Actors Presentation

- Increasing processor speed
  - How CPU works
    - clock creates square wave, propagates through logic
    - component synchronization is pretty amazing
  - Issues
    - wave is rounded off by capacitance
    - wave is distorted by crosstalk
    - spends time between extremes: heat
  - How to increase performance?
    - Increase frequency
      - distortion takes bigger bite
      - more power to dissipate (demonstrate with hands)
      - slew rate: more heat
      - lightspeed propagation
        - nanosecond is just less than a foot long
        - 3GHz, ~1/3ns, ~4in per cycle
        - ~2in per half cycle
      - size reduction
        - severely reduced die yield
        - increased capacitance and crosstalk, distortion
  - Adding cores
    - Till recently, hardware guys took brunt of Moore's Law
    - Now they need our help
    - Almost nobody smart enough to write good concurrent code
    - Search for good concurrency primitives
      - Shared memory sucks unless transactional (Clojure)
- Actors
  - message passing
  - mailbox
  - named: customized or automatic
  - constructor parameters, but should be serializable
  - mostly dependent on nothing but other actors
  - *not* functional programming: OO, more than most

        - no interface but mailbox
        - contains mutable state but doesn't share it
        - no real types, at least functionally: ActorRef
            - Not RPC: no return values
            - Typed actors available in Akka but not recommended
- Messages
    - always immutable
    - should be serializable
- Distributable
    - Each actor has associated URL; messages immutable and serializable
    - Better security: use secure transport layer like SSL
- Restartable
    - Every user actor has a supervisor—either system or parent.
    - Supervisor has method that responds to exceptions in 4 ways
        - Resume subordinate, keeping state
        - Restart subordinate, clearing state
        - Terminate subordinate permanently
        - Escalate failure—that is, fail
- Conway
    - Explain canonical problem
    - Explain actor version
    - Show statistics
    - Perhaps do stats run live
    - Explain given-up-on version, tradeoffs, complexity
- Code
    - Coordinator
        - extends Actor
        - takes factory for testing
        - results from companion object apply ()
        - ActorSystem
        - receive: Start, Complete
            - Like to call handler methods from receive
            - Dissect message classes
        - Start: creates patches, spawns PatchHandlers, sender
        - Complete: synchronizes current gen, starts next
    - PatchHandler
        - Outer appearance very similar to Coordinator

    - Both show as ActorRefs; hard to tell difference
    - ActorRefs, Props
        - Can be recreated automatically upon failure
        - Can be replicated automatically, locally and remotely
    - receive: very simple
    - Borders somewhat more complexity
- Tests
    - CoordinatorTest
        - Extends TestKit(system)
        - ImplicitSender
        - Can still have path.FunSpec
        - implicit ActorSystem
        - Real Coordinator actor being dealt with
    - unit tests
    -