Hybrid Systems
Modeling, Analysis and Control

Radu Grosu
Vienna University of Technology

Lecture 6

Continuous AND Discrete Systems

Control Theory

Continuous systems

approximation, stability
control, robustness

Computer Science

Discrete systems

abstraction, composition
concurrency, verification

Hybrid Systems

Software controlled systems

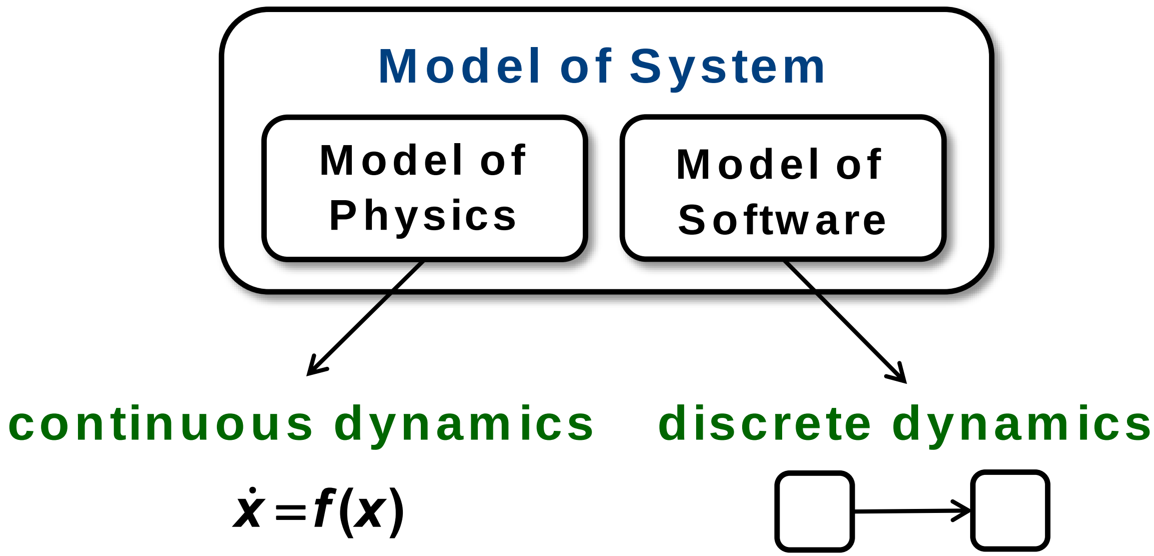
Embedded real-time systems
Multi-agent systems

