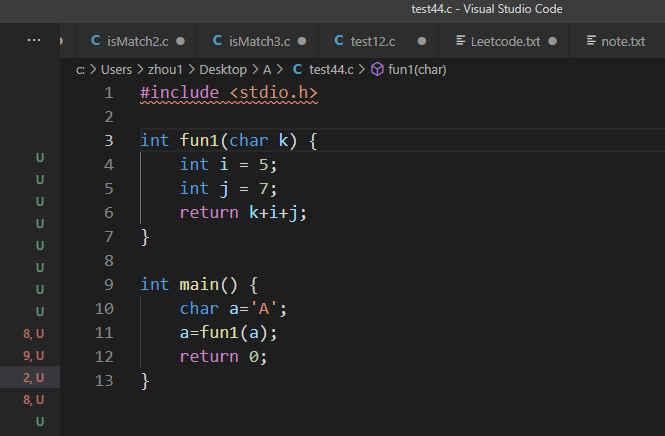
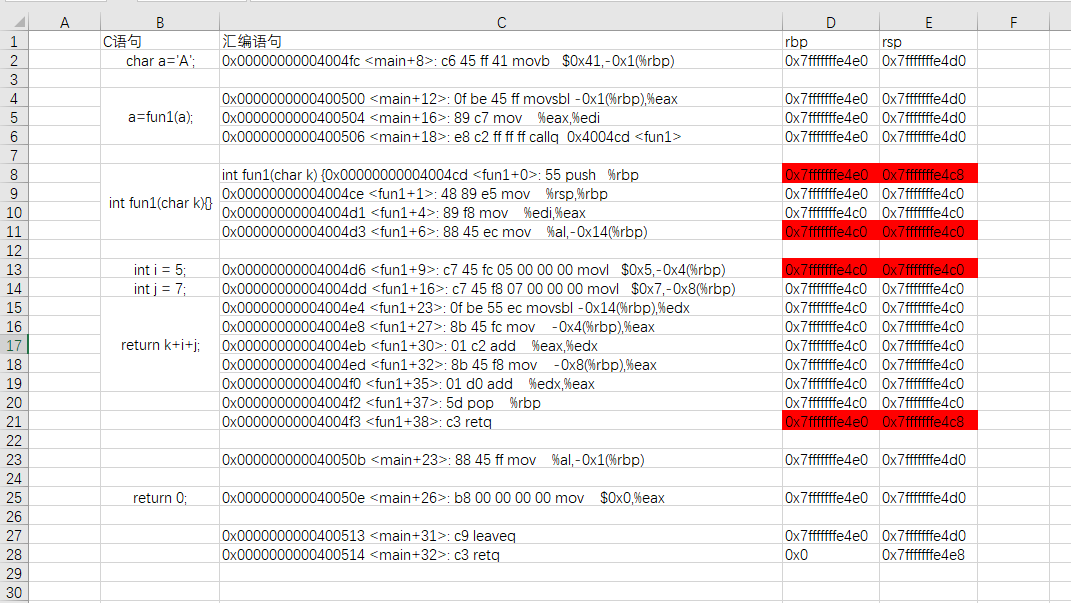
程序中堆栈是如何运行的

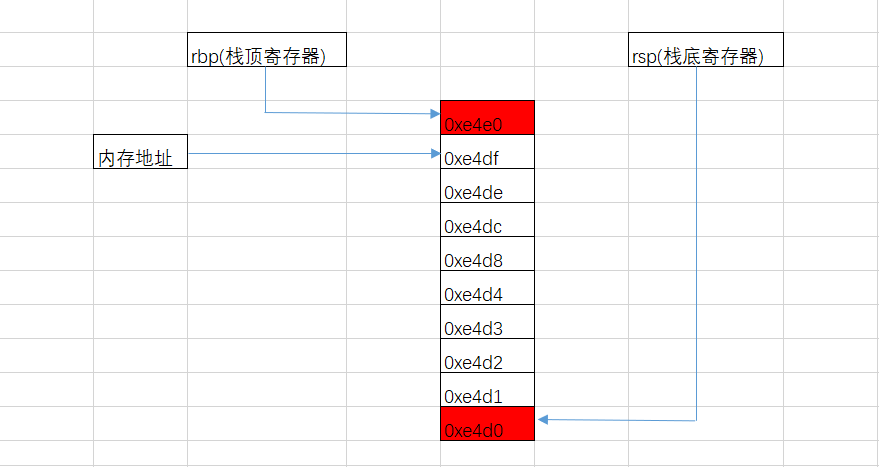
今天我们来看看堆栈是如何在程序中运行的，我们通过一个小的测试程序来进行分析：



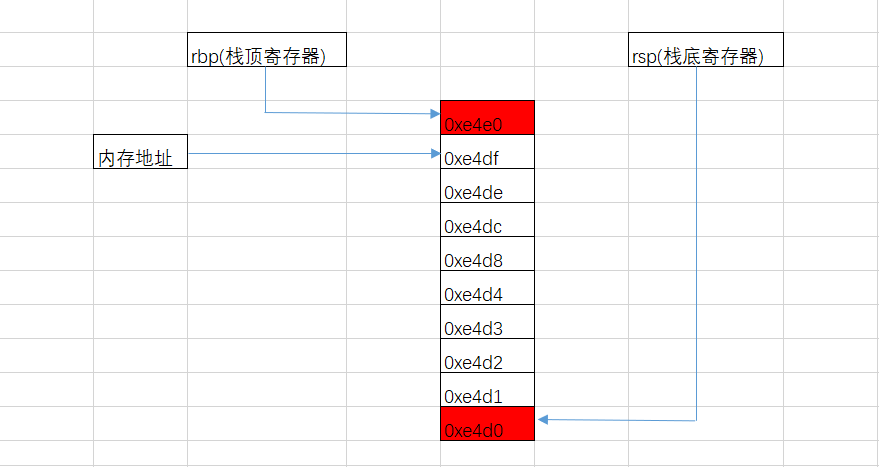
我们通过反汇编一条一条语句执行，然后查看rbp(栈顶)、rsp(栈底)寄存器的变化来进行分析：



