相关输入输出引脚以及子电路封装已经在工程文件为 storage. circ 的 MIPS regfile 子电路中实现，具体输入输出引脚如图 3.11 所示。

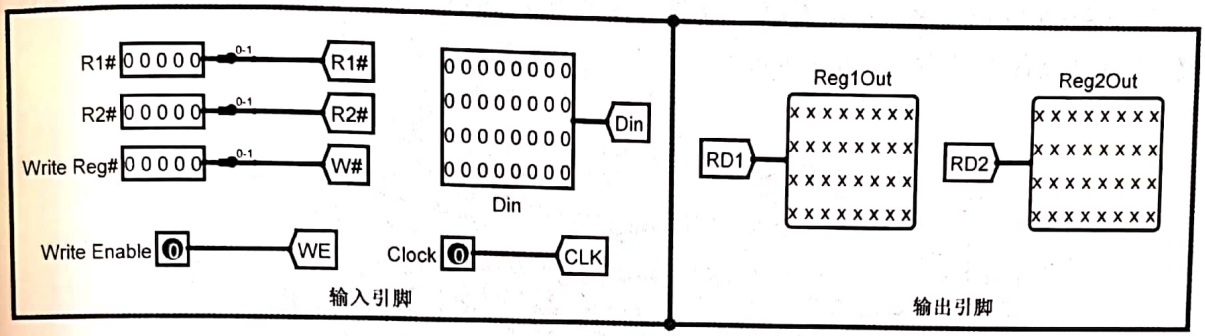

图 3.11 MIPS 寄存器文件输入输出引脚

为减少实验中绘图工作量，实验工程文件中对 5 位寄存器地址进行了简化，具体见图 3.11 左上角的引脚示意图，图中采用分线器将 5 位寄存器编号低两位引出，实际只使用了两位编号，所以最终只需实现 4 个寄存器，其中 0 号寄存器的值仍然是恒零。后续 CPU 实验中如需使用 32 个寄存器的 MIPS 寄存器文件，将提供标准组件供用户使用。

3.4.4 实验思考

(1) 0 号寄存器的值恒零，具体是如何实现的？如何实现成本最优？MIPS 指令集中为什么要引入一个恒零的寄存器？

(2) 本实验中利用译码器和解复用器均可实现 MIPS 寄存器文件中的写入控制，尝试用两种不同的方案实现。

3.5 Cache 软件仿真实验

3.5.1 实验目的

理解 Cache 3 种不同的地址映射机制，理解不同访问序列对 Cache 性能的影响，能通过 Cache 仿真软件分析、验证 Cache 数据加载的正确流程。

3.5.2 背景知识

Cache 的基本思想是在处理器附近增加一个隐藏的小容量快速存储器，将经常访问的热数据副本存放在 Cache 中，充分利用程序局部性提升 Cache 的命中率。其中，通过数据块预读策略优化空间局部性，利用 LRU 调度算法优化时间局部性，使得大多数访存请求都可以在快速存储器中获得，避免访问慢速的二级存储器，从而提升存储系统的整体性能。

主存与 Cache 均划分成大小相同的数据块，块的大小对 Cache 有较大影响，块过小无法利用预读策略优化空间局部性，块过大将使得 LRU 调度算法无法充分利用时间局部性。如何将主存数据块放置到 Cache 槽/行中(存放数据块副本和描述信息的逻辑结构)，就是 Cache 块地址映射问题，Cache 映射策略包括直接相联映射、全相联映射和组相联映射 3 种，这 3 种映射方式都可以利用组相联映射统一说明，图 3.12 是组相联映射策略示意图。其中

Cache 高速缓冲区分为 n 组，每组 k 个数据块（其中，$0<k\leqslant$Cache 块数），称为 k 路组相联，图 3.12 中描述的是 2 路组相联。主存地址细分为标记字段（t 位）、索引字段（s 位，$2^s=n$）、字地址字段（b 位）3 个部分，由索引字段部分决定主存块应该放置到 Cache 的哪个组，组内 k 个数据块可以任意放置，具体放置在哪一行取决于 LRU 调度算法。

