Cyber-Physical Systems

Discrete Dynamics

ICEN 553/453– Fall 2018

Prof. Dola Saha

Discrete Systems

- **Discrete** = "individually separate / distinct"
- A **discrete system** is one that operates in a sequence of discrete *steps* or has signals taking discrete *values*.
- It is said to have **discrete dynamics**.

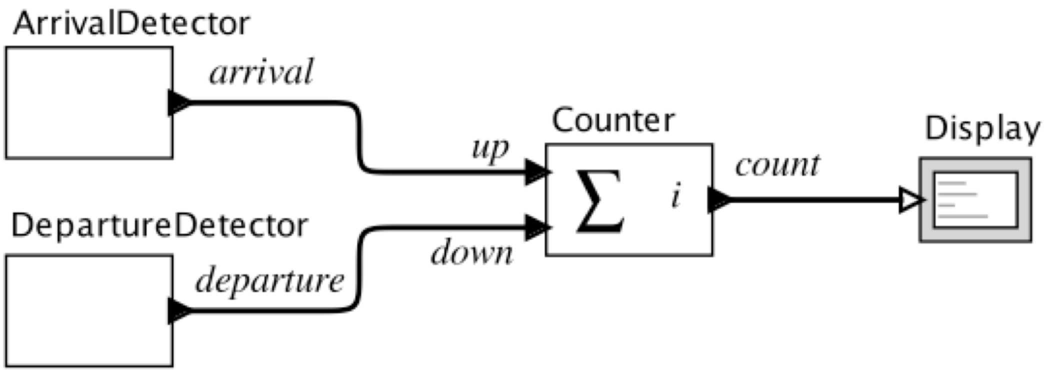
- A discrete event occurs at an instant of time rather than over time.

Discrete Systems: Example Design Problem

- Count the number of cars that are present in a parking garage by sensing cars enter and leave the garage. Show this count on a display.

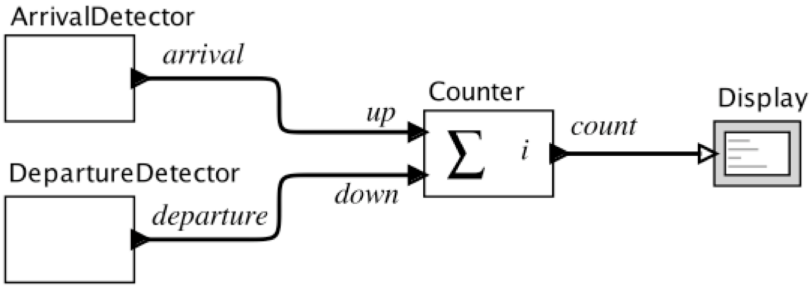
Discrete Systems

- Example: count the number of cars in a parking garage by sensing those that enter and leave:



Discrete Systems

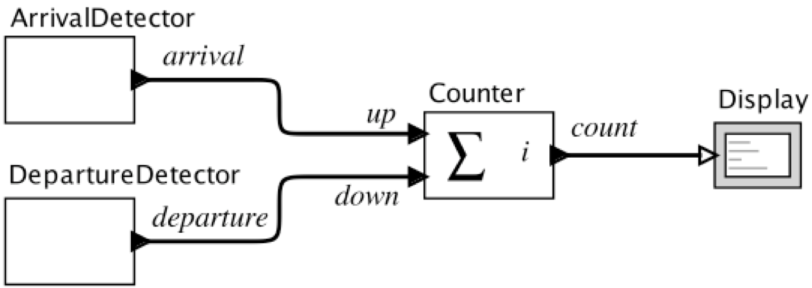
- Example: count the number of cars that enter and leave a parking garage:



- Pure signal: $up : \mathbb{R} \to \{absent, present\}$
- absent: no event at that time ; present: event at that time

