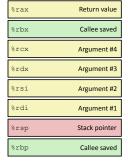
Lecture 16

x86-64 Assembly

CPSC 275
Introduction to Computer Systems

x86-64 Integer Registers %r8d %ebx %r9d %rbx %r9 %r10d %rcx %ecx %r10 %edx %r11d %rdx %r11 %rsi %esi %r12 %r12d %edi %r13d %rdi %r13 %esp %rsp %r14 %r14d %rbp %ebp %r15 %r15d - Twice the number of registers - Accessible as 8, 16, 32, 64 bits

Usage Conventions



%r8	Argument #5
%r9	Argument #6
%r10	Caller saved
%r11	Caller Saved
%r12	Callee saved
%r13	Callee saved
%r14	Callee saved
%r15	Callee saved

x86-64 Registers

- Arguments passed to functions via registers
 - If more than 6 integral parameters, then pass rest on stack
 - These registers can be used as caller-saved as well
- All references to stack frame via stack pointer
 - Eliminates need to update %ebp/%rbp
- Other Registers
 - 6 callee saved
 - 2 caller saved
 - 1 return value (also usable as caller saved)
 - 1 special (stack pointer)

x86-64 Long Swap

```
void swap_1(long *xp, long *yp)
{
  long t0 = *xp;
  long t1 = *yp;
  *xp = t1;
  *yp = t0;
}

swap:
movq (%rdi), %rdx
movq (%rsi), %rax
movq %rax, (%rdi)
movq %rdx, (%rsi)
ret
```

- Operands passed in registers
 - First (xp) in %rdi, second (yp) in %rsi
 - 64-bit pointers
- No stack operations required (except ret)
- Avoiding stack
 - Can hold all local information in registers