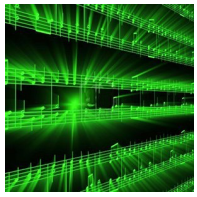
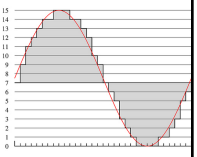
Penn Engineering

ESE





Lecture #18 – Operating Systems (OS)

# ESE 150 – Digital Audio Basics

ESE150 Spring 2021





## Observation

- **We want our phones (and computers) to do many things at once.**
- **If we dedicate a processor to MP3 decoding**
  - It will sit idle most of the time
- **MP3 decoding (and many other things) do not consume a modern processor**
- **Idea: Maybe we can share the processor among tasks?**





## Big Idea

- **Provide the illusion of (virtually) unlimited processors by sharing single processor in time**
- **Strategy**
  - Time-share processor
    - Store all process (virtual processors) state in memory
    - Iterate through processes
      - Restore process state
      - Run for a number of cycles
      - Save process state





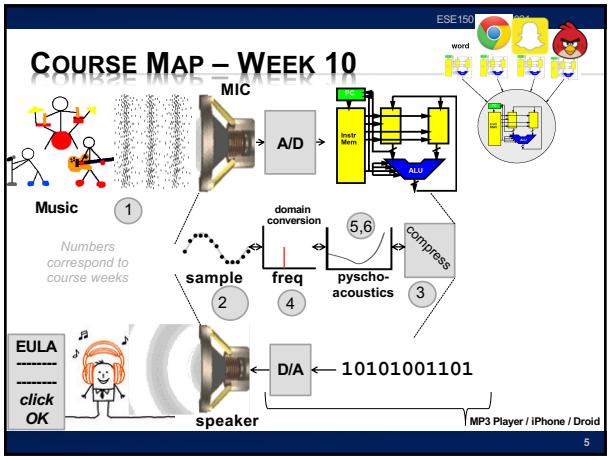
## Outline

- **Review**
- **Worksheet: Virtualization In Action**





## Course Map – Week 10








## Role of Operating System

- **Higher-level, shared support for all programs**
  - Could put it in program, but most programs need it!
  - Needs to be abstracted from program
- **Resource sharing**
  - Processor, memory, "devices" (net, printer, audio)
- **Polite sharing**
  - Isolation and protection
  - *Fences make Good Neighbors* – R. Frost
- **Idea:** Expensive/limited resources can be *shared* in time – OS manages this sharing

# VIRTUALIZATION





## IDEA

- **Virtualize the processor**
  - Make it look like we have multiple processors
  - With each program running on its own processor
- **"Own" processor**
  - Can put data in memory where it wants
  - Doesn't have to worry about another program scribbling over its memory
  - Its state is preserved and isolated
  - Looks like it runs all the time on the processor
    - Doesn't need to be programmed to allow other programs to run





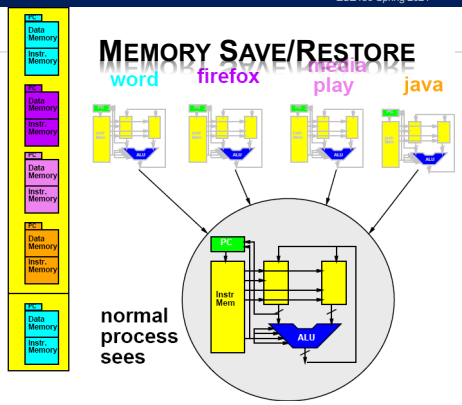
## KEY IDEA

- **Can capture state of a processor**
  - All the information that defines the current point in the computation
  - i.e. program counter, data and instruction memory
- **Can save that in memory**
  - A different memory from what the process sees
  - (could be different range of addresses)
- **Fully represents the running program**
- **Can restore that from memory to the processor**
- **Can save/restore without affecting the functional behavior of the program**