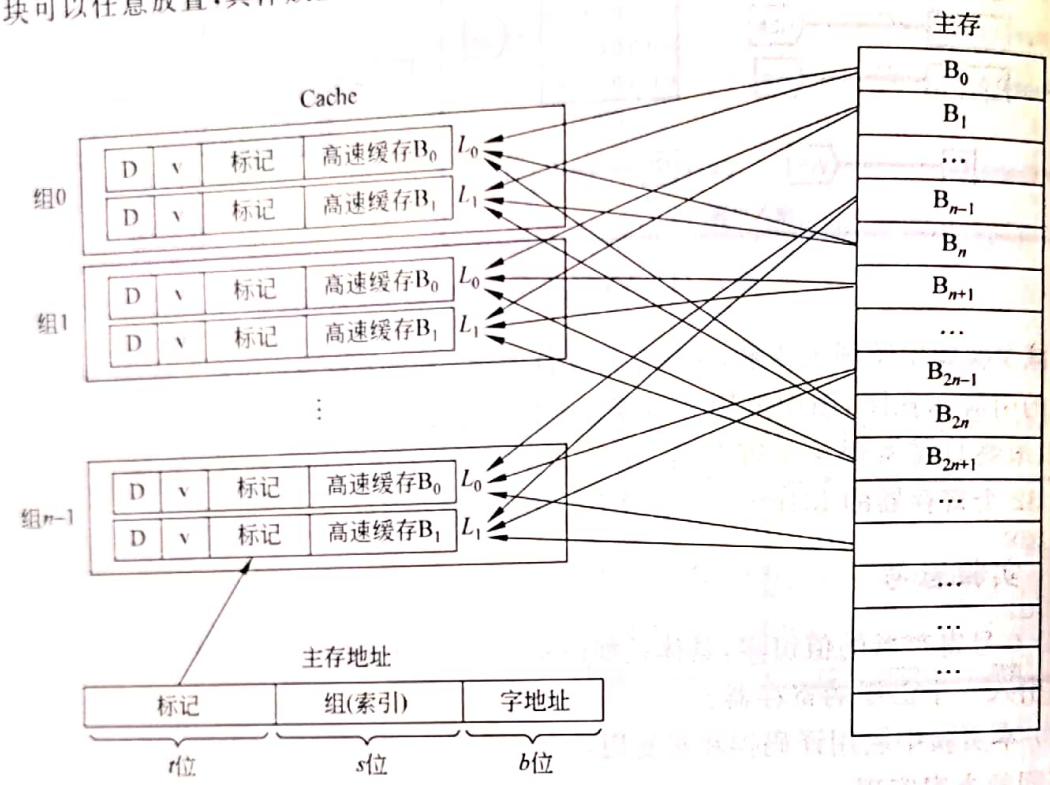

图 3.12　组相联映射策略

以主存 B_0 块为例，其索引字段为零，所以只能映射到第 0 组，至于映射到第 0 组内 k 路中具体哪一路并没有限制，但数据载入具体 Cache 行时要将标记位更新，方便下次查找。B_1 块的索引字段部分为 1，所以只能映射到第 1 组，同样 B_n 组的索引字段也是 0，也只能映射到第 0 组。

组相联进行数据查找时，首先利用主存块地址中的索引字段定位具体 Cache 组，然后将标记字段与组内所有行的标记以及有效位进行全相联并发比较，如果有相符的，则表示命中；否则数据缺失，需要直接访问主存，并且将请求数据所在的数据块载入 Cache 以方便后续访问。

1 路组相联映射就是直接相联映射，此时所有主存块都只能唯一地对应到 Cache 的某一行，数据查找时只需要将对应的 Cache 行中的标记位与主存地址中的标记位比较一次即可判断是否命中，直接相联映射查找容易，淘汰简单，但命中率较低。当 $k=$Cache 块数时，整个 Cache 只有一组，这就是全相联映射方式，主存块可以映射到 Cache 任意一块中，这种方式在数据查找时并发比较的硬件成本最高，淘汰算法复杂，但是命中率最高。

3.5.3　实验内容

CAMERA 是利用 Java 开发的一款用于高速缓存 Cache 和内存资源分配模拟的仿真软件，如图 3.13 所示，可以用于模拟仿真 Cache 的不同映射策略方案和虚存管理，能有效地帮

