

18-642:

Modal Systems & Statecharts

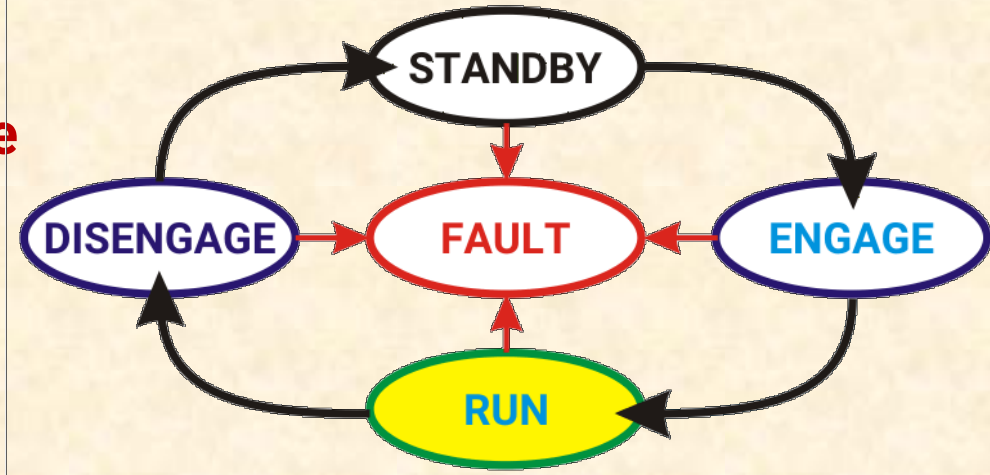
1/31/2018



State Intensive Systems

■ Anti-Patterns:

- No detailed design – just code
- Deeply nested if statements instead of switch statements for state-full code
- Mixing mode change logic with normal output sequences



■ Detailed design of state-intensive behaviors

- Operating modes, e.g., stop, start, run
- Inputs that drive sequences of events
- Key technique: statecharts (software finite state machine)

3-Speed Fan FlowChart

■ Example code for 3-speed fan

- Draw a flowchart -- how easy is it to understand this code?
- Are there any bugs in this code?

```
// SPDBUTTON:  input true on cycle when speed button depressed
// ONOFF:      input true one cycle when on/off switch depressed
static uint8_t speed; // 0=off; 1=slow; 2=medium; 3=fast

.....
if(speed == 0)
{ if(SPDBUTTON == 1 || ONOFF == 1) { speed = 1; }
} else if (SPDBUTTON == 1)
{ if (speed == 1) { speed = 2; }
  else if (speed == 2) { speed = 3;}
  else { speed = 0;}
} else if (ONOFF == 1)
{ speed = 0;}
```

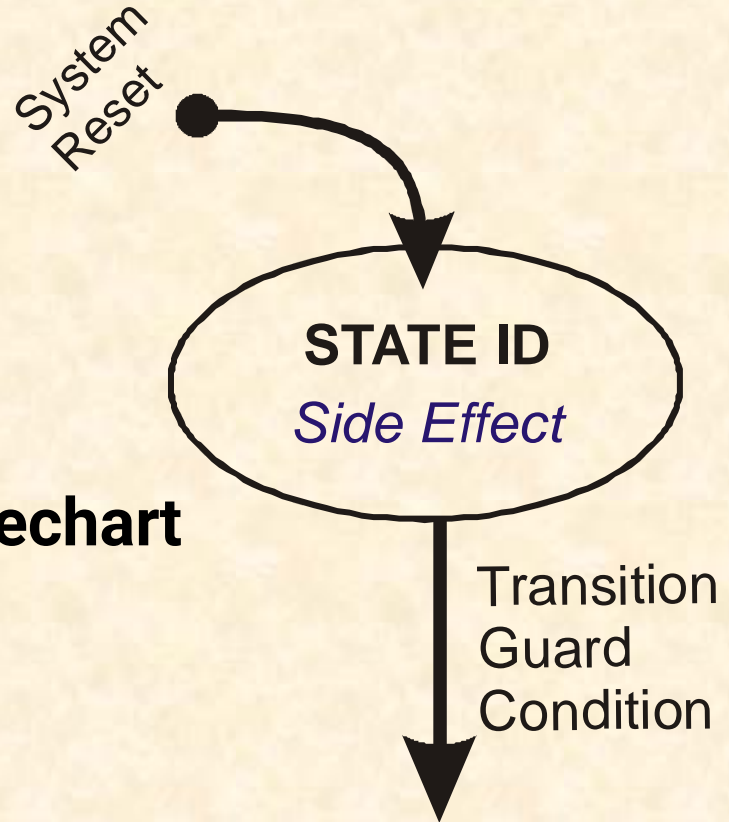
Elements of a Statechart

■ A statechart is a software Finite State Machine:

- Set of states with side effects
- Set of guards that cause transitions
 - No side effects on transitions
- Initial state

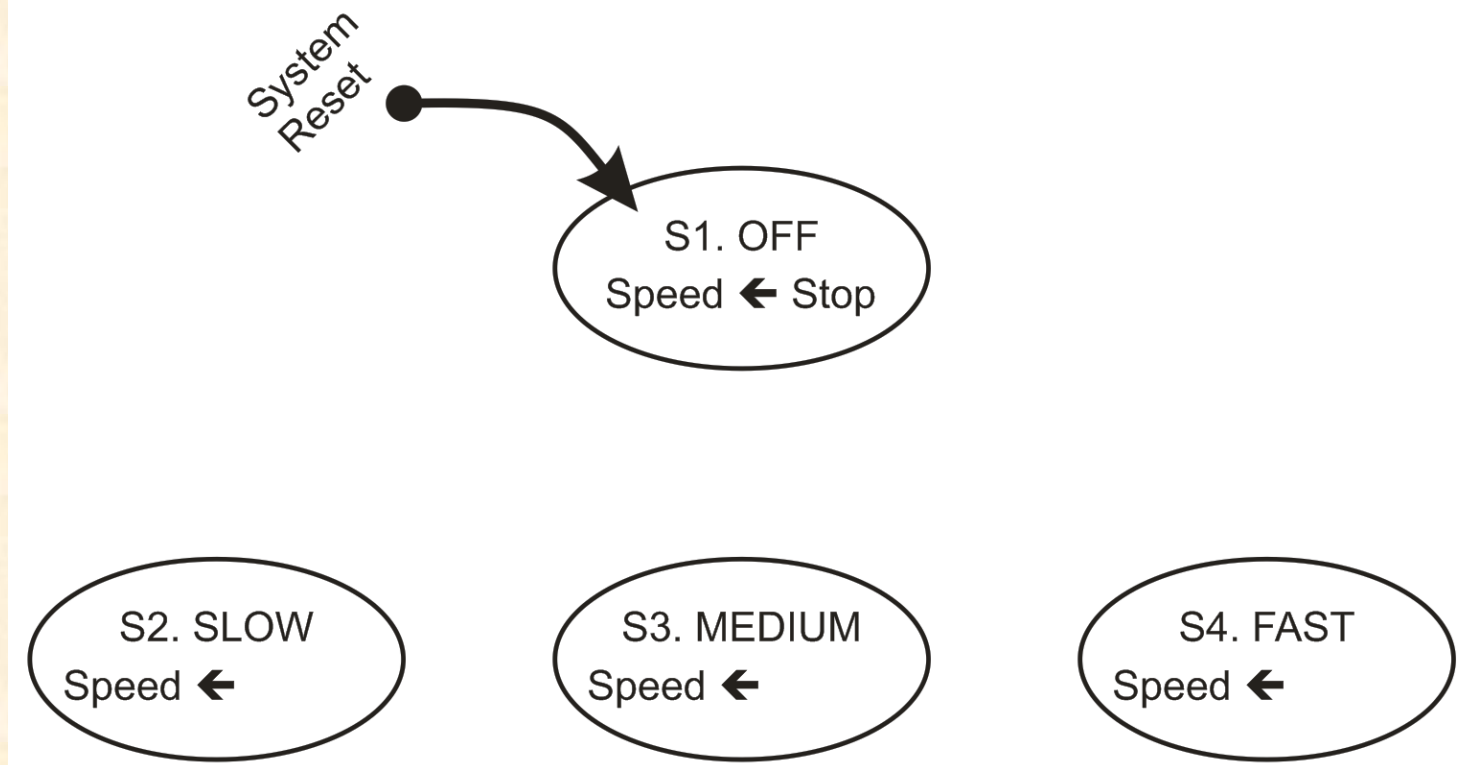
■ Convert example fan code to statechart

- (Next slide has graphic)
- Define a state for each fan speed
- Define transitions
- Easier to understand? Any bugs?

