Lecture 3 — Computer systems and the command line

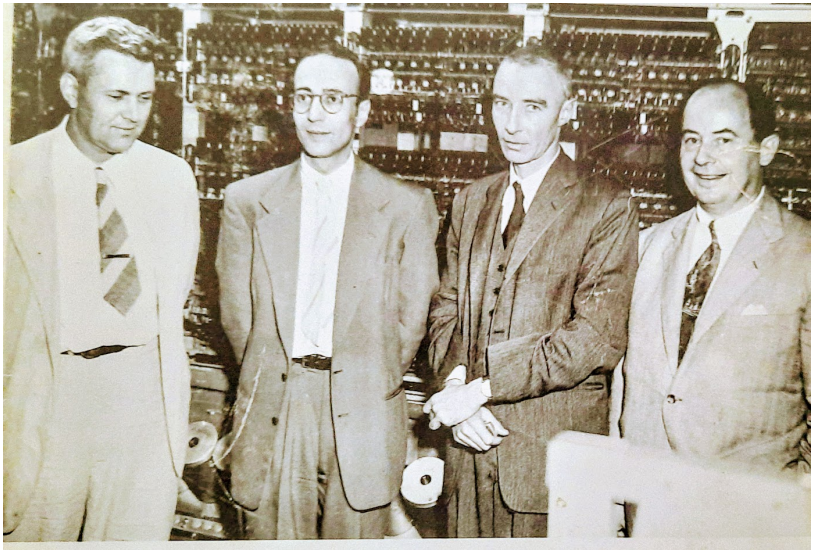
C1 Research Computing — MPhil in Data Intensive Science, Michaelmas 2026

Boris Bolliet

University of Cambridge

Outline

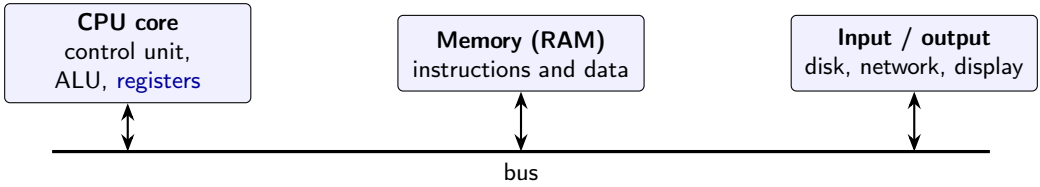
- Von Neumann model, processor cores and hardware threads
- Memory hierarchy, magnetic-core memory and disk storage
- Operating system, shell, file system, processes, permissions
- Shell scripts: arguments, redirection, exit status, loops



Principals of the Electronic Computer Project of the Institute for Advanced Study: Julian H. Bigelow, chief engineer; Herman H. Goldstine, assistant project director; J. Robert Oppenheimer, director of the Institute; and John

Source: Institute for Advanced Study, Electronic Computer Project

Von Neumann model

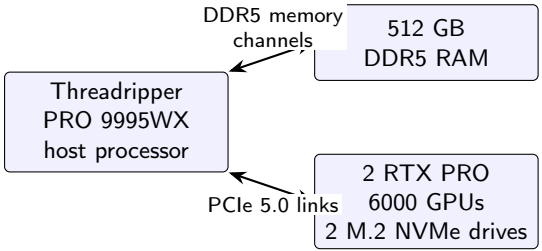


CPU: central processing unit; ALU: arithmetic logic unit; RAM: random-access memory.

- Stored-program design (1945): instructions and data are bit patterns in memory; a new program can be loaded without rewiring.
- This diagram uses one bus to represent communication between the processor, memory and input/output devices; modern computers use several distinct interconnects.
- Basic instruction cycle: fetch, decode, execute.

Von Neumann architecture

Memory and I/O links in Orion



Schematic of the Orion configuration.

