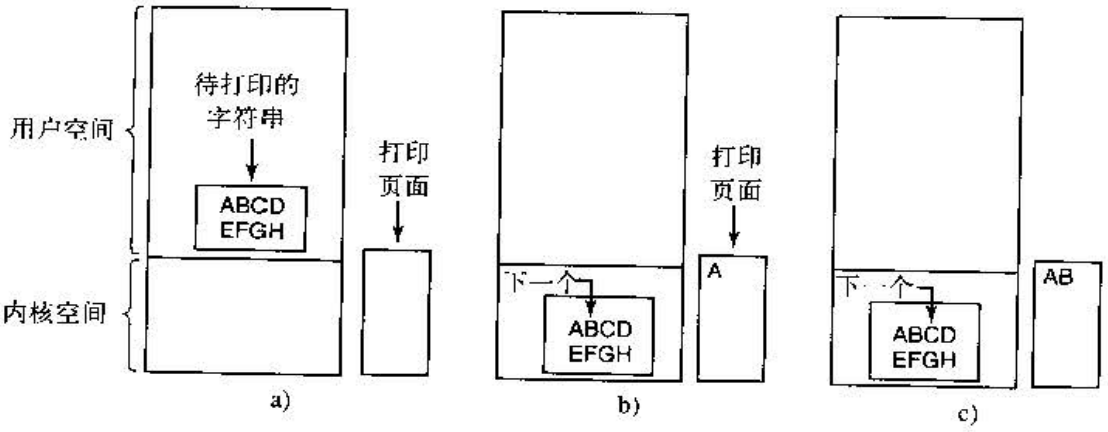

图5-7 打印一个字符串的步骤

一旦将第一个字符复制到打印机，操作系统就要查看打印机是否就绪准备接收另一个字符。一般而言，打印机都有第二个寄存器，用于表明其状态。将字符写到数据寄存器的操作将导致状态变为非就绪。当打印机控制器处理完当前字符时，它就通过在其状态寄存器中设置某一位或者将某个值放到状态寄存器中来表示其可用性。

这时，操作系统将等待打印机状态再次变为就绪。打印机就绪事件发生时，操作系统就打印下一个字符，如图5-7c所示。这一循环继续进行，直到整个字符串打印完。然后，控制返回到用户进程。

操作系统相继采取的操作总结在图5-8中。首先，数据被复制到内核空间。然后，操作系统进入一个密闭的循环，一次输出一个字符。在该图中，清楚地说明了程序控制I/O的最根本的方面，这就是输出一个字符之后，CPU要不断地查询设备以了解它是否就绪准备接收另一个字符。这一行为经常称为轮询（polling）或忙等待（busy waiting）。

```
copy_from_user(buffer, p, count);                /* p是内核缓冲区 */
for (i = 0; i < count; i++) {                    /* 对每个字符循环 */
    while (*printer_status_reg != READY) ;       /* 循环直到就绪 */
    *printer_data_register = p[i];               /* 输出一个字符 */
}
return_to_user();
```

图5-8 使用程序控制I/O将一个字符串写到打印机

程序控制I/O十分简单但是有缺点，即直到全部I/O完成之前要占用CPU的全部时间。如果“打印”一个字符的时间非常短（因为打印机所做的全部事情就是将新的字符复制到一个内部缓冲区中），那么忙等待还是不错的。此外，在嵌入式系统中，CPU没有其他事情要做，忙等待也是合理的。然而，在更加复杂的系统中，CPU有其他工作要做，忙等待将是低效的，需要更好的I/O方法。

5.2.3 中断驱动I/O

现在我们考虑在不缓冲字符而是在每个字符到来时便打印的打印机上进行打印的情形。如果打印机每秒可以打印100个字符，那么打印每个字符将花费10ms。这意味着，当每个字符写到打印机的数据寄存器中之后，CPU将有10ms搁置在无价值的循环中，等待允许输出下一个字符。这10ms时间足以进行一次上下文切换并且运行其他进程，否则就浪费了。

这种允许CPU在等待打印机变为就绪的同时做某些其他事情的方式就是使用中断。当打印字符串的系统调用被发出时，如我们前面所介绍的，字符串缓冲区被复制到内核空间，并且一旦打印机准备好接收一个字符时就将第一个字符复制到打印机中。这时，CPU要调用调度程序，并且某个其他进程将运行。请求打印字符串的进程将被阻塞，直到整个字符串打印完。系统调用所做的工作如图5-9a所示。

当打印机将字符打印完并且准备好接收下一个字符时，它将产生一个中断。这一中断将停止当前进程并且保存其状态。然后，打印机中断服务过程将运行。图5-9b所示为打印机中断服务过程的一个粗略的版本。如果没有更多的字符要打印，中断处理程序将采取某个操作将用户进程解除阻塞。否则，它将