

Jin-Soo Kim
(jinsoo.kim@snu.ac.kr)

Systems Software &
Architecture Lab.
Seoul National University

Fall 2024

I/O Devices



Three Pieces

- **Virtualization**

- Virtual CPUs
- Virtual memory

- **Concurrency**

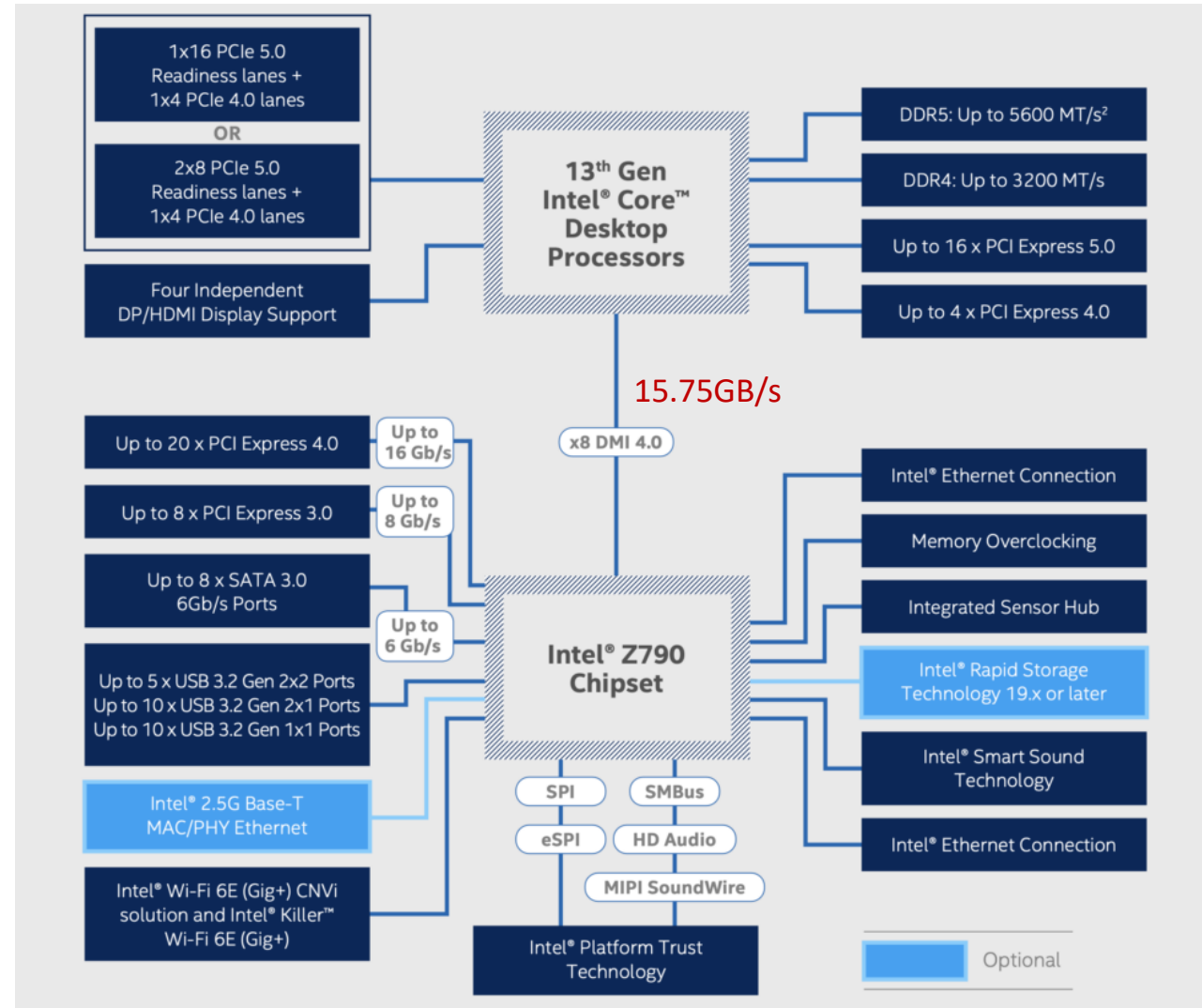
- Threads
- Synchronization

- **Persistence**

- How to make information persist, despite computer crashes, disk failures, or power outages?
- Storage
- File systems

