

Week 4: Loops and Functions in Assembly

SOLUTION

February 13-15, 2023

Consider the following C functions and the assembly code they compile to:

```

<0x400147> foo:                                int foo(int * a, int b){
    0x400147<+0>:  xorq %rax, %rax                int x = 0;
    0x40014a<+3>:  movq %rsi, %rdx                int i;
    0x40014d<+6>:  subq $1, %rdx
    0x400151<+a>:  jmp  L3                          for(i = b-1; i > -1; i--){
L0:
    0x400153<+c>:  cmpq $47, %rdx                    if(i != 47){
    0x400157<+10>: je  L1
    0x400159<+12>: movl %eax, (%rdi,%rdx,4)          a[i] = x;
    0x40015c<+15>: jmp  L2                          } else {
L1:
    0x40015e<+17>: movl $47, (%rdi,%rdx,4)          a[i] = 47;
L2:
    0x400165<+1e>: leaq (%rdx,%rax),%rax
    0x400169<+22>: decq %rdx                        x += i;
L3:
    0x40016c<+25>: cmpq $0, %rdx
    0x400170<+29>: jge  L0                          return x;
    0x400172<+2b>: retq                               }

<0x400173> main:                             int main(int argc, char ** argv){
    0x400173<+0>:  subq $24, %rsp                    int a[4];
    0x400177<+4>:  movq $47, 16(%rsp)                int y = foo(a, 4);
    0x400180<+d>:  movq %rsp, %rdi                    }
    0x400183<+10>: movl $4, %esi
    0x400188<+15>: callq foo
    0x400194<+22>: xorl %eax, %eax
    0x400196<+24>: addq $24, %rsp
    0x40019a<+28>: retq

```

1. For each variable, indicate which register that variable is stored in.

- a: %rdi
- b: %rsi
- x: %rax

- i: %rdx

2. Based on the assembly code, fill in the blanks in the C source code. You may only use the C variable names a, b, x, i, not register names. Use array notation to show any accesses to elements of a, not pointer arithmetic.

Hint: `cmp a, b` sets the same condition codes as `b-a`

3. Below is a diagram of the stack at the beginning of function `main` (that is, immediately before the instruction `subq $24, %rsp` is executed). Modify this diagram to show the state of the stack immediately before the function `foo` returns (that is, immediately before the instruction `retq` is executed). Include in your diagram an arrow labeled `%rsp` that indicates the address stored in the register `%rsp` at that point and an arrow labeled `a` that indicates the address stored in the variable `a` at that point.

Hints: Remember that you are running on a 64-bit little-endian machine. Note that the addresses in this stack diagram are 4 bytes apart.

