# Midterm Review

# Where have we been

ECE 373

# Prelims

- Questions on lab, homework or class?

  - Memory types

  - PCI connections

  - Debugging tricks

  - Delays and timers

# Types of OS's in the wild

- Single-user (Phones, PC's)

- Multi-user (Servers, mainframes, "cloud")

- Real-time (Stop lights, shuttle navigation)

- Embedded (Watch, routers, car engine, mp3)

# Device drivers

- The software that controls specific pieces of hardware

- API in the OS allows common interfaces for drivers

- Driver takes common commands from the OS and translates into hardware-specific stuff

- Many types of device drivers

# Types of drivers



- Three main classes of drivers

# Types of drivers

- Three main classes of drivers

  - Block drivers

  - Char drivers

  - Network drivers

# Types of drivers



- Three main classes of drivers

  - Block drivers

  - Char drivers

  - Network drivers

- Driver class can apply to many types of devices

  - USB device can be char, block, or network

  - Interface cards (PCIe) can be network or block

  - Graphics typically char devices

# Basic module concept

# Basic module concept

- Dynamic loading (insmod/modprobe, rmmod)

- Registering init and exit

- Kernel timeline of module diagram
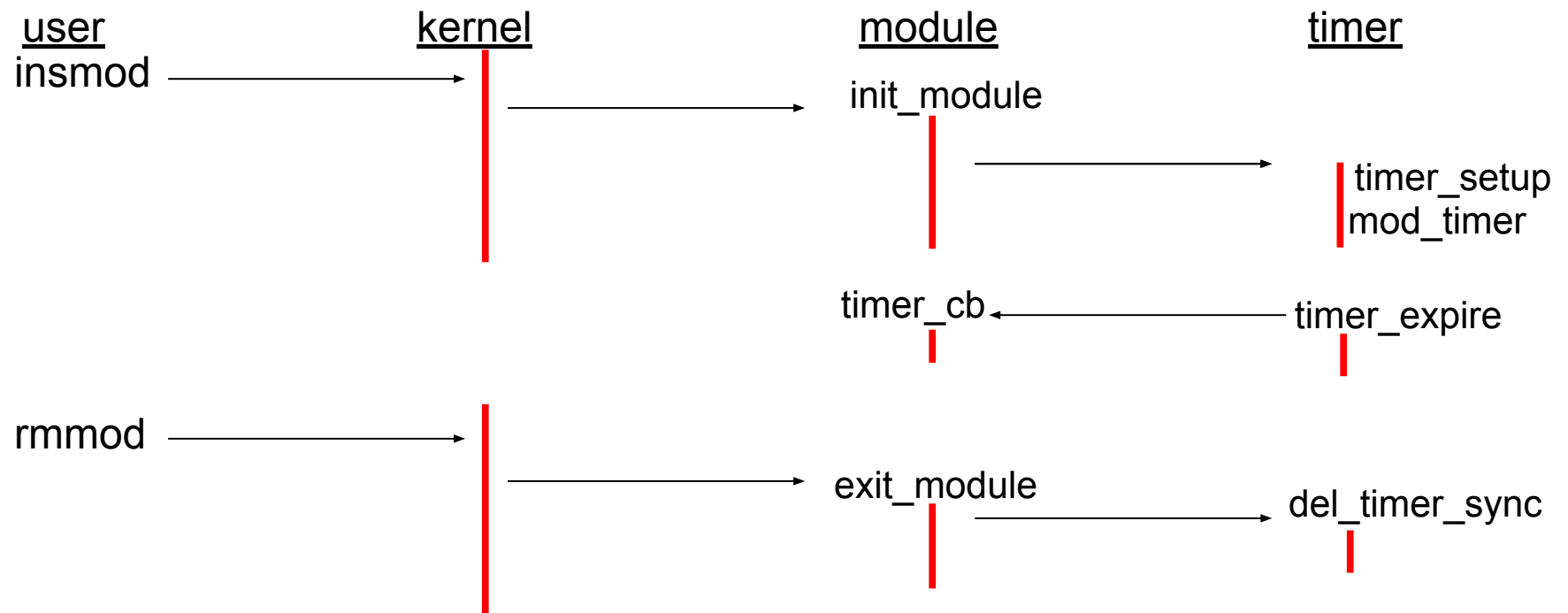
# Bits and pieces

- Minimal callback hooks and Compile headers
  - Basic #includes
  - MODULE_LICENSE(lic) – legal strings
  - __init, __exit
  - module_init(func), module_exit(func)
- Time values
  - Jiffies, HZ

# Module parameters

# Module parameters

- Gets info into the module

- `module_param(var, type, 0);`

- `/sys/module/<drivername>/parameters/<var>`

- Kernel community typically frowns on them

- `insmod <modulename> xx=11`

- `modinfo <modulename>`

# Talking to the Kernel

# Talking to the Kernel

- ioctl()
  - Older method of system call
  - Usually frowned up now-a-days (why?)

