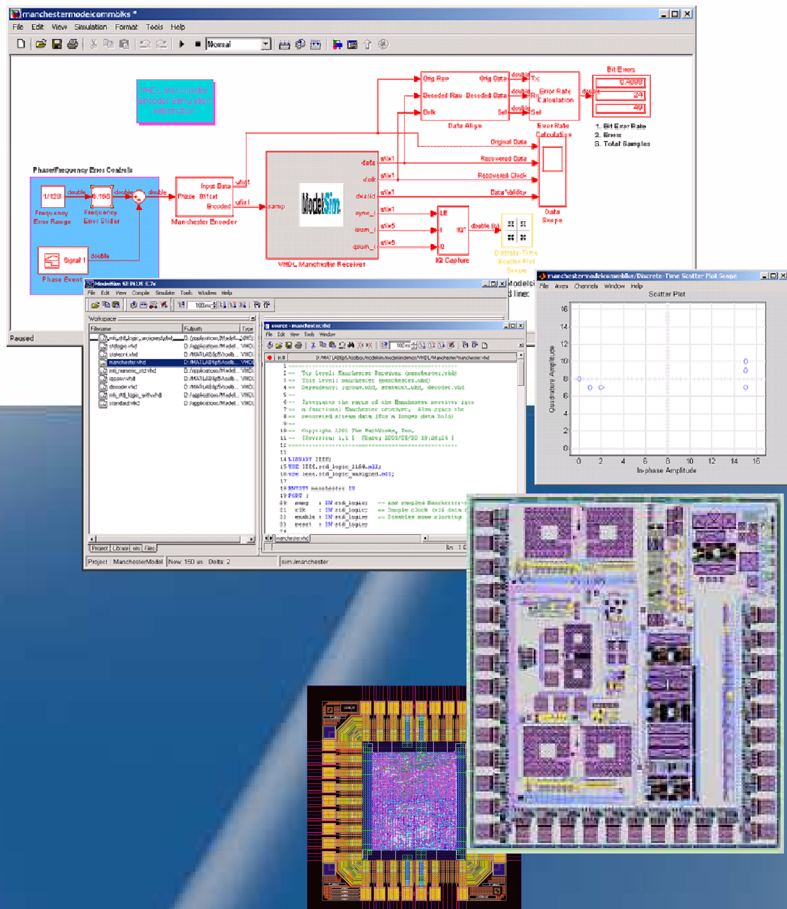
Issues in Advanced Mixed-Signal Simulation



Pieter J. Mosterman

Senior Research Scientist
The MathWorks, Inc.

Agenda

- Introduction (~5 min.)
- Demo (~25 min.)
- What is Link for ModelSim®? (~5 min.)
- Mixed signal simulation (~15 min.)
- Classes of behaviors (~25 min.)
- Summary (~5 min.)

What is the problem?

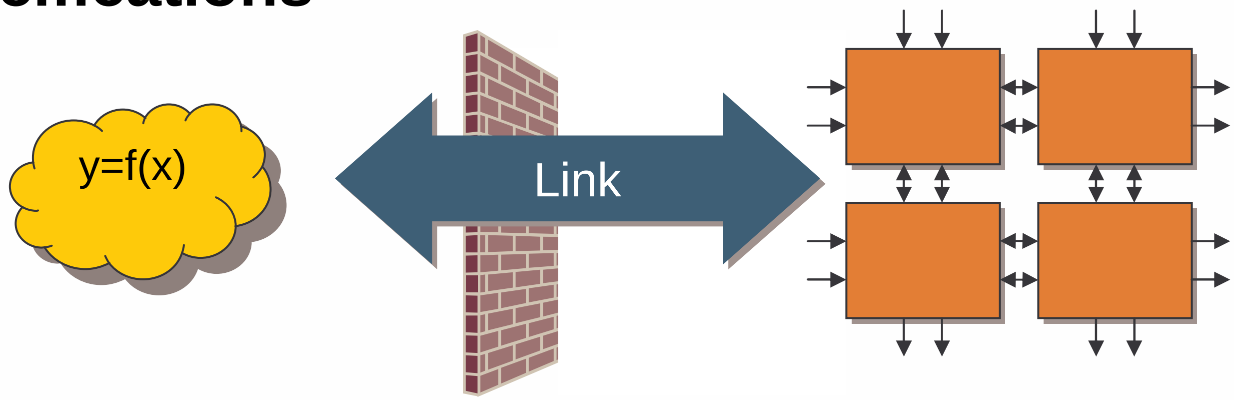
- "Hardware verification is itself becoming more challenging. Verification times have increased with rising gate count and as overall design complexity grows. According to a survey by Collett International Research in 2002 only 39% of designs were bug free at first silicon, while 60% contained logic or functional flaws. More than 20% required 3 or more silicon spins. A Collett survey also showed that nearly 50% of total engineering time was spent in verification."
- Hardware/Software Co-verification by Dr. Jack Horgan
  http://www.edacafe.com/magazine/index.php?newsletter=1&run_date=29-Mar-2004

What are the pains?