- PCIe (Peripheral Component Interconnect Express) is a point-to-point I/O interconnect. A $\times 16$ link has 16 lanes.
- The Von Neumann diagram's single bus is a simplification. Orion uses DDR5 memory channels for RAM and PCIe 5.0 links for I/O devices.
- The Threadripper PRO 9995WX supports 8 DDR5 memory channels and PCIe 5.0. Orion has 512 GB RAM, two GPUs and two M.2 NVMe drives.

PCI Express · PCI-SIG architecture · AMD 9995WX specifications

Orion workstation specifications

Motherboard	ASUS® PRO WS WRX90E-SAGE SE (WRX90, ECC RDIMM, PCIe 5.0)
Memory (RAM)	512GB Kingston DDR5 6400MHz ECC Registered (8 x 64GB)
Graphics Card	96GB PNY NVIDIA RTX Pro 6000 Blackwell Workstation Edition, 24064 CUDA CORES 4 x DP - *EDUCATION*
2nd Graphics Card	96GB PNY NVIDIA RTX Pro 6000 Blackwell Workstation Edition, 24064 CUDA CORES 4 x DP - *EDUCATION*
1st M.2 SSD Drive	4TB SAMSUNG 9100 PRO M.2, PCIe 5.0 NVMe (up to 14,800MB/R, 13,400MB/W)
2nd M.2 SSD Drive	4TB SAMSUNG 9100 PRO M.2, PCIe 5.0 NVMe (up to 14,800MB/R, 13,400MB/W)

Persistent storage: 2×4 TB NVMe SSDs; 28 TB Seagate IronWolf Pro 3.5-inch HDD. 7200 rpm; 512 MB cache. SSD: solid-state drive; HDD: hard disk drive.

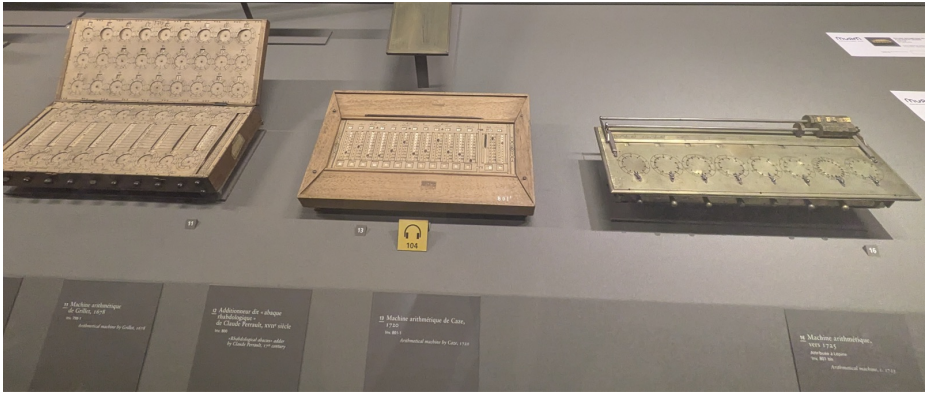
PCSpecialist build specifications, page 1; selected component rows

Cores and hardware threads

- A processor can contain many cores. Each core has control logic, arithmetic logic units (ALUs) and registers.
- A hardware thread maintains an instruction stream and its own register state.
- Two hardware threads on one core share that core's execution units.
- Orion: 96 cores $\times 2$ hardware threads per core $= 192$ logical CPUs.

AMD Ryzen Threadripper PRO 9995WX

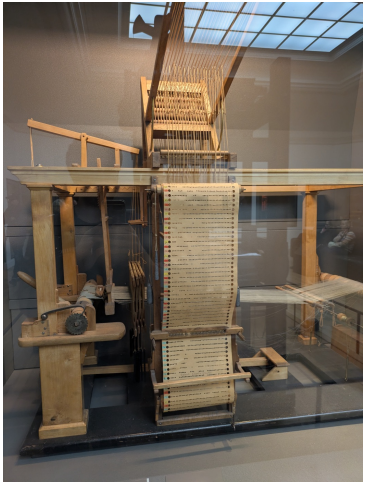
Mechanical calculators



- Grillet (1678), Perrault (17th c.), Caze (1720; c. 1725).
- The operator sets dials or rods for each calculation. No program is stored.