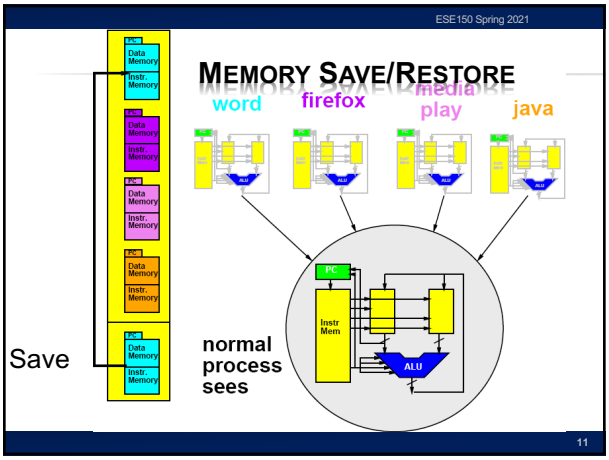
## MEMORY SAVE/RESTORE







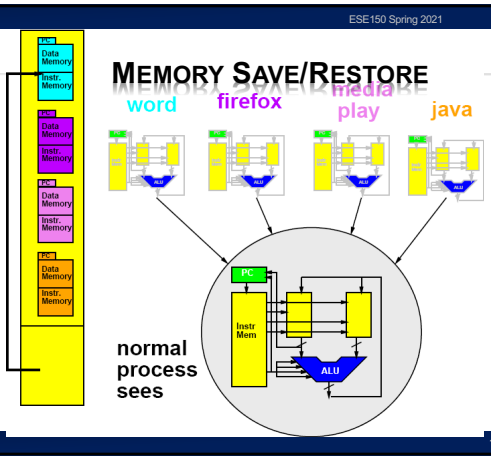

## MEMORY SAVE/RESTORE



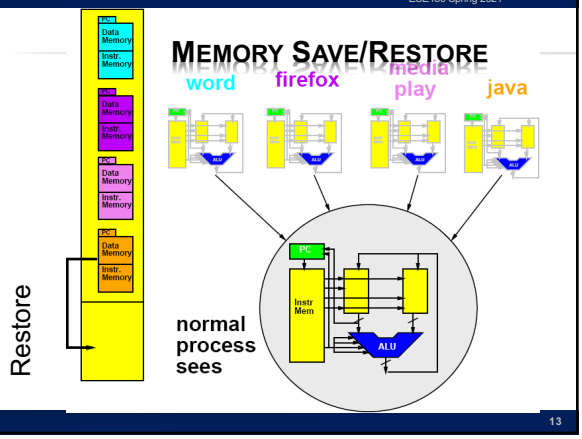





## MEMORY SAVE/RESTORE



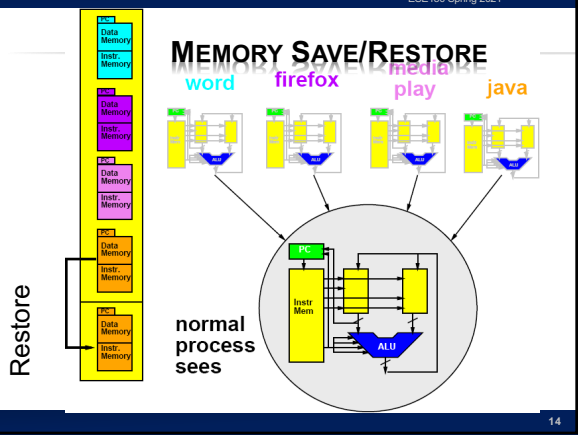
## MEMORY SAVE/RESTORE








## MEMORY SAVE/RESTORE

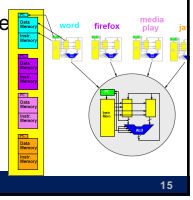







## SHARING PROCESSOR

- **Now that we can save/restore the state**
- **Can share processor among processes**
  - (Restore state; run for time; save state)
- **Isolation: none of the processes need to know about each other**
  - Each thinks it has the a whole machine
  - Just need to restore/save state around epochs where the process gets to run on the processor







Worksheet Exercise

## DEMONSTRATION





## WORKSHEET: EXECUTION EXERCISE

- **We're going to simulate the computer and watch the processor state**





## EXECUTION EXERCISE

- **Google Doc – simulate A for 12 cycles**

## SIMULATE SWAPPING

- **Imagine we ran A for 6 cycles (and saved state)**
- **Swap and Run B for 6 cycles**
- **Swap and Run A for next 6 cycles**
  - What should we get?
- **Swap and Run B for next 6 cycles**





## SIMULATE SWAPPING

- **Simulate B for 6 cycles**
  - Individually
- **What get?**





## SIMULATE SWAPPING

- **Swap in A+6 and Simulate for 6 cycles (to 12)**
  - individually
- **What get?**





## SIMULATE SWAPPING

- **Swap in B+6 and Simulate for 6 cycles (to 12)**
  - individually
- **What get?**





## SIMULATE SWAPPING

- **Imagine we ran A for 6 cycles (and saved state)**
- **Swap and Run B for 6 cycles**
- **Swap and Run A for next 6 cycles**
  - What should we get?
- **Swap and Run B for next 6 cycles**





## REVIEW: KEY IDEA

- **Can capture state of a processor**
  - All the information that defines the current point in the computation (PC, data and instruction mem)
- **Can save that in memory**
  - A different memory from what the process sees
  - (could be different range of addresses)
- **Fully represents the running program**
- **Can restore that from memory to the processor**
- **Can save/restore without affecting the functional behavior of the program**
- **Time-share processor → pretend have unlimited number**

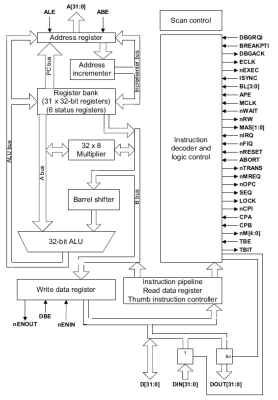
# Media Processors





# iPod Processor

- **Early based on PortalPlayer series**
  - Two ARM7TDMI cores
  - 80MHz each
- **Guesses are ARM7 or ARM8**

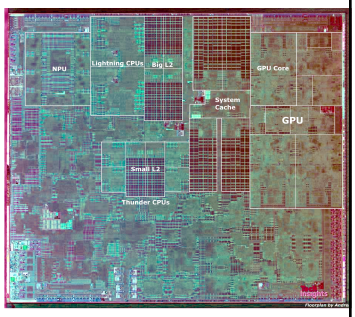







# Apple A13 Bionic

- **98mm$^2$, 7nm**
- **8.5 Billion Tr.**
- **iPhone 11 +**
- **6 ARM cores**
  - 2 fast (2.6GHz)
  - 4 low energy
- **4 custom GPUs**
- **Neural Engine**
  - 5 Trillion ops/s?








# Big Ideas

- **Virtualize hardware**
  - Identify state; save/restore from memory
- **Program view: owns complete machine**
- **Allows programs to share limited physical hardware (e.g. processor)**
  - Provide illusion of unlimited hardware
- **Operating System is the program that manages this sharing**





# Learn More

- **CIS380 – Operating Systems**





# Reminders

- **Feedback**
- **Lab9 due on Friday**