- Netlink
  - Newer, more flexible
  - Usually used with networking tools

- Pseudo filesystems
  - /proc – lots of kernel data
  - /sys – mostly module information
  - /dev – file-based connectors for read/write access
  - debugfs – useful for exposing debug hooks

# Character drivers

# Character drivers

- Standard callbacks

  - Open, read, write, release

- Coordinate with OS

  - alloc_chrdev_region()

  - cdev

  - struct file_operations

  - Linkage into /dev
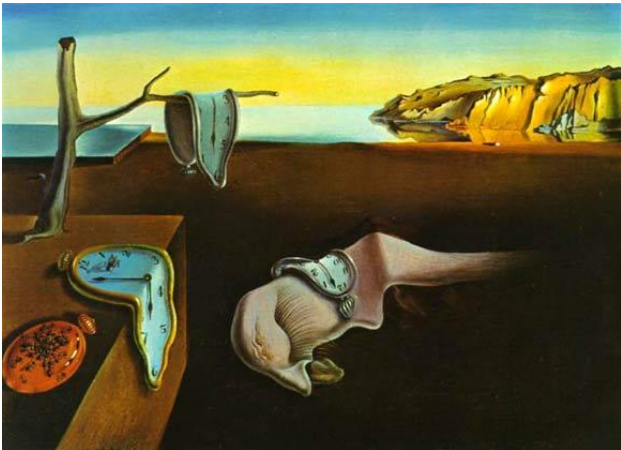
# I/O Ports



- 

- 

- 

- 

    - 

    -

# I/O Ports

- Working with slow devices

- Different "address" space from system memory

- System design coordinates device address mapping

- PC architecture evolution

  - combined many little devices into fewer large devices

  - Adds complexity to use of devices
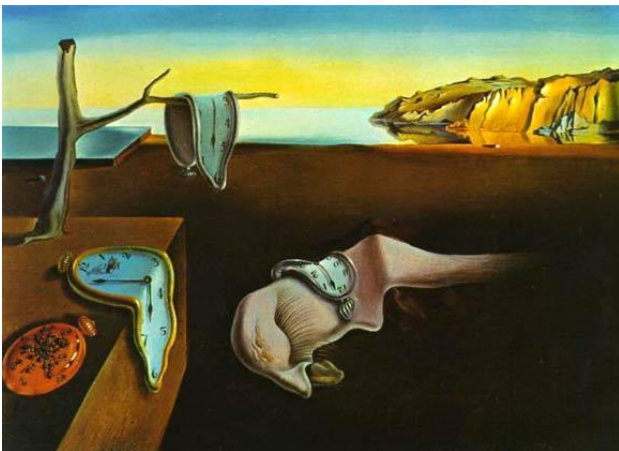
# Memory Map Types