Discrete Systems

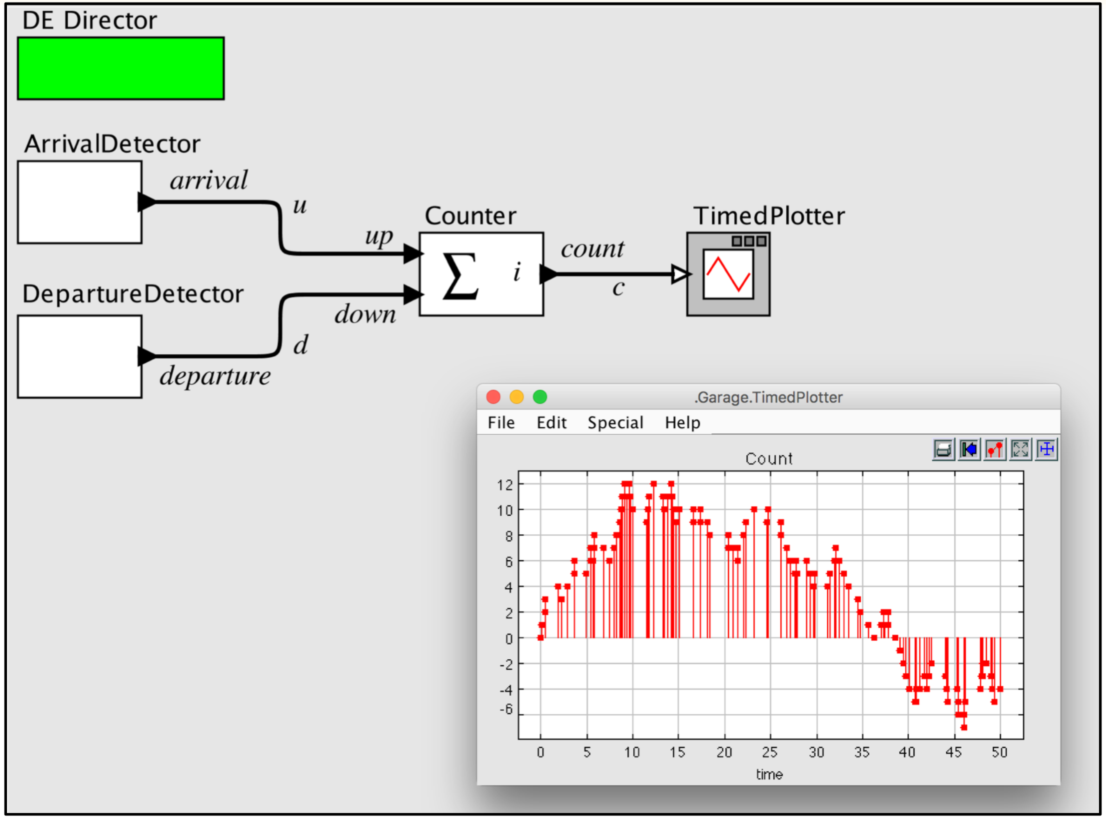
- Example: count the number of cars that enter and leave a parking garage:



- Pure signal: $up \colon \mathbb{R} \to \{absent, present\}$
- Discrete actor: $Counter \colon (\mathbb{R} \to \{absent, present\})^P \to (\mathbb{R} \to \{absent\} \cup \mathbb{N})$

