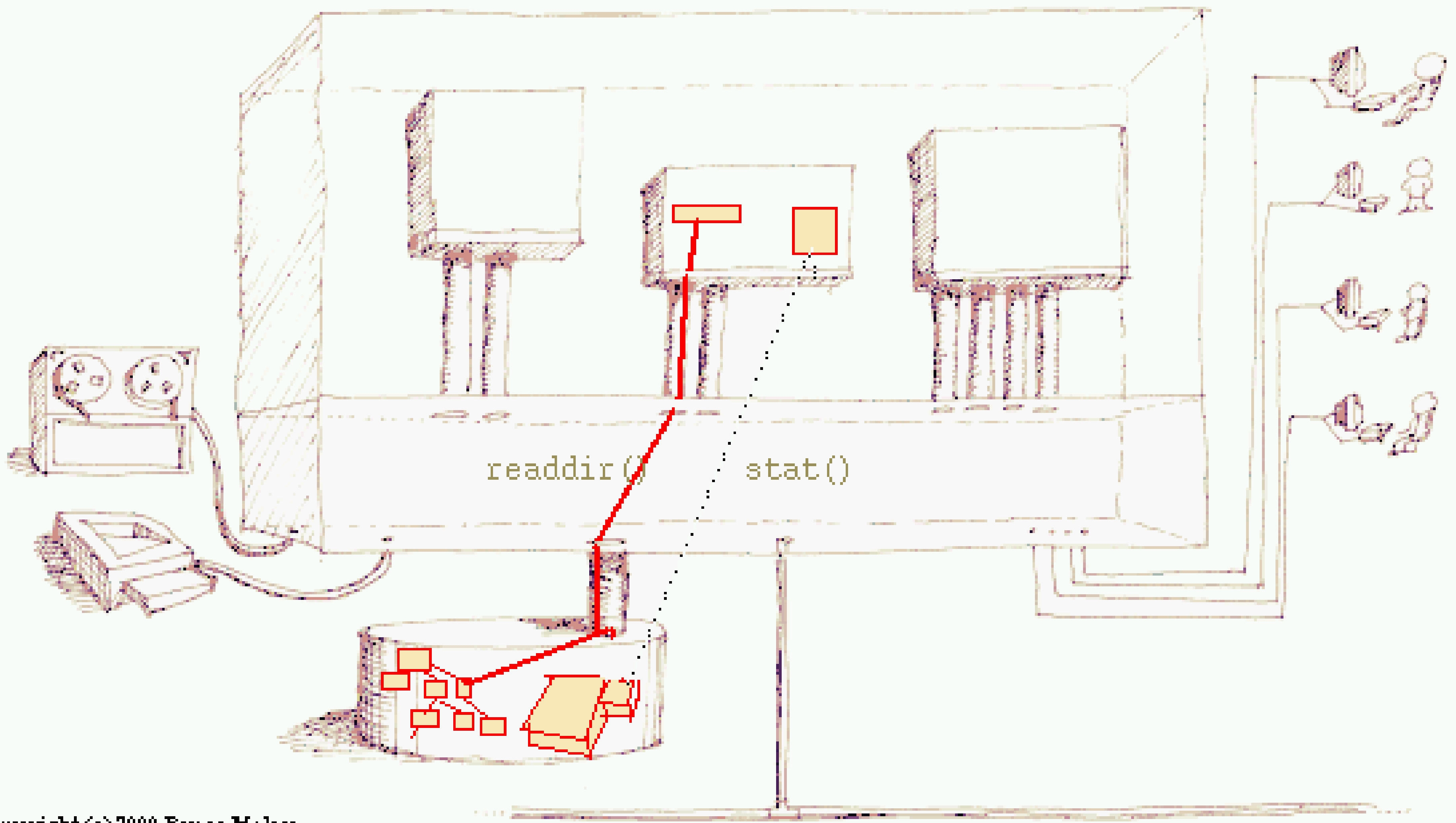
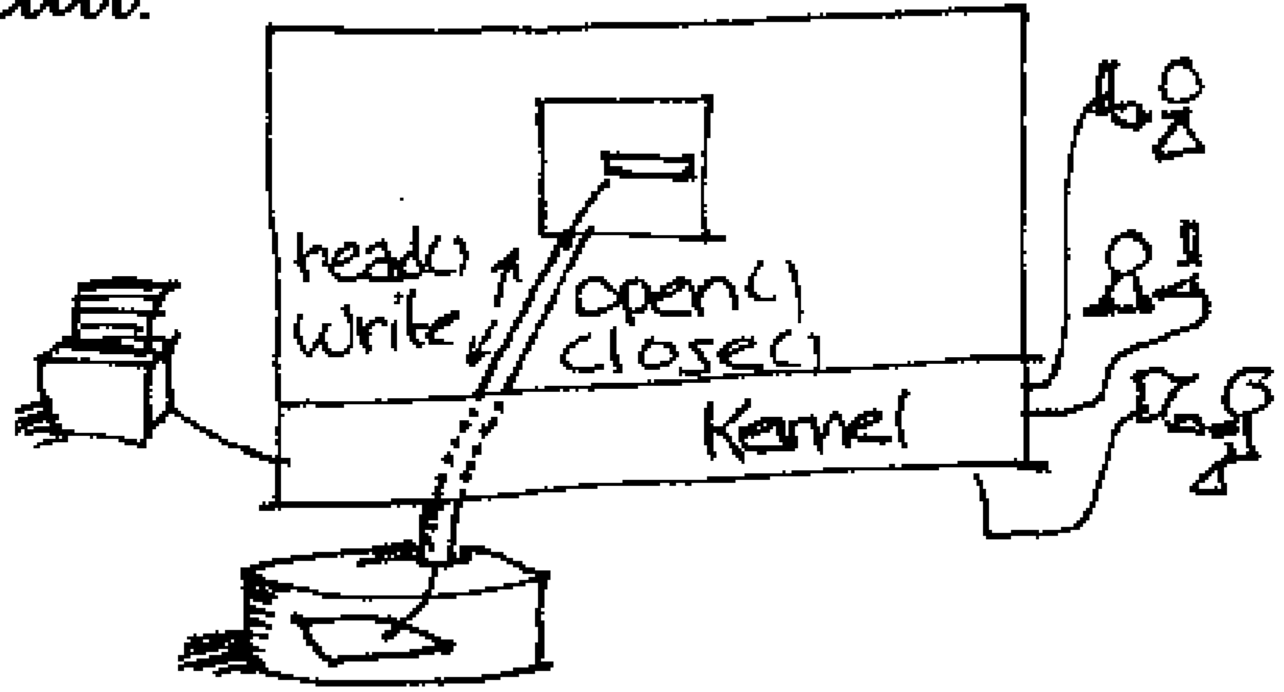
Class 3: **Directories, File Info, Bit Operations**



