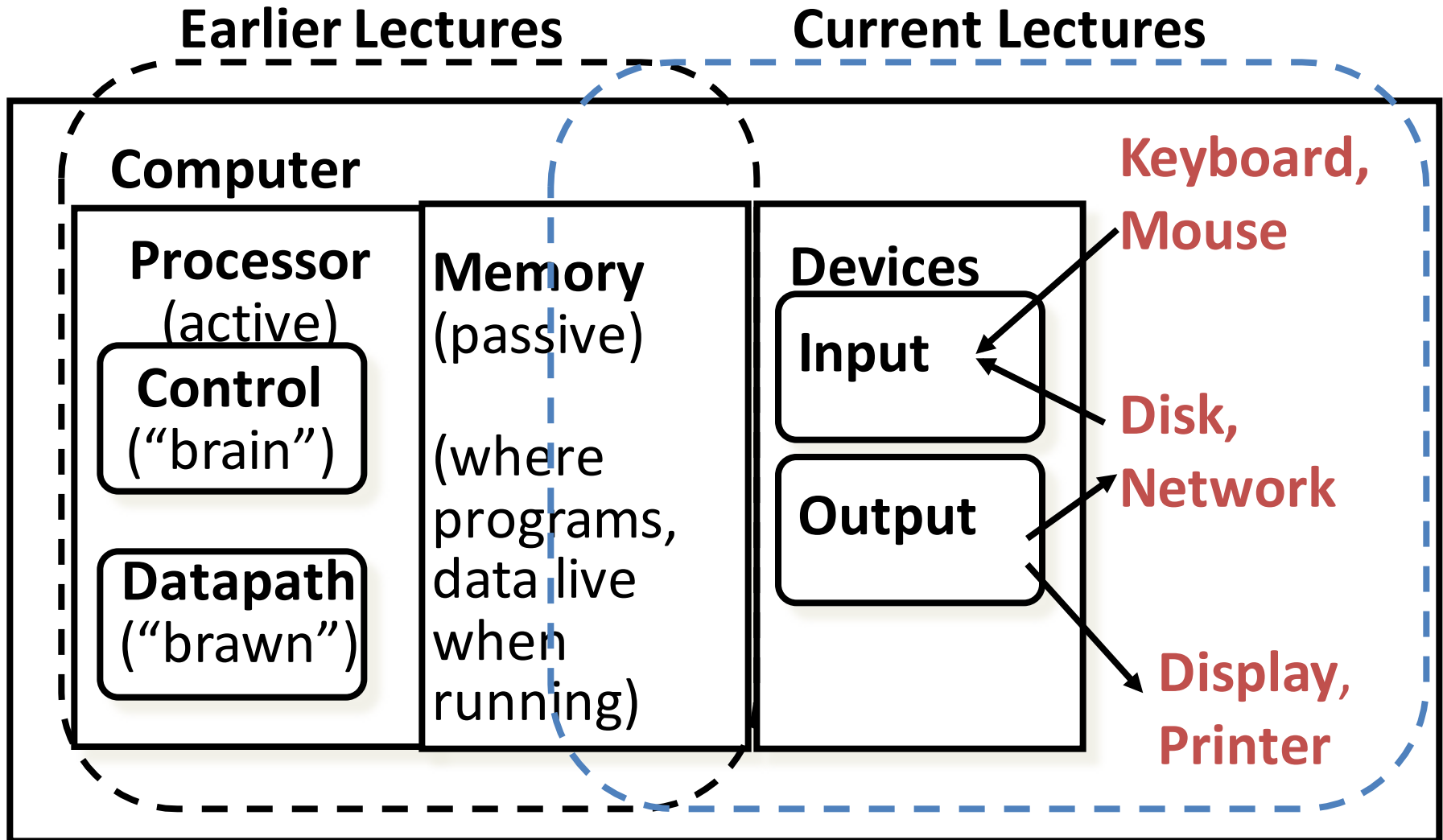
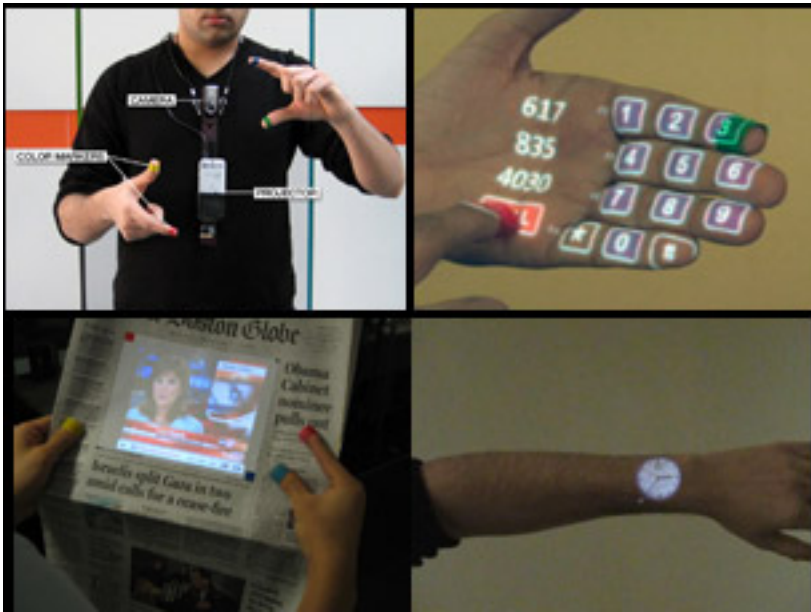