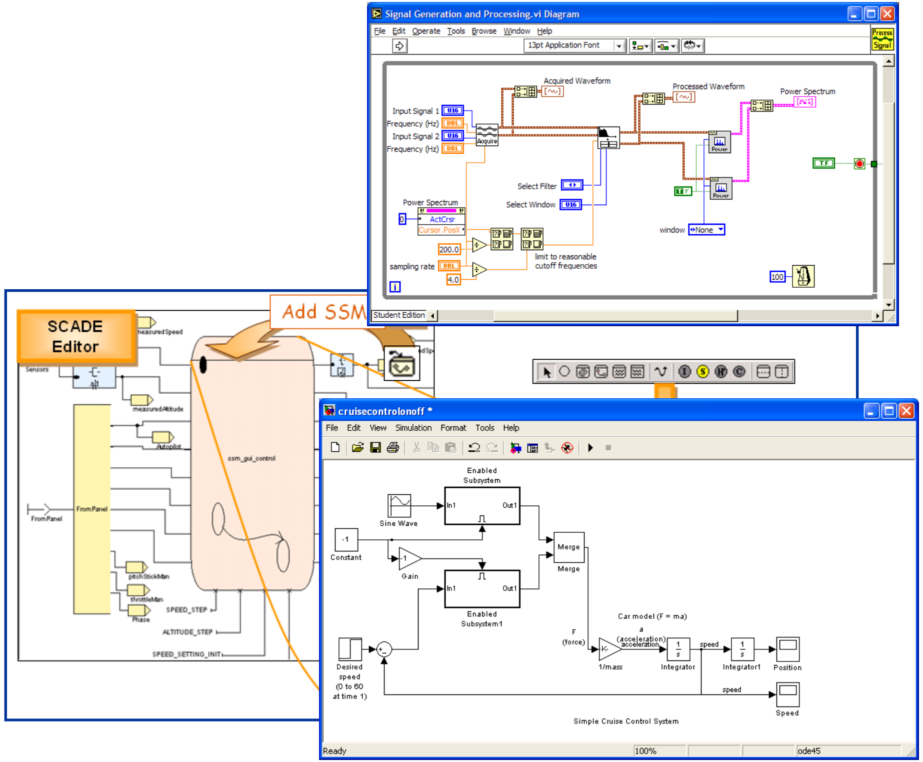
  $P = \{up, down\}$

Demonstration of Ptolemy II Model ("Program")



Actor Modeling Languages / Frameworks

- LabVIEW
- Simulink
- Scade
- …
- Reactors
- StreamIT
- …

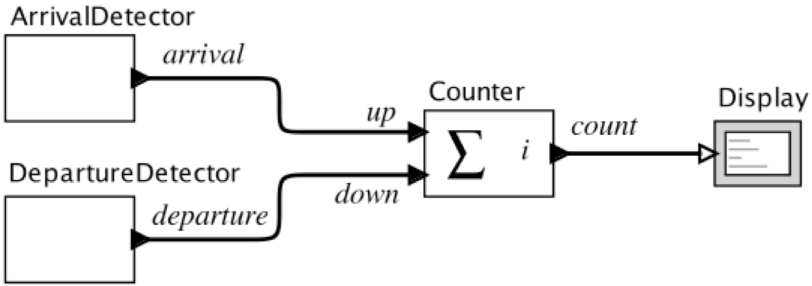


Reaction / Transition

For any $t \in \mathbb{R}$ where $up(t) \neq absent$ or $down(t) \neq absent$ the Counter **reacts**. It produces an output value in $\mathbb{N}$ and changes its internal **state**.

State: condition of the system at a particular point in time

- Encodes everything about the past that influences the system's reaction to current input



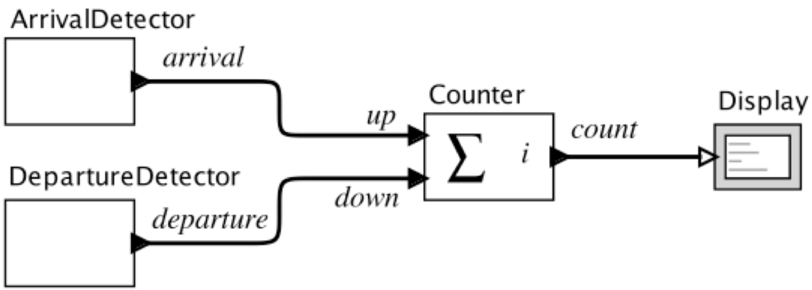
Inputs and Outputs at a Reaction

For $t \in \mathbb{R}$ the inputs are in a set

$$Inputs = (\{\mathit{up}, \mathit{down}\} \rightarrow \{\, absent, present\})$$

and the outputs are in a set

$$Outputs = (\{count\} \rightarrow \{\, absent\} \cup \mathbb{N})\ ,$$



Question

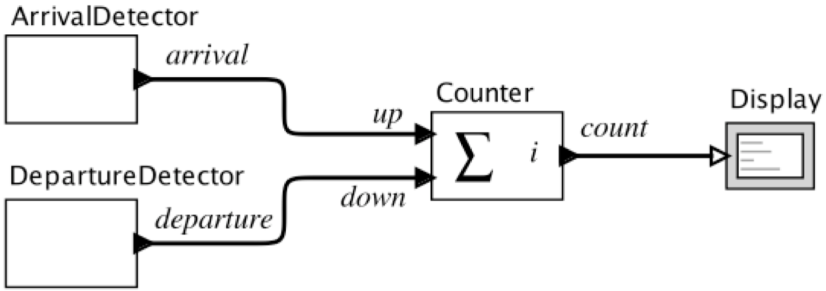
- What are some scenarios that the given parking garage (interface) design does not handle well?

