

# History, VMs, 4 Main Concepts

2 November 2025  
Lecture 2

Slides adapted from John Kubiawicz (UC Berkeley)

# Main concepts from last time

---

Operating  
system

Many core

Process

Program

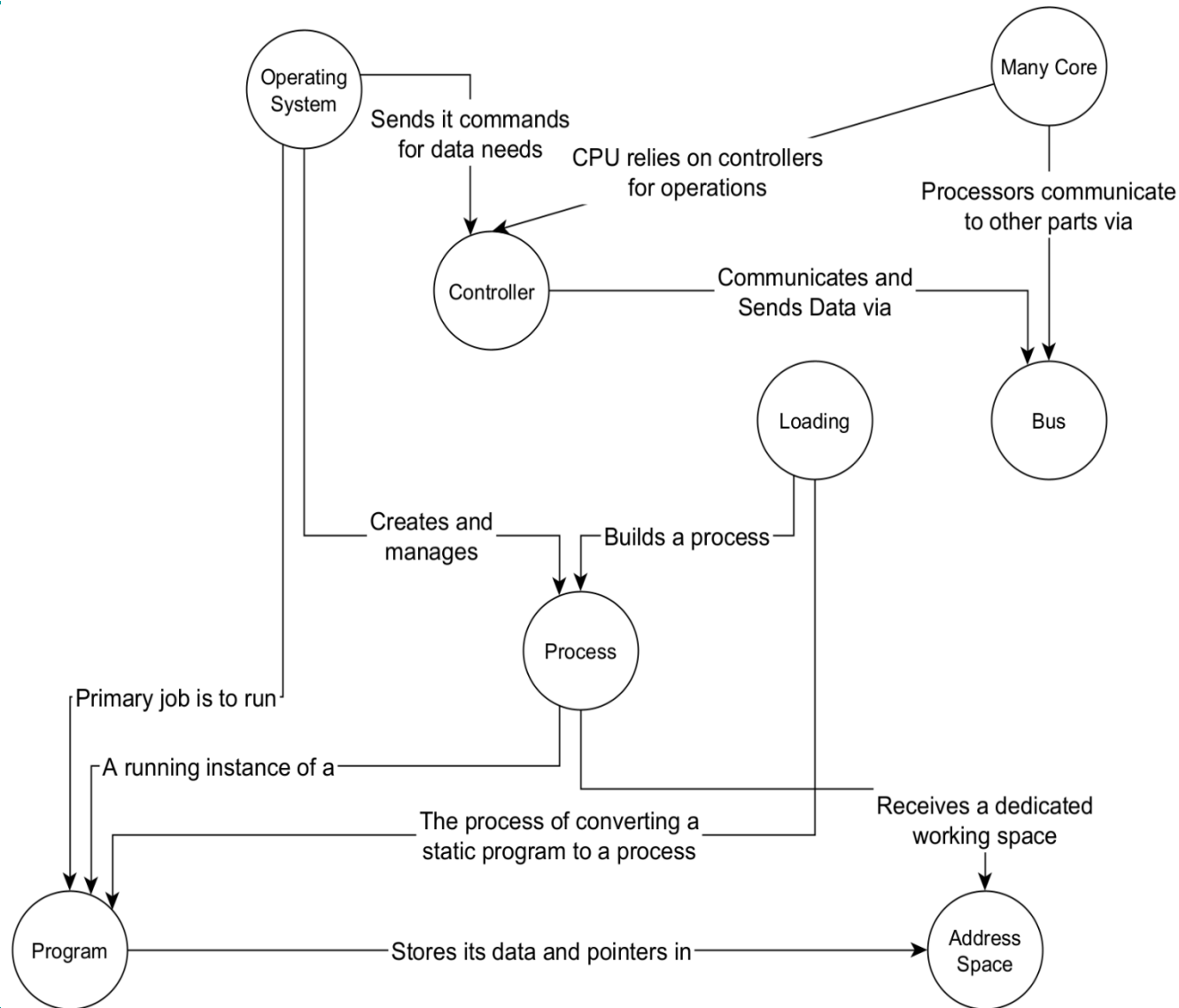
Loading

Address  
space

Controller

Bus

# Main concepts from last time



# Topics for Today

---

- (Brief) OS History
- Virtual Machines
- 4 Main OS Concepts
  - Thread
  - Address
  - Process
  - Dual mode

# Today's concepts

---



# Very Brief History of OS

---

- Several Distinct Phases:



Hardware Expensive,  
Humans Cheap

- Eniac, Multics

Hardware Cheaper,  
Humans Expensive

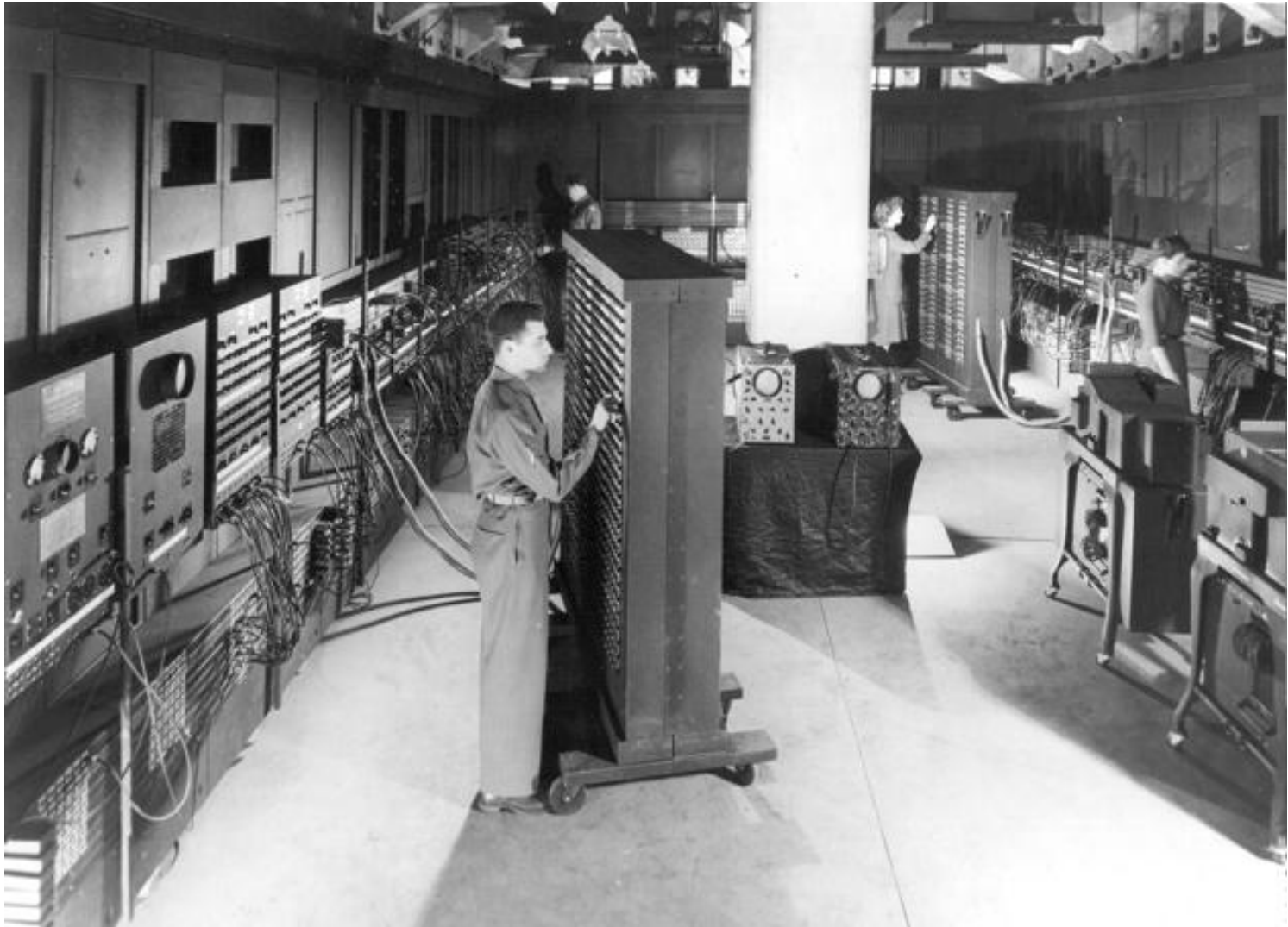
- PCs
- Workstations
- Rise of GUIs

Hardware Really  
Cheap, Humans  
Really Expensive

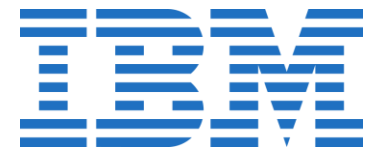
- Ubiquitous devices
- Widespread networking

