算机由运算器、存储器、控制器、输入设备和输出设备五大部件组成；2. 指令和数据以同等地位存放于存储器内。并可按地址访问；3. 指令和数据均可用二进制表示；4. 指令由操作码和地址码组成。操作码表示操作的性质、地址码表示操作数在存储器中的位置；5. 指令在存储器中按顺序存放

C. 在冯·诺依曼结构计算机中，数据可以存储在存储器中，也可以作为立即数直接包含在指令中。因此，选项 C 中的“数据都在指令中直接给出”是不正确的。正确的描述应该是，指令和数据在存储器中分开存储，指令按地址访问，数据可以通过地址访问或作为立即数直接包含在指令中获取。

2. 某程序 P 在计算机 M 上的执行时间是 10 秒。经过编译优化后，程序 P 的指令数减少了 40%，而 CPI 增加了 20%。请问编译优化后，程序 P 在计算机 M 上的执行时间是：

- A. 6 秒
- B. 7.2 秒
- C. 8 秒
- D. 12 秒

B. 新的执行时间 = 原来的执行时间 × 指令数比例 × CPI比例
= $10 \times 0.6 \times 1.2$
= 7.2

3. 假设有如下的 C 语言语句：

```
unsigned char uc=254;
char c=uc;
unsigned short int i=c;
printf(" i = %d \n",i);
```

请问，执行上述语句后，i 的值是：

- A. 2
- B. 254
- C. 65024
- D. 65534

D. 65534。

在执行这些语句后，变量i的值将是65534。让我们逐步分析代码：

1. **unsigned char uc = 254;** - 将无符号字符类型变量uc初始化为254，这是一个合法的赋值操作。
2. **char c = uc;** - 将无符号字符类型变量uc赋值给有符号字符类型变量c。由于有符号字符的范围是-128到127，而uc的值为254，超出了有符号字符类型的表示范围。根据C语言的整数转换规则，这种情况下会发生符号扩展。符号扩展将保留最高有效位的值，因此c的值将是-2。
3. **unsigned short int i = c;** - 将有符号字符类型变量c赋值给无符号短整型变量i。由于c的值为-2，这个赋值操作将进行符号扩展(补1、若是原来无符号则补0)，将其扩展为等效的无符号值。在扩展后，i的值将是65534。
4. **printf("i = %d\n", i);** - 使用printf函数打印变量i的值。%d格式说明符用于打印有符号十进制整数，因此输出为65534。

触类旁通: unsigned char uc=254;

```
char c=uc;
short int i=c;
printf(" i = %d \n",i);
```

此时答案是-2

4. 下列关于 Booth 算法的叙述中, 错误的是:
- A. 符号位不参与运算, 需要单独计算符号位
 - B. 乘数的末位增设附加位 y_{n+1} , 初值为 0
 - C. 根据 y_{n+1} 和 y_n 的差值判断各步的具体运算
 - D. 需要进行 $n+1$ 次累加操作和 n 次右移操作

D. 需要进行 $n+1$ 次累加操作和 n 次右移操作。

Booth算法是一种用于二进制乘法的算法，它可以通过加法和右移操作来实现乘法运算。下面逐个选项进行分析：

- A. Booth算法中，符号位不直接参与乘法运算，需要单独计算符号位。
- B. Booth算法中，乘数的末位会增设一个附加位 y_{n+1} ，初始值为0，用于判断是否需要加上被乘数。
- C. Booth算法根据附加位 y_{n+1} 和当前附加位 y_n 的差值来确定下一步的具体运算，可以进行加法、减法或无操作。
- D. 错误的。Booth算法中，乘法运算需要进行 n 次累加操作和 n 次右移操作，而不是 $n+1$ 次累加操作。

3.7用补码一位乘法计算xy。

(2) $x = -0.011010$, $y = -0.011101$ 。

[illegible]

5. 下列命中组合情况中，一次访存过程中不可能发生的是：A

- A. TLB 命中、页缺失、cache 缺失
- B. TLB 命中、页命中、cache 缺失
- C. TLB 缺失、页命中、cache 命中
- D. TLB 缺失、页缺失、cache 缺失

6. 某指令格式如图 1 所示，其中 OP 为操作码、M 为寻址方式、I 为变址寄存器编号、D 为形式地址。若采用先间址后变址的寻址方式，则操作数的有效地址是：B

- A. (I)+D
- B. ((I)+D)
- C. (I)+(D)
- D. ((I)+(D))

OP	M	I	D
----	---	---	---

图1 某指令格式

7. 下列关于操作数寻址方式的叙述中，错误的是：

- A. 立即数寻址方式，操作数在指令中
- B. 直接寻址方式，操作数的地址在指令中
- C. 寄存器间接寻址方式，操作数在寄存器中
- D. 间接寻址方式，操作数的地址在存储器中

C. 应为寄存器寻址

寄存器间接寻址是通过把偏移地址放在寄存器中的寻址方式。用于存放数据在内存中的偏移地址的

8. 下列关于 CISC 处理器的叙述中，错误的是：

- A. CISC 处理器普遍采用微程序控制器
- B. CISC 处理器指令字长不固定
- C. CISC 处理器寻址方式少
- D. CISC 处理器对访存指令不加限制

C. CISC 处理器寻址方式少：这是错误的叙述。CISC 处理器通常提供多种寻址方式，包括直接寻址、间接寻址、相对寻址等多种寻址方式，以满足不同的编程需求。

9. 下列关于 RISC 处理器的叙述中，错误的是：

- A. RISC 处理器普遍采用硬布线控制器
- B. RISC 处理器指令字长固定
- C. RISC 处理器内有大量的通用寄存器
- D. RISC 处理器对访存指令不加限制

D. RISC 只有 Load 指令（从主存读到寄存器）和 Store 指令（从寄存器往主存写）可以访存，因此需要较多的寄存器。

10. 下列关于主存（主存储器）和控存（控制存储器）的叙述中，错误的是：

- A. 主存在 CPU 外，控存在 CPU 内
- B. 主存按地址访问，控存按内容访问
- C. 主存用于存储指令和数据，控存用于存储微指令
- D. 主存采用 RAM 和 ROM 实现，控存采用 ROM 实现

