This document is meant to provide an organizational structure to the Stateflow Workshop to help instructors. Refer to this document while presenting the workshop as a handy brief reference guide to keep track of the motivation, necessary actions, and Stateflow concepts presented during each step of building up the Stateflow chart.

1. Show model overview
   1. Explain tunable parameters: on/off and fan switches, usage level, ambient temperature
   2. Show GUI
      1. Run model and play with GUI settings to get attendees comfortable with model
2. Add basic Stateflow chart

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Represent main states of system

**Action:** Create two states “DeviceOff” and “DeviceOn”

**Concept:** Exclusive states

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Define when system should switch from one state to another

**Action:** Add transitions “[switch\_on]” and “[!switch\_on]”

**Concept:** Transitions

**------------------------------------------------------------------------------------------------------------------------------**

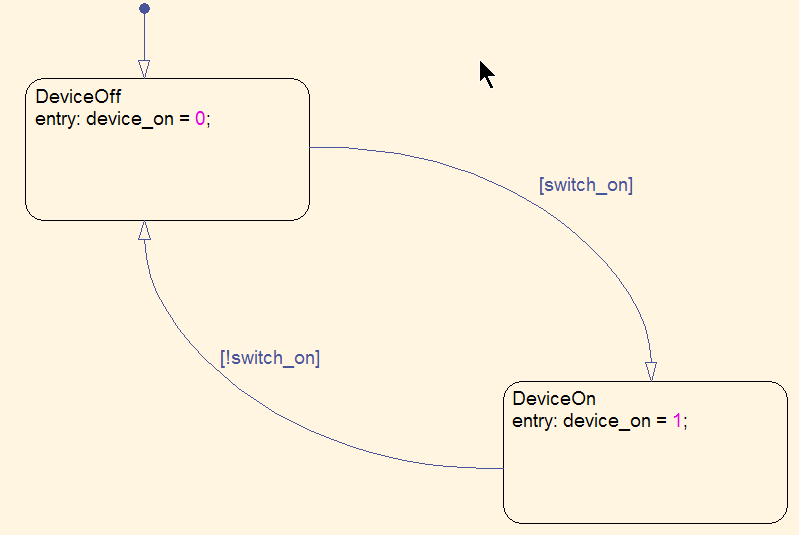
**Motivation:** Tell Simulink when device is on or off

**Actions:** Create entry actions “entry: device\_on = 0;” and “entry: device\_on = 1;”

Create variables, connect chart to Simulink model

*Run and test simulation*

**Concepts**: State entry actions, default transition, interface with Simulink, early verification, Symbol Wizard



**------------------------------------------------------------------------------------------------------------------------------**

1. Protect against overheating

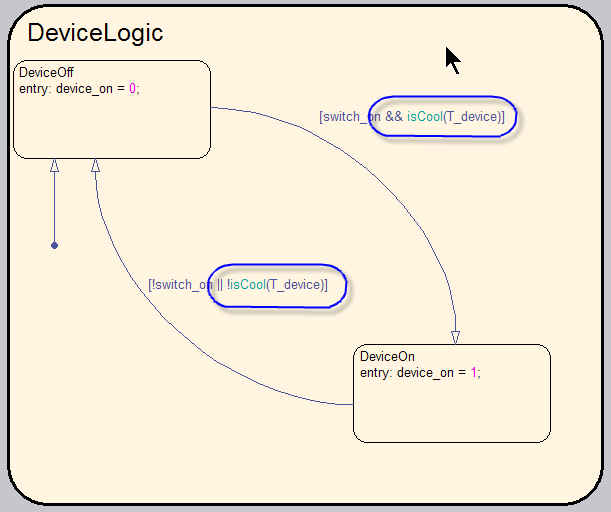
**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Need logic to activate fan when device gets too hot

