

U.PORTO

FEUP FACULDADE DE ENGENHARIA

UNIVERSIDADE DO PORTO

StateFlow
Hands-On Tutorial

HS/PDEEC 2010-03-04

José Pinto – zepinto@fe.up.pt

Session Outline

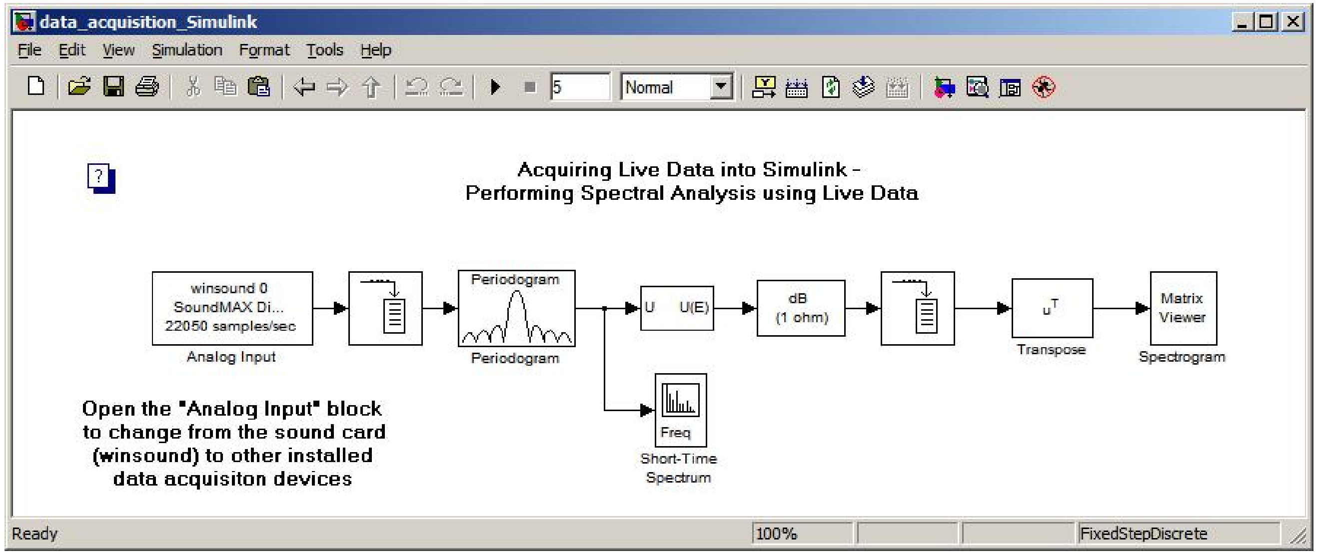
- Simulink and Stateflow
- Numerical Simulation of ODEs
  - Initial Value Problem (Hands-on)
  - ODEs with resets (Hands-on)
- Finite State Machines
  - FSMs in Stateflow (Hands-on)
- Discrete Event Systems
  - DESs in Stateflow (Hands-on)
- Hybrid Automata
  - Hybrid Systems in Stateflow (Hands-on)

1.

Simulink and Stateflow

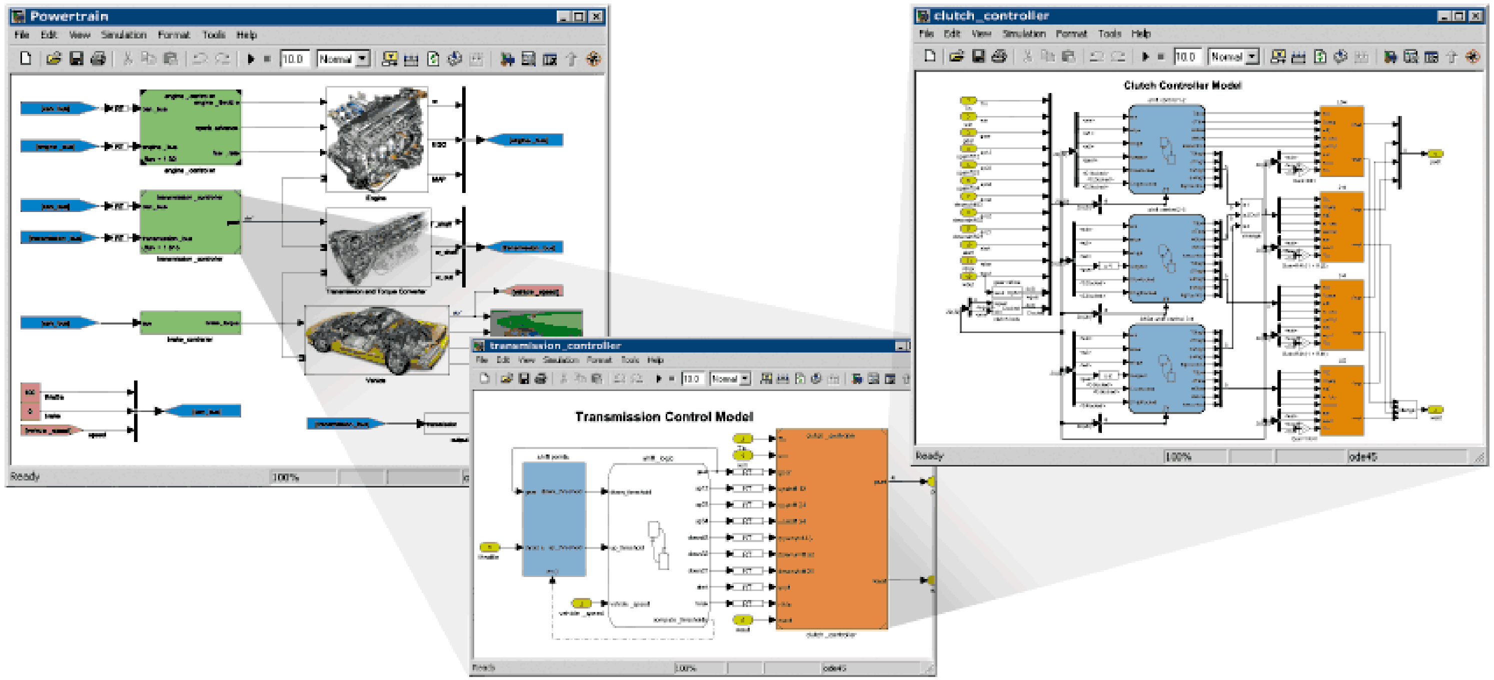
Simulink

- Can be used to model and simulate dynamical systems in a comprehensive and graphical way
- Models are described as block diagrams (boxes with inputs/outputs)



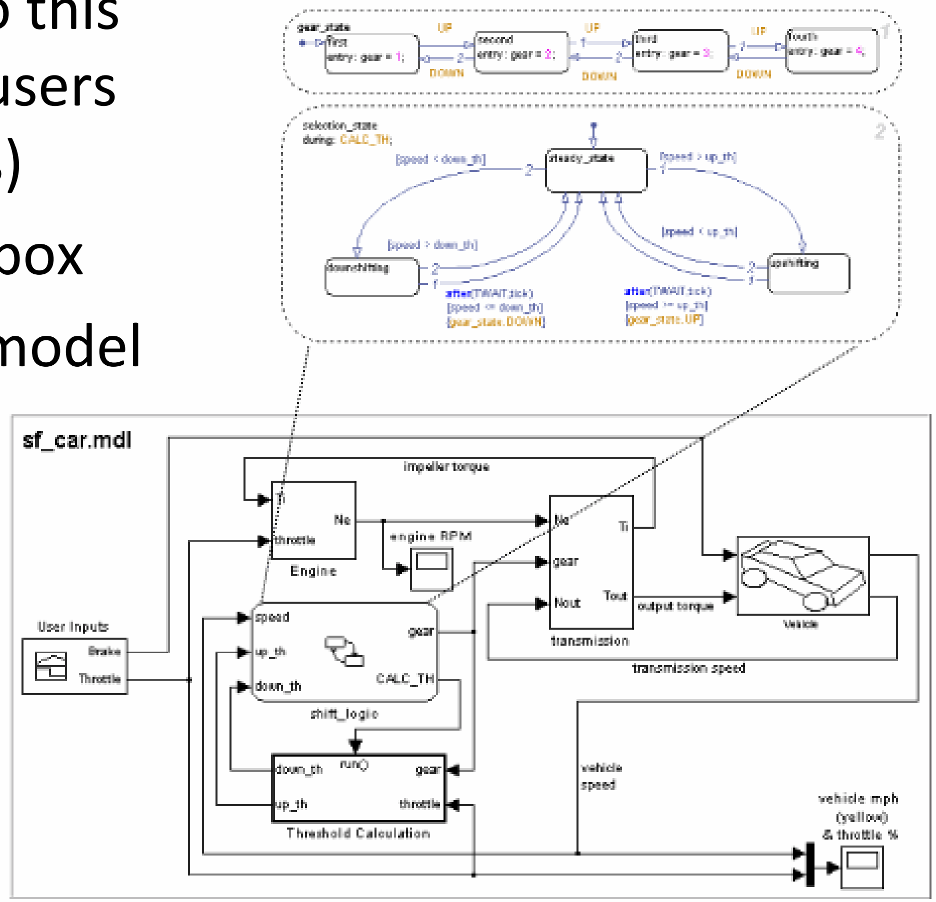
Simulink (2)

- Models are composed hierarchically allowing:
  - Modeling of complex systems in a modular and organized way
  - Different detail perspectives over the same model at design and simulation time



Simulink and Stateflow

- Simulink includes several built-in block types (Model Library)
- Additionally, extensions to this library can be created by users and companies (toolboxes)
- Stateflow is one such toolbox
- Stateflow can be used to model the behaviour of FSMs



2.

Numerical Simulation of ODEs

The Initial Value Problem

Initial value problem (IVP) $\equiv \quad \dot{x} = f(x) \qquad x(0) = x_0$

Definition: A signal $x : [0,T] \to \mathbb{R}^n$ is a ***solution*** to the IVP if

$$x(t) = x_0 + \int_0^t f(x(\tau))d\tau \qquad \forall t \in [0, T]$$

Example:

$$\dot{x} = -x + 1$$

$$x(0) = 2$$

$$x(t)?$$

[Hespanha, J. P. 05]

ODE Numerical Simulation - Euler method

Euler method (first order method):

1st partition interval into N subintervals of length $h := T/N$

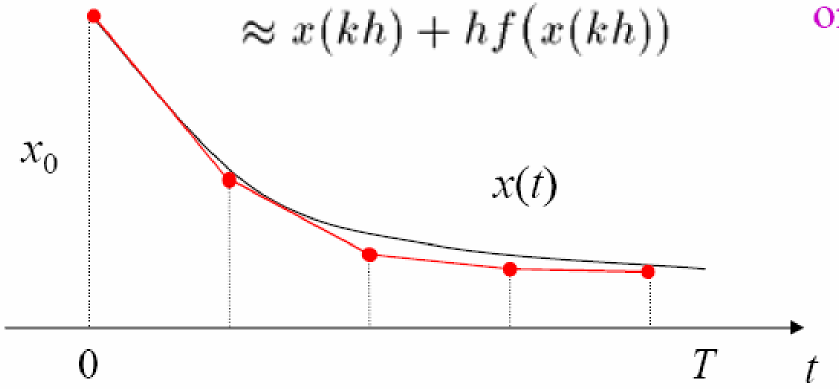
$$[kh, (k+1)h] \qquad k \in \{0, 1, \dots, N-1\}$$

2nd assume derivative of x constant on each subinterval

$$x\big((k+1)h\big) = x(kh) + \int_{kh}^{(k+1)h} f\big(x(\tau)\big)d\tau$$

$$\approx x(kh) + hf\big(x(kh)\big)$$

on each subinterval x is assumed linear



[Hespanha, J. P. 05]

ODE Simulation - Range-Kutta method

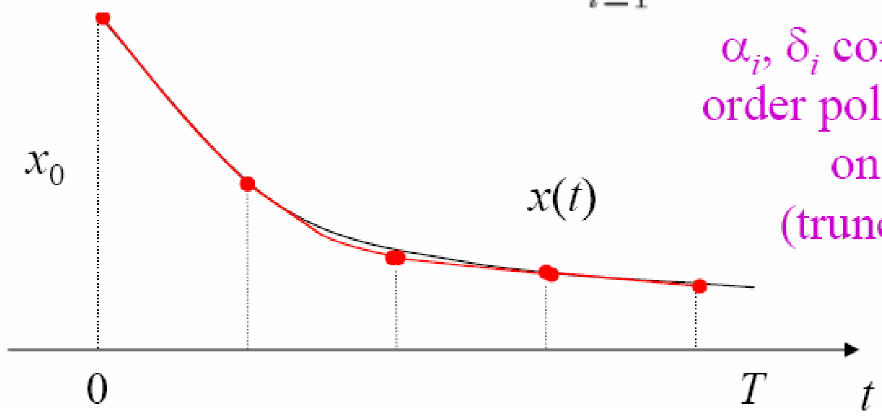
Runge-Kutta methods *(m*-order method):

1st partition interval into N subintervals of length $h := T/N$

$$[kh, (k+1)h] \qquad k \in \{0, 1, \ldots, N-1\}$$

2nd assume derivative of x constant on each subinterval

$$x\big((k+1)h\big) \approx x(kh) + h \sum_{i=1}^{m} \alpha_i f\big(x(kh) + \delta_i\big)$$



