1.1 增强1和增强2各占25%, 增强3占多大才能使总体加速比为10？

答：(1)Assume: the fraction of the time enhancement 3 must be used to achieve an overall speedup of 10 is x.

So , x=45%

假设增强1, 增强2, 增强3分别占25%、35%、10%，没改进的时间占改进后总时间的百分比？

(2)Assume:The total execution time before the three enhancements can be used is Timebefore ，The execution time for no enhancement is Timeno.

The total execution time after the three enhancements can be used is Timeafter

So，

可能的三种增强分别占15%,15%,70%，如果只要一种增强, 应该用哪种? 如果要用两种增强, 应该用哪两种？

(3)By

If only one enhancement can be implemented：

So，we must select enhancement 1 and 3 to maximize performance.

So，we must select enhancement 1 and 3 to maximize performance.

1.2 假设一个图形操作在一个原因中占10%的时间。通过加入特殊硬件可将此部分速度提高18倍。进一步，还可以再加一倍硬件将速度提高36倍。问：这额外再加的一倍硬件值得吗？

答：

So，It is not worth exploring such an further architectural change.

1.3 在很多要求实时响应的实际应用中, 计算载荷通常是固定的。随着在并行计算机中处理器数量的增加, 这个固定载荷被分配到多处理器中并行执行。假设W的20%需要串行执行，而80%可用4个结点同时执行，加速比是多少？

答：

So，a fixed-load speedup is 2.5.

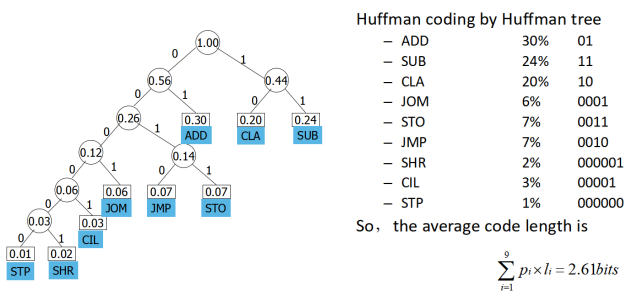
2.1

9条指令及其频率. 有若干GPR, 存储器字节寻址, 对齐, 存储器字宽16位.

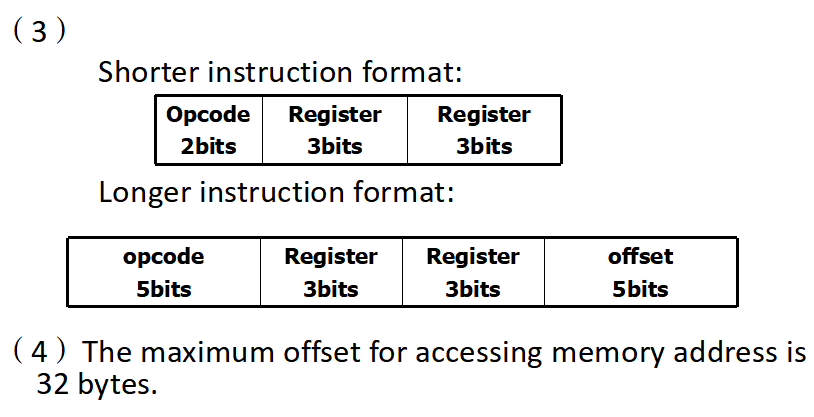
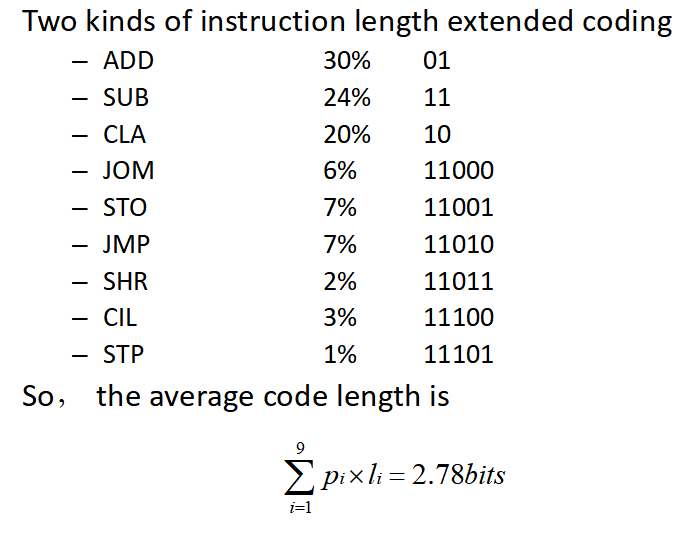
9条指令的特点:2个操作数，2种指令长度，扩展编码，短指令操作数格式:R-R，长指令操作数格式:R-M