助用户更好地理解 Cache、虚拟存储器相关概念。

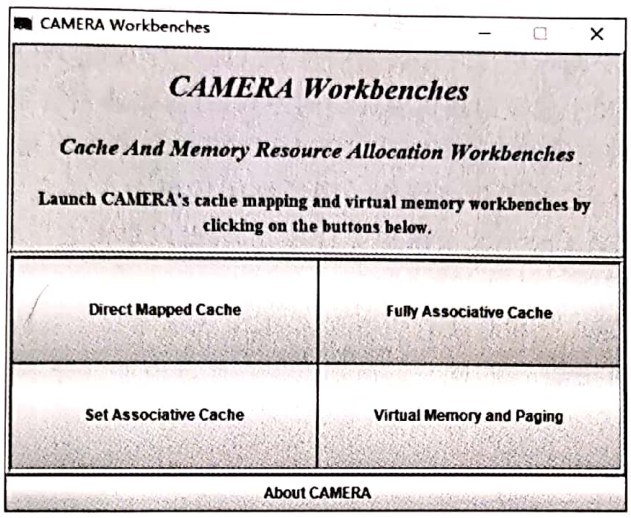

图 3.13　CAMERA——高速缓存与内存资源分配仿真软件

利用 CAMERA 仿真软件依次仿真 1 路组相联(直接相联映射),2 路组相联映射,4 路组相联映射,全相联映射,对于每一种映射方式,利用随机生成的访问序列进行数据载入仿真,对于每一次内存访问,仔细观察内存地址如何分成标记位 TAG、索引 Index、字地址 Word。另外,仔细观察除了请求数据外,还有哪些数据同时被装载到 Cache 中。注意,CAMERA 仿真器中地址序列的地址是字地址,实际计算机系统中地址序列应该是字节地址。

图 3.14 是 CAMERA 直接相联映射的仿真界面,左侧区域是 Cache 区域,Cache 缓冲区

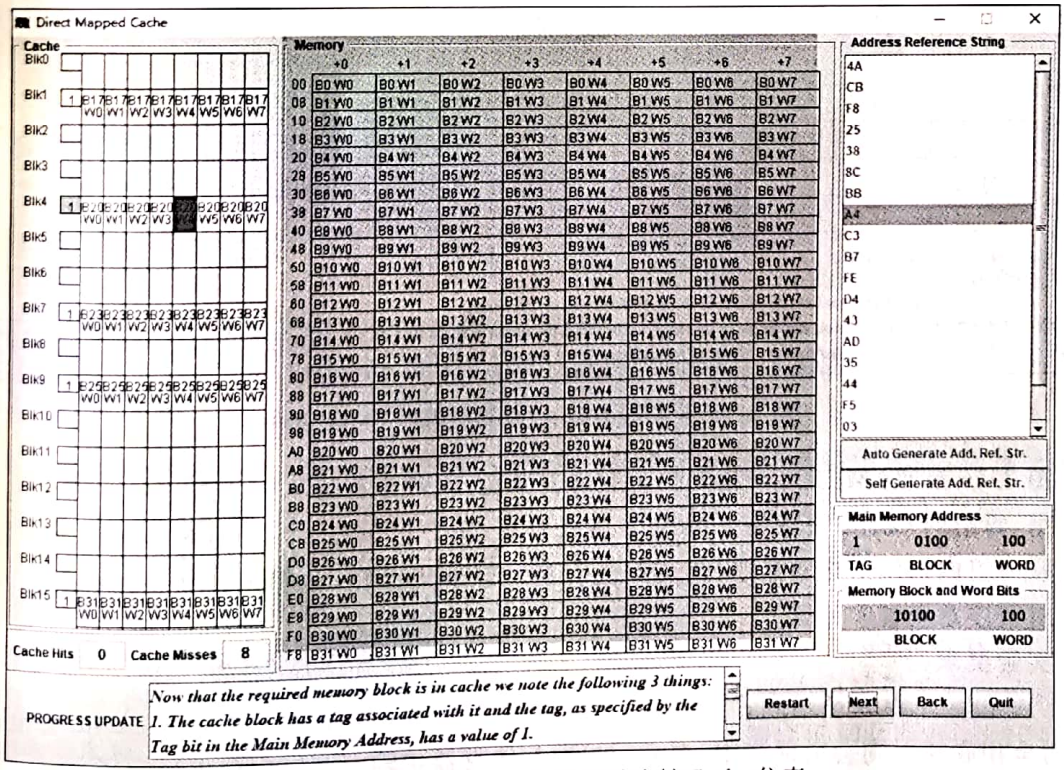

图 3.14　CAMERA——直接相联映射 Cache 仿真

包括 16 个数据块，每个数据块存放 8 个字，每个数据块包括一个 TAG 标记，存放在每一个
左侧小矩形中。中间是物理内存区域，包含 32 个数据块，每块 8 个字，总共 256 个字。右下
角主存地址共 8 位，细分为标记位 TAG、索引位 BLOCK、字地址 WORD。右侧是内存地址
访问序列，可以通过自动生成按钮随机生成访问序列，也可以通过自定义按钮输入用户定义
的访问序列。左下角的进度更新 PROGRESS UPDATE 文本框将用文本详细解释每一步
内存访问时发生了什么，进行了什么操作。请仔细阅读每一步的文字提示，以帮助加深对
Cache 机制的理解。进行 Cache 调度模拟仿真时，用户可以单击右下角区域的 Next 按钮逐
行仿真，单击 Back 按钮还可以回滚。