α_i, δ_i computed assuming a m-order polynomial approximation on each subinterval (truncated Taylor series)

[Hespanha, J. P. 05]

ODE Simulation – Variable-step methods

Variable-step methods (e.g., Euler):
Pick tolerance ε and define $t_0 := 0$

$$x(t_{k+1}) = x(t_k) + \int_{t_k}^{t_{k+1}} f(x(\tau))d\tau$$
$$\approx x(t_k) + (t_{k+1} - t_k) f(x(t_k))$$

choose t_{k+1} sufficiently close to t_k so that

$$\|f(x(t_k)) - f(x(t_{k+1}))\| \leq \epsilon$$

Simulation can be both *fast* and *accurate*:
1. when f is "flat" one can advance time fast,
2. when f is "steep" one advances time slowly (to retain accuracy)

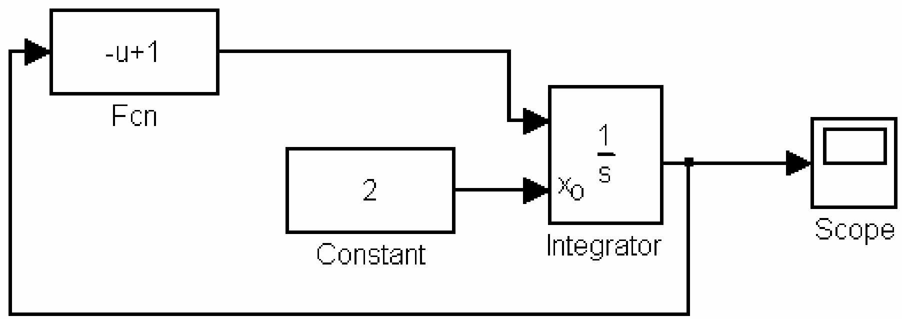
[Hespanha, J. P. 05]

Solving the IVP in Simulink

- Simulink has an inbuilt ODE solver (Integrator)
  - Different integration methods (including variable step)

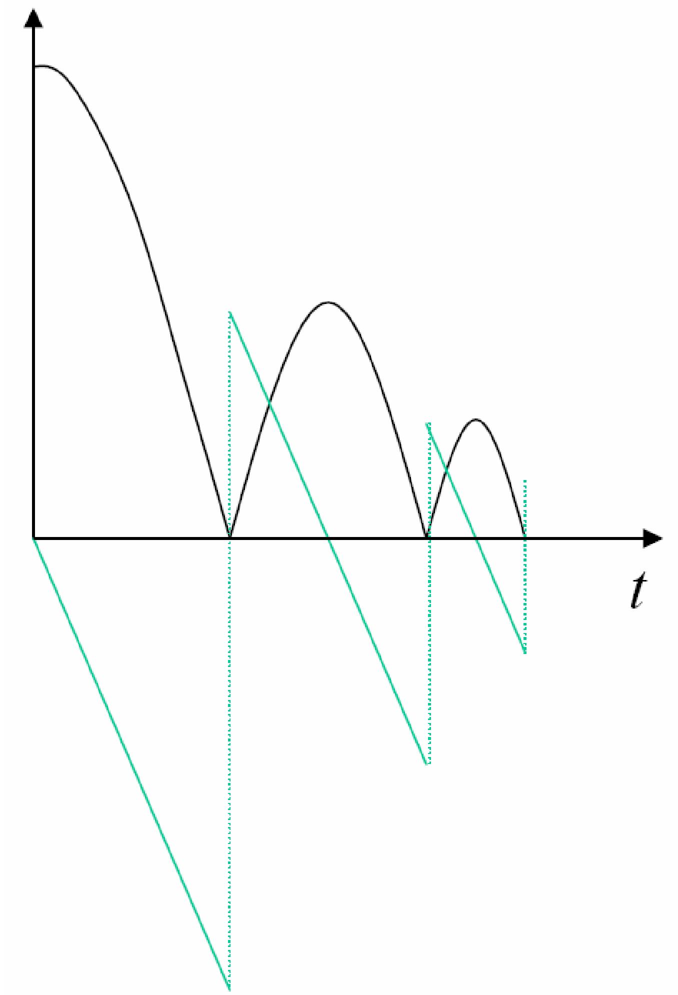
- Example:

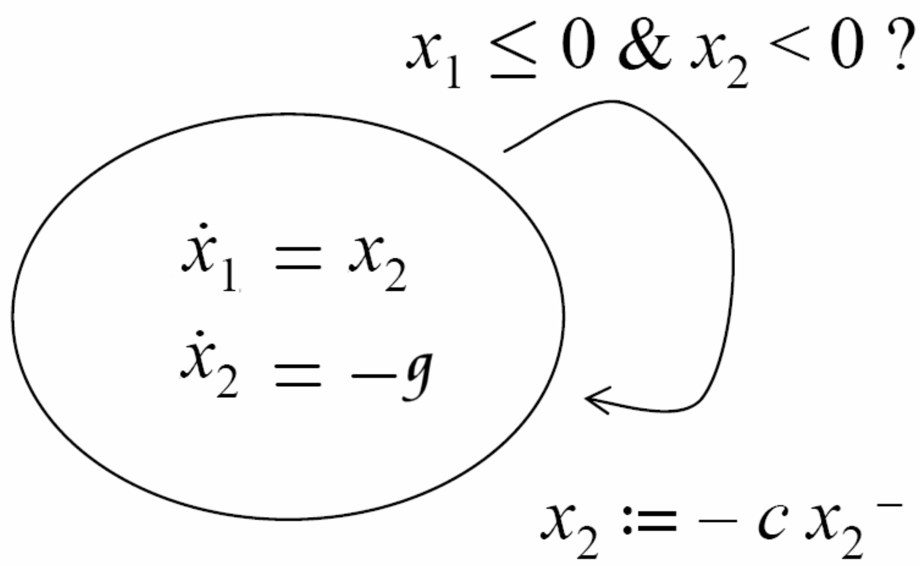
$$\dot{x} = -x + 1 \qquad x(0) = 2$$



(Integrator toolbox accepts initial value)

ODEs with resets

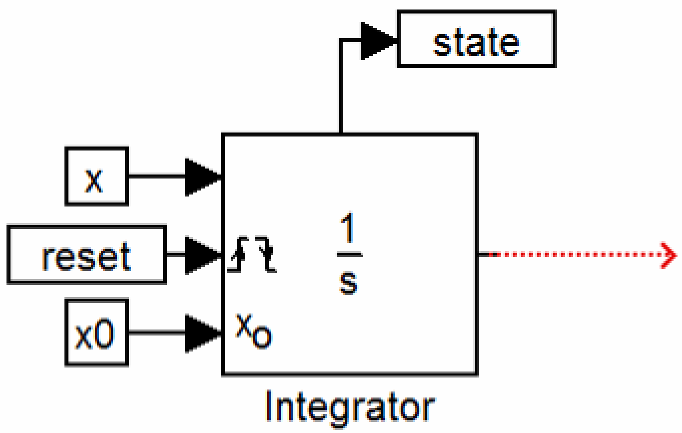




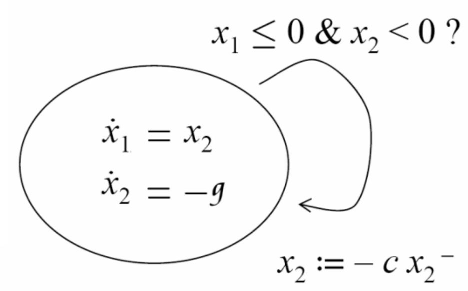
x_1: speed
x_2: acceleration
g: gravity force
c: ball elasticity constant

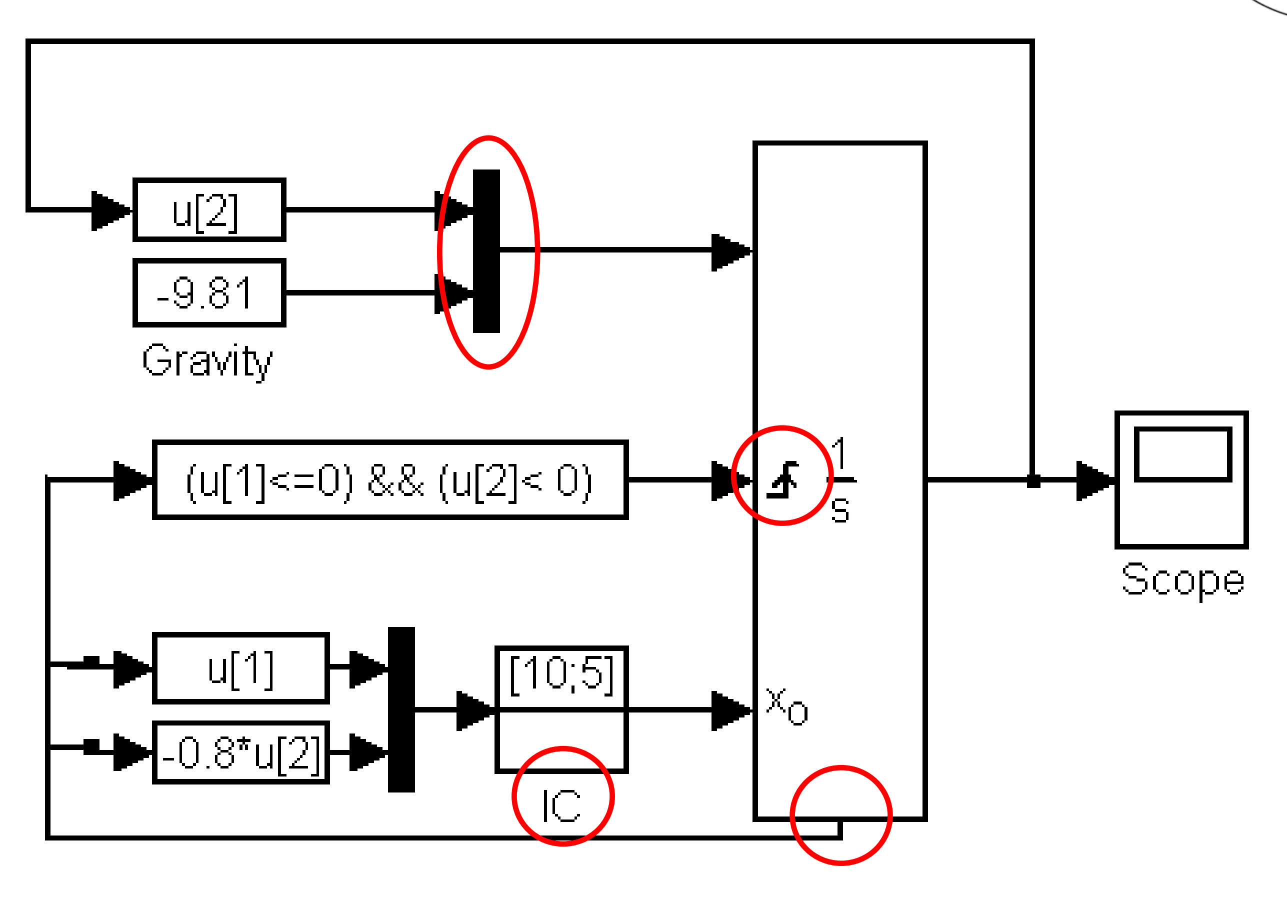
Integrator Block in Simulink

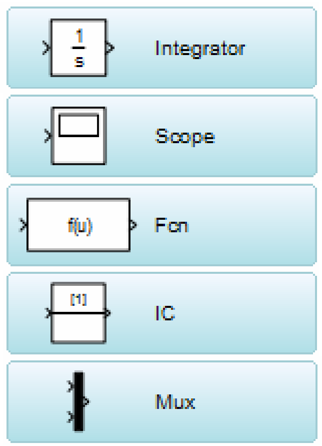
- Integrator block accepts a reset port
- Whenever a reset is triggered, its new value will be taken from the initial value (x_0) port
- State port holds previous value of x
- State can be used to determine if the integrator needs to be reset (x^-)