Class 3: Directories, File Info, Bit Ops

Recall:



Last time, we looked at ways to operate on the CONTENTS of a file:

open, read, write, lseek, close

But, there is more to a file than just contents:

1) **Properties**

   size, owner, permission , type, ...

2) **Location**

   in a directory tree

To learn about these:

We shall write a version of ls

Writing ls (with our 3-step method)

1) What does ls DO?

ls lists the contents of directories
AND displays information about files.

Examples

lists and shows info

dir

```
ls                  ls -l

ls /tmp             ls -l /etc

ls hello.c          ls -l hello.c
```

file

Other ls Options

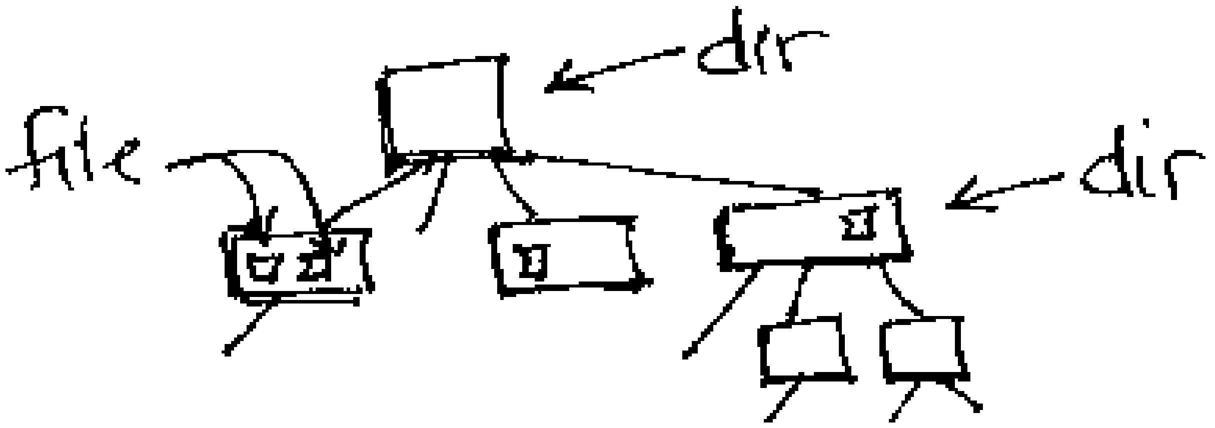
`ls -a`	shows "." files
`ls -a /`	
`ls -lu`	shows access time
`ls -s`	shows size in blocks

Note: Arguments to ls may be:

none	=>	show current directory
filenames	=>	show info about files
dirs	=>	show directory contents

3 Brief Review of the File System Tree

The disk is organized as a tree of directories, each of which contains files and/or directories.



The commands cd, pwd, ls allow you to explore a FILE SYSTEM.

4 So... writing ls should be easy:

```
     open directory             EOD?
 --> read entry      -----+
|                         
 --- display entry        |
     close  <-------------+
```

Maybe it is just like writing the who command.

Is a directory like a file?

5

Q. What IS A Directory?

A: A special kind of file that contains a list of files and or other directories.

Note: Every directory contains the items "." and ".."

6

Q: Can We Use open, read, close to Read This List?

A: Once, there was no other way

Let's Try:

```
cat / ; more /tmp;   od -c /etc
```

Ans2: *Even if you could, you dont want to, particularly since Unix supports a variety of different directory formats.*

For example, other versions of Unix and even disks from other operating systems

$\boxed{7}$ OK, How DO I Read a Directory ?

```
man -k direct
man -k direct  | grep read
```

