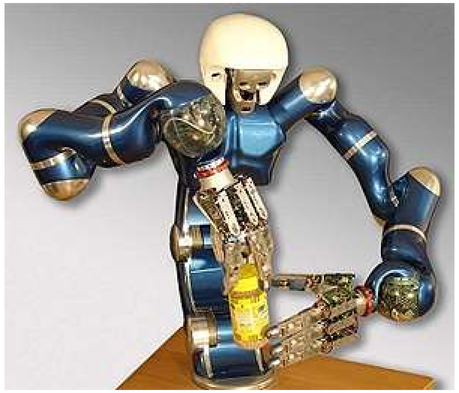
Control System Design for Automated Vehicles

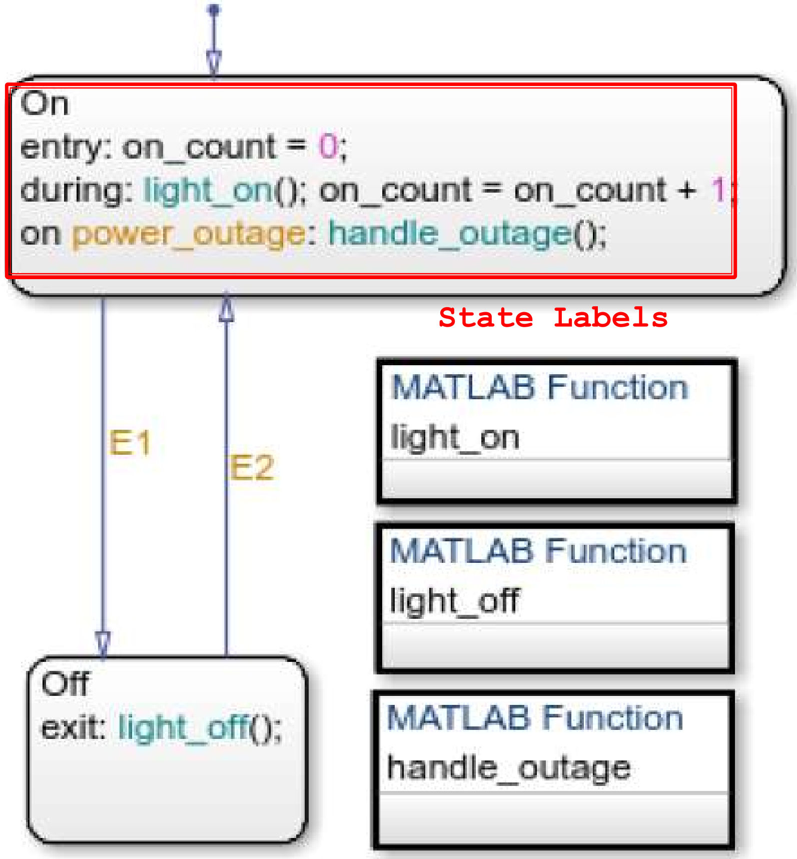
Lecture 10



State Labels

- Labels for a state appears on the top left corner of the state rectangle with the following format:

```
name/
entry:entry actions
during:during actions
exit:exit actions
on event_name:on event_name actions
on message_name:on message_name
actions
bind:events
```

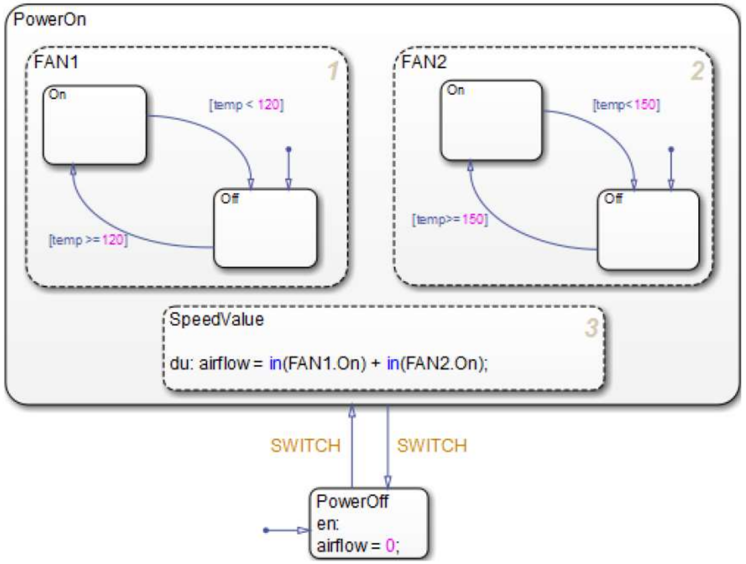


```
On
entry: on_count = 0;
during: light_on(); on_count = on_count + 1;
on power_outage: handle_outage();
```

```
Off
exit: light_off();
```

State Name

- A state label starts with the name of the state followed by an optional / character.
- State Name Example
  - PowerOn.Fan1.On
  - PowerOn.Fan1.Off
  - PowerOn.Fan2.On
  - PowerOn.Fan2.Off



State Action

- After the name, you enter “optional” action statements for the state.

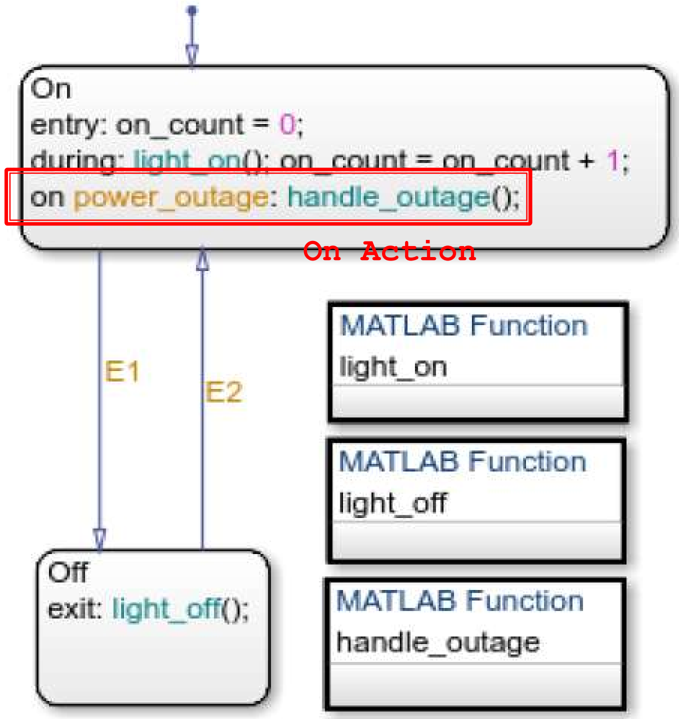
- Entry Action
  - Preceded by the prefix “entry” or “en” for short.
  - Executed whenever the state becomes active.

- During Action
  - Preceded by the prefix “during” or “du” for short.
  - Executed whenever
    - the state is already active, a new time step occurs and no valid transition to another state is available.
    - The state is already active, an event occurs and no valid transition to another state is available.

State Action

- Exit Action
  - Preceded by the prefix “exit” or “ex” for short.
  - Executed when the state was active, but becomes inactive.

- On Action
  - Preceded by the prefix “on event_name” or “on message_name”.



```
On
entry: on_count = 0;
during: light_on(); on_count = on_count + 1;
on power_outage: handle_outage();
```

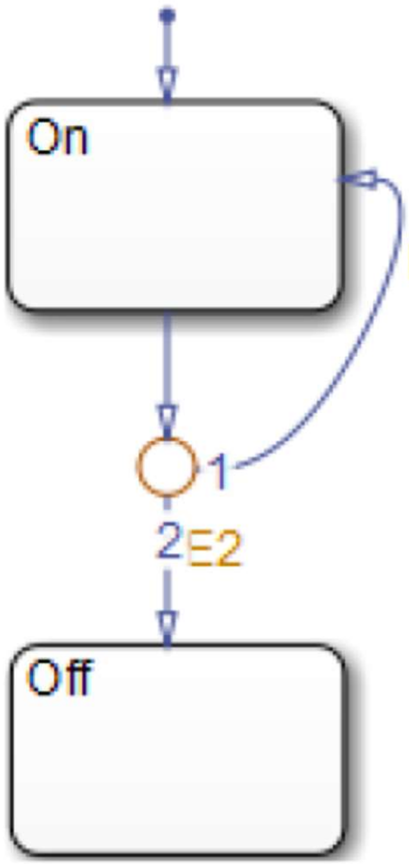
```
Off
exit: light_off();
```

Transition

- Represents the passage of the system from one mode (state) to another.
- Transition connects a “source object” and a “destination object”.
- Default transition does not have source objects.