# Memory Types

- Physical addresses – actual memory address

- Virtual addresses – relocatable, flexible, swappable

- Kernel logical – maps directly to physical, used for DMA/sharing with devices
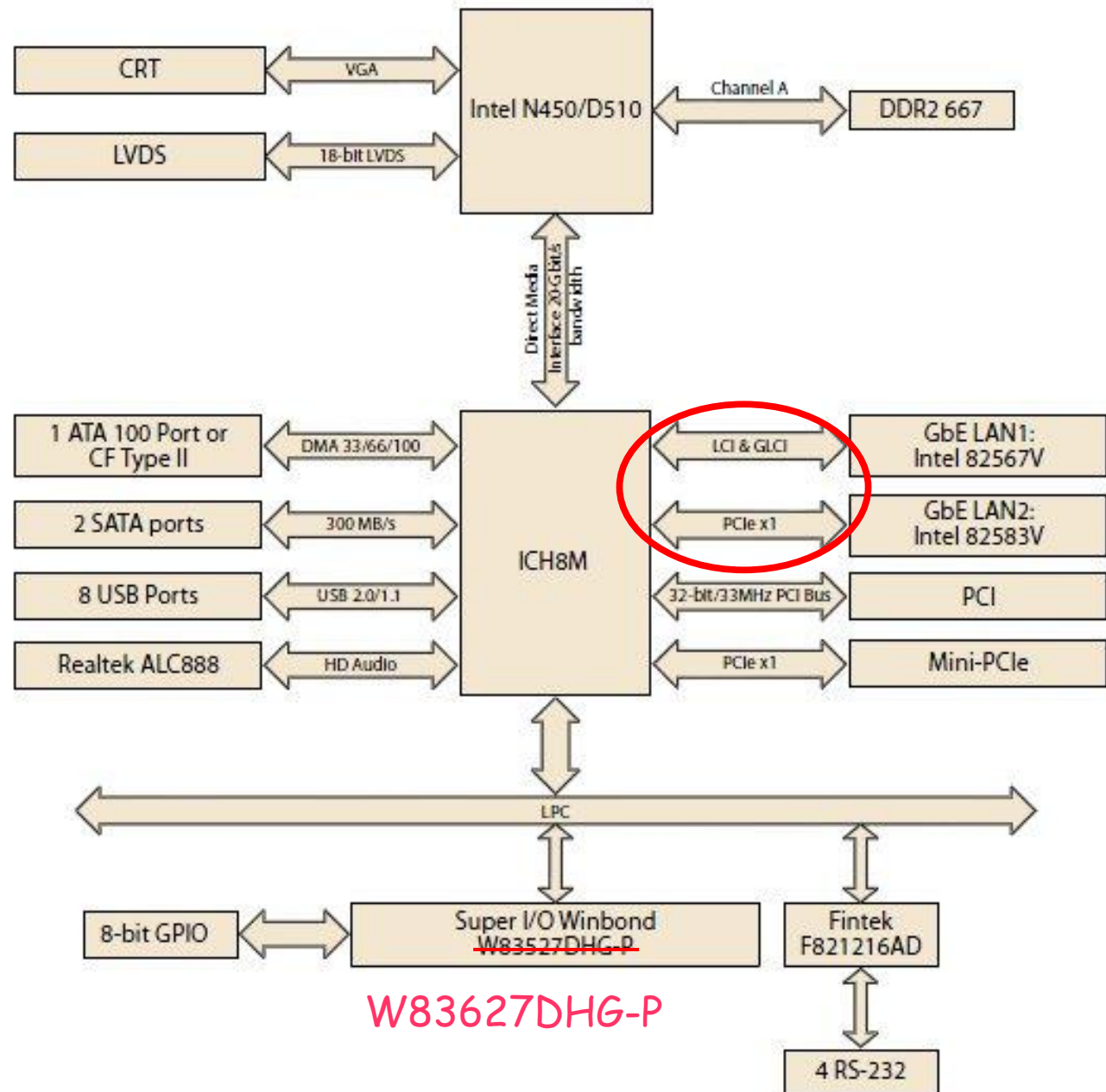
- Kernel virtual – relocatable kernel memory
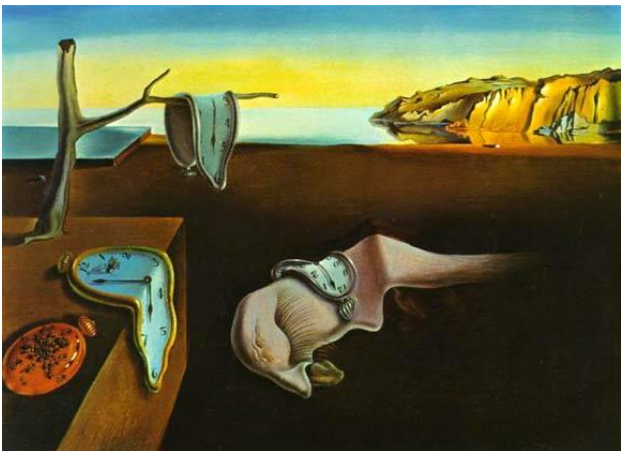
# Memory Types