Ans: *opendir(), readdir(), closedir().*

Using these Directory Functions

`#include <dirent.h>`

`opendir(name)`

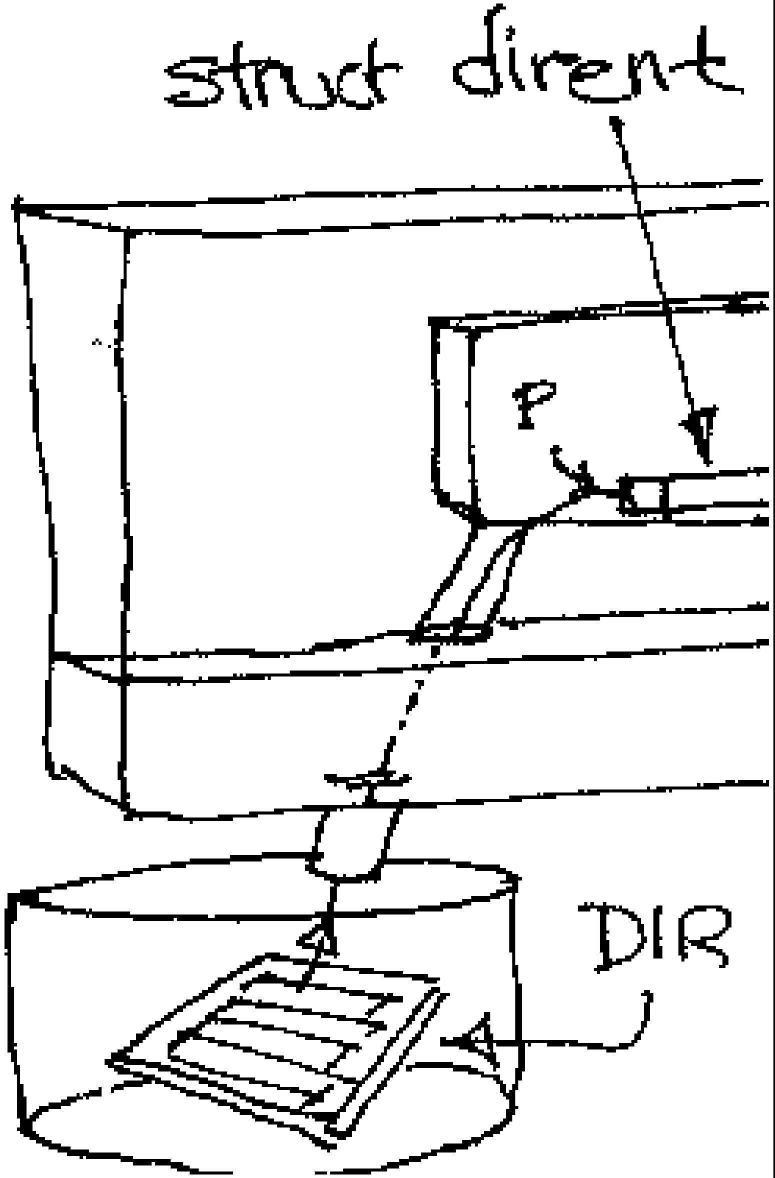
Creates a connection.
Returns a DIR *

`readdir(DIR *)`

Gets next record from dir. Returns a struct dirent *

`closedir(DIR *)`

Closes the connection.



Writing ls1.c

```
main()
    opendir
    while ( readdir )
       print d_name

    closedir
    return 0
```

Let's compile and run the code...

