COMP 310/ECSE 427
Computer Systems and Organization

Lecture #1

Introduction to Operating Systems

Prof. Joseph Vybihal

Announcements

- Instructor coordinates
- Course outline
  - – Generic OS Course
  - – Look a little at Unix, Microsoft, Apple
- Participation
- Web CT
  - – discussions, assignments, mail, lectures
- Job!
  - – Web programming $13/hr

The Operating System

What is this?

What is in it?

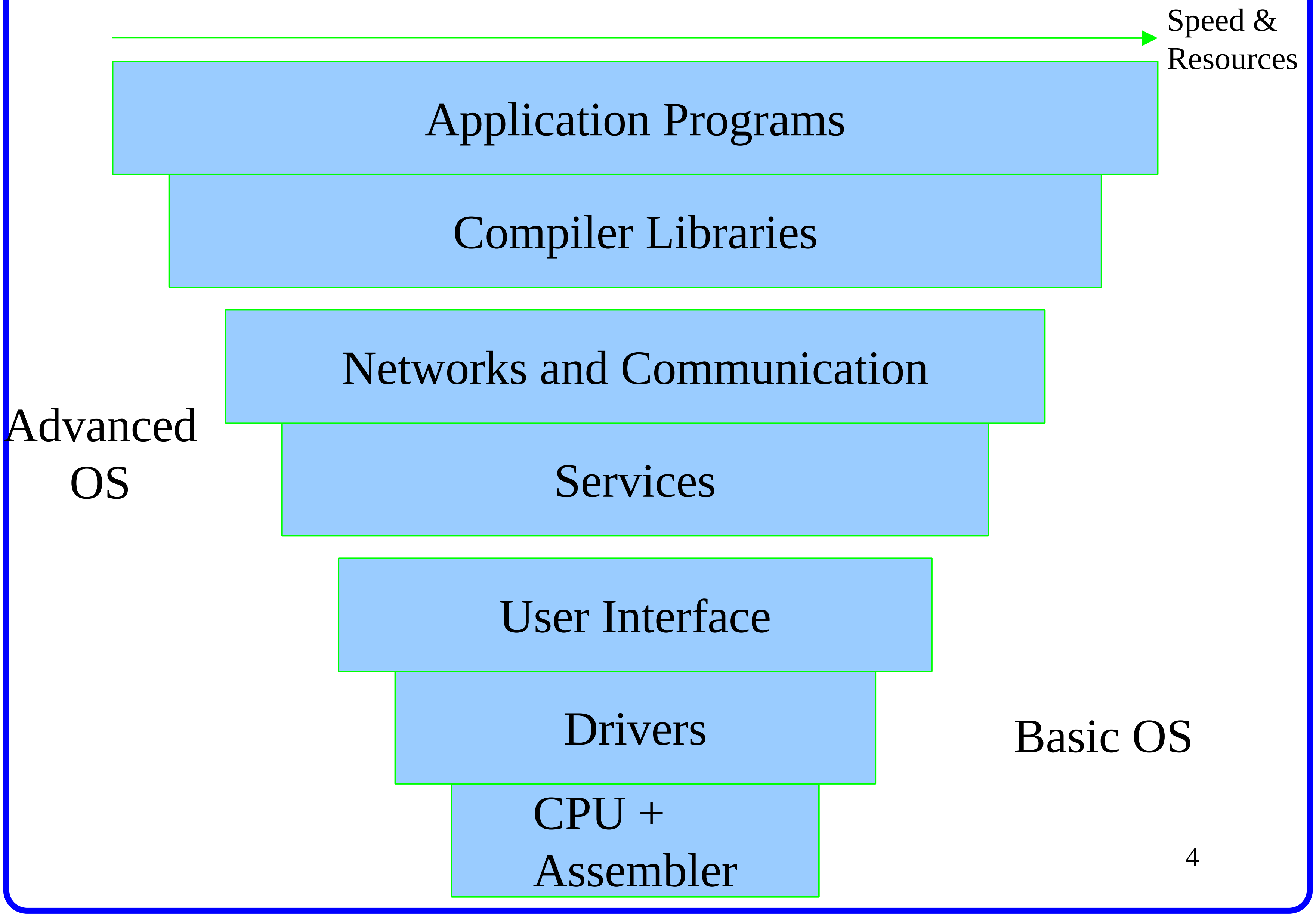
What can it do?

How can we make it do things?

How can we make this CPU write to the screen?

This is a lot of work, how can we make it simpler?

