Computer Systems and Applications

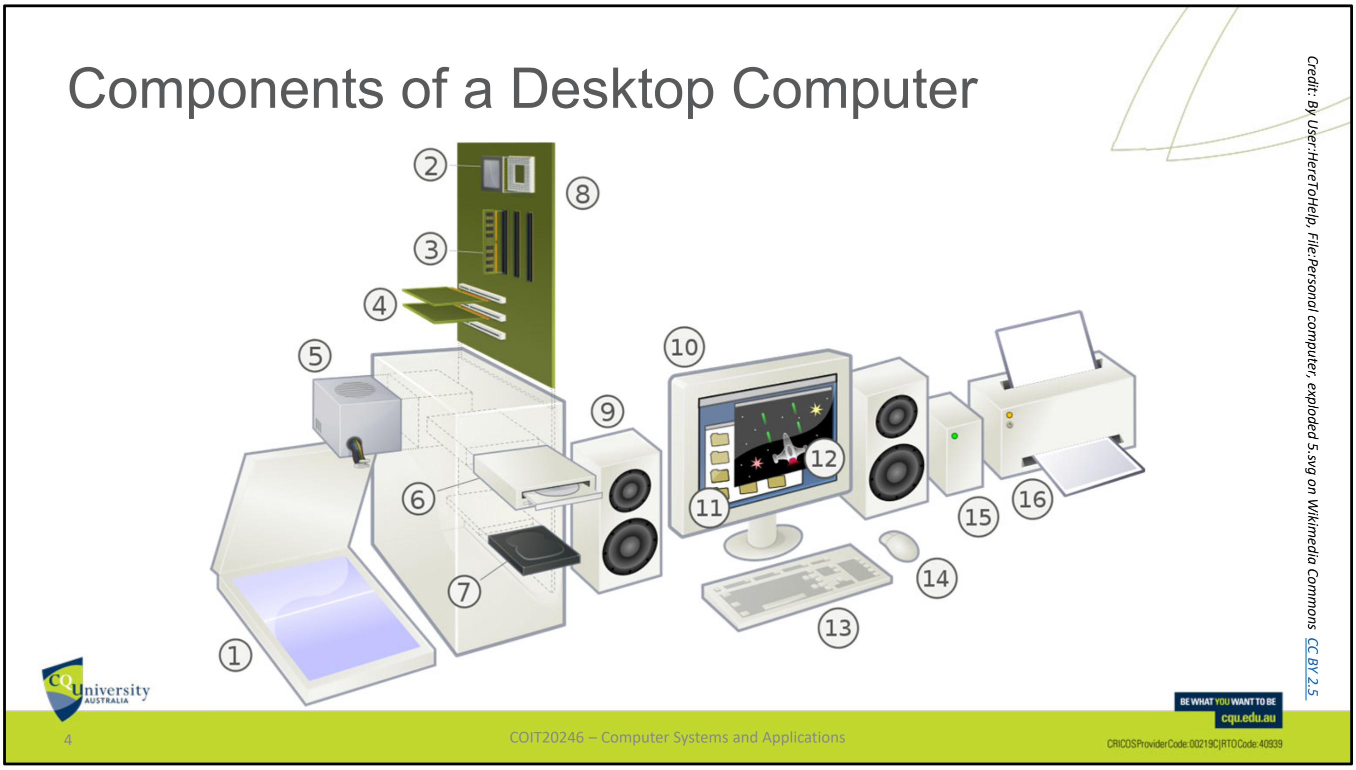
COIT20246 Networking and Cyber Security

BE WHAT YOU WANT TO BE
cqu.edu.au

What will you learn?

- Components of a computer and performance metrics
- Key features of operating systems
- Concept of virtualizing computer hardware
- How text, audio and video is converted to binary

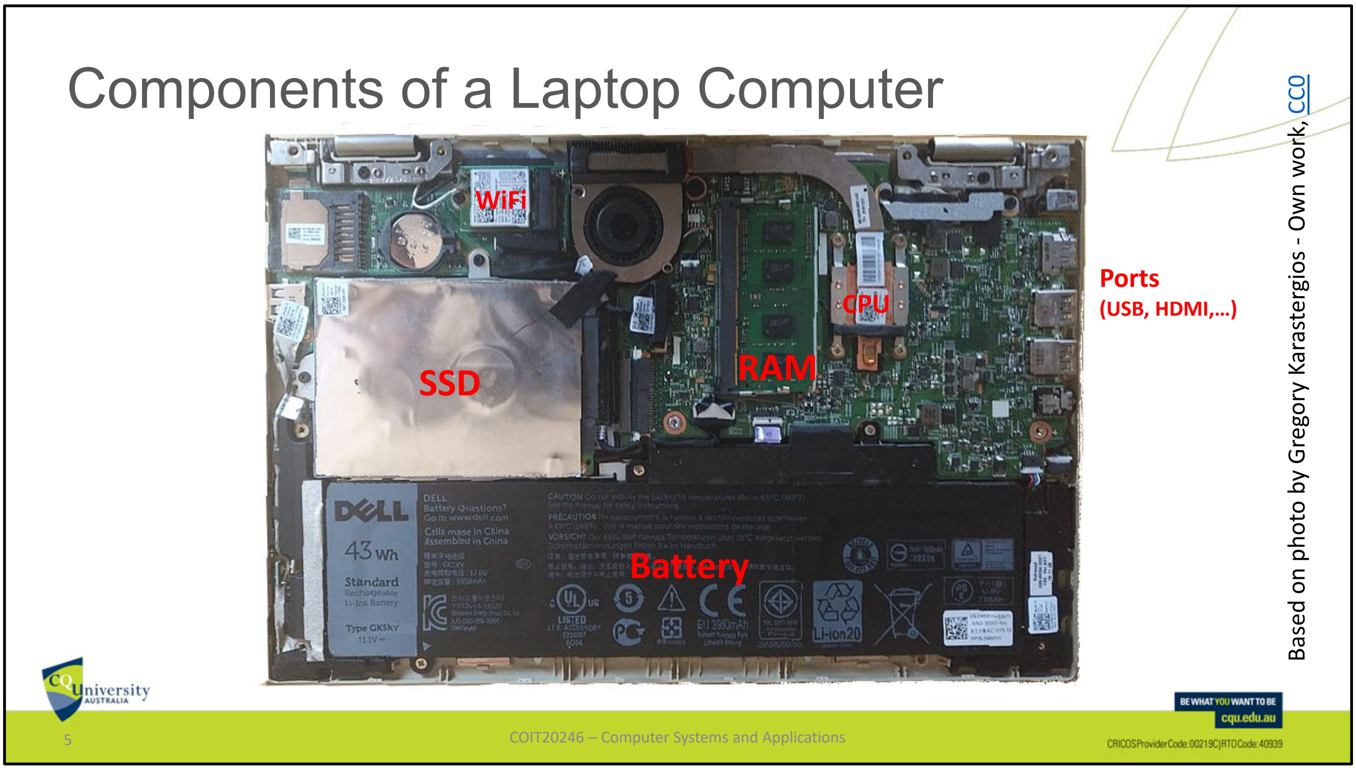
Computer Hardware



Most of us are already aware of the components in various computing devices, including desktop computers, laptops and mobile phones. Here is an example of typical PC components.

1. Scanner
2. Central Processing Unit (CPU)
3. Random Access Memory (RAM)
4. Expansion cards
5. Power Supply Unit (PSU)
6. Optical disc drive
7. Hard disk drive (HDD), or more common now is a Solid State Drive (SSD)
8. Motherboard
9. Speakers
10. Monitor
11. System software
12. Application software
13. Keyboard
14. Mouse
15. External disk drive
16. Printer

Think of your personal computer (not laptop or mobile phone) at home, or one you have recently used (e.g. in a lab). Does it have all of these components? Does it have additional components?



While in a different form factor, laptops have similar components to desktop computers.

(The components shown are guesses. It may be SSD or HDD – difficult to see from the photo).