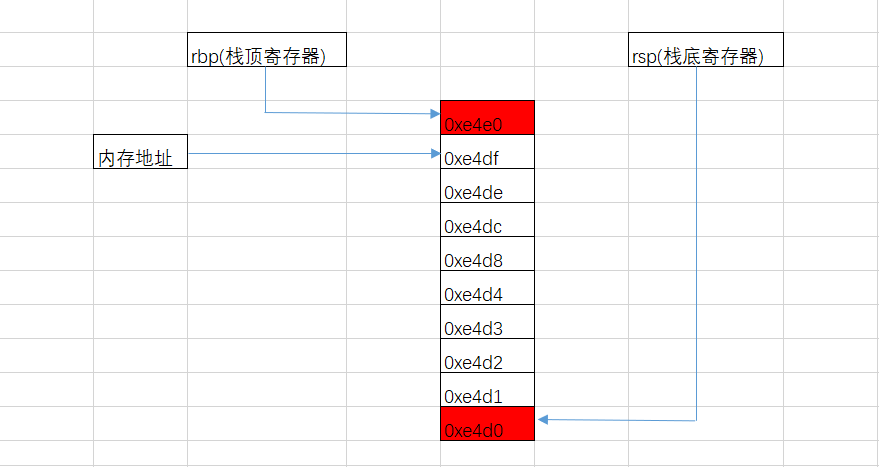
当程序运行语句char a=‘A’的时候，对应的汇编语句为（0x00000000004004fc <main+8>: c6 45 ff 41 movb $0x41,-0x1(%rbp)，rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我还是以图片的形式展示：



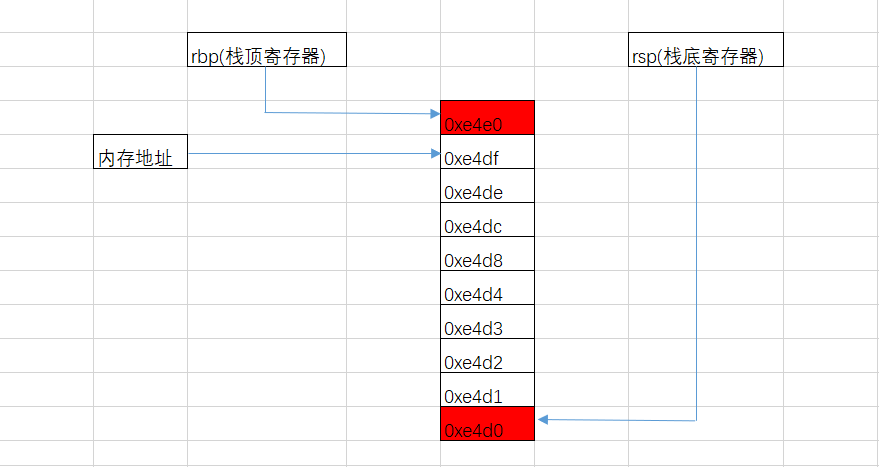
然后我们接着单步调试程序往下走(si命令)，程序跑到了a=fun1(a)这条语句，对应的汇编语句为（0x0000000000400500 <main+12>: 0f be 45 ff movsbl -0x1(%rbp),%eax），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：



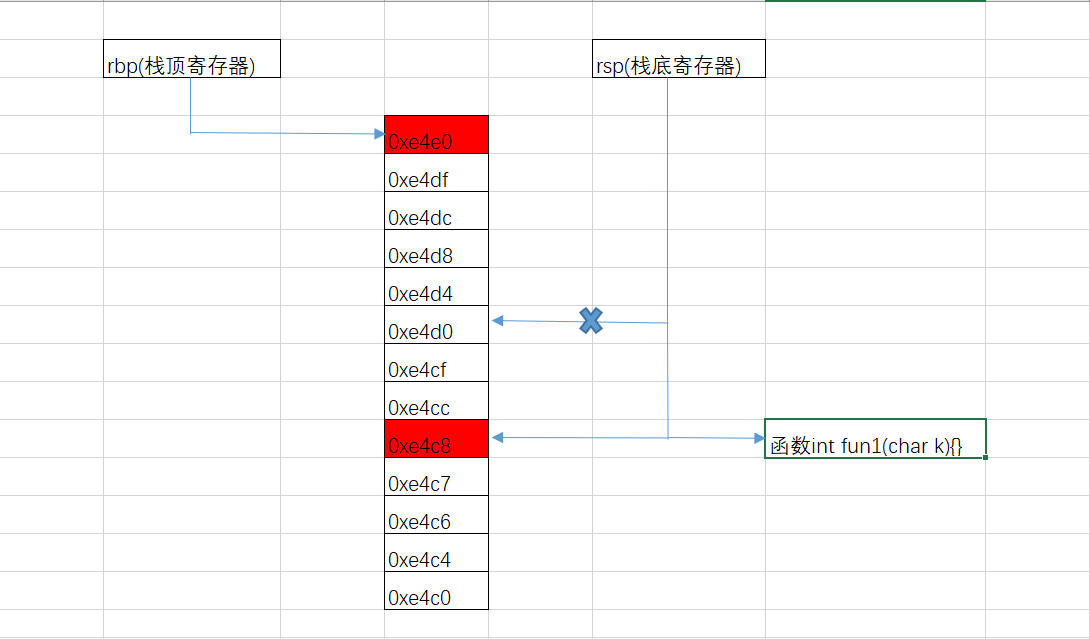
然后我们接着单步调试程序往下走(si命令)，程序还是在a=fun1(a)这条语句，对应的汇编语句为（0x0000000000400504 <main+16>: 89 c7 mov %eax,%edi），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：