Models and Tools

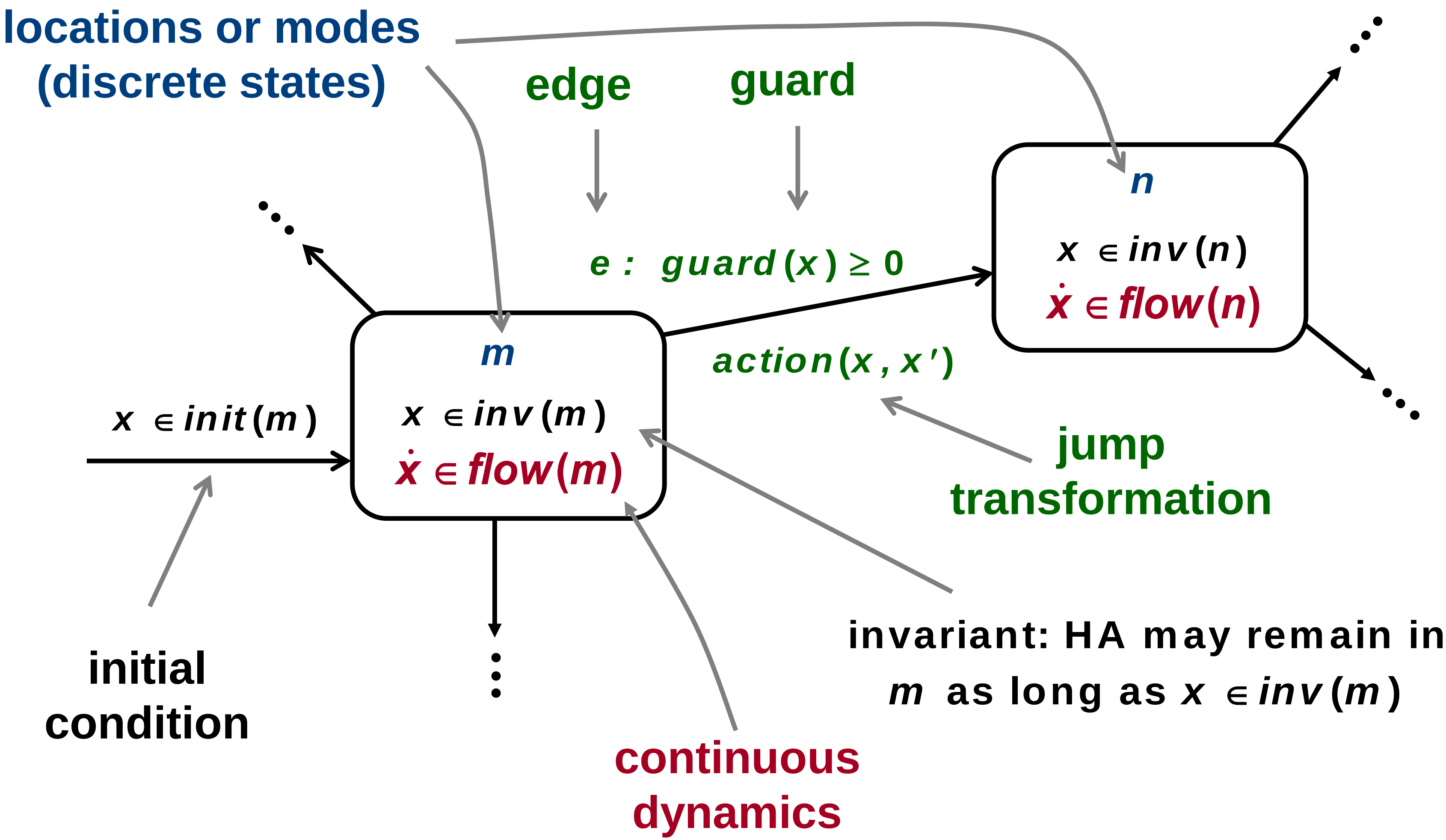
Dynamic systems with continuous & discrete state variables

	Continuous Part	Discrete Part
Models	Differential equations, transfer functions,	Automata, Petri nets, Statecharts,
Analytical Tools	Lyapunov functions, eigenvalue analysis,	Boolean algebra, formal logics, verification,
Software Tools	Matlab, Matrix_x, VisSim,	Statemate, Rational Rose, SMV,

Modeling a Hybrid System



Hybrid Automaton (HA)