D. 主存采用 RAM 和 ROM 实现，控存采用 ROM 实现：这是错误的叙述。主存通常使用 RAM（随机访问存储器）来实现，以便能够读写数据。而控存通常使用 ROM（只读存储器）来实现，因为控存中存储的微指令在运行时是只读的，不需要修改。

11. 一次总线事务中，主设备只需要给出一个首地址，从设备就能从首地址开始的若干连续单元读出或写

入多个数据。这种总线事务方式称为：D

- A. 并行传输
- B. 串行传输
- C. 同步传输
- D. 突发（猝发）传输

12. 以下选项中，哪个选项不能提高同步总线的数据传输速率：D

- A. 增加总线宽度
- B. 提高总线工作频率
- C. 支持突发（猝发）传输
- D. 采用地址/数据复用方式

13. 下列有关总线定时的叙述中，错误的是：

- A. 异步通信方式中，全互锁方式最慢
- B. 异步通信方式中，非互锁方式的可靠性最差
- C. 同步通信方式中，同步时钟信号可由多个设备提供
- D. 半同步通信方式中，握手信号的采样由同步时钟控制

C.同步通信方式中，同步时钟信号由系统同一提供。

14. 下列关于外部 I/O 中断的叙述中，正确的是：C

- A. 中断控制器按所接收中断请求的先后次序进行中断优先级排队
- B. CPU 响应中断时，通过执行中断隐指令完成对通用寄存器的保护
- C. CPU 只有在处于中断允许状态时，才能响应外部设备的中断请求
- D. 有中断请求时，CPU 立即暂停执行当前指令，转去执行中断服务程序

分析：

A:不是按照先后次序进行优先级排队,是有固定的响应优先级的

B:对通用寄存器的保护是在保护现场阶段完成的

D:只有在CPU开中断使才会响应中断请求。

15. 下列关于多重中断系统的叙述中，错误的是：B

- A. 在一条指令执行结束时响应中断
- B. 中断处理期间 CPU 处于关中断状态
- C. 中断请求的产生与当前指令的执行无关
- D. CPU 通过采样中断请求信号检测中断请求

二、简答题（5 小题，共 30 分）

1. （6 分）假设某计算机的指令长度固定为 16 位，具有双操作数、单操作数和无操作数等 3 类指令，每个操作数的地址规定用 6 位表示。

(1) 若操作码字段不固定，现已设计出 m 条双操作数指令、n 条无操作数指令。请问在此情况下，这台计算机最多可以设计出多少条单操作数指令？（3 分）



参考:

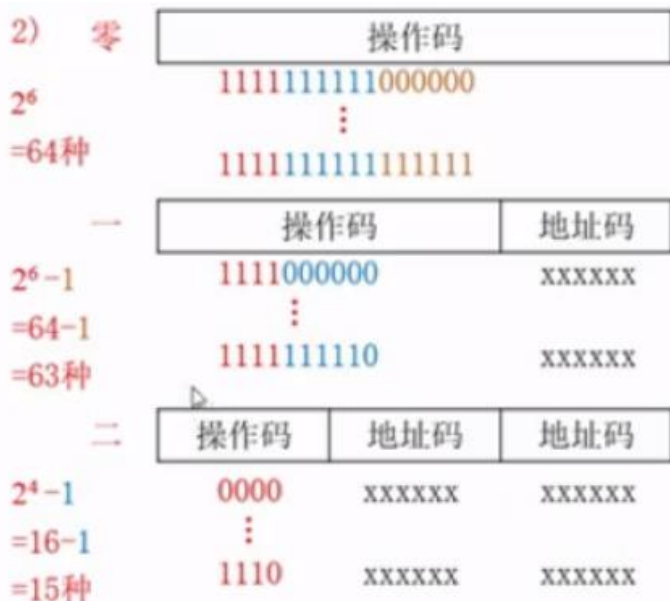
3) 采用扩展操作码技术, 若二地址指令有P条, 零地址指令有Q条, 则一地址指令最多有几种?



二地址指令操作码4位, 可有 2^4 种编码, 二地址指令有P条, 则留出 2^4-P 种编码用于一地址指令的扩展。一地址指令操作码字段可有 $(2^4-P) \times 2^6$ 种编码, 设留出n种用于零地址指令的扩展, 则 $n \times 2^6 = Q$, 那么 $n = Q / 2^6$, 故一地址指令最多有 $(2^4-P) \times 2^6 - Q / 2^6$ 种。

(2) 若操作码字段不固定, 当双操作数指令取最大数时, 且在此基础上, 单操作数指令条数也取最大数。请问这3类指令各拥有多少条指令? (3分)

$2^4-1, 2^6-1, 2^6$



扩展操作码技术: 二地址指令的操作码字段拿出若干种编码来表明此时应按一地址指令解释, 一地址指令的操作码字段拿出若干种编码来表明此时应按零地址指令解释, 以此类推。

附: 固定操作码

假设指令字长为16位，操作数的地址码为6位，指令有零地址、一地址、二地址3种格式。

- 1) 设操作码固定，若零地址指令有M种，一地址指令有N种，则二地址指令最多有几种？
- 2) 采用扩展操作码技术，二地址指令最多有几种？
- 3) 采用扩展操作码技术，若二地址指令有P条，零地址指令有Q条，则一地址指令最多有几种？