Recall : 5 components of any Computer



Motivation for Input/Output

- I/O is how humans interact with computers
- I/O gives computers long-term memory.
- I/O lets computers do amazing things:



MIT Media Lab

“Sixth Sense”

<http://youtu.be/ZfV4R4x2SK0>

- Computer without I/O like a car w/no wheels; great technology, but gets you nowhere

I/O Device Examples and Speeds

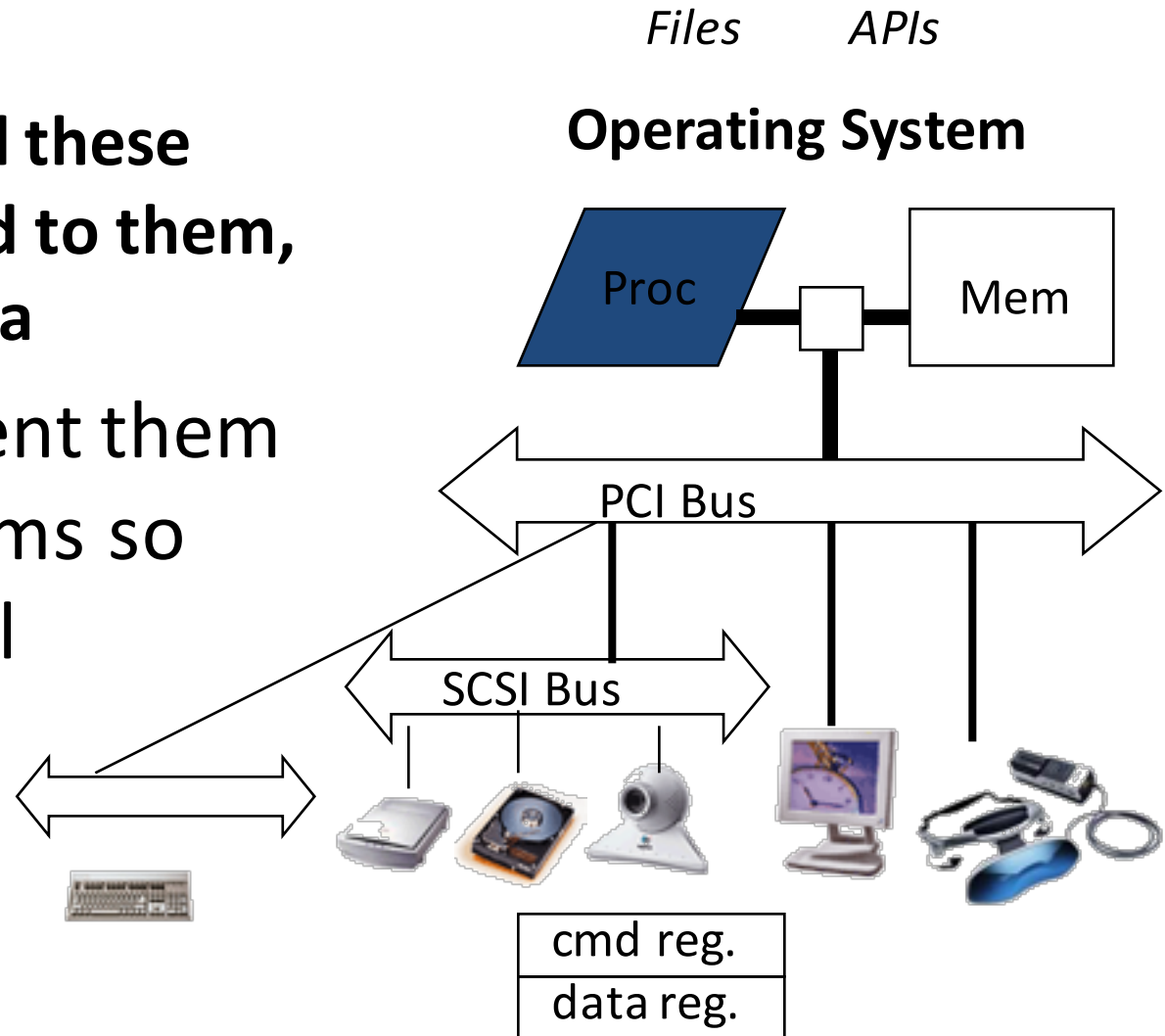
- I/O Speed: bytes transferred per second
(from mouse to Gigabit LAN: 7 orders of magnitude!)