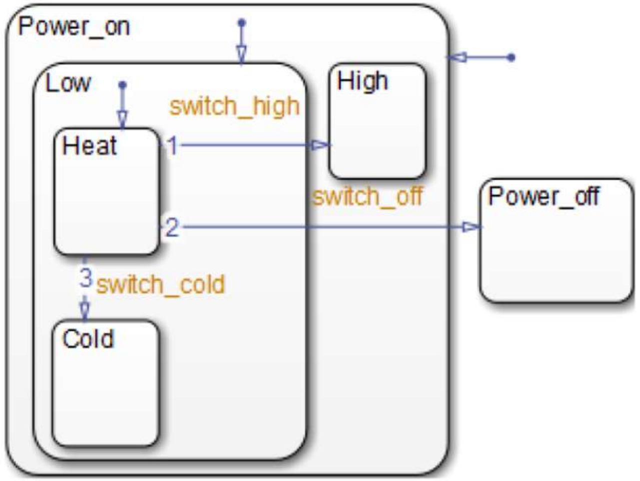
Junction

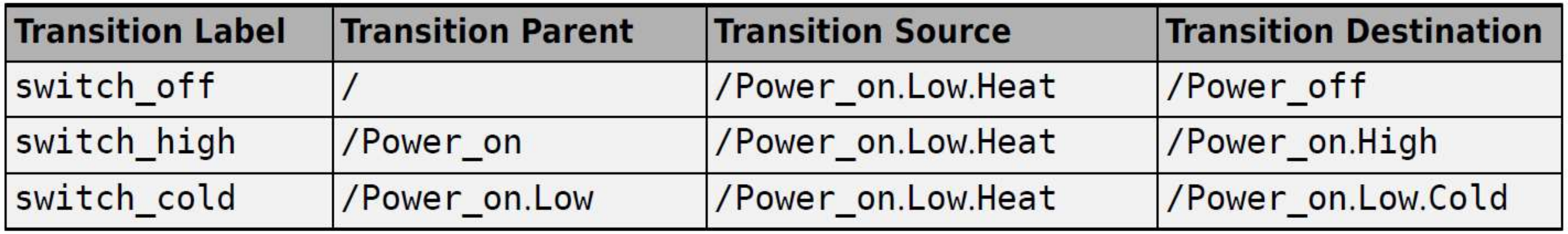
- Junction divide a transition into transition segments
  - During the transition, each segment is evaluated to determine the validity of a full transition.



Transition Hierarchy

- The hierarchy for a transition is described in terms of its parent, source, and destination states.

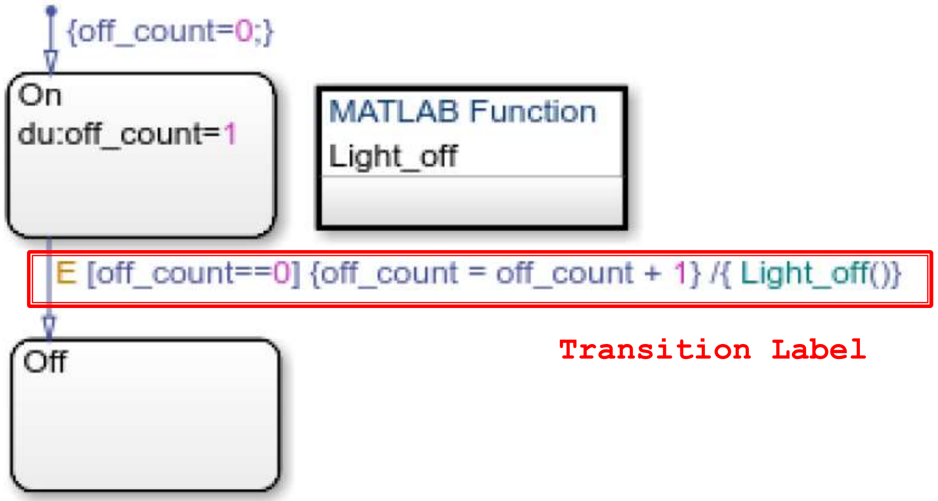




Transition Label	Transition Parent	Transition Source	Transition Destination
switch_off	/	/Power_on.Low.Heat	/Power_off
switch_high	/Power_on	/Power_on.Low.Heat	/Power_on.High
switch_cold	/Power_on.Low	/Power_on.Low.Heat	/Power_on.Low.Cold

Transition Label

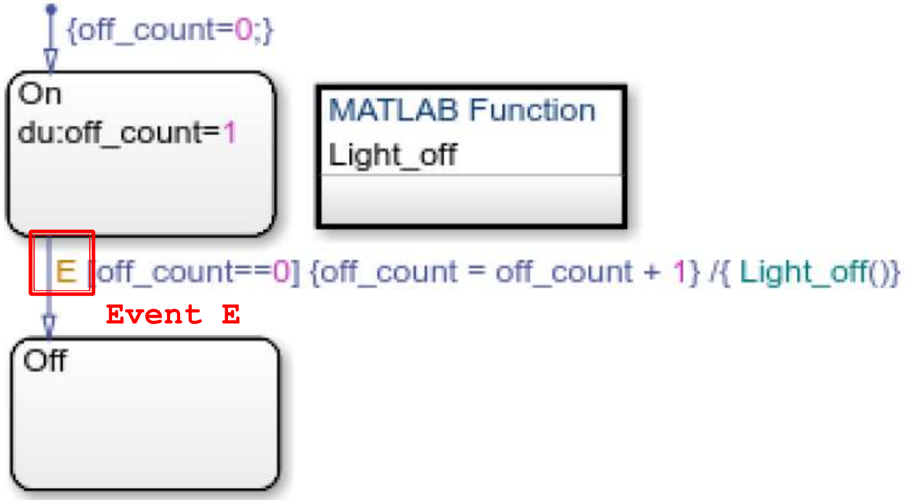
- Transition “label” consist of an event or message, a condition and a transition action.
- Transition label format
  - event_or_message[condition]{condition_action}/transition_action
- If condition is “not” specified, an implied condition evaluates to true.


{off_count=0;}
On
du:off_count=1
MATLAB Function
Light_off
E [off_count==0] {off_count = off_count + 1} /{ Light_off()}
Off

Transition Label

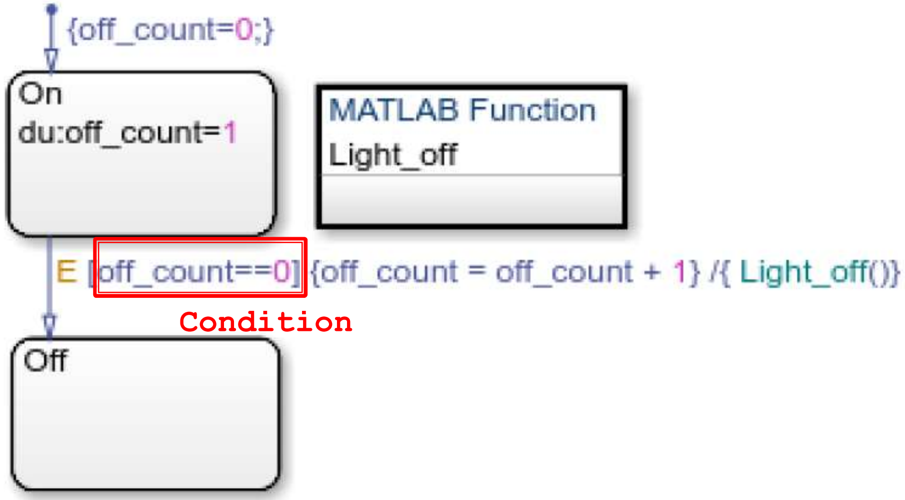
Transition Label

- Event or Message Trigger
  - Specifies an event or massage that causes the transition to occur when the condition is true.
  - In this example, the broadcast of event “E” triggers the transition from “On” to “off” if the condition [off_count == 0] is true.