The Operating System



Speed & Resources

Application Programs

Compiler Libraries

Networks and Communication

Services

Advanced OS

User Interface

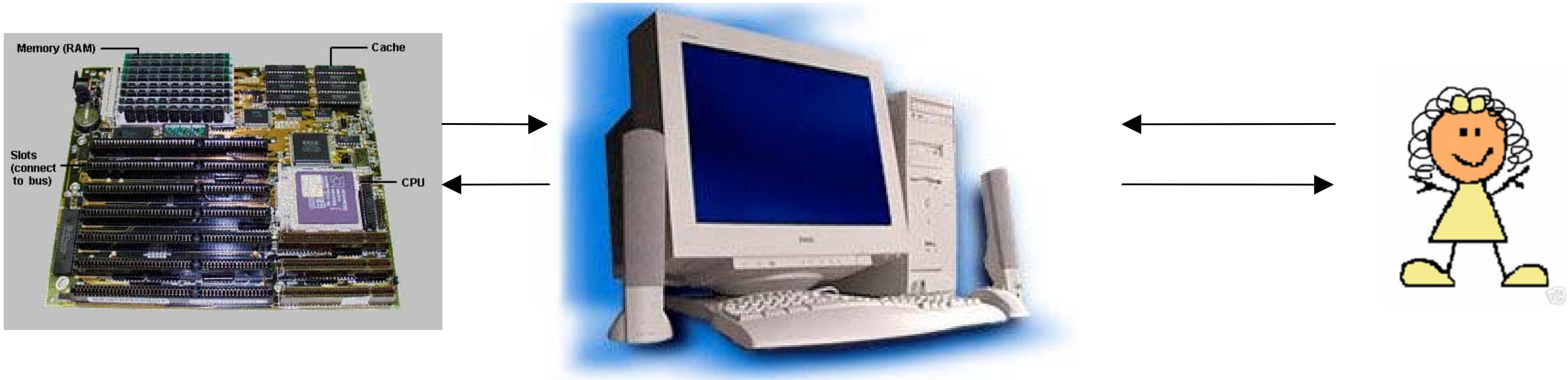
Drivers

CPU + Assembler

Basic OS

Why is the OS Needed?

It is an interface



They speak different languages … a **translator** is needed

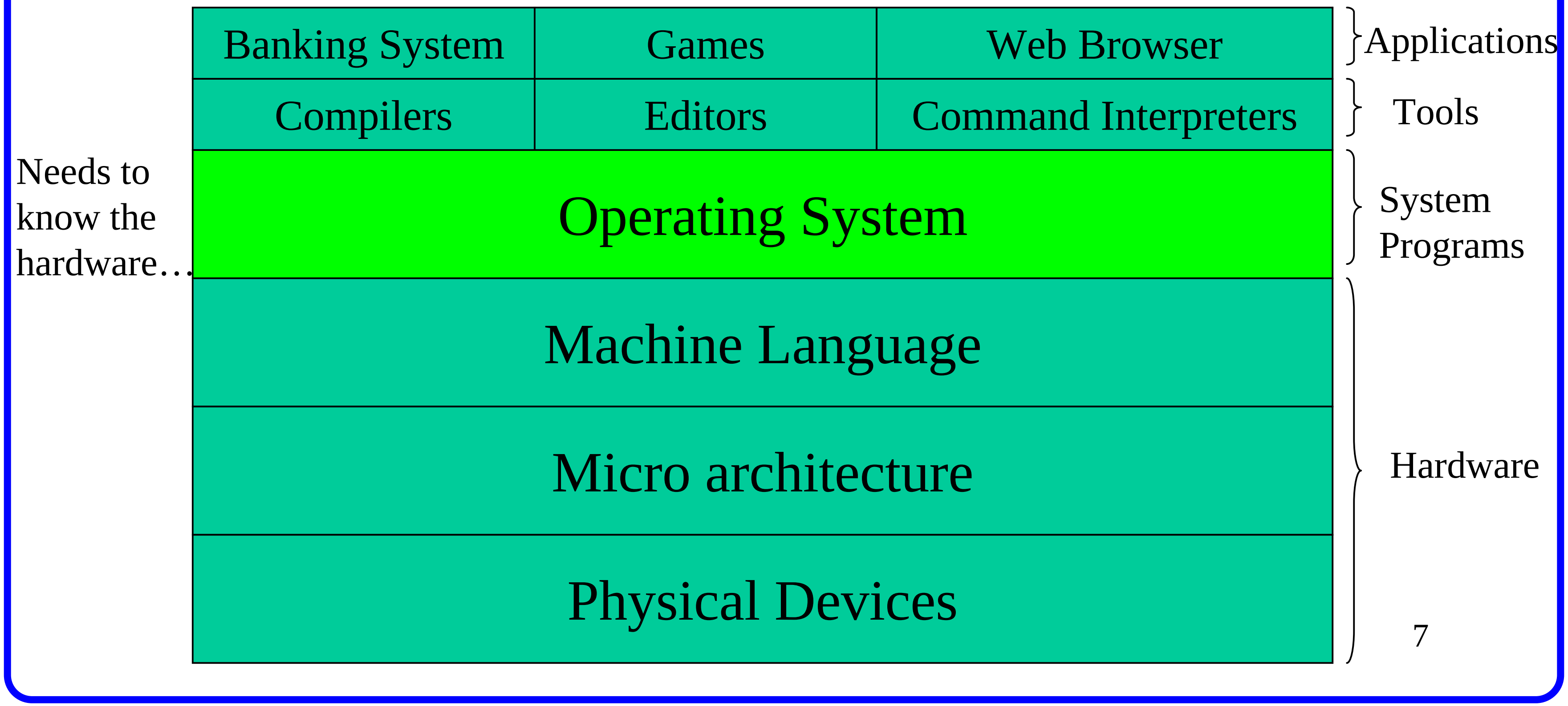
Computers are designed to be easy to build.

Operating systems are designed to make the computer accessible to people.

(and to software)?

Why is the OS Needed?