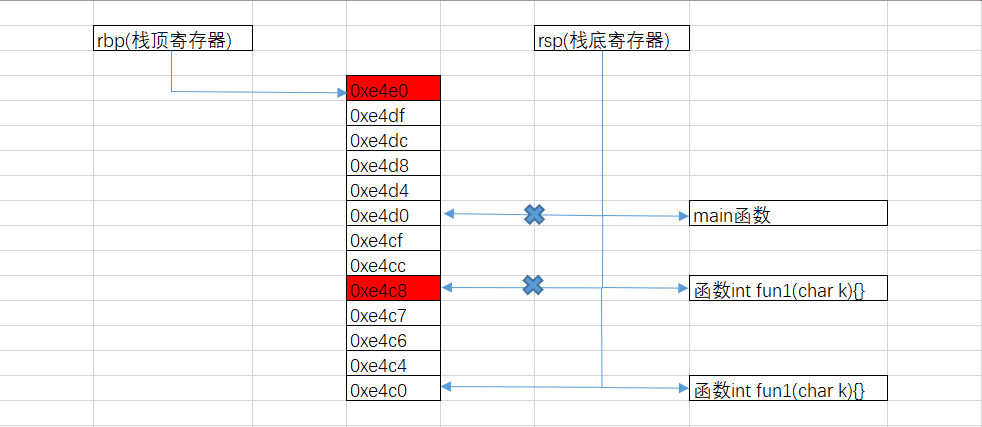
然后我们接着单步调试程序往下走(si命令)，程序还是在a=fun1(a)这条语句，对应的汇编语句为（0x0000000000400506 <main+18>: e8 c2 ff ff ff callq 0x4004cd <fun1>），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：



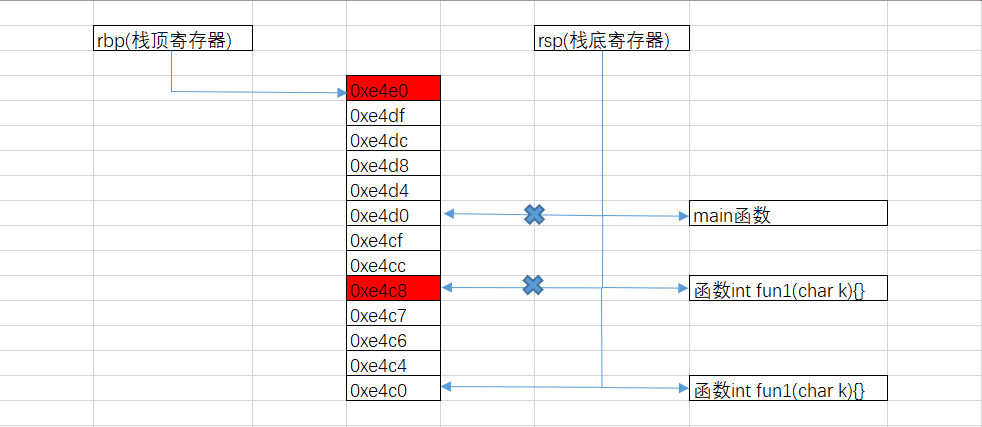
然后我们接着单步调试程序往下走(si命令)，程序进入了int fun1(char k){}这个子函数，对应的汇编语句为（int fun1(char k) {0x00000000004004cd <fun1+0>: 55 push %rbp），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4c8，为了方便表示，我继续以图片的形式展示：



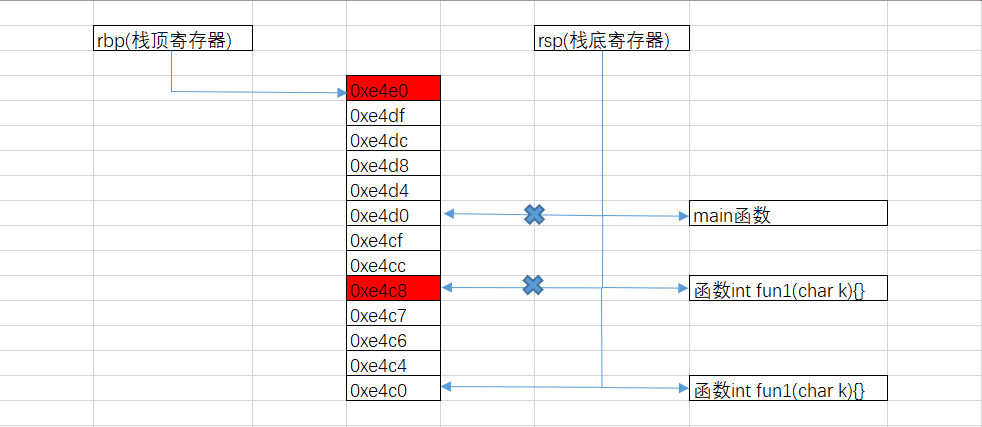
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，对应的汇编语句为（0x00000000004004ce <fun1+1>: 48 89 e5 mov %rsp,%rbp），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



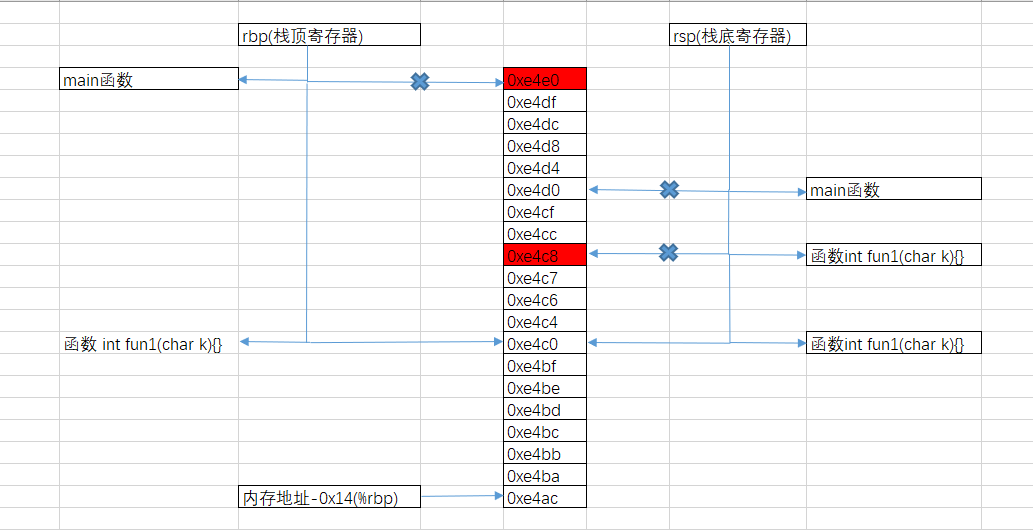
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，对应的汇编语句为（0x00000000004004d1 <fun1+4>: 89 f8 mov %edi,%eax），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