Other computers …

- Mobile phones
- Tablets
- Workstations
- Servers
- Mainframes
- Supercomputers

- Embedded systems
- Network switch, router, AP
- Printers, copiers, projectors
- Wearable devices
- TV, media players
- Microcontroller boards (Arduino, Raspberry Pi)
- Sensors/actuators (Internet of Things)

Computer Specifications and Quantities (example)

	Dell Precision 5820 Desktop	MacBook Pro 13	Google Pixel 7 Pro	Arduino Uno Rev3
CPU	Intel Xeon, 3.6 GHz, 4 core, 8 threads	Apple M2, 3.5 GHz, 8-core CPU, 10-core GPU	Google Tensor G2, 2.8 GHz, 8-core CPU, 7-core GPU; Titan M2 security coprocessor	ATmega328P 16 MHz
RAM	16 GB GDDR6 ECC	8 GB	12 GB LPDDR5	2 KB
Disk	512 GB PCIe SSD	256 GB SSD	128 GB	32 KB
Display	30 in 2560x1600 LED-backlit LCD	13.3 in 2560x1600 LED-backlit IPS	6.7 in 1440x3120 OLED	-
Network	Intel 1 Gb/s Ethernet, WiFi 802.11ac, Bluetooth	WiFi 6 802.11ax, Bluetooth 5	WiFi 6 802.11ax, Bluetooth 5.2, UWB, NFC, 5G	-
Approx cost	$10,000	$2,000	$1,300	$40

Selected specifications for some example computers in early 2023. You are not expected to understand all the values, but do take note of how different quantities are measured. For example:

- CPU, processing speed in Hz (cycles per second), although the number of processors or "cores" is also important.
- Storage, both RAM and disk, in Bytes. Not captured here, but the speed of access is also important (e.g. SDD vs HDD)
- Displays in size (inches/cm) and resolution (pixels)
- Networking in bits per second (b/s or bps). Check online for range of speeds offered by WiFi 802.11ac vs 802.11ax.

Common Units and Prefixes

- Processing speed: Hz (cycles per second)
- Storage: B (Byte = 8 bits)
- Communications speed: b/s (bits per second)

(Hz also used for signal frequency and bandwidth, e.g. WiFi channels)

- Prefixes

P > T > G > M > K > 1 > m > μ > n > p

$10^{15} > 10^{12} > 10^{9} > 10^{6} > 10^{3} > 1 > 10^{-3} > 10^{-6} > 10^{-9} > 10^{-12}$

While for communications we use decimal base (K = 10^3 = 1000), be aware for storage we often use binary base (K = 2^{10} = 1024), and use "Ki" instead of "K"

How much memory does your computer have?

System Information says 16.0 GiB = 16 x 2^{30} = 17,179,869,184 Bytes,
but only 15.6 GiB of physical memory = 16,750,372,454 Bytes

System Information

File Edit View Help

System Summary: Hardware, Components, Software

Item	Value
Installed Physical Memory (RAM)	16.0 GB
Total Physical Memory	15.6 GB
Available Physical Memory	4.02 GB
Total Virtual Memory	25.6 GB
Available Virtual Memory	7.36 GB

But PowerShell shows different values ???

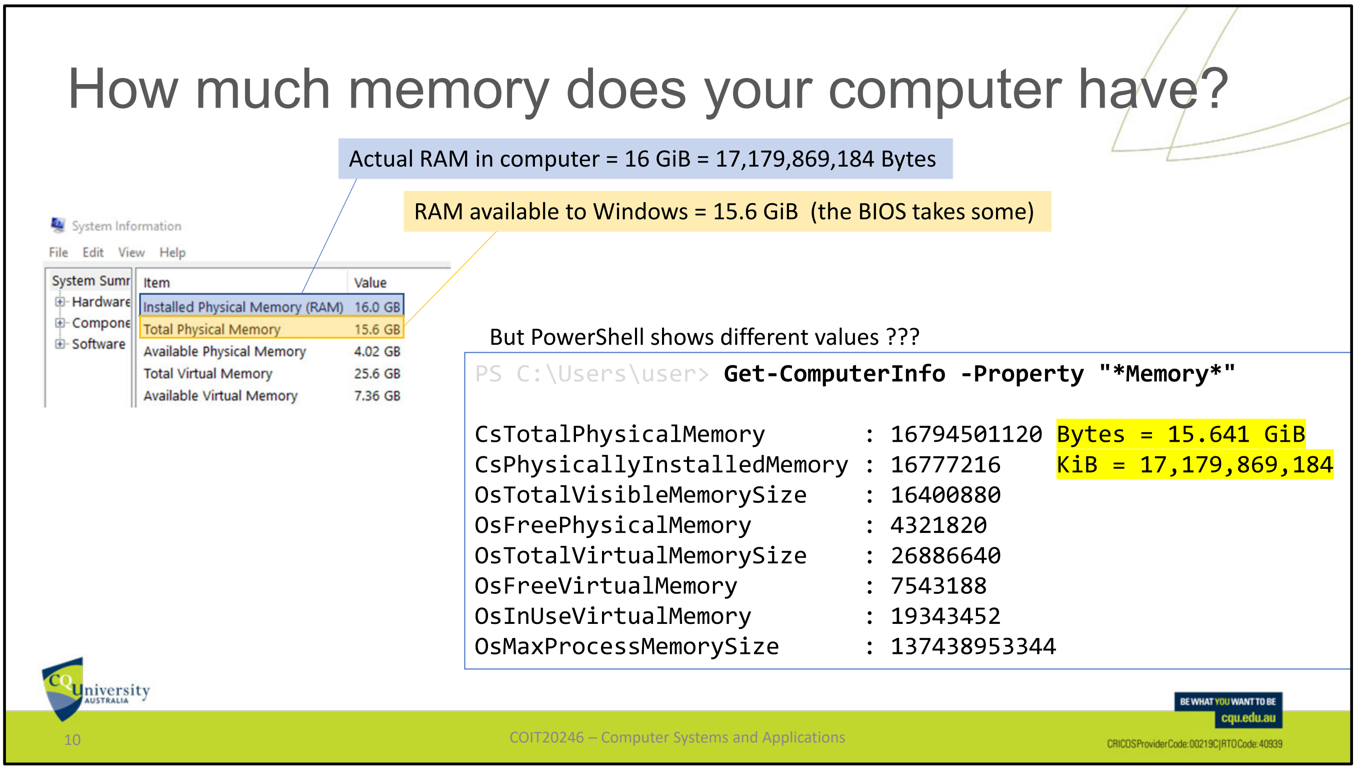
```
PS C:\Users\user> Get-ComputerInfo -Property "*Memory*"

CsTotalPhysicalMemory       : 16794501120
CsPhysicallyInstalledMemory : 16777216
OsTotalVisibleMemorySize    : 16400880
OsFreePhysicalMemory        : 4321820
OsTotalVirtualMemorySize    : 26886640
OsFreeVirtualMemory         : 7543188
OsInUseVirtualMemory        : 19343452
OsMaxProcessMemorySize      : 137438953344
```


In Windows, open “System Information” to see the amount of RAM on your computer. This example shows 16 GB, as expected.

Note that memory (RAM, disks) is typically reported in binary base, i.e. powers of 2. More accurate, the RAM is 16 GiB. That is, 17,179,869,184 Bytes, not 16,000,000,000 Bytes.

PowerShell will report various memory sizes with the “Get-ComputerInfo” command. But they don’t seem to match what System Information reports. Try to work out why they are different …



First note that while there is 16 GiB installed in the computer, the BIOS may reserve some RAM. That RAM reserved by the BIOS is not available to the operating system. So Windows has a total physical memory of 15.6 GiB.

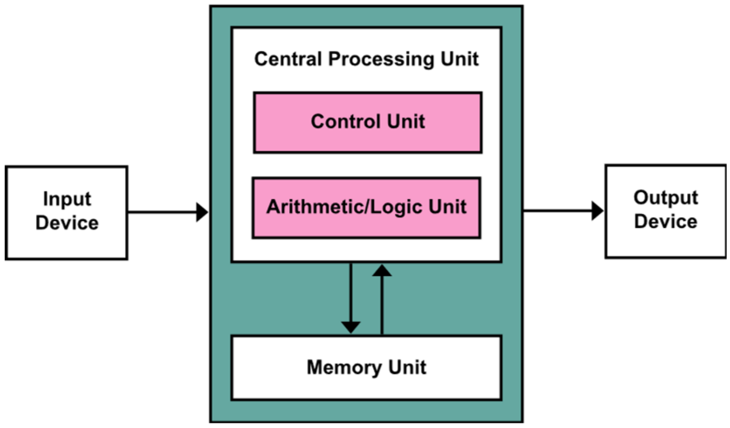
Next, be careful of the prefixes/units used in PowerShell. For CSTotalPhysicalMemory it is “Bytes” but for CSPhysicallyInstalledMemory it is “KiB”.

We can convert CSTotalPhysicalMemory to GiB by dividing by 2^30, that is, 16,794,501,120 / 2^30 = 15.6410980 GiB. This matches that reported for Total Physical Memory in System Information (well close, it appears System Information shows only one decimal point).

And we can convert CSPhysicallyInstalledMemory to Bytes by multiplying by 2^10, that is, 16,777,216 x 1024 = 17,179,869,184 Bytes. This exactly matches the value shown in System Information, and what we expect.

In summary, my computer has 17,179,869,184 Bytes of actual memory, but the BIOS reserves 385,368,064 Bytes, leaving 16,794,501,120 Bytes for the operating system.