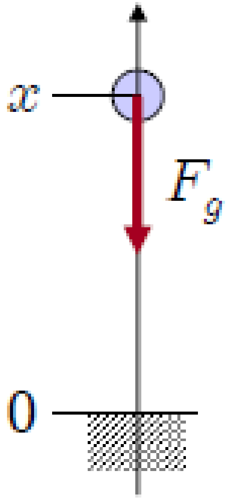


Example: Bouncing Ball

Ball has mass *m* and position *x*

Ball initially at position x_0 and at rest

Ball bounces when hitting ground at $x = 0$



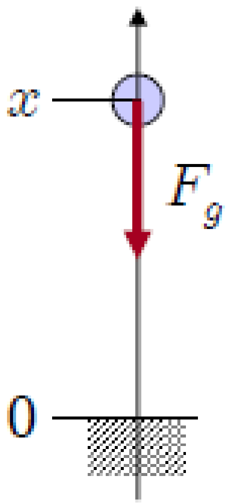
Bouncing Ball: Free Fall

Condition for free fall: $x \geq 0$

Physical law: $F_g = -mg = -m\ddot{x}$

Differential equations: $\dot{x} = v$

First order $\dot{v} = -g$

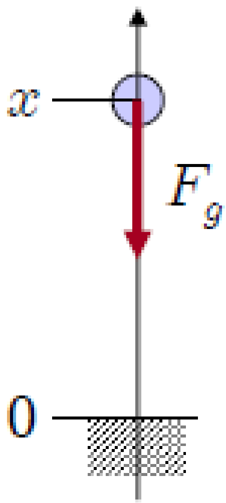


Bouncing Ball: Bouncing

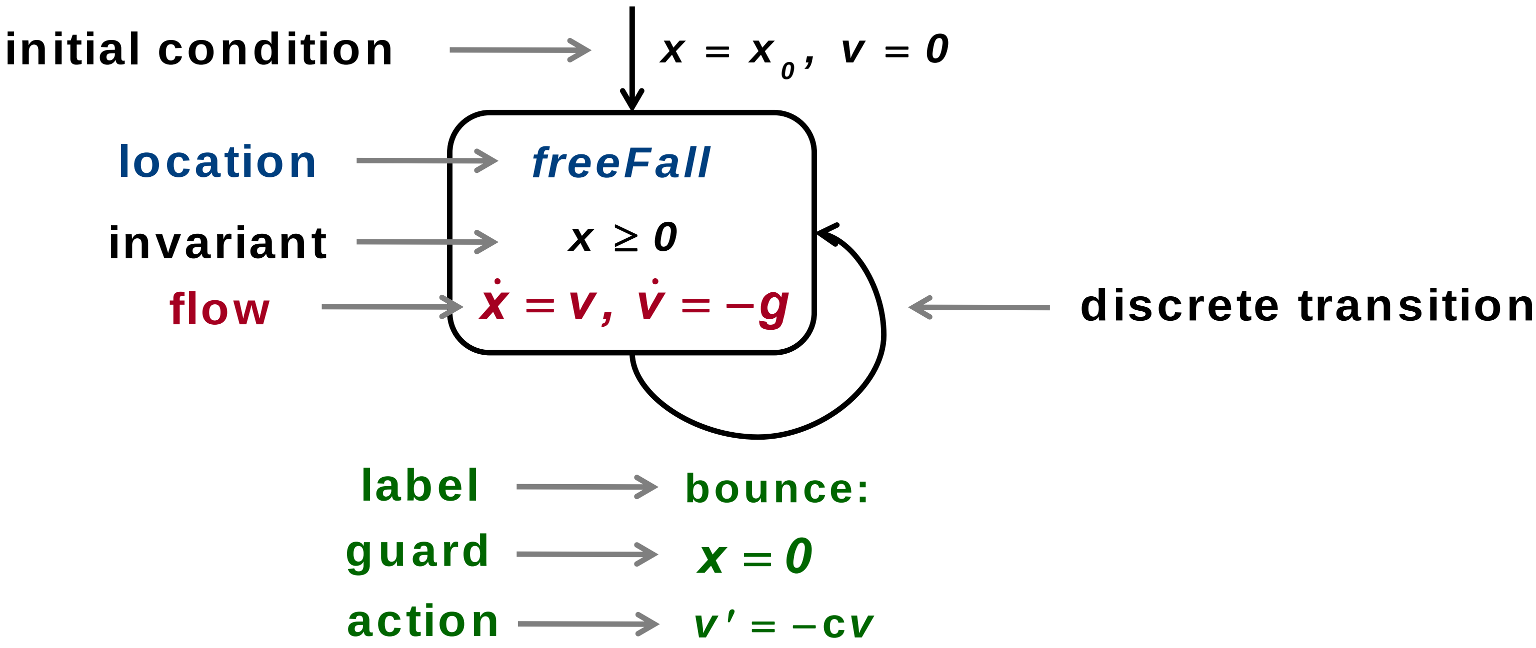
Condition for bouncing: $x = 0$

Action for bouncing: $v' = -cv$

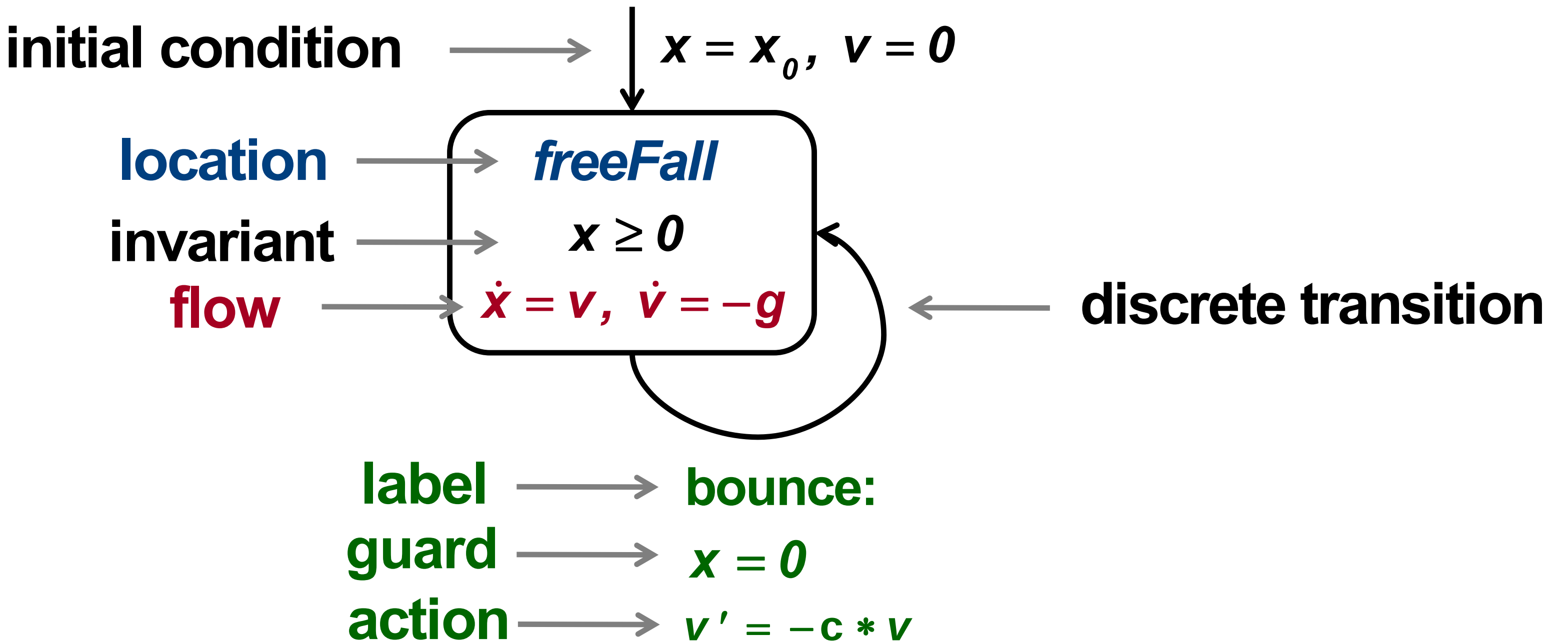
Coefficient c: **deformation, friction**