How Well Does ls1.c Work?

a) No columns

solution: doable, interesting

b) Not sorted

solution: use qsort() on array

c) No -a option (lists all files)

solution: simple string handling

d) No -l option

solution: .... hmmmm.

9 How Do We Add the -l Option ?

Note: The other info is NOt in the directory

Q: What Does -l Display?

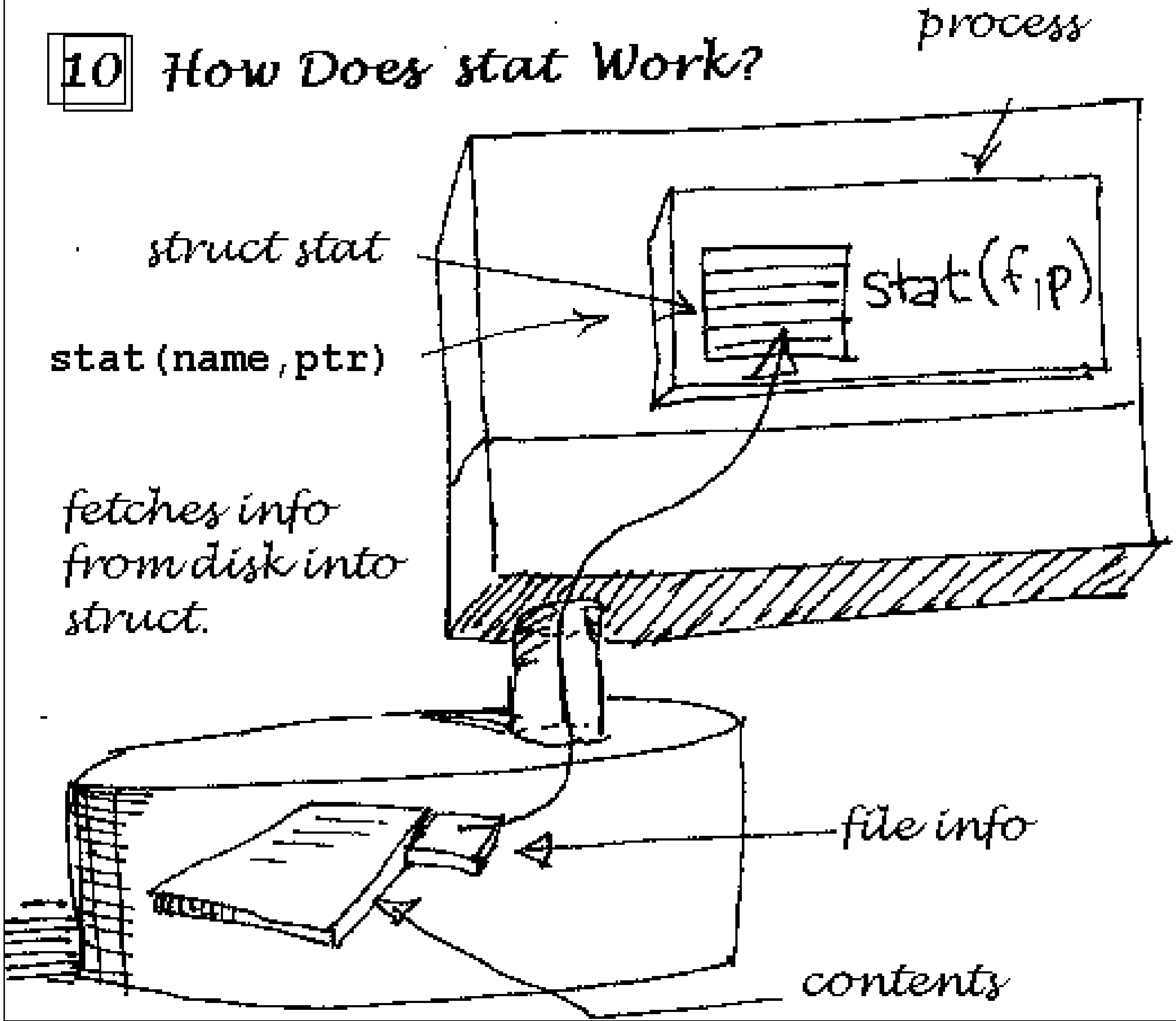
A: modtime, size, owner, group, links, type, permission,...