1. Time and effort to verify a design: As designs get more complex, the test benches are an order of magnitude more complex, and consume 40-60% of project resources.
   - Test bench HDL code will *not* be synthesized – i.e., will not be a part of the shipping product – "throw-away" code
   - HDL test benches need to run in HDL simulators, and HDL simulators are *extremely* slow
2. Time and effort to construct and maintain test benches: For each line of HDL design code in a design, a user typically needs 10 lines of HDL test bench code to simulate, test, and verify that 1 line of HDL code.
   - Constructing a test bench in a textual language is at least as complex as the original design itself
   - Maintaining the test bench from one generation of a design to the next is very resource-intensive

What are the pains? Solutions?

- **Engineers need to verify that ASIC/FPGA implementations correctly match their system specifications**



- **Using the Link for ModelSim®, these engineers can co-simulate their MATLAB® and Simulink® designs with equivalent Verilog and VHDL**

Agenda

- Introduction
- **Demo**
- What is Link for ModelSim®?
- Mixed Signal Simulation
- Classes of behaviors
- Summary

Demo

Live MATLAB Demo

- **Edge detection in lane departure detection demo**
  - **Derive unit requirements from system specification**
  - **Design space exploration using floating point**
  - **Conversion to fixed point**
  - **Co-simulate with HDL**
  - **Verify system-level behavior**
  - **Perform system integration**

Agenda

- Introduction
- Demo
- **What is Link for ModelSim®?**
- Mixed Signal Simulation
- Classes of behaviors
- Summary

Link for ModelSim® allows engineers to share models instead of I/O vectors.

- The HDL is verified in the context of an entire system and not just as a stand-alone component

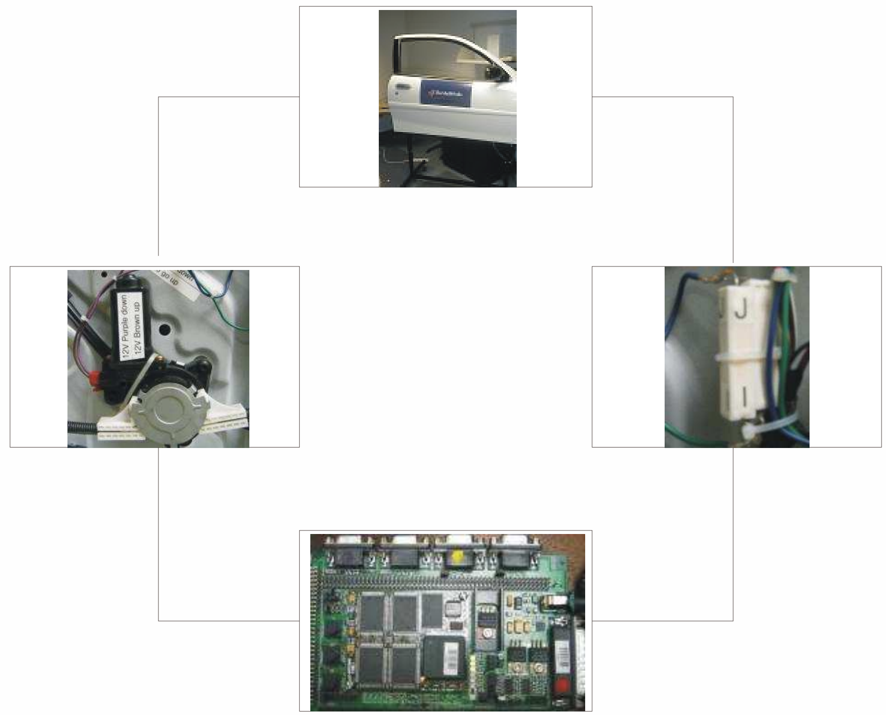
- System performance metrics, e.g. PER, BER, S/N ratio can be measured

Agenda

- Introduction
- Demo
- What is Link for ModelSim®?
- **Mixed Signal Simulation**
- Classes of behaviors
- Summary

Hybrid dynamic systems

- Two types of behavior
  - Continuous
  - Discrete
- An embedded controller
  - Plant
    - Continuous-time behavior
    - Sporadic discrete events
  - Controller
    - Discrete-time behavior
    - Frequent periodic events



Executing a hybrid dynamic system

- **Integrate continuous behavior**

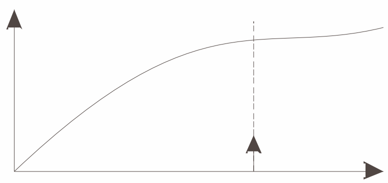
$$\dot{x} = f(x,u,t)$$

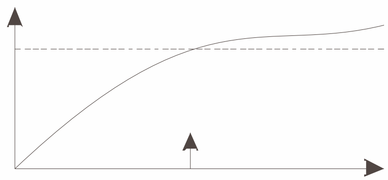
- **Discrete event behavior**
  - **Time events**
    - **Pre-determined time of occurrence**