# ENIAC (source: US Army)

---



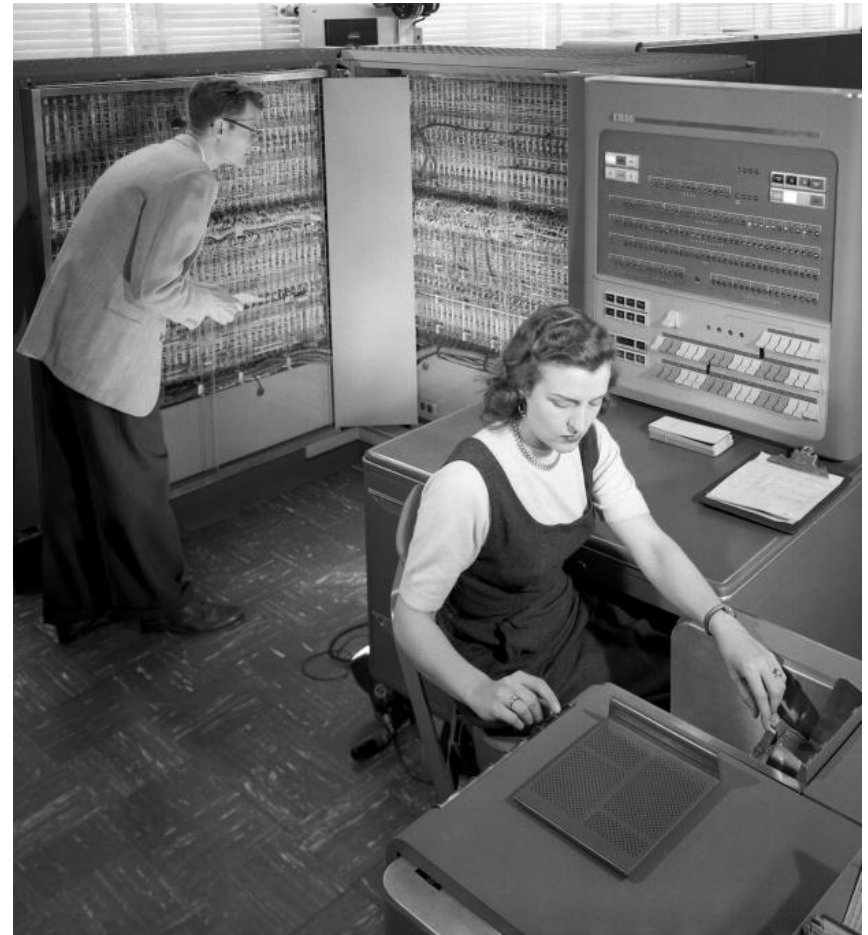
# IBM Beginnings



***"I think there is a world market for maybe five computers"***

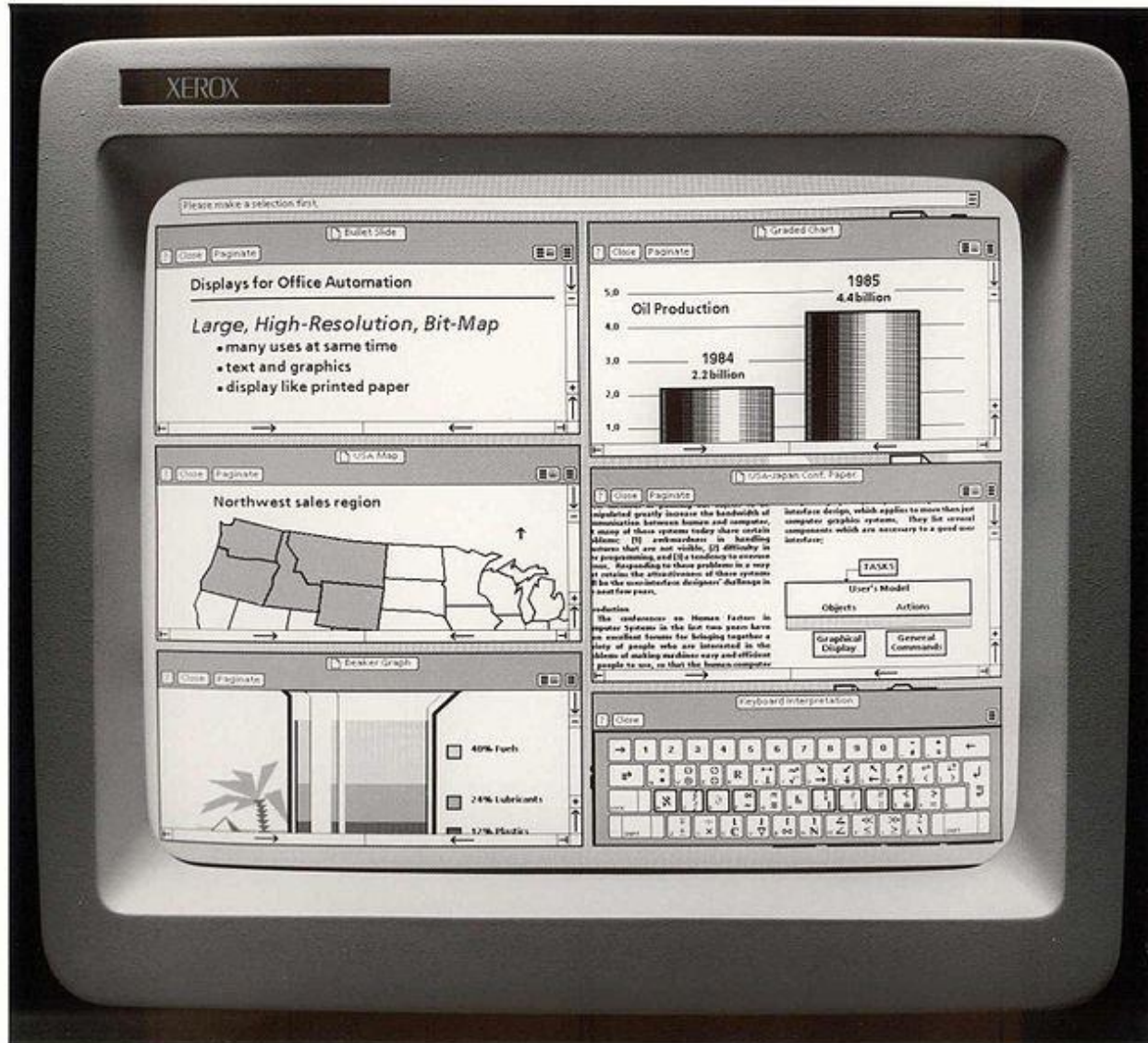
- Attributed to Thomas J. Watson in 1943
- Probably never said, but in 1953, IBM assumed only 20 prospective companies could buy the IBM 701

**IBM type 704 (source: Langley NACA)**





# The First GUIs: XEROX



# Today: Hand held

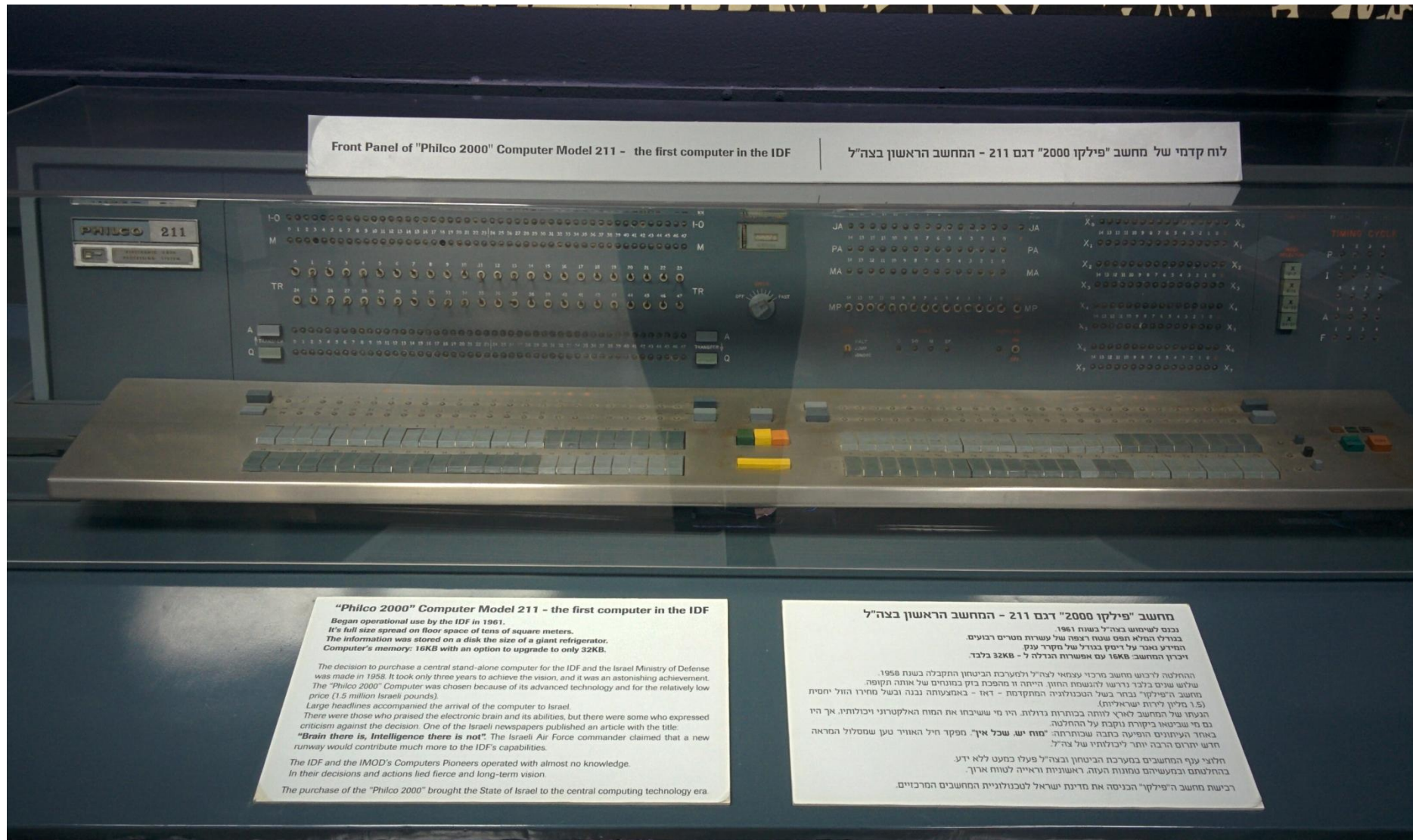
---



Image source: Blue Ion  
<http://www.blueion.com/blog/2014/01/07/too-big-to-ignore/>



# First computer in IDF



# First computer in IDF

## מחשב "פילקו 2000" דגם 211 - המחשב הראשון בצה"ל

נכנס לשימוש בצה"ל בשנת 1961.

בגודלו המלא תפס שטח רצפה של עשרות מטרים רבועים.

המידע נאגר על דיסק בגודל של מקרר ענק.

זיכרון המחשב: 16KB עם אפשרות הגדלה ל - 32KB בלבד.

ההחלטה לרכוש מחשב מרכזי עצמאי לצה"ל ולמערכת הביטחון התקבלה בשנת 1958.

שלוש שנים בלבד נדרשו להגשמת החזון. הייתה זו מהפכת בזק במונחים של אותה תקופה.

מחשב ה"פילקו" נבחר בשל הטכנולוגיה המתקדמת - דאז - באמצעותה נבנה ובשל מחירו הזול יחסית (1.5 מליון לירות ישראליות).

הגעתו של המחשב לארץ לוותה בכותרות גדולות. היו מי ששיבחו את המוח האלקטרוני ויכולותיו, אך היו גם מי שביטאו ביקורת נוקבת על ההחלטה.

באחד העיתונים הופיעה כתבה שכותרתה: **"מוח יש, שכל אין"**. מפקד חיל האוויר טען שמסלול המראה חדש יתרום הרבה יותר ליכולותיו של צה"ל.

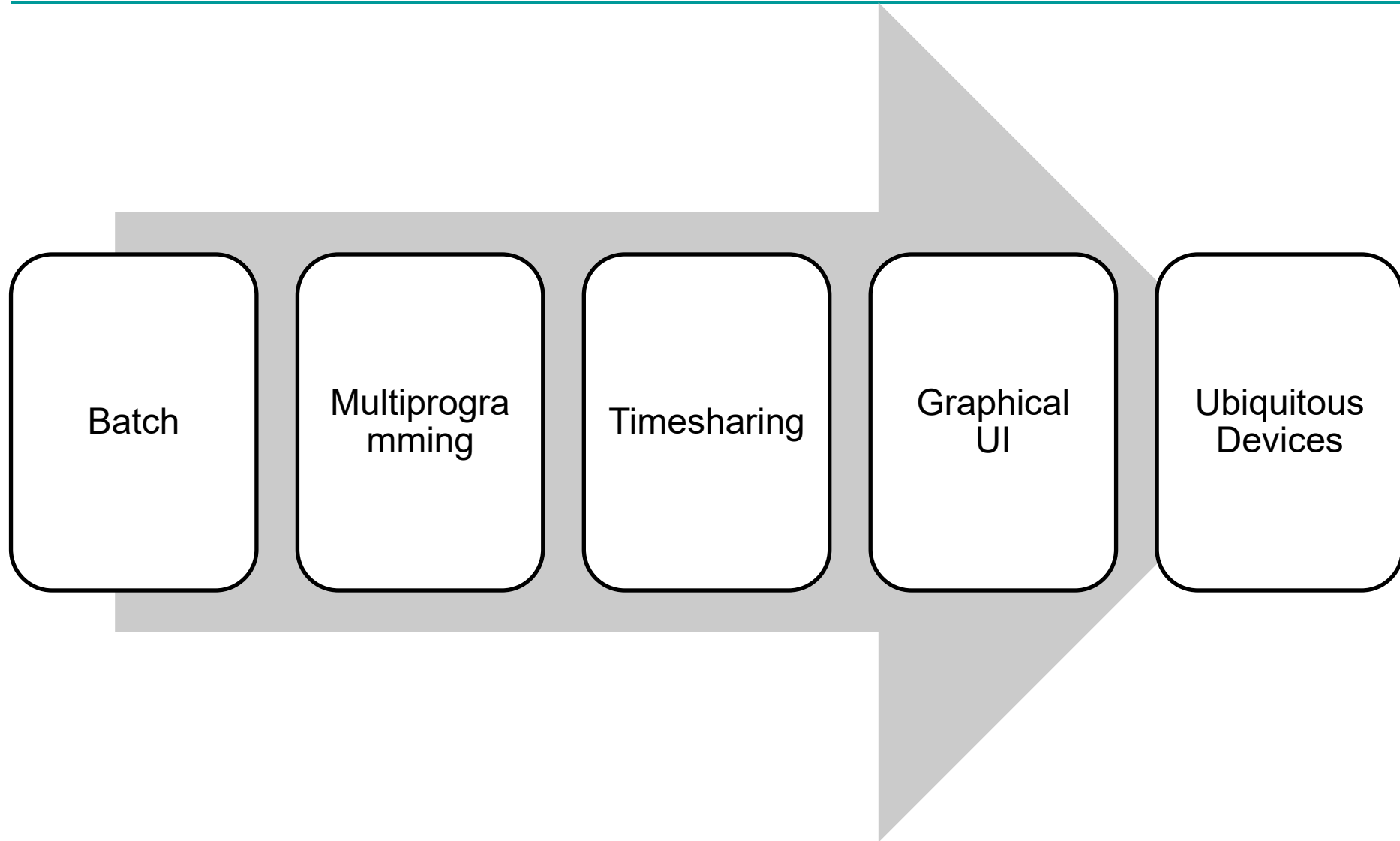
חלוצי ענף המחשבים במערכת הביטחון ובצה"ל פעלו כמעט ללא ידע.