最多可表示 $2^4=16$ 种不同的操作码，对应16种不同的指令，其中M种用于零地址指令，N种用于一地址指令，故二地址指令最多有 $16-M-N$ 种。

实例参考王道164页第8题

https://blog.csdn.net/KK_2018

2. (6分) 图2为数据校验流程图，请根据图2回答下述问题：

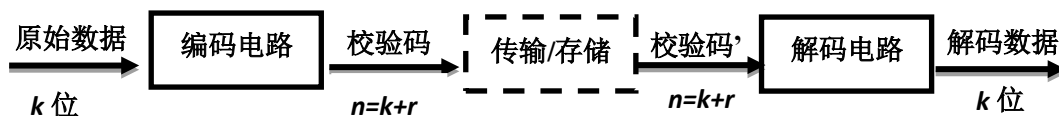


图2 数据校验流程图

- (1) 根据图2所示，k位原始数据，经编码电路后，生成 $n=k+r$ 位的校验码。请问增加的r位信息的作用是什么？（2分）答：校验
- (2) 当传输或存储环节出现问题时，校验码'和校验码会出现不同。如果采用的是奇偶校验，请问如何根据生成的检测位G判断是否出错？实际电路是采用哪种门电路完成？（2分）

答：在奇偶校验中，生成的检测位（通常称为奇偶校验位）用于检测数据中是否存在错误。下面是根据生成的奇偶校验位G判断是否出错的步骤：

1. 统计数据中二进制位为1的个数（包括奇偶校验位G）。
2. 如果二进制位中1的个数是偶数，那么数据没有错误，因为偶数个1可以通过奇偶校验恢复为正确的偶数个1。
3. 如果二进制位中1的个数是奇数，那么数据存在错误，因为奇数个1无法通过奇偶校验恢复为正确的偶数个1。

在实际电路中，可以使用异或门来完成奇偶校验。

- (3) 如果编码电路采用n位海明码，为了能够实现1位检错，则 n, k, r 应满足什么关系？若使得其可以同时检测两位错，可采用什么方式实现？（2分）

答：在海明码中，n表示总位数（包括数据位和校验位， $= k+r$ ），k表示数据位数，r表示校验位数。为了实现1位检错的能力，n，k，和r需要满足以下关系：

找到最小的r使得 $n \leq 2^r - 1$

其中， $^$ 表示幂运算。这个关系保证了码字空间中的每个码字之间的汉明距离至少为3，从而使得海明码能够检测并纠正1位错误。这样，校验位的数量就能够满足1位错误的检错需求。

要实现同时检测两位错误，可以采用扩展的海明码。多了一位奇偶校验位，所以码距多了1。这个奇偶校验位将海明码的每一位（包括信息位和校验位）作为信息位进行校验。

3. (6分) 设某DRAM的存储体为64行×64列结构，存储器读写周期为 $0.25\mu s$ ，刷新间隔为2ms。请

回答下述问题：

- (1) 该存储器地址线采用复用技术，则有几根地址线管脚？为了支持地址复用技术，需要增加哪两个信号？（2分）

该存储器地址线采用复用技术时，地址线的数量应等于 DRAM 的行数和列数的位数之和。给定的存储体为 64 行 x 64 列，即 $R = 6$ 位， $C = 6$ 位。因此，地址线的数量为 $R + C = 6 + 6 = 12$ 根。

为了支持地址复用技术，需要增加以下两个信号：

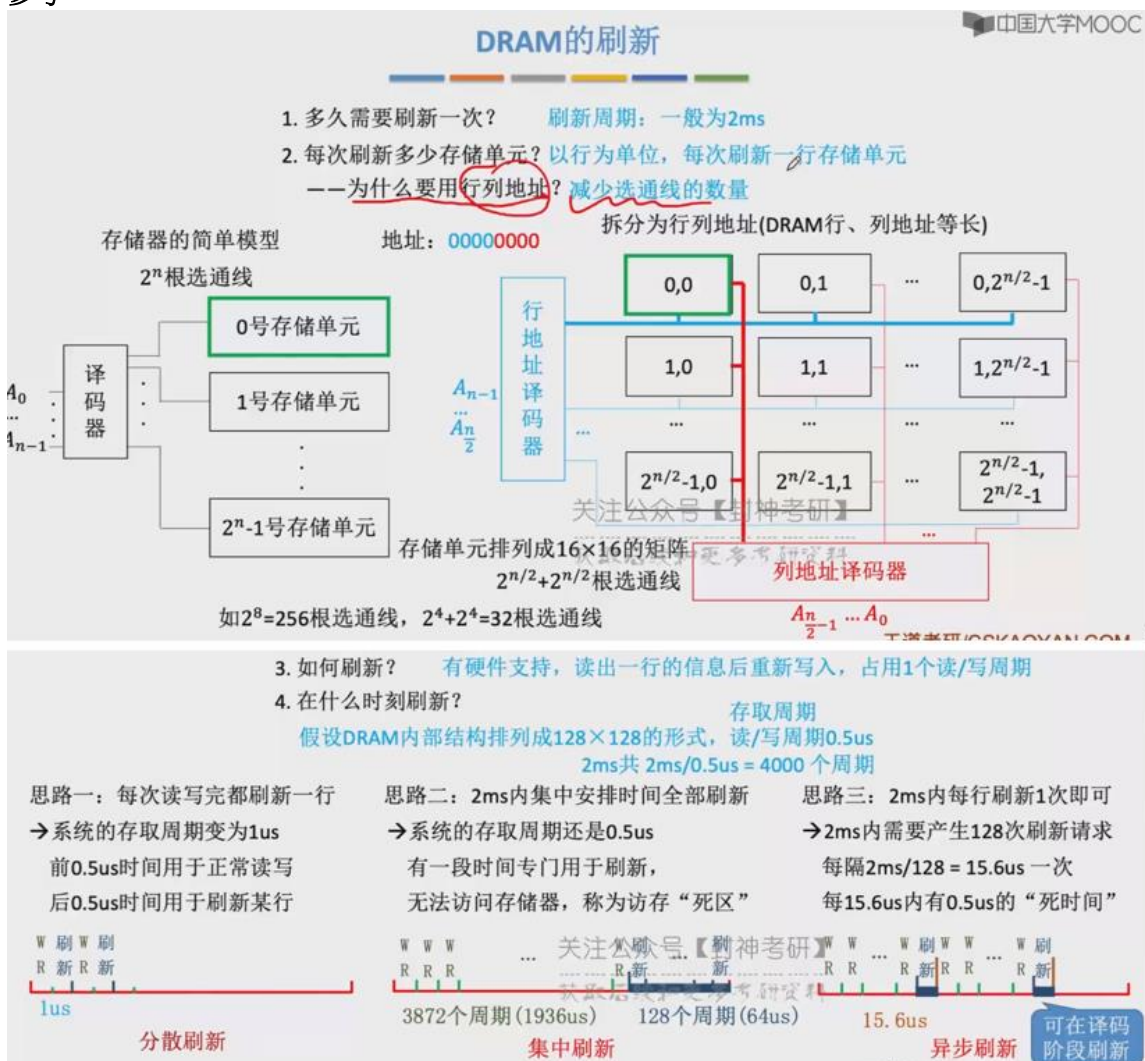
- RAS（行地址选通信号）：用于选择要访问的行。
- CAS（列地址选通信号）：用于选择要访问的列。

