Circuit Level Verification of a High-Speed Toggle

Chao Yan & Mark Greenstreet

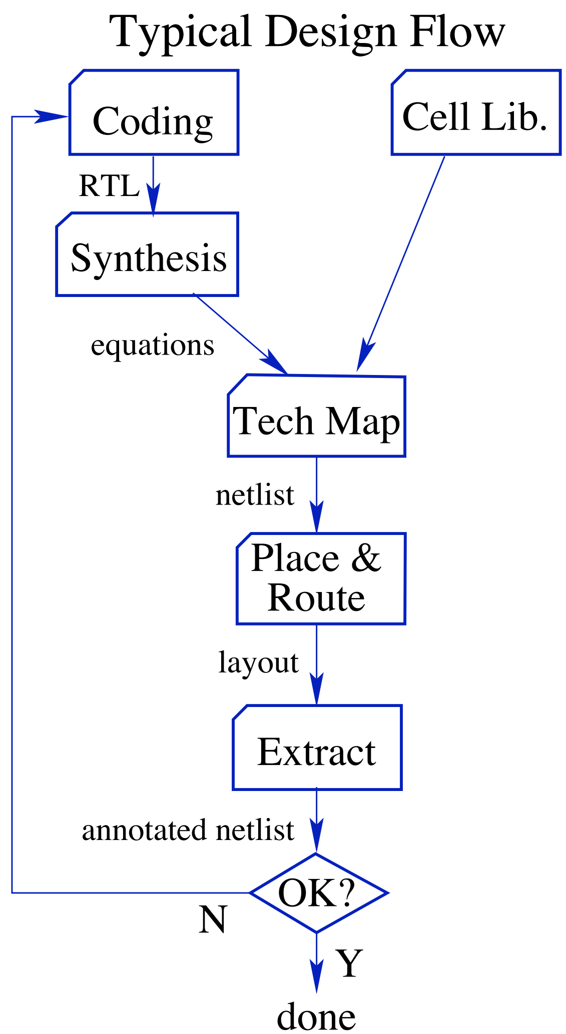
University of British Columbia

Overview

- Motivation
- Coho
  - Projection Based Reachability Analysis
  - Numerical Issues
- Verification Example
  - Toggle Circuit
  - Toggle Specification
  - Verification Using Coho
- ***Formal Verification of Digital Circuits Using SPICE-Level Models is Possible.***