Transition Label

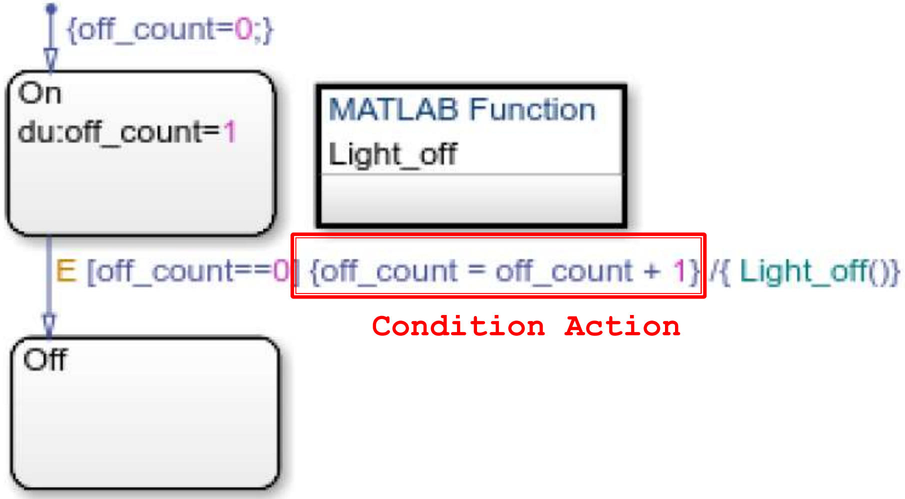
- Condition
  - A Boolean expression that validates a transition for the specified event.
  - Define the condition in squared brackets ([]).
  - If no condition is specified, an implied condition evaluates to true.



```
{off_count=0;}
du:off_count=1
E [off_count==0]{off_count = off_count + 1} /{ Light_off()}
```

Transition Label

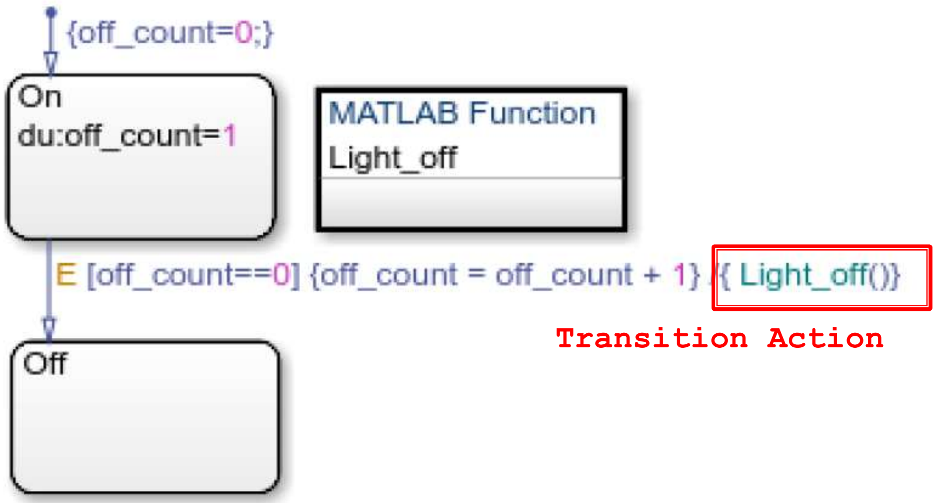
- Condition Action
  - Executes after the condition for the transition is evaluated as true.
  - In the previous example, if the event “E” occurs and the condition “[off_count == 0]” is true, then the condition action “{off_count = offcount + 1}” is immediately executed.



Transition Label

- Transition Action
  - Executes after the transition to the destination is determined to be valid.
  - In the previous example, if the event “E” occurs and the condition “[off_count == 0]” is true, then the transition action “{Light_Off()}” is executed when the transition from “On” to “Off” is determined to be valid

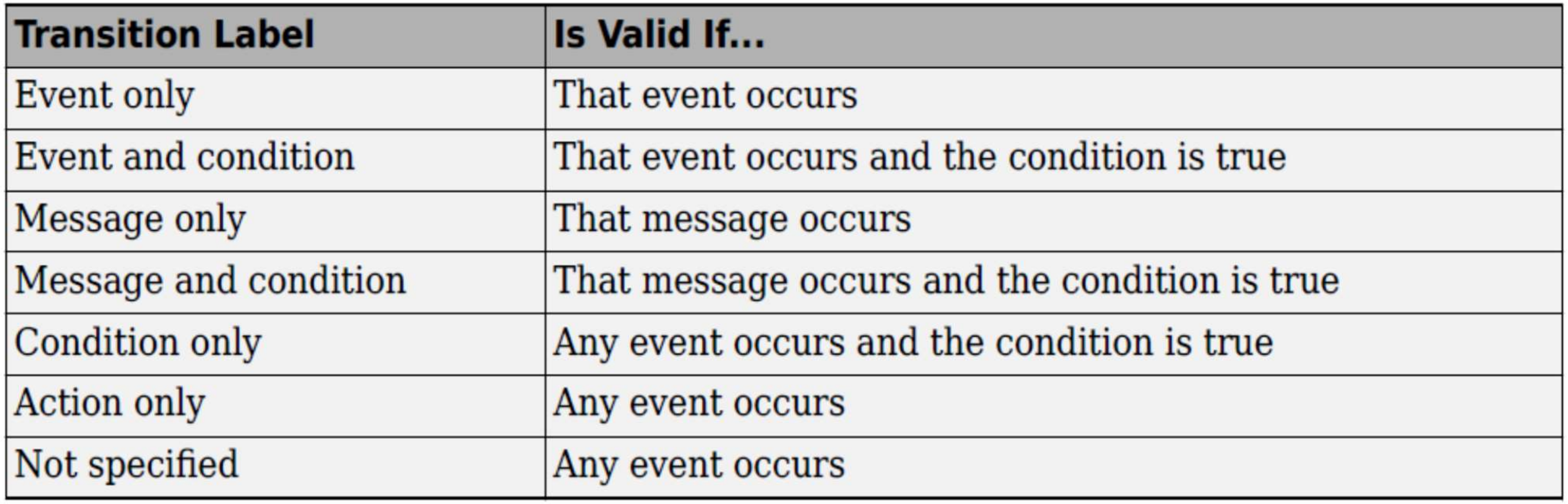
- The transition action occurs after “On” becomes inactive, but before “Off” becomes active.


{off_count=0;}
On
du:off_count=1
MATLAB Function
Light_off
E [off_count==0] {off_count = off_count + 1} /{ Light_off()}
Off

Transition Action

Valid Transition

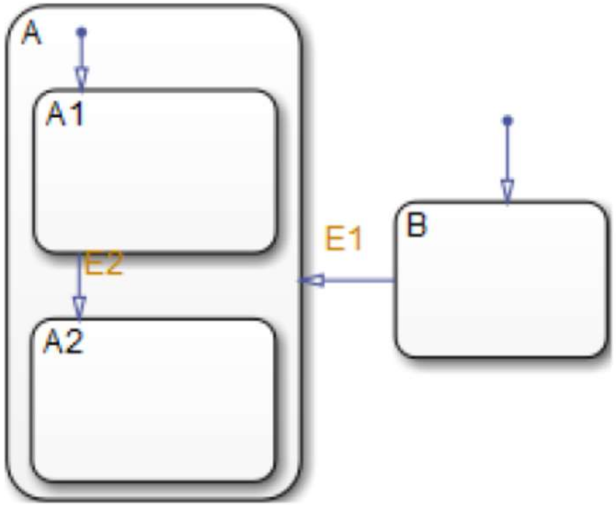
- The following table lists possible combination of valid transition labels.

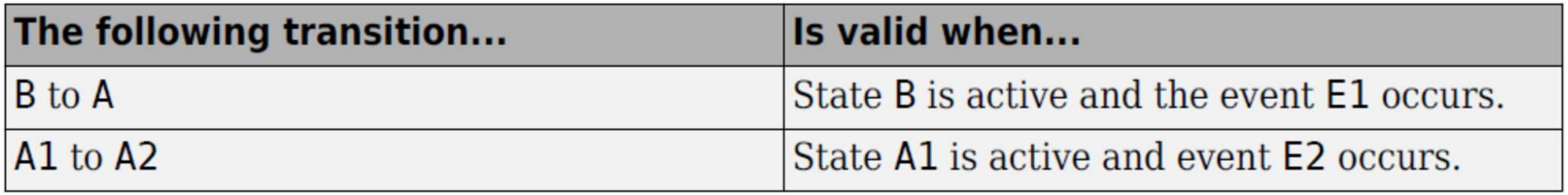


Transition Label	Is Valid If...
Event only	That event occurs
Event and condition	That event occurs and the condition is true
Message only	That message occurs
Message and condition	That message occurs and the condition is true
Condition only	Any event occurs and the condition is true
Action only	Any event occurs
Not specified	Any event occurs

Example

- This example shows simple transitions to and from exclusive (OR) states.