**Action:** Add “&&isCool(T\_device)” and “||!isCool(T\_device)” to transitions

Drag T\_device signal to T\_device input in Simulink

**Concept:** Complex transitions



**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Create room in chart to create isCool function

**Action:** Add superstate “DeviceLogic” to contain existing states

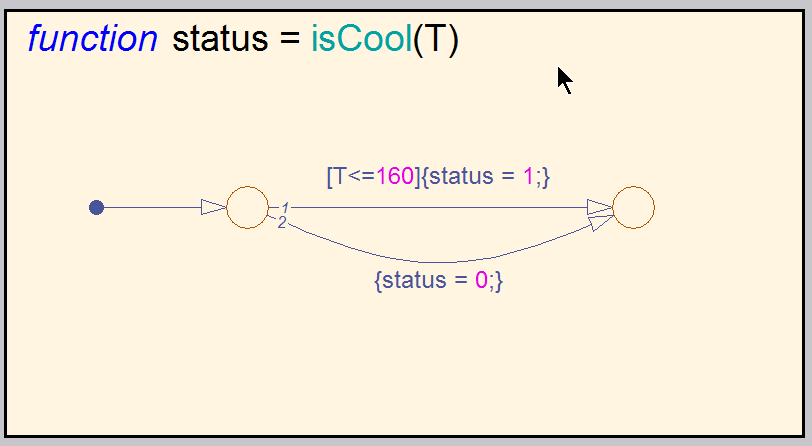
**Concept:** Hierarchical states, subcharts

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Need to define isCool function

**Action:** Create graphical function isCool with “[T<=160]{status=1;}” for one path and {status = 0;} for the other path

**Concept:** Graphical functions, transition priorities



**------------------------------------------------------------------------------------------------------------------------------**

1. Add mechanism to turn fan on and off

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Need logic to turn fan on and off

**Action:** Create new state FanLogic

**Concept:** Parallel states

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Define modes for fans

**Actions:** Create states “FanOff” and “FanOn”

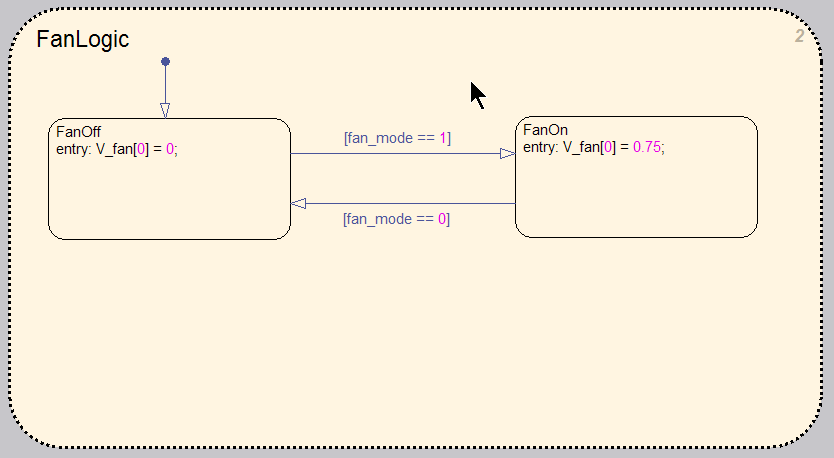
Create transitions “[fan\_mode==1]” and “[fan\_mode==0]”

Create entry actions “entry: V\_fan[idx]=0” and “entry: V\_fan[idx]=0.75”

Create default transition “{idx=0;}”

Define variables and output V\_fan to Voltage Scale Slider block

**Concept:** Referencing arrays



**------------------------------------------------------------------------------------------------------------------------------**

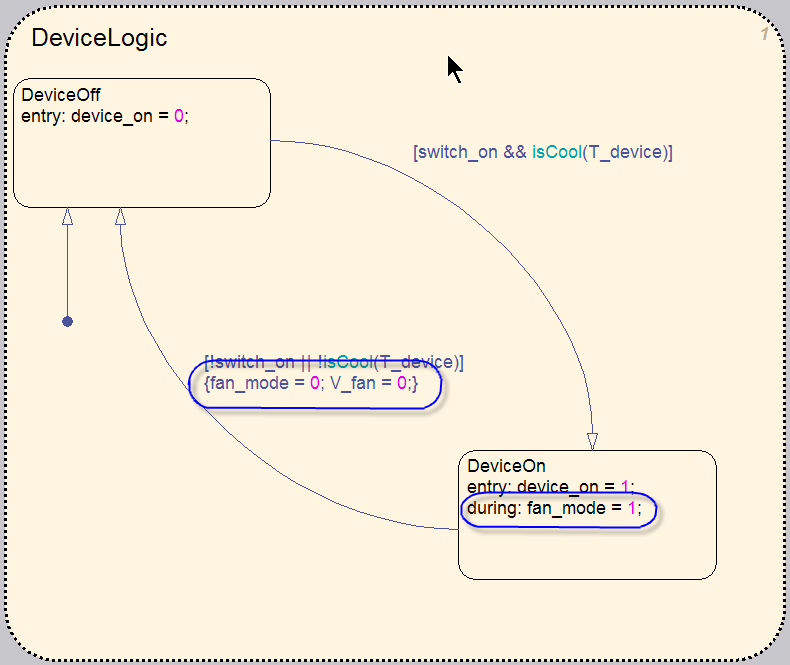
**Motivation:** Turn on fan from DeviceLogic state