Bouncing Ball: Hybrid Automaton

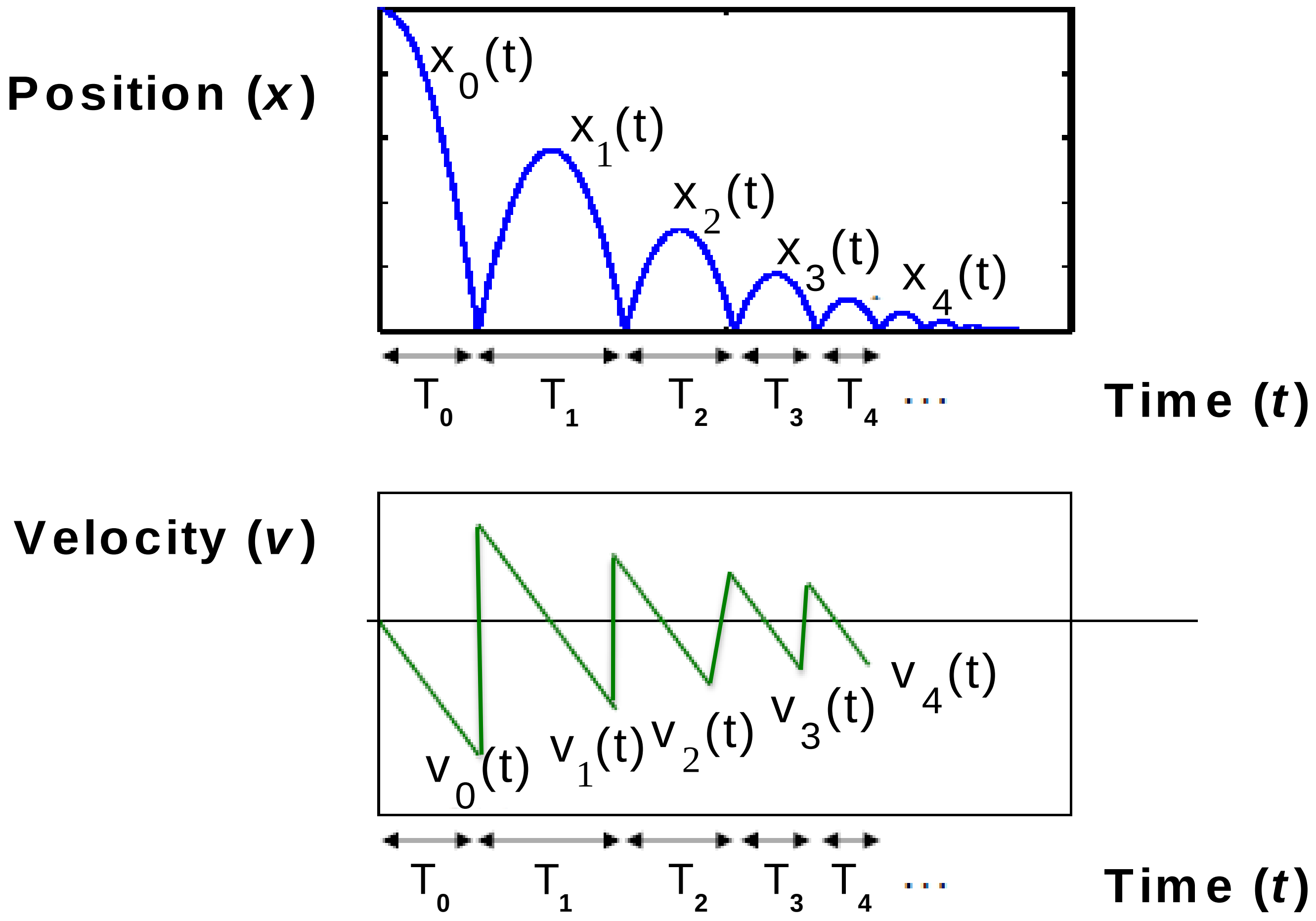


Bouncing Ball: Associated Program

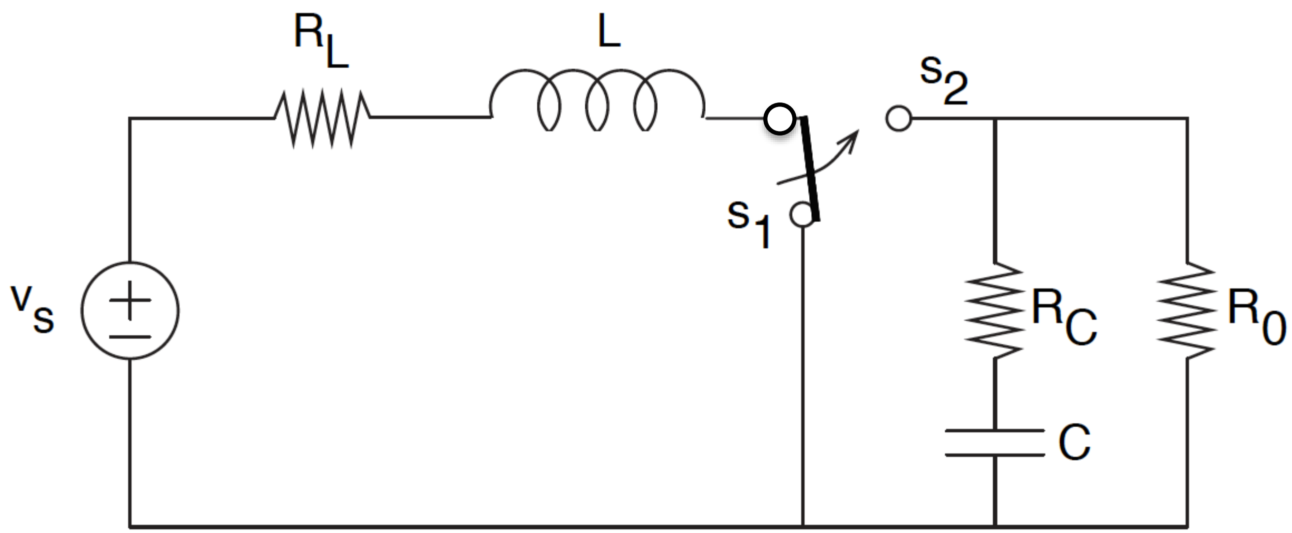


float $x = x_0,\ v = 0;\ d = d_0;$ // initial condition
while true { // main loop
  while ($x \geq 0$) { // free fall
    $x = x + v * d;$ // $\dot{x} = v$
    $v = v - g * d;$ } // $\dot{v} = -g$
  $v = -c * v;\ x = 0;$ } // bounce

Execution of Bouncing Ball



Boost DC-DC Converter



$i_L = i_0$

$v_c = v_0$

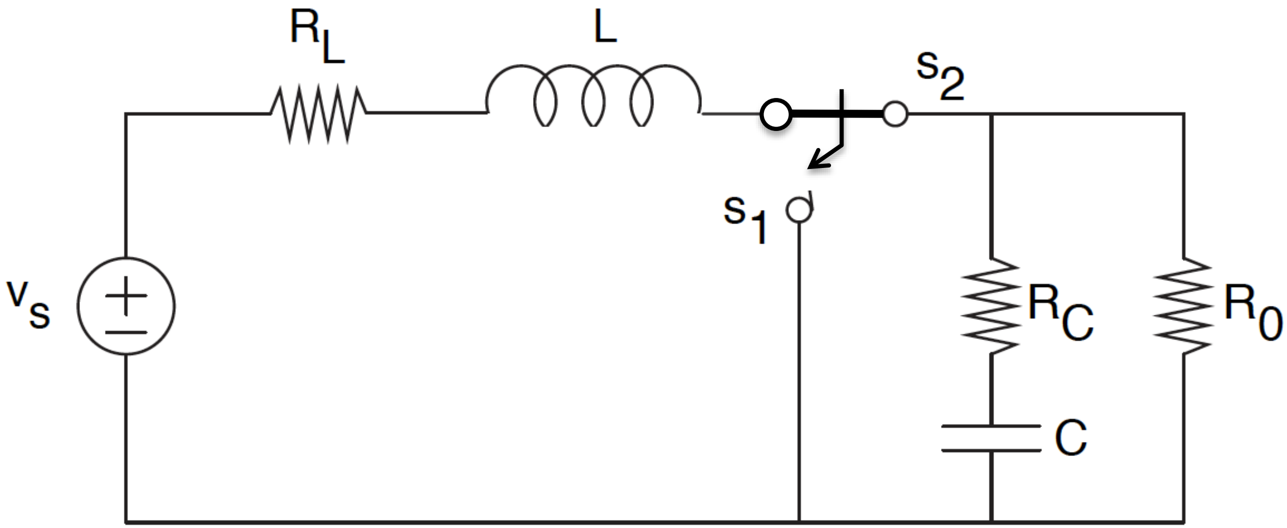
s_0

$s = 0$

$$\dot{i}_L = -\frac{R_L}{L} * i_L + \frac{1}{L} * v_S$$

$$\dot{v}_C = -\frac{1}{C} * \frac{1}{R_C + R_0} * v_C$$

Boost DC-DC Converter



$i_L = i_0$

$v_c = v_0$

$s = 1$

s_0

$s = 0$

$$\dot{i}_L = -\frac{R_L}{L} * i_L + \frac{1}{L} * v_S$$

$$\dot{v}_C = -\frac{1}{C} * \frac{1}{R_C + R_0} * v_C$$

s_1

$s = 1$

$$\dot{i}_L = -\frac{1}{L}\left(R_L + \frac{R_C * R_0}{R_C + R_0}\right) * i_L - \frac{1}{L}\frac{R_0}{R_C + R_0} * v_C + \frac{1}{L} * v_S$$

$$\dot{v}_C = \frac{1}{C} * \frac{R_0}{R_C + R_0} * i_L - \frac{1}{C} * \frac{1}{R_C + R_0} * v_C$$