Modern System Architecture

- Intel 13th Gen Intel Core Desktop Processor (a.k.a Raptor Lake)

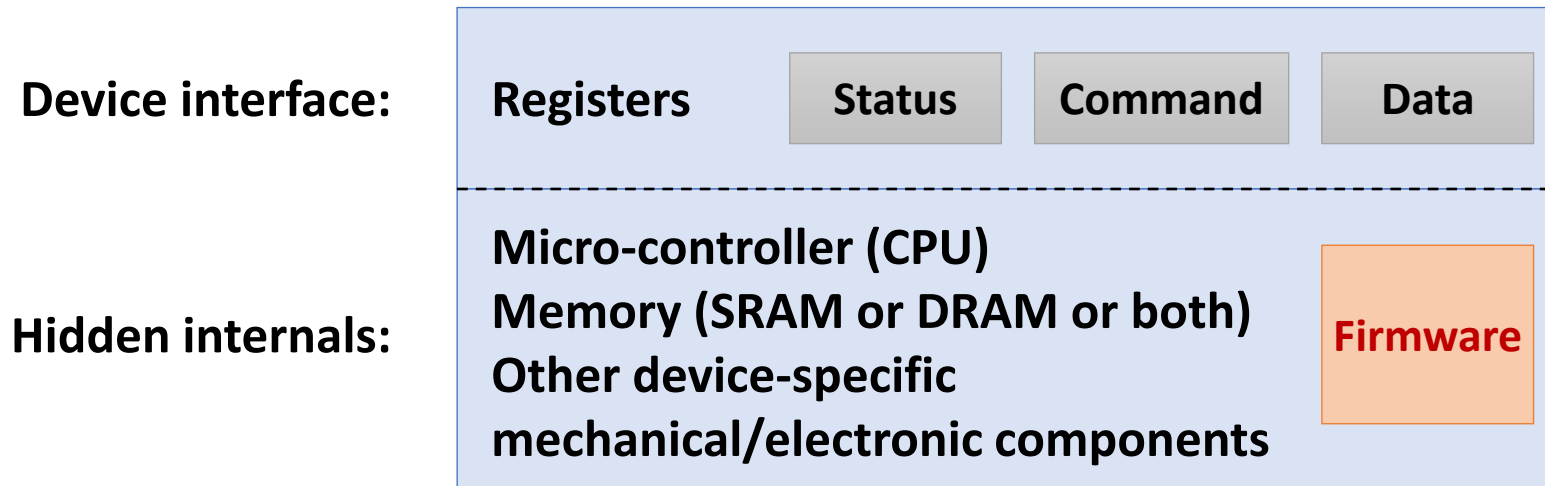


44.8GB/s/module
25.6GB/s/module
~4GB/s/lane
~2GB/s/lane

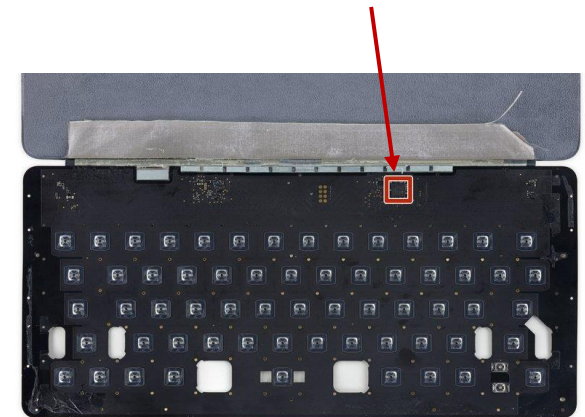
Source: <https://arstechnica.com/gadgets/2022/09/intels-first-13th-gen-core-cpus-include-few-surprises-but-many-cores/>

A Typical I/O Device

- **Control:** Special instructions (e.g., in & out in x86) vs. memory-mapped I/O (e.g., load & store)
- **Data transfer:** Programmed I/O (PIO) vs. DMA
- **Status check:** Polling vs. Interrupts