Photo: Musée des Arts et Métiers, Paris | Mechanical calculator

Programs on punched cards



- In the Jacquard loom (1804), each card controls one row of weaving. A hole raises a thread; an unpunched position leaves it down.
- Changing the card sequence changes the woven pattern without changing the loom.
- Babbage designed the Analytical Engine (1837) to read instructions from cards. Hollerith's tabulators used cards to count the 1890 US census.
- The IBM card (1928) had 80 columns and 12 rows. Each column encoded one character. Punched-card decks remained in use for computer programs into the 1970s.

Photo: Musée des Arts et Métiers, Paris | Jacquard machine | Analytical Engine | Punched card

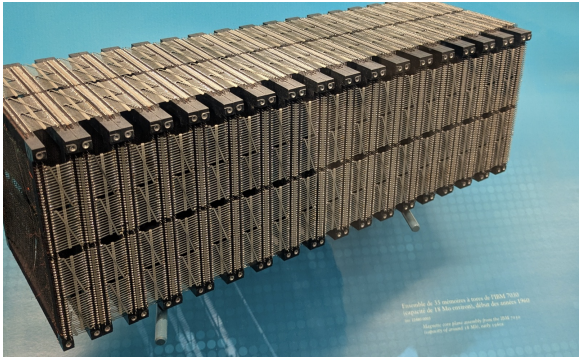
Memory hierarchy and access times

Level	What	Size	Access time
registers	values used directly by instructions	16 × 64-bit general-purpose registers per hardware thread	0.3 ns, 1 cycle
L1 cache	copy of recent data, per core	32–128 KB	1 ns, 4 cycles
L2, L3 cache	larger copies, per core or shared	1–16 MB; tens of MB	4–20 ns
RAM	main memory (DRAM chips)	8–128 GB (laptop) 256–512 GB (CSD3)	100 ns, 300 cycles
SSD	flash storage, no moving parts	TB	100 µs, 10^5 cycles
HDD, network	spinning disk; another machine	TB–PB	1–10 ms, 10^7 cycles

- At 3–4 GHz, one clock cycle lasts about 0.25–0.33 ns.
- Caches keep copies of recently used data close to the CPU. A later read can be faster when the data remain in cache.
- Contiguous arrays can help the CPU reuse nearby values from cache.

Processor register | AMD64 register model | Memory hierarchy | CPU cache | Latency numbers every programmer should know

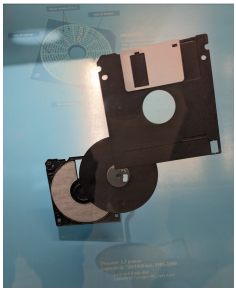
Magnetic-core memory (1960s)



- Each bit was stored in a magnetised ferrite ring. Wires passed through the rings to set or read their values.
- Magnetic-core memory served as main memory from the mid-1950s to the early 1970s.
- The IBM 7030 Stretch (1961) held up to 262 144 64-bit words, or 2 MiB. The lecture laptop's 128 GiB of RAM is 65 536 times that capacity.

Photo: Musée des Arts et Métiers, Paris | Magnetic-core memory | IBM 7030 Stretch

Disk storage: 1956 to SSD





- The IBM 350 (1956) stored five million characters on fifty 24-inch platters.
- The 3.5-inch floppy disk (left) stored 1.44 MB on flexible magnetic media.
- The 2001 hard disk (right) stored 40 GB on two platters. Its moving head made access slower than RAM.
- The lecture laptop uses a 1 TB SSD for storage and 128 GiB of RAM for active programs and data.