Simplified Model of a Computer



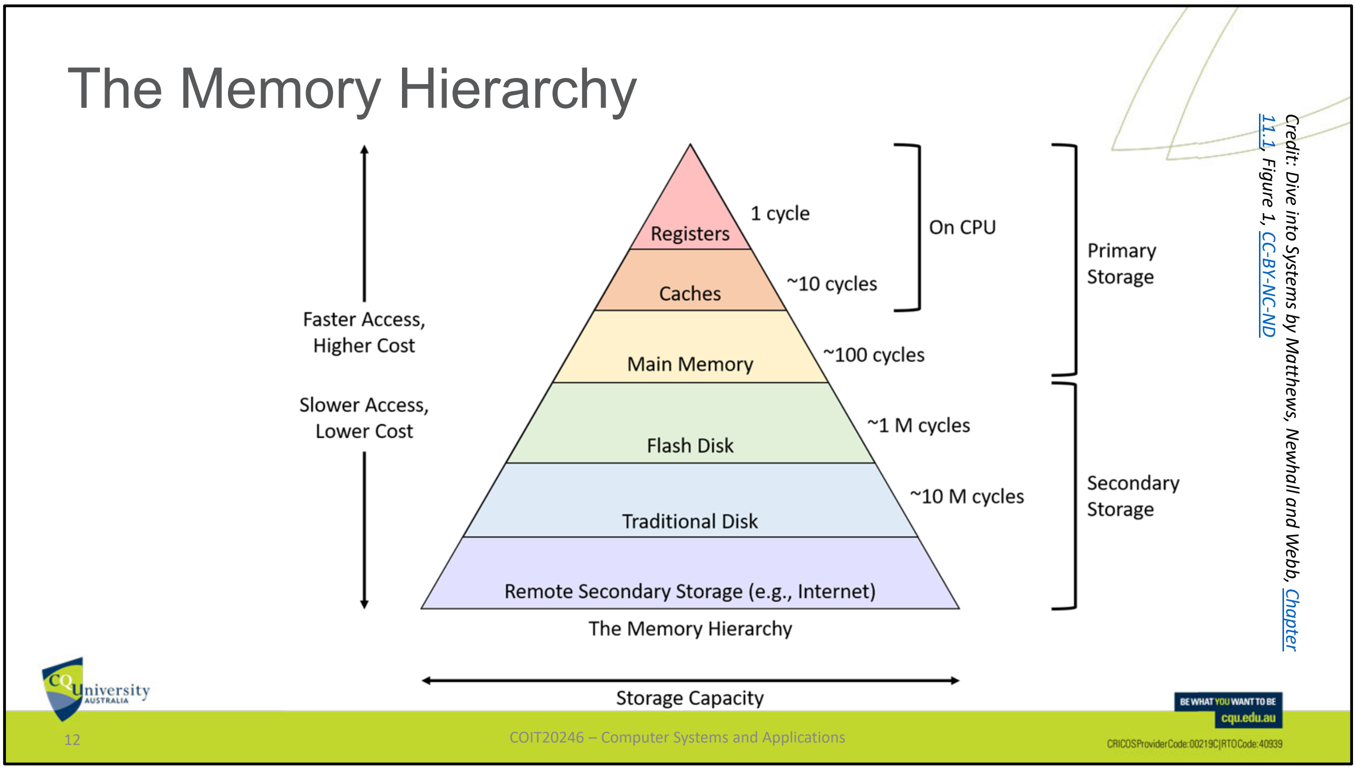
von Neumann architecture

ALU: performs arithmetic and logic operations on *operands*, e.g. add, mul, and, xor. Operations are denoted by *opcodes*.

CU: manages sequence of operations to be performed by ALU by directing operation of memory, input/output devices. Keeps track of operation to be executed with *program counter*.

Memory: stores data and instructions (i.e. programs to be executed).
- Volatile: requires power to maintain stored information, e.g. RAM, CPU cache
- Non-volatile: ROM, Flash,

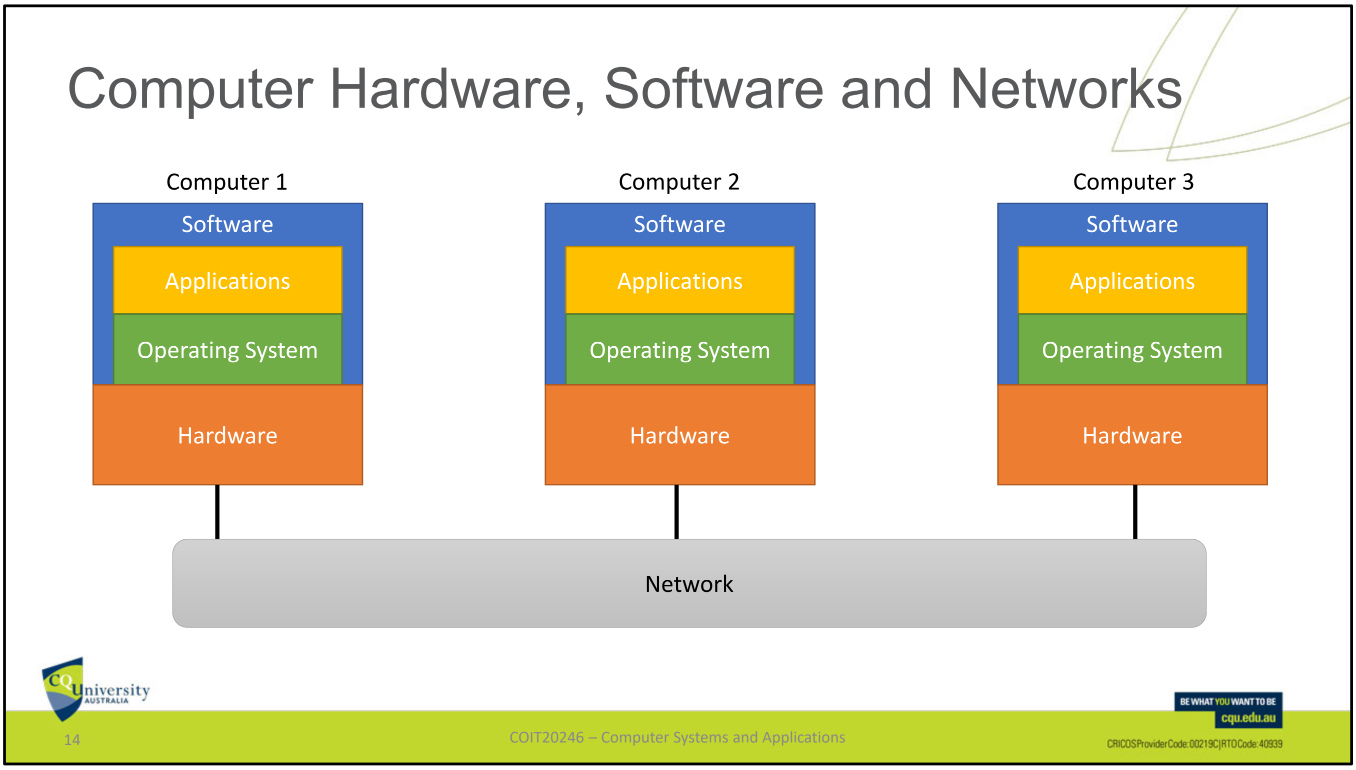
For more information about the von Neumann architecture, including example of how the components interact, see Dive Into Systems, chapter 5, https://diveintosystems.org/book/C5-Arch/von.html



This figure shows different types of memory, ranging from registers used in the CPU to online storage. Generally, the faster to access memory, the higher the cost of that memory. For more explanation about the memory hierarchy and the different levels, see Dive into Systems https://diveintosystems.org/book/

Computer Hardware and Software

- A computer program contains set of instructions for CPU to execute
  - Application software is a computer program to perform some task for the user
- Operating system is collection of software that makes it easy to run applications, e.g. allow programs to …
  - Interact with input/output devices
  - Share memory and access storage
  - (Appear to) run at same time as other programs
- An OS virtualises resources to applications
  - E.g. a disk stores bits in blocks; the OS presents the disk as files/directories to applications
- Modern operating systems contain other supporting software for …
  - Networking, security, user management, …