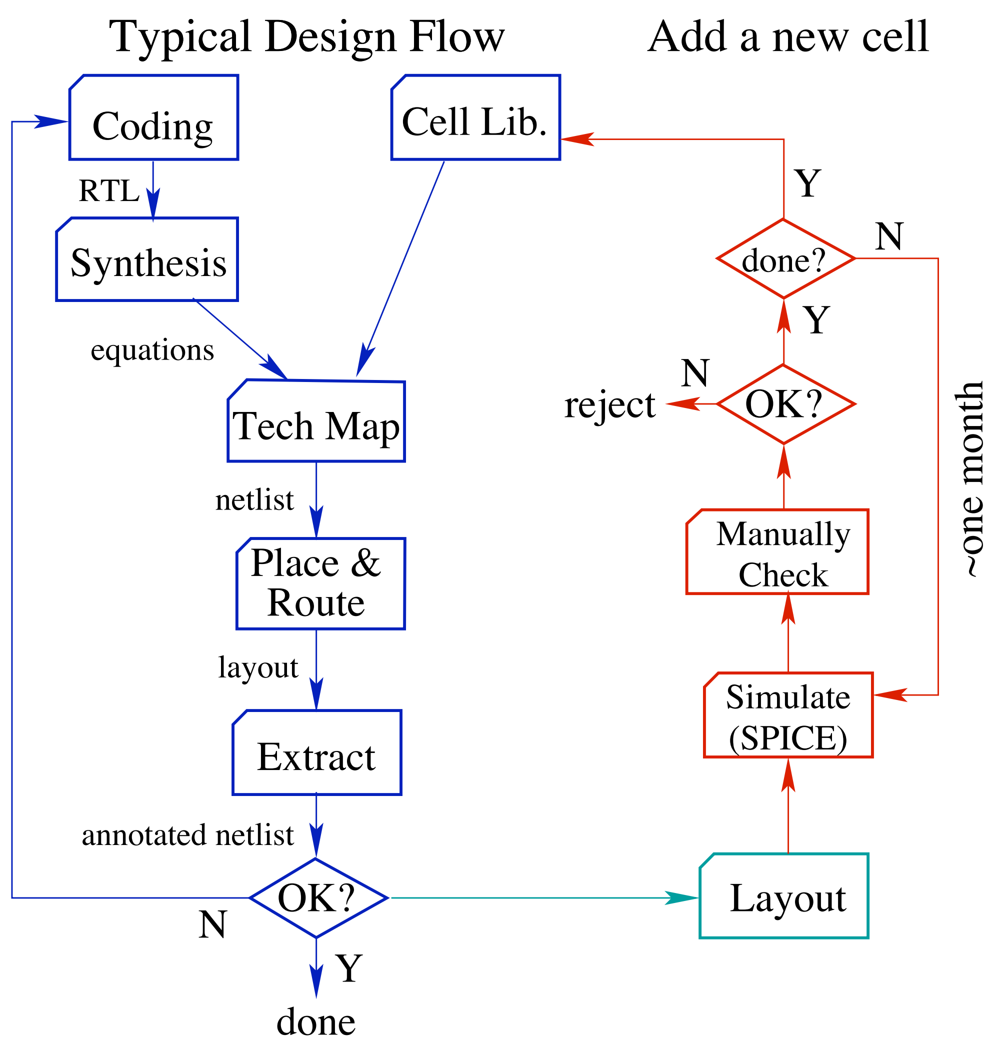
Motivation

- Design Flow



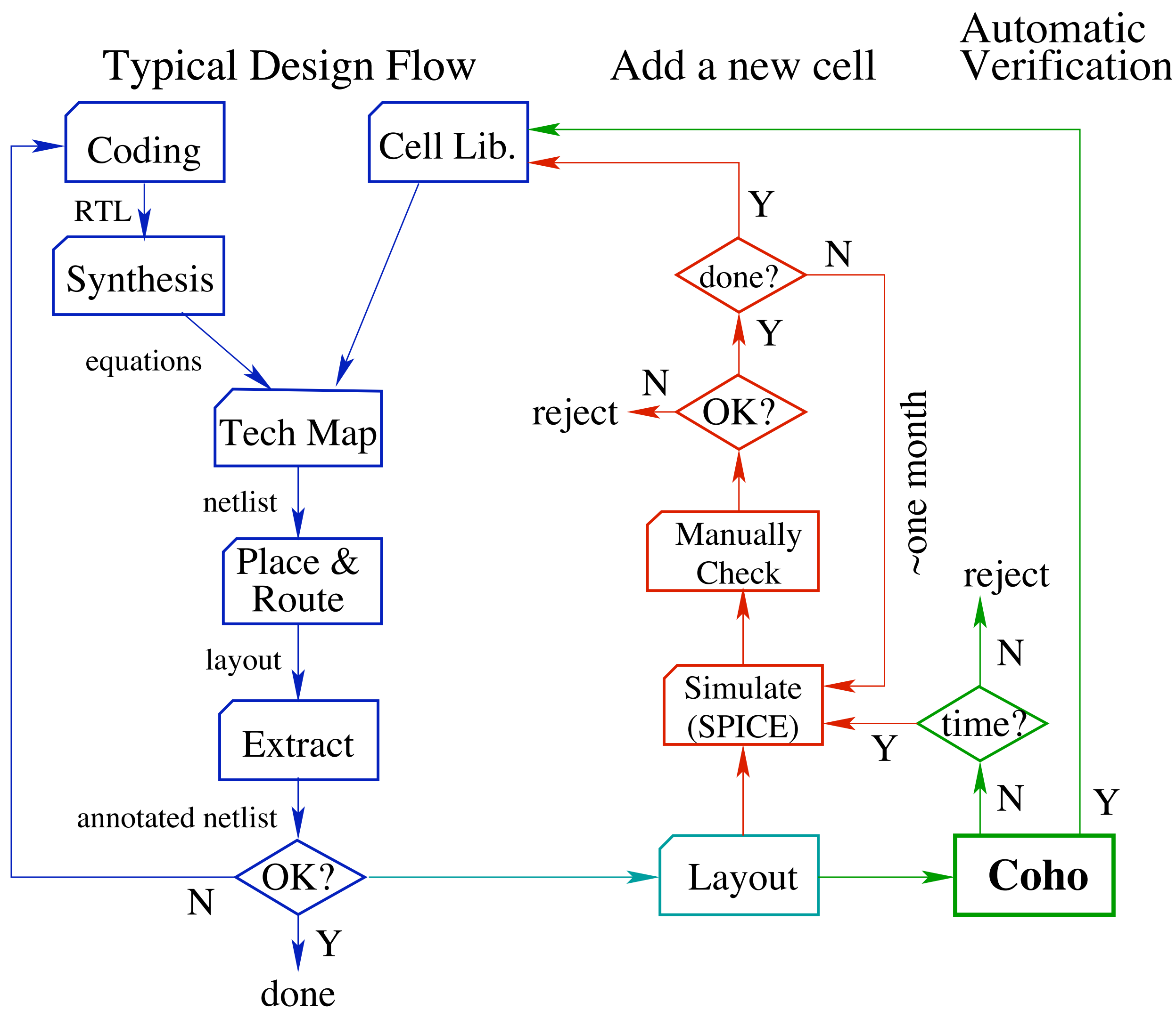
Motivation

- Design Flow



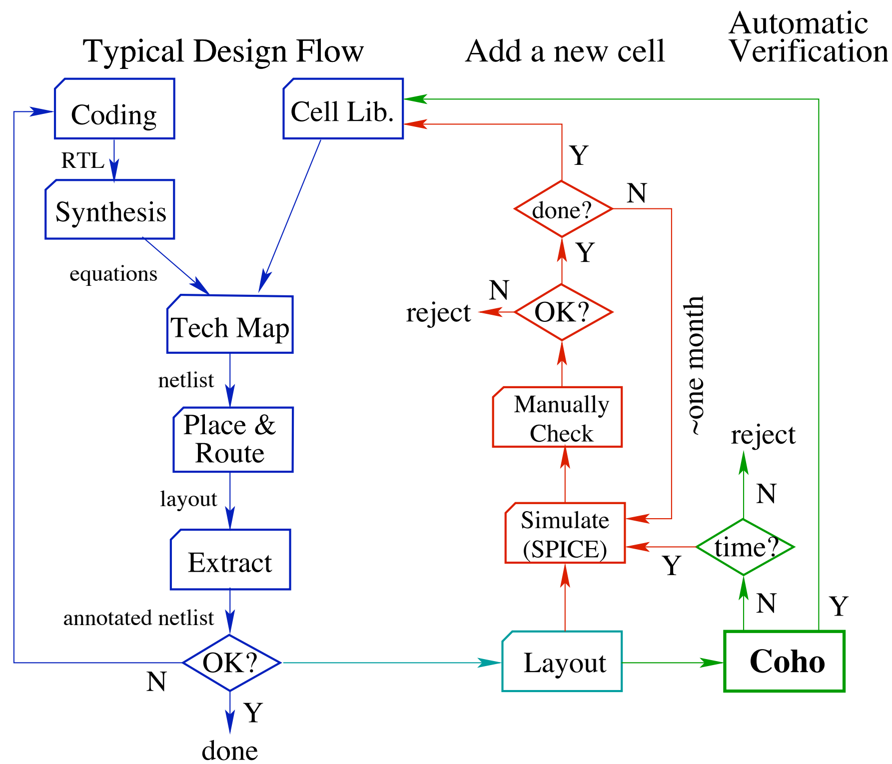
Motivation

- Design Flow



Motivation

- Design Flow
- Similar Problems
  - crosstalk analysis
  - power noise problems
  - leaky transistors
  - mixed-signal design