图 3.15 是 CAMERA 全相联映射的仿真界面，其界面布局、Cache 块数、块大小、内存大
小等均与直接相联映射完全一致，唯一的区别是主存储器地址划分不同，仿真时请注意观察
和分析。

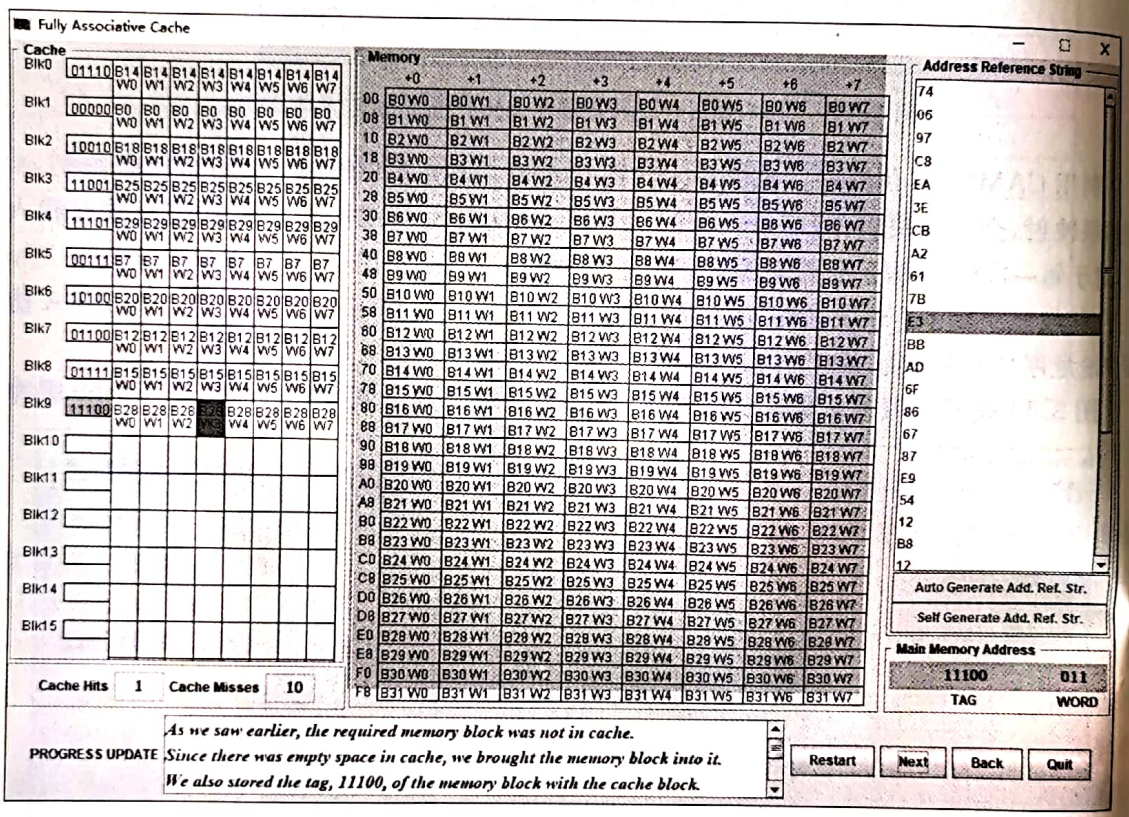

图 3.15　CAMERA——全相联映射 Cache 仿真

图 3.16 是 CAMERA 组相联映射的仿真界面，它和全相联不同的是，Cache 若干块合并
成一组，每组的 Cache 块数为 k，称为 k 路组相联，用户可以在 CAMERA 中设置这个参数，
组相联中组内采用全相联策略，组间采用直接相联映射策略，主存地址划分也有略微区别，
仿真时请注意观察和分析。

实验要求：

(1) 设计一组地址访问序列，当这组访问序列循环时，直接相联映射 Cache 会全部缺
失，而全相联映射则会全部命中，在 CAMERA 仿真器中输入对应序列进行验证。