$$t_n = nextEventTime()$$

  - **State events**
    - **When a model variable exceeds a threshold**

$$g(x,u,t) \geq 0$$

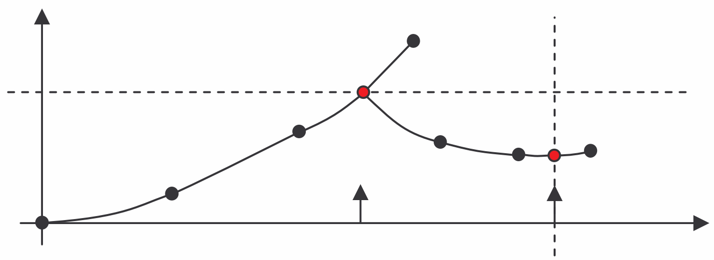


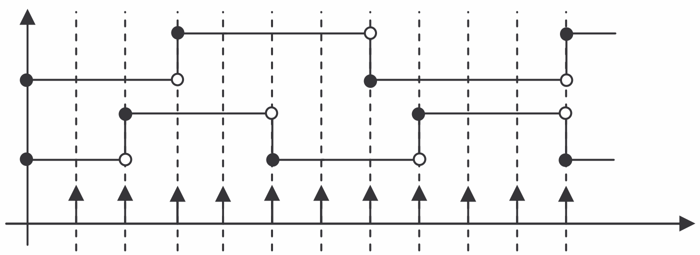


The MathWorks MATLAB&SIMULINK

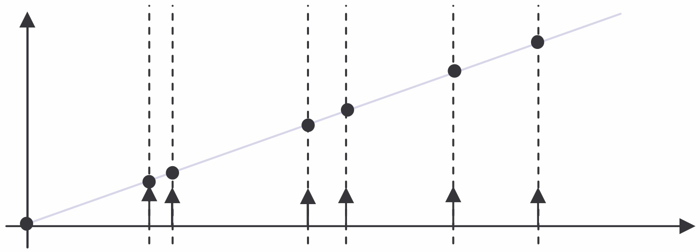
How to handle a discrete event

- **Time-driven**
  - **Time integrated**
    - **Integrate up till event time**
    - **Inefficient for time events**
  - **Sampled time**
    - **Run scheduler at lowest rate**
    - **Inefficient for widely spaced events**



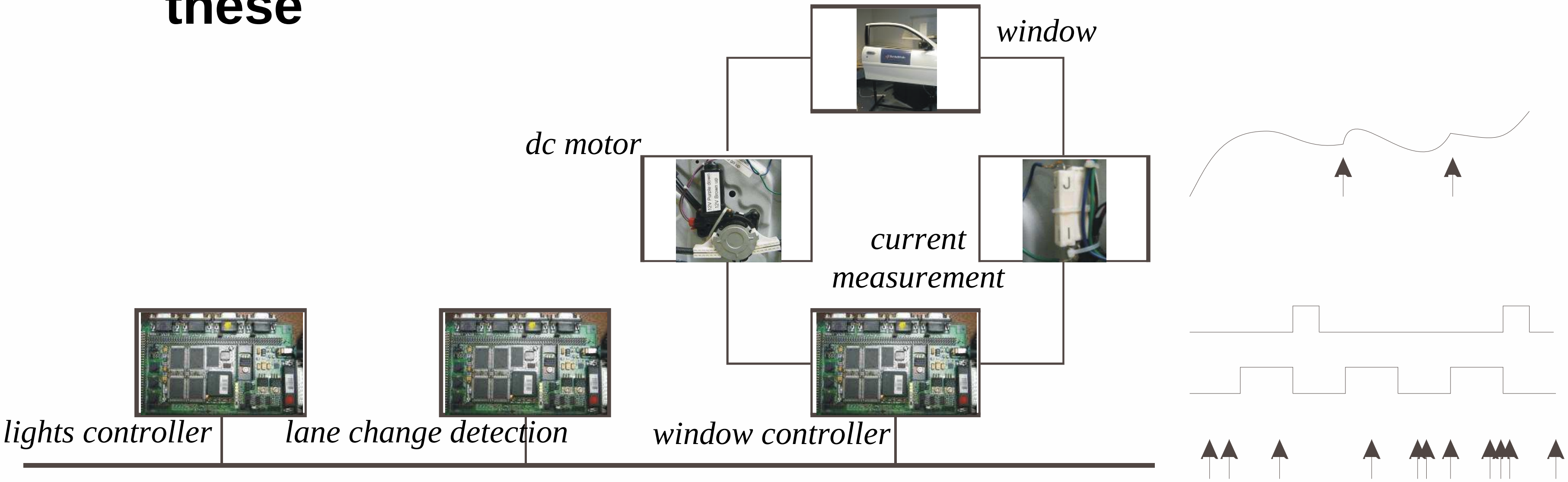


- **Event-driven**
  - **Jump to event time immediately**
  - **Does not apply to state events**



Where do these paradigms apply?