$s = 0$

Boost DC-DC Converter

$i_L = i_0$

$v_c = v_0$

s_0

$s = 0$

$$\dot{i}_L = -\frac{R_L}{L} * i_L + \frac{1}{L} * v_S$$

$$\dot{v}_C = -\frac{1}{C} * \frac{1}{R_C + R_0} * v_C$$

$s = 1$

s_1

$s = 1$

$$\dot{i}_L = -\frac{1}{L}\left(R_L + \frac{R_C * R_0}{R_C + R_0}\right) * i_L - \frac{1}{L}\frac{R_0}{R_C + R_0} * v_C + \frac{1}{L} * v_S$$

$$\dot{v}_C = \frac{1}{C} * \frac{R_0}{R_C + R_0} * i_L - \frac{1}{C} * \frac{1}{R_C + R_0} * v_C$$

$s = 0$

float $i_L = i_0$, $v_c = v_0$, $d = d_0$; **bool** $s = 0$;

while true {

while $(s = 0)$ **{**

$$i_L = i_L - \left(\frac{R_L}{L} * i_L - \frac{1}{L} * v_S\right) * d$$

$$v_C = v_C - \frac{1}{C} * \frac{1}{R_c + R_0} * v_C * d$$

read(*s*) }

while $(s = 1)$ **{**

$$i_L = i_L - \left(\frac{1}{L}\left(R_L + \frac{R_C * R_0}{R_c + R_0}\right) * i_L + \frac{1}{L}\frac{R_0}{R_c + R_0} * v_C - \frac{1}{L} * v_S\right) * d$$

$$v_C = v_C + \left(\frac{1}{C} * \frac{R_0}{R_c + R_0} * i_L - \frac{1}{C} * \frac{1}{R_c + R_0} * v_C\right) * d$$

**read(*s*) } }

Execution of Boost DC-DC Converter

Parameters :