ODEs with resets in Simulink

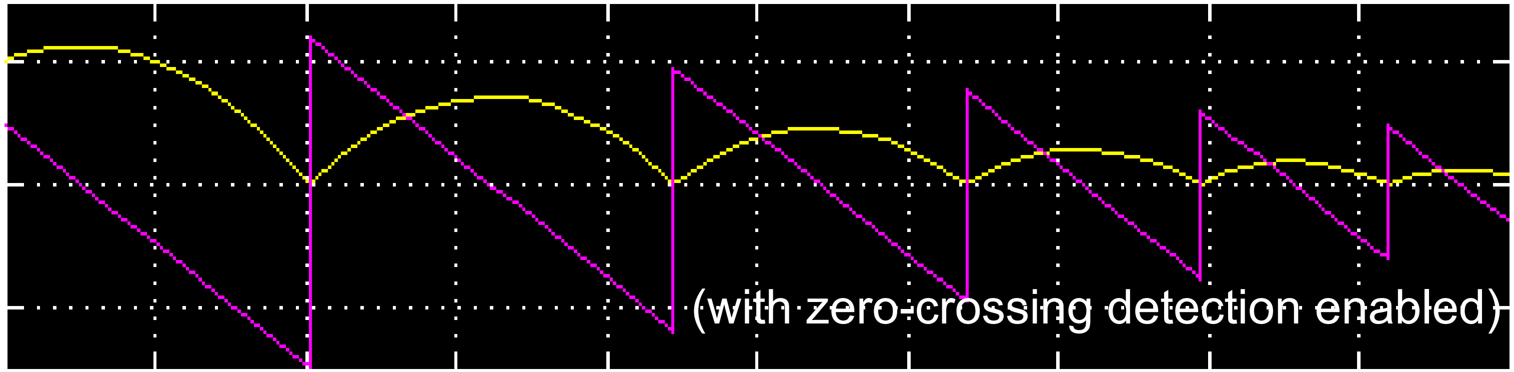


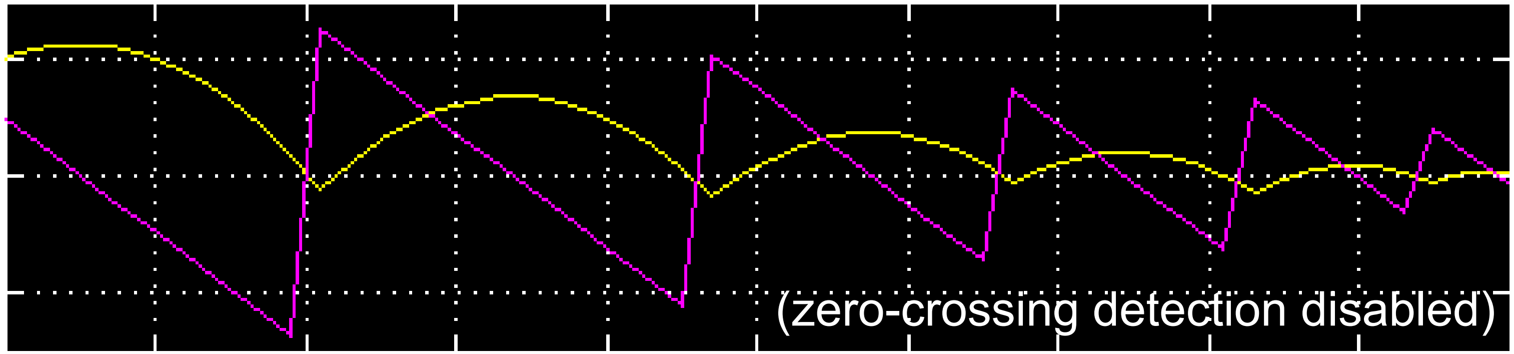




ODEs with resets in Simulink (3)

- Simulink ODE solver detects zero-crossing behaviour
- When a reset is detected, the solver goes "back in time" to determine where the reset occurred



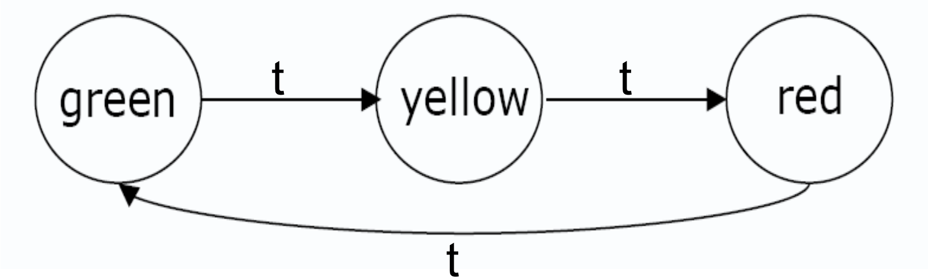


3.

Finite State Machines

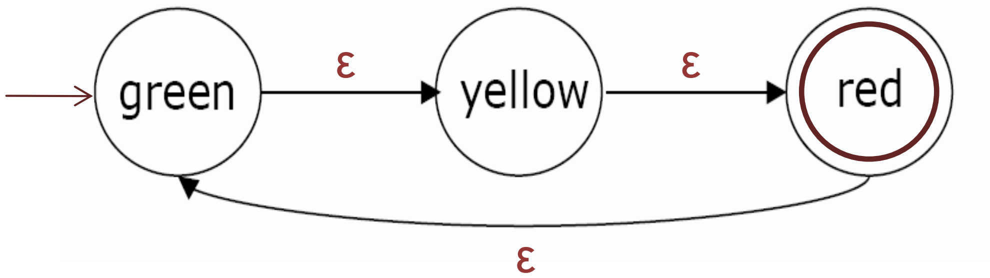
Finite State Machines (FSMs)

- Model of systems whose behavior can be modeled as a set of states and transitions between states. This systems are sometimes called reactive systems.
- Finite number of states
- Systems modeled by FSMs:
  - Pattern recognition
  - ATMs
  - Computational processes
  - Human intelligence?



Mathematical Model of FSMs

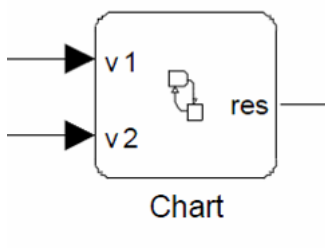
- A FSM is a quintuple $(\Sigma, S, s_0, \delta, F)$, where:
  - Σ is an input alphabet (finite set of symbols)
  - S is a finite, non-empty, set of states
  - S_0 is the initial state, where $S_0 \in S$
  - δ is a state-transition function: $\delta: S \times \Sigma \rightarrow S$
  - F is a finite, (possibly empty) set of final states



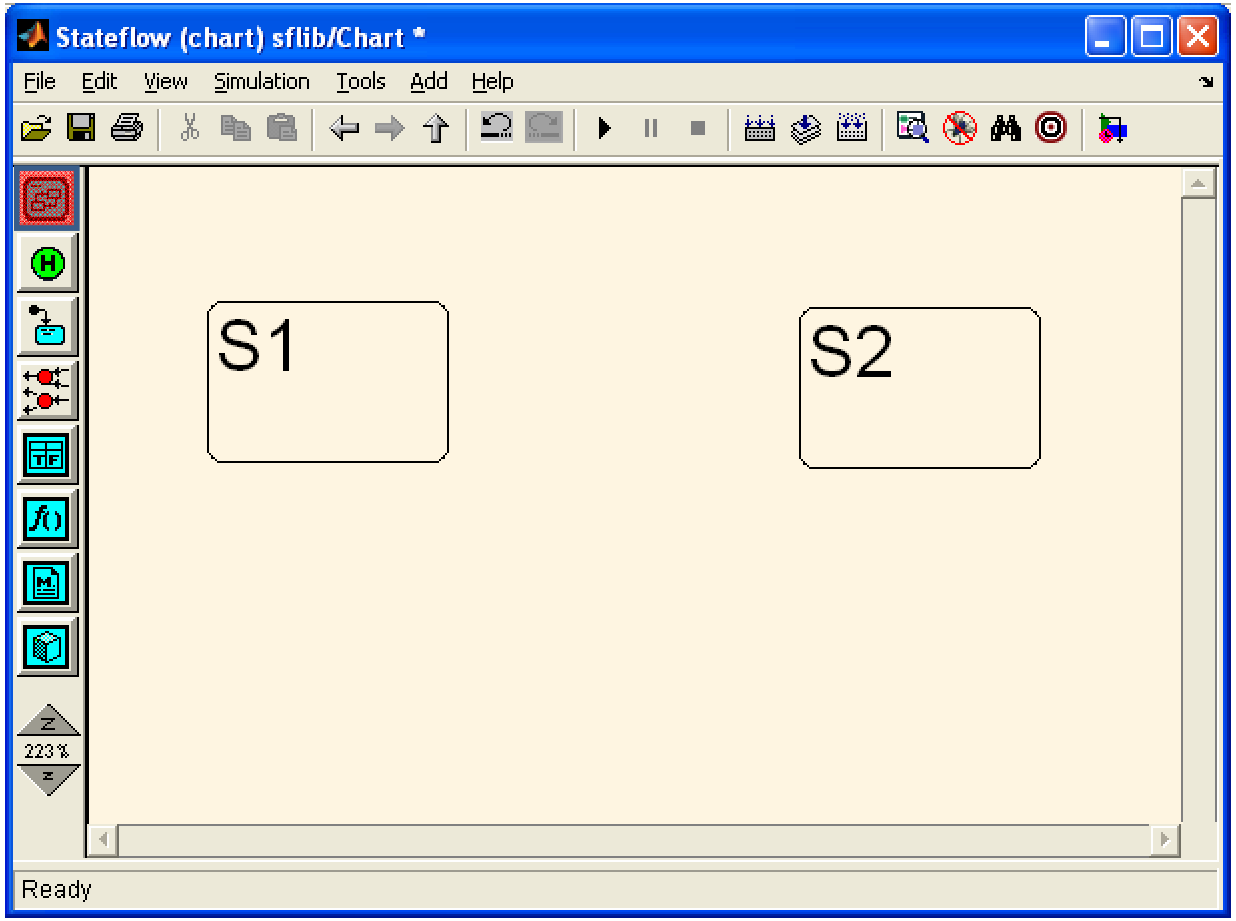
Σ = [ε]
S = [green, yellow, red]
S_0 = green
δ = [green/ε $\rightarrow$ yellow, yellow/ε $\rightarrow$ red, red/ε $\rightarrow$ green]
F = [red]

Simulation of FSMs - StateFlow

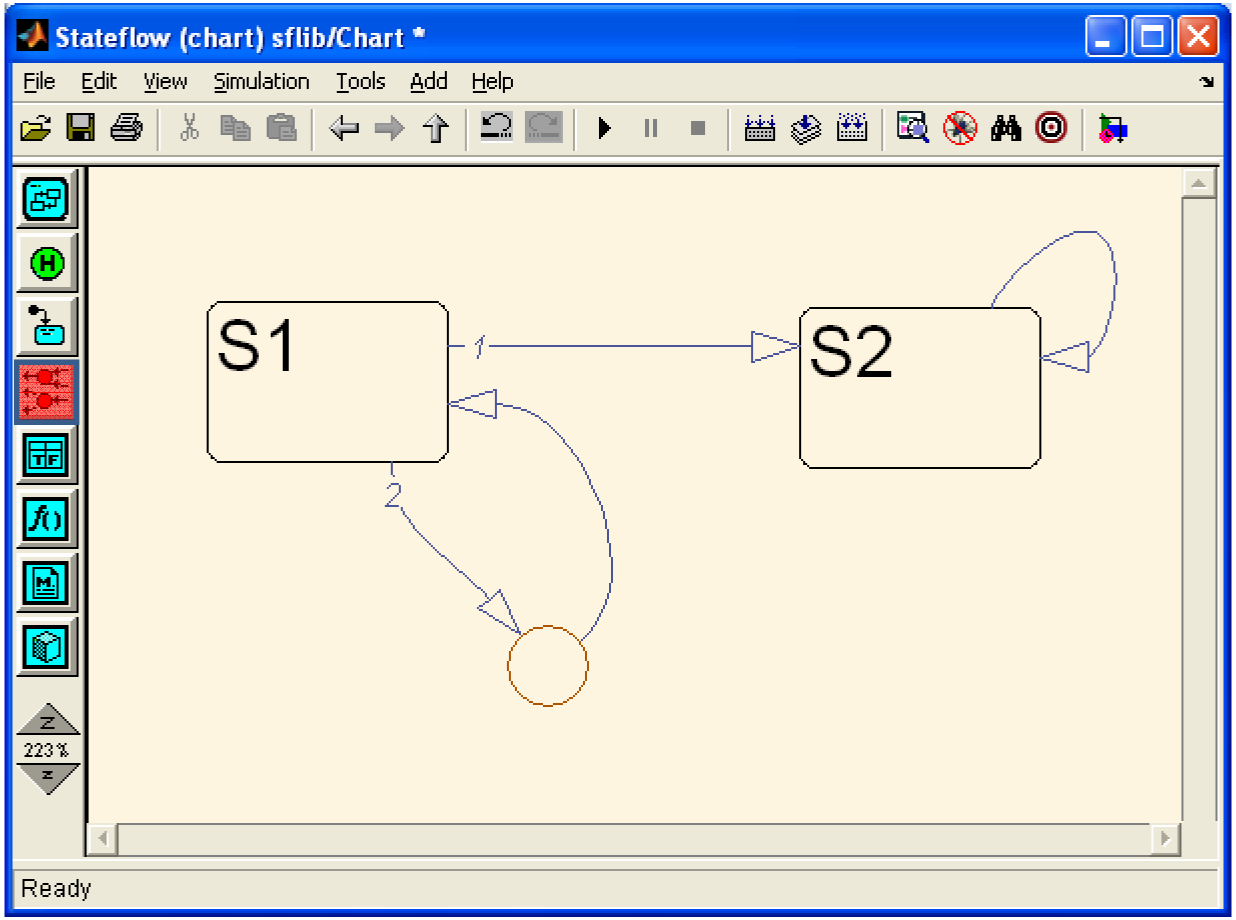
- Simulink block (toolbox) for modeling Finite State Machines
- Stateflow charts receive inputs from Simulink and provide outputs (signals, events)
- Simulation advances with time
- Hybrid state machine model that combines the semantics of Mealy and Moore charts with the extended Stateflow chart semantics.