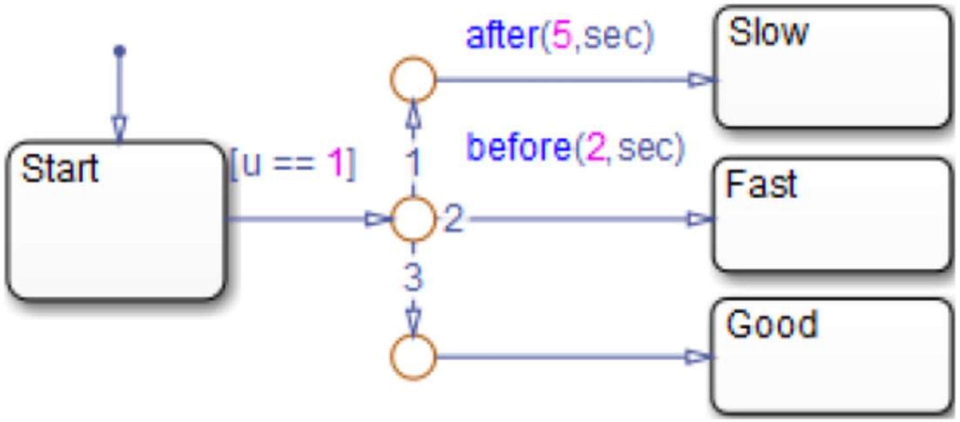


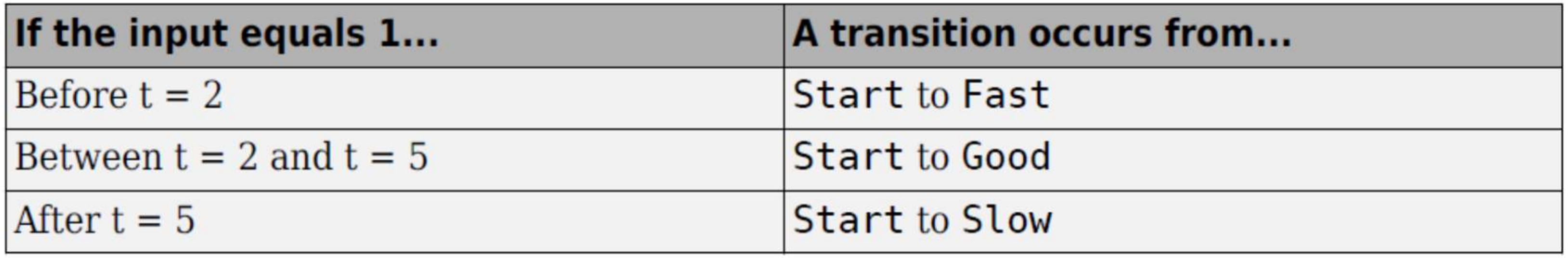


The following transition...	Is valid when...
B to A	State B is active and the event E1 occurs.
A1 to A2	State A1 is active and event E2 occurs.

Example

- The following chart shows transitions to and from connective junctions
- The chart uses temporal logic to determine when the input “u” equals 1.



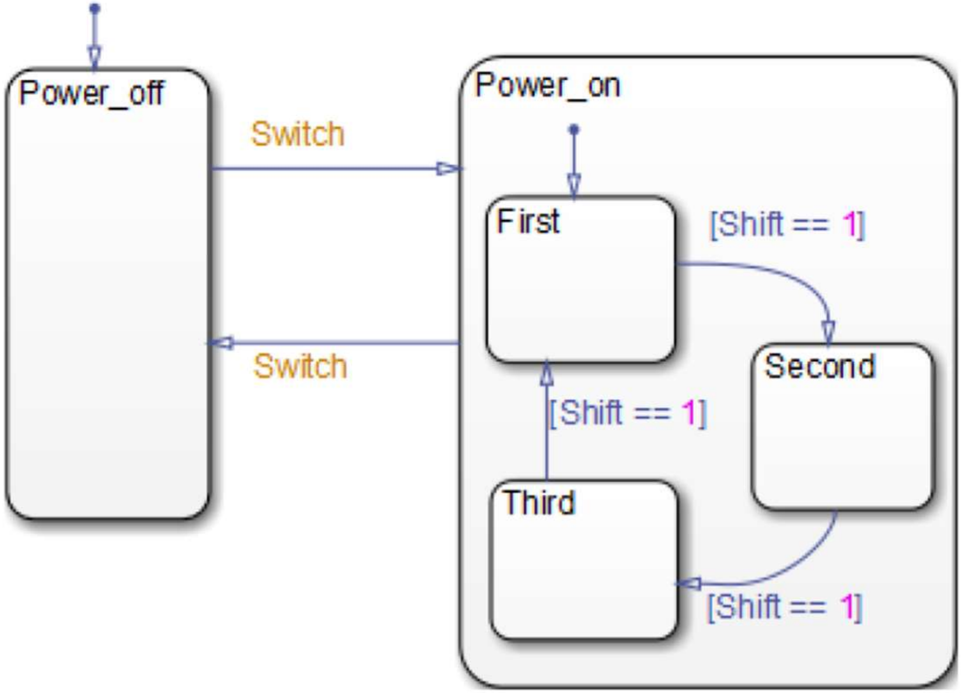


If the input equals 1…	A transition occurs from…
Before t = 2	Start to Fast
Between t = 2 and t = 5	Start to Good
After t = 5	Start to Slow

Transition to and from Exclusive (OR) Superstates

- Example
  - “Power_off” ➔ “Power_on” by “Switch” event.
  - “Power_on” has three substates: “First”, “Second”, and “Third”.
  - By default, “First” becomes active.

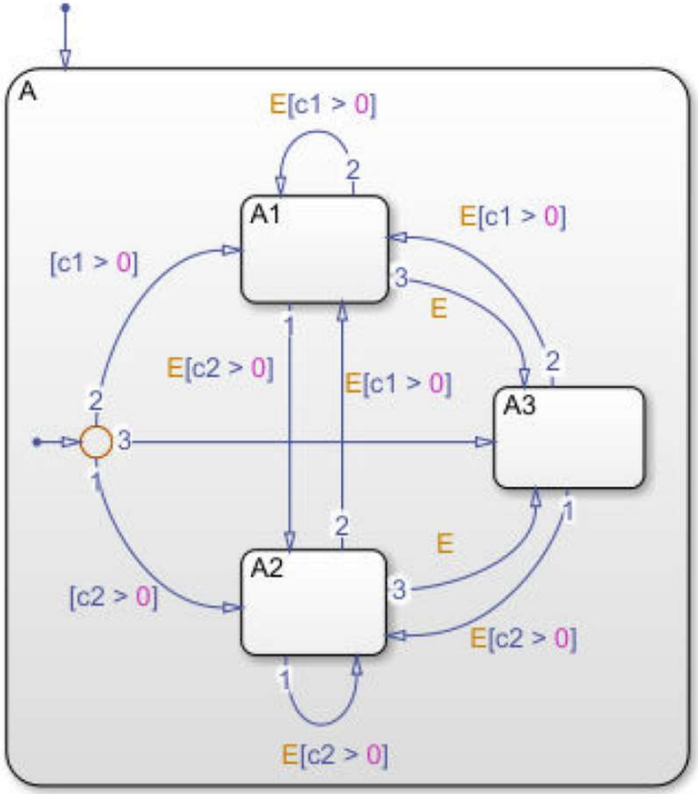
- When “Shift” equals 1, the system transitions from “First” to “Second”, “Second” to “Third”, “Third” to “First”.



Inner Transitions

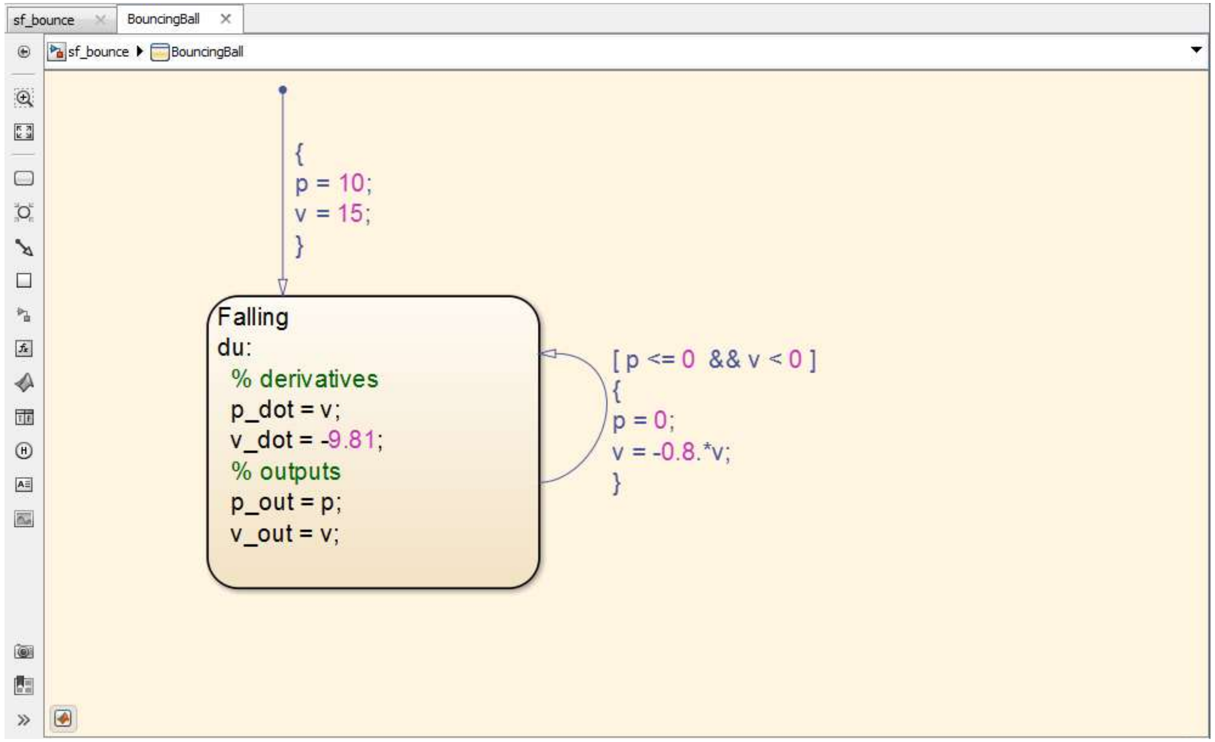
- An inner transition is a transition that does not exit the source state.