$U_s = 20V$

$L = 1mH$

$C = 50nF$

$R_L = 1kW$

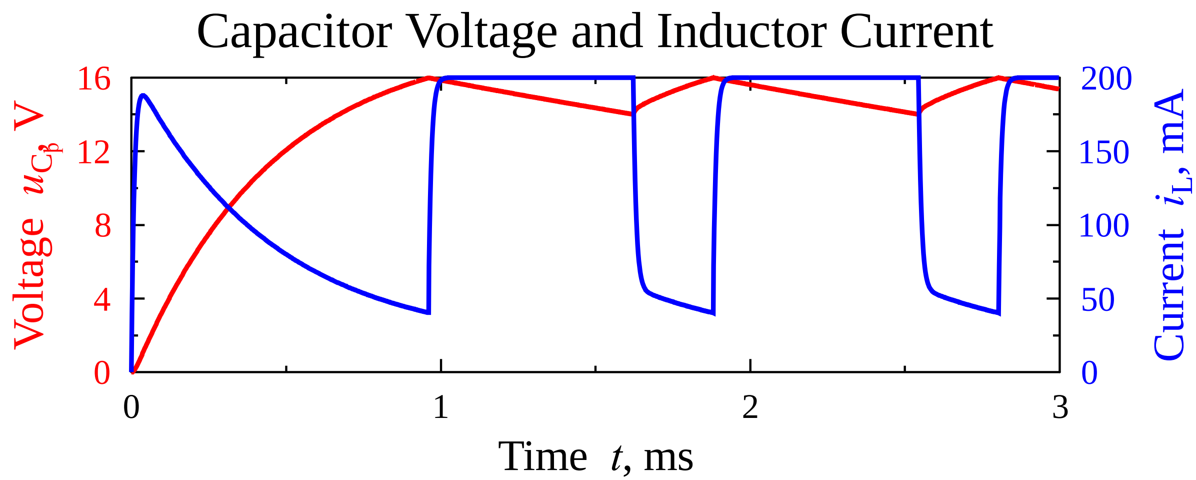
$R_C = 10W$

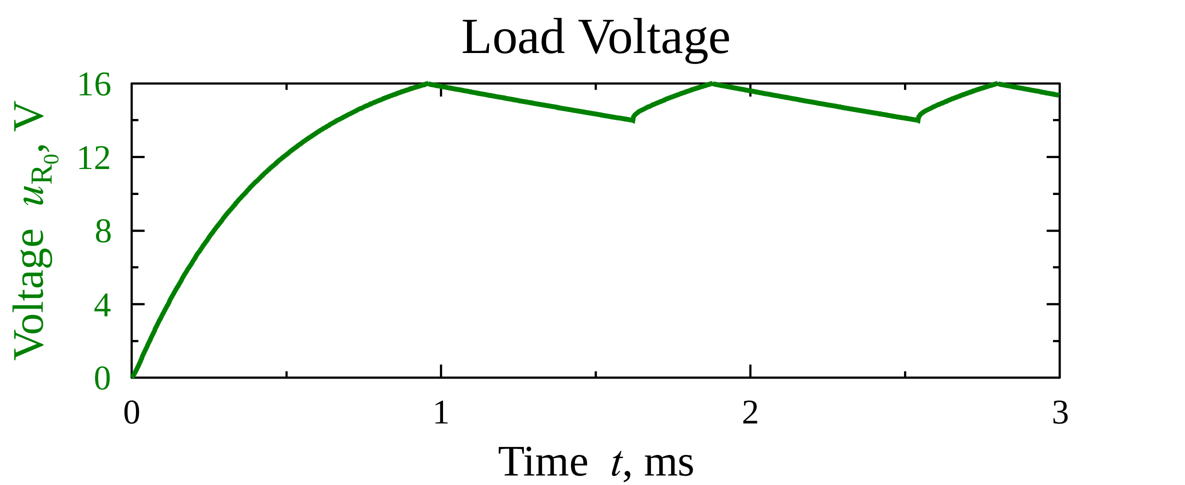
$R_0 = 10kW$

$dt = 200ns$

$U_{max} = 16V$

$U_{min} = 14V$





Hybrid Automaton $\mathbb{H}$

Variables: Continuous variables $x = [x_1, \ldots, x_n]$

Control Graph: Finite directed multigraph (V, E)

- Finite set V of control modes
- Finite set E of control switches

Vertex labeling functions: for each $v \in V$

- Initial states: init(v)(x) defines initial region
- Invariant: inv(v)(x) defines invariant region
- Continuous dynamics: $\dot{x}$ is in flow(v)(x)

Edge labeling functions: for each $e \in E$

- Guard: guard(e)(x) defines enabling region
- Update: action(e)(x, x') defines the reset region
- Synchronization labels: label(e) defines communication

Executions of a Hybrid Automaton

State: (m, x) such that $x \in \mathrm{inv}(m)$

Initialization: (m, x) such that $x \in \mathrm{init}(m)$

Two types of state updates:

Discrete switches: $(m, x) \to^{a} (m', x')$ if

$$e = (m, m') \in E \;\wedge\; \mathrm{label}(e) = a \;\wedge$$

$$\mathrm{guard}(e)(x) \geq 0 \;\wedge\; \mathrm{action}(e)(x, x')$$

Continuous flows: $(m, x) \to^{f} (m, x')$ if $\exists\, f : [0,T] \to \mathbb{R}^n$.

$$f(0) = x \;\wedge\; f(T) = x'$$

$$\forall 0 \leq t \leq T.\; f(t) \in \mathrm{inv}(m) \;\wedge\; \dot{f}(t) \in \mathrm{flow}(m)(f(t))$$

References

- T. A. Henzinger, **"The theory of hybrid automata“**, Logic in Computer Science, 1996. LICS '96. Proceedings., Eleventh Annual IEEE Symposium on, New Brunswick, NJ, 1996, pp. 278-292.