- **Discrete event intensive models**
  - **Controller area network (CAN) in automobiles**
- **Many events that do not affect continuous behavior**
- **Expensive to stop the numerical solver for each of these**



Efficiency

- **Two separate solvers**
  - **Time-driven solver**
    - **Numerical solver integrates time (step *h*)**

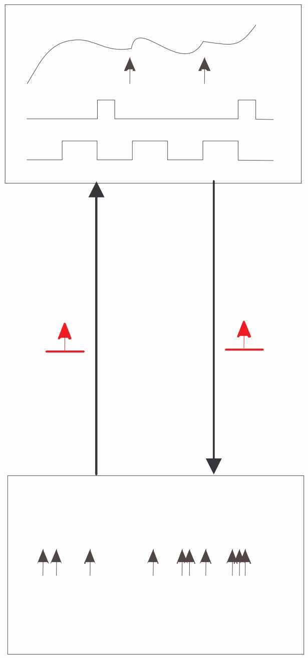
$$k_1 = hf(x_n, y_n) \qquad k_3 = hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$

$$k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \qquad k_4 = hf(x_n + h, y_n + k_3)$$

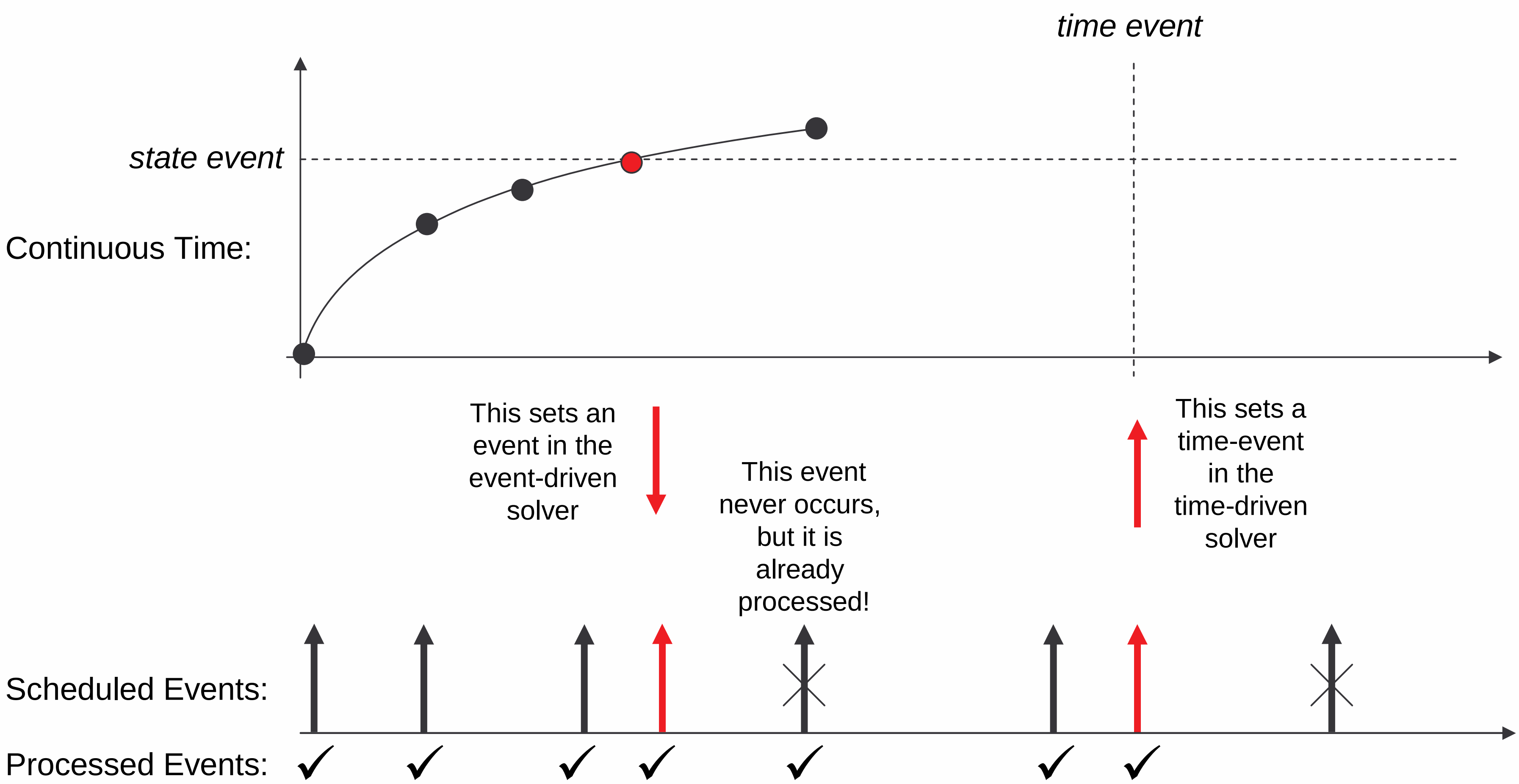
$$y_{n+1} = y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5)$$

    - **Sampled time event schedule (time up to which to integrate)**
  - **Event-driven solver**
    - **Event calendar**