- Need #1 – Human Machine Interface
- Need #2 – Program Interface
- Need #3 – Hardware Management



Banking System | Games | Web Browser

Compilers | Editors | Command Interpreters

Operating System

Machine Language

Micro architecture

Physical Devices

Applications

Tools

System Programs

Hardware

Needs to know the hardware…

NEED #1:

The Human Machine Interface

- Input:
  - – keyboard, mouse, scanner, …
- Output:
  - – screen, printer, …
- Network:
  - – modem, Ethernet, …
- Algorithmic:
  - – scripts, programs, …
- Environments:
  - – single or multi-processing, …

Provides an **abstraction** for the human… (they become a user)?

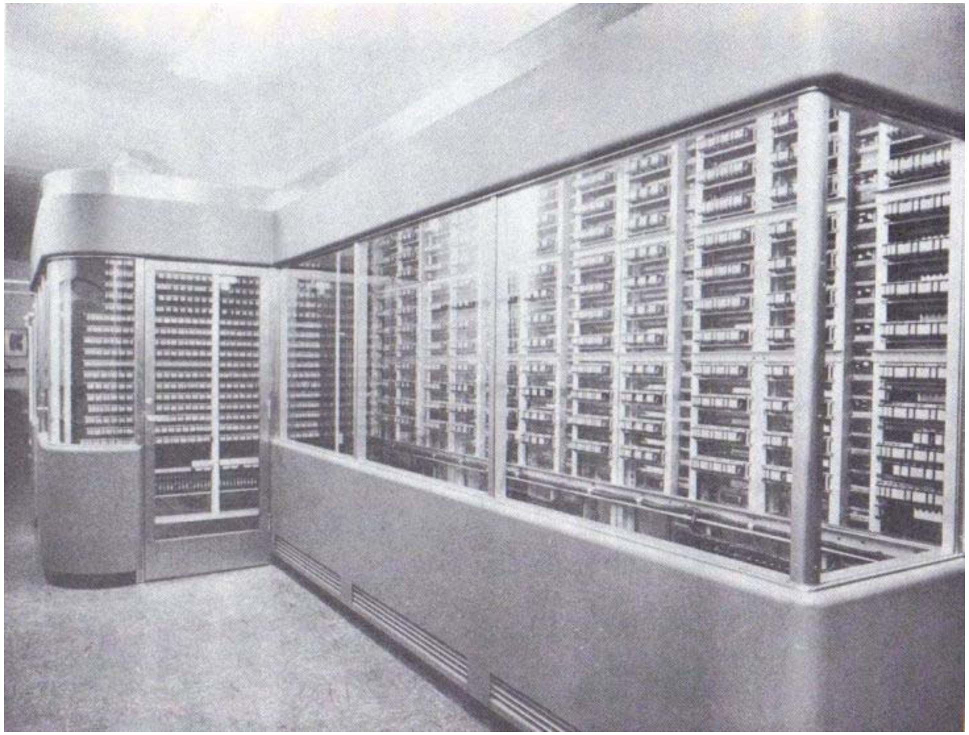
A history of human to machine interfaces

1941 Konrad Zuse (Z3)?

German built – during WW2 – Bombed by the Allies

IO: ticker tape input & typewriter output,
control panel OS & music box operation, relays for memory

1944 Howard Aiken & IBM The Mark 1



- 800 km of wire
- 3 million electrical connections
- Add in 0.3 sec
- Multiply in 6 sec
- Divide in 11.4 sec

- Used electromechanical relays and rotating shafts for data
- Sequencer controlled by punch cards

The operating system

SLOW
A lot of clicking

Memory initialized to off

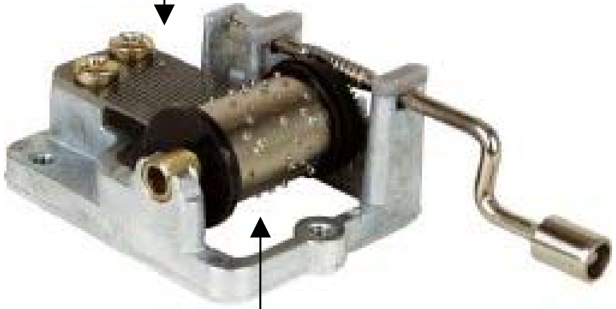
on

off

N wires

Byte not invented yet

Metallic electrodes



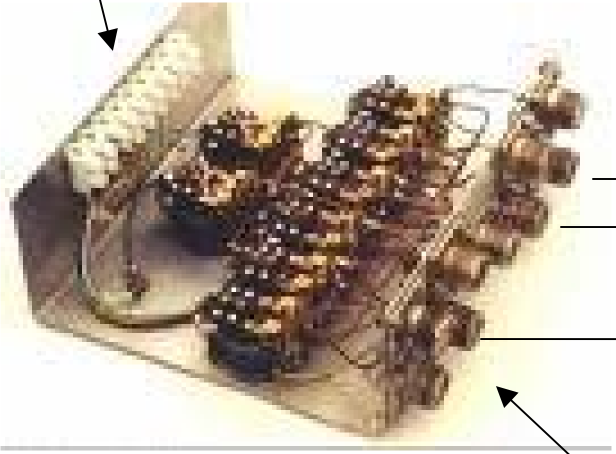
Electrically charged drum

When charged bump touched electrode, current flows and flips a switch.

The BYTE concept not needed – N wires only required

outputs to subsystems

Each wire represents a circuit subsystem that carries out a function: add, move, compare.



