Operating Systems 18/19
Task 3: RTC Driver

Tom Spink

tspink@inf.ed.ac.uk
IF-1.47

Overview of Task 2

Task 2 was to create a physical page allocator, based on the buddy algorithm.

Any questions?

Overview

Task 3 is to implement a device driver for a

Real Time Clock

Deadline

The **STRICT** deadline is: **Week 10 28/03/2019**

Thursday at **4pm GMT**

Specification Document

Available here:

https://www.inf.ed.ac.uk/teaching/courses/os/

What is a device?

A device is something that puts information in, and/or gets information out of the computer.

Devices are normally real entities that are physically attached to the computer.

Devices can be:

- Input devices, e.g. mouse, keyboard, microphone, joystick, real time clock.
- Output devices, e.g. monitor, printer, headphones.
- Input/Output devices, e.g. hard-disk, USB stick.

What is a device driver?

A device driver is an operating system program, or a set of routines, that manage the lifetime and operation of a particular device.

They provide a software abstraction, of a hardware device.

Device drivers are usually vendor-specific, meaning they know how to control exactly one type of device, made by a specific manufacturer.

Often, they implement a generic interface to that device, so that the operating system doesn’t have to account for differences between vendors.

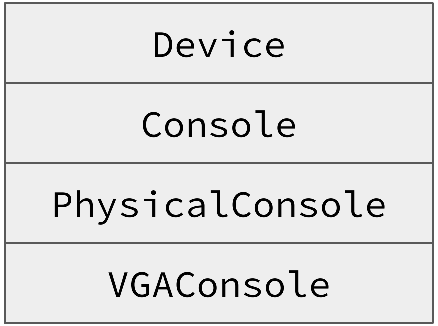
InfOS Device Model

Every attached device is represented by an instance of a subclassed “`Device`” object.

The subclass implements the functionality of the device driver.

Devices may depend on other devices.

There are normally multiple levels in the inheritance hierarchy. More derived classes implement more specific functionality.

