C

CS 2130: Computer Systems and Organization 1

October 26, 2022

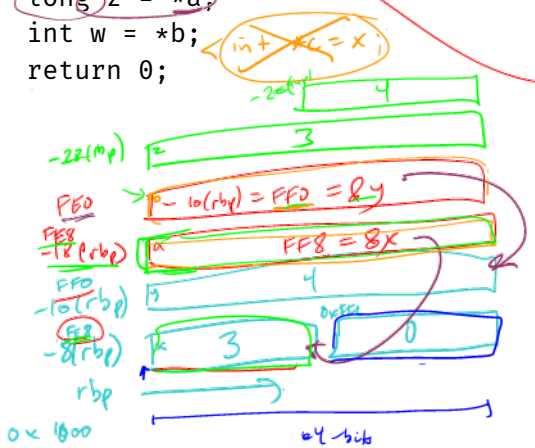
Announcements

- Homework 7 due Tuesday at 11pm
- Exam 2 next Friday

Example

```
int main() {
    int x = 3;
    long y = 4;
    int *a = &x;
    long *b = &y;
    long z = *a;
    int w = *b;
    return 0;
}
```

```
0000000000000000 <main>:
   0:   55                      push   %rbp
   1:   48 89 e5                mov    %rsp,%rbp
   4:   31 c0                   xor    %eax,%eax
   6:   c7 45 fc 00 00 00 00    movl   $0x0,-0x4(%rbp)
   d:   c7 45 f8 03 00 00 00    movl   $0x3,-0x8(%rbp)
  14:   48 c7 45 f0 04 00 00    movq   $0x4,-0x10(%rbp)
  1b:   00
  1c:   48 8d 4d f8             lea    -0x8(%rbp),%rcx
  20:   48 89 4d e8             mov    %rcx,-0x18(%rbp)
  24:   48 8d 4d f0             lea    -0x10(%rbp),%rcx
  28:   48 89 4d e0             mov    %rcx,-0x20(%rbp)
  2c:   48 8b 4d e8             mov    -0x18(%rbp),%rcx
  30:   48 63 09                movslq (%rcx),%rcx
  33:   48 89 4d d8             mov    %rcx,-0x28(%rbp)
  37:   48 8b 4d e0             mov    -0x20(%rbp),%rcx
  3b:   48 8b 09                mov    (%rcx),%rcx
  3e:   89 4d d4                mov    %ecx,-0x2c(%rbp)
  41:   5d                      pop    %rbp
  42:   c3                      retq
```



Example

Swap Example

```
void swap(int *a, int *b) {
    int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

printf ("%d %d\n", x, y)

Pointers

- All pointers are the same size: address size in underlying ISA
- Two special int types (defined using typedef)
  - `size_t` - integer the size of a pointer (unsigned)
  - `ssize_t` - integer the size of a pointer (signed)
  - With our compiler and ISA, these are both variants of `long`

Pointers and Arrays

`*x` and `x[0]` are equivalent

- Pointer to single value and pointer to first value in array
- Treat array as pointer to the first value (lowest address)
- Indexing into array: `x[n]` and `*(x+n)`
  - If `x` is an `int *`, then `x+1` points to **next int** in memory
  - Adding 1 to pointer adds `sizeof()` the type we're pointing to