Switch box

add

move

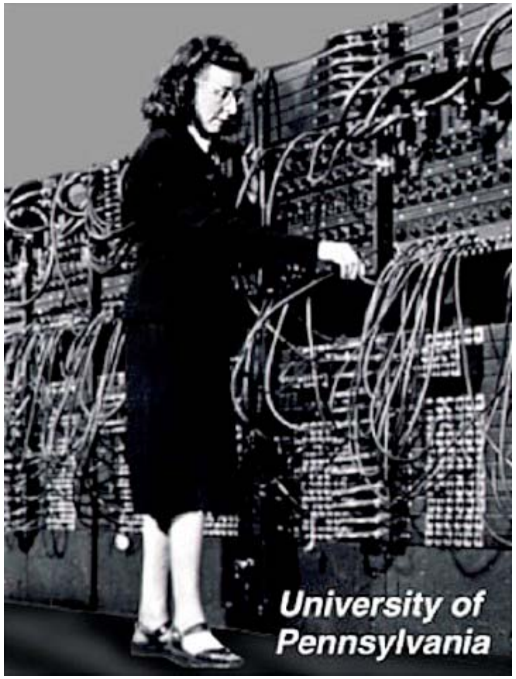
compare

inputs

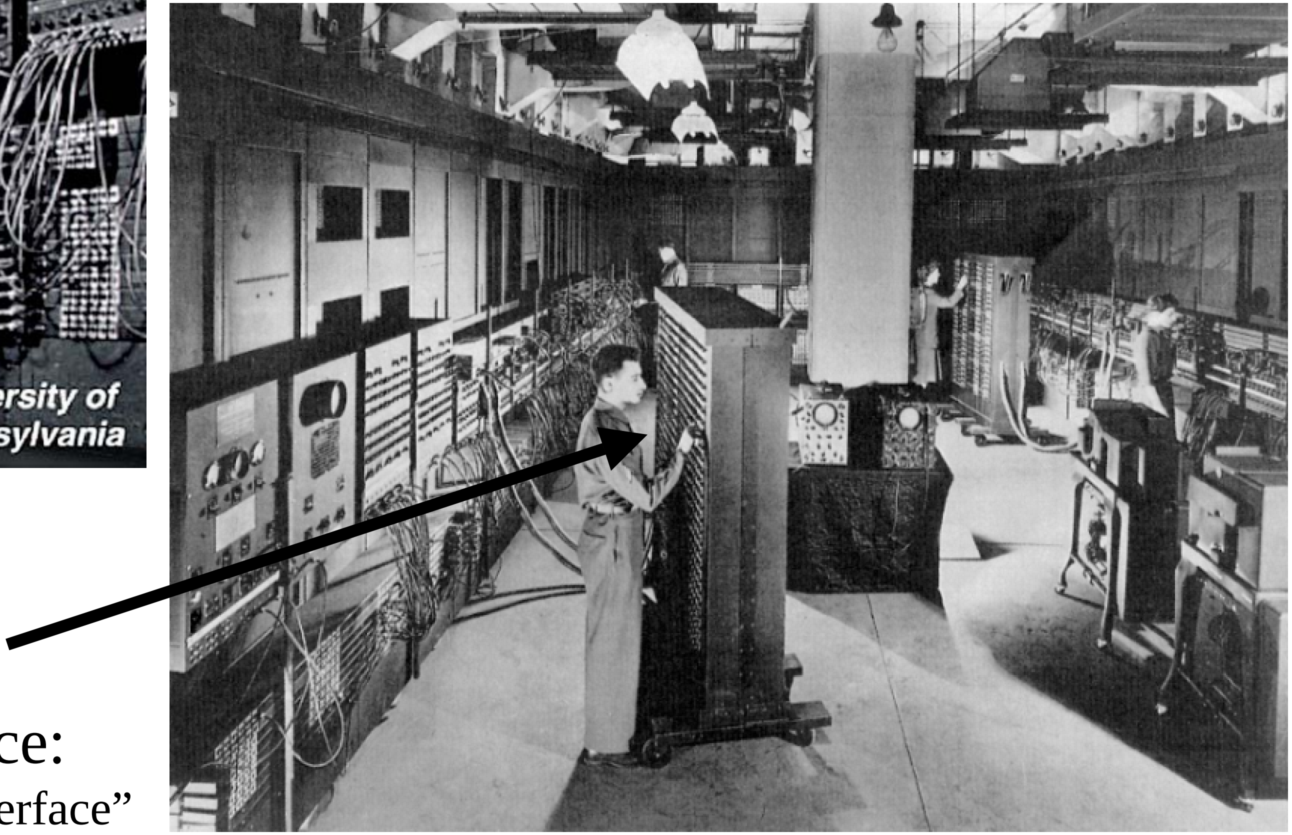
1946

Direct Feed Input Devices

WIRED FEED

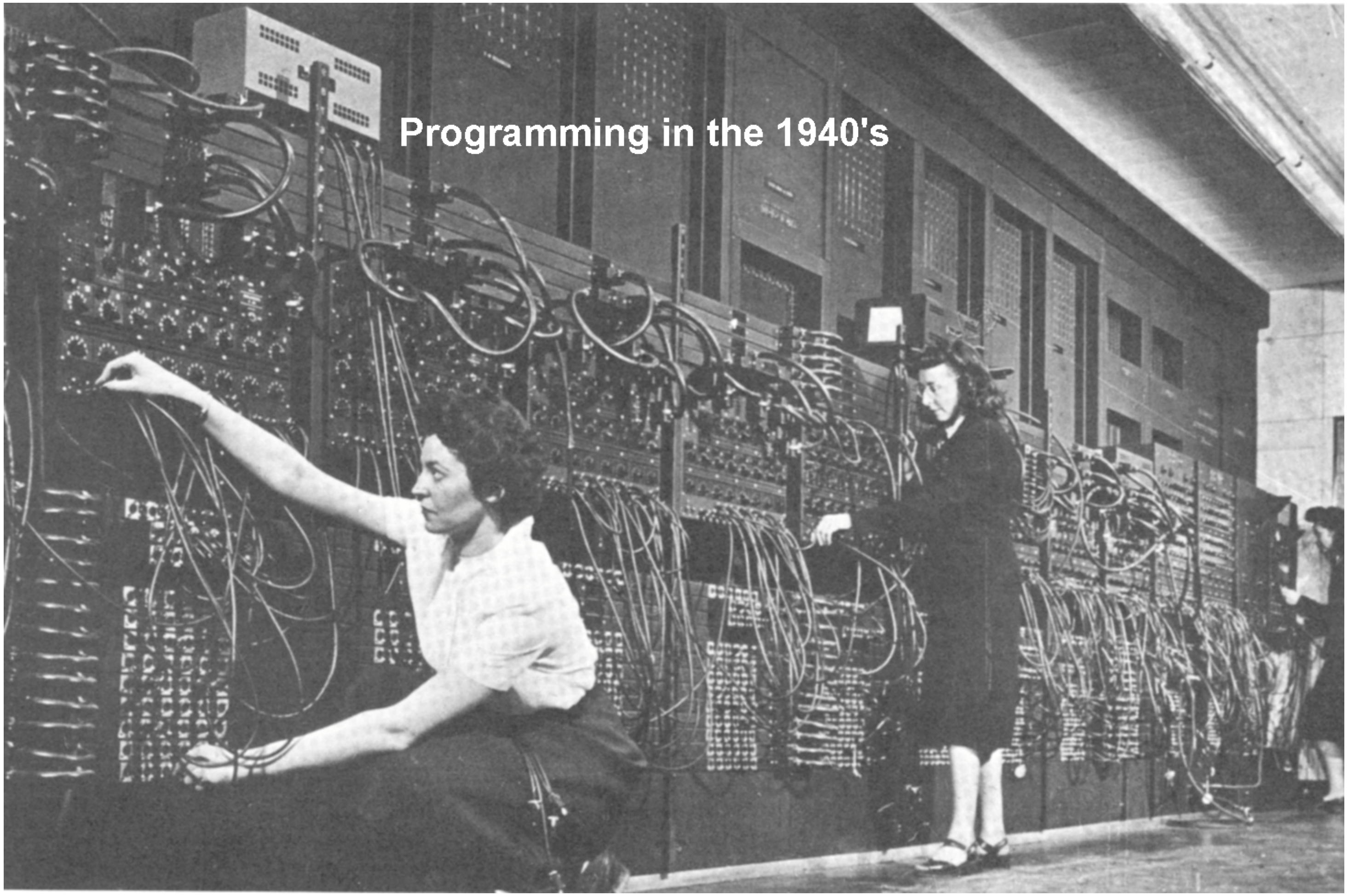


OS = Turn on, go to address zero, execute.