- (2) 如果采用集中式刷新方式，其死区时间是多少？（1分）

0.25us*64

- (3) 为了提升效率，现采用异步刷新方式。请画出异步刷新方式的时间分配情况，并进行解释。（3分）

参考



4. （6分）设主存容量为 1MB；cache 容量为 16KB，块长为 4 个字，每字 32 位。请问：

- (1) 直接相联映射方式时，主存地址为 ABCDEH 的存储单元在 cache 中的什么位置？（3分）

主存和Cache按字节编址,

Cache容量16KB= 2^{14} B, 地址共格式为14位, 分为16KB/(4*32/8B)= 2^{10} 块, 每块4*32/8=16B= 2^4 B, Cache地址格式为:

Cache字块地址 (10位)	字块内地址 (4位)
-----------------	------------

主存容量1MB= 2^{20} B, 地址共格式为20位, 分为1MB/(4*32/8B)= 2^{16} 块, 每块 2^4 B, 采用直接映射方式, 主存字块标记为20-14=6位, 主存地址格式为:

主存字块标记 (6位)	Cache字块地址 (10位)	字块内地址 (4位)
-------------	-----------------	------------

主存地址为ABCDEH=1010 1011 1100 1101 1110B, 主存字块标记为101010, Cache字块地址为11 1100 1101, 字块内地址为1110, 故该主存单元应映射到Cache的101010块的第1110字节, 即第42块第14字节位置。或者在Cache的第11 1100 1101 1110=3CDEH字节位置。

(2) 4路组相联映射方式时, 主存地址为 ABCDEH 的存储单元在 cache 中的什么位置? (3 分)
主存字块标记 (8位)+cache组号 (8位)+字块内地址 (4位)
1010 1011 1100 1101 1110

5. (6 分) 设某计算机有 4 个中断源 A、B、C、D, 其硬件排队优先级为 A->B->C->D 降序排列, 其屏蔽字如表 1 所示:

表 1 各中断源对应的屏蔽字

中断源	屏蔽字			
	A	B	C	D
A	1	1	0	1
B	0	1	0	0
C	1	1	1	1
D	0	1	0	1

- (1) 请根据表 1 的屏蔽字, 给出四个中断源的中断处理顺序。(2 分)
C>A>D>B
- (2) 若 4 个中断源同时发出中断请求, 请画出 CPU 执行程序的轨迹。(4 分)

D	-----
C	-----
B	-----
A	-----
CPU	-----

三、综合题（4 小题，共 40 分）

1. (8分) 假设 Logisim 中已经有 8 位加法器的电路, 其电路封装如图 3 所示; 其中, X 和 Y 为 8 位输入, S 为 8 位输出, $C0$ 为低位进位信号, $C8$ 为最高位进位, $C7$ 为次高位进位。现要求利用该 8 位加法器以及一些辅助的电路, 实现一个 8 位可控加减法电路, 该 8 位可控加减法电路可以完成 8 位有符号整数的加减法运算和 8 位无符号整数的加减法运算。该电路的输入为 X (8 位) 和 Y (8 位) 以及 Sub , 输出为 S (8 位) 以及 OF 和 UOF 。 $Sub=0$ 时做加法运算, $Sub=1$ 时做减法运算。 OF 为有符号整数加减运算溢出标志, UOF 为无符号整数加减运算溢出标志。请画出该 8 位可控加减法电路。提示: 可以利用 Logisim 中的分线器实现 1 位扩展为 8 位的功能, 如图 4 所示。Logisim 中的门电路、多路选择器如图 5 所示。

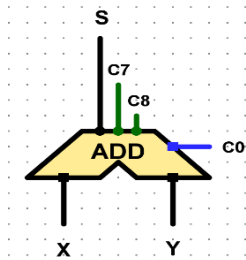


图 3 8 位加法器封装图

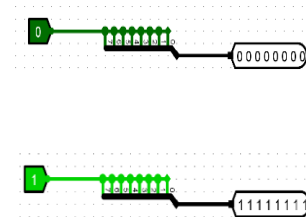


图 4 利用分线器实现 1 位扩展为 8 位

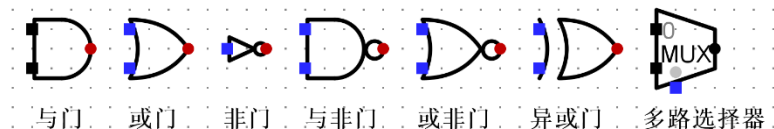
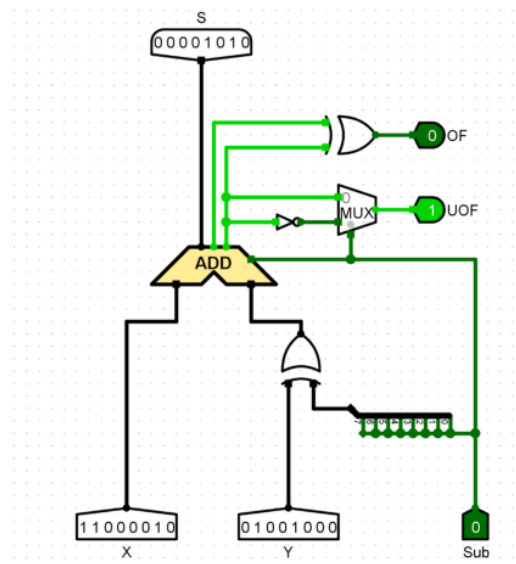


