CS 33: Introduction to Computer Organization

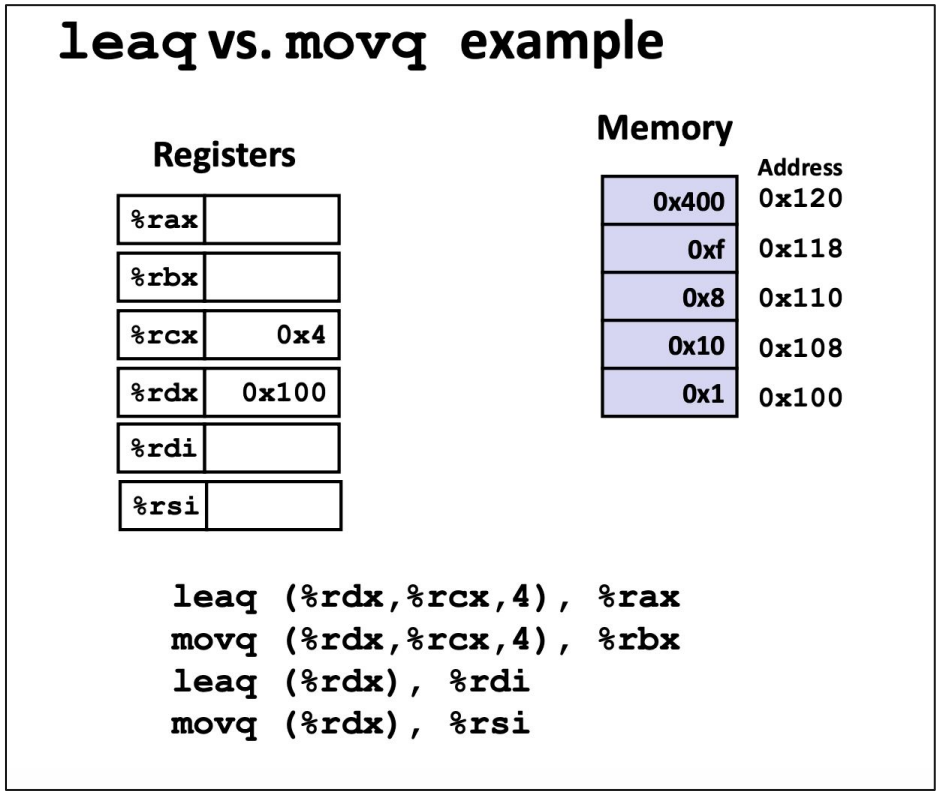
TA: Aalisha Dalal
LA: Jonathan Myong
Office Hours: Friday, 9:30-11:30AM

Focus of the Discussion

- Queries from previous discussion
- Stack Trace
- Arrays memory representation
- Worksheet Problems

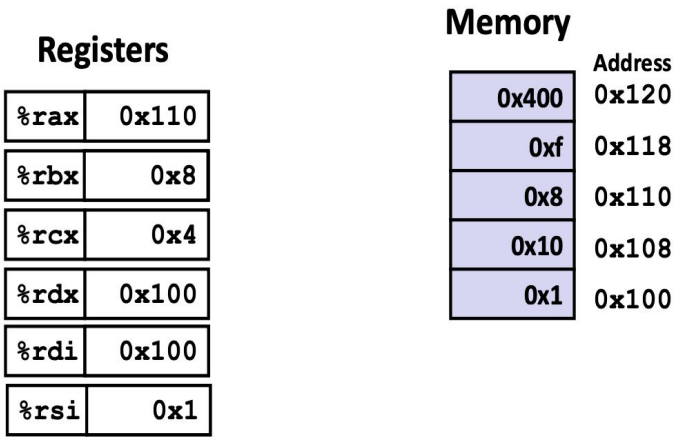
leaq vs. movq instruction

`leaq` vs. `movq` example



leaq vs. movq instruction

leaq vs. movq example (solution)




```
leaq (%rdx,%rcx,4), %rax
movq (%rdx,%rcx,4), %rbx
leaq (%rdx), %rdi
movq (%rdx), %rsi
```

CMP Instruction

```
(gdb) list
1       int max(int a, int b)
2       {
3         if(a<b)
4           return b;
5         else
6           return a;
7
8       }
9
10
(gdb) disassemble max
Dump of assembler code for function max:
   0x00000000004004d8 <+0>:     push   %rbp
   0x00000000004004d9 <+1>:     mov    %rsp,%rbp
   0x00000000004004dc <+4>:     mov    %edi,-0x4(%rbp)
   0x00000000004004df <+7>:     mov    %esi,-0x8(%rbp)
   0x00000000004004e2 <+10>:    mov    -0x4(%rbp),%eax
   0x00000000004004e5 <+13>:    cmp    -0x8(%rbp),%eax
   0x00000000004004e8 <+16>:    jge    0x4004ef <max+23>
   0x00000000004004ea <+18>:    mov    -0x8(%rbp),%eax
   0x00000000004004ed <+21>:    jmp    0x4004f2 <max+26>
   0x00000000004004ef <+23>:    mov    -0x4(%rbp),%eax
   0x00000000004004f2 <+26>:    pop    %rbp
   0x00000000004004f3 <+27>:    retq
End of assembler dump.
```


```
0x00000000004004e5 <+13>:    cmp    -0x8(%rbp),%eax
0x00000000004004e8 <+16>:    jge    0x4004ef <max+23>
0x00000000004004ea <+18>:    mov    -0x8(%rbp),%eax
0x00000000004004ed <+21>:    jmp    0x4004f2 <max+26>
0x00000000004004ef <+23>:    mov    -0x4(%rbp),%eax
```