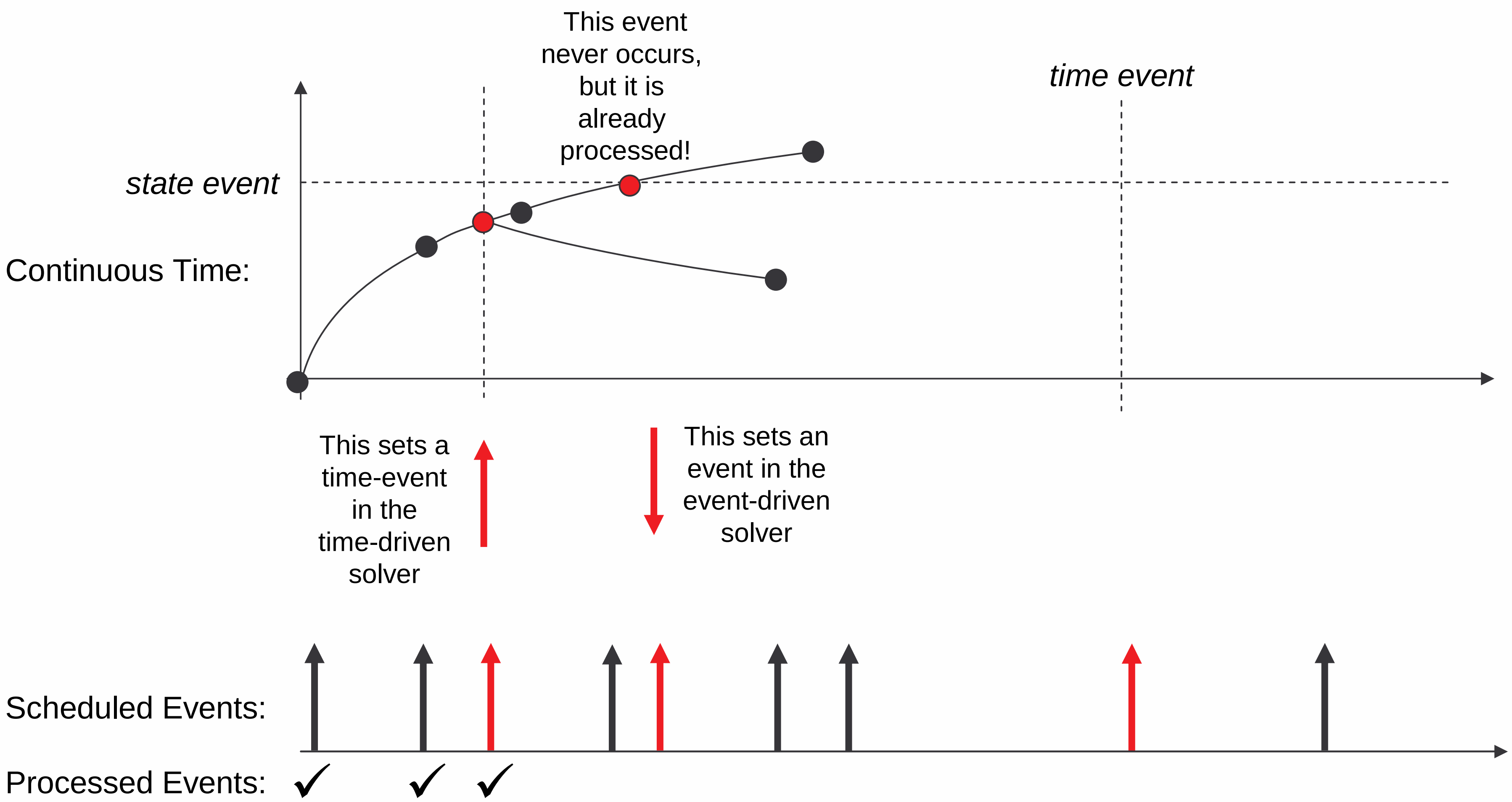
20 [ms]	open_tonneau
2020 [ms]	move_roof_up_cmd
2150 [ms]	move_down_window
2250 [ms]	stop_moving_window



Event-driven leads



Time-driven leads



Agenda

- Introduction
- Demo
- What is Link for ModelSim®?
- Mixed Signal Simulation
- **Classes of behaviors**
- Summary

Hybrid behavior

- Introduce ideal diodes
  - Make highly nonlinear behavior piecewise linear
    - freewheeling diode

$$if \quad s_{diode} \quad then \;\; v_{diode} = \sum v_i \quad else \;\; i_{diode} = 0$$

- Switching between modes of continuous behavior
  - Diode, s_{diode}, autonomous switch triggered by physical quantities

$$s_{diode} = \sum v_i < 0$$

  - Different sets of equations

Computational causality

- **When switching equations**
  - **Computational causality may change**
- **Example**
  - **When the diode closes, equations change**
    - **From**

$$v_{diode} = \sum v_i$$

    - **To**

$$i_{diode} = 0$$

  - **Therefore, in this equation**
    - v_{diode} **becomes unknown**
    - i_{diode} **becomes known**

