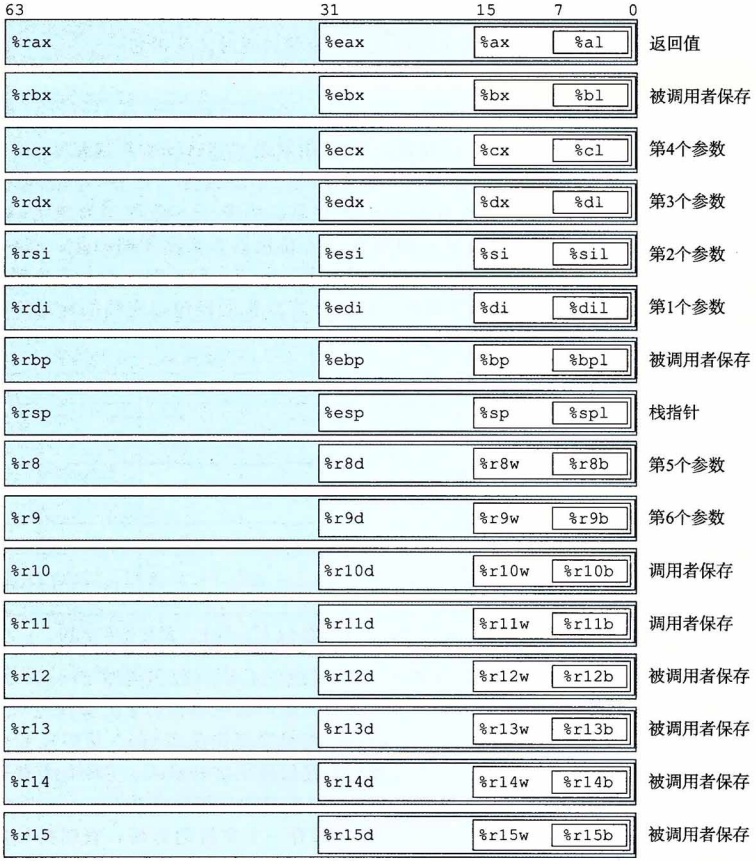

图 3-2　整数寄存器。所有 16 个寄存器的低位部分都可以作为字节、字(16 位)、双字(32 位)和四字(64 位)数字来访问

如图 3-2 中嵌套的方框标明的，指令可以对这 16 个寄存器的低位字节中存放的不同大小的数据进行操作。字节级操作可以访问最低的字节，16 位操作可以访问最低的 2 个字节，32 位操作可以访问最低的 4 个字节，而 64 位操作可以访问整个寄存器。

在后面的章节中，我们会展现很多指令，复制和生成 1 字节、2 字节、4 字节和 8 字节值。当这些指令以寄存器作为目标时，对于生成小于 8 字节结果的指令，寄存器中剩下的字节会怎么样，对此有两条规则：生成 1 字节和 2 字节数字的指令会保持剩下的字节不变；生成 4 字节数字的指令会把高位 4 个字节置为 0。后面这条规则是作为从 IA32 到 x86-64 的扩展的一部分而采用的。

就像图 3-2 右边的解释说明的那样，在常见的程序里不同的寄存器扮演不同的角色。其中最特别的是栈指针%rsp，用来指明运行时栈的结束位置。有些程序会明确地读写这个寄存器。另外 15 个寄存器的用法更灵活。少量指令会使用某些特定的寄存器。更重要的