For $t \in \mathbb{R}$ the inputs are in a set

$$Inputs = (\{\mathsf{up}, \mathsf{down}\} \to \{\, absent, present\})$$

and the outputs are in a set

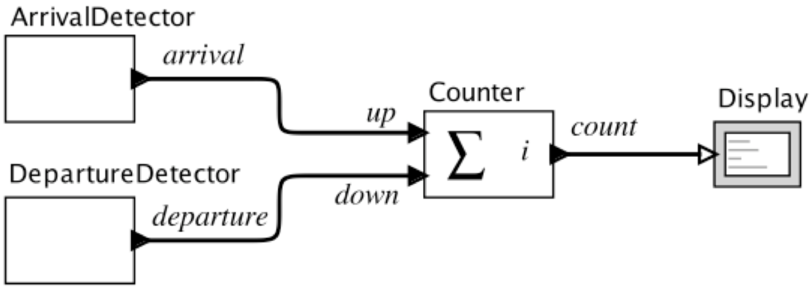
$$Outputs = (\{count\} \to \{\, absent\} \cup \mathbb{N}) \ ,$$



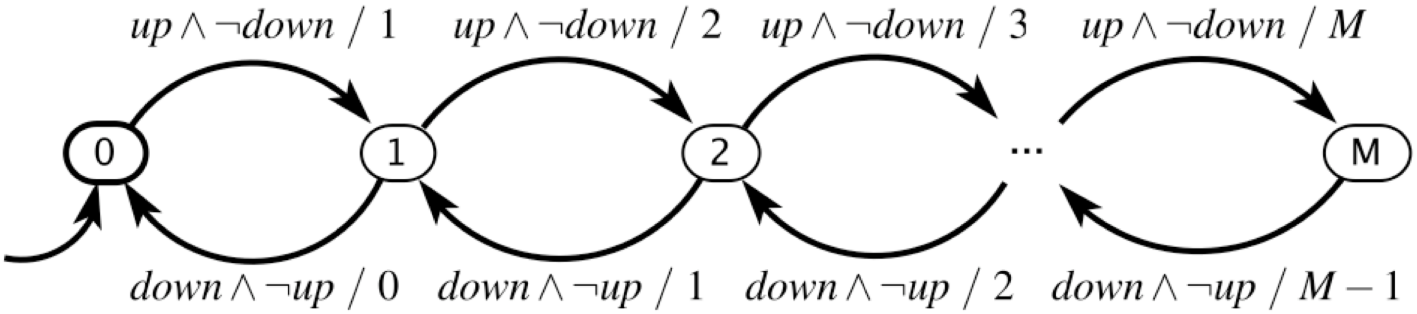
State Space

A practical parking garage has a finite number M of spaces, so the state space for the counter is

$$States = \{0, 1, 2, \cdots, M\} \ .$$



Garage Counter Finite State Machine (FSM)



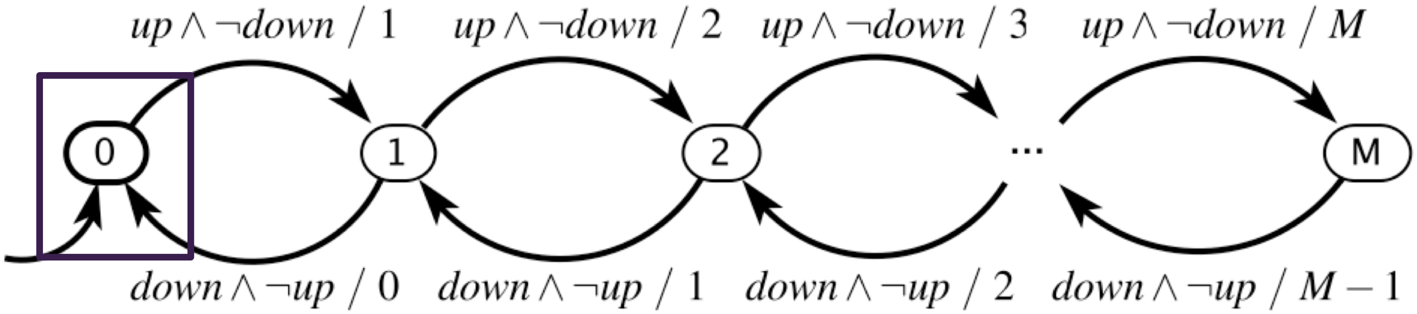
Guard $g \subseteq Inputs$ is specified using the shorthand

