```
9     400555:  f3 c3                repz retq                F4: Return
      .
      .
      .
10    400560:  e8 e3 ff ff ff       callq  400548 <first>    M1: Call first(10)
11    400565:  48 89 c2             mov    %rax,%rdx         M2: Resume
```

每条指令都有一个标号，类似于图 3-27a。从 main 调用 first(10)开始，到程序返回 main 时为止，填写下表记录指令执行的过程。

指令			状态值(指令执行前)					
标号	PC	指令	%rdi	%rsi	%rax	%rsp	* %rsp	描述
M1	0x400560	callq	10	—	—	0x7fffffffe820	—	调用 first(10)
F1								
F2								
F3								
L1								
L2								
L3								
F4								
M2								

3.7.3　数据传送

当调用一个过程时，除了要把控制传递给它并在过程返回时再传递回来之外，过程调用还可能包括把数据作为参数传递，而从过程返回还有可能包括返回一个值。x86-64 中，大部分过程间的数据传送是通过寄存器实现的。例如，我们已经看到无数的函数示例，参数在寄存器%rdi、%rsi 和其他寄存器中传递。当过程 P 调用过程 Q 时，P 的代码必须首先把参数复制到适当的寄存器中。类似地，当 Q 返回到 P 时，P 的代码可以访问寄存器%rax 中的返回值。在本节中，我们更详细地探讨这些规则。

x86-64 中，可以通过寄存器最多传递 6 个整型(例如整数和指针)参数。寄存器的使用是有特殊顺序的，寄存器使用的名字取决于要传递的数据类型的大小，如图 3-28 所示。会根据参数在参数列表中的顺序为它们分配寄存器。可以通过 64 位寄存器适当的部分访问小于 64 位的参数。例如，如果第一个参数是 32 位的，那么可以用%edi 来访问它。

操作数大小(位)	参数数量					
	1	2	3	4	5	6
64	%rdi	%rsi	%rdx	%rcx	%r8	%r9
32	%edi	%esi	%edx	%ecx	%r8d	%r9d
16	%di	%si	%dx	%cx	%r8w	%r9w
8	%dil	%sil	%dl	%cl	%r8b	%r9b

图 3-28　传递函数参数的寄存器。寄存器是按照特殊顺序来使用的，而使用的名字是根据参数的大小来确定的

如果一个函数有大于 6 个整型参数，超出 6 个的部分就要通过栈来传递。假设过程 P 调用过程 Q，有 n 个整型参数，且 $n > 6$。那么 P 的代码分配的栈帧必须要能容纳 7 到 n 号参数的存储空间，如图 3-25 所示。要把参数 1～6 复制到对应的寄存器，把参数 7～n 放