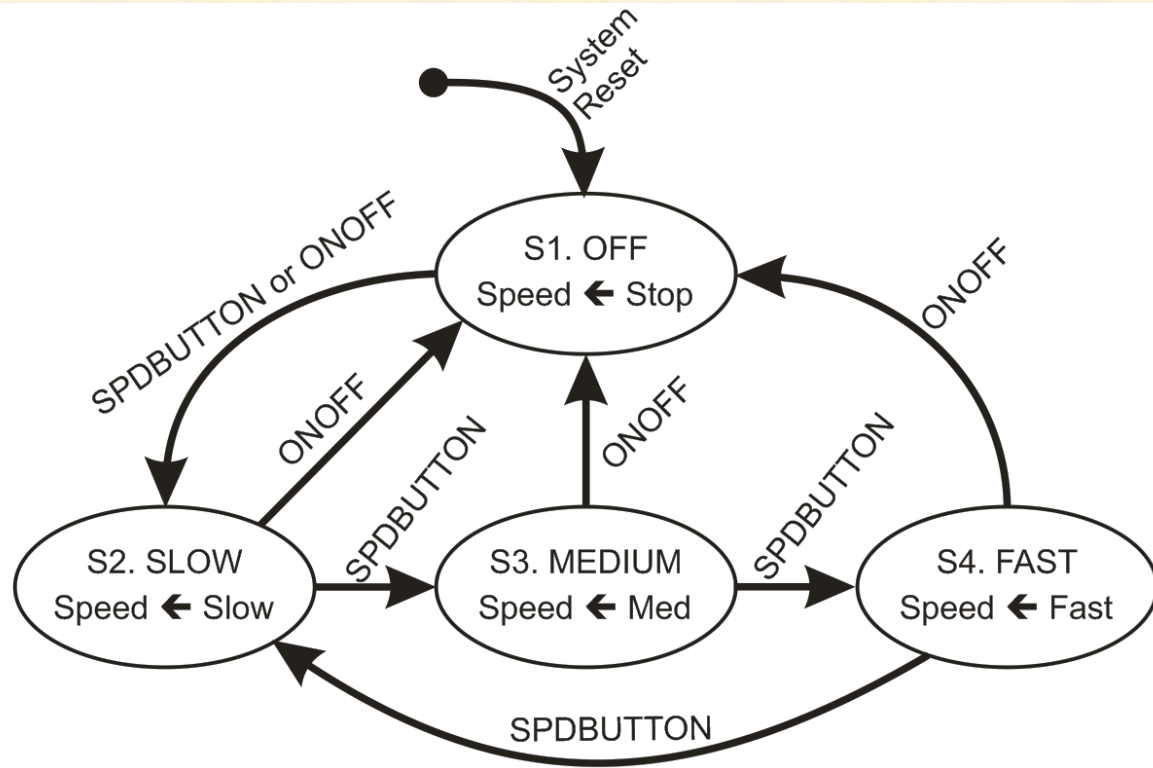


Convert to StateChart Exercise: 3-speed fan

- Define initial state, side effects, transitions



Corrected Statechart



■ This is a controller for a multi-speed motor or other similar application

- Inputs: SPDBUTTON, ONOFF
- Outputs: Speed = {Stop, Slow, Med, Fast}
- State names (arbitrary labels): {OFF, SLOW, MEDIUM, FAST}
- System Reset is to state s1

Fan Example Code – part 1

```
static enum CurrState {OFF, SLOW, MEDIUM, FAST}; // define states
static const uint8_t SpdOff=0; // define speed constant values
static const uint8_t SpdSlow=10;
static const uint8_t SpdMed=15;
static const uint8_t SpdFast=25;
CurrState = OFF; // initialize state machine to OFF
void ProcessStates(void) // run periodically from main loop
{ switch (CurrState)
  { case OFF: // State S1
    speed(SpdOff); // Take action in state
    // Test arc guards and take transitions
    if (SpdButton() == TRUE || OnOffButton() == TRUE) {CurrState = SLOW;}
    break; // go to end of switch statement
  case SLOW: // State S2
    speed(SpdSlow); // take action
    if (SpdButton() == TRUE) {CurrState = MEDIUM;}
    if (OnOffButton() == TRUE) {CurrState = OFF;}
    break;
```