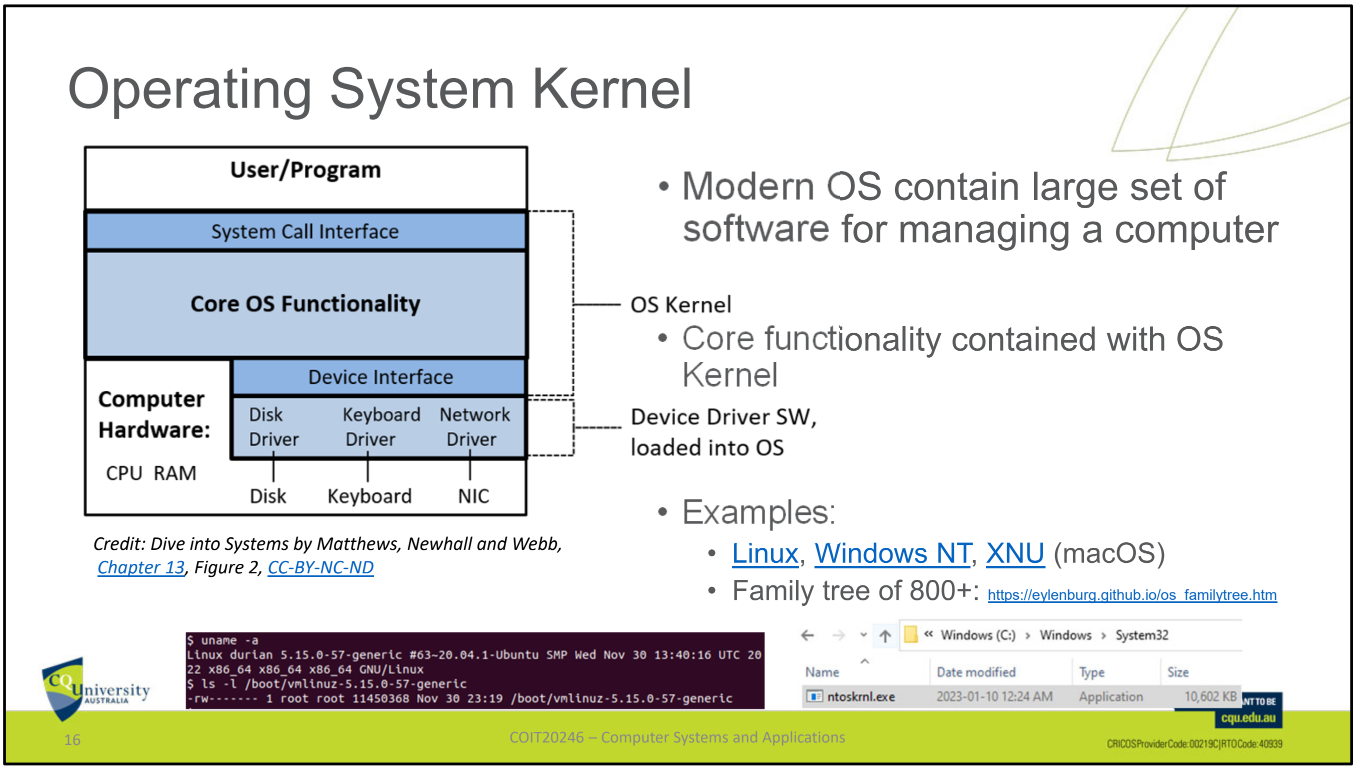


A simplified model of a computer splits it into hardware and software, where the software is further split into application software and the operating system. Applications do not directly access the hardware; they do so via a common interface provided by the operating system.

Of course, many tasks performed with computers today actually involve multiple computers communicating. Those computers communicate across a network.

In the following we will see some more detail and examples of operating systems, applications and networks (specifically, the Internet).

Operating Systems

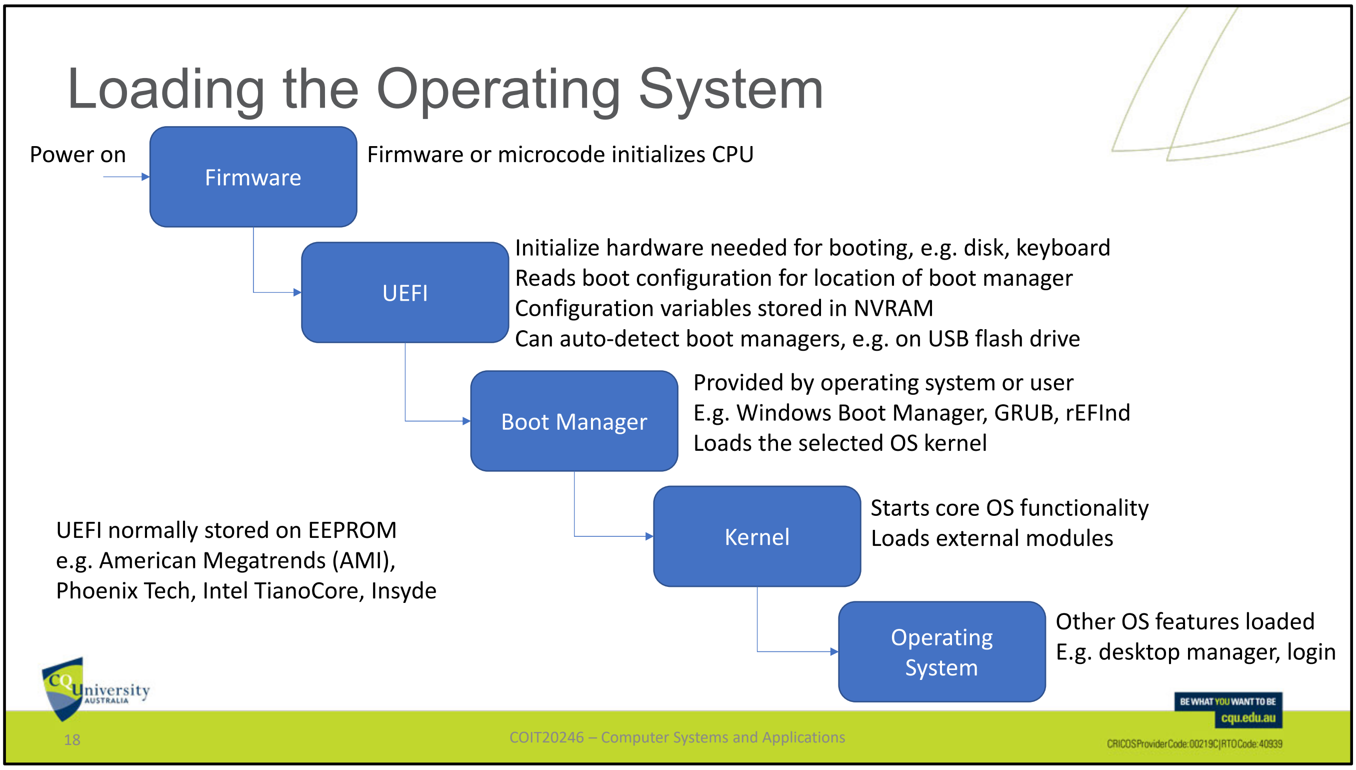


For the examples, following the links to see detailed pictures and further details. To see a list and timeline of 800+ operating systems, see https://eylenburg.github.io/os_familytree.htm.

- Windows NT: https://learn.microsoft.com/en-us/windows-hardware/drivers/kernel/overview-of-windows-components
- Linux:
    - Linux Kernel Diagram (simple): https://makelinux.github.io/kernel/diagram/
    - Linux Kernel Map (interactive): https://makelinux.github.io/kernel/map/
    - Kernel source code: https://kernel.org/
- macOS:
    - XNU code: https://github.com/apple/darwin-xnu
    - Kernel Programming Guide: https://developer.apple.com/library/archive/documentation/Darwin/Conceptual/KernelProgramming/

Common Features of OS Kernel

- Memory management: manage how software accesses physical memory
- File system: files and directories stored on physical storage
- Processing: scheduling software to be executed on the CPU
- Device management: drivers and interfaces for keyboards, mouse, displays, audio, USB, …
- Networking: drivers for network devices; implement network protocols
- Interfaces: present standard interfaces to application software to use OS features



This is a simplified view of the boot process for common operating systems. This illustrates the UEFI process. Prior to UEFI the BIOS process was used (and is still used in some legacy systems today).