**Action:** Create during action “during:fan\_mode=1;” in “DeviceOn” state

Create condition action “{fan\_mode=0}” on outgoing transition from “DeviceOn”

*Run and test simulation*

**Concept:** State during actions



**------------------------------------------------------------------------------------------------------------------------------**

1. Add second fan speed

**------------------------------------------------------------------------------------------------------------------------------**

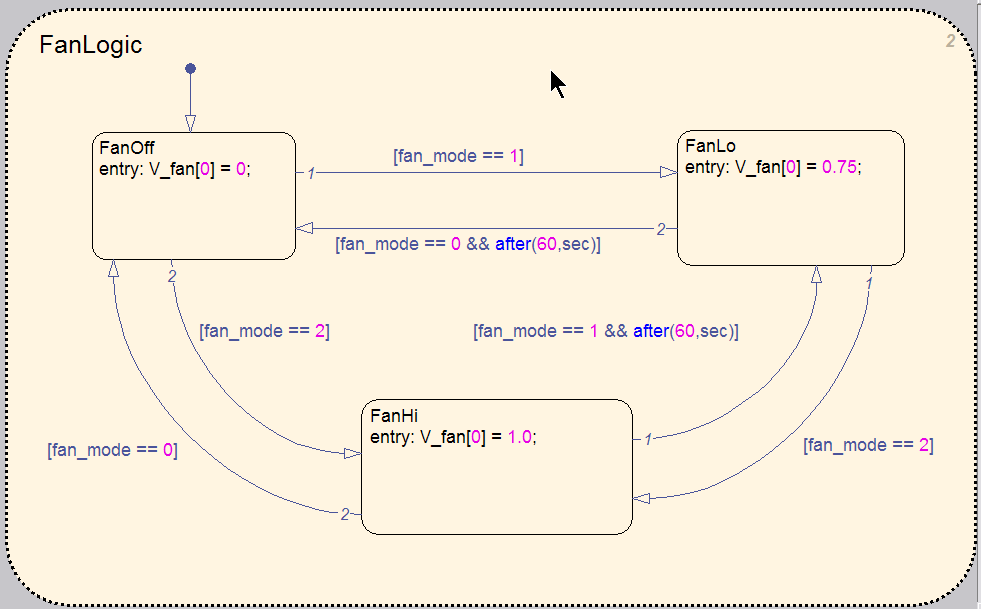
**Motivation:** Allow for fan to go high if temperature goes above 120 degrees

**Action:** Change “FanOn” state to “FanLo”

Add “FanHi” state with entry action “entry:V\_dan[idx] = 1.0];

Add FanHi ingoing transitions [fan\_mode == 2] from other two states

Add outgoing transitions [fan\_mode == 0] and [fan\_mode == 1] to “FanOff” and “FanLo”, respectively

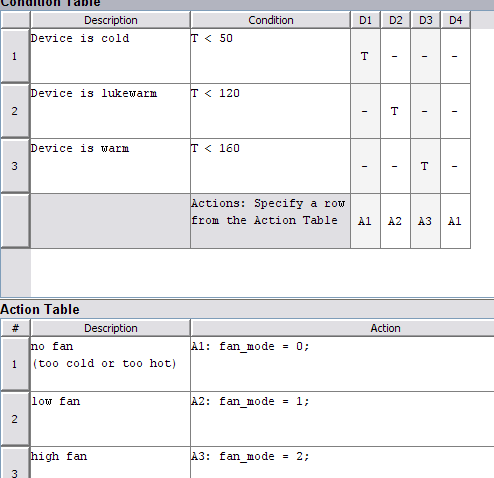


**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Define when to switch to FanOff, FanLo, or FanHi

**Action:** Create “FanTable” truth table function with conditions “T<50”, “T<120”, “T<160” and actions “fan\_mode=0;” “fan\_mode=1;”, and “fan\_mode=2;”