Implicit modeling

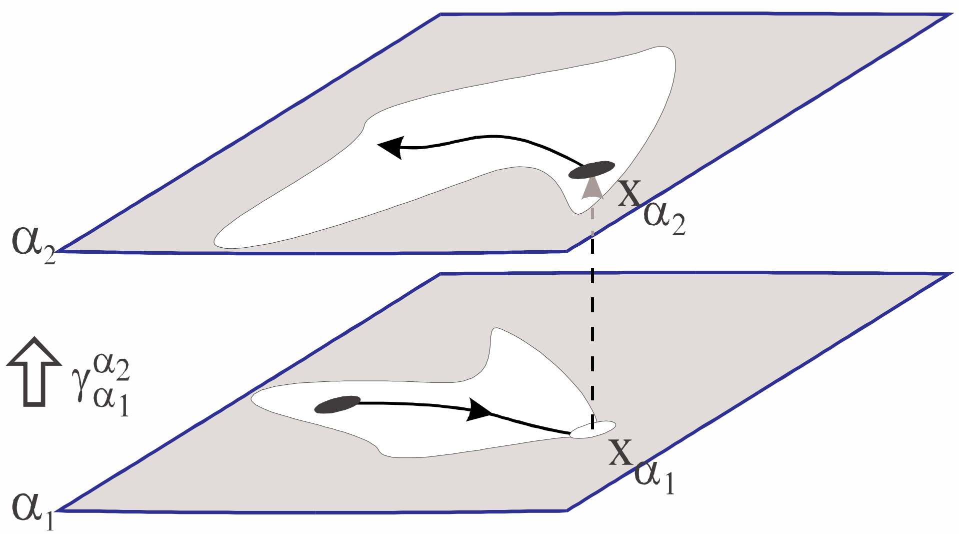
- Deal with causal changes cumerically
- Diode behavior
  - Residue on i_{diode}

$$0 = if \quad s_{diode} \quad then \sum v_i \quad else \quad i_{diode}$$

- Implicit numerical solver (e.g., DASSL)
  - Designed to handle this formulation

Hybrid dynamic behavior

- **Geometric View**
  - **Modes of continuous, smooth, behavior**
  - **Patches of admissible state variable values**



Specification parts

- **Hybrid behavior specification**
  - **A function, f, that defines continuous, smooth, behavior for each mode**

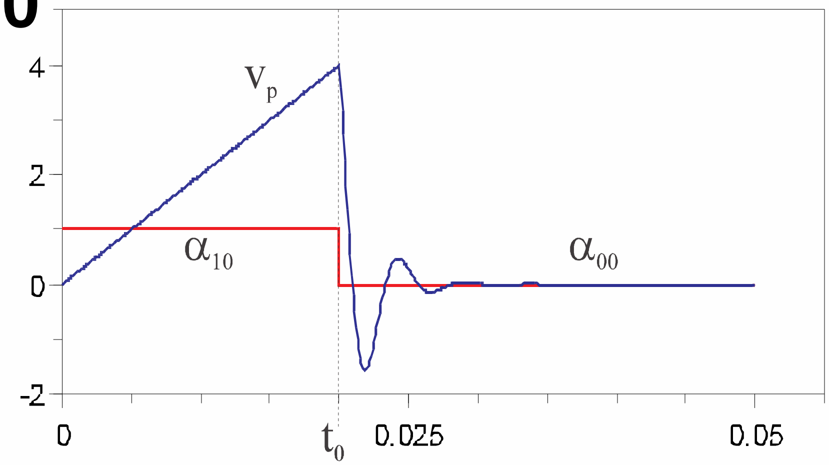
$$f_{\alpha_i}: E_{\alpha_i}\dot{x} + A_{\alpha_i}x + B_{\alpha_i}u = 0$$