Applications

Types of Applications

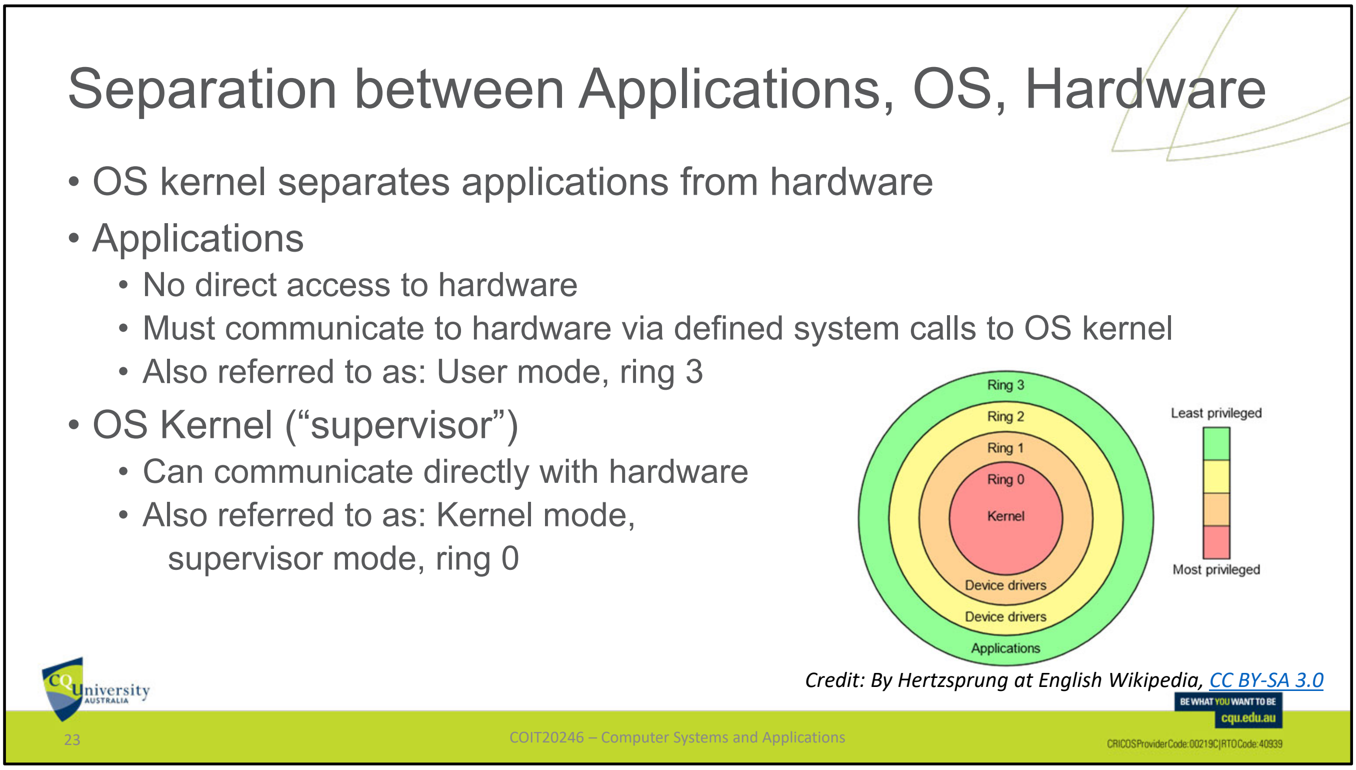
- Standalone vs Networked Applications
  - Standalone: run on single computer, network access not needed for primary operation
  - Networked: primary operation of application depends on multiple devices communicating across network
    - Client/Server model: server provides services to client; asymmetrical; usually request/response; centralized approach
      - Web application: client = web browser; server = web server(s)
    - Peer-to-peer model: devices play roles of both client and server; distributed approach
- Application Requirements
  - Real-time vs Non-real-time Applications
    - Real-time: sender/receive communicate as if they were at same physical location
  - One-way vs Two-way communications
    - Streaming from server to client; vs interactive communications

Our focus is on networked applications.

Application Performance Requirements

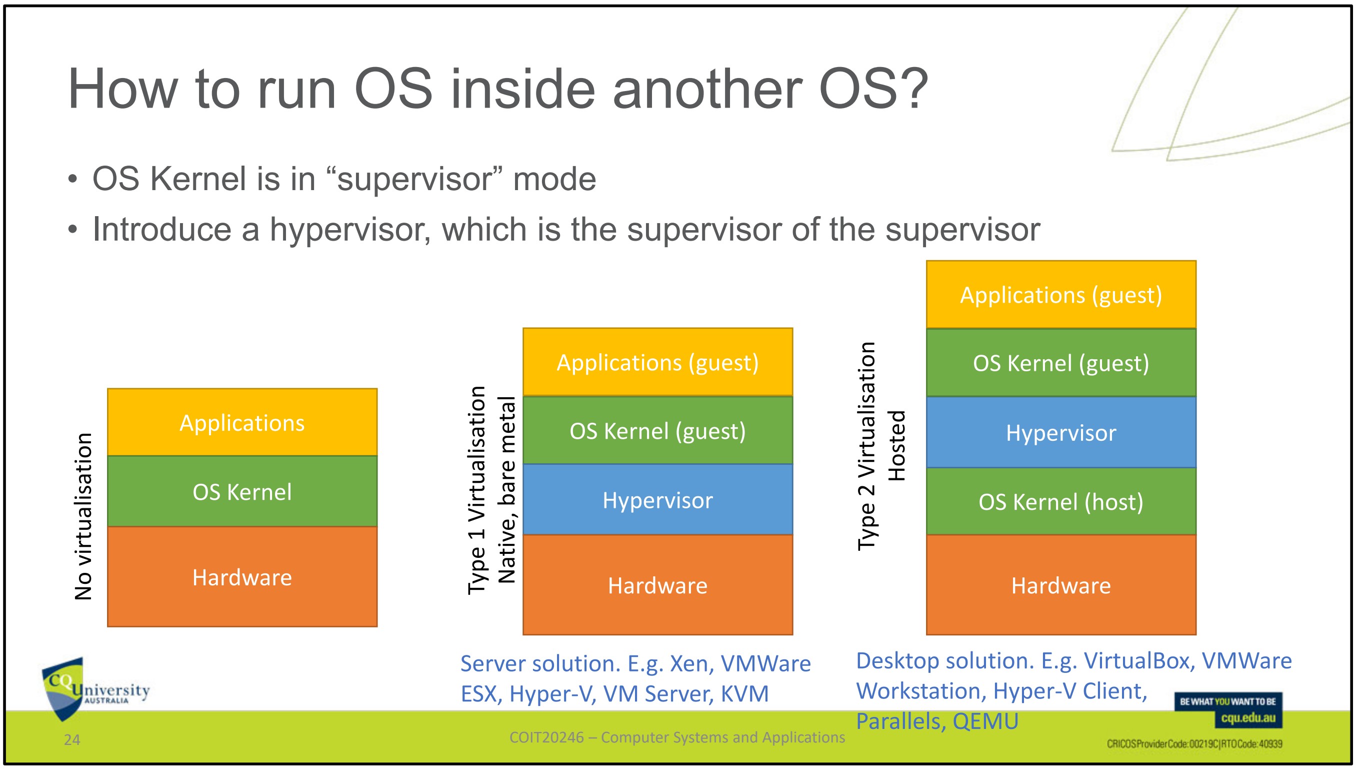
- Throughput
  - Rate at which data is delivered to destination, i.e. bits per second
    - Also referred to as goodput
- Error rate
  - Error: not all data arrives correctly at destination
  - Multimedia applications can tolerate some errors; human user may not notice reduction in quality of playback
  - Data applications
- Delay
  - Time for message to travel from source to destination
  - Interactive applications have tight delay requirements, e.g. <400ms in voice call may be unusable
  - Streaming applications can tolerate large delay using buffers
    - Related: jitter = variation of delay.

Virtualisation



Most operating systems logically separate software into layers or rings. The software in each ring has similar levels of privileges. The typical 4 ring model has the highest privileged software, the OS kernel at ring 0. It has access to all hardware and other software. The least privileged software are end-user applications, which can only access the hardware view the inner rings. This provides a level of security. In this example, the inner rings are for device drivers: less privilege than kernel, but more than applications. There are variants of this ring model today. It is not important to know the ring model, but to be aware when others refer to virtualization approaches, they may refer to "rings".

The OS kernel, as the most privileged software, is often referred to as a "supervisor".



Now consider how can we run one OS inside another, or more generally, run multiple OS on one people of hardware. One approach is with virtualization. We introduce a new component that will be the supervisor of the "guest" OS we want to run. As this new component is a supervisor of a guest OS kernel supervisor, it is referred to as a hypervisor.

The two common approaches to virtualization with a hypervisor are shown.

In type I virtualization, the hypervisor runs directly on the hardware. Multiple guest OS can then run on the hypervisor. An example where a type 1 hypervisor is used is a server computer for a company that will multiple server applications on different operating systems, e.g. a web server on Linux, an Outlook/Exchange email server on Windows Server, a file server on FreeBSD. The hypervisor runs directly on the server computer, as the server computer has the sole purpose of running multiple operating systems.

In type 2 virtualization, the hypervisor runs as an application on a host OS. For example, if you have Windows 11 installed on your personal (desktop or laptop) computer, you can then run one or more other operating systems at the same time via the hypervisor. This will allow you to test other operating systems or to use specialist software, while still using the host OS. There may be multiple guest OS running.