- Example
  - If “[c1>0]” is true, the transition to A1 is true.
  - If “[c2>0]” is true, the transition to A2 is valid.
  - If neither “[c1>0]” nor “[c2>0]” is true, the transition to A3 is valid.
  - The transition among A1, A2, and A3 are determined by E, “[c1>0]” , and “[c2>0]”



Self Loop

- A transition that originates from and terminates on the same state
- Example
  - open_system('sf_bounce.slx')



```
{
p = 10;
v = 15;
}
```

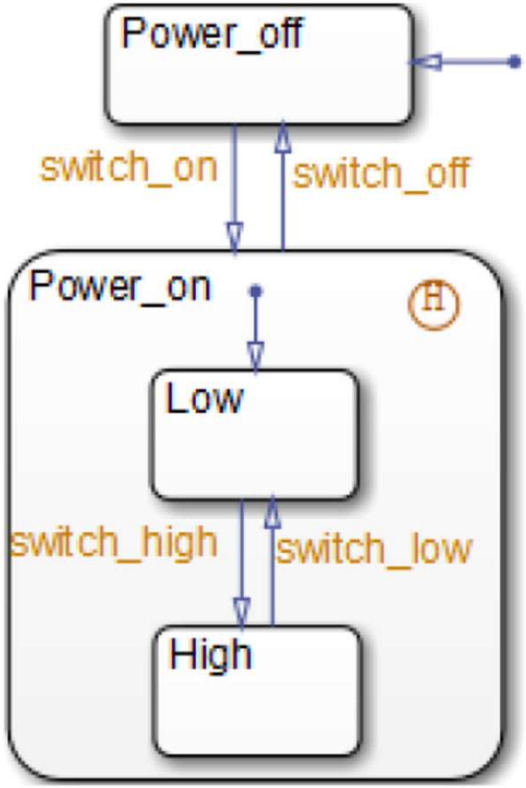
```
Falling
du:
  % derivatives
  p_dot = v;
  v_dot = -9.81;
  % outputs
  p_out = p;
  v_out = v;
```

```
[ p <= 0  && v < 0 ]
{
p = 0;
v = -0.8.*v;
}
```

History Junction

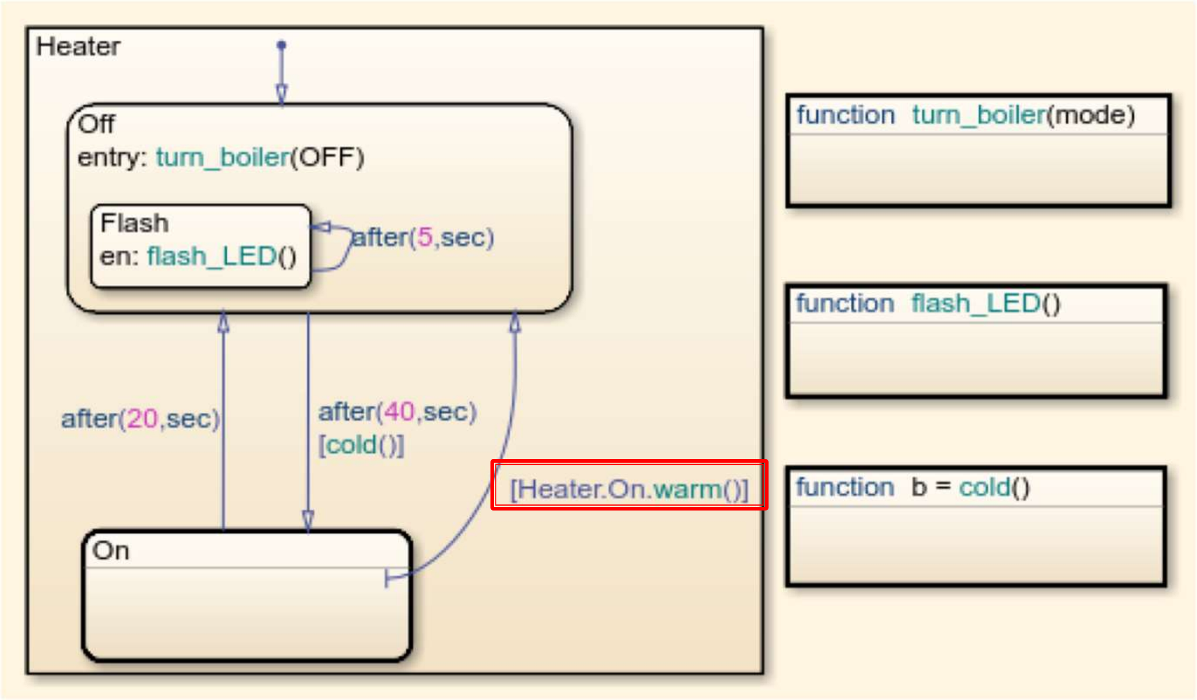
- History junction in a superstate indicates that historical state activity information is used to determine the next state to become active.

- Example
  - "Power_on" has history junction and contains two substates.
  - When "switch_on" occurs, transition to "Power_on" occurs.
  - First time superstate "Power_on" is entered, substate "Power_on.Low" is entered.
  - <u>Next time, "switch_high" occurs and "Power_on.High" becomes active.</u>
  - At some point "switch_off" occurs, "Power_off" becomes active.
  - When "switch_on" reoccurs. "Power_on.High" becomes active, because it was the last active substate.



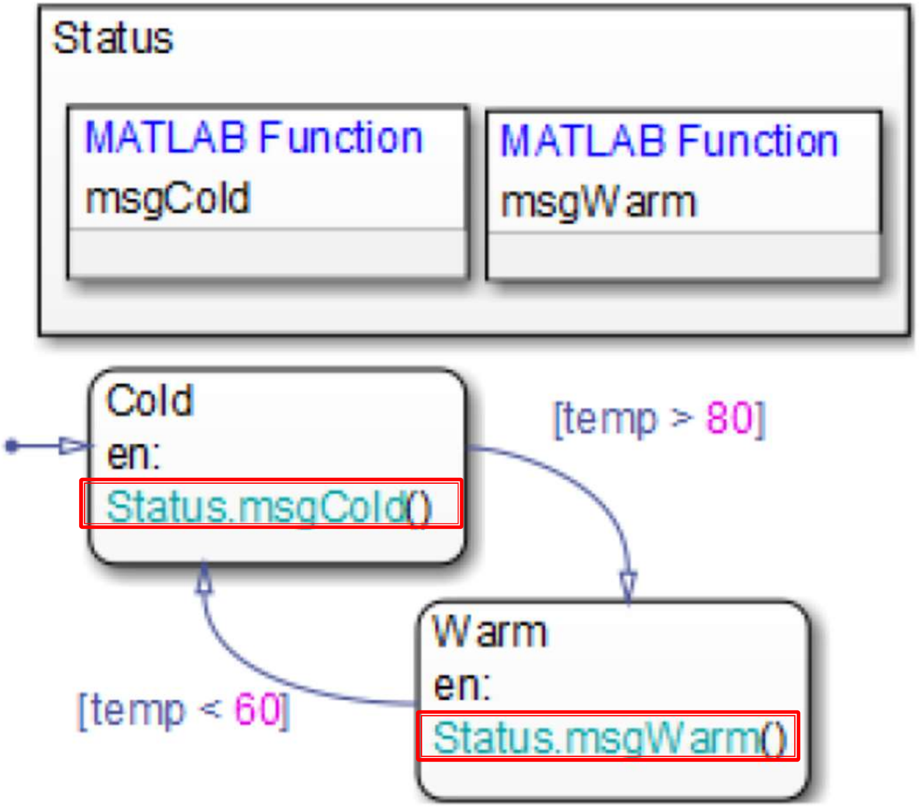
Boxes

- A box is a graphical object that organizes other objects in your chart, such as function and states.
- Example of grouping “Off” and “On” states
  - “sf_boiler”

