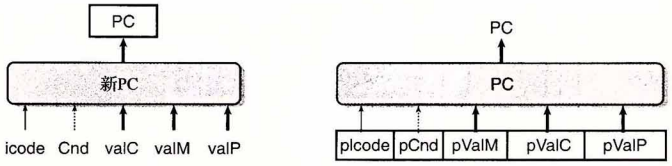
种修改过的设计为“SEQ+”。

我们移动 PC 阶段，使得它的逻辑在时钟周期开始时活动，使它计算当前指令的 PC 值。图 4-39 给出了 SEQ 和 SEQ+在 PC 计算上的不同之处。在 SEQ 中(图 4-39a)，PC 计算发生在时钟周期结束的时候，根据当前时钟周期内计算出的信号值来计算 PC 寄存器的新值。在 SEQ+中(图 4-39b)，我们创建状态寄存器来保存在一条指令执行过程中计算出来的信号。然后，当一个新的时钟周期开始时，这些信号值通过同样的逻辑来计算当前指令的 PC。我们将这些寄存器标号为“pIcode”、“pCnd”等等，来指明在任一给定的周期，它们保存的是前一个周期中产生的控制信号。



a) SEQ的新PC计算　　b) SEQ+的PC选择

图 4-39　移动计算 PC 的时间。在 SEQ+中，我们将计算当前状态的程序计数器的值作为指令执行的第一步

图 4-40 给出了 SEQ+硬件的一个更为详细的说明。可以看到，其中的硬件单元和控制块与我们在 SEQ 中用到的(图 4-23)一样，只不过 PC 逻辑从上面(在时钟周期结束时活动)移到了下面(在时钟周期开始时活动)。

旁注　SEQ+中的 PC 在哪里

SEQ+有一个很奇怪的特色，那就是没有硬件寄存器来存放程序计数器。而是根据从前一条指令保存下来的一些状态信息动态地计算 PC。这就是一个小小的证明——我们可以以一种与 ISA 隐含着的概念模型不同的方式来实现处理器，只要处理器能正确执行任意的机器语言程序。我们不需要将状态编码成程序员可见的状态指定的形式，只要处理器能够为任意的程序员可见状态(例如程序计数器)产生正确的值。在创建流水线化的设计中，我们会更多地使用到这条原则。5.7 节中描述的乱序(out-of-order)处理技术，以一种完全不同于机器级程序中出现的顺序的次序来执行指令，将这一思想发挥到了极致。

SEQ 到 SEQ+中对状态单元的改变是一种很通用的改进的例子，这种改进称为电路重定时(circuit retiming)[68]。重定时改变了一个系统的状态表示，但是并不改变它的逻辑行为。通常用它来平衡一个流水线系统中各个阶段之间的延迟。

4.5.2　插入流水线寄存器

在创建一个流水线化的 Y86-64 处理器的最初尝试中，我们要在 SEQ+的各个阶段之间插入流水线寄存器，并对信号重新排列，得到 PIPE－处理器，这里的“－”代表这个处理器和最终的处理器设计相比，性能要差一点。PIPE－的抽象结构如图 4-41 所示。流水线寄存器在该图中用黑色方框表示，每个寄存器包括不同的字段，用白色方框表示。正如多个字段表明的那样，每个流水线寄存器可以存放多个字节和字。同两个顺序处理器的硬件结构(图 4-23 和图 4-40)中的圆角方框不同，这些白色的方框表示实际的硬件组成。