**Concept:** Truth table functions



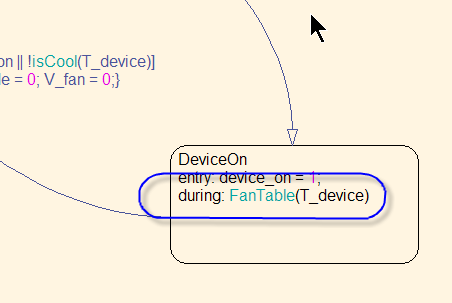
**------------------------------------------------------------------------------------------------------------------------------**

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Call truth table function

**Action:** Change during action to “during: FanTable(T\_device)”

*Run and test simulation*



**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Create time delay so fan does not turn off and on repeatedly

**Action:** Add “&& after(60,sec)” to FanLo->FanOff and FanHi->FanLo transitions

**Concept:** Temporal logic

*Run and test simulation*

**------------------------------------------------------------------------------------------------------------------------------**

1. Add backup fan

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Create mechanism to trigger a backup fan

**Action:** Enclose contents of FanLogic into state “FanSystem” and subchart it

Add history junction to “FanSystem”

Create self-transition to FanSystem

**Concept:** History junctions, self-transitions

**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Trigger backup fan if primary fan is broken

**Action:** Add following to self-transition: “[isFanBroken(fan\_mode,fan\_rpm[0],t) && idx == 0]{ V\_fan[0] = 0.0; idx = 1;}

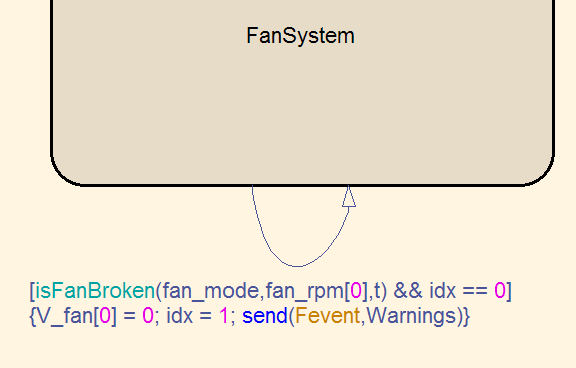
Define embedded MATLAB function isFanBroken:

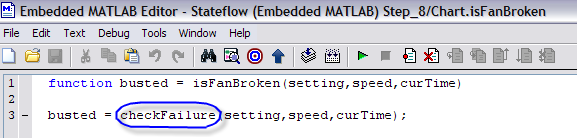
function busted = isFanBroken(setting,speed,curTime)

busted = checkFailure(setting,speed,curTime);

Drag RPM\_fan signal to fan\_rpm input in Simulink model

*Run and test simulation*

**Concept:** Embedded MATLAB functions



**------------------------------------------------------------------------------------------------------------------------------**

1. Add warning system

**------------------------------------------------------------------------------------------------------------------------------**

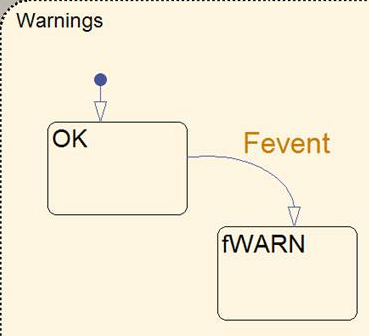
**Motivation:** Alert operator if primary fan fails

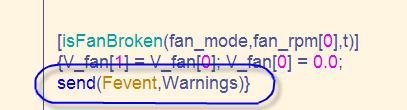
**Action:** Add parallel state named “Warnings” with states “OK” and “fWARN”

Add event “Fevent” to transition from “OK” to “fWARN”

Add “send(Fevent,Warnings)” to FanSystem self-transition

**Concept:** Event broadcasting





**------------------------------------------------------------------------------------------------------------------------------**

**Motivation:** Send warning information to Simulink model

**Action:** Output state activity of fWARN state

Connect fWARN output to a Display block

*Run and test simulation*

**Concept:** Output state activity

**------------------------------------------------------------------------------------------------------------------------------**