图 5 Logisim 中的门电路和多路选择器

答：只画出可控加减法电路（没有 OF 和 UOF）给 6 分

画出 OF 输出电路给 2 分

画出 UOF 电路给 2 分



2. （8分）主存地址空间大小为 512MB，按字节编址。指令 cache 和数据 cache 分离，均有 16 行，每个 cache 行的数据块大小均为 128B。数据 cache 采用全相联映射方式。现有两个功能相同的程序 A 和 B，其代码如图 6 所示。假定 int 型数据为 32 位补码，程序编译时 i、j、sum 均分配在寄存器中。数组 a 按行优先方式存放，首地址为 1000（十进制）。请回答如下问题，要求说明理由或给出计算过程：
- (1) 若不考虑用于 cache 一致性维护和替换算法的控制位，则数据 cache 的总容量是多少？（4分）
- (2) 程序 A 和 B 的数据访问命中率各是多少？哪个程序的执行时间更短？（4分）

<pre> 程序A: int a[512][512]; int sum_array1() { int i,j,sum=0; for (i = 0; i < 512; i++) for (j = 0; j < 512; j++) sum += a[i][j]; return sum; } </pre>	<pre> 程序B: int a[512][512]; int sum_array2() { int i,j,sum=0; for (j = 0; j < 512; j++) for (i = 0; i < 512; i++) sum += a[i][j]; return sum; } </pre>
--	--

图 6 程序 A 和程序 B 的代码

参考本题，仅变换了数字大小：

某计算机的主存地址空间大小为256MB，按字节编址。指令Cache和数据Cache分离，均有8个Cache行，每个Cache行大小为64B，数据Cache采用直接映射方式。现有两个功能相同的程序A和B，其伪代码如下：

<pre> 程序 A: int a[256][256] int sum_array1 () { int i,j,sum=0; for(i=0;i<256;i++) for(j=0;j<256;j++) sum+=a[i][j]; return sum; } </pre>	<pre> 程序 B: int a[256][256] int sum_array2 () { int i,j,sum=0; for(j=0;j<256;j++) for(i=0;i<256;i++) sum+=a[i][j]; return sum; } </pre>
---	---

假定int类型数据用32位补码表示，程序编译时i、j、sum均分配在寄存器中，数组a按行优先方式存放，其首地址为320（十进制数）。请回答下列问题，要求说明理由或给出计算过程。

- 1) 若不考虑用于Cache一致性维护和替换算法的控制位，则数据Cache的总容量为多少？
- 2) 数组元素a[0][31]和a[1][1]各自所在的主存块对应的Cache行号分别是多少（Cache行号从0开始）？
- 3) 程序A和B的数据访问命中率各是多少？哪个程序的执行时间更短？

【解析】

（1）主存容量256MB，按字节寻址的地址位数应为28位，数据Cache分为8行（用3位地址），每行64B（用6位地址），因此Cache中每个字块的Tag字段的位数应是28-9=19位，还要使用一个有效位，二者合计为20位；因此数据Cache的总容量应为：64B×8+(20/8×8)B = 532B。

（2）数组a在主存的存放位置及其与Cache之间的映射关系如下图所示。

数组A[0][31]所在的主存块对应的Cache行号是：

$(320+31 \times 4) \div 64 = 6$ ，

数组A[1][1]所在主存块对应的Cache行号：

$((320+256 \times 4 + 1 \times 4) \div 64) \bmod 8 = 5$ 。

所以 a[0][31]所在主存块映射到Cache第6行，

a[1][1]所在主存块映射到Cache第5行。

（3）编译时i、j、sum均分配在寄存器中，故数据访问命中率仅考虑数组a的情况。

①这个程序的特点是数组中的每一个int类型的数据只被使用一次。数组A按行优先存放，数据Cache正好放下数组半行中的全部数据，即数据的存储顺序与使用次序有更高的吻合度，每个字块存16个int类型的数据，访问每个字块中头一个字不会命中，但接下来的15个字都会命中，访问全部字块都符合这一规律，命中率是15/16，即程序A的数据访问命中率为93.75%；

②而程序B是按照数组的列执行外层循环，在内层循环过程中，将连续访问不同行的同一列的数据，不同行的同一列数据使用的是同一个Cache单元，每次都不会命中，命中率是0，程序执行特别慢。

根据上述计算出的命中率，得出程序B每次取数都要访问主存，所以程序A的执行比程序B快得多。

3. (8分) 某 CPU 时钟频率为 100MHZ，硬盘以 256B 大小的扇区为单位传输数据，CPU 访问硬盘的速率为 10MB/s。请问：

(1) 采用程序中断方式，启动阶段发送命令和参数需要 80 个时钟周期，中断服务开销需要 300 个时钟周期，实际传输需要 1000 个时钟周期，程序中断方式 CPU 时间占用率是多少？(3 分)

(2) 采用 DMA 方式，预处理阶段需要 100 个时钟周期，后处理阶段需要 300 个时钟周期，DMA 方式 CPU 时间占用率是多少？(3 分)

(3) 通过 CPU 时间占有率的比较，可以得出什么结论？为什么？(2 分)

要达到10mb/s，每秒传输的次数为：10mb/256b=40960次

(1) 采用程序中断方式，CPU 时间占用率计算如下：

总时间 = 启动阶段时间 + 中断服务时间 + 传输时间 = 80 + 300 + 1000

CPU 时间占用率 = $1380 \times 40960 \div 100\,000\,000 = 56.5248\%$

(2) 采用 DMA 方式，CPU 时间占用率计算如下：

DMA处理的3个阶段，只有第1阶段（预处理阶段）和第3阶段（后处理阶段）需要CPU参与，

总时间 = 预处理阶段时间 + 后处理阶段时间

总时间 = 100个时钟周期 + 300个时钟周期

CPU 时间占用率 = $400 \times 40960 \div 100\,000\,000 = 16.3\%$

(3) DMA方式的CPU时间占用率较低，相比于程序中断方式，DMA（直接内存访问）方式的CPU时间占用率较低的原因是因为DMA允许数据在外设和内存之间直接传输，而无需CPU的干预。