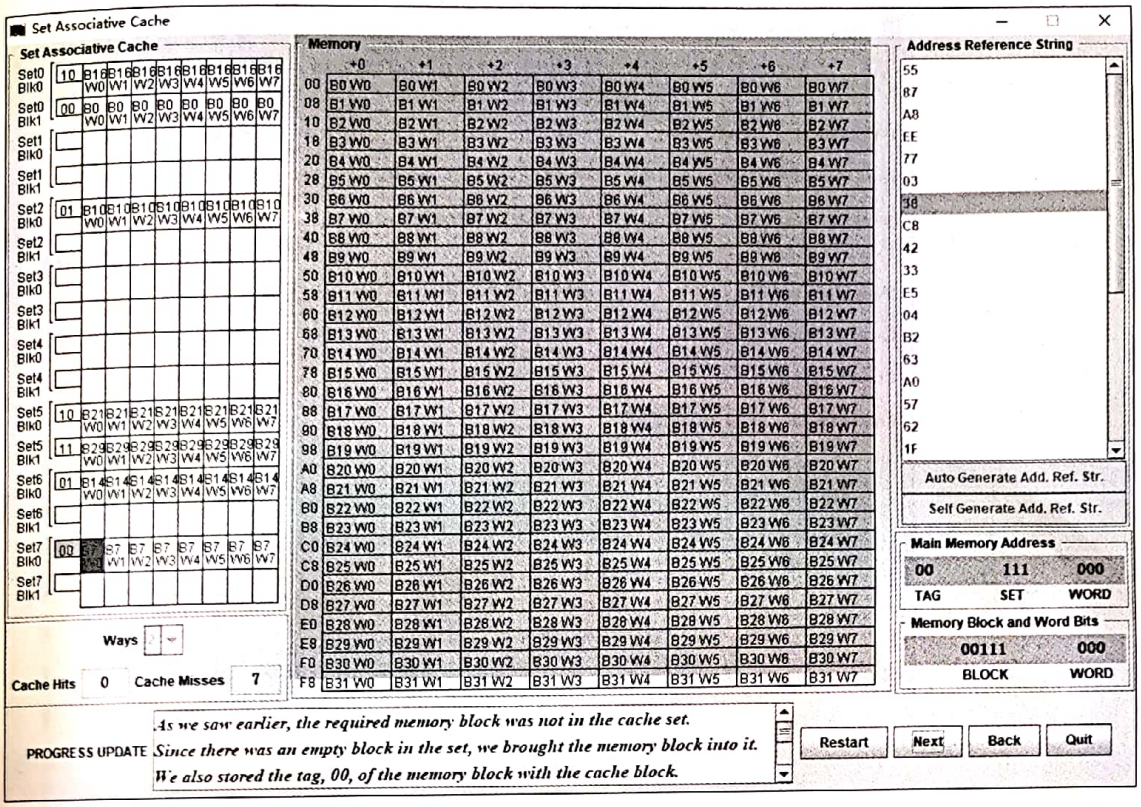

图 3.16　CAMERA——组相联映射 Cache 仿真

(2) 设计一组地址访问序列，当这组访问序列循环时，全相联映射 Cache 会全部缺失，但直接相联映射则缺失率不高，在 CAMERA 仿真器中输入对应序列进行验证。

3.6　Cache 性能分析实验

3.6.1　实验目标

理解程序局部性原理，理解 Cache 参数对 Cache 命中率的影响，理解内存访问模式对 Cache 命中率的影响，能利用相关知识提升程序性能。

3.6.2　背景知识

本实验将使用 MARS 仿真器进行，MARS 是一款 MIPS 汇编程序仿真器，具体参考第 10 章的 MARS 用户指南，本次实验将在 MARS 中运行一个名为 Cache. s 的汇编程序，并且利用数据 Cache 仿真插件对程序的执行性能、Cache 性能等进行精准分析。

数据 Cache 仿真插件如图 3.17 所示，该插件可以配置 MIPS 数据 Cache 地址映射策略、替换算法、Cache 容量、Cache 块数、Cache 块大小、Cache 组块数，Cache 块数等参数。当用户开启该插件并单击连接 MIPS 处理器按钮时，程序执行过程中会提供主存访问次数、Cache 命中次数、Cache 缺失次数、Cache 命中率的实时统计，界面底部的运行日志窗口还可以提供详细的 Cache 运行日志。