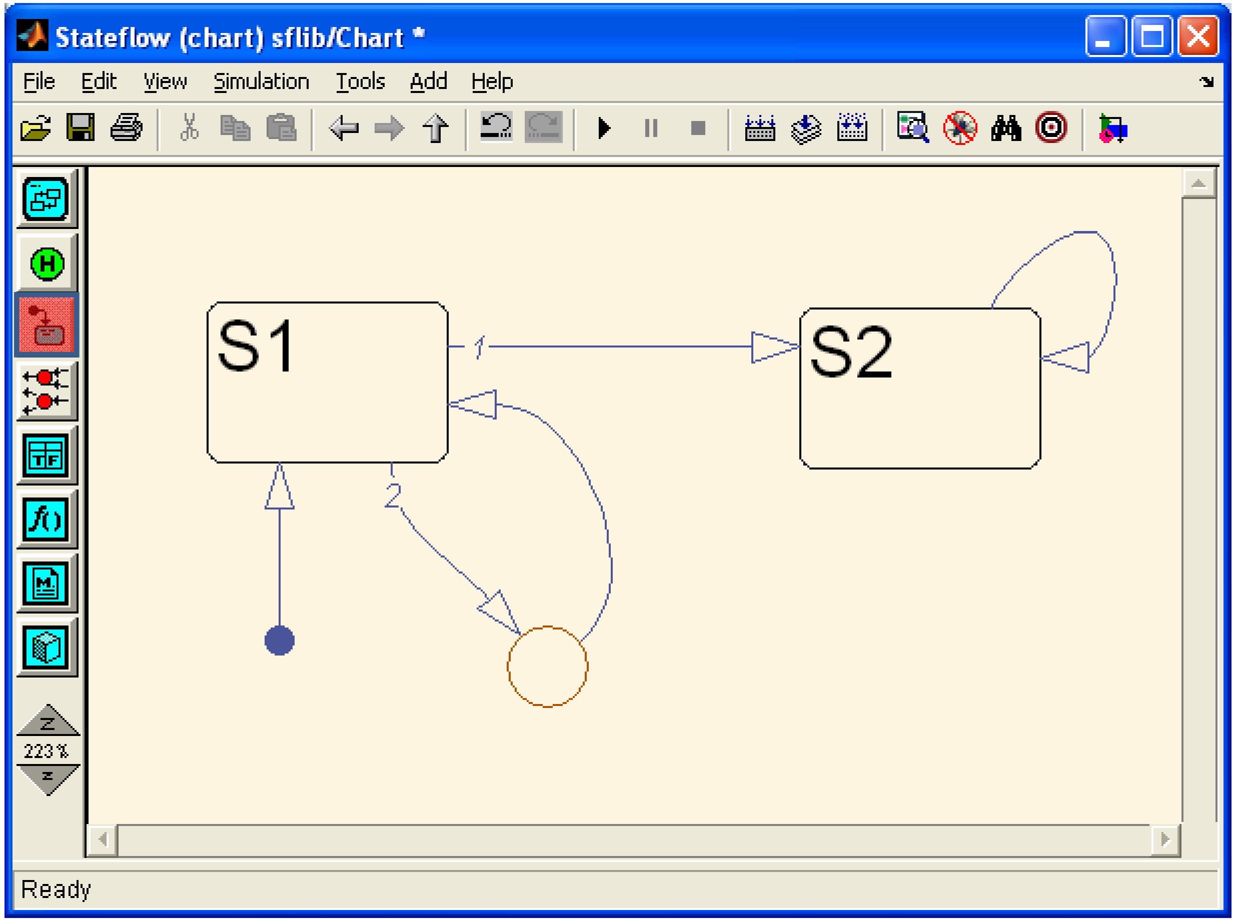
StateFlow – States



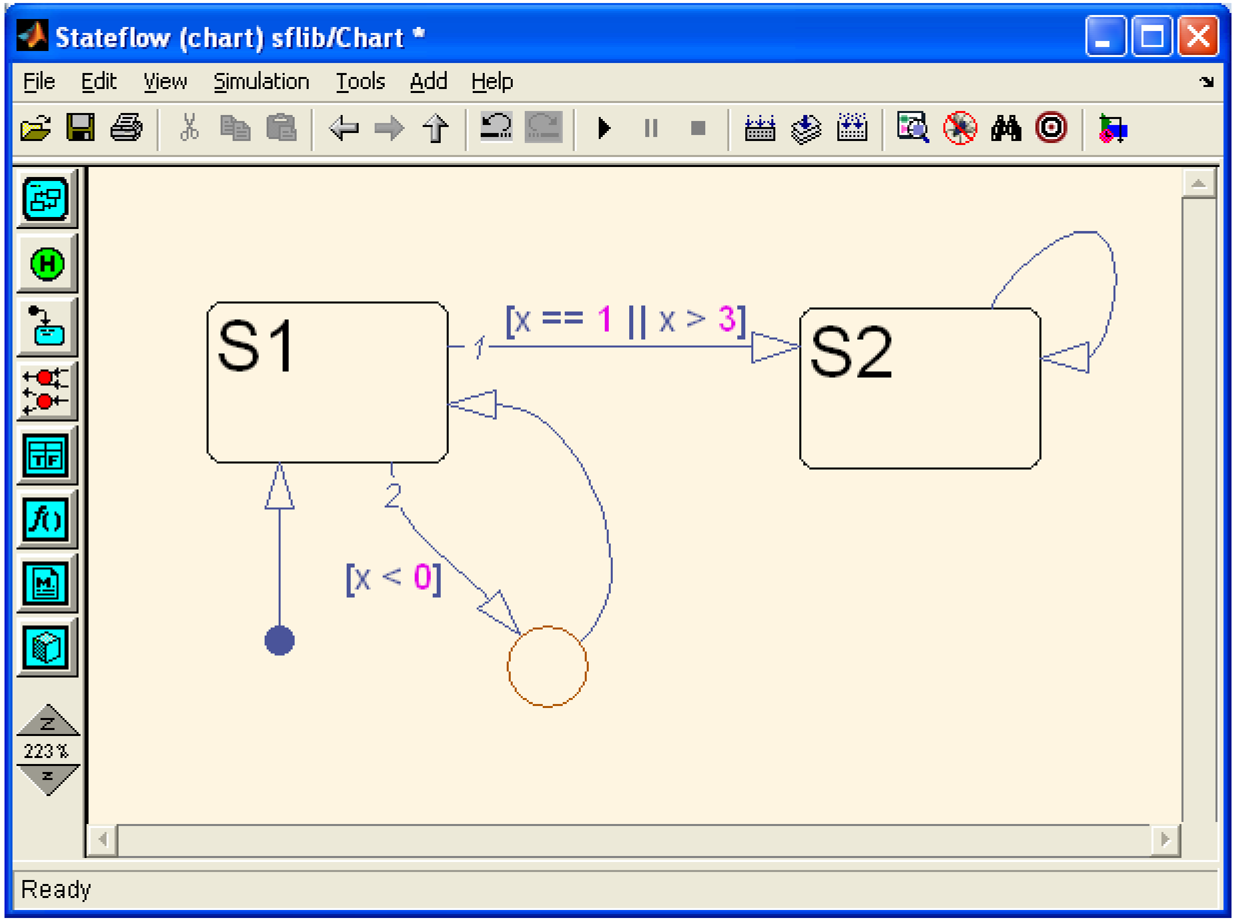
StateFlow – Transitions



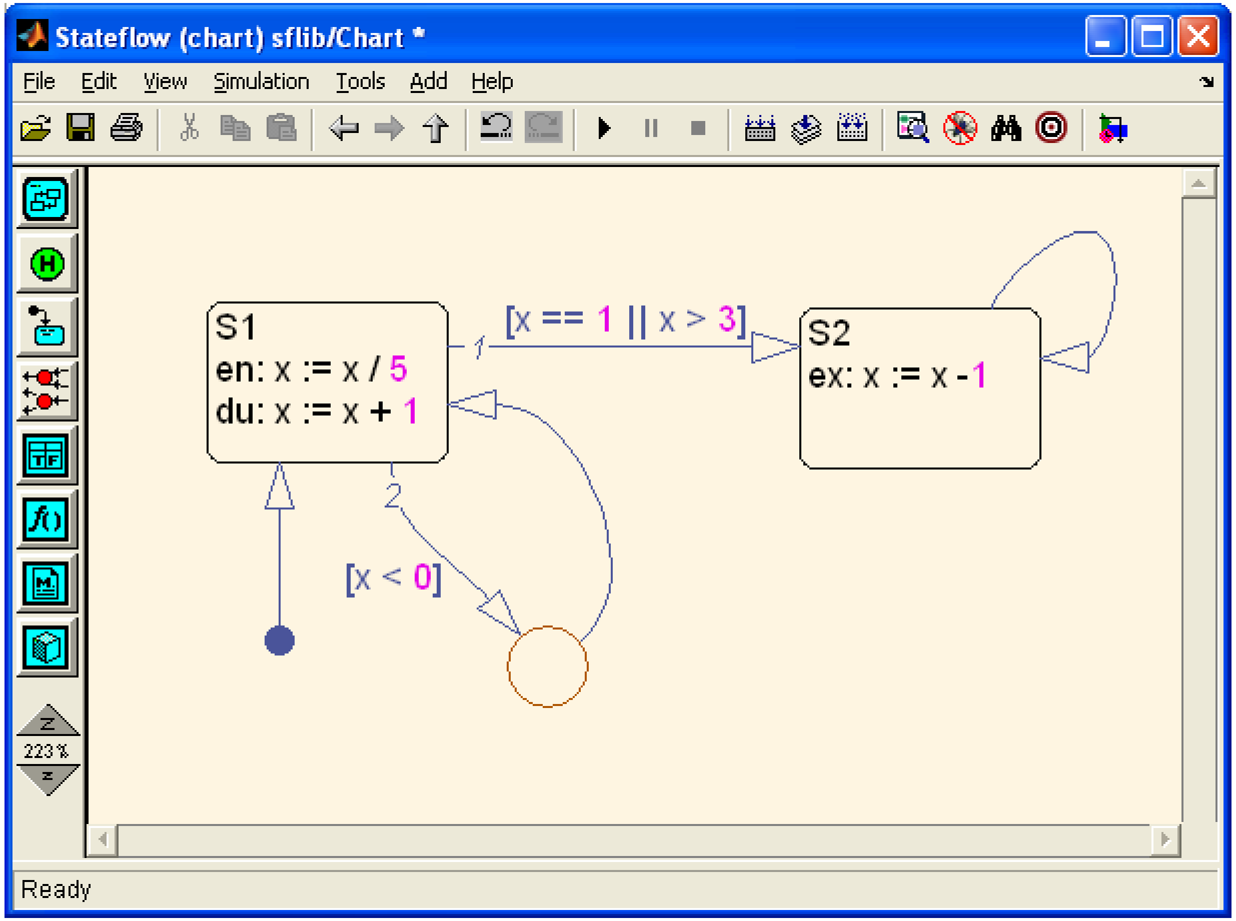
StateFlow – Initial State



StateFlow – Transition Conditions

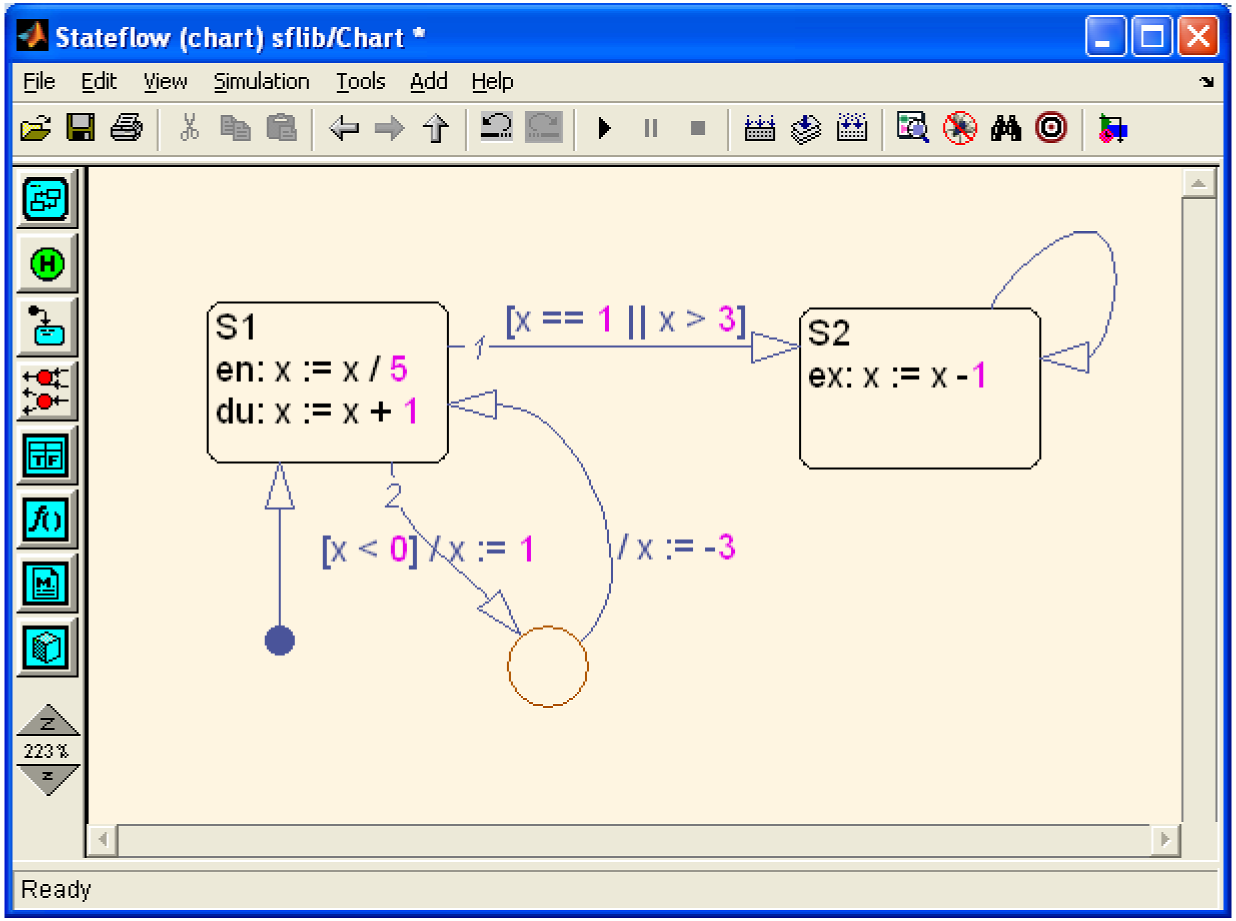


StateFlow – State Actions

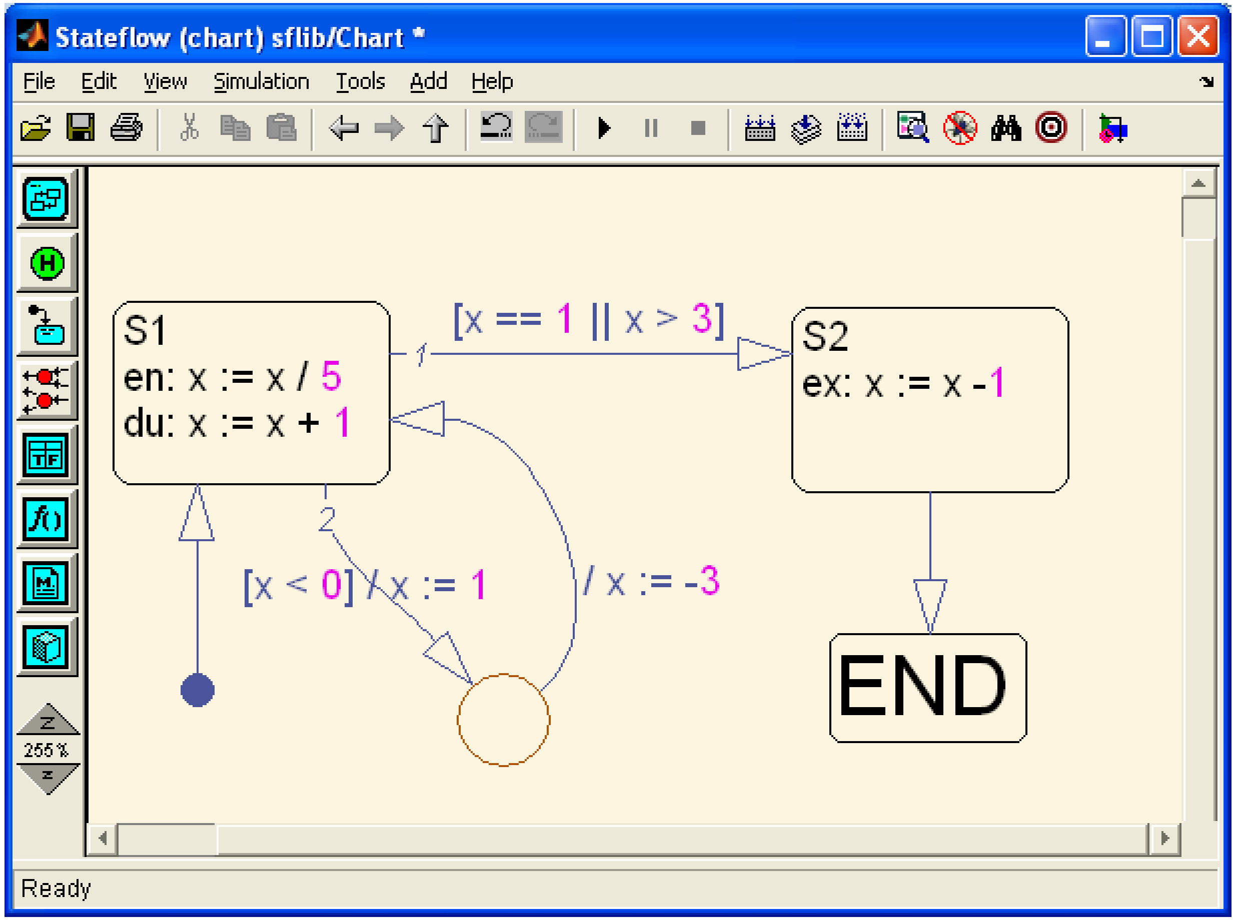


entry: Quando entra no estado, **during**: enquanto está no estado, **exit**: quando sai do estado

StateFlow – Transition Actions

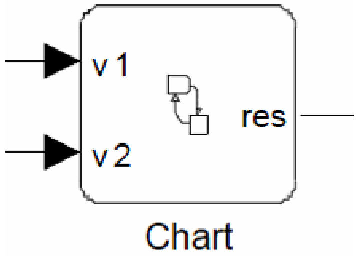


StateFlow – Final States



Stateflow integration with Simulink