Q: Where do we learn about reading these properties?

A: Search the manual

Result: the stat() system call

10 How Does stat Work?



What Do We Get From stat ?

The struct contains:

st_mode,
st_uid,
st_gid,
st_size,
st_links,
st_mtime, ...

So.. We Just:

1) Read a dirent for the filename,
2) Use stat to get file info for that name
3) Display the items in the struct

(code: stat1.c)

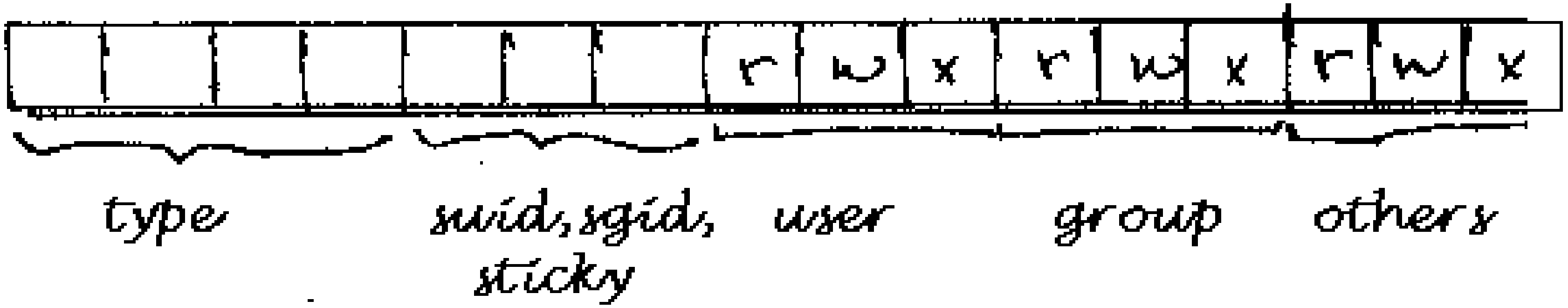
How'd We Do with stat1.c?

filename?	Perfect!
filesize?	Perfect!
mod time:	Use ctime() to convert
owner, group:	These are stored as numbers. We need to map numbers to names.
type?	??
permission?	??

in the mode

Where Are Type and Permissions Stored?

A: st_mode is a 16-bit value. Fields are encoded in sections of this value:



Four bits means 16 patterns:

Each file type has a pattern.

Each permission is on or off, so a single 1 or 0 will do.

Programming for File Types and File Permissions -- Subfield Coding

* Subfield coding is not magic:

617-222-3333	phone
011-22-4567	social security #
102.102.34.34	IP number

* A computer stores integers as a string of bits:

$$215 = \boxed{0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1}$$

$$? = \boxed{0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0}$$

* As with telephone numbers, people define the size and coding in each region.

Programming for File Types and File Permissions -- Masking to Read Fields

Q: How do I examine a bit or subfield?