在DMA方式下，CPU在传输数据之前只需进行预处理和后处理操作，而实际的数据传输过程由DMA控制器完成。DMA控制器直接和外设和内存进行数据传输，无需CPU的介入。

相比之下，程序中断方式需要CPU在启动阶段发送命令和参数，处理中断服务以及完成后处理，这些操作都需要消耗CPU的时间。

因此，DMA方式减少了CPU的负担，使其可以同时进行其他任务，降低了CPU的时间占用率。

需要注意的是，虽然DMA方式的CPU时间占用率较低，但DMA控制器本身会消耗一定的系统资源。因此，在选择使用DMA方式还是程序中断方式时，需要综合考虑系统的整体性能和资源利用效率。

4. (16分) 单周期 MIPS 处理器和多周期 MIPS 处理器的数据通路分别如图 8 与图 9 所示。请回答下列问题：

(1) 多周期 MIPS 处理器数据通路与单周期 MIPS 处理器数据通路有哪些不同？(2 分)

(2) 请修改单周期 MIPS 处理器的数据通路，使其能支持 MIPS 指令系统的 j 指令。j 指令属于 J 型指令，其指令格式如图 7 所示；汇编代码为：j imm26；RTL 描述为：PC <- {(PC+4)_{31:28},imm26<<2}。

要求：①请画出需要增加或修改的数据通路及控制信号；②请给出单周期 MIPS 处理器数据通路 j 指令的执行流程及每一步的控制信号。(7 分)

(3) 请修改多周期 MIPS 处理器的数据通路，使其同样能支持 MIPS 指令系统的 j 指令。要求：①请画出需要增加或修改的数据通路及控制信号；②请填写表 2 中多周期 MIPS 处理器数据通路 j 指令取指周期和执行周期的操作流程及控制信号。(7 分)