Photos: Musée des Arts et Métiers, Paris | IBM 350 | Hard disk drive | Solid-state drive

✎ Orion: CPU topology and RAM

```
$ ssh orion lscpu
Model name:          AMD Ryzen Threadripper PRO 9995WX
CPU(s):              192
Thread(s) per core: 2
Core(s) per socket: 96
Socket(s):           1
$ ssh orion free -h
Mem:                 502Gi total
```

- One processor socket contains 96 cores; each supports two hardware threads.
- Linux reports 192 logical CPUs: 96×2. A logical CPU is one hardware thread.
- RAM modules: 8×64 GB (binary capacity) = 512 GiB nominal. Linux reports 502 GiB; about 9 GiB is reserved for firmware/devices.

GB and GiB | lscpu | free

Operating system

- An operating system manages hardware and provides services to programs.
- The kernel is its central program. It schedules processes and manages memory, files and devices.
- Python's `open("C9.csv").readline()` requests the first line through a system call to the kernel.
- Examples: macOS; Linux, including Rocky Linux 8 on CSD3.



Her (2013)

Operating system

Shell

- A shell is a program that reads commands and starts other programs: zsh (macOS), bash (Linux, CSD3).
- A command names a program and may include options and arguments. For example, `ls -l C9.csv` asks `ls` to list the file in long format. The shell searches for programs using `PATH`.
- `man ls` (manual) opens the reference page of a command, `q` closes it. On Linux, `ls --help` prints a summary.
- Terminal provides the window; the shell runs inside it and reads the commands.

The Missing Semester: the shell

Shell startup files

Bash

- `~/.bashrc`
- `~/.bash_profile`

Zsh

- `~/.zshrc`
- `~/.zprofile`

- Startup files store settings such as `PATH`, the prompt and aliases. Example: `alias ll='ls -l'`.

Bash startup files | Zsh startup files

File system

- A file system organises stored data into files and directories. The operating system presents them in a tree rooted at `/`.
- Examples: `/usr/bin`, `/opt/homebrew/bin`, `/home/<crsid>` (Linux), `/Users/<name>` (macOS).
- `~` is the home directory; `.` the current directory, `..` its parent.
- An **absolute path** starts at `/`; a **relative path** starts at the current directory.
- Names are case-sensitive on Linux; spaces need quotes.

Filesystem Hierarchy Standard

Directory tree

Directory	Contents	Example
`/bin`, `/usr/bin`	the system's programs (binaries)	`/bin/ls`, `/usr/bin/git`
`/usr`	programs, libraries, manuals; `/usr/local`: local installs	`/usr/share/man`
`/opt`	optional add-on software	`/opt/homebrew/bin/python3.13`
`/etc`	configuration, as text files	`/etc/hosts`, `/etc/shells`
`/var`	variable data: logs, databases, mail	`/var/log`
`/tmp`	temporary files, cleared at reboot	
`/dev`	devices, presented as files	`/dev/disk3s1s1`, `/dev/null`
`/home`, `/Users`	home directories (Linux, macOS)	`/Users/boris`
`/proc`, `/sys` (Linux)	kernel and process information	`/proc/cpuinfo`
`/mnt`, `/Volumes`, `/rds`	other disks, mounted in the tree	`/rds/user/<crsid>/hpc-work`

- The Filesystem Hierarchy Standard describes common Linux directory names. macOS has additional directories, including `/Applications` and `/Users`.
- A mounted filesystem is attached at a directory. `df -h` reports filesystems and their mount points.

Filesystem Hierarchy Standard | CSD3: storage

✎ Filesystem tree

```
$ ls /
Applications  Library  System  Users  Volumes  bin  dev  etc
home  opt  private  sbin  tmp  usr  var
$ which ls git python3.13
/bin/ls
/usr/bin/git
/opt/homebrew/bin/python3.13
$ ls /usr/bin | wc -l
     915
$ ls -ld /tmp
lrwxr-xr-x@ 1 root  wheel  11 16 Aug  2025 /tmp -> private/tmp
```