  - **An inequality, γ, that defines admissible state variable values**

$$\gamma_{\alpha_i}^{\alpha_{i+1}}: C_{\alpha_i}x + D_{\alpha_i}u \geq 0$$

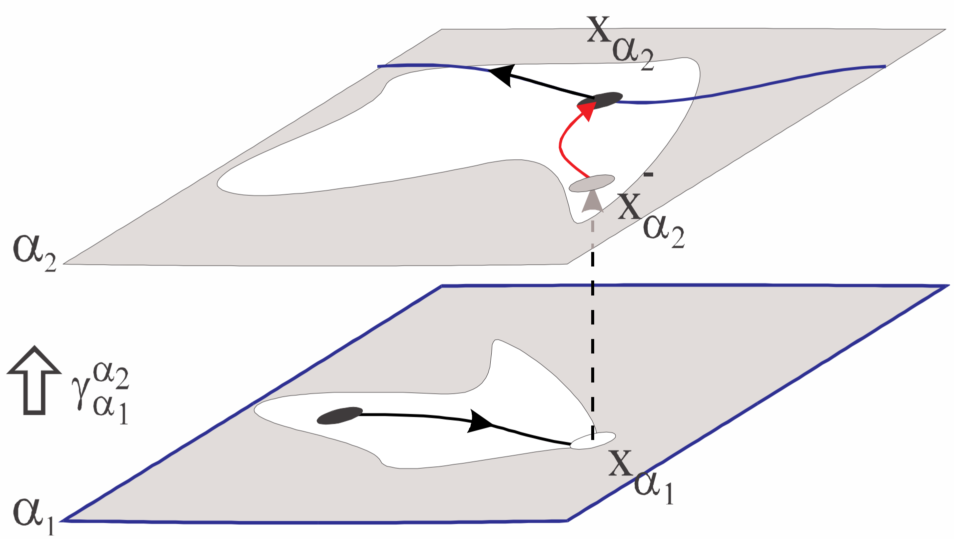
Dynamics

- **RLC behavior characteristics**
  - C^0**, i.e., no jumps in state variables**
  - **Steep gradients**
- **Example**
  - **When a switch opens, current quickly reduces to 0**



Hybrid dynamic behavior - refined

- **Geometric View**
  - **Modes of continuous, smooth, behavior**
  - **Patches of admissible state variable values**
  - **Manifold of dynamic behavior**



Specification parts

- **Hybrid behavior specification**
  - **A function, f, that implicitly defines for each mode**
    - **continuous, smooth, behavior**
    - **state variable value jumps**

$$f_{\alpha_i}: E_{\alpha_i}\dot{x} + A_{\alpha_i}x + B_{\alpha_i}u = 0$$

  - **An inequality, γ, that defines admissible generalized state variable values**

$$\gamma_{\alpha_i}^{\alpha_{i+1}}: C_{\alpha_i}x + D_{\alpha_i}u \geq 0$$

  - **For explicit reinitialization (semantics of x^-)**

$$f_{\alpha_i}: E_{\alpha_i}\dot{x} + A_{\alpha_i}x + B_{\alpha_i}^{u}u + B_{\alpha_i}^{x}x^- = 0$$

Projections

- **Linear time invariant index 2 system**
  - **Derive pseudo Kronecker Normal Form (numerically stable)**

$$\left[\begin{array}{c:cc} E_{11} & 0 & 0 \\ \hdashline 0 & 0 & E_{22,12} \\ 0 & 0 & 0 \end{array}\right]\begin{bmatrix} \dot{x}_f \\ \dot{x}_{i,1} \\ \dot{x}_{i,2} \end{bmatrix} + \left[\begin{array}{c:cc} A_{11} & A_{12,1} & A_{12,2} \\ \hdashline 0 & A_{22,11} & A_{22,12} \\ 0 & 0 & A_{22,22} \end{array}\right]\begin{bmatrix} x_f \\ x_{i,1} \\ x_{i,2} \end{bmatrix} + \left[\begin{array}{c} B_1 \\ \hdashline B_{2,1} \\ B_{2,2} \end{array}\right] u = 0$$

  - **After integration (no impulsive input behavior), consistent values are**

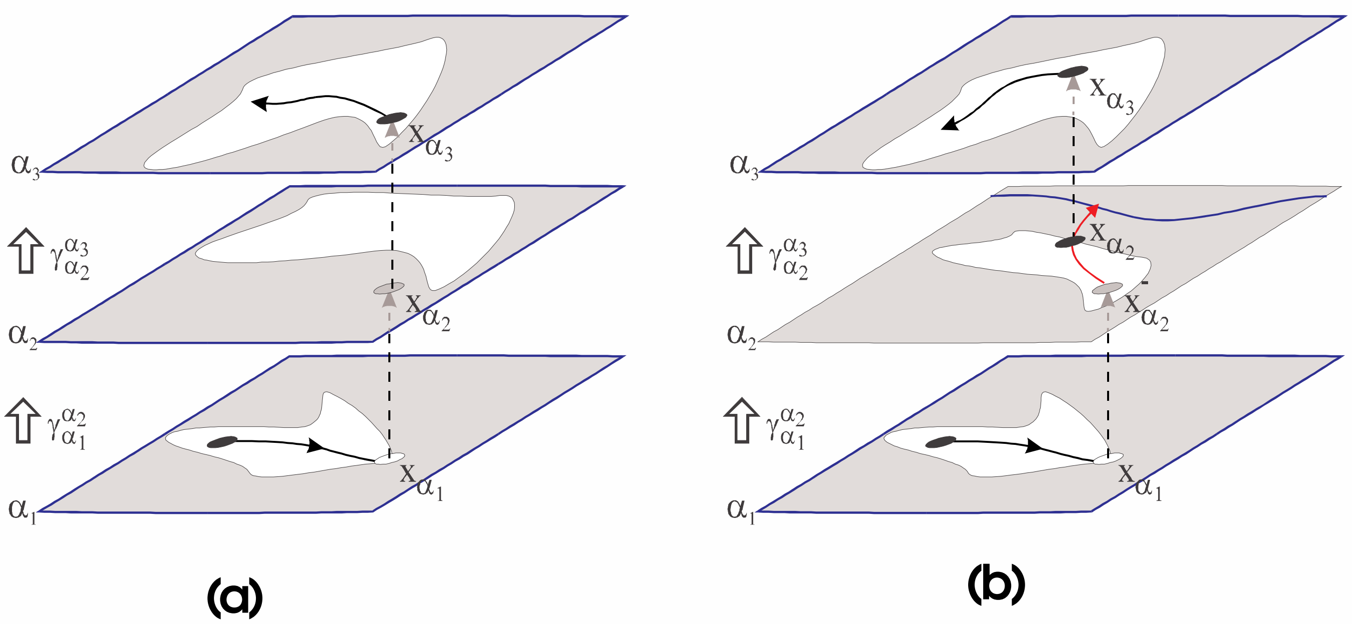
$$x_f = x_f^- - E_{11}^{-1} A_{12,1} A_{22,11}^{-1} E_{22,12} (x_{i,2} - x_{i,2}^-)$$