OP (6位)	imm26 (26位)
---------	-------------

图 7 MIPS 的 J 型指令格式

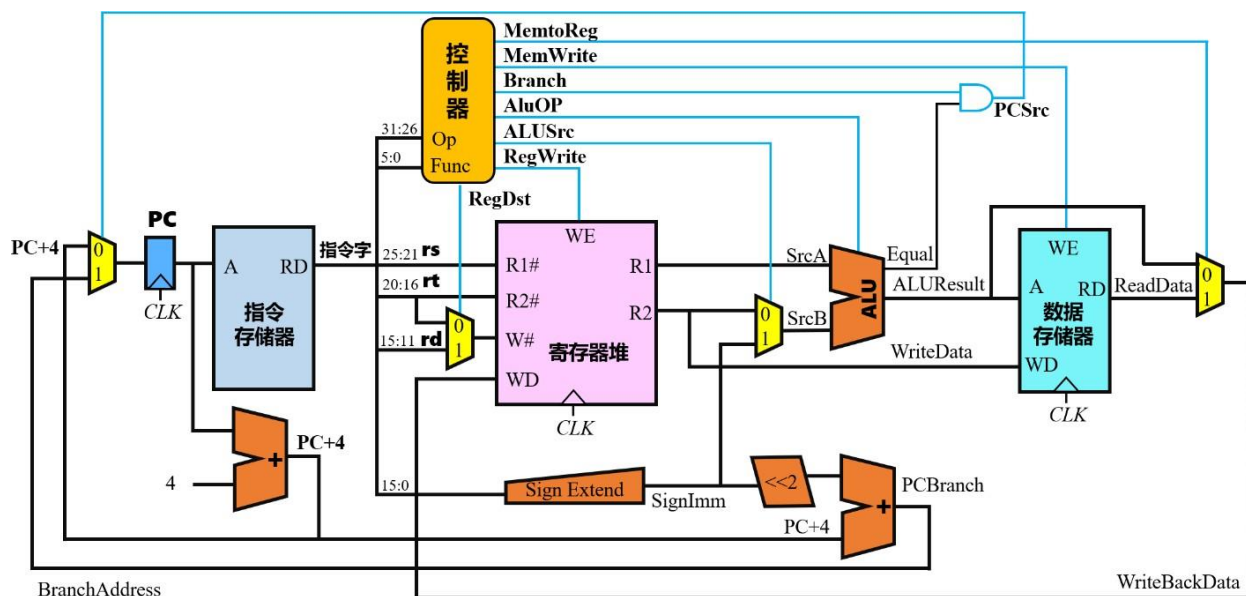


图 8 单周期 MIPS 处理器的数据通路

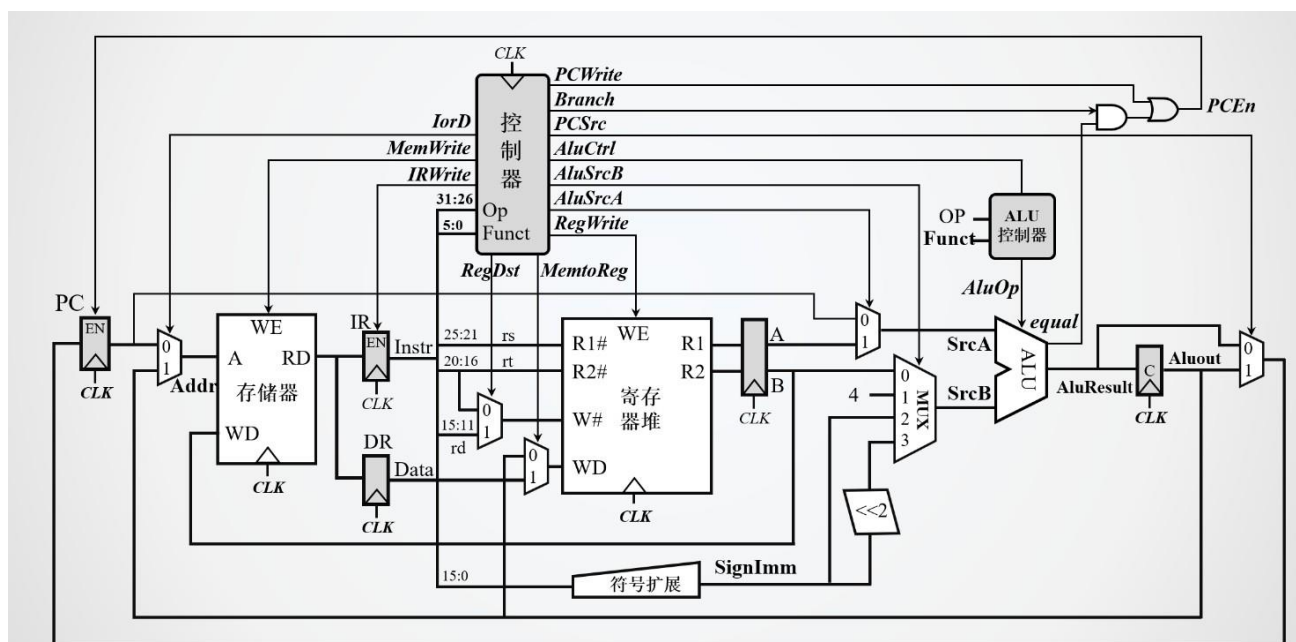


图 9 多周期 MIPS 处理器的数据通路

表 2 多周期 MIPS 处理器数据通路 j 指令的操作流程及控制信号

机器周期	时钟周期	操作流程（操作）	控制信号（非零值）
取指周期	T1		
	T2		
执行周期	T3		

本题不会做，如果有做出结果的同学想向您请教QAQ

以下是gpt的结果，仅供参考：

(1) 多周期 MIPS 处理器数据通路与单周期 MIPS 处理器数据通路之间存在以下不同点：

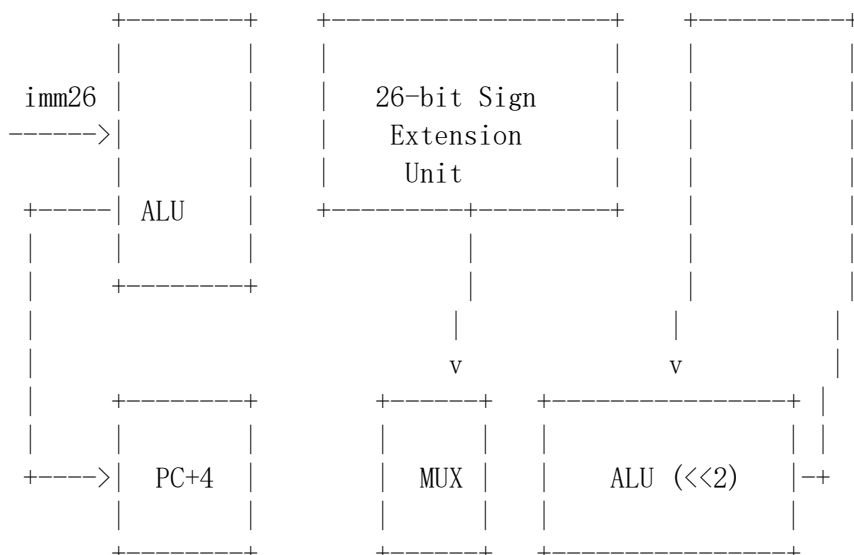
1. 分离的时钟周期：多周期 MIPS 处理器将指令执行过程分为多个时钟周期，每个时钟周期执行不同的阶段（如取指、译码、执行、访存和写回），而单周期 MIPS 处理器在一个时钟周期内完成一条指令的全部执行。
2. 控制信号的引入：多周期 MIPS 处理器引入了更多的控制信号，用于控制不同阶段的操作。这些控制信号包括时钟使能信号、分支信号、跳转信号、内存读写信号等，用于控制数据通路中各个模块的操作。
3. 时钟周期的长度不一：由于多周期 MIPS 处理器将指令执行过程分为多个阶段，每个阶段可能需要不同的时钟周期数来完成。因此，多周期 MIPS 处理器的时钟周期长度可以不同，相比之下，单周期 MIPS 处理器的时钟周期长度是固定的。
4. 数据存储器的访问方式：在单周期 MIPS 处理器中，指令和数据存储器共享同一个存储器模块，指令和数据的访问是互斥的，即不能同时进行。而在多周期 MIPS 处理器中，通常会使用独立的指令存储器和数据存储器，允许同时访问指令和数据存储器。
5. 部分模块的重复利用：多周期 MIPS 处理器的某些模块（如寄存器堆和 ALU）可能会在不同的时钟周期中被多次使用，以支持多个阶段的执行。而单周期 MIPS 处理器的每个模块在一个时钟周期内只会被使用一次。

总的来说，多周期 MIPS 处理器通过引入多个时钟周期和更多的控制信号，实现了对指令执行过程的细分和灵活控制，从而提高了处理器的效率和灵活性。

(2)

为了支持 MIPS 指令系统的 j 指令，需要对单周期 MIPS 处理器的数据通路进行修改。以下是需要增加或修改的数据通路及控制信号：

1. 增加的数据通路：
 - 增加一个 26 位的立即数扩展单元，用于将 imm26 扩展为 32 位的目标地址。
 - 增加一个 MUX，用于选择 PC 的来源，即选择是由 PC+4 还是由目标地址计算得到的新 PC 值。
2. 修改的控制信号：
 - 增加一个 j 指令的控制信号，用于指示当前指令是否为 j 指令。
 - 修改 MUX 的控制信号，使其根据 j 指令的控制信号选择 PC 的来源。