We haven't considered performance. In general, the more guest OS running at the same time, the more hardware resources are required (especially RAM).

Virtualisation Benefits

- Advantages
  - Do not need separate computers to run
  - Can scale up faster (easy to add another guest)
  - Can be more efficient use of hardware resources
- Disadvantages
  - Performance: sharing resources across multiple guests
  - High initial investment of time for learning/setup
  - Potential security leakage across virtualised systems (on the same host)
  - Failure of hardware may impact multiple guests
  - Cost of licensing hypervisor

Later we will see more details of pros/cons for virtualisation in the cloud

Representing Information in Binary

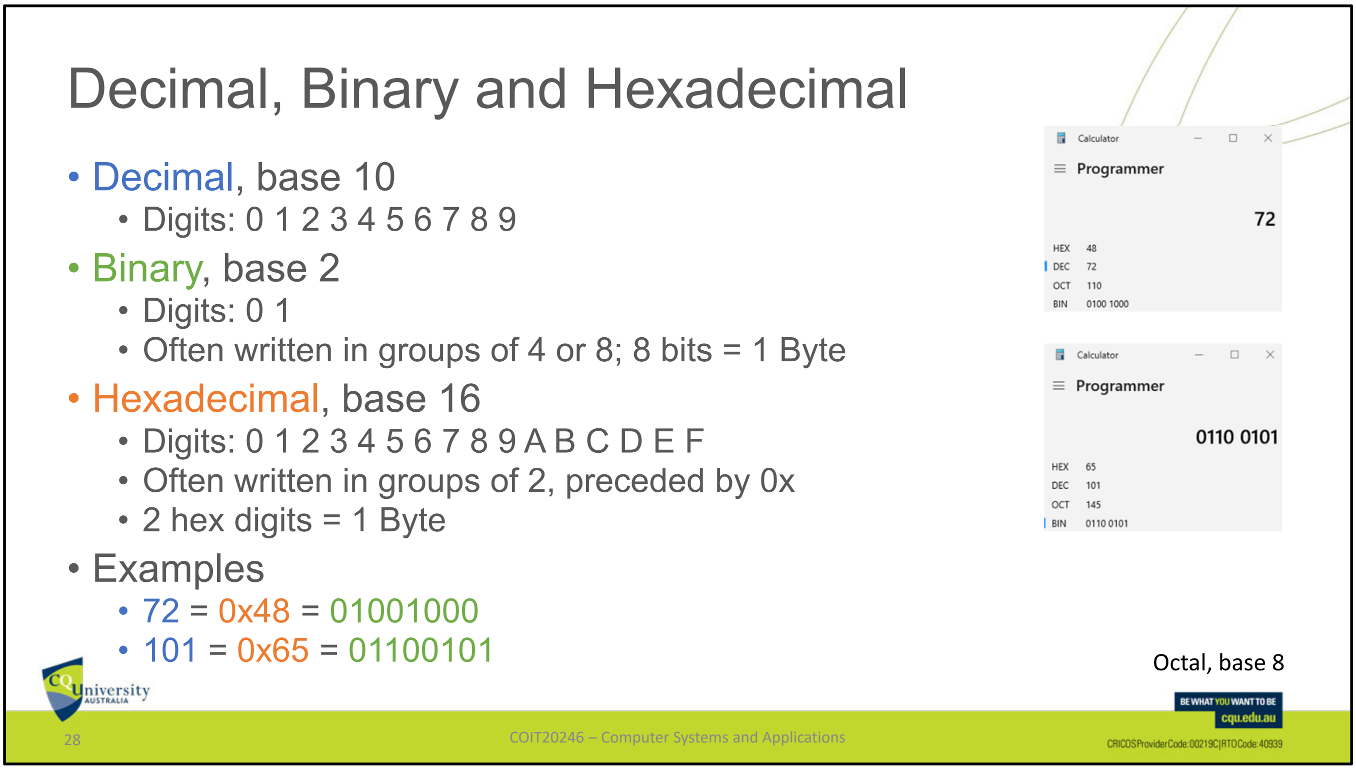
Information Represented as Bits

- Digital computers and communications operate on binary digits or bits (2 possible values, 0 or 1)
  - Text, documents, images, applications, audio, video, sensor data
  - Different encodings to map these data into binary
- In this unit, 1^{st} bit is left most bit and most significant bit
  - Example: 01001111
  - 0 is 1^{st} bit and 1 is 8^{th} bit
  - 2^{nd} bit has value 64 in decimal and 8^{th} bit has value 1 in decimal

CQUniversity AUSTRALIA

Modern computer and communications systems represent information as binary digits. You should have covered different numbering systems (binary, decimal) in previous studies, if not, visit the review/practice material on Moodle.

Note that sometimes binary numbers can be written most significant bit first or least significant bit first. For simplicity in this unit, we will always use most significant bit at the left most position. That is read left to right, as you do with decimal numbers.



Three number systems we will commonly use are decimal, binary and hexadecimal. You should have covered these in high school, so we will not cover in any depth here. We assume you can convert between them. Use a calculator! For example, the Windows Calculator app has a "Programmer" mode which allows you to enter values in any of the number systems, and it will show you the value in the others.

We will not use octal much in this unit: it is a base 8 number system, i.e. digits 0 1 2 3 4 5 6 7. It maps directly to a Byte.

Encoding Text: ASCII (Decimal)

ASCII: American Standard Code for Information Interchange
(Also referred to as ANSI encoding when 8-bit values)
IRA: International Reference Alphabet (subset of Unicode)

0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	‘	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	’	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	–	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

29 COIT20246 – Computer Systems and Applications

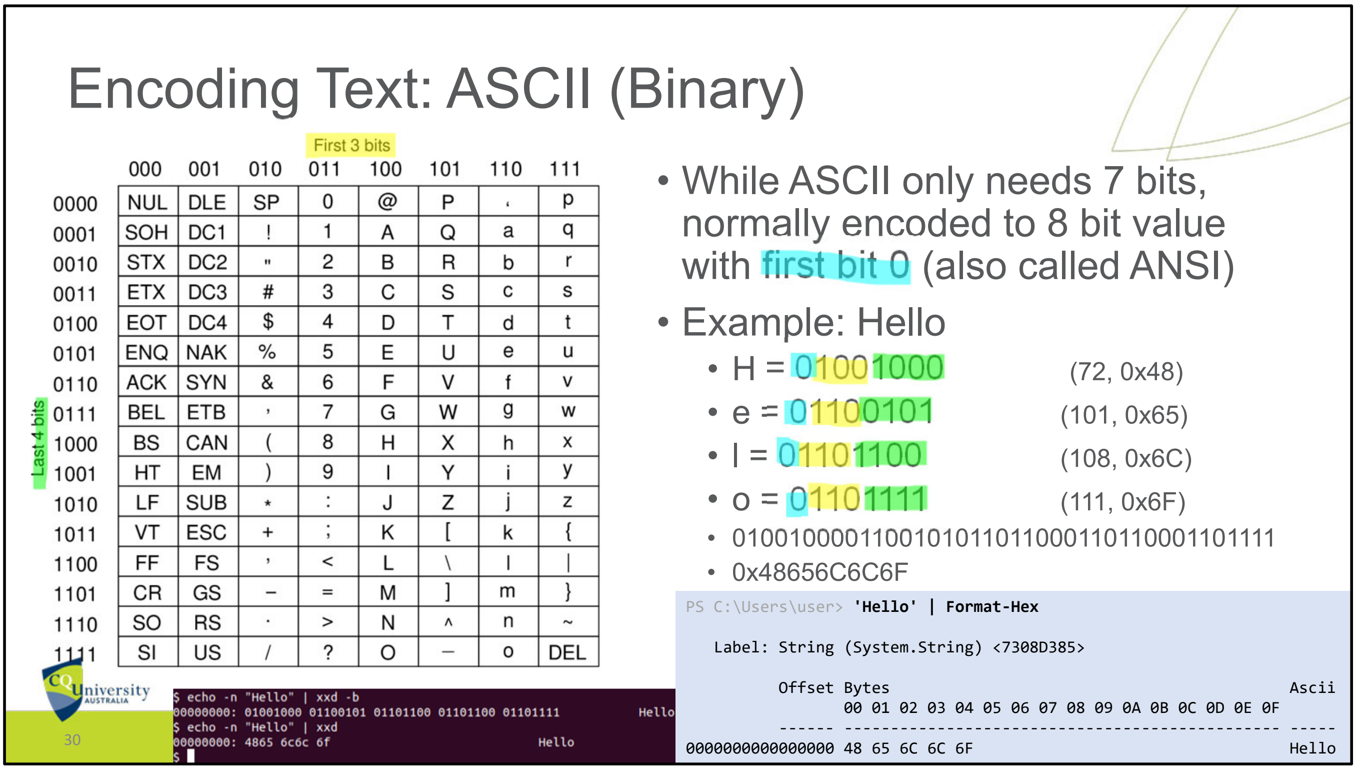
BE WHAT YOU WANT TO BE cqu.edu.au

CRICOS Provider Code:00219C|RTO Code:40939

We commonly want to encode English characters into binary. ASCII encoding is a popular approach. Unicode was later developed to include additional characters, especially from other languages. ASCII is a subset of Unicode and also referred to as the International Reference Alphabet. We still commonly just called it ASCII.