$$up \land \lnot down$$

which means

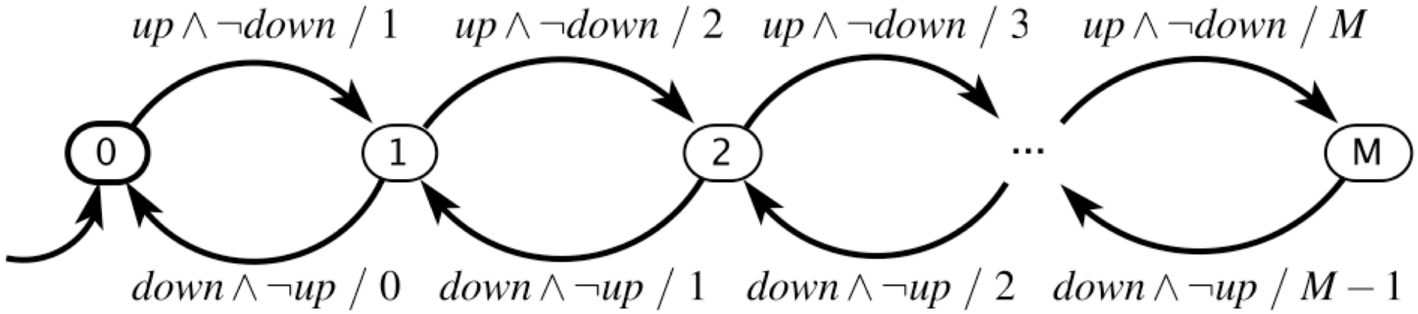
Inputs(up) = present,   Inputs(down) = absent

Garage Counter Finite State Machine (FSM)



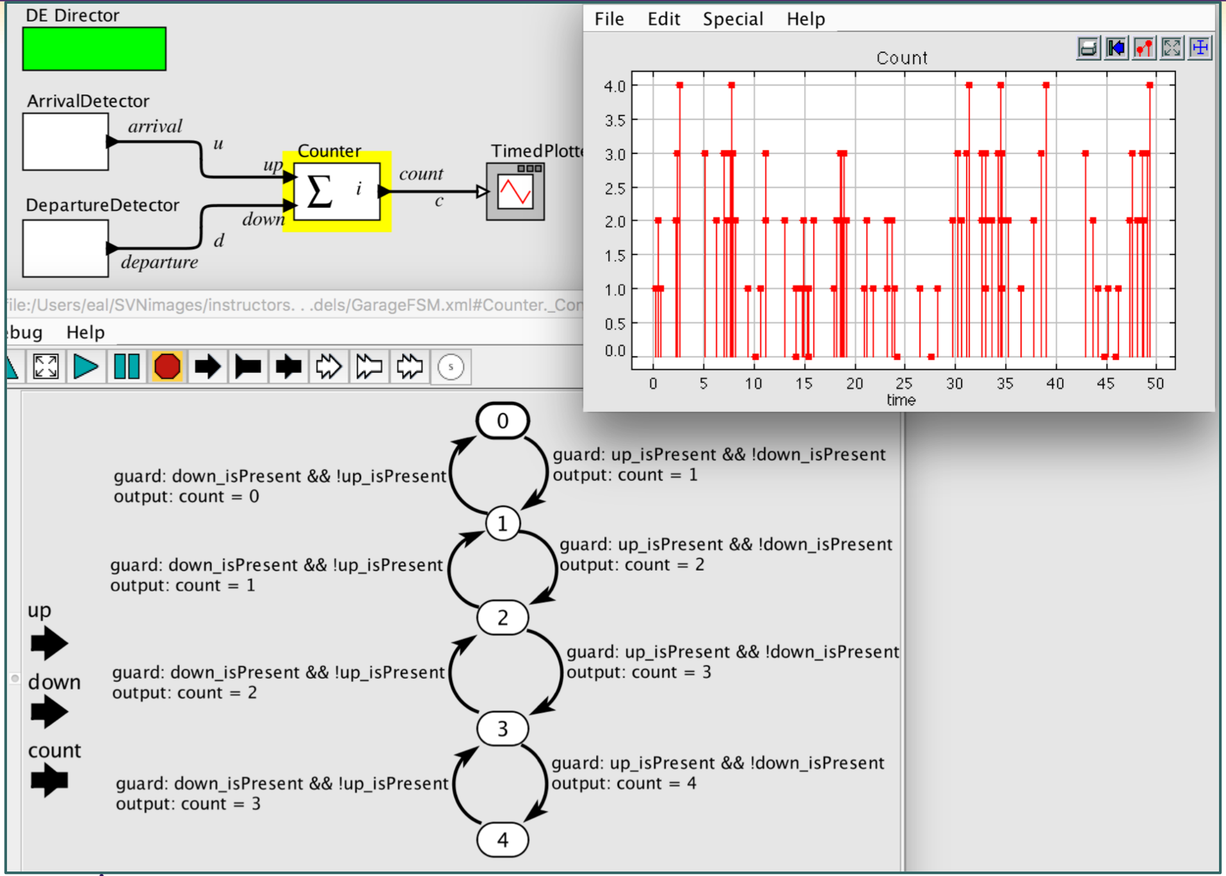
Initial state

Garage Counter Finite State Machine (FSM)