3.8 虚拟存储器软件仿真实验

3.8.1 实验目的

熟悉虚拟存储系统框架结构以及工作流程，能够利用内存访问可视化插件进行虚存页面访问流程的分析。

3.8.2 背景知识

虚拟存储器中有3种地址空间，第一种是虚拟地址空间，也称为虚拟空间，它是应用程序员用来编写程序的地址空间；第二种是主存的地址空间，也称物理地址空间或实地址空间；第三种是辅存地址空间，也就是磁盘存储器的地址空间。与这3种地址空间分别对应的是3种地址，即虚拟地址（虚地址）、物理地址（实地址）和磁盘存储器地址（也称磁盘地址或辅存地址），本节主要讨论页式虚拟存储器相关内容。

页式虚拟存储器中，虚拟空间和主存空间均划分成固定大小的页。虚拟地址被划分成虚拟页号（Virtual Page Number，VPN）和虚拟页偏移量（Virtual Page Offset，VPO）；同时物理地址被划分成物理页号（Physical Page Number，PPN）和物理页偏移量（Physical Page Offset，PPO）两部分。VPN和PPN分别构成虚拟地址和物理地址的高位部分；VPO和PPO分别构成虚拟地址和物理地址的低位地址，其位数决定了页面大小。因为物理页和虚拟页大小相同，因此VPO和PPO的位数相同。

程序运行时，CPU以虚拟地址访问主存，虚拟地址必须由存储管理单元（Memeory Management Unit，MMU）转换成物理地址才能访问主存，虚拟地址到物理地址的转换本质上就是如何将VPN转换成PPN，通常采用以VPN为索引的页表实现。每个页表项（Page Table Entry，PTE）存储一个有效位和物理页号，有效位为1时则表示页表项中的物理页号有效，否则表示缺页。页表常驻内存，相当于一个以虚拟页号VPN作为索引的数组，因此页表项的数量与虚拟空间的页数相等，如虚拟页号字段为20位，则页表中表项的数量为2^{20}，如虚拟空间过大，页表将消耗较大的内存空间，因此通常页表会采用多级结构。虚拟地址与物理地址的转换过程如图3.25所示，注意其中页表基址寄存器（Page Table Base Register，PTBR）存放的是页表首地址，不同的进程拥有不同的页表，进程切换的时候可以通过切换PTBR寄存器的值实现进程页表的快速切换。

由于页表存放在主存中，即使不发生缺页异常，也必须先访问主存，获得页表项PTE后才能实现虚拟地址到物理地址的转换，然后再利用物理地址访问主存获得数据，这大大降低了访问效率。为了提升地址转换的性能，根据局部性原理，现代处理器都维护着一个转换旁路缓冲器（Translation Look-aside Buffer，TLB）作为页表项的高速缓存。TLB用于存储经常访问的页表项（有效位、VPN、PPN），和Cache中缓存的页表块相比，其粒度更小，关联度更高，通常采用全相联或组相联的映射机制以及随机淘汰策略。

有了TLB后，虚拟地址向物理地址转换时，首先利用VPN作为关键字到TLB中进行页表项查找，如果某个页表项的VPN与之相同，且有效位为1，则TLB命中，直接获取对应表项中的PPN，生成物理地址；如果TLB缺失，则由MMU计算页表项地址（Page Table