OS Input device:

“command-line interface”

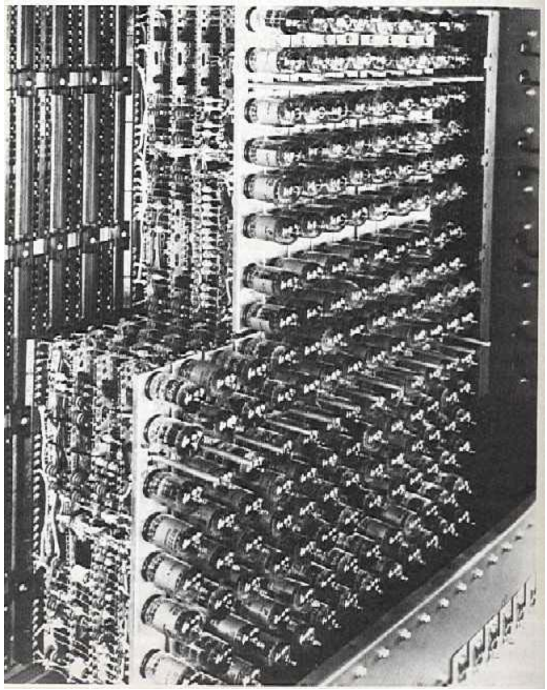


The wires where the program.
Control was then passed to the CPU for execution.

TABLE TOP FEED

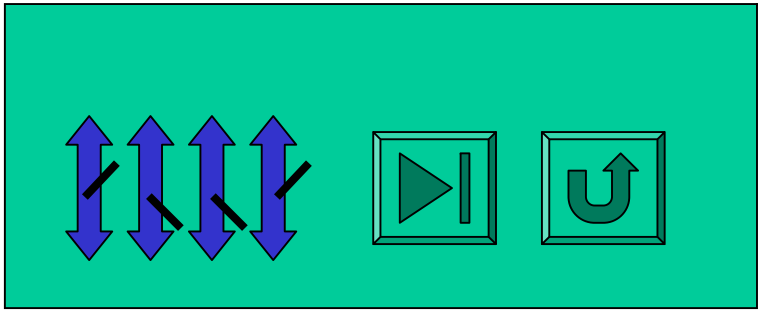
1950's

A bit!!