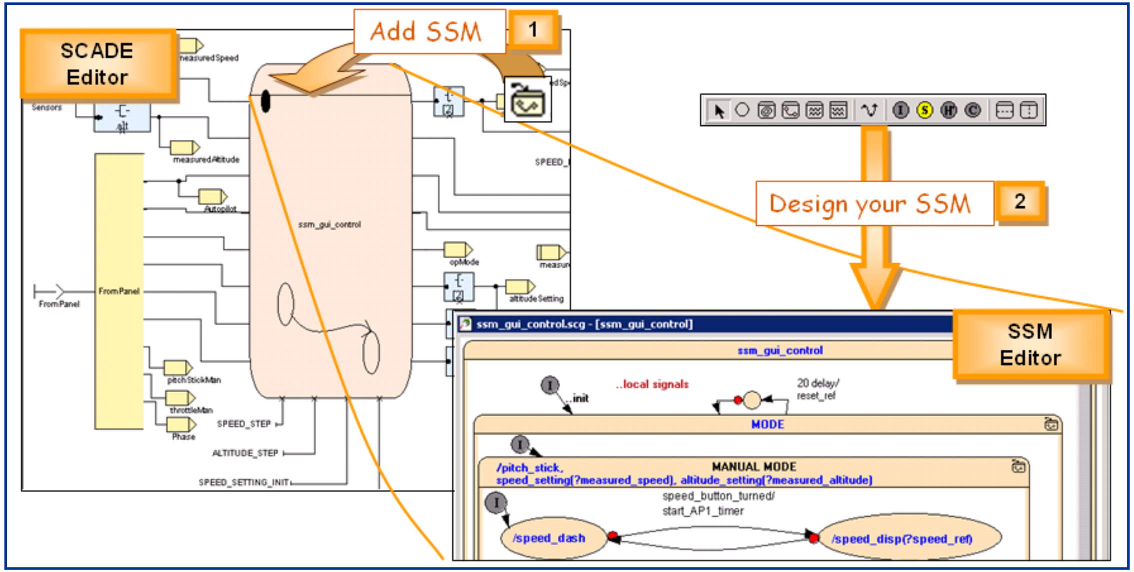
Output

Ptolemy II Model

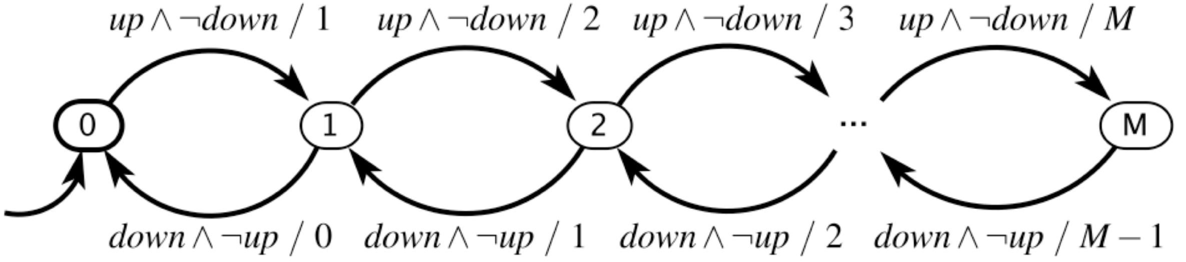


FSM Modeling Languages / Frameworks

- LabVIEW Statecharts
- Simulink Stateflow
- Scade
- …



Garage Counter Mathematical Model



Formally: $(States, Inputs, Outputs, update, initialState)$, where

- $States = \{0, 1, \cdots, M\}$
- $Inputs = (\{\text{up}, \text{down}\} \rightarrow \{absent, present\}$
- $Outputs = (\{count\} \rightarrow \{absent\} \cup \mathbb{N})$
- $update : States \times Inputs \rightarrow States \times Outputs$
- $initialState = 0$