位移寻址模式

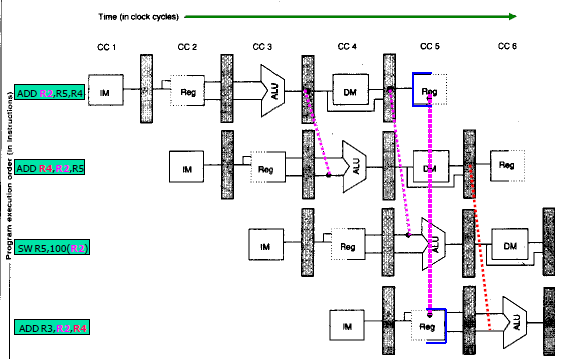
（1）



（2）



3.1在下面的代码中辨别出所有的数据相关. 哪些相关是用前送解决的数据冒险?



3.2 如何修改下面代码,来利用一个延迟槽？

LW R2,100(R3)

Loop: ADDI R3,R3,#4

BEQ R3,R4,Loop

Delayed branch slot🡪 LW R2,100(R3)

3.3

假设R3的初始值是R2+396。 采用经典的RISC五段整数流水线, 假设所有的存储器访问需要一个时钟周期

A.显示在无任何前送或旁路,但寄存器读写可在一个周期进行的情况下指令序列的时序情况. 假设转移是通过清空流水线来解决. 如果所有的存储器访问需要一个周期, 这个循环需要多少个周期完成

B.显示出带有正常的前送和旁路情况下的时序情况。假设对分支的预测是不转移。如果存储访问需要一个周期，这个循环需要多少个周期完成？

C.假设RISC流水线有单周期的分支延迟和正常的前送和旁路. 对包括延迟槽的循环中的指令进行调度. 你可以重排指令和修改指令的操作数, 但不能做其它改变如改变指令数量和操作码. 显示出流水线时序图并计算执行这个循环所需的周期数。

指令1 2关于R1 RAW

指令2 3关于R1 RAW

指令4 5关于R2 RAW

指令5 6关于R4 RAW

答：A. ·The loop iterates 396/4=99 times.

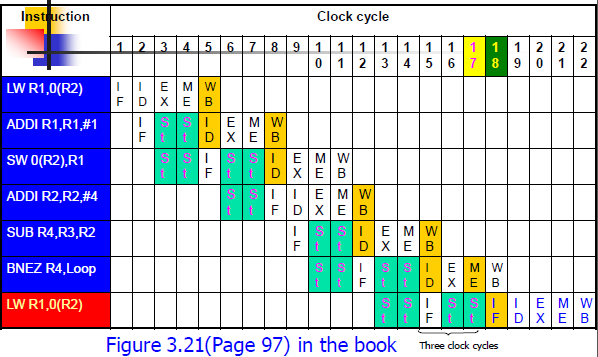
·Go through one complete iteration of the loop and the first instruction in the next iteration.

·Total length=the length of iterations 0 through 97(The first 98 iterations should be of the same length) +the length of the last iteration.