RAM

1	0	0	1



4-bit direct feed device

- Switches on/off
- Submit address ▶|
- Switches on/off
- Submit data ▶|
- Repeat
- When done: Run ↺

Only binary and machine language.

OS = I/O board, exec address zero.

Code in RAM
(Programming)?

(not invented yet)?

This.

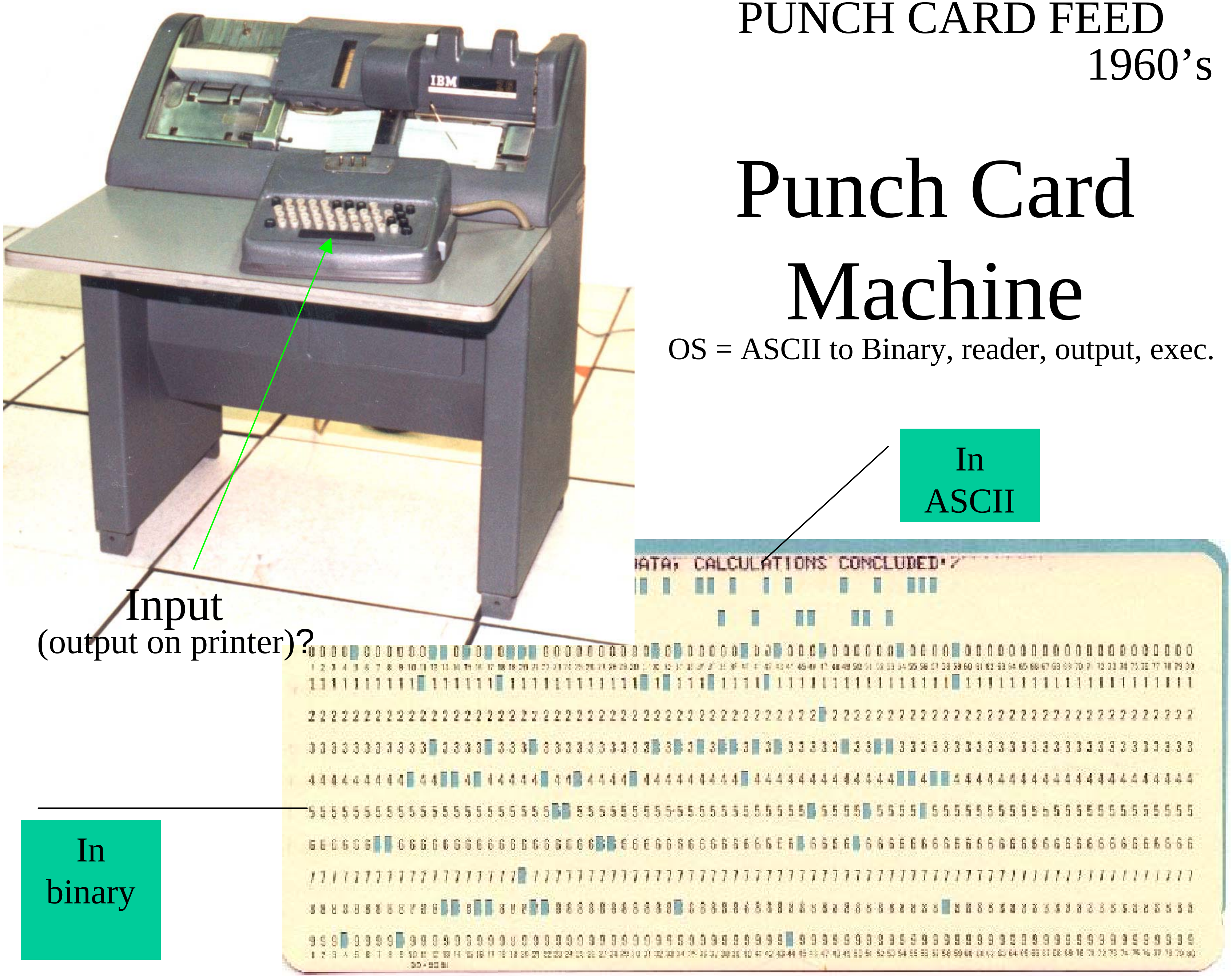
Java or C	Assembler	Machine Language	
if (x > 10)?	SUB x, 10	0000	101 000001 01010
x = x + 1;	BGT skip	0001	011 0101
else	SUB x, 1	0010	101 000001 00001
x = x – 1;	MOV x, acc	0011	000 000001 11111
	BR next	0100	100 0111
	skip:	0101	010 000001 00001
	ADD x, 1	0110	000 000001 11111
	MOV x, acc	0111	
	next:	*Address*	------- *Code/Data* ---------

Bits are sent to sequencer....

PUNCH CARD FEED
1960’s

Punch Card Machine

OS = ASCII to Binary, reader, output, exec.

