Operators

Young W. Lim

2017-01-31 Tue

Outline

Based on

"Self-service Linux: Mastering the Art of Problem Determination", Mark Wilding
"Computer Architecture: A Programmer's Perspective", Bryant & O'Hallaron

Integer Arithmetic Operations

1. Load Effective Address
2. Inc, Dec, Neg, Complement
3. Add, Sub, Mult, Xor, Or, And
4. Left shift, Arithmetic & Logical right shift

1) Load Effective Address (1)

- leal
- the same form as the instruction which reads from memory to a register
- the 1st operand looks like a memory reference
- no actual memory access
- just copy the location (effective address) to the destination
- used to generate pointers

```
leal S, D     ; &S -> D
```

1) Load Effective Address (2)

```
leal S, D     ; &S -> D
```

```
leal 7(%edx, %edx, 4), %eax
```

```
x is the value of %edx
x + 4*x + 7 ->  %eax
```

Unary & Binary Operators

```
incl (%esp)
subl %eax, %edx

addl %ecx, (%eax)
subl %edx, 4(%eax)
imull $16, (%eax, %edx, 4)
incl 8(%eax)
decl %ecx
subl %edx, %eax
```

Shift Operators

```
int shift_left2_rightn(int x, int n)
{
  x <<= 2;
  x >>= n;
  return x;
}

movl 12(%ebp), %ecx
movl 8(%ebp), %eax
```

Arithmetic Routine

```
int arith(int x, int y, int z)
{
  int t1 = x+y;
  int t2 = z*48;
  int t3 = t1 & 0xFFFF;
  int t4 = t2 * t3;

  return t4;
}
```

```
movl 12(%ebp), %eax
movl 16(%ebp), %edx
addl 8(%ebp), %eax
leal (%edx, %edx, 2), %edx
sall $4, %edx
andl $65536, %eax
imull %eax, %edx
movl %edx, %eax
```