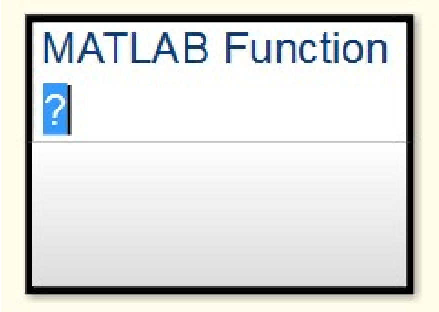


Boxes

- Example of grouping “msgCold” and “msgWarm” MATLAB functions.
  - The state “Cold” invokes the function “Status.msgCold()”



Reusable Functions



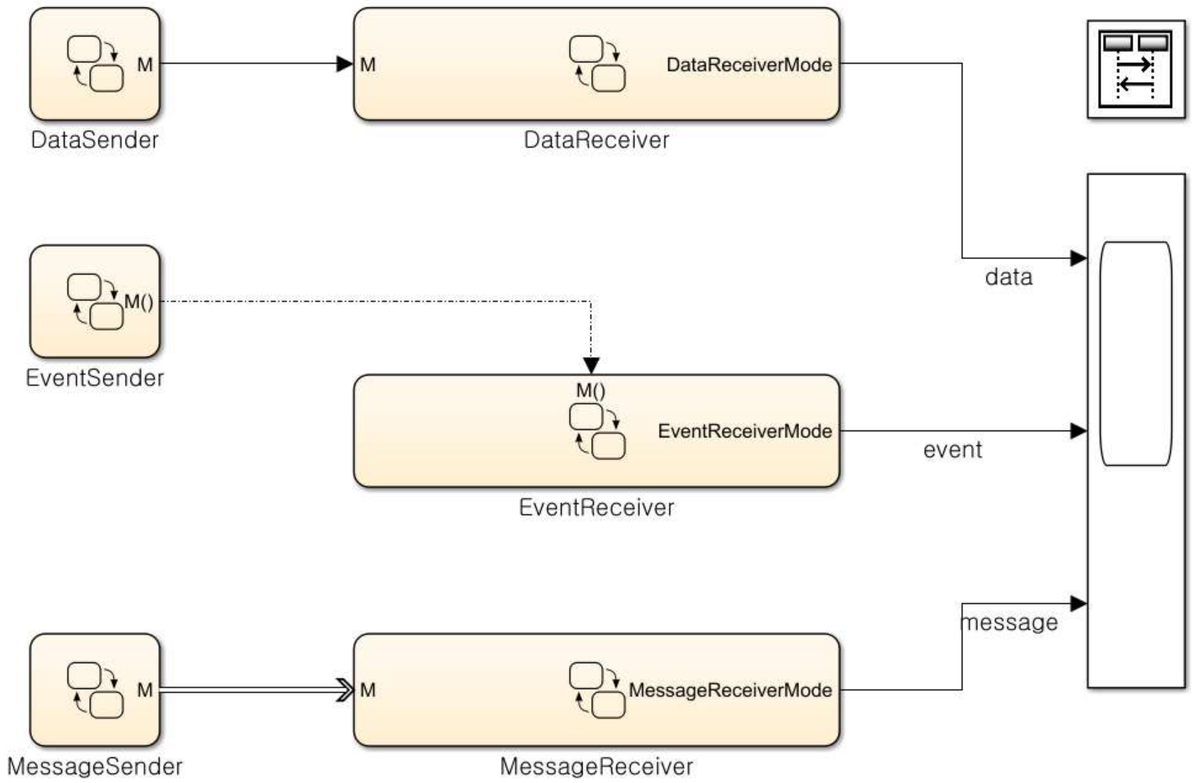
- State actions and transition conditions can be complicated. In this case, express the conditions or actions using reusable functions

- Syntax
  - Follow the conventional Matlab function syntax.
  - [*return_val1*, *return_val2*,...] = *function_name*(*arg1*, *arg2*,...)

Data, Message or Event

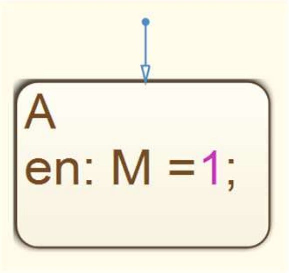
- open ('sf_msg_basic_semantics.slx')

- This simulation runs at 1 Hz using fixed step solver.