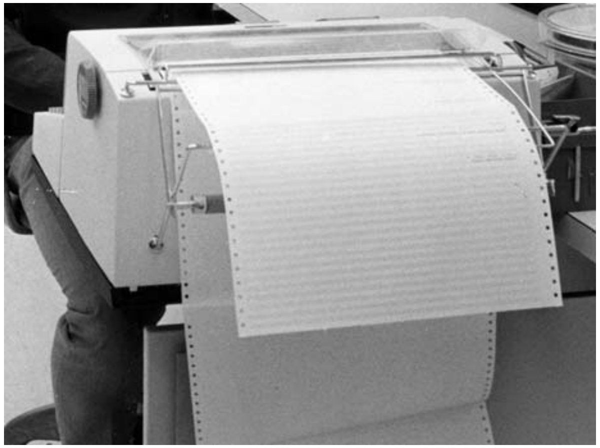


Input
(output on printer)?

In ASCII

In binary

TELETYPE:
1970’s



I/O finally on same device!

KEYBOARD:
1980’s

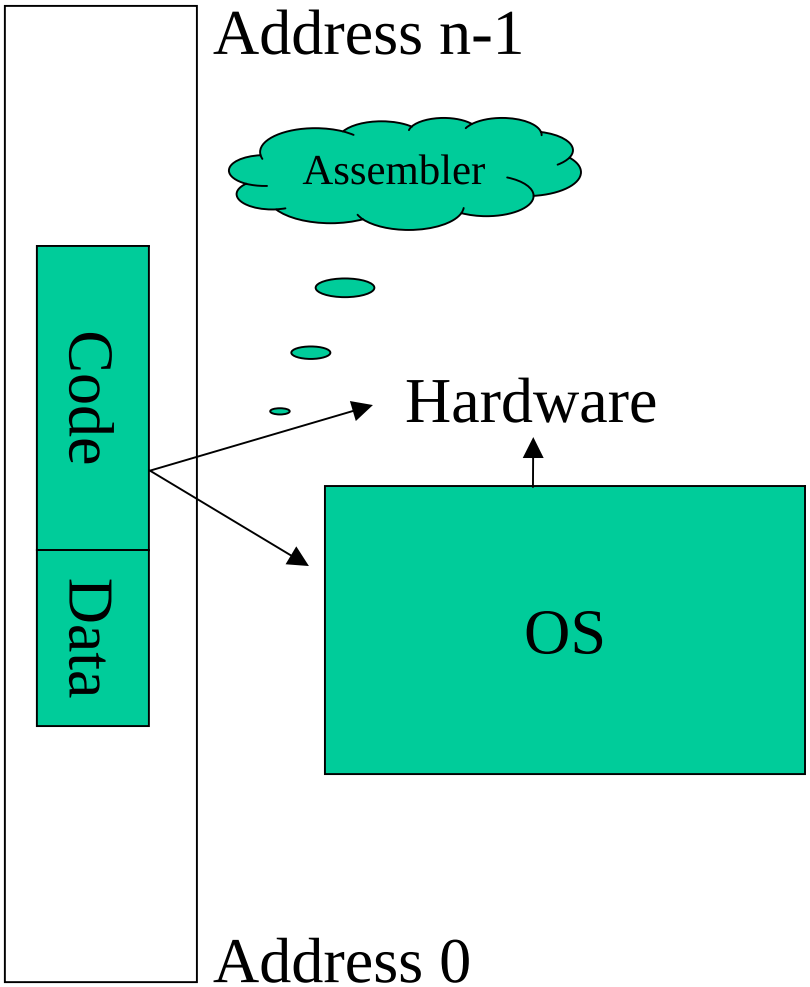
The OS manages the communication between the peripherals, the CPU, and the human.

MOUSE:

STYLUS: 1990’s

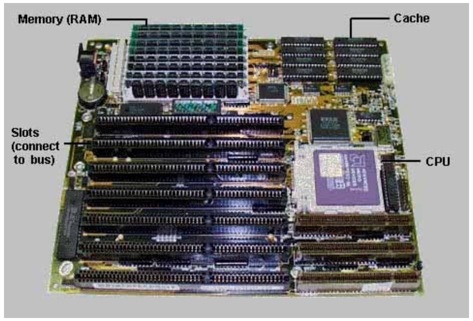
NEED #2

Interface for the Program



RAM

System Interfacing



Peripheral Interfacing

User Interfacing

Process Interfacing

For Example

- Program → Computer Screen (discuss)?
  - Use assembler to access the video card’s RAM
  - Use pre-built function in OS to access video card’s RAM
  - Use pre-built function in C to access pre-built function in OS to access video card’s RAM

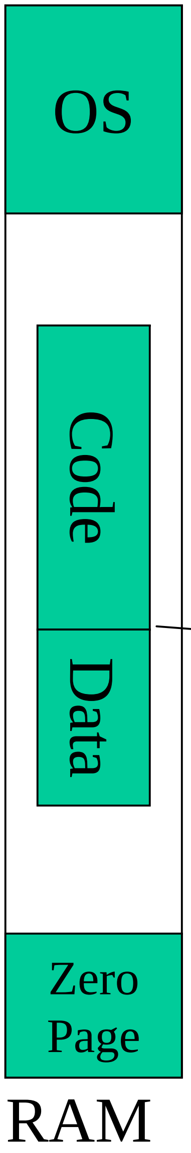
- What about…
  - Program to Printer
  - Program to Keyboard
  - Program to Network
  - Touch screen, stylus, character recognition, etc.