- `which` prints the path of each program found through `PATH`. `wc -l` counts lines; this command reports 915 entries.
- `ls -l` uses long format. The `-d` option lists the directory itself. The `l` at the start of the mode shows that macOS `/tmp` is a symbolic link to `/private/tmp`.

which | Symbolic link

File system formats

Format	Used on	Note
APFS	macOS (the lecture laptop's disk)	case-insensitive by default
ext4, XFS	Linux (Ubuntu, Rocky Linux, cloud)	case-sensitive; journaled
NTFS	Windows	read-only on macOS without extra software
exFAT, FAT32	USB sticks, SD cards	no owner or permissions; FAT32 files $<$ 4 GiB
NFS	network: CSD3 home directories	one server, seen by every node
Lustre	parallel: `/rds` on CSD3	many servers at once, for the I/O of jobs

```
$ diskutil info / | grep Personality
   File System Personality:   APFS
$ df -Th /        # Linux: -T adds a Type column (ext4, xfs, nfs, lustre)
```

- A filesystem records directory entries, file data and metadata on a volume. The directory paths can look the same across different filesystems.
- FAT32 does not store Unix ownership or executable permissions. The `chmod u+x` command cannot add an executable bit to a FAT32 file.

File system | APFS | ext4 | Lustre | CSD3: storage

✎ Navigation and files

```
$ cd ~/attractor
$ curl -LO https://teaching.borisbolliet.io/c1-2026/materials/class03-demo.zip
$ unzip -j class03-demo.zip && rm class03-demo.zip
$ pwd
/Users/boris/attractor
$ ls -l
$ mkdir scratch && cp C9.csv scratch/copy.csv && ls scratch
copy.csv
$ mv scratch/copy.csv scratch/C9_copy.csv
$ rm -r scratch
```

- `curl -L` follows redirects; `-O` saves using the remote filename. `unzip -j` removes archive paths while extracting.
- `pwd` prints the current directory. `ls -l` lists files in long format. `mkdir`, `cp` and `mv` create, copy and rename files or directories.
- `rm -r` removes a directory and its contents without a confirmation prompt. `man ls` opens the manual page; press `q` to close it.

GNU coreutils manual

✎ Data from shell commands

```
$ wc -l C9.csv
     142 C9.csv
$ head -3 C9.csv | cut -d, -f1,2
time (days), algae (10^6 cells/ml)
0.00,1.92
1.02,1.58
$ grep -c ",0.00$" C9.csv
68
$ sort -t, -k2 -n C9.csv | tail -1 | cut -d, -f1,2
14.00,2.78
```

- `wc -l` counts lines. `head -3` prints the first three. `cut -d, -f1,2` selects comma-separated fields 1 and 2.
- `grep -c` counts matching lines. `sort -t, -k2 -n` sorts numerically by field 2; `tail -1` prints the final line.
- `|` (pipe) feeds one program's output to the next. 142 lines = header + 141 samples; 68 days with the supply off.

Processes and environment

```
$ ps            # processes of this terminal
$ top           # all processes, live (q to quit)
$ echo $PATH
$ which python3.13
$ export MYVAR=1 ; echo $MYVAR
```

- `ps` lists processes attached to the terminal. `top` updates a live process list. Press `q` to quit.
- A running program is a process with a process ID (PID). `Ctrl-C` interrupts the foreground process; `kill <PID>` sends a signal to the selected process.
- **Environment variables** (`PATH`, `HOME`, `VIRTUAL_ENV`) are inherited by every program the shell starts.

ps | environ

Permissions

```
$ ls -l period.py
-rw-r--r--  1 boris  staff  61  8 Oct 07:04 period.py
$ chmod u+x period.py && ls -l period.py
-rwxr--r--  1 boris  staff  61  8 Oct 07:04 period.py
```

- In `-rw-r--r--`, the first `-` marks a regular file; the next triplets are permissions for owner (`boris`), group (`staff`) and others.
- `r`=4, `w`=2, `x`=1: `rw-` is $1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 6$; `r--` is 4.
- The three triplets give 644: owner 6, group 4, others 4. `chmod u+x` adds execute permission for the owner; the new mode is 744.
- A script needs `x` to run as `./period.py`; `python period.py` needs only `r`.