Device	Behavior	Partner	Data Rate (KBytes/s)
Keyboard	Input	Human	0.01
Mouse	Input	Human	0.02
Voice output	Output	Human	5.00
Floppy disk	Storage	Machine	50.00
Laser Printer	Output	Human	100.00
Magnetic Disk	Storage	Machine	10,000.00
Wireless Network	I or O	Machine	10,000.00
Graphics Display	Output	Human	30,000.00
Wired LAN Network	I or O	Machine	125,000.00

When discussing transfer rates, use 10^x

What do we need to make I/O work?

- A way to connect many types of devices
- A way to control these devices, respond to them, and transfer data
- A way to present them to user programs so they are useful

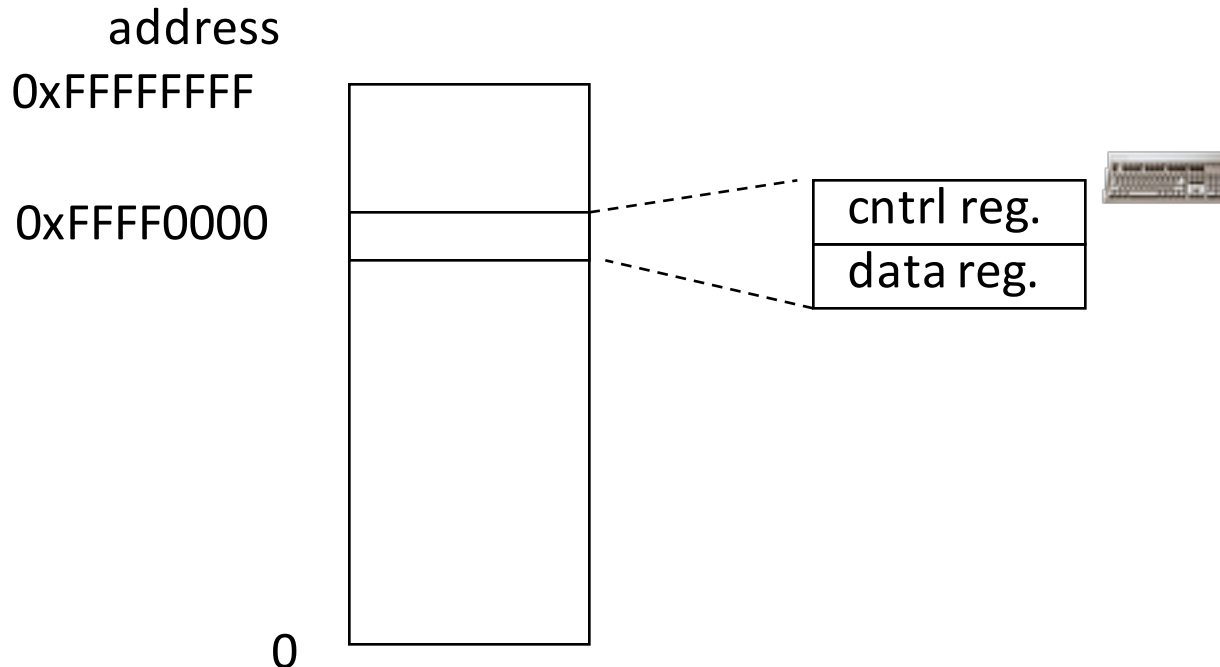


Instruction Set Architecture for I/O

- What must the processor do for I/O?
 - Input: reads a sequence of bytes
 - Output: writes a sequence of bytes
- Some processors have special input and output instructions
- Alternative model (used by MIPS):
 - Use loads for input, stores for output (in small pieces)
 - Called **Memory Mapped Input/Output**
 - A portion of the address space dedicated to communication paths to Input or Output devices (no memory there)

Memory Mapped I/O



- Certain addresses are not regular memory
- Instead, they correspond to registers in I/O devices



Processor-I/O Speed Mismatch

- 1GHz microprocessor can execute 1 billion load or store instructions per second, or 4,000,000 KB/s data rate
 - I/O devices data rates range from 0.01 KB/s to 125,000 KB/s
- Input: device may not be ready to send data as fast as the processor loads it
 - Also, might be waiting for human to act
- Output: device not be ready to accept data as fast as processor stores it
- What to do?