NEED #3

Hardware Management

- Boot the computer (CMOS not OS, but reboot/hibernate/etc.)?
- Give access to the CPU and provide features like multi-processing
- Give access to devices and provide for features like priority and queue
  - Disk drives
  - Screen
  - Keyboard and mouse
  - Printers
  - Networks

Complete Basic OS View



2

Program Execution Cycle

- User types program in and OS finds place in memory for information.
- When user says RUN, the OS passes first instruction of code to CPU.
- OS has now lost control of CPU
- CPU executes the code.

Last instruction is the address to the OS, or it crashes (or virus).

1

BOOT OPERATION

- Load OS & Format RAM
- Give address to first OS instruction to CPU IP.
- Pass control to CPU

CPU Execution Cycle

- Load info at IP
- Execute if possible else crash
- Increment IP
- Go to step 1

3

At a specific address

Buffer for key status
Buffer for 12 characters

Basic OS Architecture

User Interface	Get input from user and display results. Windowed or command-lined.
Memory Manager	Organizes RAM and remembers where everything is in RAM.
Disk / Storage Manager	Algorithms for finding and saving binary to and from disk drives
Process Manager	Algorithms for passing control to and from the CPU – OS – Process
Network Manager	Algorithms to integrate the OS with a network.
Hardware Manager	The drivers that provide extra code to the OS in order to interface with new hardware

Part 2

Machines That Contain Operating Systems

Operating System Types

- Preset / Controller
  - An operating system constructed to perform specific duties. Commonly seen on real-time hardware devices like robot arms, cell phones, washing machines, etc.
- General Purpose
  - Designed to allow a human to construct and execute algorithms in a particular language under an executing environment.

Preset Devices

- Each button set to a single action
- Not general purpose
- Easy to use

OS = Mapping between key and function

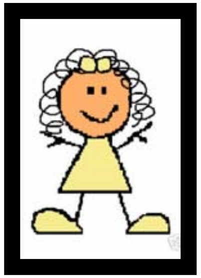
```
Selection = button();
Switch(Selection)?
{
   1: Menu(); break;
   2: Dial(); break;
   :
   :
}
```

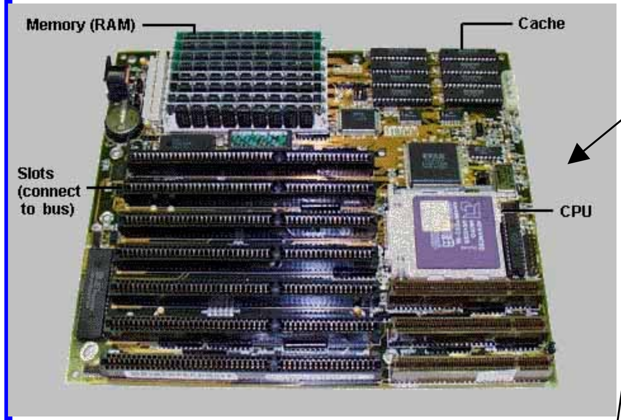
General Purpose OS

(Interface)?

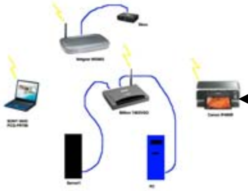
A Manager

The OS





Result: Life easier…



Applications / Languages

(Interface)?

So Many Different Kinds

Preset

- Cell Phone
- Calculator
- Dishwasher
- Gas pump
- ENIAC
- Real-time Systems

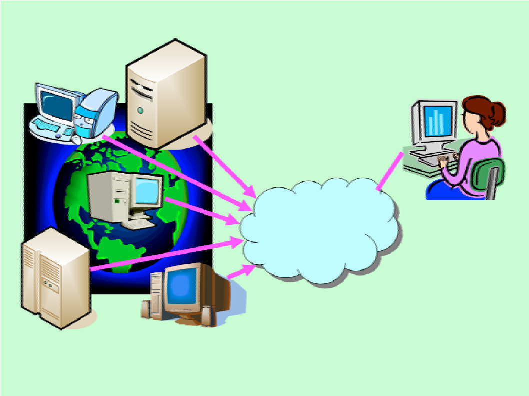
General Purpose

- Single CPU Single Process
- Single CPU Batch Process
- Single CPU Multi Process
- Multi CPU Single Process
- Multi CPU Multi Process
- Distributed CPU Single Process
- Distributed CPU Multi Process

Microsoft, Apple, Unix, …

Future Operating Systems

- Gear (Belt – Glasses – Earphone – Mic. - bluetooth)?
- Personal networks
- OS On A Stick (almost here…appliances!)?
- The laser keyboard & 3D Interfaces
- Shared Distributed OS (parts of OS on different computers!)?
- Code Migration Processing



Discuss…

Part 3

Things To Do

Research at McGill

• Compilers and Concurrency	• Prof. Laurie Hendren
• Networks	• Prof. Bettina Kemp Prof. Maheswaran
• Internet Technology	• Prof. Joseph Vybihal (also AI)?

At Home …

- Using any OS, identify the features it has and associate it with one area in the Basic OS Architecture.
- Know the following terminology:
  - Instruction Pointer
  - CPU and Program Execution Cycle
  - General Purpose OS, Preset OS and Controllers
  - BOOT
  - The Basic OS Architecture
- Have you every crashed your OS? How?
- What bug do you think exists in your OS?