בהחלטתם ובמעשיהם טמונות העזה, ראשוניות וראייה לטווח ארוך.

רכישת מחשב ה"פילקו" הכניסה את מדינת ישראל לטכנולוגיית המחשבים המרכזיים.

# Evolving Hardware & OS

---



# Making new OS is expensive

---

## Small OS

- 100K lines

## Large OS

- 10M lines
- 5M in a browser!

100-1000  
person-years  
of effort



# OS Archaeology

---

- Because of the cost of developing an OS from scratch, most modern OSes have a long lineage:
- Multics → AT&T Unix → BSD Unix → Ultrix, SunOS, NetBSD,...
- Mach (micro-kernel) + BSD → NextStep → XNU → Apple OSX, iPhone iOS
- Linux → Android OS
- CP/M → QDOS → MS-DOS → Windows 3.1 → NT → 95 → 98 → 2000 → XP → Vista → 7 → 8 → phone → 10 → 11...
- Linux → RedHat, Ubuntu, Fedora, Debian, Suse,...

# So Far

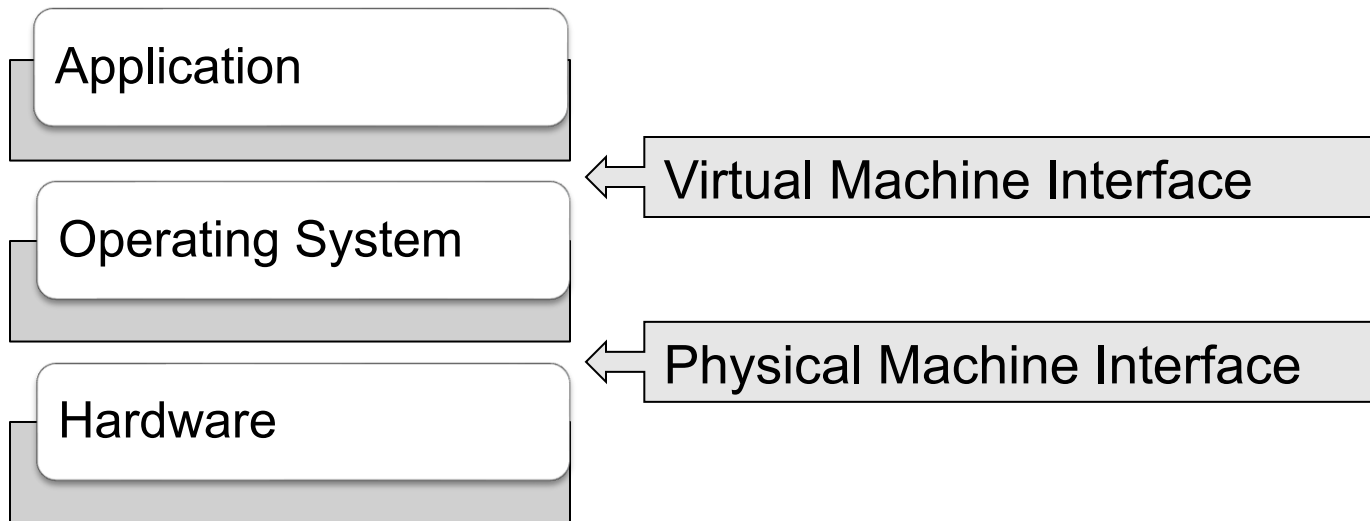
---

- (Brief) OS History
- Virtual Machines
- 4 Main OS Concepts
  - Thread
  - Address
  - Process
  - Dual mode



# Virtual Machine Abstraction

---



- Software Engineering Problem:
  - Turn hardware/software **quirks**  $\Rightarrow$  what programmers **want/need**
  - Optimize for convenience, utilization, security, reliability, etc...
- For Any OS area (e.g. file systems, virtual memory, networking, scheduling):
  - What's the **hardware** interface? (physical reality) Hardware Abstraction Layer (HAL) hides
  - What's the **application** interface? (nicer abstraction)

# Virtual Machines

---

- Software emulation of an abstract machine
  - Give programs illusion they own the machine
  - Make it look like hardware has features you want
- Three types of “Virtual Machines”
  - **Container**: Isolate processes from one another at the file system and environment variables
  - **Process VM**: supports the execution of a single program; this functionality typically provided by OS
  - **System VM**: supports the execution of an entire OS and its applications (e.g., VMWare Fusion, Virtual box, Parallels Desktop, Xen)



# Containers

- Virtual Machine light (VM--)
  - Less isolation than full VM
  - Shared physical resources (scheduler, main memory)
  - Separate parts of file system, OS libraries
- Popular for isolating applications and reducing potential conflicts



