Exceptional Control Flow

Exceptional Control Flow (Chapter 8)

I Context

CPU ⟷ Infinite loop of (Fetch, Decode, Execute)

Most of the time Next Instruction is valP

Rest (that we know about)

- jump
- call
- ret

} This non-linear change in control is

React to changes in program state

Q: Why is this not sufficient for a system

A $\boxed{\text{system}}$ is a set of interacting / interdependent component parts forming a complex/intricate whole

In computer systems,

- Processor
- I/O Devices
  - Disk – File system
  - Network
  - GPU / Mouse / Keyboard
- OS / kernel
- Processes

Look at Process List

Distributed -- change the boundary
Networking

A: Need to "react" to changes in $\boxed{\text{system state}}$

Data arrives from Disk, Network, Keyboard, Graphics Card + Divide by 0, Access illegal mem

Ctrl+C at keyboard

System timer expires ⟷ Run a different process

Exceptional Control Flow

-- change instruction sequence to respond to changes in system state

asynchronous synchronous

Exist at all levels of a computer system

HW/OS	1. Exceptions/Interrupts
OS	2. Process context switch
OS/Process	3. Signals
Process	4. Non local jumps direct : setjump / longjump HLL : C++ / Java Exception try catch