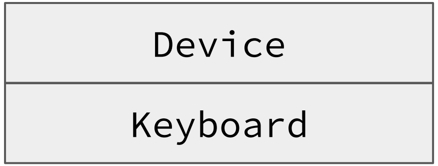


Device

Console

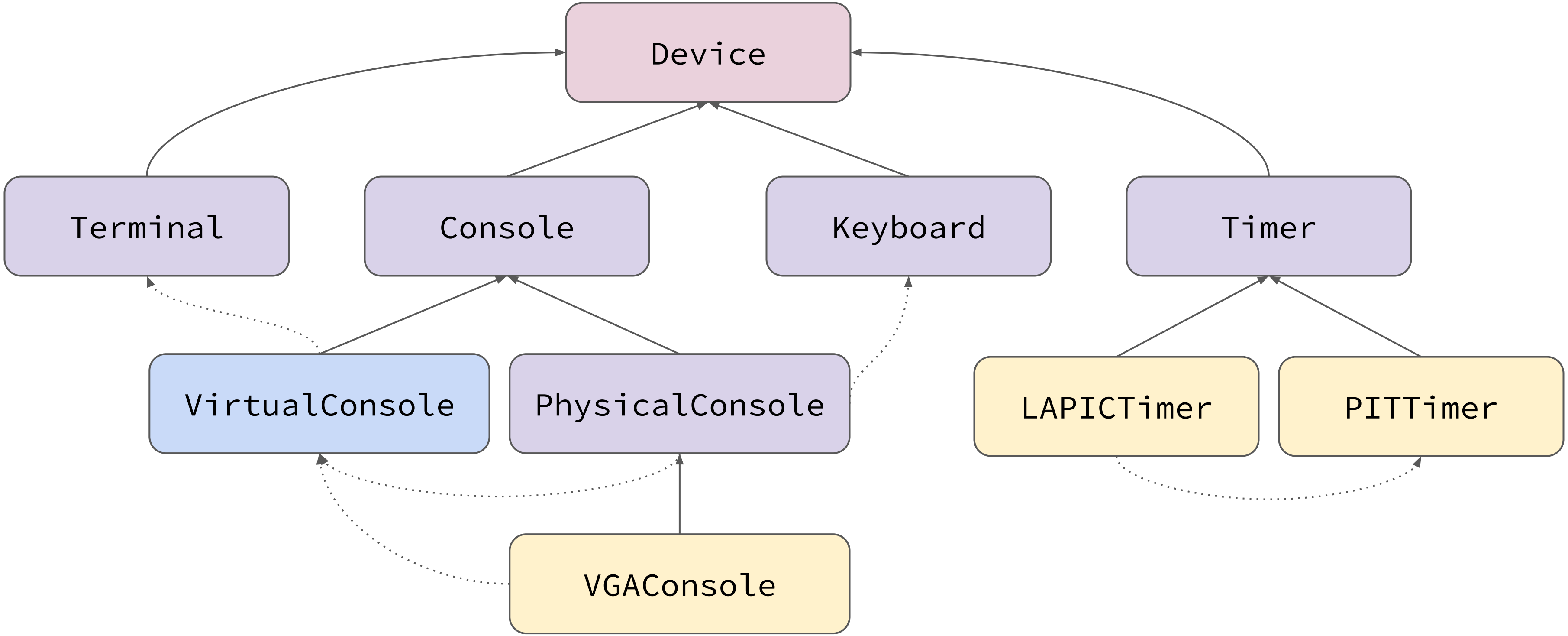
PhysicalConsole

VGAConsole



Device

Keyboard



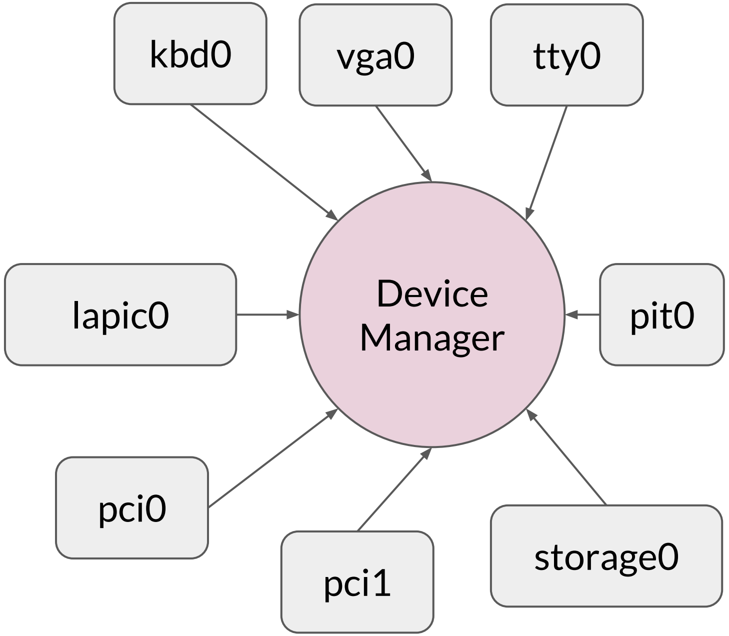
Subset of device inheritance hierarchy/interactions

Registering and Resolving Devices

When the system boots, the platform is probed, and discovered devices are instantiated, then registered with the "Device Manager".

When a component of the system requires a particular device (or class of device) the "Device Manager" is asked to return it.

When a device depends on another device, the dependee must have been initialised first!



Observing the Device Tree

Every device is assigned a unique name by the device manager.

These names can be listed in the InfOS shell, by issuing the command:

```
> /usr/ls /dev
Directory Listing of ‘/dev’:
  ioapic0 (0 bytes)
  vc0 (0 bytes)
  pci3 (0 bytes)
  ...
```

InfOS Keyboard Device Driver

A keyboard is a very straightforward input device.

The generic InfOS keyboard interface requires a “sink” to be registered.

When a key is pressed, the keyboard device tells the “sink” that a key is “down”. When the key is released, the “sink” is told the key is “up”.

A “sink” is anything that is interested in listening for key presses - such as the “physical console”. The physical console registers itself with the keyboard as a “sink”.

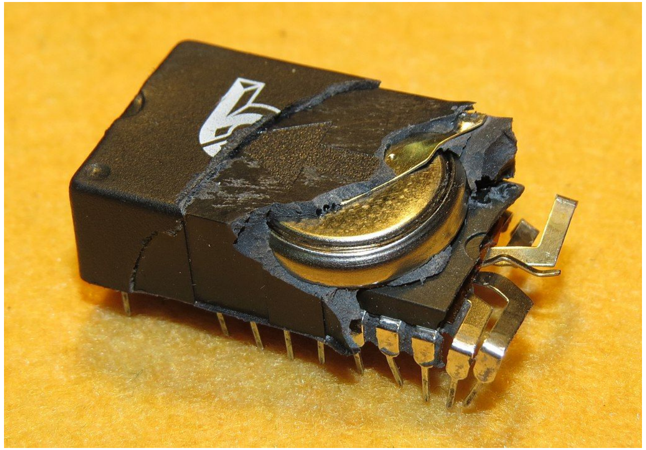
The keyboard driver uses interrupts to detect when a key is pressed or released. When the interrupt fires, the sink is notified accordingly.

Real Time Clock

A real time clock (RTC) is a device that provides the system with a value (hopefully) representing the current “time of day”.

The actual device itself simply increments it’s internal counter every second, and is usually battery powered, so that time continues progressing, when when the system is powered off.

The internal counter may or may not be set to the correct time - the device doesn’t care about this.



RTC Driver

You are tasked with implementing an RTC driver for the CMOS RTC.

This involves subclassing the RTC device class (already provided in the skeleton), and interrogating the CMOS register space.

The RTC device class provides an interface for accessing the current time-of-day.

This interface is realised through the "`read_timepoint`" method, and you must implement this routine.

This routine should interrogate the real RTC device, and fill in the values of the "`tp`" structure (passed in as a reference parameter).

CMOS RTC

The CMOS RTC is a device that is present on nearly every modern x86-based platform. It can be accessed by reading from I/O registers in a specific way.

On x86, the I/O registers are accessed using the `__in{b,w,l}` and `__out{b,w,l}` helper functions. The suffix indicates the width of the access.

- `b` = 1 byte/8 bits
- `w` = 2 bytes/16 bits
- `l` = 4 bytes/32 bits

These helpers are in the `infos/arch/x86/pio.h` header file, so you'll need to `#include` this in your source-code to use them.

Reading the values

Read the description of the CMOS area on the OSDev website in **great detail**.

Read the coursework specification document in **great detail**.

Pay attention to how values should be correctly read from the RTC, with regards to the access protocol for the CMOS registers, and the conditions that should be met to perform the read.

Do not use inline assembly for your implementation. InfOS provides all the functionality you require through helper methods.

Interpreting the values

You will need to determine whether or not the RTC is providing values in binary coded decimal (BCD), or binary format.

Your implementation MUST support both of these formats.

https://en.wikipedia.org/wiki/Binary-coded_decimal

Testing

You have no influence over whether or not the RTC provides values in BCD or binary format - this is up to QEMU. Write a small (external) test program to check that your conversion algorithms work.

Boot InfOS and run `/usr/date`

The results should be that the system date and time, should be approximately equal to the hardware date and time, which should be the same as your host system's date and time.

If you see:
warning: No RTC available to synchronise TOD
then your driver hasn't been loaded.

Questions/Clarifications?