CS 201 Lecture 7
Finite State Machines

Spring 2014
Finite State Machines

• There is always a limit to how much you can do in one clock cycle
• Think of finite state machines as hardware programs
  – Each state is like a “statement”, where you define what happens for a clock cycle you are in that state
Finite State Machine

- Control Unit
  - Status
  - Current State / Control Signals
- Functional Unit / Datapath
  - Results
Multiplier Example

• Compute the multiplication of two unsigned numbers using repeated addition
  – \( A \times B = A + A + A \ldots + A \) (\( B \) times)

• Inputs
  – A, B, Start
    • A and B should be saved at the time Start is set to 1

• Outputs
  – Result, Done
    • Result is only valid if Done is 1
Multiplier Control Unit

• Inputs
  – Start, Status from function unit (is the multiplication done?), asynchronous Reset

• Outputs
  – Control signal to begin multiplication (Load)
  – Current state
Multiplier State Diagram

- **Reset**
  - **Idle / Start**
    - Start = 1 /
      - Load = 1
  - **Multiplying**
    - Start = 1 /
      - Load = 1
  - **Done**
    - Done = 1

**Arc** = state transition (in/out)
**Oval** = current state

© John A. Stratton, 2014
Multiplier Control Unit Design

• Assign a number / bit pattern to each state
• Compute next state values based on current state and inputs
• Compute flip-flop inputs based on desired state transition
• Compute outputs based on current state and inputs