72MHz 32-bit
ARM Cortex-M3

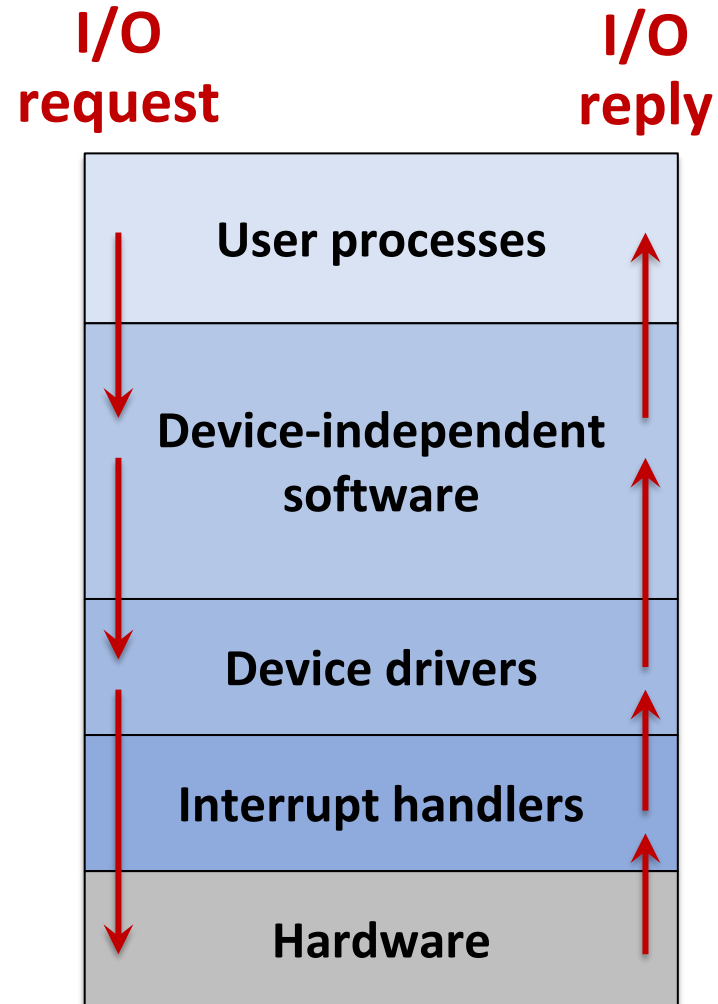


Source: <https://www.ifixit.com/Teardown/Smart+Keyboard+Teardown/53052>

Classifying I/O Devices

- device
 - Stores information in fixed-size blocks, each one with its own address
 - Typically, 512B or 4KB per block
 - Can read or write each block independently
 - Disks, tapes, etc.
- **Character device**
 - Delivers or accepts a stream of characters
 - Not addressable and no seek operation supported
 - Printers, networks, mouse, keyboard, etc.

I/O Stack



Make I/O call, format I/O, spooling

Naming, protection, blocking, buffering, allocation

Set up device registers, check status

Wake up driver when I/O completed

Perform I/O operation

Device Drivers

- **Device-specific code to control each I/O device**
 - Require to define a well-defined model and a standard interface
- **Implementation**
 - Statically linked with the kernel
 - Selectively loaded into the system during boot time
 - Dynamically loaded into the system during execution (especially for hot pluggable devices)
- **Variety is a challenge**
 - Many, many devices
 - Each has its own protocol

OS Reliability



OS Reliability and Device Drivers

- Reliability remains a crucial, but unresolved problem
 - 5% of Windows systems crash every day
 - Huge cost of failures: stock exchange, e-commerce, etc.
 - Growing “unmanaged systems”: digital appliances, CE devices
- OS extensions are increasingly prevalent
 - 70% of Linux kernel code
 - Over 35,000 drivers with over 120,000 versions on WinXP
 - Written by less experienced programmers
- Extensions are a leading cause of OS failure
 - Drivers cause 85% of WinXP crashes
 - Drivers are 7 times buggier than the kernel in Linux