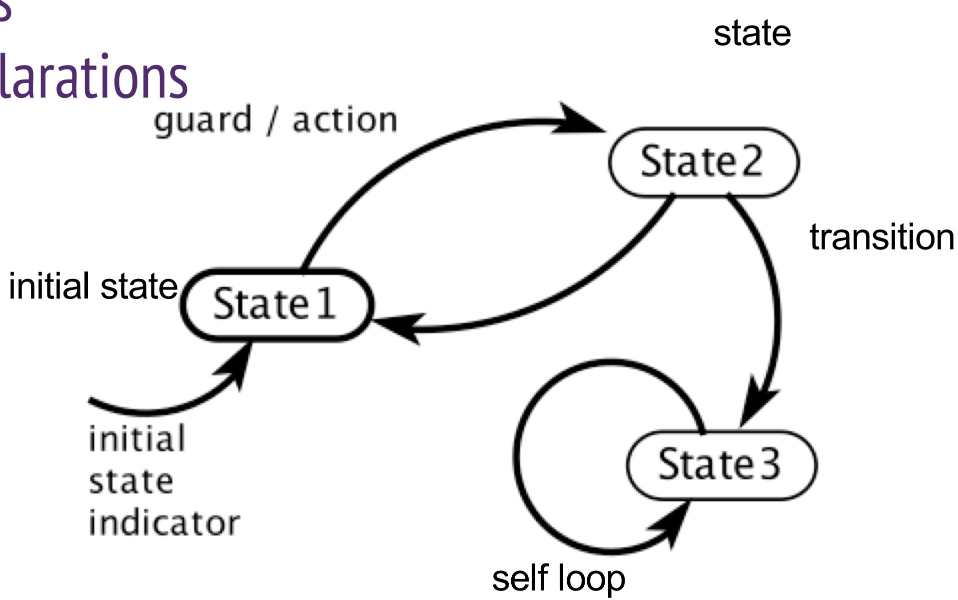
The picture above defines the update function.

FSM Notation

Input declarations

Output declarations

Extended state declarations



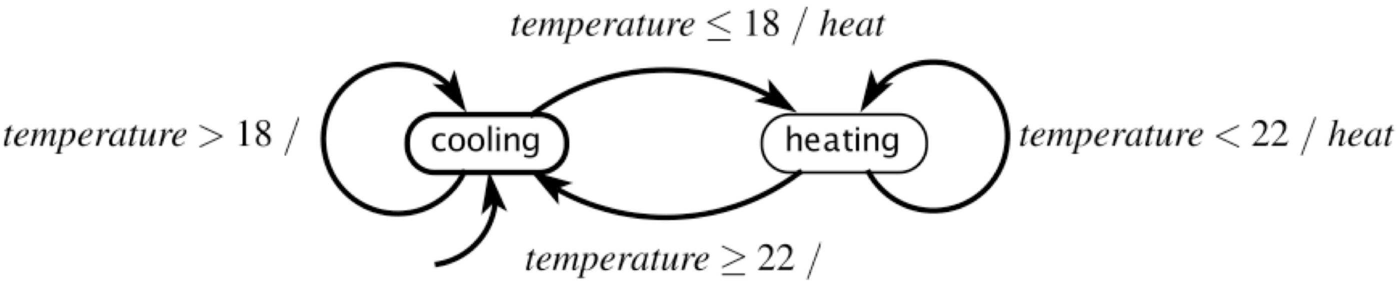
Examples of Guards for Pure Signals

true	Transition is always enabled.
p_1	Transition is enabled if p_1 is $present$.
$\neg p_1$	Transition is enabled if p_1 is $absent$.
$p_1 \wedge p_2$	Transition is enabled if both p_1 and p_2 are $present$.
$p_1 \vee p_2$	Transition is enabled if either p_1 or p_2 is $present$.
$p_1 \wedge \neg p_2$	Transition is enabled if p_1 is $present$ and p_2 is $absent$.

Guards for Signals with Numerical Values

p_3	Transition is enabled if p_3 is *present* (not *absent*).
$p_3 = 1$	Transition is enabled if p_3 is *present* and has value 1.
$p_3 = 1 \wedge p_1$	Transition is enabled if p_3 has value 1 and p_1 is *present*.
$p_3 > 5$	Transition is enabled if p_3 is *present* with value greater than 5.