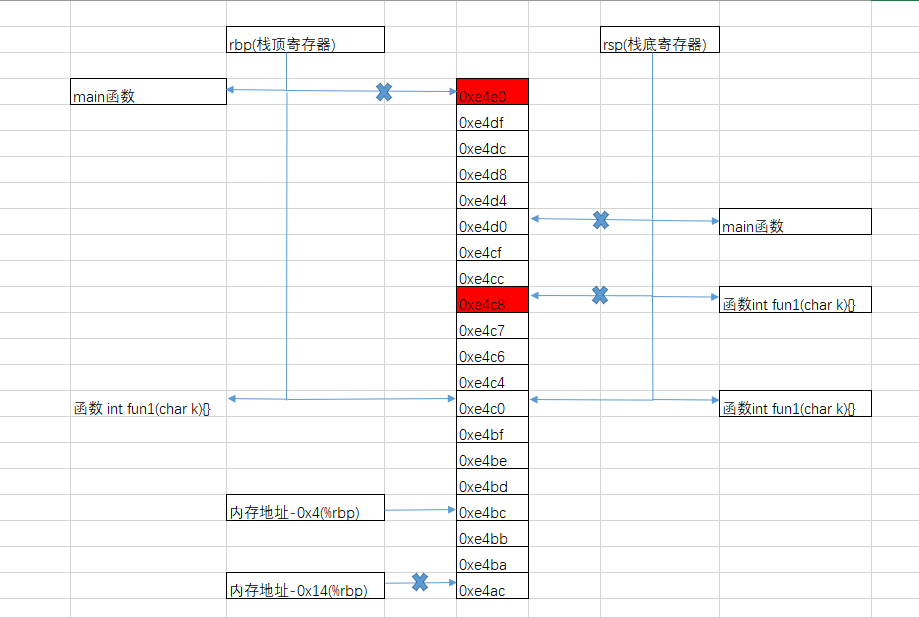
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，对应的汇编语句为（0x00000000004004d1 <fun1+4>: 89 f8 mov %edi,%eax），rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



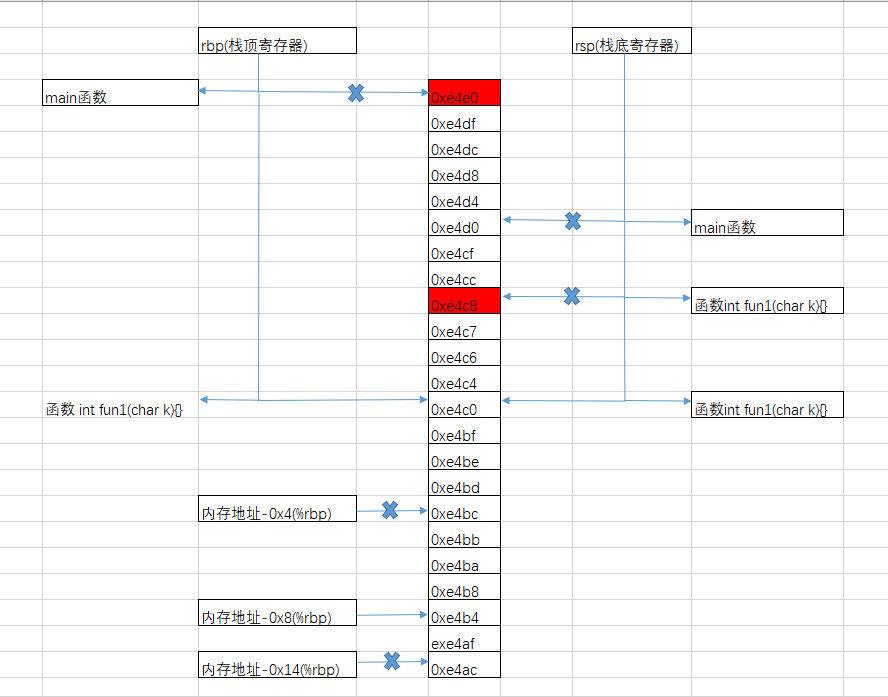
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，对应的汇编语句为（0x00000000004004d3 <fun1+6>: 88 45 ec mov %al,-0x14(%rbp)），rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