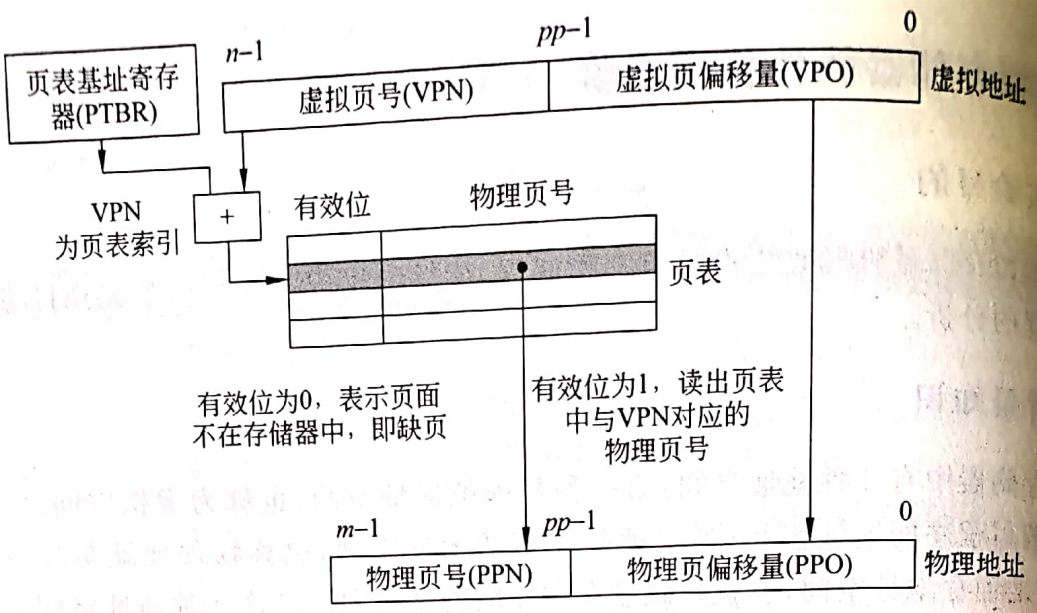

图 3.25　虚拟地址与物理地址转换过程

Entry Address,PTEA)访问主存获得页表项 PTE,实现 VPN 到 PPN 的转换;如果发页,还需要操作系统触发缺页异常程序,进行调页处理,这个过程非常漫长。另外,还需当前页的(VPN、PPN)对应关系更新到 TLB 表中以方便后续访问,注意更新 TLB 的过能会触发 TLB 淘汰。加入 TLB 后虚拟地址转换流程如图 3.26 所示。

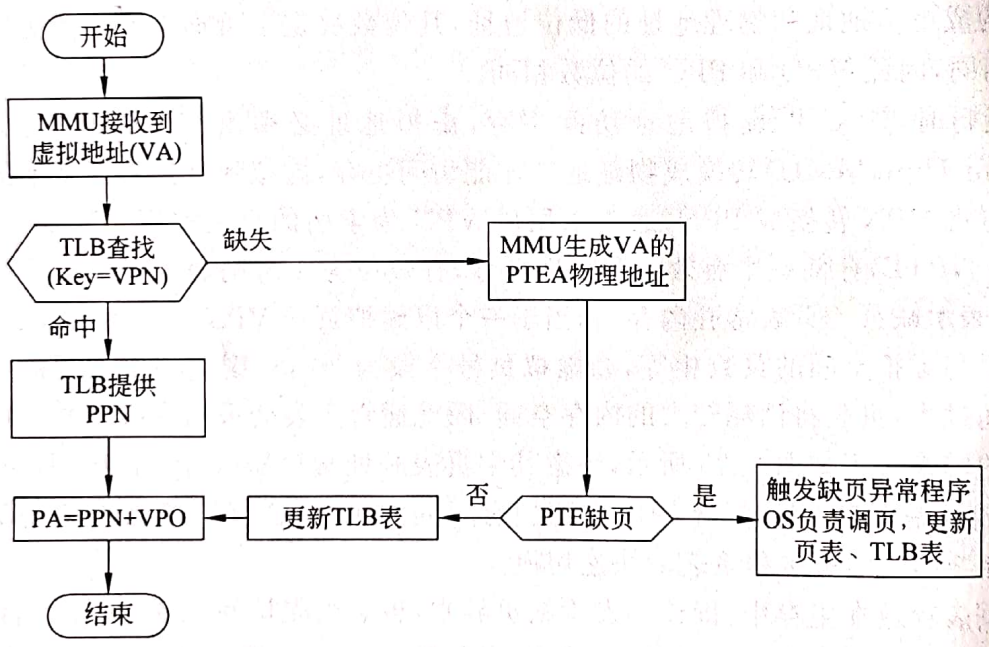

图 3.26　TLB 加速虚拟地址转换流程

3.8.3　实验内容

本实验中将使用 CAMERA 的虚存仿真功能,CAMERA 模拟仿真了一个拥有 256 虚存空间的虚存系统,其中物理地址空间为 128B,共包含 4 个物理页(Page Frame),每小为 32B;虚存系统共包含 8 个虚拟页,虚地址和实地址构成如图 3.27 所示。

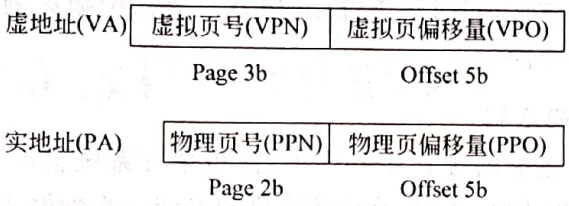

图 3.27　虚地址实地址构成

图 3.28 是 CAMERA 软件中虚拟存储器仿真的主界面，图中左上角区域显示了物理内存空间，物理内存区域右侧是 TLB 表和页表，最开始时两个表都应该为空；右上角显示的是虚地址访问序列，系统可以帮助用户自动生成随机访问序列进行仿真，用户也可以根据自己的需要定制虚地址访问序列进行仿真；左下角的进度更新 PROGRESS UPDATE 文本框将用文本详细解释每一次虚存访问发生了什么，进行了什么操作。请仔细阅读每一步的文字提示，以帮助加深对虚拟存储器的理解。进行虚存访问模拟仿真时，用户可以单击右下角区域的 Next 按钮进行仿真，单击 Back 按钮还可以回滚。