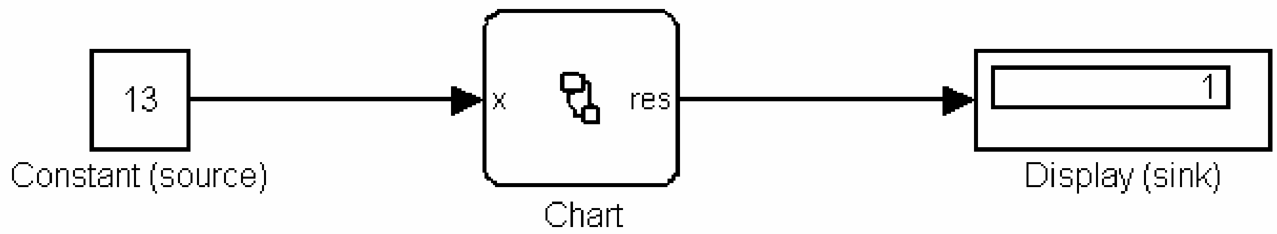
- User defines variables to be used inside Stateflow chart
- Variable types are important!
- Variables may be:
  - Inputs from Simulink
  - Outputs to Simulink
  - Local, Constant, …



- To define variables use Model Explorer (Ctrl-R)

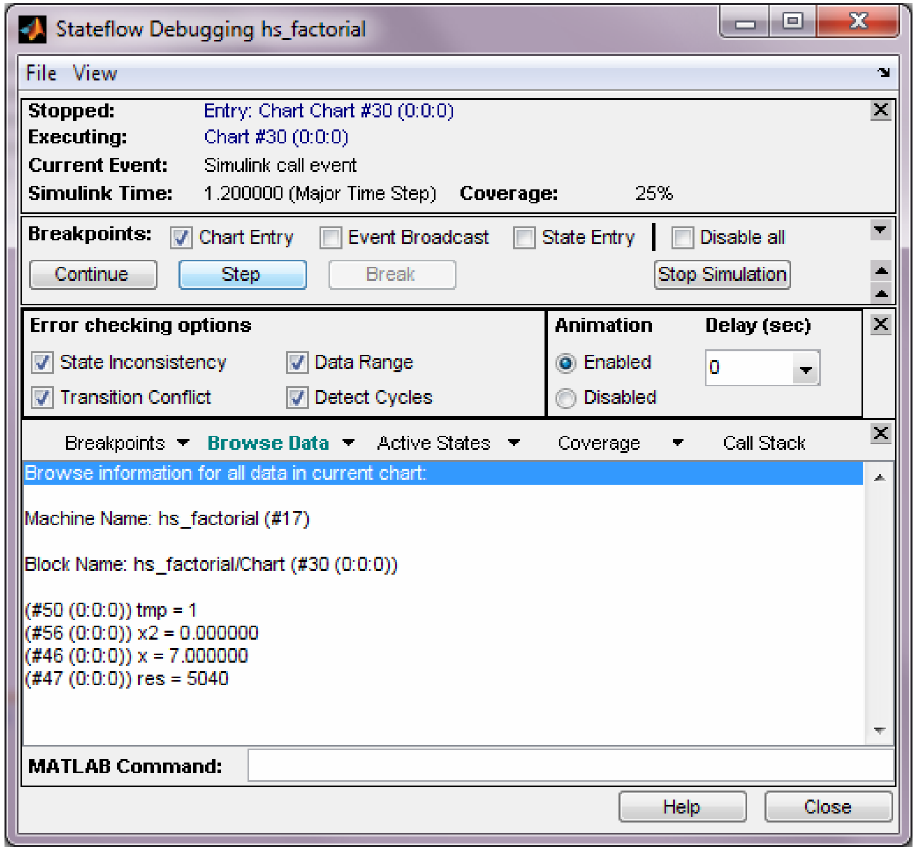
Simulink/Stateflow Hands-on

- Create a stateflow chart that calculates the factorial of a number (*uint32*), given as input



Stateflow Debugger

- Use “step” to execute the FSM step-by-step
- Use “browse data” to monitor variable changes



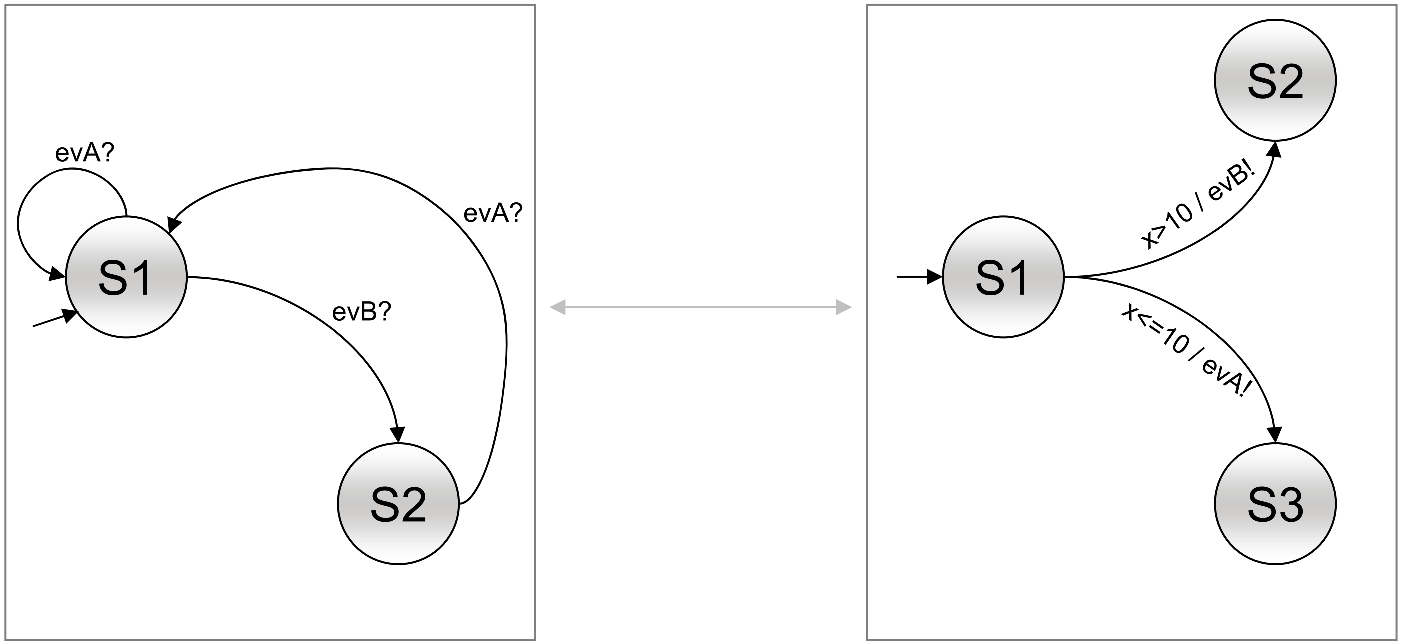
4.