Linux containers

Home

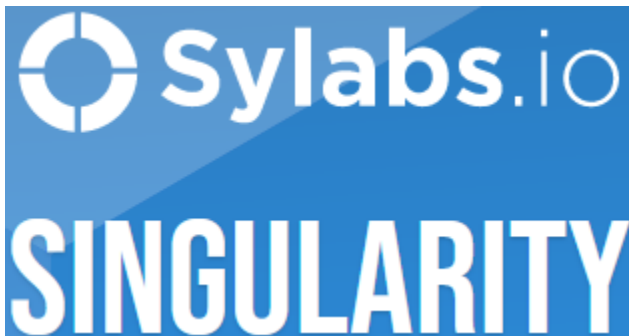
LXC

LXD

LXCFS

distrobuilder

CGManager



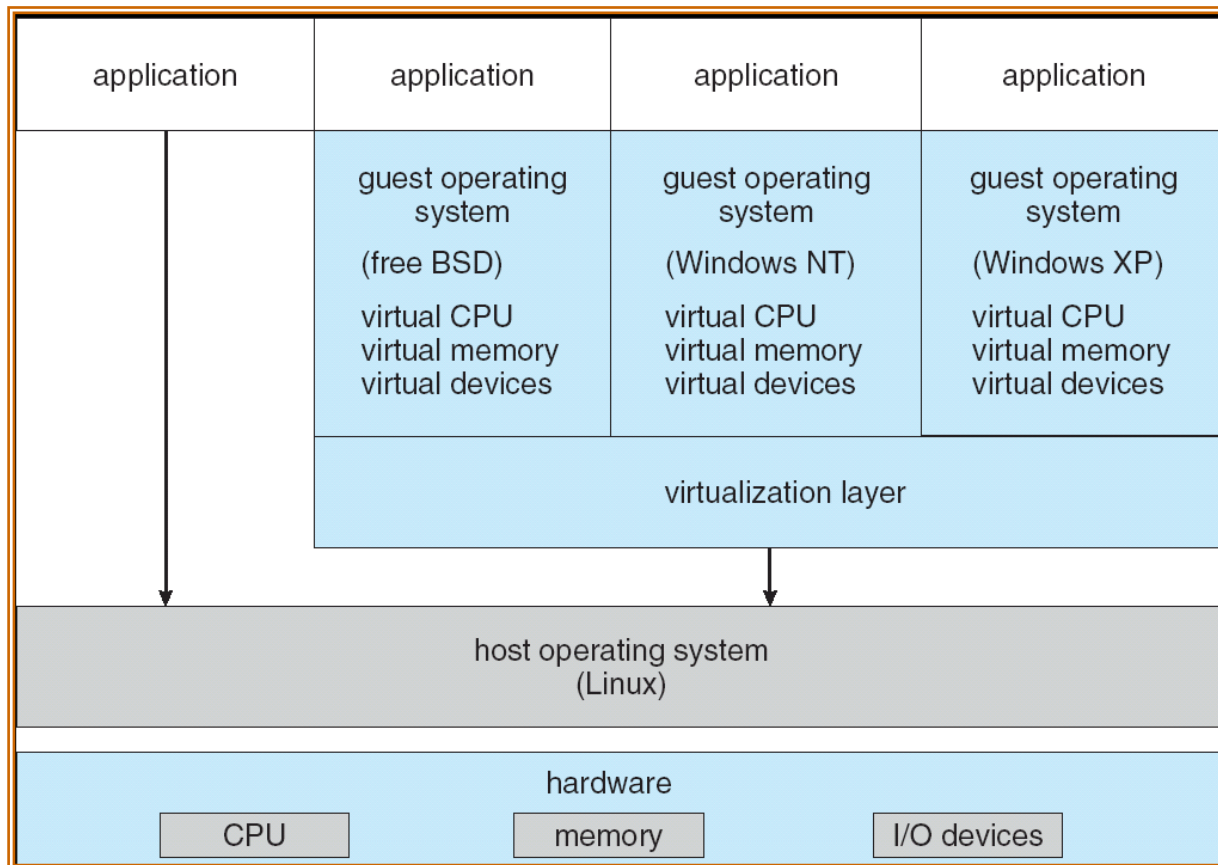
# Process VMs

---

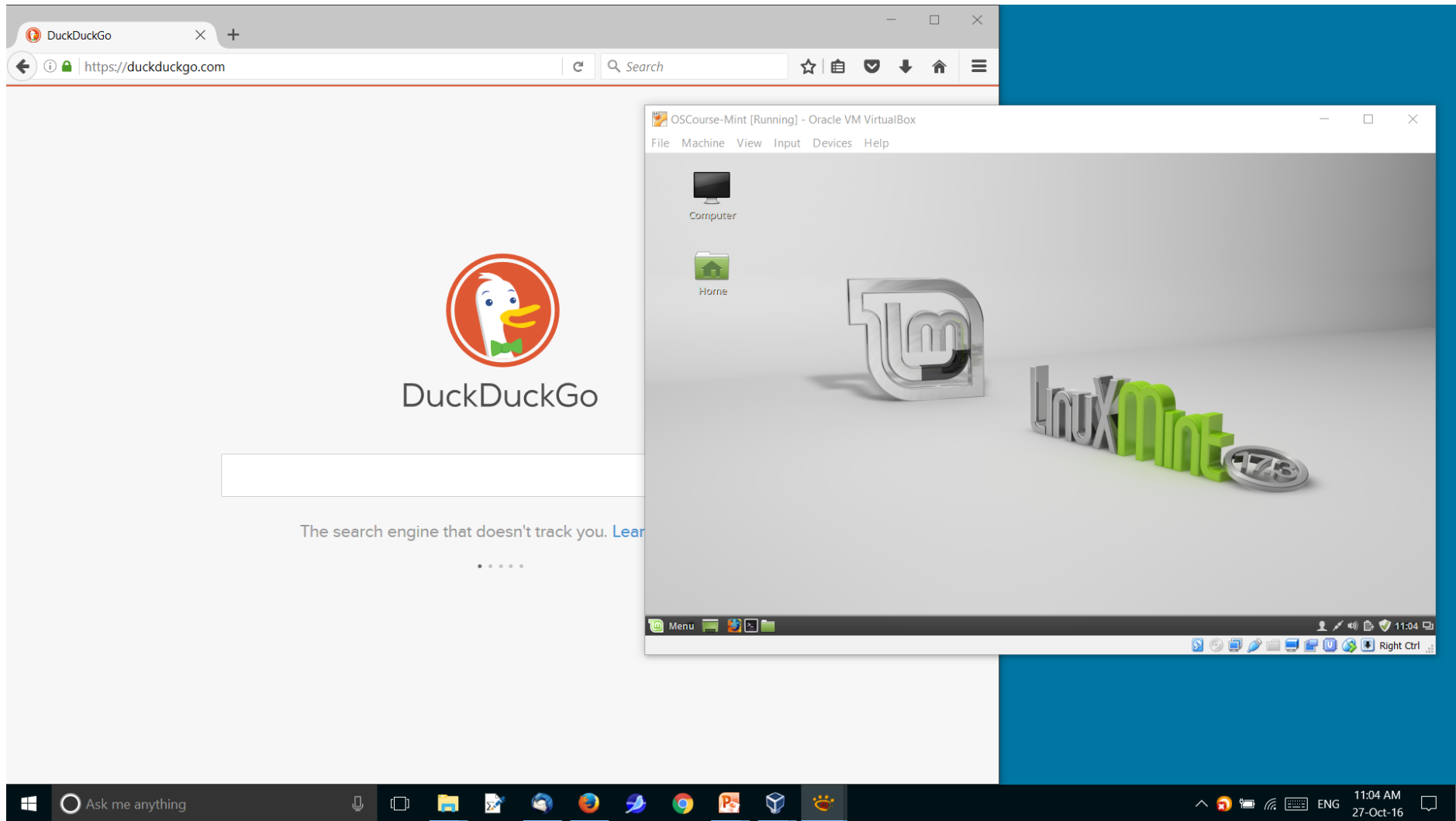
- Programming simplicity
  - Each process thinks it has all memory/CPU time
  - Each process thinks it owns all devices
  - Different devices appear to have same high level interface
  - Device interfaces more powerful than raw hardware
    - Bitmapped display  $\Rightarrow$  windowing system
    - Ethernet card  $\Rightarrow$  reliable, ordered, networking (TCP/IP)
- Fault Isolation
  - Processes unable to directly impact other processes
  - Bugs cannot crash whole machine
- Protection and Portability
  - Java interface safe and stable across many platforms

# System VMs: Layers of OSes

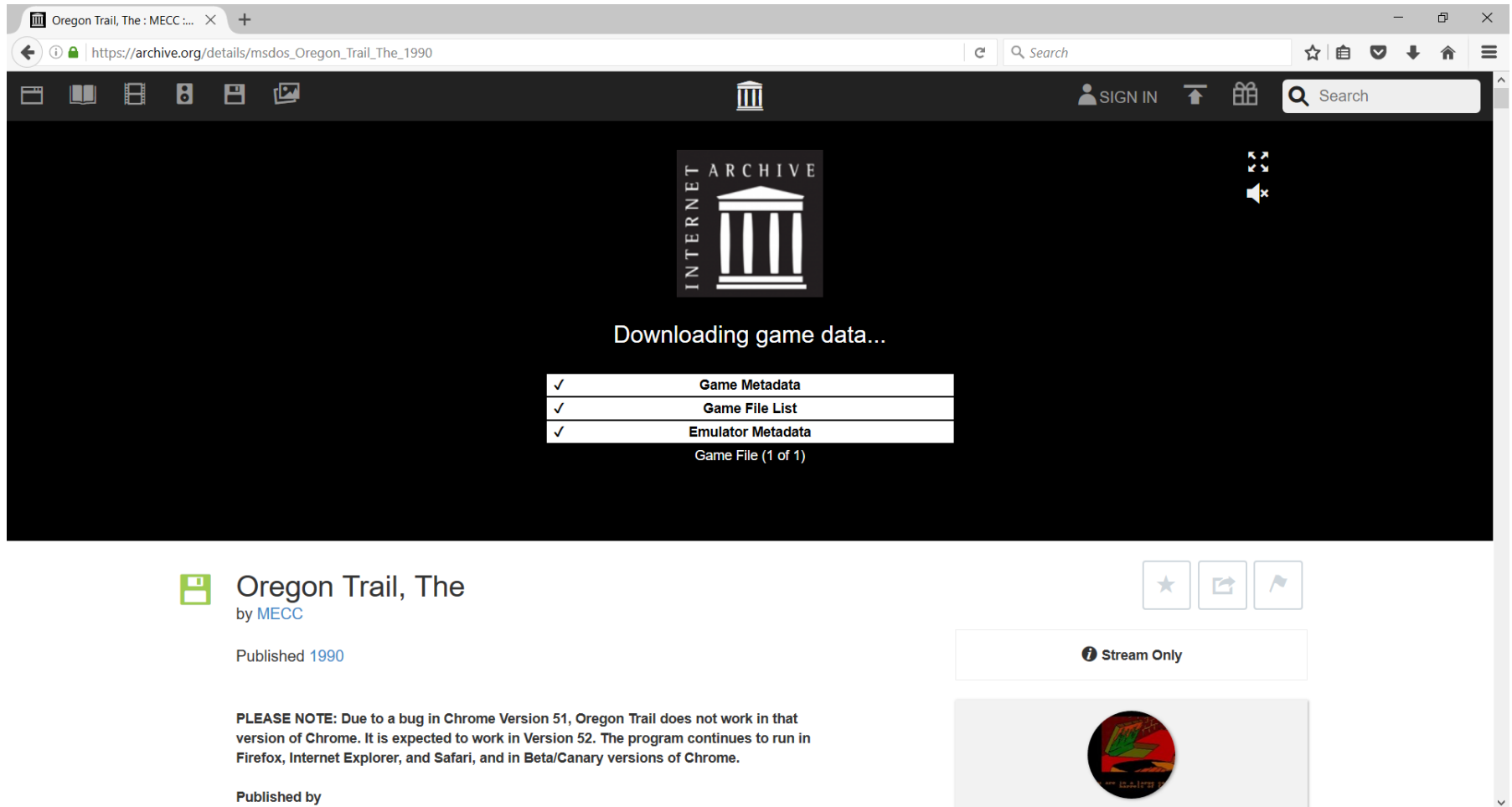
- Useful for OS development
  - When OS crashes, restricted to one VM
  - Can aid testing programs on other OSs



# How it looks



# Why not in a browser?



Oregon Trail, The : MECC : ...

https://archive.org/details/msdos\_Oregon\_Trail\_The\_1990

Internet Archive

Downloading game data...

✓	Game Metadata
✓	Game File List
✓	Emulator Metadata

Game File (1 of 1)

Oregon Trail, The  
by MECC

Published 1990

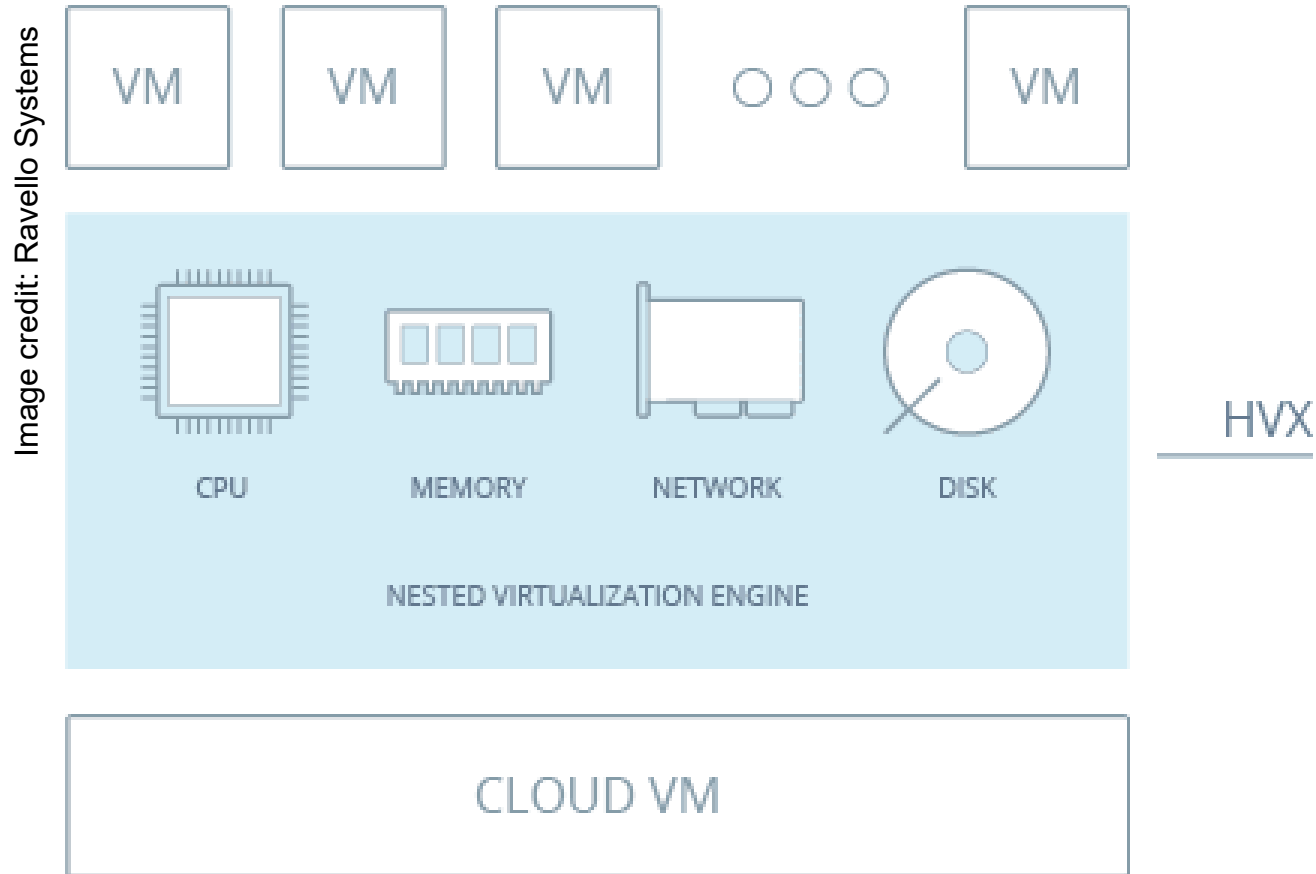
PLEASE NOTE: Due to a bug in Chrome Version 51, Oregon Trail does not work in that version of Chrome. It is expected to work in Version 52. The program continues to run in Firefox, Internet Explorer, and Safari, and in Beta/Canary versions of Chrome.

Published by

Stream Only

Of course, this is sandboxed...

# VM inside a VM



Why not run a VM inside a VM?  
Turtles all the way down...