- vmalloc() - allocates kernel virtual memory

- kmalloc() - allocates kernel logical memory
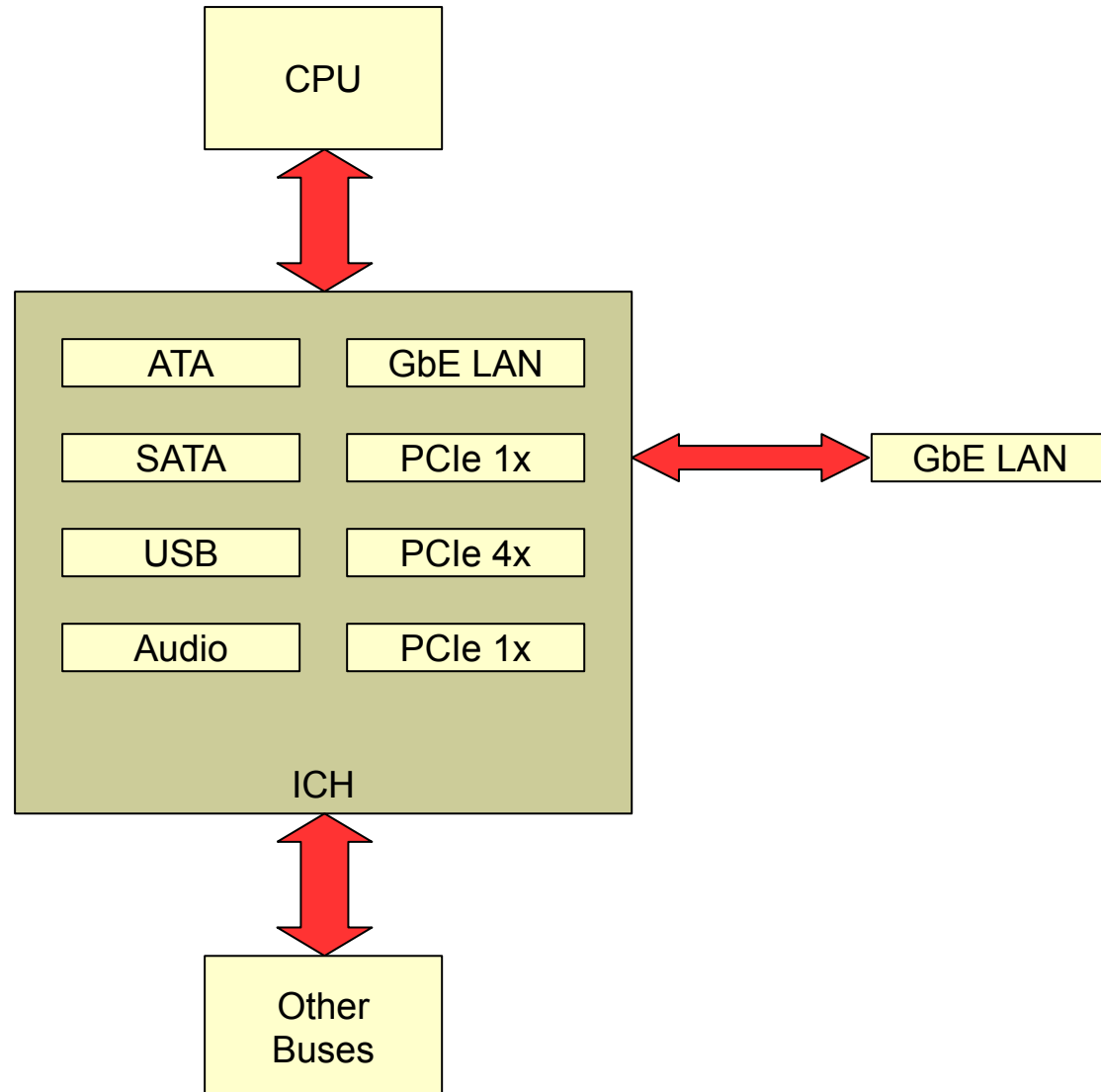
# Current PCs

- CPU
- Chipset
- Super I/O

# ICH Reality

# PCI

# PCI



- Communication/connection method for devices
- Devices use both port mapped and memory mapped I/O (focus on MMIO only for midterm)
- BAR – base address register
  - Starting address for device memory map
  - Driver uses `iowrite32()` and `ioread32()` to access device registers (or `writel()`/`readl()`)
- Callbacks
  - Probe, remove, sleep, resume

# Debugging

- Find a way to reproduce the bug

  - Gather info for tracking the problem

  - Prove it is fixed

# Debugging

- Find a way to reproduce the bug

  - Gather info for tracking the problem

  - Prove it is fixed

- Printk

  - KERN_INFO, etc.  Where do these go?

  - WARN(), BUG()

# Debugging

- Find a way to reproduce the bug

    - Gather info for tracking the problem

    - Prove it is fixed

- Printk

    - KERN_INFO, etc.  Where do these go?

    - WARN(), BUG()

- Stack trace

    - Gives idea of where things broke

    - Objdump helps decode stack info

# Debugging

- Find a way to reproduce the bug

  - Gather info for tracking the problem

  - Prove it is fixed

- Printk

  - KERN_INFO, etc.  Where do these go?

  - WARN(), BUG()

- Stack trace

  - Gives idea of where things broke

  - Objdump helps decode stack info

- /proc – various kernel data for monitoring

- Kgdb – useful to break into kernel

- Ethtool – monitor network devices

- Bus trace tools

# Delays and time keeping

- How jiffies relate to time
  - HZ

- Delays vs. Sleeping

- Pre-emption and its impacts

- `get_cycles()` and why use it

# Time for Timer

- ## Basics

  - timer_setup(t_var, t_callback, t_flags)

  - mod_timer(t_var, interval)

  - del_timer_sync(t_var)

# Time for Timer

- **Basics**

  - `timer_setup(t_var, t_callback, t_flags)`

  - `mod_timer(t_var, interval)`

  - `del_timer_sync(t_var)`

- **Callback function**

  - Called when timer expires

  - Given pointer to `t_var` as an argument

  - `timer_cb(struct timer_list *t)`

  - Use `from_timer()` to get back to private driver struct

# Getting information

- Device spec sheets

- The web (LXR, etc.)

- Books

- Kernel and driver source code