

Operators

Young W. Lim

2017-01-31 Tue

1 Introduction

- References
- Arithmetic Operators
- Unary & Binary Operations
- Shift Operations

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

I, the copyright holder of this work, hereby publish it under the following licenses: GNU head Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled GNU Free Documentation License.

CC BY SA This file is licensed under the Creative Commons Attribution ShareAlike 3.0 Unported License. In short: you are free to share and make derivative works of the file under the conditions that you appropriately attribute it, and that you distribute it only under a license compatible with this one.

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

$x + 4*x + 7 \rightarrow \%eax$

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
```