# VMs in Action

---

Every cloud  
service provider

Shared  
hardware for  
servers

Data Centers

Rapid  
provisioning,  
scale up, and  
load balancing

# So Far

---

- (Brief) OS History
- Virtual Machines
- 4 Main OS Concepts
  - Thread
  - Address
  - Process
  - Dual mode

# What is an OS, Really?

---

## Most Likely:

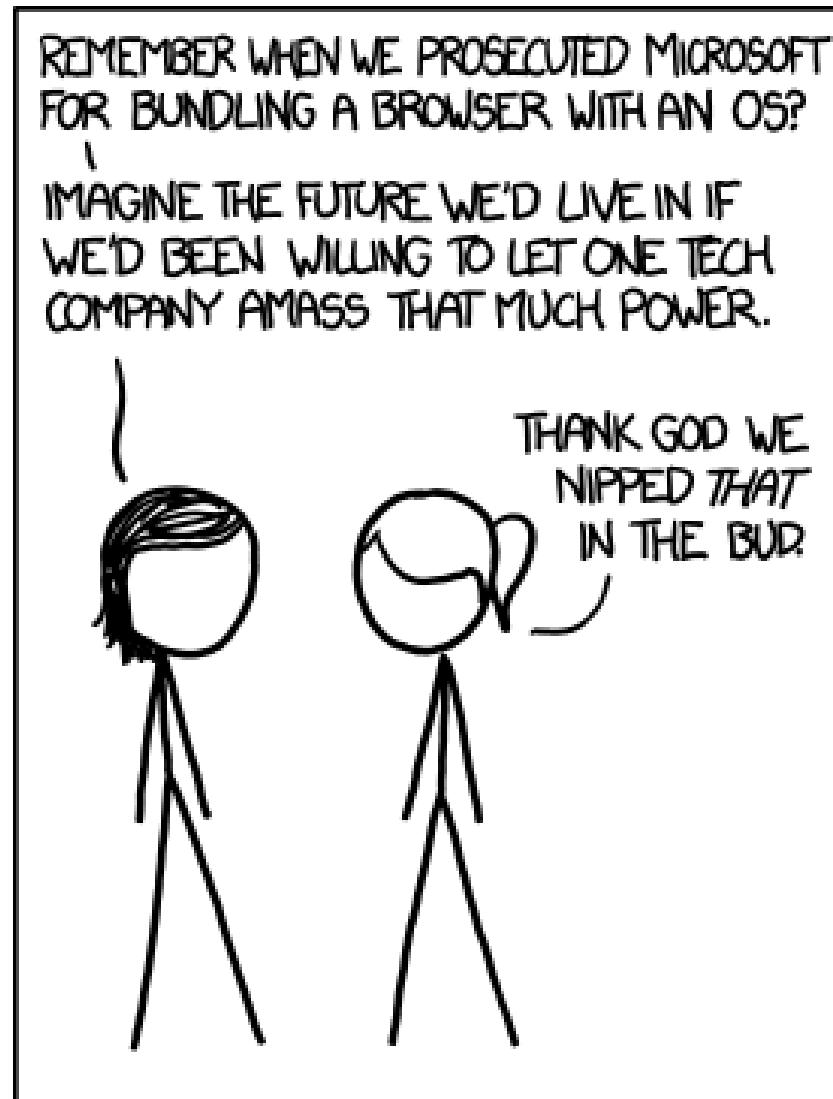
- Memory Management
- I/O Management
- CPU Scheduling
- Communications?  
(Does Email belong in OS?)
- Multitasking/multiprogramming?

## What about?

- File System?
- Multimedia Support?
- User Interface?
- Internet Browser? 😊



































Is this only interesting to academics?

# <https://xkcd.com/1118/>



# Top 10 Tech Companies 2024

Develops their own OS?

	Rank	Name	Market Cap	Price	Today	Price (30 days)	Country
<input checked="" type="checkbox"/>	1	 <b>NVIDIA</b> NVDA	\$3.621 T	\$147.63	▼ 0.84%		 USA
<input checked="" type="checkbox"/>	2	 <b>Apple</b> AAPL	\$3.430 T	\$226.96	▼ 0.12%		 USA
<input checked="" type="checkbox"/>	3	 <b>Microsoft</b> MSFT	\$3.141 T	\$422.54	▼ 0.68%		 USA
<input checked="" type="checkbox"/>	 4	 <b>Alphabet (Google)</b> GOOG	\$2.191 T	\$179.86	▼ 1.33%		 USA
<input checked="" type="checkbox"/>	 5	 <b>Amazon</b> AMZN	\$2.189 T	\$208.18	▼ 0.89%		 USA
<input checked="" type="checkbox"/>	6	 <b>Meta Platforms (Facebook)</b> META	\$1.487 T	\$589.34	▼ 0.40%		 USA
	7	 <b>TSMC</b> TSM	\$1.043 T	\$201.20	▲ 0.00%		 Taiwan
<input checked="" type="checkbox"/>	8	 <b>Tesla</b> TSLA	\$1.031 T	\$321.22	▲ 8.19%		 USA
<input checked="" type="checkbox"/>	9	 <b>Broadcom</b> AVGO	\$857.70 B	\$183.64	▼ 0.09%		 USA
<input checked="" type="checkbox"/>	 10	 <b>Oracle</b> ORCL	\$524.42 B	\$189.25	▲ 1.55%		 USA

<https://companiesmarketcap.com/tech/largest-tech-companies-by-market-cap/>

# Operating System “Definition”

---

No universally accepted definition

“Everything a vendor ships when you order an operating system” is good approximation

- But varies wildly

“The one program running at all times on the computer” is the **kernel**.

- Everything else is either a system program (ships with the operating system) or an application program

# 4 Fundamental OS Concepts

## Thread



- Single unique execution context
- Program Counter, Registers, Execution Flags, Stack

## Address Space with Translation

- Programs execute in an *address space* that is distinct from the memory space of the physical machine



## Process



- An instance of an executing program is *a process consisting of an address space and one or more threads of control*

## Dual Mode operation/Protection

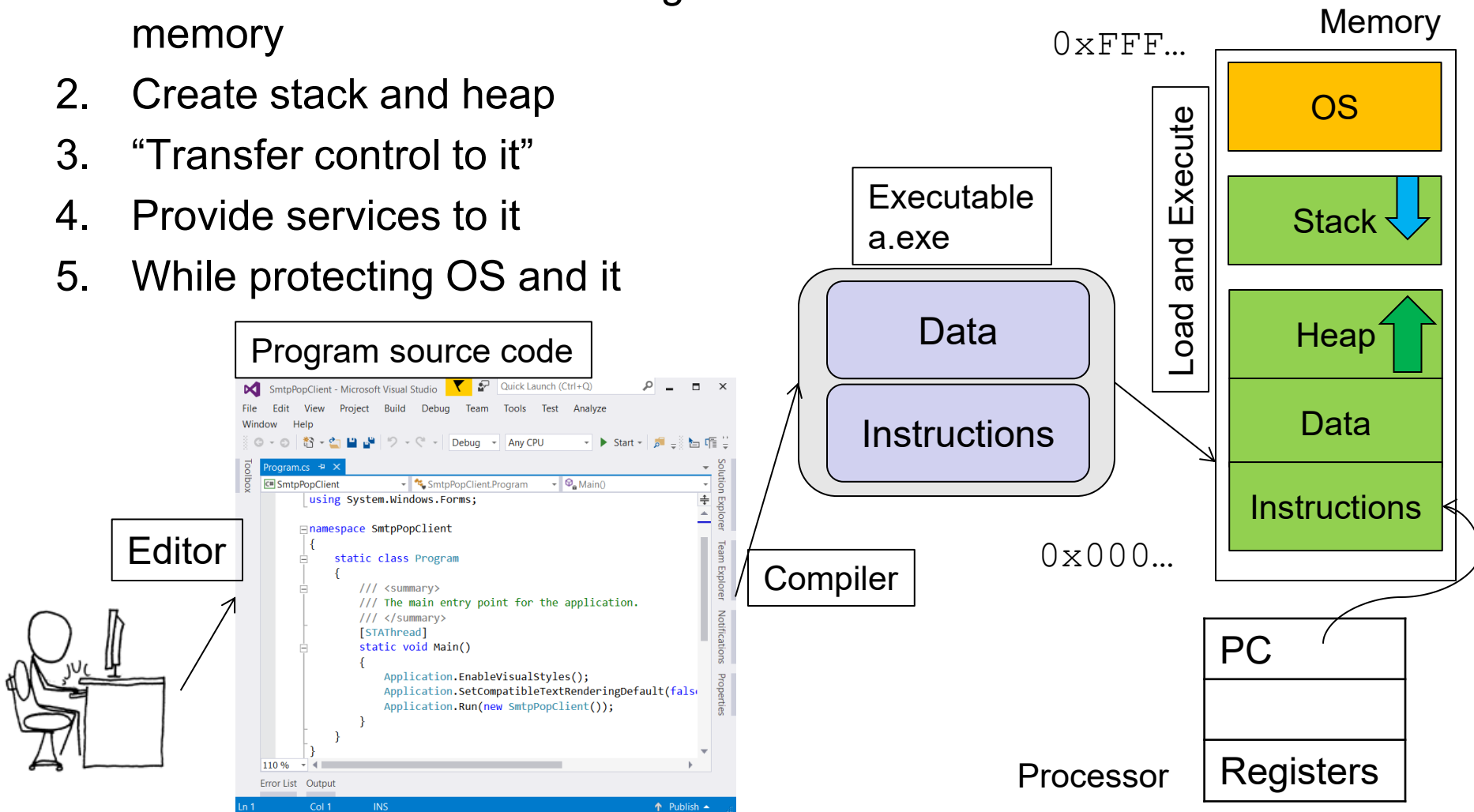
- Only the “**system**” can access certain resources
- The OS and the hardware are protected from user programs and user programs are isolated from one another by **controlling the translation** from program virtual addresses to machine physical addresses