Processor Checks Status before Acting

- Path to a device generally has 2 registers:
 - **Control Register**, says it's OK to read/write (I/O ready) [think of a flagman on a road]
 - **Data Register**, contains data
- Processor reads from Control Register in loop, waiting for device to set **Ready** bit in Control reg (0  1) to say its OK
- Processor then loads from (input) or writes to (output) data register
 - Load from or Store into Data Register resets Ready bit (1  0) of Control Register
- This is called "**Polling**"

I/O Example (polling)

- Input: Read from keyboard into \$v0

```
Waitloop:    lui    $t0, 0xffff #ffff0000
             lw     $t1, 0($t0) #control
             andi   $t1, $t1, 0x1
             beq    $t1, $zero, Waitloop
             lw     $v0, 4($t0) #data
```

- Output: Write to display from \$a0

```
Waitloop:    lui    $t0, 0xffff #ffff0000
             lw     $t1, 8($t0) #control
             andi   $t1, $t1, 0x1
             beq    $t1, $zero, Waitloop
             sw     $a0, 12($t0) #data
```

“Ready” bit is from processor’s point of view!


Cost of Polling?

- Assume for a processor with a 1GHz clock it takes 400 clock cycles for a polling operation (call polling routine, accessing the device, and returning). Determine % of processor time for polling
 - Mouse: polled 30 times/sec so as not to miss user movement
 - Floppy disk (Remember those?): transferred data in 2-Byte units and had a data rate of 50 KB/second. No data transfer can be missed.
 - Hard disk: transfers data in 16-Byte chunks and can transfer at 16 MB/second. Again, no transfer can be missed. (we'll come up with a better way to do this)

% Processor time to poll

- Mouse Polling [clocks/sec]
= 30 [polls/s] * 400 [clocks/poll] = 12K [clocks/s]
- % Processor for polling:
 $12 * 10^3 \text{ [clocks/s]} / 1 * 10^9 \text{ [clocks/s]} = 0.0012\%$
☞ Polling mouse little impact on processor

% Processor time to poll hard disk

- Frequency of Polling Disk
 $= 16 \text{ [MB/s]} / 16 \text{ [B/poll]} = 1\text{M [polls/s]}$
- Disk Polling, Clocks/sec
 $= 1\text{M [polls/s]} * 400 \text{ [clocks/poll]}$
 $= 400\text{M [clocks/s]}$
- % Processor for polling:
 $400 * 10^6 \text{ [clocks/s]} / 1 * 10^9 \text{ [clocks/s]} = 40\%$
 Unacceptable

(Polling is only part of the problem – main problem is that accessing in small chunks is inefficient)

What is the alternative to polling?

- Wasteful to have processor spend most of its time “spin-waiting” for I/O to be ready
- Would like an unplanned procedure call that would be invoked only when I/O device is ready
- Solution: use **exception mechanism** to help I/O. **Interrupt** program when I/O ready, return when done with data transfer

Peer Instruction

- 1) A faster CPU will result in faster I/O.
- 2) Hardware designers handle mouse input with interrupts since it is better than polling in almost all cases.
- 3) Low-level I/O is actually quite simple, as it's really only reading and writing bytes.

	123
A:	FFF
B:	FFT
B:	FTF
C:	FTT
C:	TFF
D:	TFT
D:	TTF
E:	TTT

Peer Instruction Answer

- 1) Less sync data idle time
- 2) Because mouse has low I/O rate polling often used
- 3) Concurrency, device requirements vary!

- TRUE**
- 1) A faster CPU will result in faster I/O.
 - 2) Hardware designers handle mouse input with interrupts since it's better than polling in almost all cases.
 - 3) Low-level I/O is actually quite simple, as it's really only reading and writing bytes.
- FALSE**

	1	2	3
A:	F	F	F
B:	F	F	T
B:	F	T	F
C:	F	T	T
C:	T	F	F
D:	T	F	T
D:	T	T	F
E:	T	T	T

“And in conclusion...”

- I/O gives computers their 5 senses + long term memory
- I/O speed range is 7 Orders of Magnitude (or more!)
- Processor speed means must synchronize with I/O devices before use
- Polling works, but expensive
 - processor repeatedly queries devices
- Interrupts work, more complex
 - we’ll talk about these next
- I/O control leads to **Operating Systems**