·We have assumed the version of DLX described in Figure 3.21(Page 97) in the book,which resolves branches in MEM.在MEM阶段才能解决转移问题

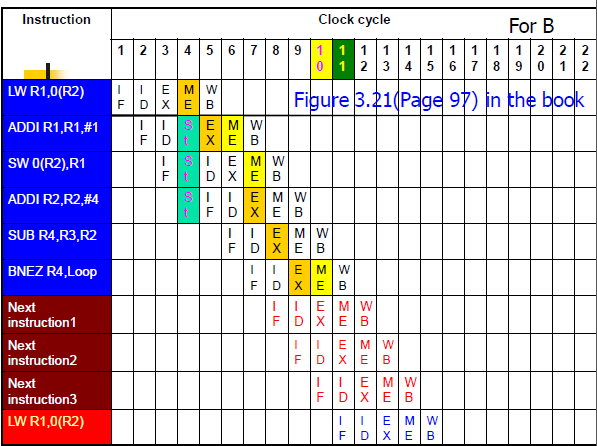
·From this Figure, the second iteration begin 17 clocks after the first iteration and the last iteration takes 18 cycles to complete.在图中, 第二次迭代开始于第一次迭代的17周期之后, 最后一次迭代需要18个周期

·Total length=17×98+18=1684 clock cycles



B. ·From this Figure, the second iteration begin 10 clocks after the first iteration and the last iteration takes 11 cycles to complete.

·Total length=10×98+11=991 clock cycles



C. Loop: LW R1,0(R2); load R1 from address 0+R2

ADDI R1,R1,#1; R1=R1+1

SW 0(R2),R1; store R1 at address 0+R2

ADDI R2,R2,#4; R2=R2+4

SUB R4,R3,R2; R4=R3-R2

BNEZ R4,Loop; Branch to loop if R4!=0

Reorder instructions to :

Loop: LW R1,0(R2); load R1 from address 0+R2

ADDI R2,R2,#4; R2=R2+4

SUB R4,R3,R2; R4=R3-R2

ADDI R1,R1,#1; R1=R1+1

BNEZ R4,Loop; Branch to loop if R4!=0

SW -4(R2),R1; store R1 at address 0+R2

·From Figure the second iteration begin 6 clocks after the first iteration and the last iteration takes 10 cycles to complete.

·Total length=6×98+10=598 clock cycles

Loop: LW R1,0(R2); load R1 from address 0+R2

stall

ADDI R1,R1,#1; R1=R1+1

SW 0(R2),R1; store R1 at address 0+R2

ADDI R2,R2,#4; R2=R2+4

SUB R4,R3,R2; R4=R3-R2

stall

BNEZ R4,Loop; Branch to loop if R4!=0

stall

Loop: LW R1,0(R2); load R1 from address 0+R2

(stall) ADDI R2,R2,#4; R2=R2+4

ADDI R1,R1,#1; R1=R1+1

SW -4(R2),R1; store R1 at address 0+R2

SUB R4,R3,R2; R4=R3-R2

stall

BNEZ R4,Loop; Branch to loop if R4!=0

stall

Loop: LW R1,0(R2); load R1 from address 0+R2

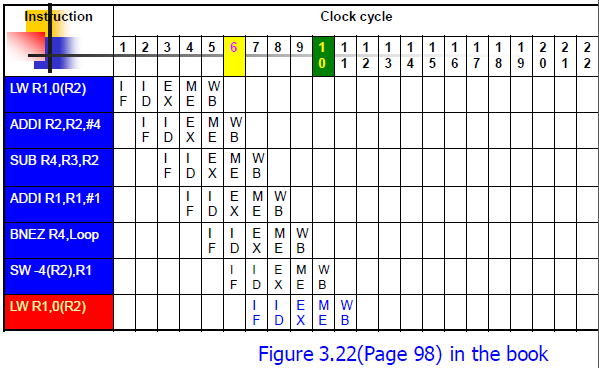
(stall) ADDI R2,R2,#4; R2=R2+4

SUB R4,R3,R2; R4=R3-R2

(stall) ADDI R1,R1,#1; R1=R1+1

BNEZ R4,Loop; Branch to loop if R4!=0

(stall) SW -4(R2),R1; store R1 at address 0+R2



ADDI R1,R0,#400

Loop: LW R2, 0(R1); load R1 from address 0+R2

stall

ADDI R4, R2, #30; R1=R1+1

SW 0(R1), R4; store R1 at address 0+R2

SUBI R1, R1, #4; R2=R2+4

stall

BNEZ R1, Loop; Branch to loop if R4 != 0

stall

ADDI R1,R0,#400

Loop: LW R2, 0(R1); load R1 from address 0+R2

stall SUBI R1, R1, #4

ADDI R4, R2, #30; R1=R1+1

SW +4(R1), R4; store R1 at address 0+R2

BNEZ R1, Loop; Branch to loop if R4 != 0

stall

4.1 只考虑数组访问产生的cache活动, 假设整数为字的大小. 当cache是直接映像且stride=132时, 预计的缺失率是多少？当stride=131时呢？如果将cache是2-路组相联时情况又会怎样？