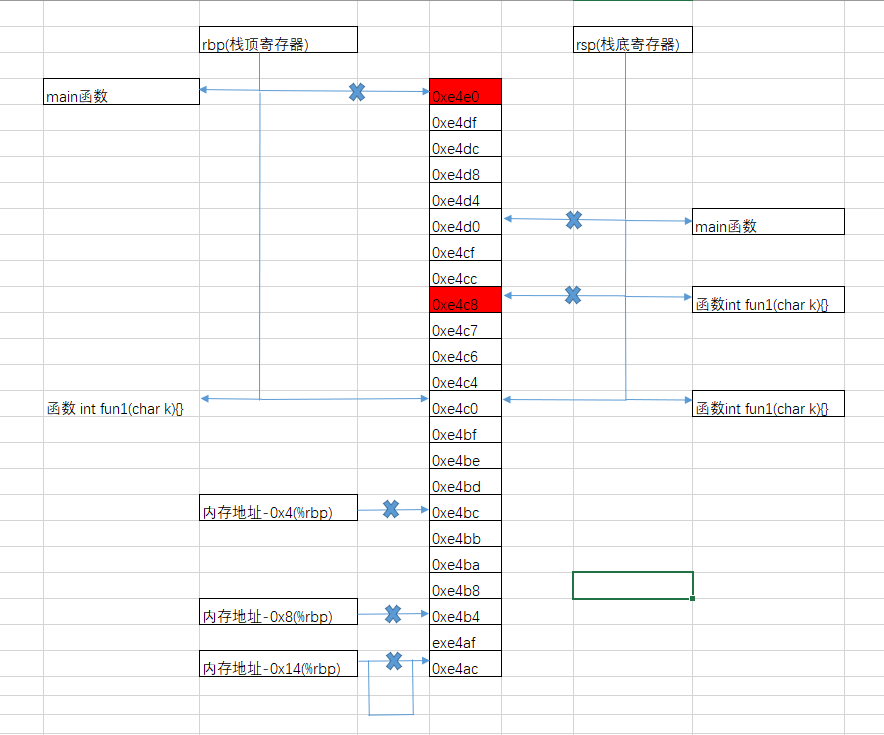
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行int i = 5;这条语句;对应的汇编语句为（0x00000000004004d6 <fun1+9>: c7 45 fc 05 00 00 00 movl $0x5,-0x4(%rbp)），rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



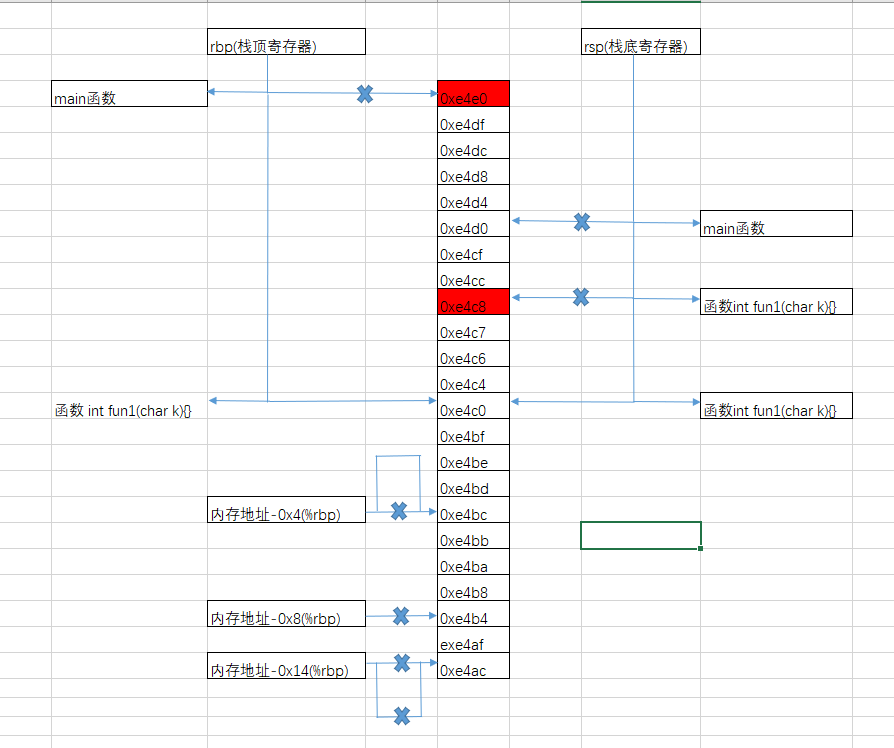
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行int j = 7;这条语句;对应的汇编语句为（0x00000000004004dd <fun1+16>: c7 45 f8 07 00 00 00 movl $0x7,-0x8(%rbp))，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



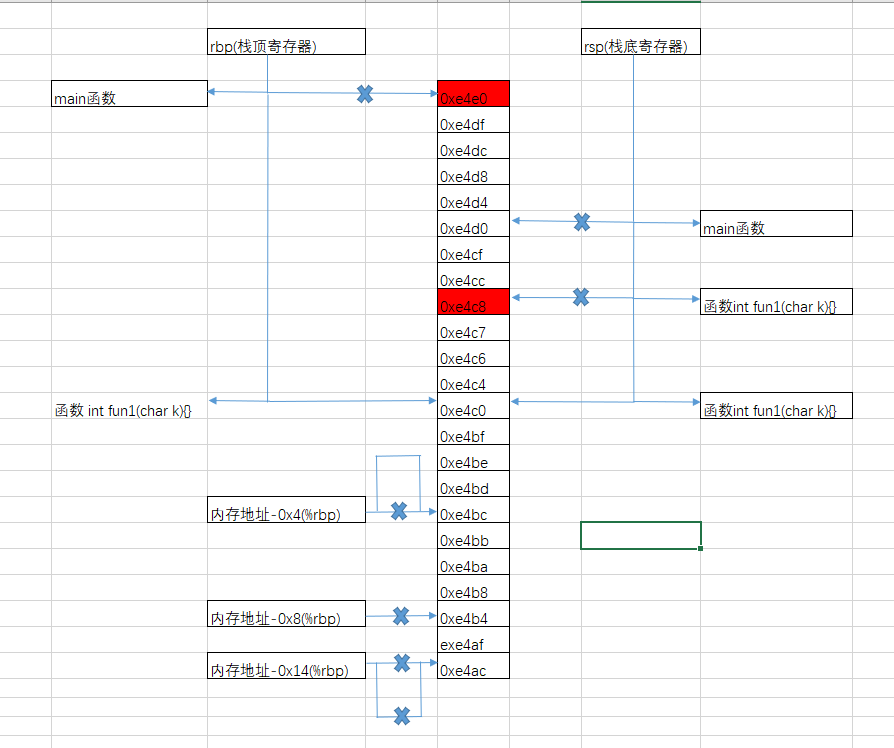
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004e4 <fun1+23>: 0f be 55 ec movsbl -0x14(%rbp),%edx)，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