ASCII maps 128 characters into 7-bit binary values, or the decimal values 0 to 127. The first table shows the decimal values, and the next table (next slide) shows the binary values.

ASCII includes uppercase English characters, lowercase English characters, digits, punctuation characters, as well as codes/keys commonly used (old) computers. There are 94 printable characters, some other characters that can be entered by a keyboard (e.g. space, escape, delete), and others that are control characters for devices (e.g. for printers). For example, the Escape key is encoded to decimal 27 or binary 0011011.



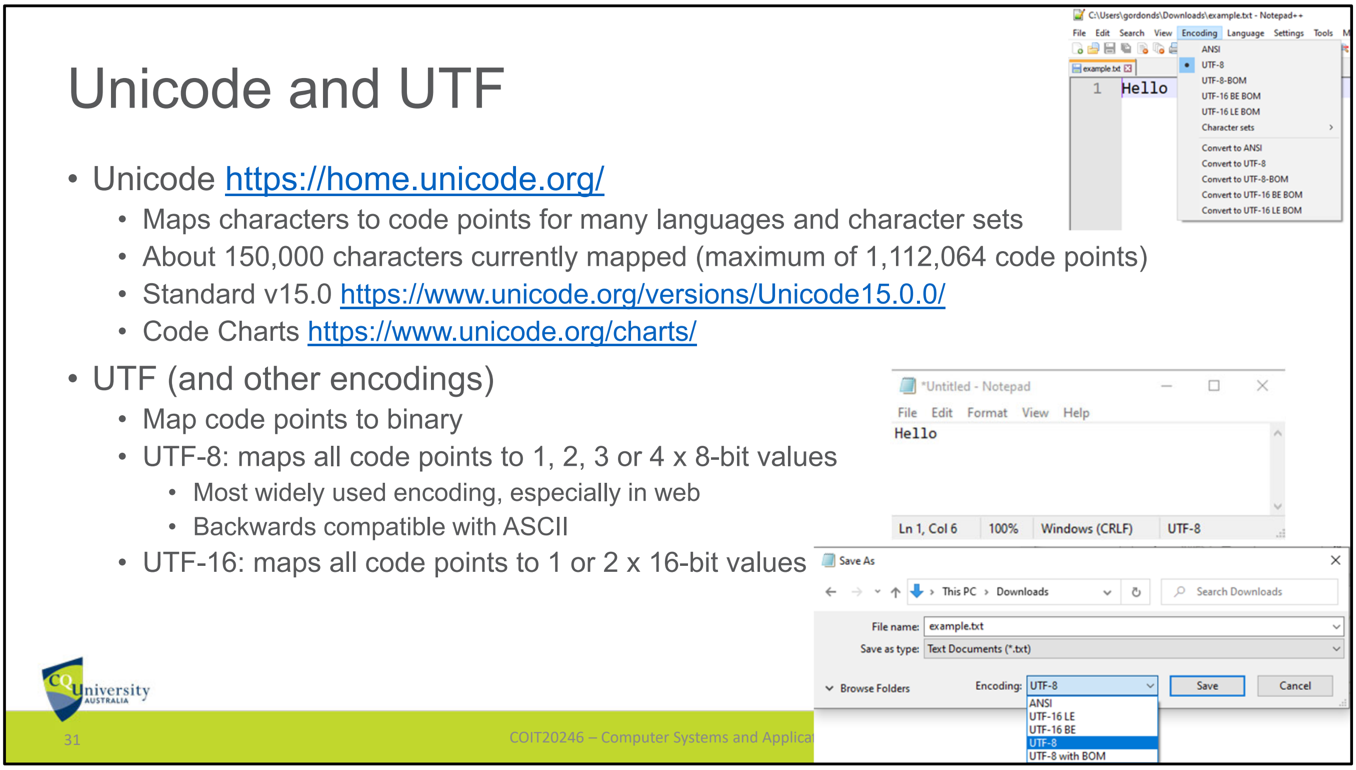
While ASCII only has 128 values and only requires 7 bits, in modern computer systems a single ASCII character is typically encoded to an 8-bit value, with the first bit is always 0. We will use that 8-bit encoding in examples, as that is what we will see when looking at real files.

The example shows the ASCII encoding of the string “Hello” into a 40-bit binary value. Next to each letter in the example, the corresponding decimal and hex values are shown in parentheses.

The examples at the bottom show the output from commands in Linux Bash and Windows PowerShell that confirm our calculation.

In PowerShell the module Format-Hex will take a string or file input and show the hex values with 8-bit ASCII encoding. It doesn’t show the binary values.

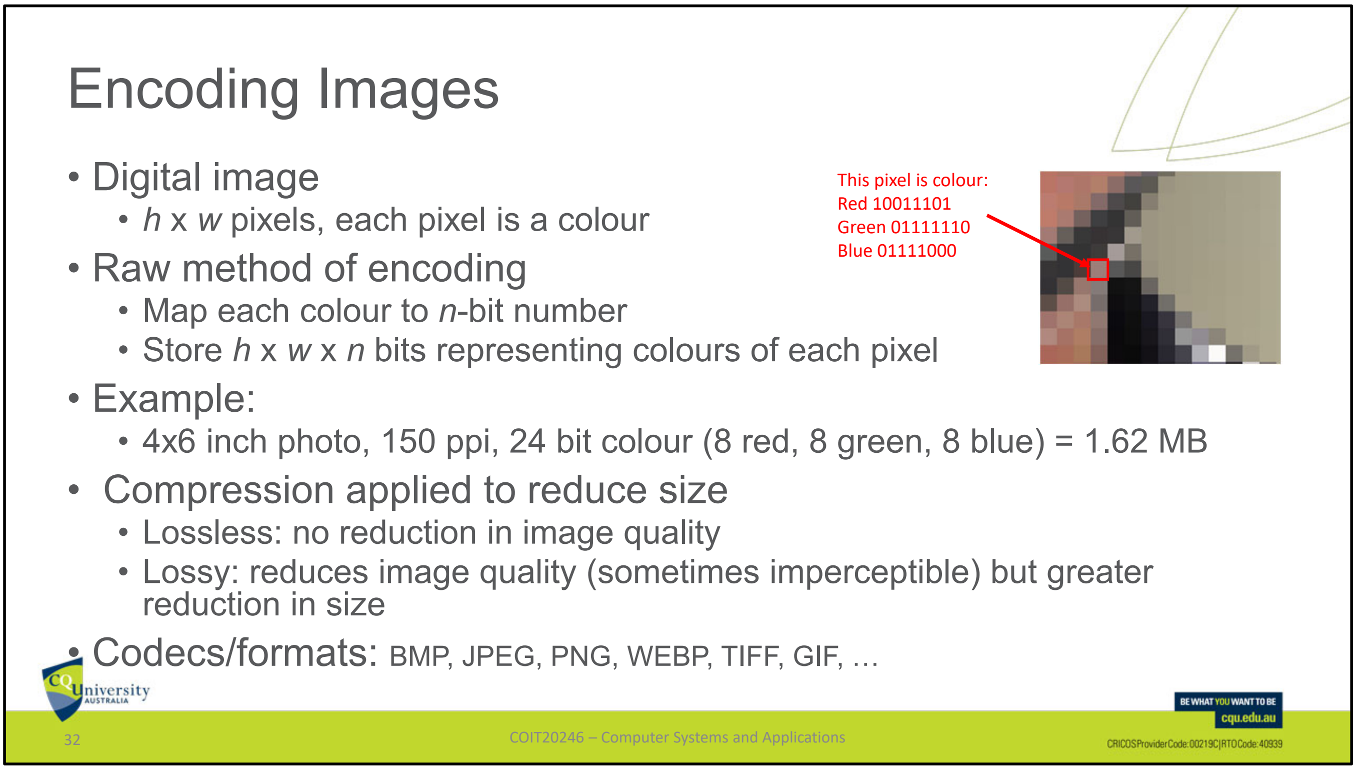
In Bash the command xxd will show either the binary values (using option –b) or the hex values (using no option). In the example, a string is produced using the echo command.