# OS' Bottom Line: Run Stuff

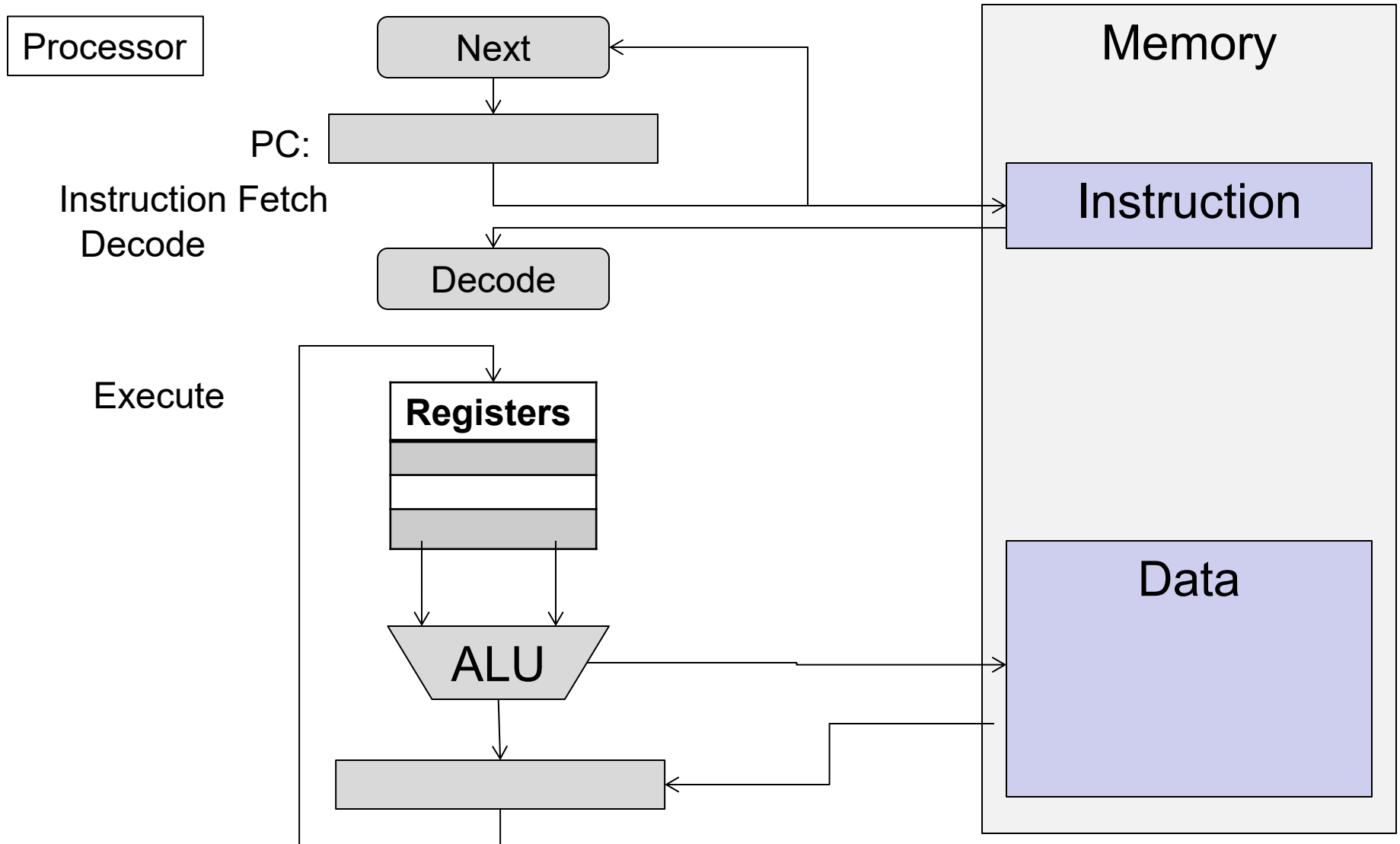


1. Load instruction and data segments of executable file into memory
2. Create stack and heap
3. “Transfer control to it”
4. Provide services to it
5. While protecting OS and it

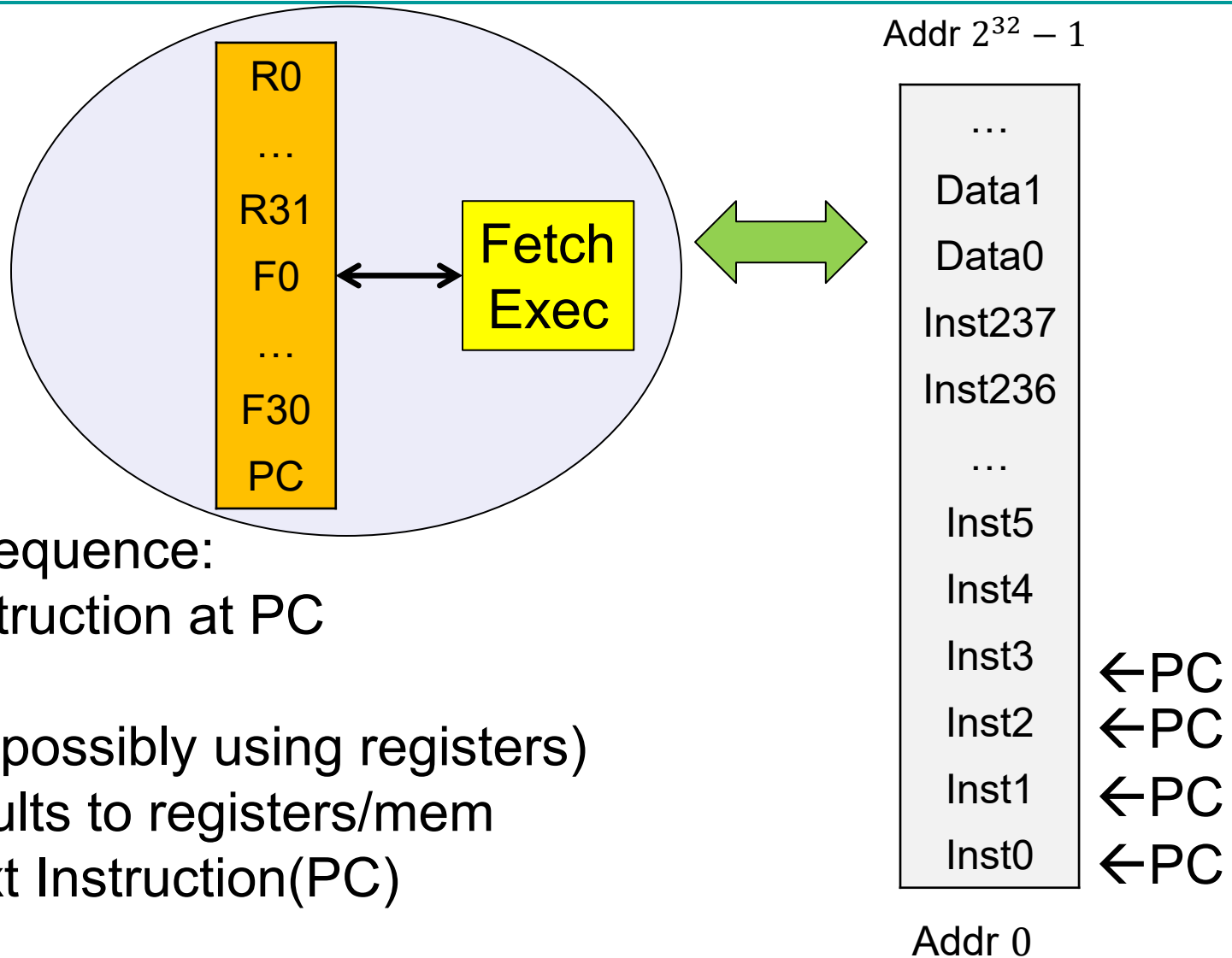




# The Instruction Cycle



# What happens during program execution?



# First OS Concept: Thread of Control

---



- Thread: Single unique execution context
  - Program Counter, Registers, Execution Flags, Stack
- A thread is executing on a processor when it is resident in the processor registers.
- PC register holds the address of executing instruction in the thread.
- Certain registers hold the *context* of thread
  - Stack pointer holds the address of the top of stack
    - Other conventions: Frame Pointer, Heap Pointer, Data
  - May be defined by the instruction set architecture or by compiler conventions
- Registers hold the root state of the thread.
  - The rest is “in memory”

# So Far

---

- (Brief) OS History
- Virtual Machines
- 4 Main OS Concepts
  - Thread
  - Address
  - Process
  - Dual mode



# Protecting Programs

- **Problem:** Run multiple applications in such a way that they are protected from one another
- **Goal:**
  - Keep **User Programs** from crashing the **OS**
  - Keep **User Programs** from crashing **each other**
  - [Keep Parts of **OS** from crashing other parts?]
- (Some of the required) Mechanisms:
  - **Address Translation**
  - Dual Mode Operation

Later

## Simple Policy:

- Programs are not allowed to read/write memory of other Programs or of the Operating System

Task Manager

File Options View

Processes Performance App history Startup Users Details Services

Name	Status	5% CPU	43% Memory	1% Disk	2% Network	0% GPU	GPU
<b>Apps (9)</b>							
> Firefox (10)		0.7%	1,212.5 MB	0.1 MB/s	0.5 Mbps	0%	
> Microsoft PowerPoint (32 bit)		0%	92.6 MB	0 MB/s	0 Mbps	0%	
> Task Manager		0.4%	29.2 MB	0 MB/s	0 Mbps	0%	
> Thunderbird (32 bit)		0.1%	316.6 MB	0.1 MB/s	0.1 Mbps	0%	
> Toggl Track		0%	61.3 MB	0.1 MB/s	0 Mbps	0%	
> Windows Explorer (4)		0.2%	88.9 MB	0 MB/s	0 Mbps	0%	
> WinEdt 10.3		0%	17.6 MB	0 MB/s	0 Mbps	0%	
> XnView MP		0%	31.3 MB	0.1 MB/s	0 Mbps	0%	
> Zoom Meetings (32 bit) (2)		0.2%	57.0 MB	0 MB/s	0.1 Mbps	0%	
<b>Background processes (107)</b>							
> Adobe Acrobat Update Service ...		0%	0.3 MB	0 MB/s	0 Mbps	0%	
AlpsAlpine Pointing-device Driv...		0%	0.4 MB	0 MB/s	0 Mbps	0%	
AlpsAlpine Pointing-device Driv...		0%	1.0 MB	0 MB/s	0 Mbps	0%	
AlpsAlpine Pointing-device Driv...		0%	0.4 MB	0 MB/s	0 Mbps	0%	
> Antimalware Service Executable		0.3%	102.9 MB	0.1 MB/s	0 Mbps	0%	
AntMsnEwd		0.1%	0.6 MB	0 MB/s	0 Mbps	0%	

< >

⬅ Fewer details End task

# Threads View

ProcessThreadsView - C:\Program Files\Microsoft Office\root\Office16\POWERPNT.EXE

File Edit View Options Help

Thread ID	Context Swi...	Last Context Switches	Status	Base Priority	Dynamic Priority	Created Time	User Time	Kernel Time	Visible Windows	Hidden Windows	Window Title	Window Class	Start Address	Stack Base	Stack Limit	Stack Size	TEF
30188	862,614	0	UserRequest	8	10	08/10/2023 11:10:03	00:00:21.406	00:00:05.750	2	16	317-Grading-...	PPTFrameClass	POWERPNT.EX...	000000007C940000	000000007C8A8000	0000000000098000	00C
17316	198,947	18	UserRequest	8	8	08/10/2023 11:10:04	00:00:00.078	00:00:00.015	0	0			POWERPNT.EX...	000000007ED00000	000000007ECFA000	0000000000060000	00C
3624	158,473	0		8	8	08/10/2023 11:10:04	00:00:00.828	00:00:00.046	0	0			POWERPNT.EX...	000000007DD00000	000000007DCF5000	0000000000080000	00C
10080	135,819	0	UserRequest	8	9	08/10/2023 11:10:04	00:00:02.156	00:00:00.843	0	0			POWERPNT.EX...	000000007DA00000	000000007D9EC000	0000000000014000	00C
13684	129,816	0	UserRequest	8	8	08/10/2023 11:10:04	00:00:00.046	00:00:00.125	0	1			POWERPNT.EX...	000000007D400000	000000007D3FE000	0000000000002000	00C
21020	12,521	1	UserRequest	8	8	08/10/2023 11:10:09	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	0000000000B00000	0000000000A00000	0000000000002000	00C
28676	12,358	1	UserRequest	8	8	08/10/2023 11:10:09	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	0000000000C00000	0000000000BFD000	0000000000003000	00C
25552	3,958	0	UserRequest	8	8	08/10/2023 14:01:29	00:00:00.031	00:00:00.000	0	0			POWERPNT.EX...	00000000027C0000	00000000027BB000	0000000000005000	00C
8108	3,230	0	UserRequest	8	9	08/10/2023 11:10:04	00:00:00.203	00:00:00.015	0	0			POWERPNT.EX...	000000007E600000	000000007E5F5000	000000000000B000	00C
29652	1,100	0	DelayExecution	8	8	08/10/2023 12:55:33	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	00000000026C0000	00000000026BD000	0000000000003000	00C
9988	1,063	0	WtUserRequest	8	10	08/10/2023 12:55:32	00:00:00.000	00:00:00.000	0	2			POWERPNT.EX...	0000000001200000	00000000011FB000	0000000000005000	00C
24936	988	0	WtUserRequest	8	10	08/10/2023 11:10:14	00:00:01.562	00:00:01.421	0	0			POWERPNT.EX...	0000000000400000	00000000003F8000	0000000000008000	00C
28412	651	0	UserRequest	8	8	08/10/2023 12:55:33	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	0000000001E00000	0000000001DFD000	0000000000003000	00C
17608	384	0	UserRequest	8	8	08/10/2023 11:10:04	00:00:00.000	00:00:00.031	0	0			POWERPNT.EX...	000000007DE00000	000000007DDF9000	0000000000007000	00C
836	377	0	UserRequest	8	8	08/10/2023 11:10:04	00:00:00.031	00:00:00.031	0	0			POWERPNT.EX...	000000007E300000	000000007E2F9000	0000000000007000	00C
11840	373	0	UserRequest	13	15	08/10/2023 14:12:47	00:00:00.000	00:00:00.000	0	2			POWERPNT.EX...	00000000028C0000	00000000028BD000	0000000000003000	00C
23292	338	0	UserRequest	8	8	08/10/2023 11:10:04	00:00:00.062	00:00:00.015	0	0			POWERPNT.EX...	000000007E000000	000000007DDF9000	0000000000007000	00C
30372	282	0	UserRequest	8	8	08/10/2023 12:55:33	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	0000000002280000	000000000227D000	0000000000003000	00C
24288	274	0	UserRequest	8	10	08/10/2023 13:27:13	00:00:00.015	00:00:00.000	0	2			POWERPNT.EX...	0000000000E00000	0000000000DF1000	000000000000F000	00C
27624	253	0	UserRequest	8	8	08/10/2023 11:10:04	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	000000007E200000	000000007E1F9000	0000000000007000	00C
25692	241	0	WtQueue	8	9	08/10/2023 14:15:49	00:00:00.000	00:00:00.000	0	0			POWERPNT.EX...	0000000000200000	00000000001FB000	0000000000005000	00C
19660	139	0	UserRequest	10	10	08/10/2023 12:55:33	00:00:00.015	00:00:00.000	0	0			POWERPNT.EX...	0000000002000000	0000000001F06000	00000000000FA000	00C