Discrete Event Systems

Discrete Event Systems

- Discrete and qualitative changes
- State transitions caused by occurrence of asynchronous discrete events
- Parallel execution of multiple systems
- Synchronization by event-passing

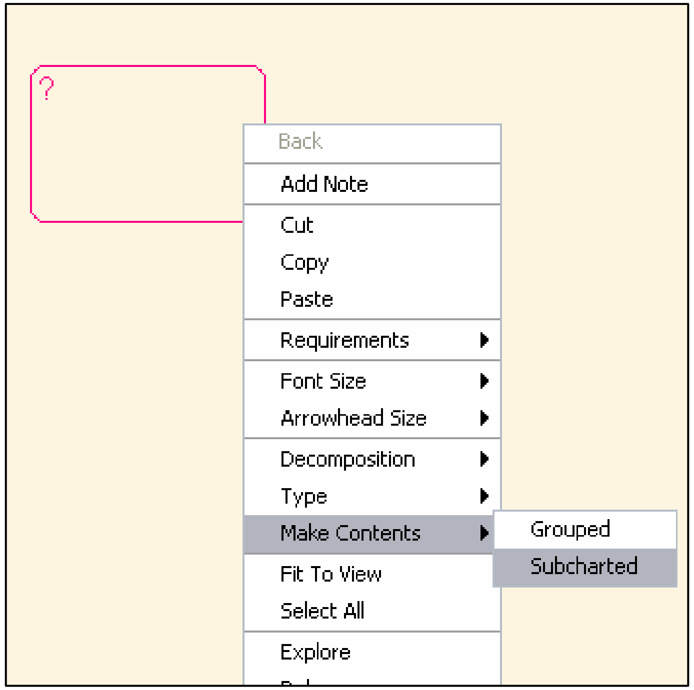
DES Example



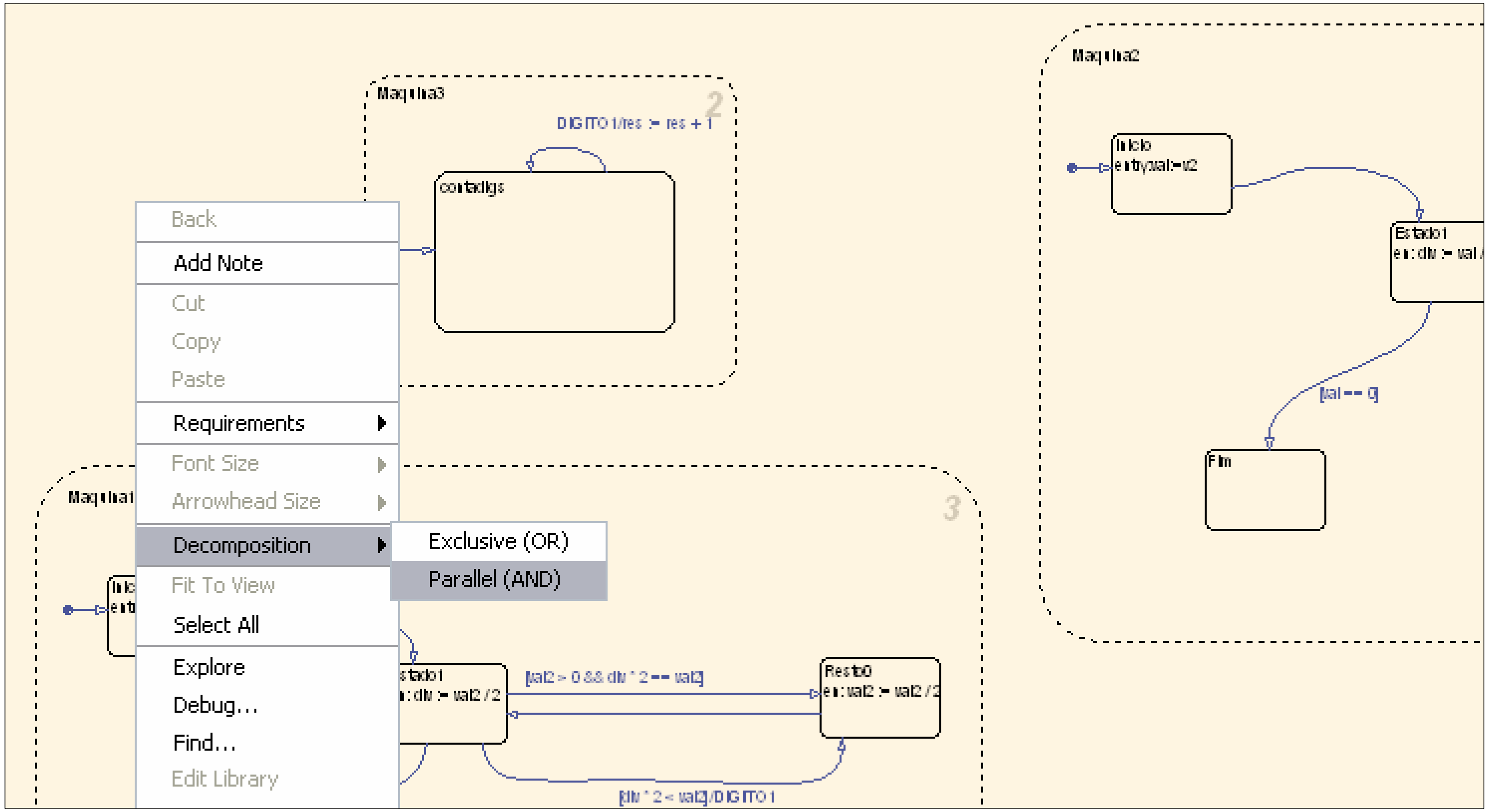
- Synchronization through events
- Parallel execution
- receive? / broadcast!

Machine composition in Stateflow

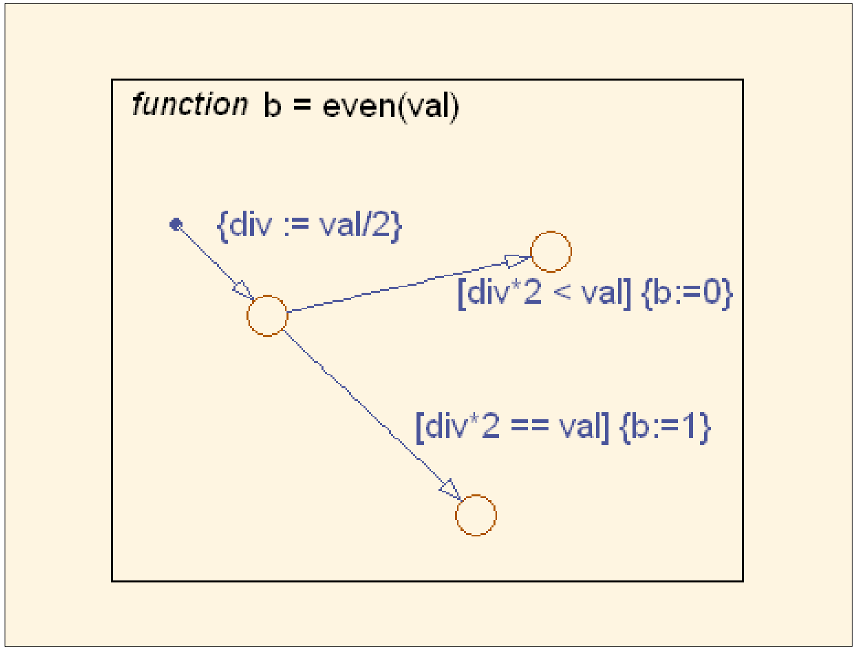
- Grouped: Inner states are visible
- Subcharted: Creates a sub-chart
- Default behavior: sub-chart is executed



Parallel composition in Stateflow

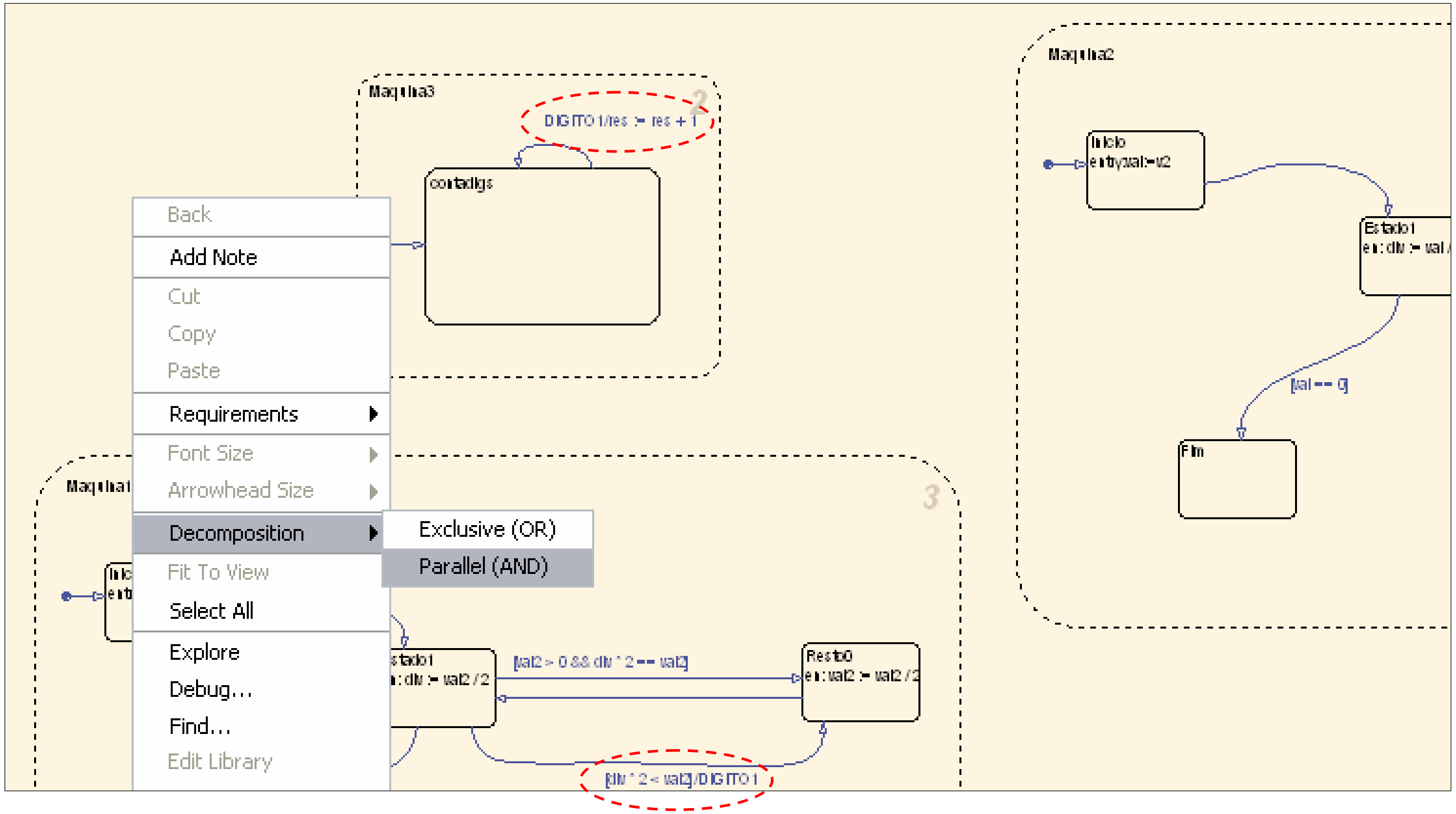


Composition: Graphical functions



- [even(x)] and [~even(x)] are now valid conditions
- {actions} are called condition actions and are evaluated even when destination (state) is not valid

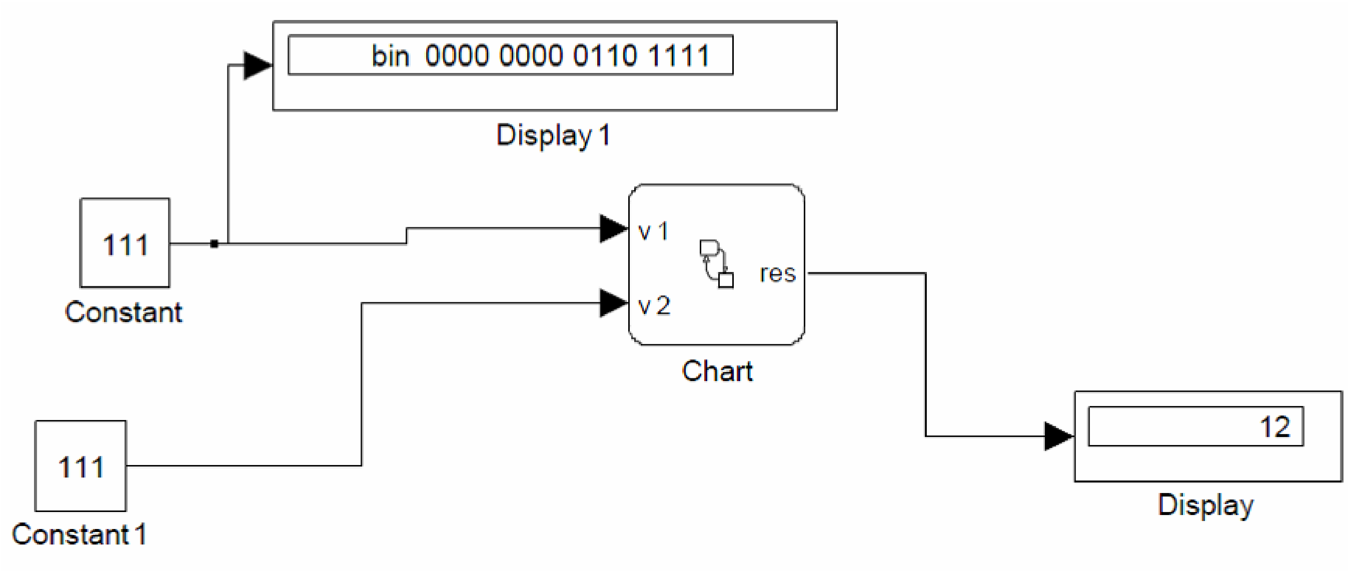
Event passing in Stateflow



- Use Model Explorer to add Event data types 
- Events used both as conditions and actions

DES in Stateflow Hands-on

- Create a stateflow chart that receives 2 inputs (uint32) and sums the number of 1 digits in their binary representation
- Use event passing to count all the 1 digits

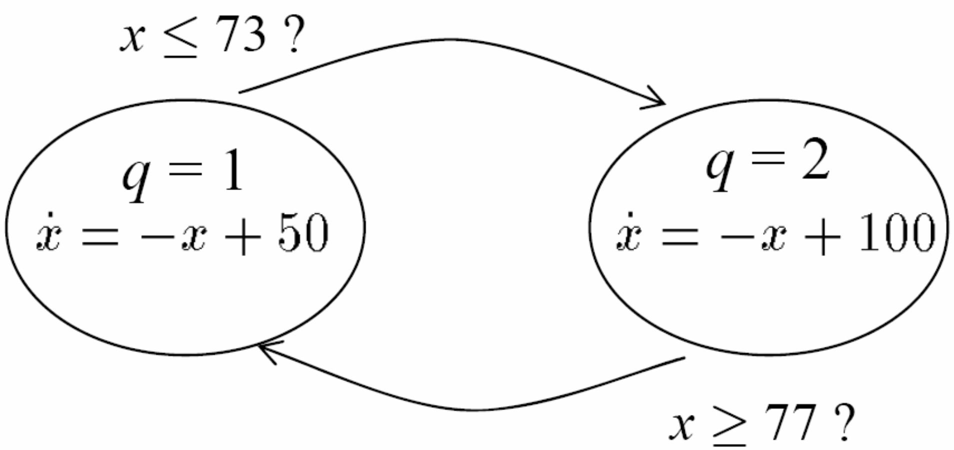


4.

