

In-Vehicle Networking

Lecture 3 Behavior Modeling – State Charts
BAE 5030 - 003
Fall 2008
Instructor: Marvin Stone
Biosystems and Agricultural Engineering
Oklahoma State University

Behavioral Modeling

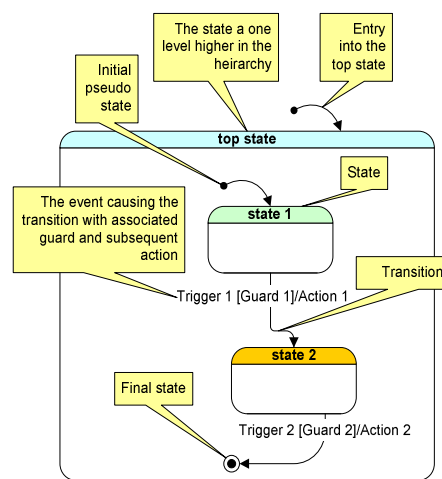
- Logic tables – a tabular relationship between all inputs and outputs. Inputs and outputs can include parameters in memory
- Statecharts – modeling technique suited to describing systems where past state or memory defines the transitions to the next state
 - Mealy automata – processes occur on transitions between states
 - Moore automata – processes occur within a state
 - UML - Unified Modeling Language – specifies a syntax for state machine modeling

Statechart definitions

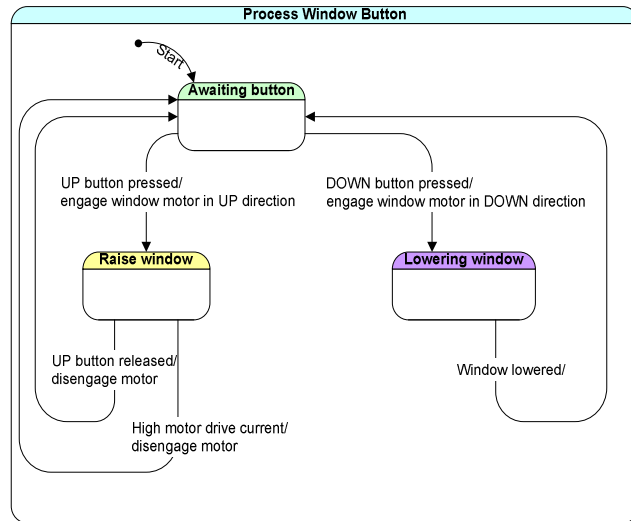
- *State* - a distinguishable consistent characteristic behavior of a system that persists for a significant period of time
- *Event* – a change in stimulus to a system that occurs in an infinitesimal period of time
- *Transitions* - responses to events that move the system from state to state
- *Finite State Machine* – A model of a system and its behavior that captures states, events, and transitions between states
- Finite State Machine characteristics
 - Characteristic behavior within a *state* is distinct and unchanging
 - Finite number of *states*
 - Residence time in *states* is a significant period
 - *Transitions* are the response of the system to *events*
 - *Transitions* between *states* are distinct and unchanging
 - *Transitions* between *states* are finite in number
 - *Transition time* between *states* is infinitesimal

Statechart elements and syntax

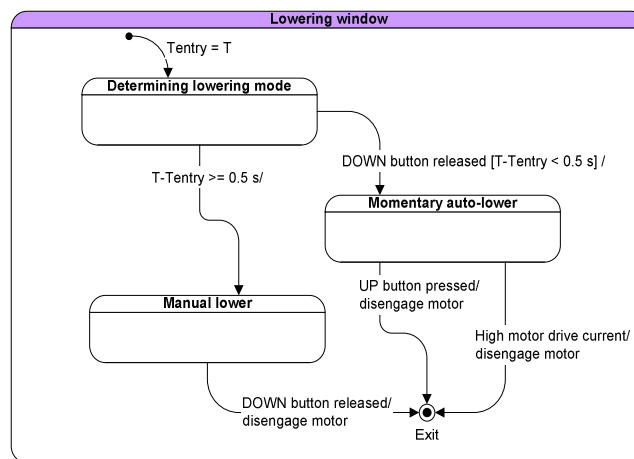
- Start occurs at the initial pseudo state
- The trigger event causes the change of state if the guard condition is true
- System may exit the top state when the final state is reached
- Time between states is infinitesimal
- Elements (Simple statecharts)
 - Initial state
 - State
 - Transition
 - Event
 - Trigger
 - Guard
 - Action
 - Final state



Statechart Example

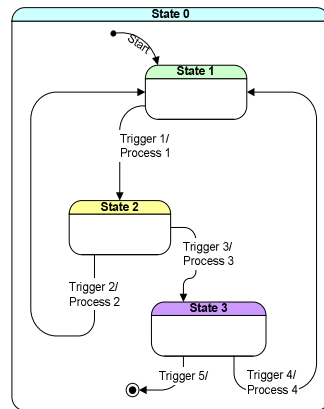


Hierarchical State Example



Software structure to realize a state machine

- Many possibilities but typical is infinite loop encompassing a case structure. (see Samek)
- Consider the following example:



```

void Process_State_0( void )
{
  state = STATE_1
  while(!Trigger_5 & (state == STATE_3)){
    switch (state) {
      case STATE_1:
        if (Trigger_1) {
          process_1();
          state = STATE_2
        }
        break;
      case STATE_2:
        if (Trigger_2) {
          process_2();
          state = STATE_2
        }
        if (Trigger_3) {
          process_3();
          state = STATE_2
        }
        break;
      case STATE_3:
        if (Trigger_4) {
          process_4();
          state = STATE_1
        }
    }
  }
}
  
```

Concurrent tasks