Stack Data:

00000000 7ECFF8A0 00000000 67E259F0 -> api-ms-win-appmodel-runtime-l1-1-2

00000000 7ECFF860 00000000 BE046960 -> Status: 0x8081x

00000000 7ECFF848 00000000 67E25BF8 -> AppPolicyGetThreadInitializationType

00000000 7ECFF810 00000000 67E25BF8 -> AppPolicyGetThreadInitializationType

00000000 7ECFF7E0 00000000 67E25BF8 -> AppPolicyGetThreadInitializationType

00000000 7ECFF740 00000000 975D33D0 -> C:\WINDOWS\SYSTEM32\kernel.appcore.dll

00000000 7ECFF6F0 00000000 67E25BF8 -> AppPolicyGetThreadInitializationType

00000000 7ECFF6C0 00000000 67E25BF8 -> AppPolicyGetThreadInitializationType

00000000 7ECFF658 00000000 67E25BF8 -> AppPolicyGetThreadInitializationType

00000000 7ECFF650 00000000 BE046898 -> Locating procedure "%s" by name

00000000 7ECFF630 00000000 BE046810 -> minkernel\ntdll\ldrsnap.c

00000000 7ECFF5E0 00000000 BE0468D8 -> minkernel\ntdll\ldrapi.c

00000000 7ECFF4F8 00000000 67E259F0 -> api-ms-win-appmodel-runtime-l1-1-2

00000000 7ECFE1F0 00000000 F237EB20 -> JSTimers e70f

00000000 7ECFE188 00000000 67E32770 -> void \_\_cdecl AI::Transceiver::Receiver::Main(class std::shared\_ptr<struct AI::Transceiver::Receiver::This>)

00000000 7ECFE0E8 00000000 67E32770 -> void \_\_cdecl AI::Transceiver::Receiver::Main(class std::shared\_ptr<struct AI::Transceiver::Receiver::This>)

00000000 7ECFE0D0 00000000 67E32688 -> D:\a\_work\1\s\src\ai\include\AI\Transceiver\Receiver.hpp

00000000 7ECFE0B8 00000000 67E32770 -> void \_\_cdecl AI::Transceiver::Receiver::Main(class std::shared\_ptr<struct AI::Transceiver::Receiver::This>)

00000000 7ECFE0A0 00000000 67E32688 -> D:\a\_work\1\s\src\ai\include\AI\Transceiver\Receiver.hpp

00000000 7ECFE030 00000000 67E32688 -> D:\a\_work\1\s\src\ai\include\AI\Transceiver\Receiver.hpp

00000000 7ECDFD70 00000000 67E31A60 -> D:\a\_work\1\s\src\ai\include\AI\Protocol.hpp

00000000 7ECDFE98 00000000 67E32770 -> void \_\_cdecl AI::Transceiver::Receiver::Main(class std::shared\_ptr<struct AI::Transceiver::Receiver::This>)

00000000 7ECDFD58 00000000 67E32770 -> void \_\_cdecl AI::Transceiver::Receiver::Main(class std::shared\_ptr<struct AI::Transceiver::Receiver::This>)

00000000 7ECDFD50 00000000 67E31A60 -> D:\a\_work\1\s\src\ai\include\AI\Protocol.hpp

00000000 7ECDFD18 00000000 67E31A60 -> D:\a\_work\1\s\src\ai\include\AI\Protocol.hpp

00000000 7ECDFC90 00000000 67E3CA30 -> \_\_cdecl AI::Serialization::EnumItem<class std::vector<enum std::byte,class std::allocator<enum std::byte>>>::EnumItem(unsigned int,enum AI::Serialization::Command,unsigned int,class

00000000 7ECDFC80 00000000 67E3C830 -> D:\a\_work\1\s\src\ai\include\AI\Serialization.hpp

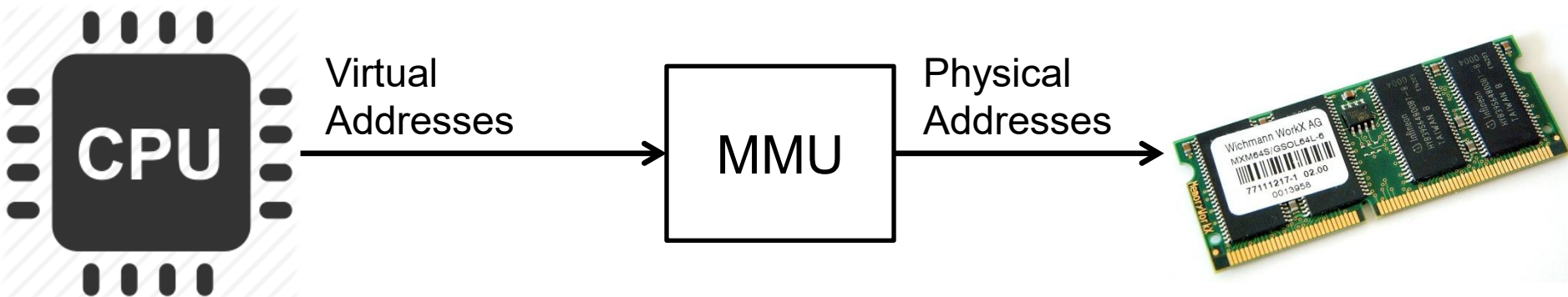
65 Threads, 1 Selected

NirSoft Freeware. <http://www.nirsoft.net>

# Address Translation



- Address Space
  - A group of memory addresses usable by something
  - Each program (process) and kernel has potentially different address spaces.
- Address Translation:
  - Translate from Virtual Addresses (emitted by CPU) into Physical Addresses (of memory)
  - Mapping *often* performed in Hardware by **Memory Management Unit (MMU)**