CMP Instruction - II

```
(gdb) list
1       int max(int a, int b)
2       {
3         if(a>b)
4           return a;
5         else
6           return b;
7
8       }
9
10
(gdb) disassemble max
Dump of assembler code for function max:
   0x00000000004004d8 <+0>:     push   %rbp
   0x00000000004004d9 <+1>:     mov    %rsp,%rbp
   0x00000000004004dc <+4>:     mov    %edi,-0x4(%rbp)
   0x00000000004004df <+7>:     mov    %esi,-0x8(%rbp)
   0x00000000004004e2 <+10>:    mov    -0x4(%rbp),%eax
   0x00000000004004e5 <+13>:    cmp    -0x8(%rbp),%eax
   0x00000000004004e8 <+16>:    jle    0x4004ef <max+23>
   0x00000000004004ea <+18>:    mov    -0x4(%rbp),%eax
   0x00000000004004ed <+21>:    jmp    0x4004f2 <max+26>
   0x00000000004004ef <+23>:    mov    -0x8(%rbp),%eax
   0x00000000004004f2 <+26>:    pop    %rbp
   0x00000000004004f3 <+27>:    retq
End of assembler dump.
```

```
0x00000000004004e5 <+13>:    cmp    -0x8(%rbp),%eax
0x00000000004004e8 <+16>:    jle    0x4004ef <max+23>
0x00000000004004ea <+18>:    mov    -0x4(%rbp),%eax
0x00000000004004ed <+21>:    jmp    0x4004f2 <max+26>
0x00000000004004ef <+23>:    mov    -0x8(%rbp),%eax
```

Understanding Function Stack Trace

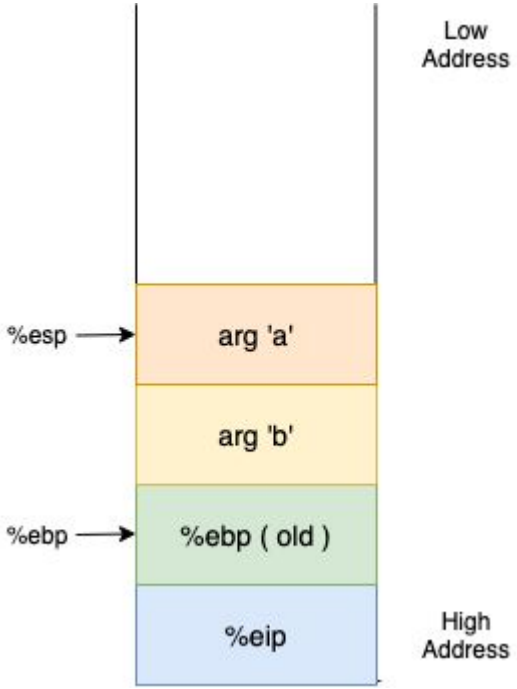
```
11      int main()
12      {
13
14      int a = 14;
15      int b = 17;
16
17      int c = max(a, b);
18
19      return 0;
20
```

```
(gdb) disassemble main
Dump of assembler code for function main:
   0x00000000004004f4 <+0>:     push   %rbp
   0x00000000004004f5 <+1>:     mov    %rsp,%rbp
   0x00000000004004f8 <+4>:     sub    $0x10,%rsp
   0x00000000004004fc <+8>:     movl   $0xe,-0x4(%rbp)
   0x0000000000400503 <+15>:    movl   $0x11,-0x8(%rbp)
   0x000000000040050a <+22>:    mov    -0x8(%rbp),%edx
   0x000000000040050d <+25>:    mov    -0x4(%rbp),%eax
   0x0000000000400510 <+28>:    mov    %edx,%esi
   0x0000000000400512 <+30>:    mov    %eax,%edi
   0x0000000000400514 <+32>:    callq  0x4004d8 <max>
   0x0000000000400519 <+37>:    mov    %eax,-0xc(%rbp)
   0x000000000040051c <+40>:    mov    $0x0,%eax
   0x0000000000400521 <+45>:    leaveq
   0x0000000000400522 <+46>:    retq
End of assembler dump.
```

Understanding Function Stack Trace

```
(gdb) list
1       int max(int a, int b)
2       {
3         if(a>b)
4           return a;
5         else
6           return b;
7
8       }
9
10
```

```
(gdb) disassemble max
Dump of assembler code for function max:
   0x00000000004004d8 <+0>:     push   %rbp
   0x00000000004004d9 <+1>:     mov    %rsp,%rbp
   0x00000000004004dc <+4>:     mov    %edi,-0x4(%rbp)
   0x00000000004004df <+7>:     mov    %esi,-0x8(%rbp)
   0x00000000004004e2 <+10>:    mov    -0x4(%rbp),%eax
   0x00000000004004e5 <+13>:    cmp    -0x8(%rbp),%eax
   0x00000000004004e8 <+16>:    jle    0x4004ef <max+23>
   0x00000000004004ea <+18>:    mov    -0x4(%rbp),%eax
   0x00000000004004ed <+21>:    jmp    0x4004f2 <max+26>
   0x00000000004004ef <+23>:    mov    -0x8(%rbp),%eax
   0x00000000004004f2 <+26>:    pop    %rbp
   0x00000000004004f3 <+27>:    retq
End of assembler dump.
```



Array Representation

$$A = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix}$$

How would we access this location from the starting memory address ?

Worksheet Problems

https://tinyurl.com/y6lgkjcj

Questions