chmod

Shell scripts

```
#!/bin/bash
set -euo pipefail
file="$1"
n=$(( $(wc -l < "$file") - 1 ))
peak=$(sort -t, -k2 -n "$file" | tail -1 | cut -d, -f1,2)
echo "$file: $n samples, peak (day,algae) $peak"
```

```
$ chmod u+x summarise.sh && ./summarise.sh C9.csv
C9.csv: 141 samples, peak (day,algae) 14.00,2.78
```

- `#!/bin/bash` selects Bash. `"$1"` is the first argument; quotes keep a filename with spaces together.
- `$(...)` inserts command output; `$((...))` computes integers. `wc -l` counts the header, so the script subtracts one.
- `-u` rejects unset variables; `pipefail` propagates pipeline failures; `-e` exits on most command errors.

Bash manual: shell scripts | set

Redirection and exit status

```
$ ./summarise.sh C9.csv > summary.txt      # output to a file (overwrite)
$ ./summarise.sh C1.csv >> summary.txt     # append
$ cat summary.txt
C9.csv: 141 samples, peak (day,algae) 14.00,2.78
C1.csv: 366 samples, peak (day,algae) 335.05,1.91
$ ./summarise.sh missing.csv 2> errors.txt # error messages to a file
$ echo $?
1
$ ./summarise.sh C9.csv > all.txt 2>&1     # both streams into one file
```

- A process has standard input (`stdin`, 0), standard output (`stdout`, 1) and standard error (`stderr`, 2). `>` overwrites a file; `»` appends; `2>` redirects errors. In `> file 2>&1`, both output streams go to the same file.
- A command returns an exit status: 0 usually means success, and a non-zero value indicates failure. `$?` stores the previous command's status. `A && B` runs B only if A succeeds.

Bash manual: redirections | exit status

Loops over files

```
$ for f in C*.csv; do ./summarise.sh "$f"; done
C1.csv: 366 samples, peak (day,algae) 335.05,1.91
C10.csv: 198 samples, peak (day,algae) 7.00,2.19
C2.csv: 190 samples, peak (day,algae) 133.00,1.85
...
C9.csv: 141 samples, peak (day,algae) 14.00,2.78
$ for f in C*.csv; do ./summarise.sh "$f"; done > peaks.txt
$ wc -l peaks.txt
      10 peaks.txt
```

- In `for f in C*.csv; do ...; done`, the shell expands `C*.csv` to matching filenames. Each iteration assigns one name to `f`; the example processes C1.csv, C10.csv and the other runs.
- Redirecting the loop's output to `peaks.txt` writes one summary line for each matching file.

Bash manual: looping constructs | filename expansion

✎ Fetching ten runs

```
#!/bin/bash
# fetch_runs.sh -- download the runs listed in runs.txt from figshare
set -euo pipefail
while read -r name id; do
  echo "$name.csv <- figshare file $id"
  curl -fsSL -o "$name.csv" "https://ndownloader.figshare.com/files/$id"
done < runs.txt
```

```
$ ./fetch_runs.sh && for f in C*.csv; do ./summarise.sh "$f"; done
```

- `while read -r name id; do ...; done < runs.txt` reads each line into `name` and `id`. The `-r` option preserves backslashes literally.
- `curl -f` treats HTTP errors as failures. `-s` hides progress, `-S` still prints errors, `-L` follows redirects and `-o` names the output file.

curl manual | the data set on figshare

Summary

- The processor executes instructions stored in memory. Registers, caches, RAM and persistent storage differ in capacity and access time.
- The kernel manages processes, memory, files and devices. The shell starts programs; paths identify files and permissions control access.
- Pipes and redirection connect command output to other commands or files. Scripts use arguments and loops to process files.