# You can't read or corrupt what you can't ask for

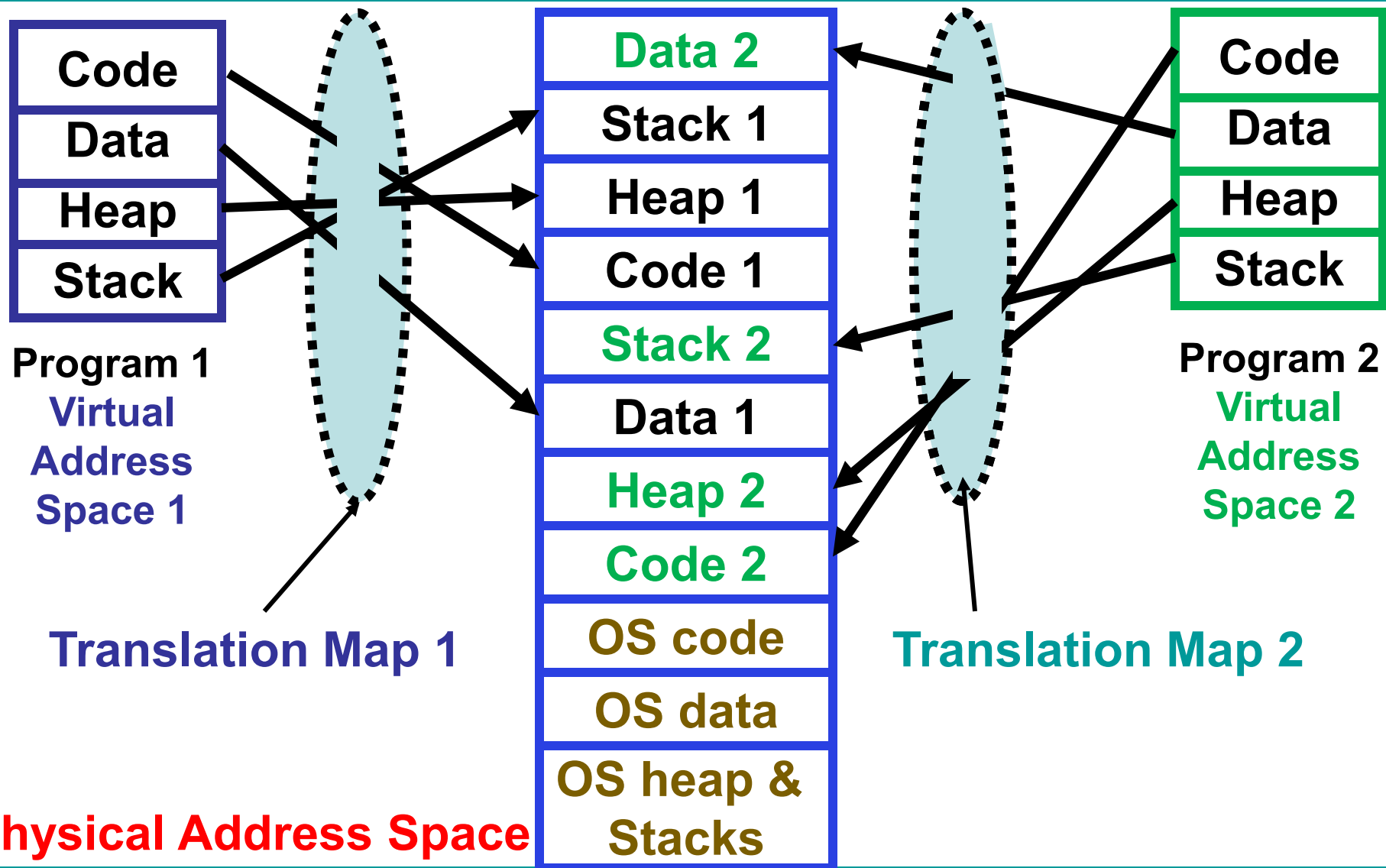
---



<https://cdn3.whatculture.com/images/2015/03/The-Matrix-Neo-Mouth.jpg>



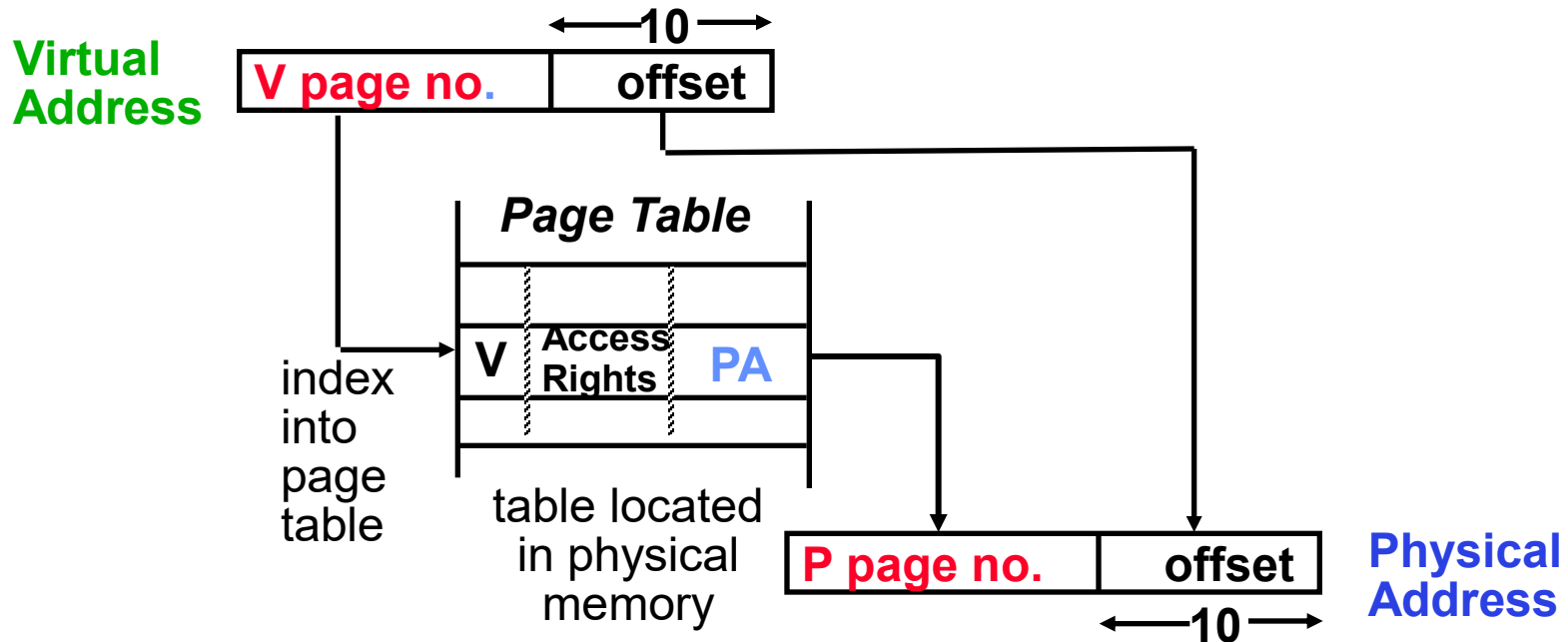
# Example of Address Translation



# Address Translation Details

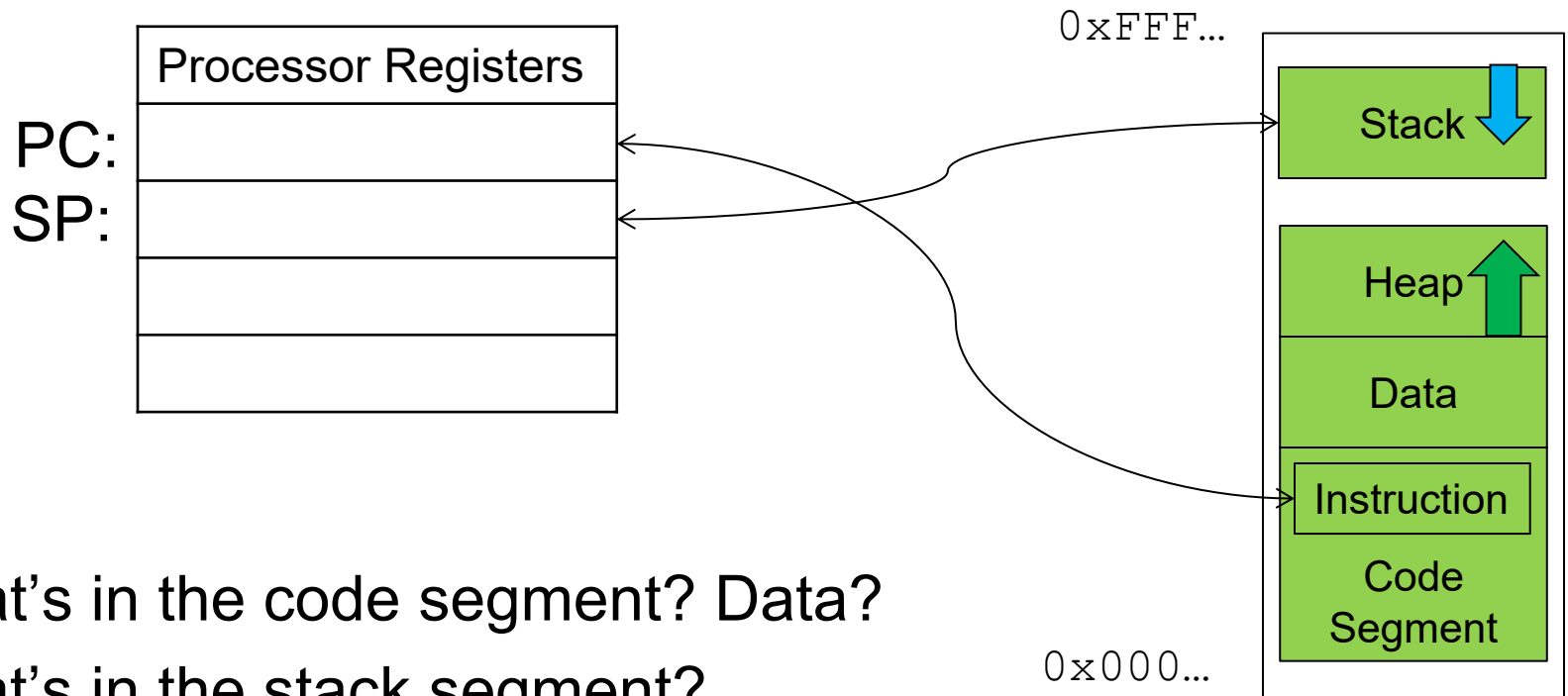


- For now, assume translation happens with table (called a Page Table):



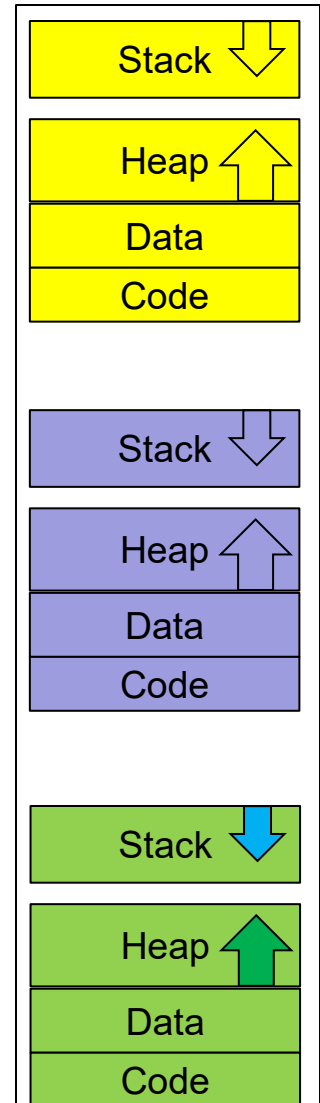
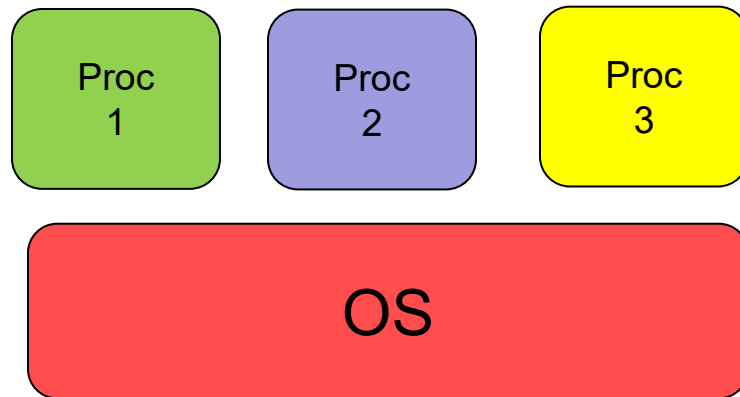
- Translation helps protection:
  - Control translations, control access
  - Should Users be able to change Page Table???

# Address Space: In a Picture



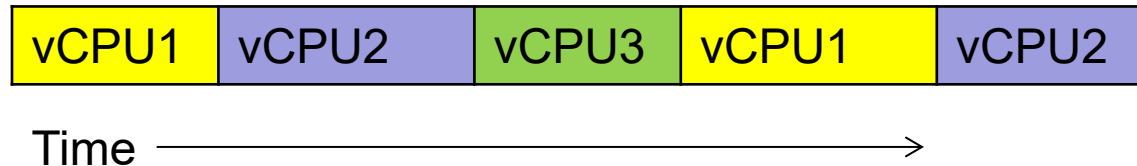
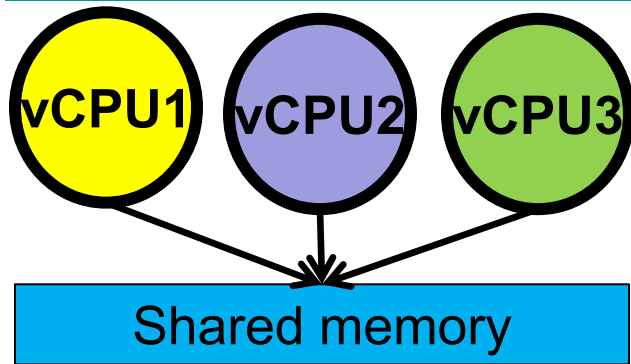
- What's in the code segment? Data?
- What's in the stack segment?
  - How is it allocated? How big is it?
- What's in the heap segment?
  - How is it allocated? How big?

# Multiprogramming – Multiple Threads of Control





# How can we give the illusion of multiple processors?



- Assume a single processor. How do we provide the illusion of multiple processors?
  - **Multiplex in time!**
- Each virtual “CPU” needs a structure to hold:
  - Program Counter (PC), Stack Pointer (SP)
  - Registers (Integer, Floating point, others...?)
- How **switch** from one virtual CPU to the next?
  - **Save** PC, SP, and registers in current state block
  - **Load** PC, SP, and registers from new state block
- What **triggers** switch?
  - Timer, voluntary yield, I/O, other things

# Conclusion

---

- (Brief) OS History
- Virtual Machines
- 4 Main OS Concepts
  - Thread
  - Address
  - Process
  - Dual mode