$$x_{i,1} = A_{22,11}^{-1}(-B_{2,1} u + E_{22,12}\dot{x}_{i,2}) - A_{22,12} x_{i,2}$$

$$x_{i,2} = -A_{22,22}^{-1} B_{2,2} u$$

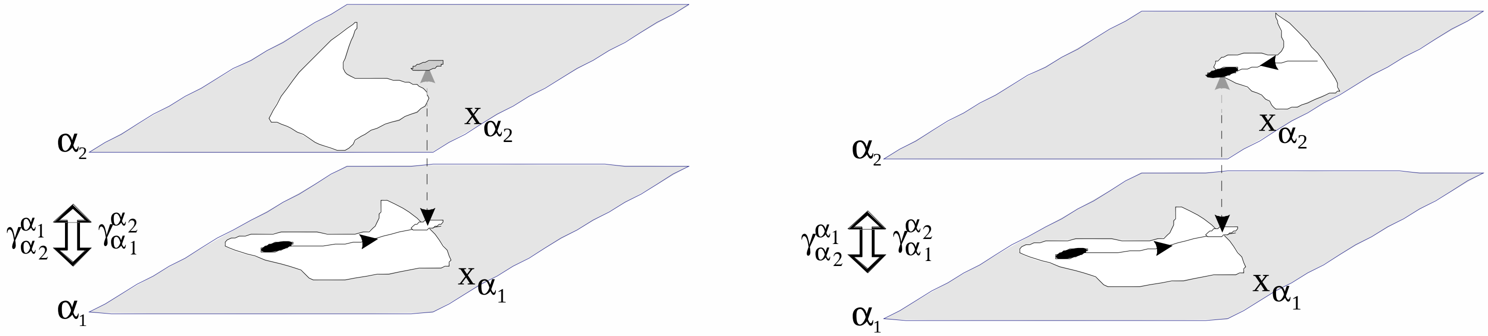
Sequences of mode changes

a) State outside of a patch in the new mode
b) During projection state values are reached outside of a patch in the new mode



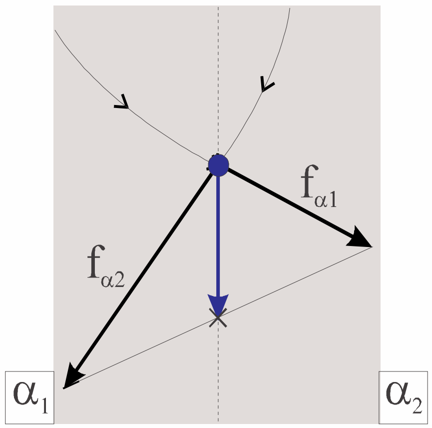
Chattering

- What if the new mode switches back
  - Immediately ⇒ inconsistent model, no solution
  - After infinitesimal period of time ⇒ chattering behavior, solve with
    - equivalent control
    - equivalent dynamics



Equivalent dynamics

- Chattering
  - Fast component
    - remove
  - Slow component
    - weighted mean of instantaneous vector fields (Filippov Construction)
  - Sliding behavior



Agenda

- Introduction
- What is Link for ModelSim®?
- Demo
- Mixed Signal Simulation
- **Summary**

Summary of key features

- Integrate system level design with implementation
- No duplication of testbench design effort
- Verification of system level properties

- Advanced simulation technologies are required

Summary of behavior classes

- **Phase space transition behavior classification**
  - **Mythical (state invariant)**
  - **Pinnacle (state projection aborted)**
  - **Continuous**
    - **interior (continuous behavior)**
    - **boundary (further transition after infinitesimal time advance)**
    - **sliding (repeated transitions after each infinitesimal time advance)**
- **Combinations of behavior classes**

The figures on slide 22, 25, and 28 have been previously published on page 626, and the figure on slide 29 on page 627 of the *Proceedings of the 2003 Winter Simulation Conference* (S. Chick, P.J. Sanchez, D. Ferrin, and D.J. Morrice, eds.) in a paper entitled “Mode Transition Behavior in Hybrid Dynamic Systems”.