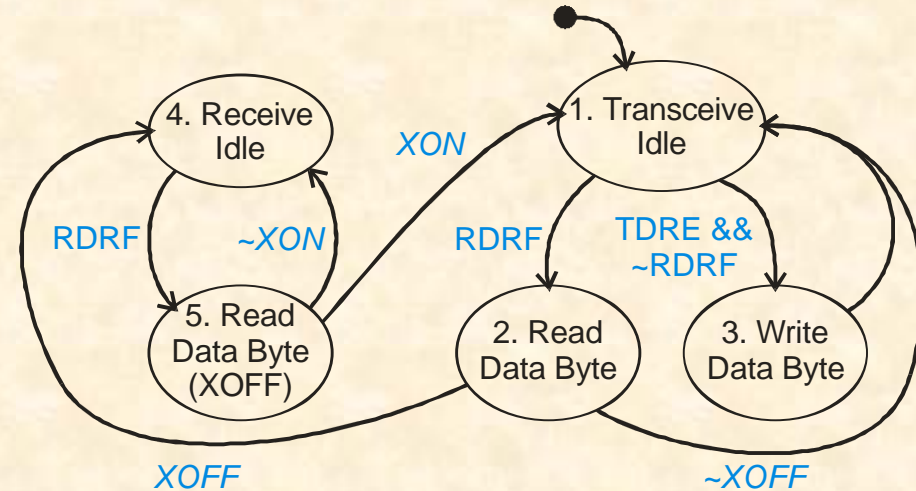
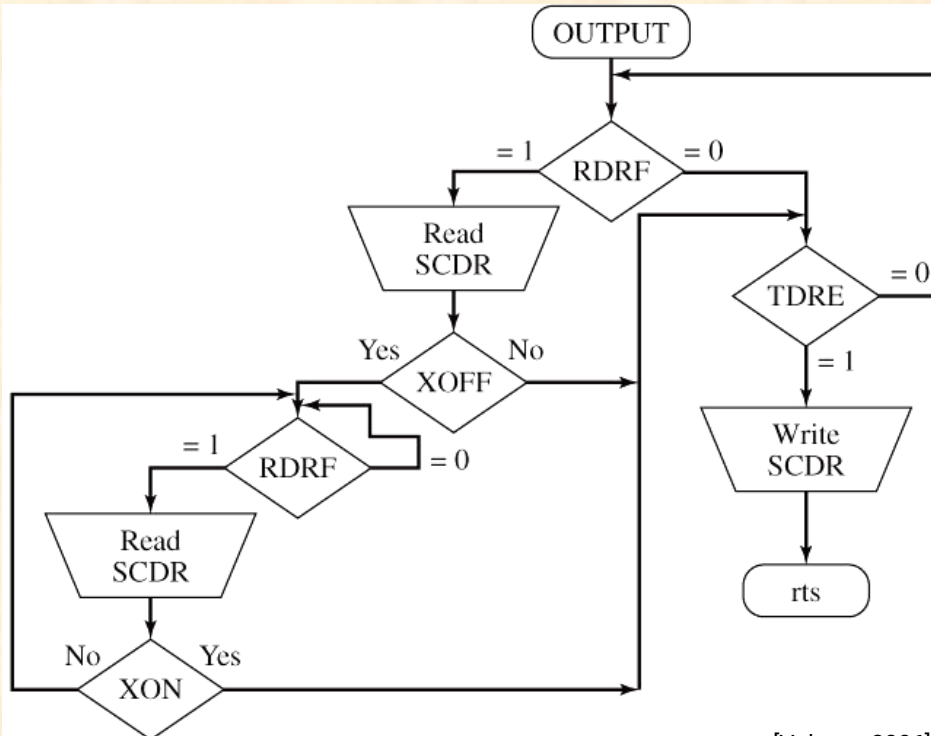
Fan Example Code – part 2

```
case MEDIUM: // State S3
    speed(SpdMed); // take action
    if (SpdButton() == TRUE) {CurrState = FAST;}
    if (OnOffButton() == TRUE) {CurrState = OFF;}
    break;
case FAST: // State S4
    speed(SpdFast); // take action
    if (SpdButton() == TRUE) {CurrState = SLOW;}
    if (OnOffButton() == TRUE) {CurrState = OFF;}
    break;
default: // Error: invalid state
    error(INVALID_STATE_ERROR); // should never get here
}
}
```


Half-Duplex Serial Port Example

RDRF = "Receive Data Register Full" → Data byte arrived
TDRE = "Transmit Data Register Empty" → Done sending

SCDR = "Serial Comms. Data Reg."
XON/XOR → Flow Control



Best Practices for Statecharts

■ Use statecharts for stateful code

- Maps to easier-to-test switch statement
- Avoid actions on arcs to simplify code
- Move complex behaviors to per-state subroutine helper functions to limit cyclomatic complexity



<https://goo.gl/ocnSRS>

■ Summary of pitfalls

- Some code is better as flowchart if there is no state history
- Don't let statechart get too complex
 - Might need to decompose into nested or parallel state machines