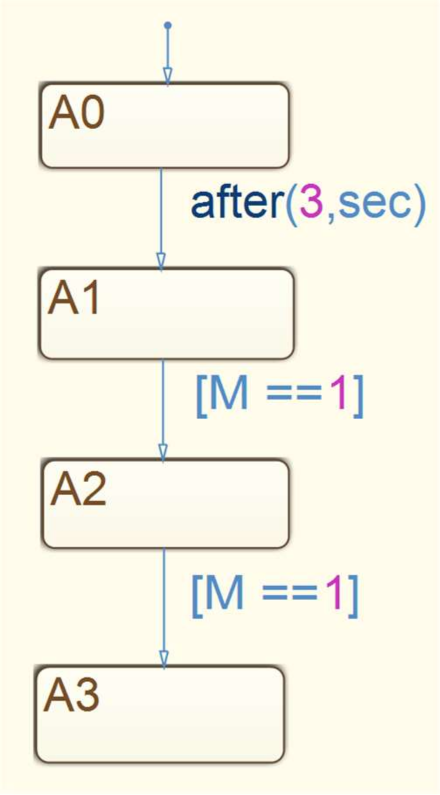


Data

- DataSender

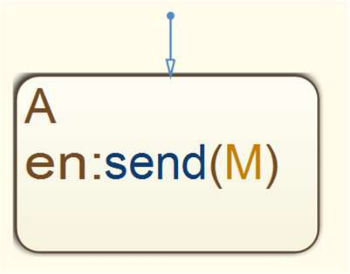


- DataReceiver

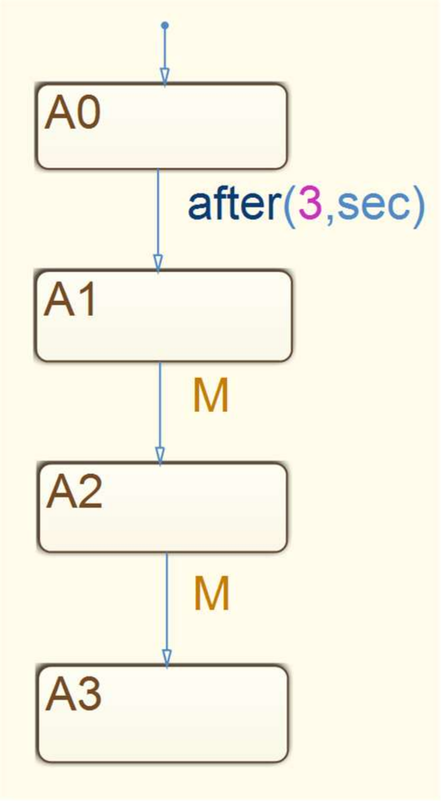


Event

- EventSender

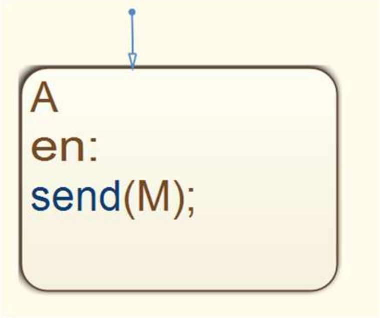


- EventReceiver

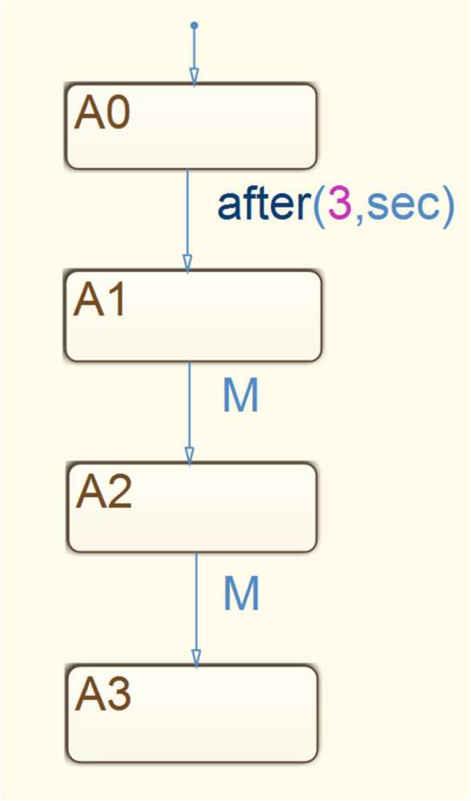


Message

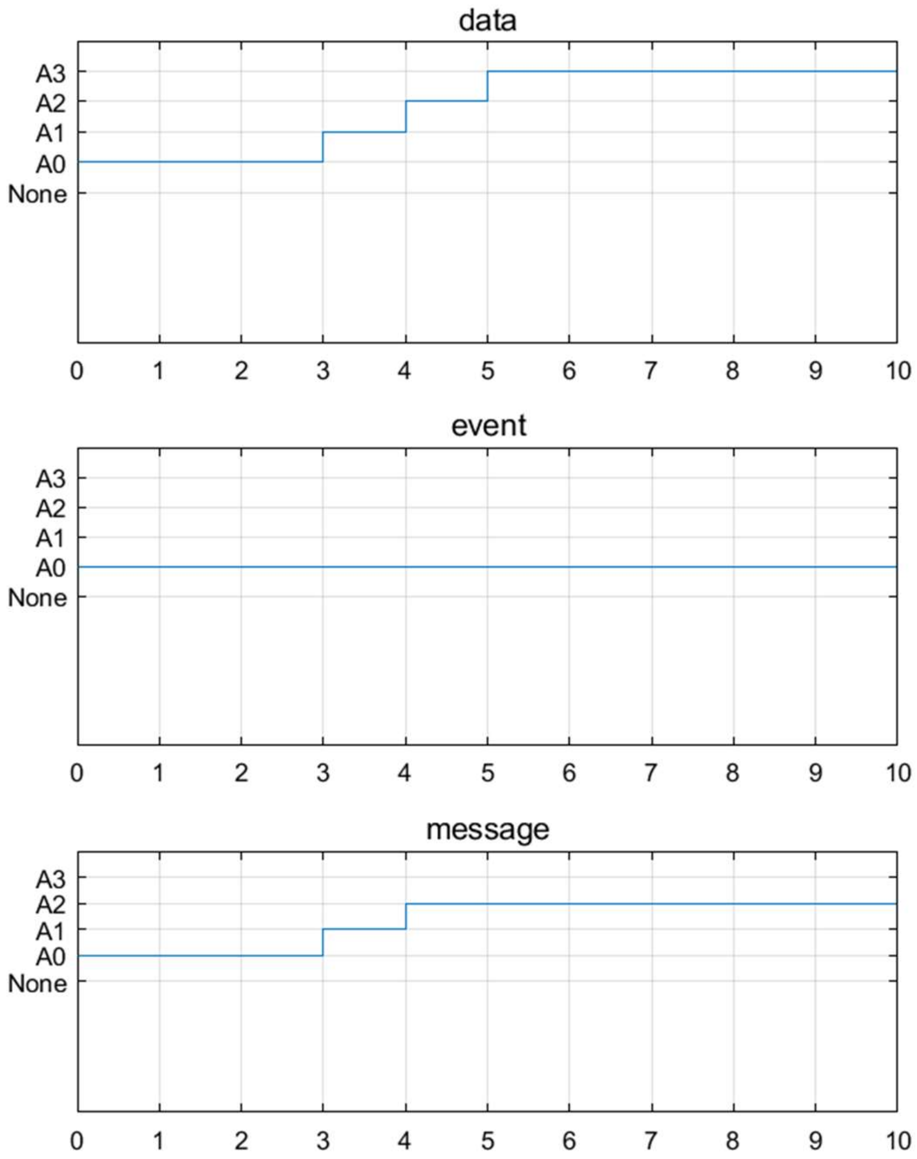
- MessageSender



- MessageReceiver

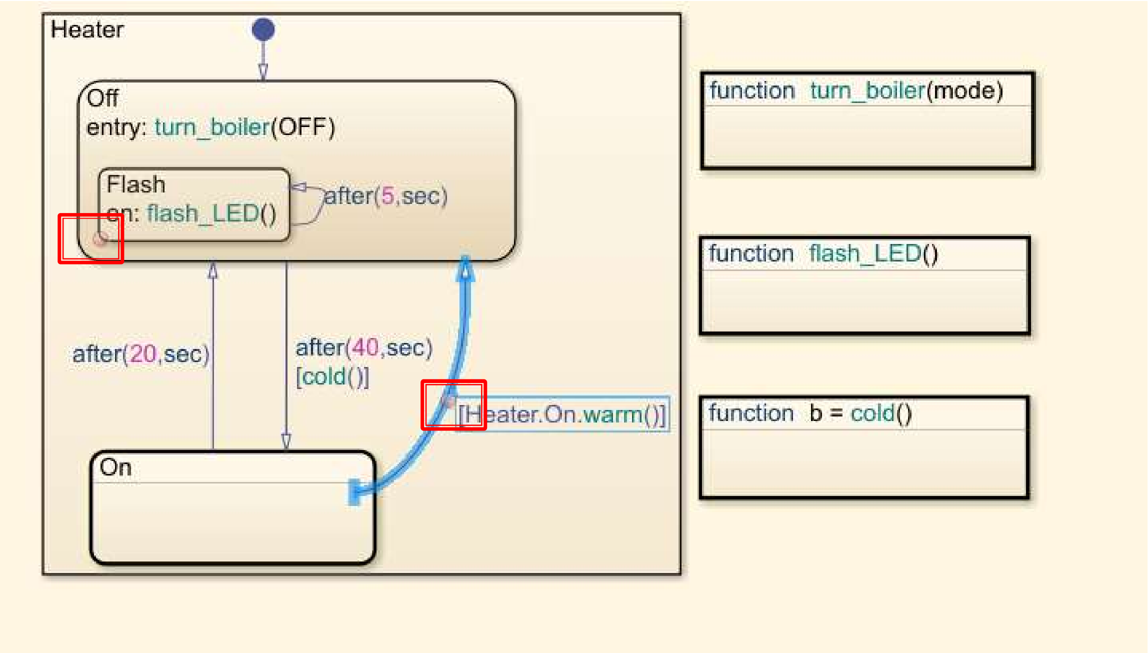


Result


data
A3
A2
A1
A0
None
0
1
2
3
4
5
6
7
8
9
10
event
A3
A2
A1
A0
None
0
1
2
3
4
5
6
7
8
9
10
message
A3
A2
A1
A0
None
0
1
2
3
4
5
6
7
8
9
10

Debugging Stateflow

- Set or Remove Breakpoints on States and Transitions
    - To set a breakpoint on a state or transition, right-click inside the chart and select Set Breakpoint.
    - To remove the breakpoint, right-click at the Breakpoint and select Clear Breakpoint.

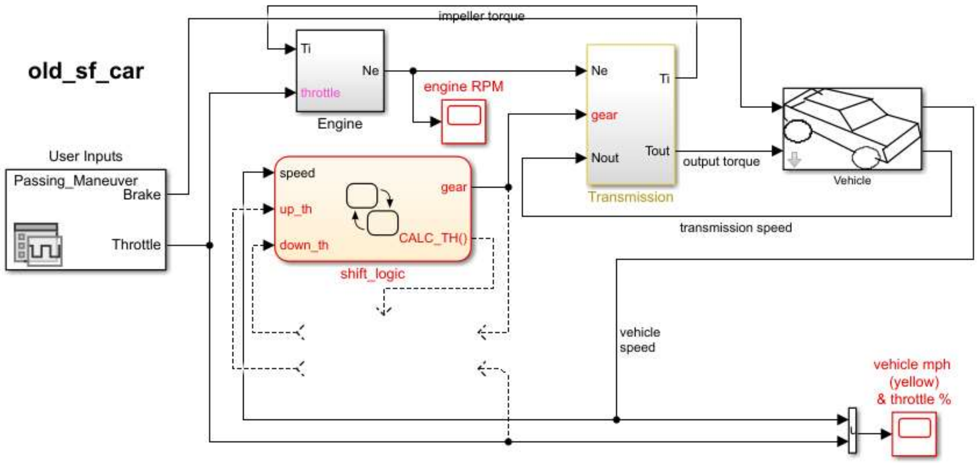


Old_sf_car Example

- Open the model ‘old_sf_car’ in the Simulink.
- This model contains the function-call subsystem named “Threshold Calculation“.
- When you run this model, the chart “shift_logic” broadcasts the output event “CALC_TH” to trigger the function-call subsystem.
- The “Threshold Calculation” function interpolates two values “down_th“ and “up_th“