Hybrid Automata

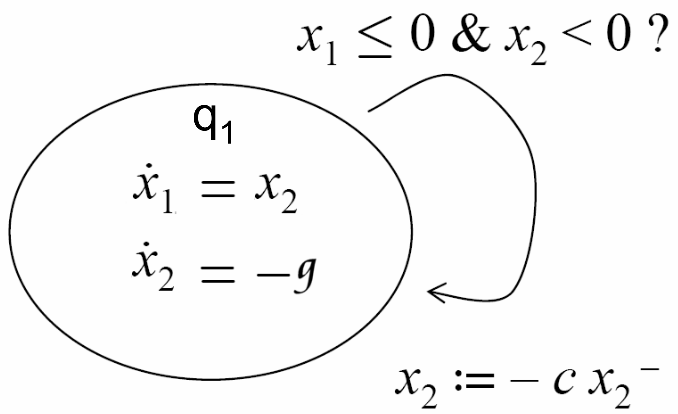
Hybrid Automata

$$\mathcal{Q} \equiv \text{set of discrete states}$$
$$\mathbb{R}^n \equiv \text{continuous state-space}$$
$$f : \mathcal{Q} \times \mathbb{R}^n \to \mathbb{R}^n \equiv \text{vector field}$$
$$\varphi : \mathcal{Q} \times \mathbb{R}^n \to \mathcal{Q} \equiv \text{discrete transition}$$
$$\rho : \mathcal{Q} \times \mathbb{R}^n \to \mathbb{R}^n \equiv \text{reset map}$$



[Hespanha, J. P. 05]

Bouncing Ball re-revisited



Q	$\equiv$ set of discrete states
$\mathbb{R}^n$	$\equiv$ continuous state-space
$f : Q \times \mathbb{R}^n \to \mathbb{R}^n$	$\equiv$ vector field
$\varphi : Q \times \mathbb{R}^n \to Q$	$\equiv$ discrete transition
$\rho : Q \times \mathbb{R}^n \to \mathbb{R}^n$	$\equiv$ reset map

$$Q = \{q_1\}$$

$$f(q, x_1, x_2) = \{q_1 \longrightarrow \dot{x}_1 := x_2, \dot{x}_2 := -g\}$$

$$\varphi(q, x_1, x_2) = \{q_1, x_1 \leq 0 \wedge x_2 < 0 \longrightarrow q_1\}$$

$$\rho(q, x_1, x_2) = \{q_1, x_1 \leq 0 \wedge x_2 < 0 \longrightarrow x_2 := -c \times x_2\}$$

Simulation of Hybrid Automata

$Q \equiv$ set of discrete states

$\mathbb{R}^n \equiv$ continuous state-space

$f : Q \times \mathbb{R}^n \to \mathbb{R}^n \equiv$ vector field

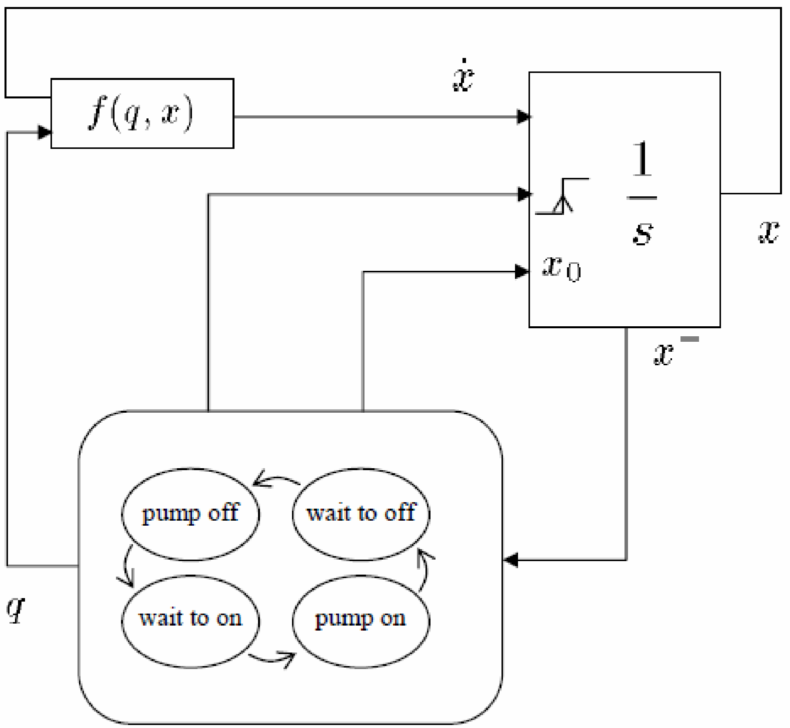
$\varphi : Q \times \mathbb{R}^n \to Q \equiv$ discrete transition

$\rho : Q \times \mathbb{R}^n \to \mathbb{R}^n \equiv$ reset map

$x := \rho(q_1, x^-)$ $\quad \varphi(q_1, x^-) = q_2$?

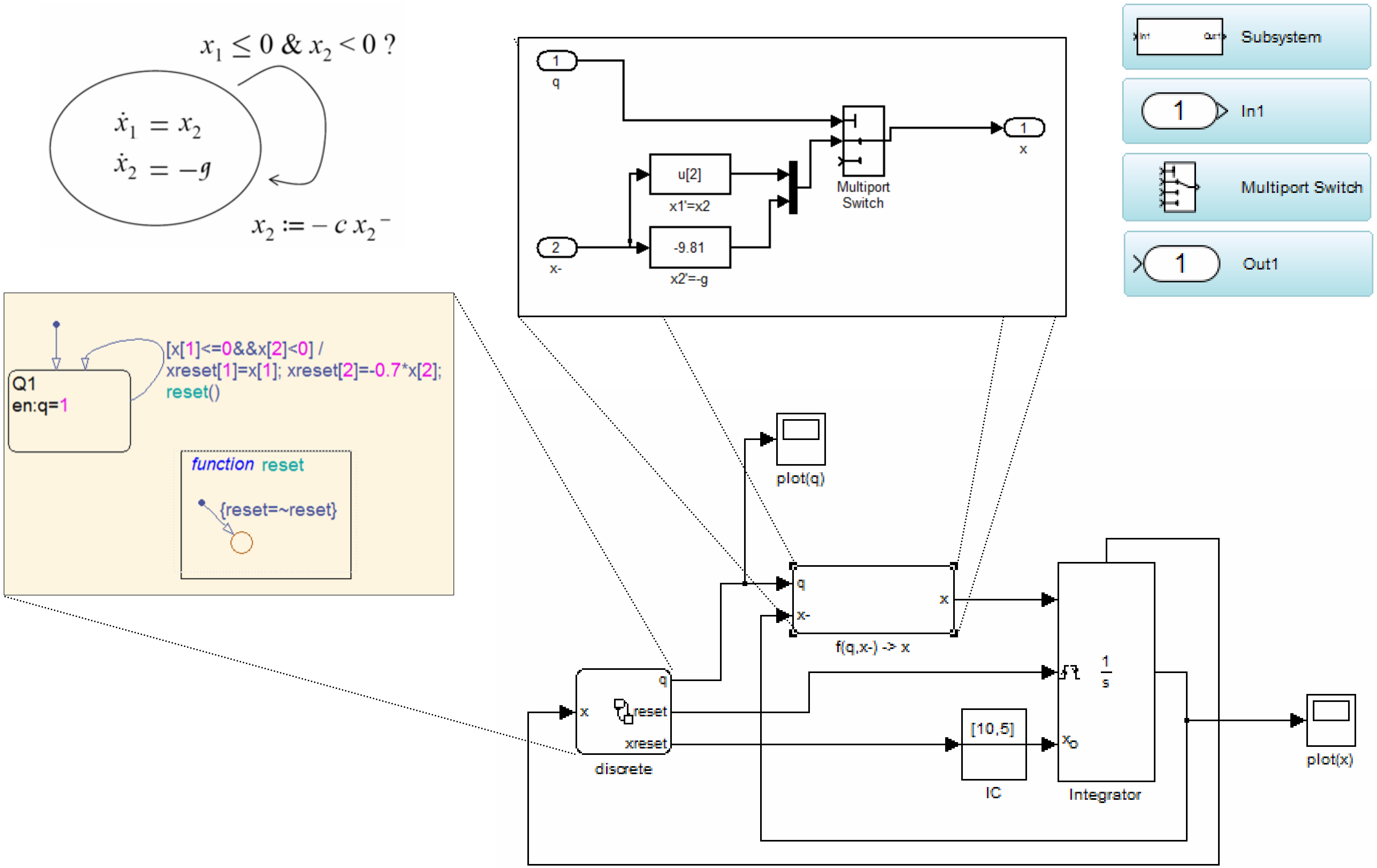
mode q_2: $\dot{x} = f(q_2, x)$

mode q_1: $\dot{x} = f(q_1, x)$



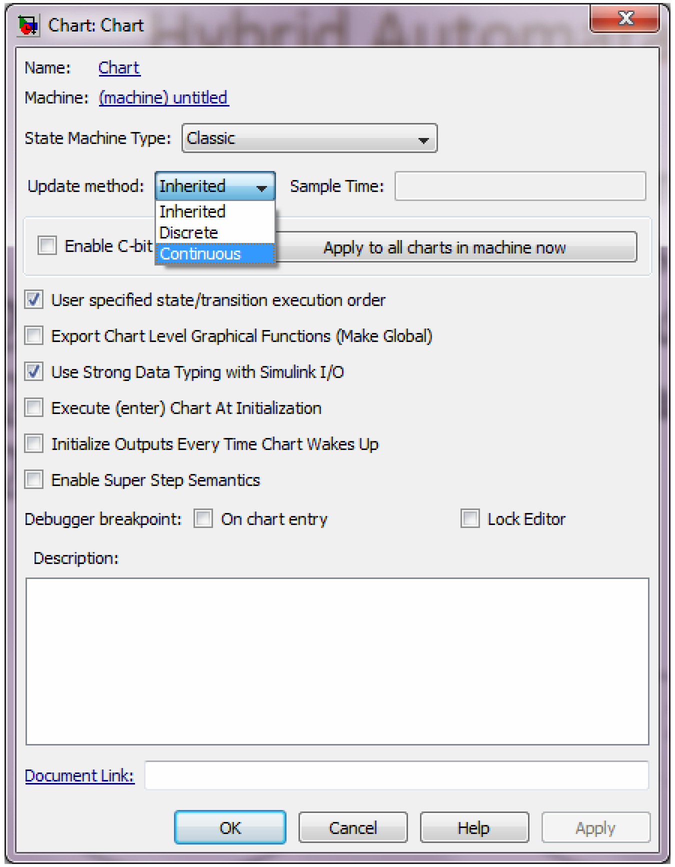
[Hespanha, J. P. 05]