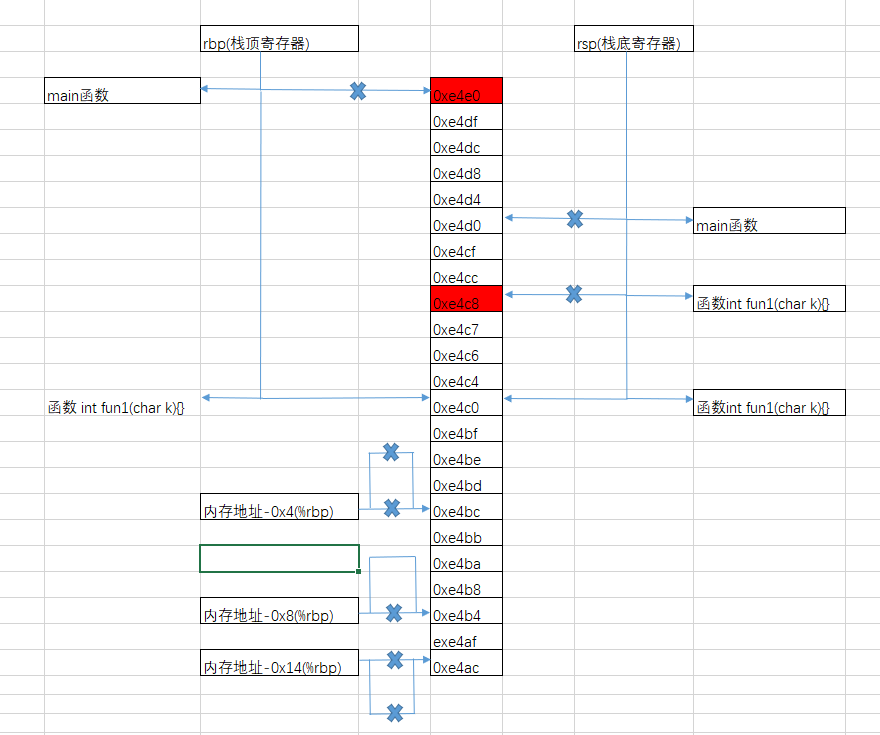
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004e8 <fun1+27>: 8b 45 fc mov -0x4(%rbp),%eax)，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



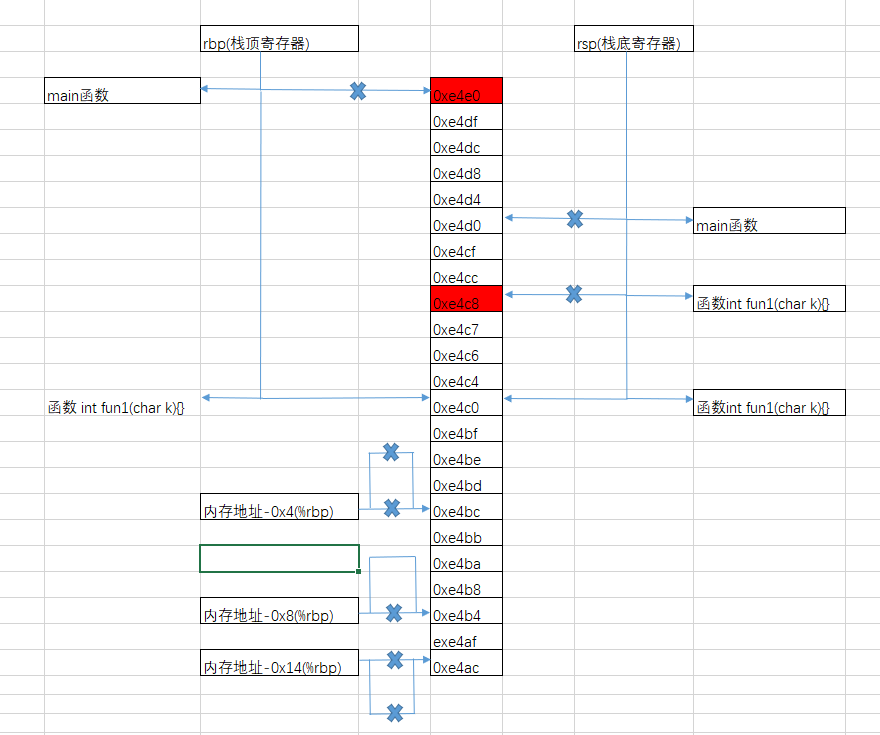
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004eb <fun1+30>: 01 c2 add %eax,%edx)，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