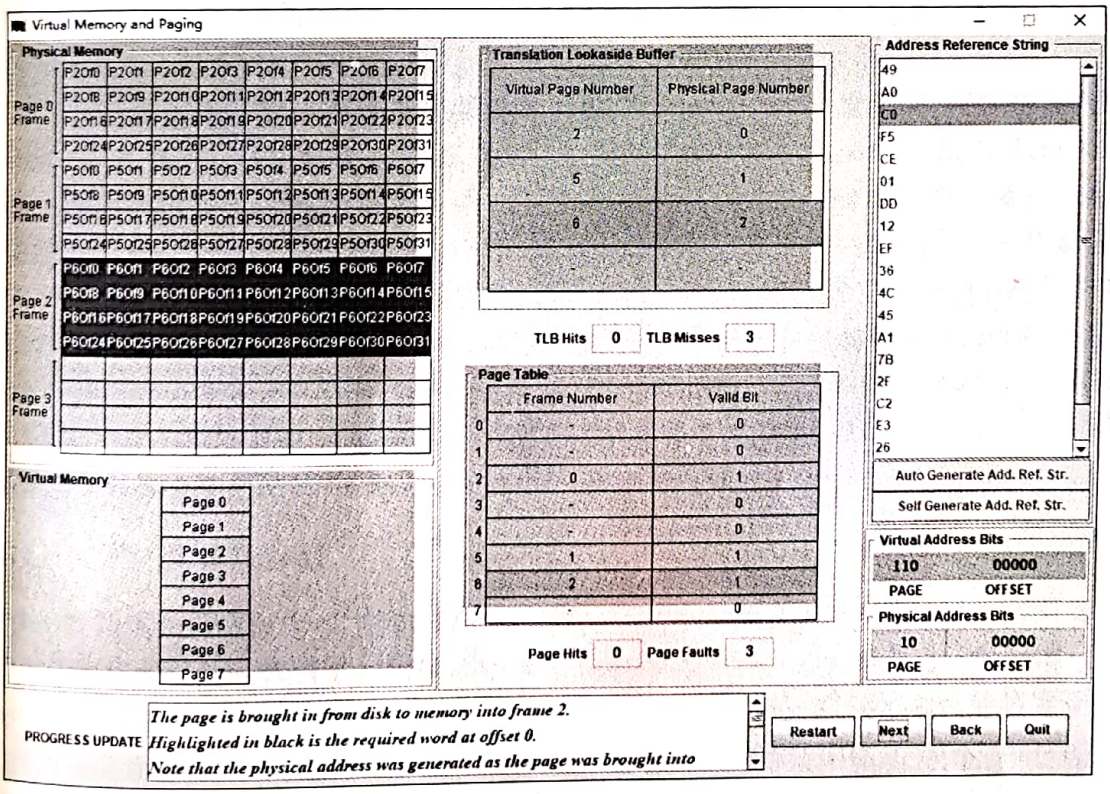

图 3.28　CAMERA——虚拟存储器仿真

实验 1(虚存载入过程观察)

单击 CAMERA 软件界面右侧的自动生成虚拟地址访问序列按钮，系统将自动生成虚拟地址访问序列，假设这些虚拟地址都是由 MIPS 指令中的 LW 指令生成的，单击 Next 按钮开始虚存模拟仿真。注意，每单击一次 Next 按钮，都要仔细阅读 PROGRESS UPDATE 文本框的文字内容，尝试真正了解每一次虚存访问 TLB 表、页表以及物理内存的变化。仿

真结束后，记下虚存访问序列，同时记下 TLB 以及页表命中缺失的次数，记下访问序列，是否发生页命中，分析对应原因。

实验 2（最糟糕的访问序列）

设计一个包含 10 个地址的虚存访问序列，使得虚存系统会产生 10 次 TLB 缺失和 10 次页缺失，设计完成后，可以利用定制地址访问序列按钮一次性输入 10 个地址，然后进行仿真验证自己的设计。

实验 3（存储系统优化）

对于实验 2 中设计的能产生 10 次 TLB 缺失以及 10 次页缺失的访问序列，能否通过调整 TLB 大小、页表大小、内存大小等其中的一个参数，使得 TBL 缺失次数仍然为 10，但页缺失次数小于 10，并给出你的答案。