ASCII was developed in the 1960's in America. Unicode was developed in 1990's by computer hardware and software companies to cover a wide number of languages and scripts. There are two practical parts:

1. Unicode defines how characters from different languages/scripts map to unique code points (numbers, usually given in hex).
2. Encodings, such as UTF-8 and UTF-16, define how the code points are mapped to binary.

When you save a text file in Windows Notepad (or other text editors), a default encoding will be used, e.g. UTF-8. However you may be able to select an alternative encoding. Notepad++ allows changing/converting encoding via the menu bar.



Here we present a brief overview of common digital images, like photos. (We don't cover vector graphics, i.e. where mathematical representations of lines are stored).

The example uses a 4x6 inch photo, which is about 10cmx15cm. We assume 150 pixels per inch (ppi). That means 600 pixels x 900 pixels, or a total of 540,000 pixels. Each pixel is a single colour, represented by a 24 bit number. Hence there are 540,000 x 24 bits = 12,960,000 bits. As there are 8 bits per Byte, this is 1,620,000 Bytes.

The process of converting an image (or other information) to binary is called encoding, while from binary back it is called decoding. Together, an encoder and decoder is called a codec.

You do not need to understand how codecs work. There are many different ones, with the main tradeoff between:

- Reducing the resulting number of bits, i.e. the fewer the bits the better
- Maintaining the quality of the source, i.e. the less quality degradation the better
- Reducing time for the codec to do processing, i.e. the less time the better

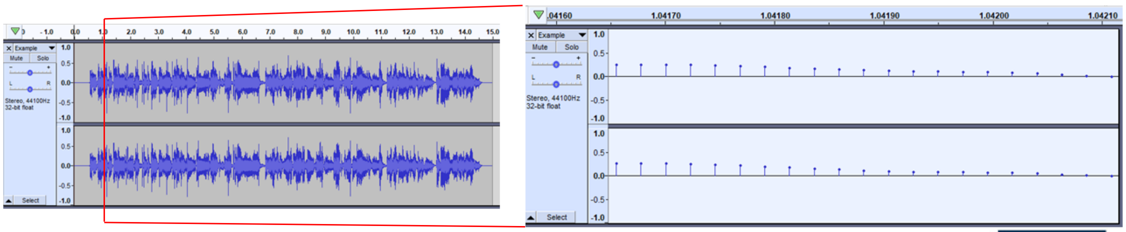
(There are also other issues, like whether the codec algorithm is free to use, without patent licensing, and whether it is widely implemented in software and hardware).

Credit: The image is part of a photo of a staff member in a Zoom meeting with headset on. It is in the public domain. GIMP was used to Zoom in, and then the colour picker used to identify the colour of the selected pixel. [Steven Gordon]

There is no one best codec. There are many available, and often it depends on the end application as to which is most appropriate.

Encoding Audio

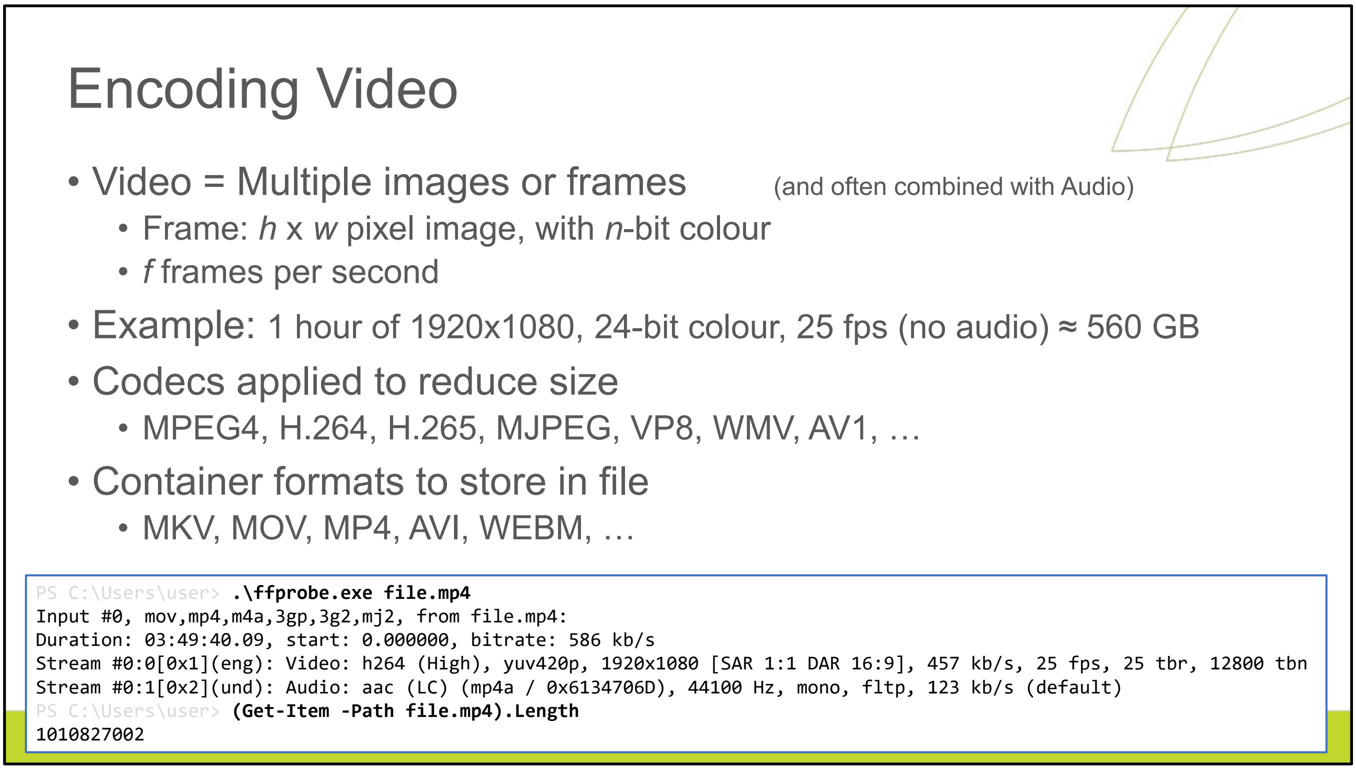
- Analog waveforms are converted to/from binary using different encoders/decoders (or codecs)
  - May be uncompressed or compressed
  - May be lossless or lossy
- Tradeoff: number of bits vs audio quality (vs processing time)
- Examples: PCM (.wav), FLAC, Dolby Digital (AC3), G.711, MP3, AAC, WMA, …



The picture at the bottom shows a 15 second piece of audio (from a Zoom meeting). The two waveforms are for the left and right channel (recorded in stereo). They represent the audio amplitude (strength) over time. A codec converts the original audio input (waveform) into binary. There are different approaches, with the most basic approach measuring the audio amplitude at fixed time intervals, and mapping that amplitude to a discrete value on some predetermined scale. For example, a common approach for high quality audio (CD quality) is to sample 44100 times per second (44.1kHz) and map each measured value on a scale of 0 to 65,535 (i.e. a 16-bit value). That is represented as a 16-bit binary number.

So 15 seconds of audio, sampled at 44.1kHz, with 16-bit samples, would require 1.323 MB of storage. This is using the Pulse Code Modulation (PCM) codec, which is often saved as a .wav file.

Other codecs, like MP3 and AAC, apply compressed so the file size is even smaller. However some of those codecs may lose some of the information, so the reproduced audio is not exactly the same as the original audio. These are called lossy codecs (as opposed to lossless codecs). However good lossy codecs will make it difficult for a human listener to tell the different between the original audio and that reproduced.



A movie, TV show or Zoom recording normally contains two streams: a video stream with moving pictures, and an audio stream. The audio stream is created as in the previous slide.

A video stream at the basic level is a set of images changing frequently. Those images are referred to frames, and the rate of change is the frame rate, measured in frames per second. Common frame rates at 25 fps and 30 fps.

Codecs are applied to compress the video stream, taking advantage in compression techniques applied in images, as well as the (lack of) change of pixels across frames.

Note that it can get confusing between the difference between codecs and containers (as some are named similarly, and often we see the container name in the file extension). Containers are just a way to store the video stream compressed by a codec in a file.

The example at the bottom is the result of running FFProbe on a video file in PowerShell (some parts of the output have been removed to save space). FFProbe is part of the free, open source, widely-used video processing software FFMpeg (https://ffmpeg.org/). It shows information about the file, specification that its duration is 3 hrs 49 minutes, the video is encoded with H.264, 1920x1080 resolution and 25 fps. The audio is encoded with AAC, with 44.1 kHZ sample rate and mono. The last command gets the file length: about 1 GB. Compare that to the example calculation of 560 GB for 1 hour of non-compressed video, and you can realise significant gains can be made with compression (good codecs).