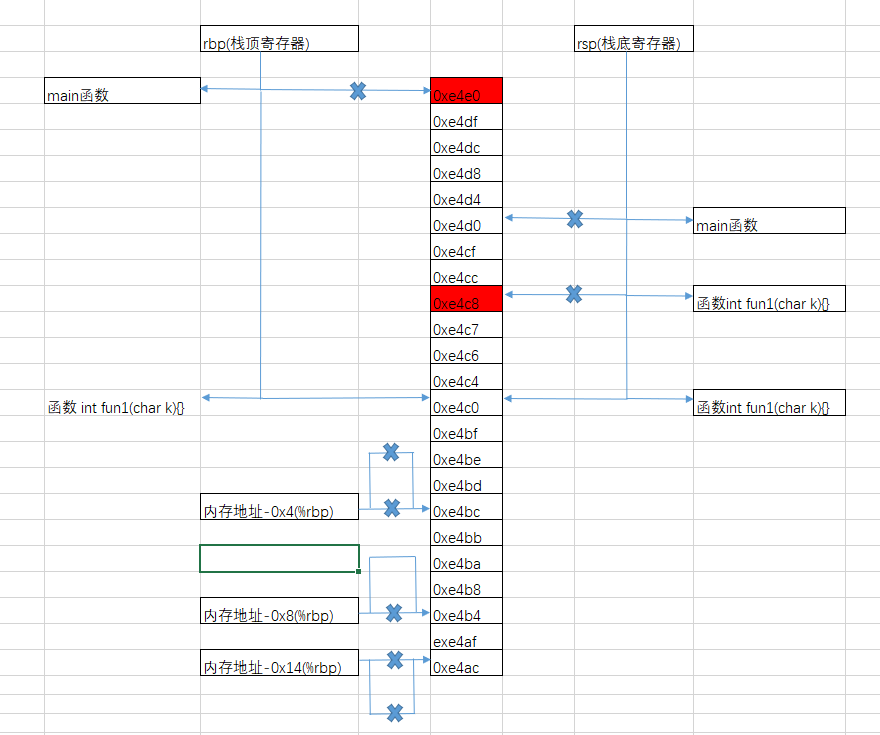
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004ed <fun1+32>: 8b 45 f8 mov -0x8(%rbp),%eax)，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



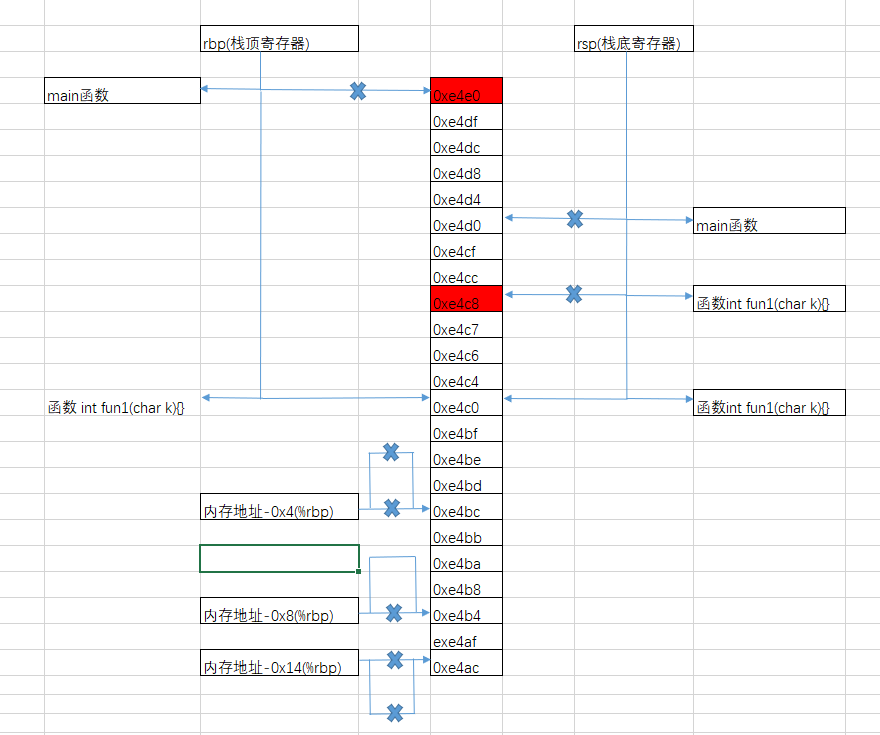
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004f0 <fun1+35>: 01 d0 add %edx,%eax)，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



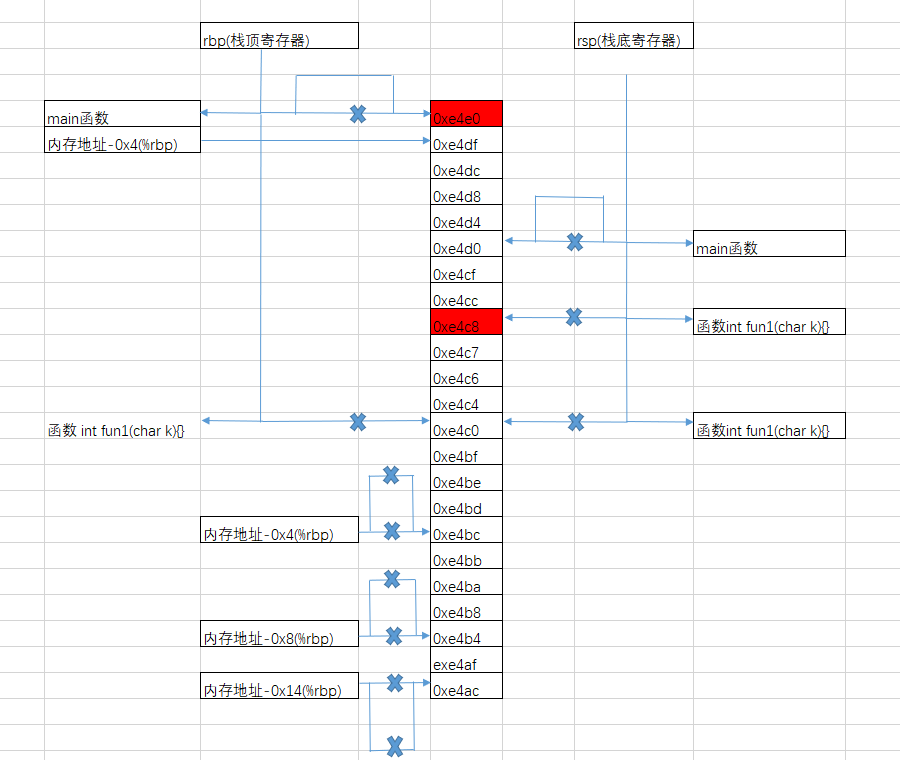
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004f2 <fun1+37>: 5d pop %rbp)，rbp（栈顶寄存器）的地址是：0x7fffffffe4c0，rsp（栈底寄存器）的地址是：0x7fffffffe4c0，为了方便表示，我继续以图片的形式展示：