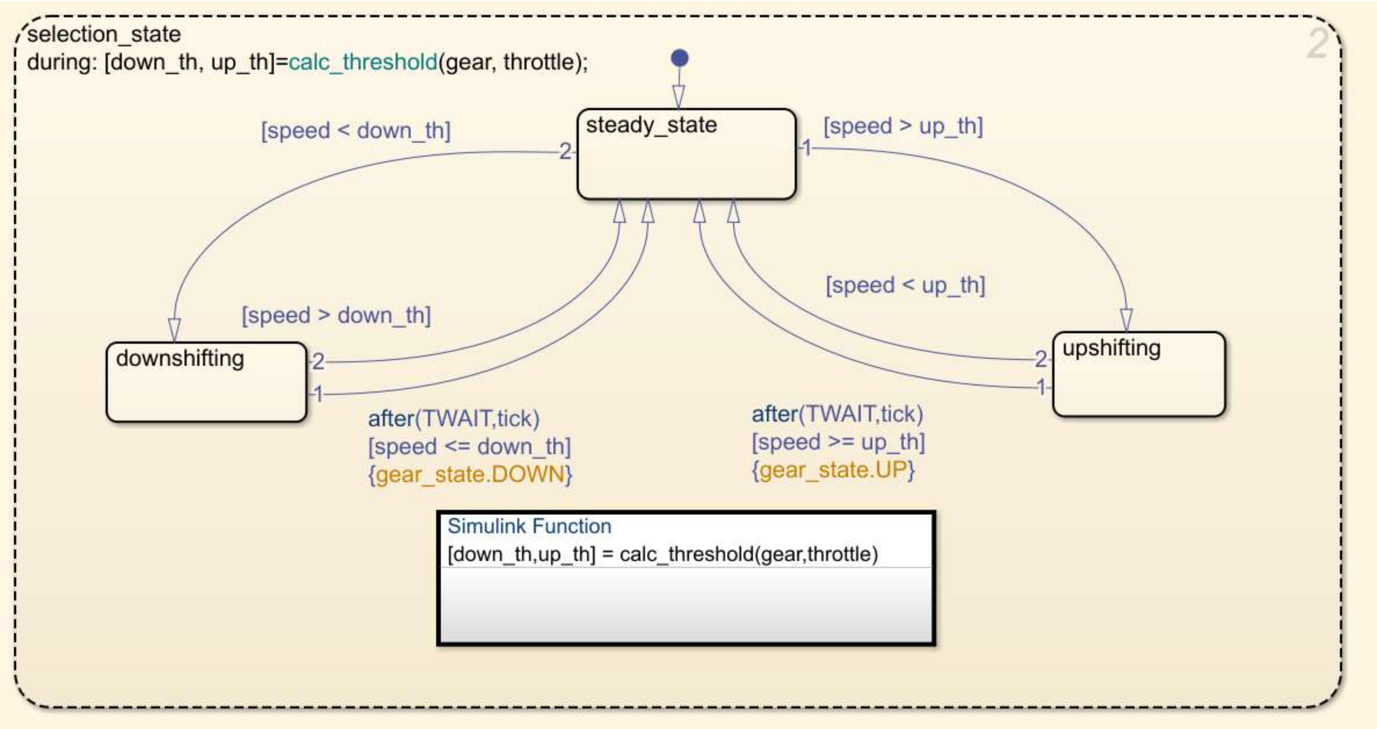
Old_sf_car Example

- Add a Simulink function to the Chart.
  - In the Simulink model, right click the threshold calculation block and select cut.
  - Open the shift logic chart.
  - In the chart, right-click below selection state and select “Paste”.
  - Expand the new Simulink function so that the signature fits inside the function box.



Old_sf_car Example

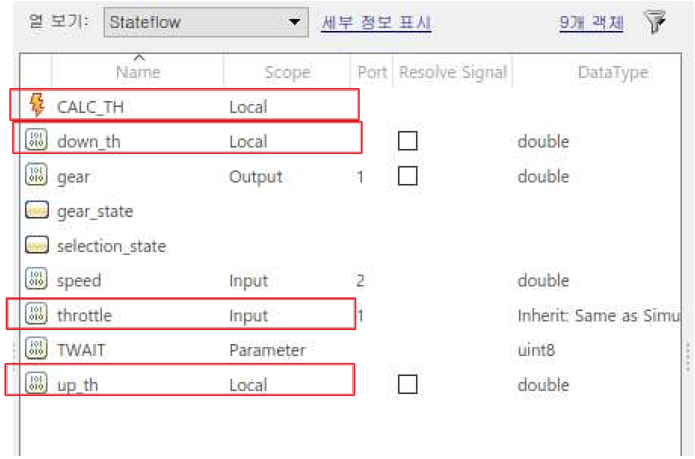
- Expand the border of “selection state” to include the new function
- Rename the Simulink function from “Threshold Calculation” to “calc_threshold”.



selection_state
during: [down_th, up_th]=calc_threshold(gear, throttle);

Old_sf_car Example

- Change the Scope of Chart Data
  - In the Model Explorer, change the scope of chart-level data “up_th“ and “down_th“ to local because calculation for those data now occur inside the chart.



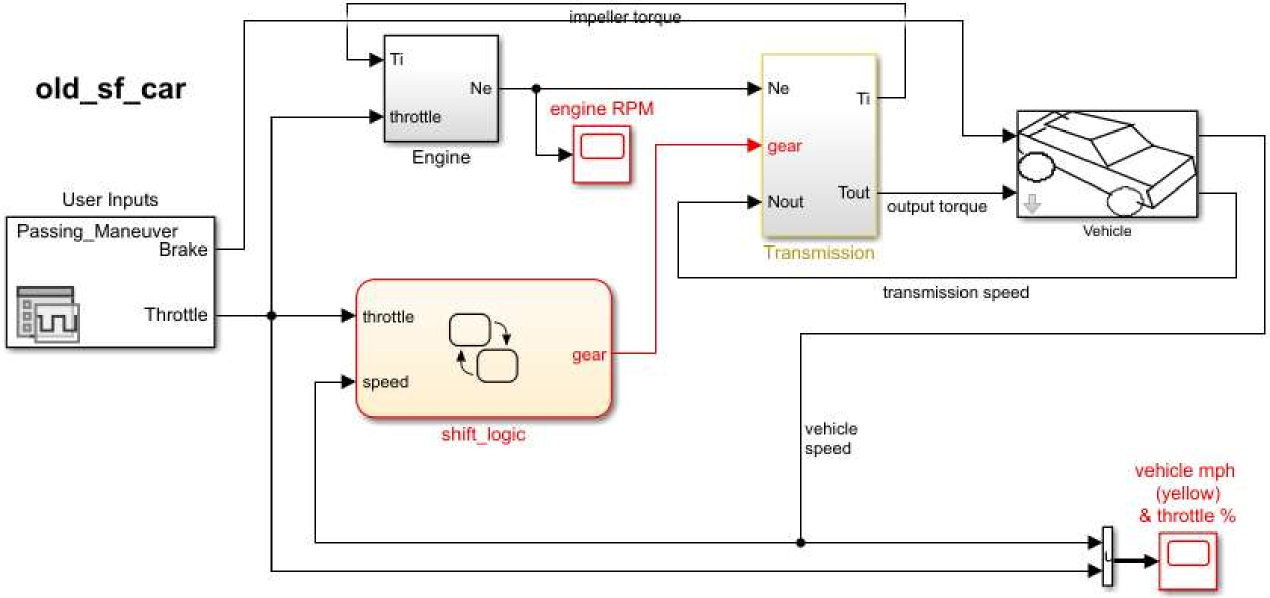
열 보기: Stateflow ▾ 세부 정보 표시 9개 객체

Name	Scope	Port	Resolve Signal	DataType
CALC_TH	Local			
down_th	Local		□	double
gear	Output	1	□	double
gear_state				
selection_state				
speed	Input	2		double
throttle	Input	1		Inherit: Same as Simu
TWAIT	Parameter			uint8
up_th	Local		□	double

- Update State Action in the Chart
  - In the state flow editor, change the during action to call the Simulink function “calc_threshold” such as [down_th, up_th]=calc_threshold(gear, throttle);.

Old_sf_car Example

- Add Data to the Chart
  - Because the function “calc_threshold” takes “throttle” as an input, you must define that data as a chart input.
- Remove Unused Items in the Model.
- Run the new model and check if the result match the original.



old_sf_car
impeller torque
Ti
Ne
throttle
Engine
engine RPM
Ne
Ti
gear
Nout
Tout
Transmission
output torque
Vehicle
transmission speed
User Inputs
Passing_Maneuver
Brake
Throttle
throttle
speed
gear
shift_logic
vehicle
speed
vehicle mph
(yellow)
& throttle %