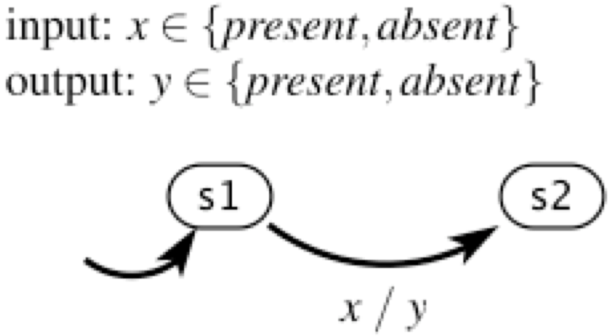
Example of *Modal* Model: Thermostat



When does a reaction occur?

- Suppose all inputs are discrete and a reaction occurs *when any input is present*. Then the below transition will be taken whenever the current state is s1 and *x* is present.

- This is an *event-triggered model*.

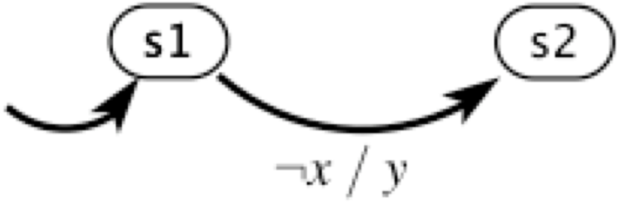


When does a reaction occur?

- Suppose *x* and *y* are discrete and pure signals. When does the transition occur?

input: $x \in \{present, absent\}$

output: $y \in \{present, absent\}$



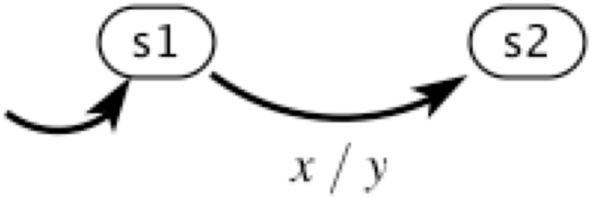
Answer: when the *environment* triggers a reaction and x is absent. If this is a (complete) event-triggered model, then the transition will never be taken because the reaction will only occur when x is present!

When does a reaction occur?

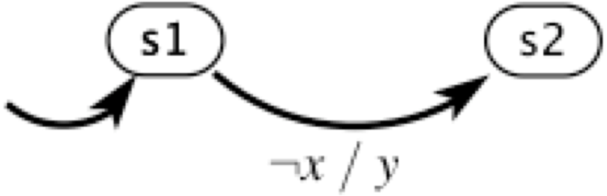
- Suppose all inputs are discrete and a reaction occurs *on the tick of an external clock*.

- This is a *time-triggered model*.

input: $x \in \{present, absent\}$
output: $y \in \{present, absent\}$

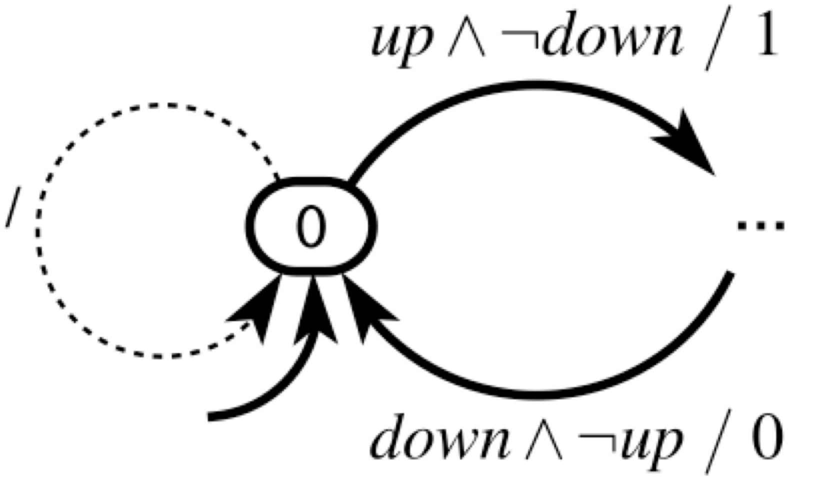


input: $x \in \{present, absent\}$
output: $y \in \{present, absent\}$



More Notation: Default Transitions

- A default transition is enabled if it either has no guard or the guard evaluates to true. When is the below default transition enabled?



Default Transitions

- Example: Traffic Light Controller