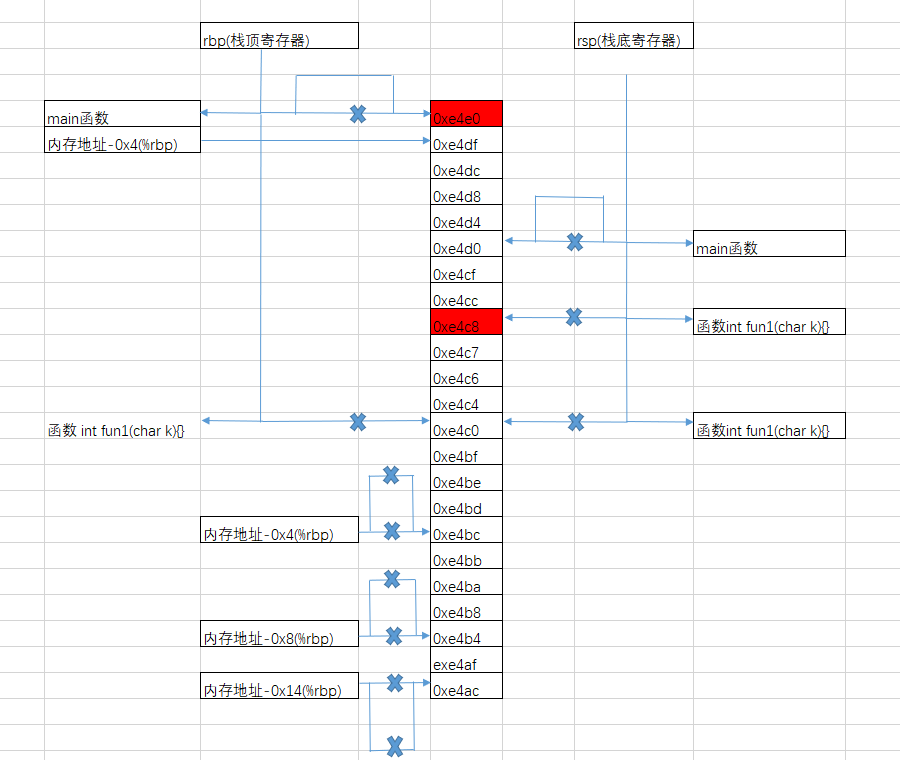
然后我们继续单步调试程序往下走(si命令)，程序还是在int fun1(char k){}这个子函数，运行return k+i+j;这条语句;对应的汇编语句为（0x00000000004004f3 <fun1+38>: c3 retq)，rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4c8，为了方便表示，我继续以图片的形式展示：



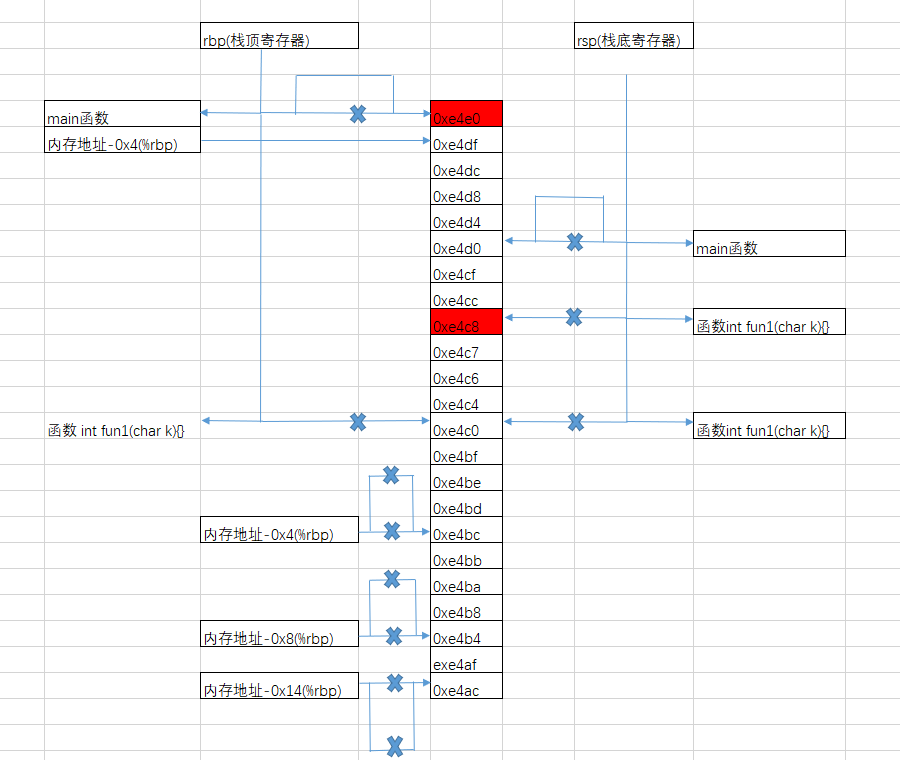
然后我们继续单步调试程序往下走(si命令)，对应的汇编语句为（0x000000000040050b <main+23>: 88 45 ff mov %al,-0x1(%rbp))，rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：



然后我们继续单步调试程序往下走(si命令)，程序运行return 0;这条语句；对应的汇编语句为（0x000000000040050e <main+26>: b8 00 00 00 00 mov $0x0,%eax)，rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：



然后我们继续单步调试程序往下走(si命令)，对应的汇编语句为（0x0000000000400513 <main+31>: c9 leaveq），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：



然后我们继续单步调试程序往下走(si命令)，程序结束；对应的汇编语句为（0x0000000000400514 <main+32>: c3 retq），rbp（栈顶寄存器）的地址是：0x7fffffffe4e0，rsp（栈底寄存器）的地址是：0x7fffffffe4d0，为了方便表示，我继续以图片的形式展示：