Bouncing Ball in StateFlow



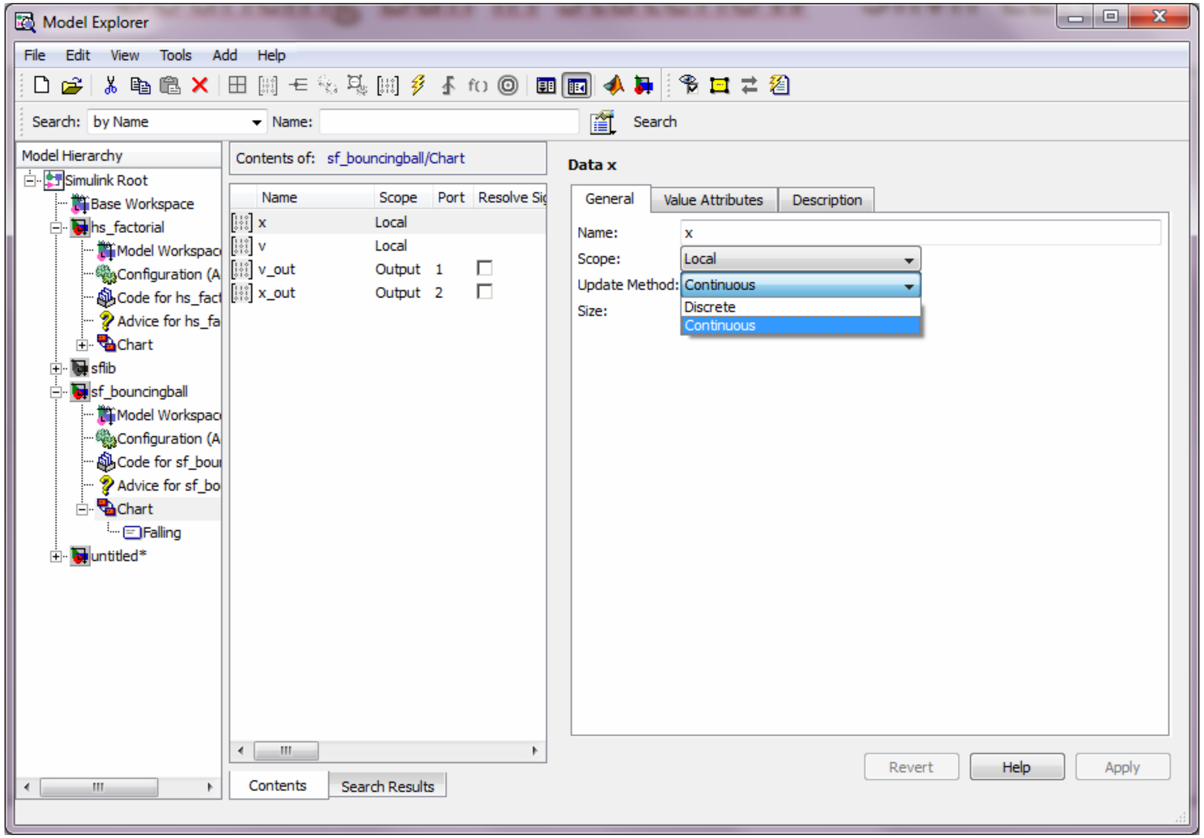
Bouncing Ball in Stateflow - SIMPLER

- Go to “File->Chart properties”…
- Select “Continuous” update method



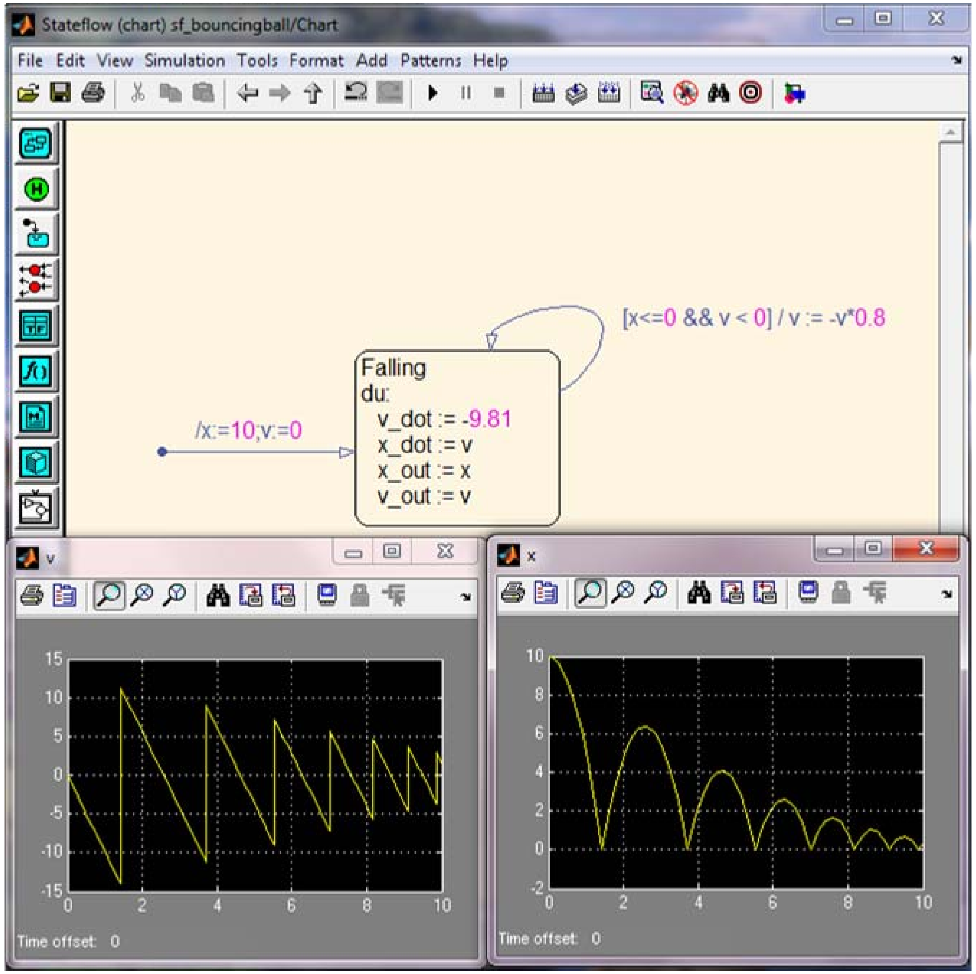
Bouncing Ball in Stateflow - SIMPLER

- In Model Explorer you can now local variables of type double as continuous variables (update method)



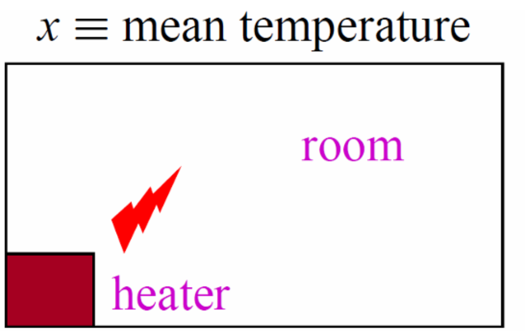
Bouncing Ball in Stateflow - SIMPLER

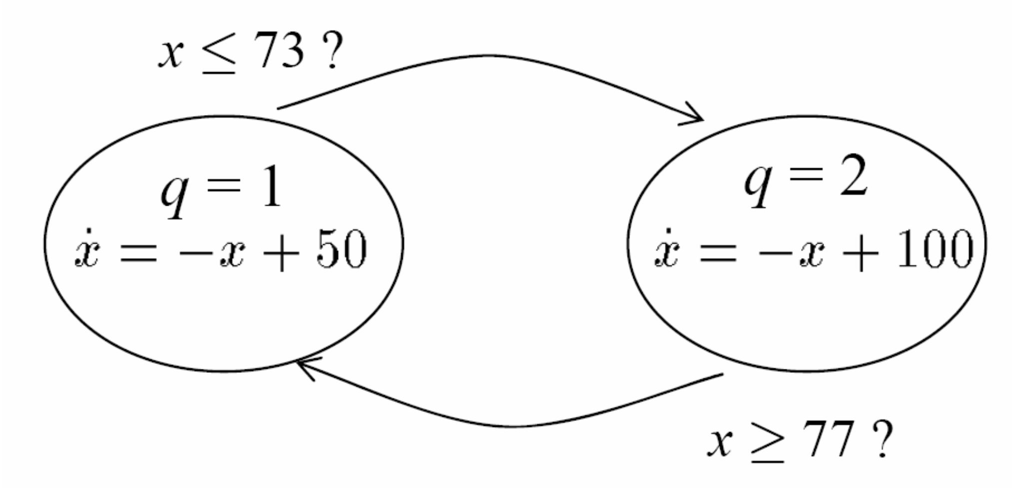
- For each continuous local variable “x”…
- stateflow automatically defines “x_dot”…
- which is derivative of “x”



Exercise 1: Thermostat Hybrid System

$$
\begin{array}{ll}
\mathcal{Q} & \equiv \text{set of discrete states} \\
\mathbb{R}^n & \equiv \text{continuous state-space} \\
f: \mathcal{Q} \times \mathbb{R}^n \rightarrow \mathbb{R}^n & \equiv \text{vector field} \\
\varphi: \mathcal{Q} \times \mathbb{R}^n \rightarrow \mathcal{Q} & \equiv \text{discrete transition} \\
\rho: \mathcal{Q} \times \mathbb{R}^n \rightarrow \mathbb{R}^n & \equiv \text{reset map}
\end{array}
$$

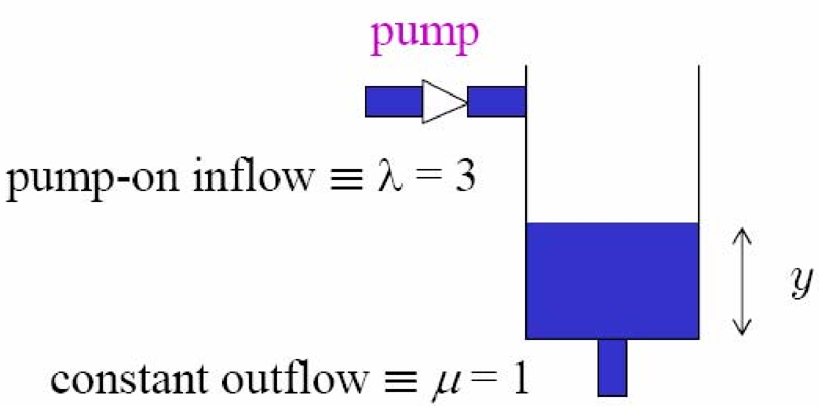




1. Define the HS formally
2. Simulate Hybrid System

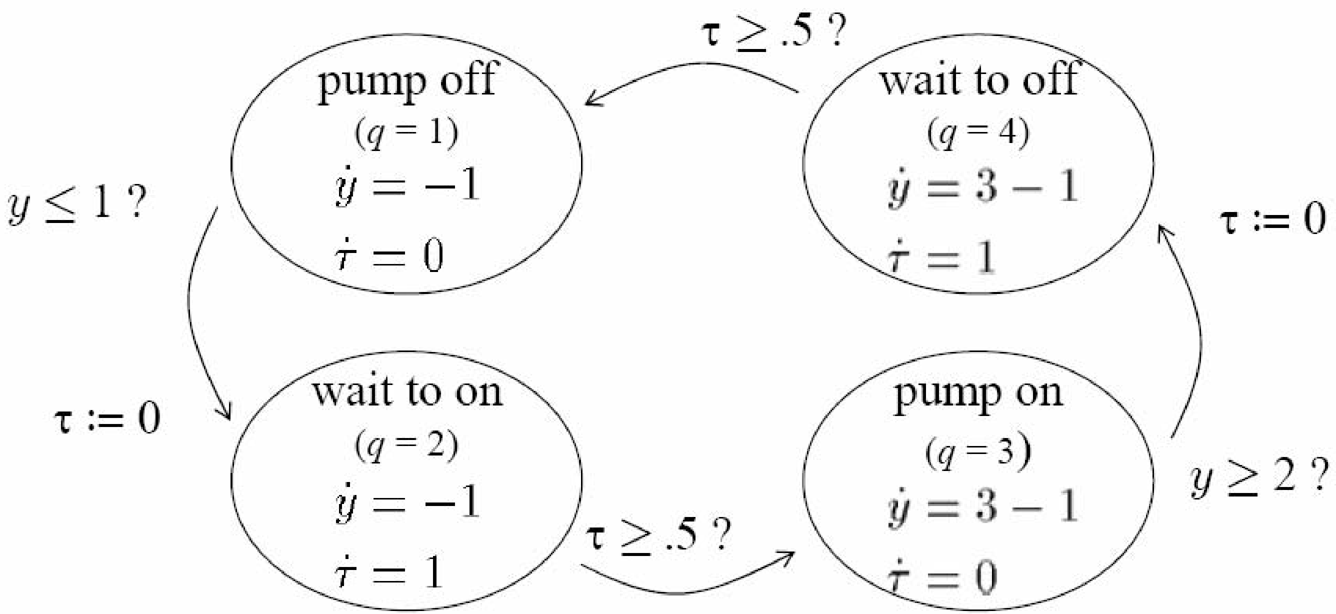
[Hespanha, J. P. 05]

Exercise 2: Tank Hybrid System



goal ≡ prevent the tank from emptying or filling up

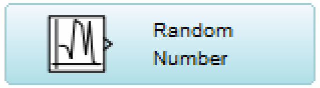
δ = .5 ≡ delay between command is sent to pump and the time it is executed



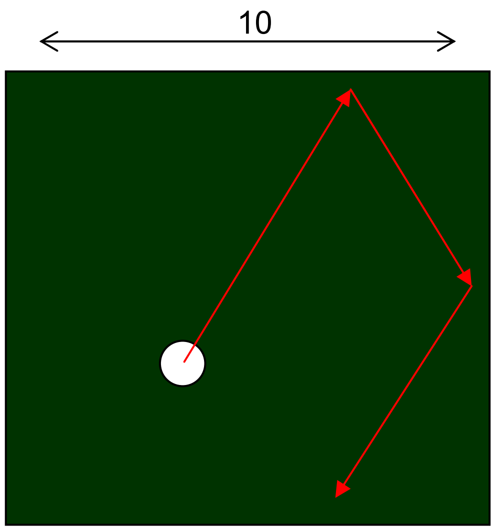
[Hespanha, J. P. 05]

Exercise 3: Billiard Hybrid System

- Ball inside a 10x10 box
- Ball starts at unknown position (x,y) and with unknown velocity vector (v_x,v_y)



- Ball bounces in any of the 4 walls
- Constant energy dissipation factor between 0 and 1



END

References:

- http://www.ece.ucsb.edu/~hespanha/ece229/
- http://www.mathworks.com/access/helpdesk/help/pdf_doc/stateflow/sf_ug.pdf