答：If stride=132 and the cache is direct-mapped如果stride=132且cache是直接映像的 Page 201、211

The number of blocks in the cache is 256/16=16 cache中的块数=16

For array[0]: 对于array[0]

\* The block number = 0/4 =0 块号=0

\* it maps to cache number = 0 mod16 = 0, tag=0/16=0 映射到cache的入口号=0

For array[132]: 对于array[132]

\* The block number = 132/4 =33 块号=33

\* it maps to cache number : 33 mod 16=1, tag=33/16=2 映射到cache的入口号=1

So, 2 compulsory misses on the first iteration of i, and 9999\*2 hits thereafter,

and the miss rate =2/(2\*10000)=1/10000 所以, i的头一个迭代是两次强迫缺失, 然后是9999×2次的命中,所以缺失率是1/10000.

If stride=131 and the cache is direct-mapped

The block number of the cache is 256/16=16

For array[0]:

\* The block number = 0/4 =0

\* It maps to cache number = 0 mod16= 0, tag=0/16=0

For array[131]:

\* The block number = 131/4 = 32

\* It maps to cache number = 32 mode 16 =0, tag=32/16=2

So, 1 compulsory miss and 1 conflict miss on the first iteration, and 2 conflict misses on every following iterations, and the miss rate=1.

If stride=132 and the cache is two-way set associative Page 224-227、211

The block number of the cache is 256/16=16

The number of sets is: 16/2=8

Each set contains 2 blocks

For array[0]:

\* The block number = 0/4 =0

\* It maps to set number = 0 mod 8 =0, Tag=0/8=0

\* It can be placed in block 0 of set 0.

For array[132]:

\* The block number = 132/4 =33

\* It maps to set number = 33 mod 8 = 1, tag=33/8=4

\* It can be placed in block 0 of set 1

So, The block address of array[132] = 132\*4/16 =33

So, 2 compulsory misses on the first iteration of i, and 9999\*2 hits thereafter,

and the miss rate =2/(2\*10000)=1/10000

If stride=131 and the cache is two-way set associative

The block number of the cache is 256/16=16

The number of sets is: 16/2=8

Each contains 2 blocks

For array[0]:

\* The block number = 0/4 =0

\* It maps to set number = 0 mod 8 =0, Tag=0/8=0

\* It can be placed in block 0 of set 0.

For array[131]:

\* The block number = 131/4 = 32

\* It maps to set number = 32 mod 8, Tag=32/8=4

\* It can be placed in bock 1 of set 0

So, 2 compulsory misses on the first iteration of i, and 9999\*2 hits thereafter,

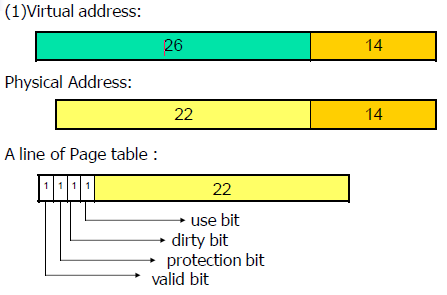
and the miss rate =2/(2\*10000)=1/10000

4.2

一个虚拟存储系统有如下特性 40位虚拟字节地址, 16KB页大小, 36位物理字节地址

(1) 求页表大小(用4个其他bit, 假设不存储磁盘地址)

(2) 假设虚拟存储系统用2-路组相联TLB实现,这个有256个TLB入口.画出虚-实映射图.



So,the total size of the page table for each process on this machine is:

2(40-14) ×(4+(36-14))bit=226×26bit=208M(Byte)