控制信号：

- j 指令控制信号：用于指示当前指令是否为 j 指令。
- MUX 控制信号：用于选择 PC 的来源，如果是 j 指令，则选择 ALU 输出，否则选择 PC+4。

单周期 MIPS 处理器数据通路 j 指令的执行流程及每一步的控制信号如下：

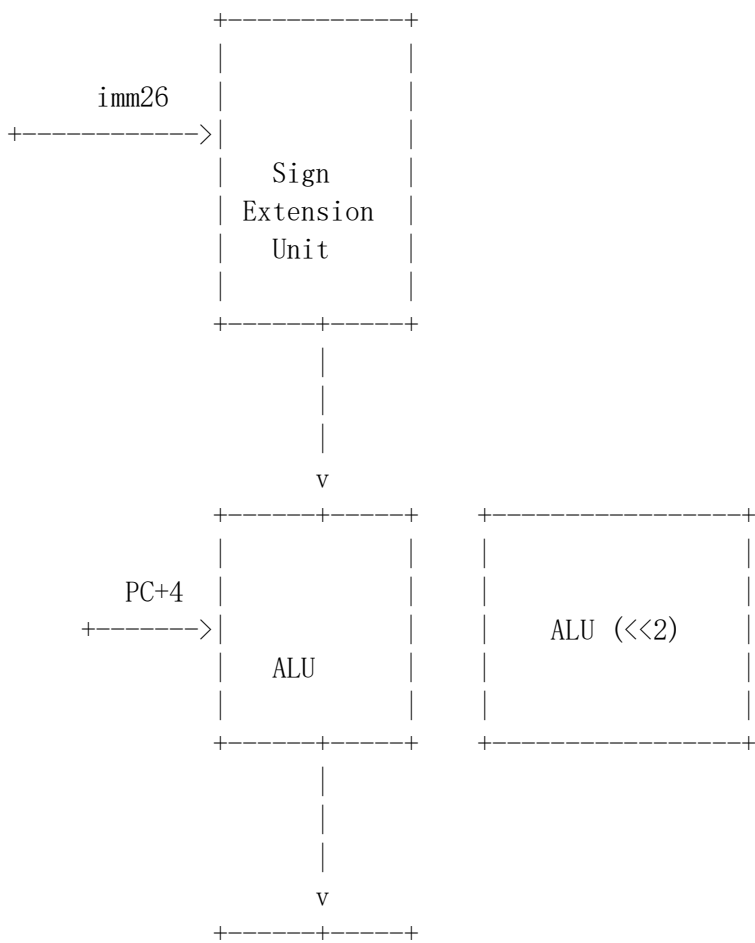
1. 取指阶段：
 - 从指令存储器（Instruction Memory）读取指令。

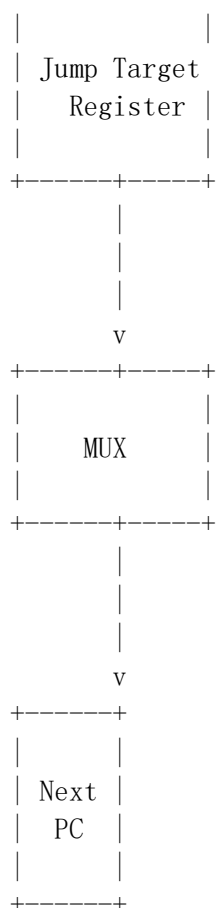
- 控制信号：j 指令控制信号为高，MUX 控制信号选择 PC+4。
- 译码阶段：
 - 无需特殊操作，继续传递控制信号。
 - 执行阶段：
 - ALU 将 PC+4 作为第一个操作数。
 - imm26 经过符号扩展后与 00 进行拼接，作为第二个操作数。
 - 控制信号：无特殊操作。
 - 访存阶段：
 - 无需特殊操作，继续传递控制信号。
 - 写回阶段：
 - 无需特殊操作，继续传递控制信号。
 - PC 更新：
 - 选择 MUX 输出作为新的 PC 值。

(3)

- 增加的数据通路：
 - 增加一个 26 位的立即数扩展单元，用于将 imm26 扩展为 32 位的目标地址。
 - 增加一个目标地址寄存器 (Jump Target Register)，用于存储 j 指令的目标地址。
 - 增加一个 MUX，用于选择下一个 PC 值，即选择是由 PC+4 还是由目标地址计算得到的新 PC 值。
- 修改的控制信号：
 - 增加一个 j 指令的控制信号，用于指示当前指令是否为 j 指令。
 - 修改 MUX 的控制信号，使其根据 j 指令的控制信号选择下一个 PC 的来源。
 - 修改控制信号的传递和控制逻辑，以支持 j 指令的取指和执行阶段。

下面是多周期 MIPS 处理器数据通路修改后的示意图：





控制信号：

- j 指令控制信号：用于指示当前指令是否为 j 指令。
- MUX 控制信号：用于选择下一个 PC 的来源，如果是 j 指令，则选择目标地址寄存器的值，否则选择 PC+4。
- 其他控制信号的传递和控制逻辑需要根据具体的多周期 MIPS 处理器设计进行修改。

下面是多周期 MIPS 处理器数据通路 j 指令取指周期和执行周期的操作流程及控制信号：

取指周期：

1. 取指阶段：
 - 从指令存储器（Instruction Memory）读取指令。
 - 控制信号：j 指令控制信号为高，通知控制器当前指令为 j 指令。

执行周期：

1. 取指阶段：
 - 控制信号：无特殊操作。
2. 译码阶段：
 - 无需特殊操作，继续传递控制信号。
3. 执行阶段：
 - ALU 将 PC+4 作为第一个操作数。
 - imm26 经过符号扩展后与 00 进行拼接，作为第二个操作数。
 - 控制信号：无特殊操作。
4. 访存阶段：
 - 无需特殊操作，继续传递控制信号。
5. 写回阶段：
 - 无需特殊操作，继续传递控制信号。
6. PC 更新阶段：
 - 目标地址寄存器存储着扩展后的 imm26。

- MUX 控制信号选择目标地址寄存器的值作为下一个 PC 的来源。