A: Use "bitwise AND: &" to MASK the rest

Ex:

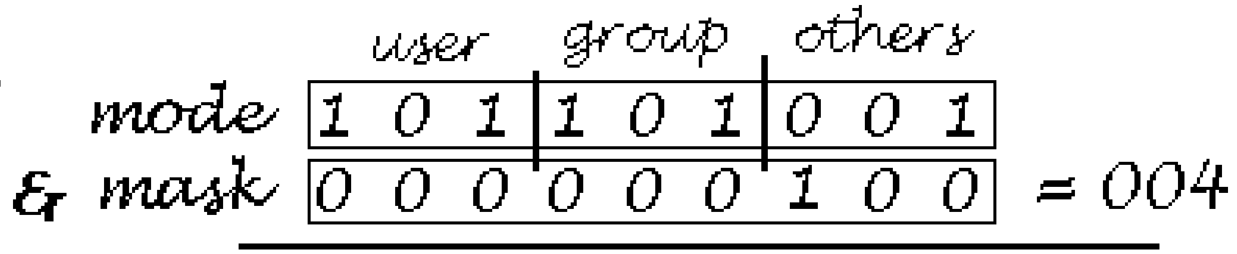
$$\begin{array}{rl} & \boxed{1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1} \qquad \text{value}\\ \& & \boxed{0\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0} \qquad \text{mask}\\ \hline & \boxed{0\ \ 0\ \ 0\ \ \underbrace{1\ \ 0\ \ 1}\ \ 0\ \ 0\ \ 0} \qquad \text{result}\end{array}$$

* The & operator compares values bit by bit. Its result contains 1's only where both numbers have a 1.

* 1's in the mask allow a value to `show through'

Testing Permission Bits with Masks:

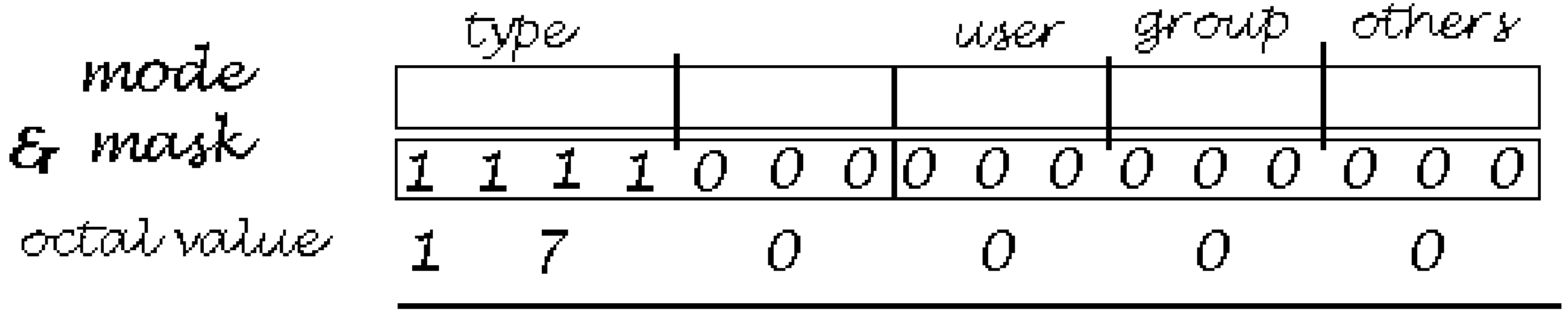
Picture:



Code:

```
if ( st_mode & 004 )
     printf("readable by others");
if ( st_mode & 002 )
     printf("writable by others");
```

Testing File Type with Masks



* Masks are in <sys/stat.h>

[14] How Do We Convert UID to Name?

We check the manual and learn of:
/etc/passwd,
and ypcat passwd

The getpwuid() function provides
access to user information.

The function returns a pointer to a
struct that contains information
about the user, just as struct stat
contains information for a file.

[15] How Do We Convert GID to Name?

Works very much like getpwuid(),
just a different file and function.

[16] stat2.c Now Has Correct Format

type, permissions, links, owner, group, size, mod time, name!

[17] Writing ls2.c

Combine ls1.c to get names with stat2.c to get info.