Coho

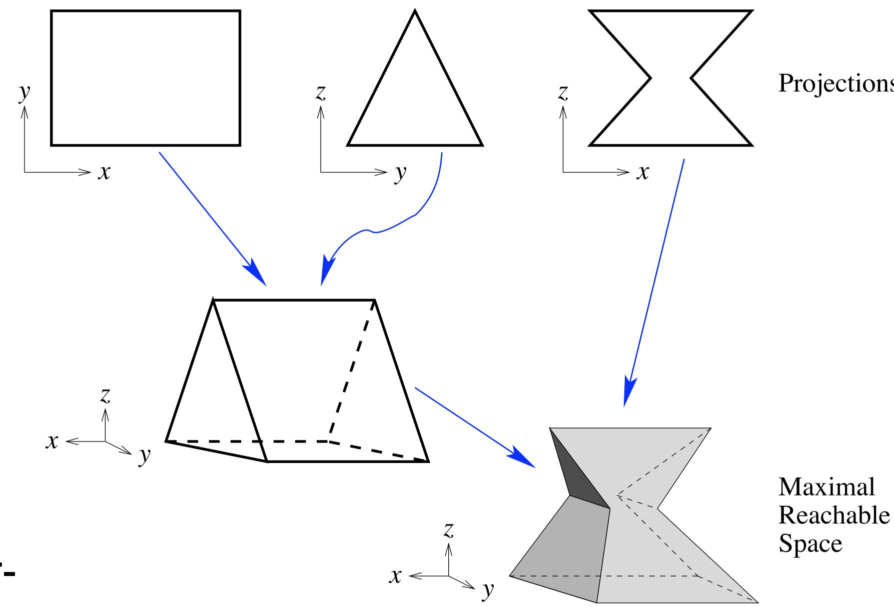
- Reachability method for verifying real circuits
- Approximate the non-linear ordinary differential equations (ODEs) in small neighborhoods by linear differential inclusions:

$$Ax + b - u \quad \leq \quad \dot{x} \quad \leq \quad Ax + b + u$$

- Projection based representation of reachable space

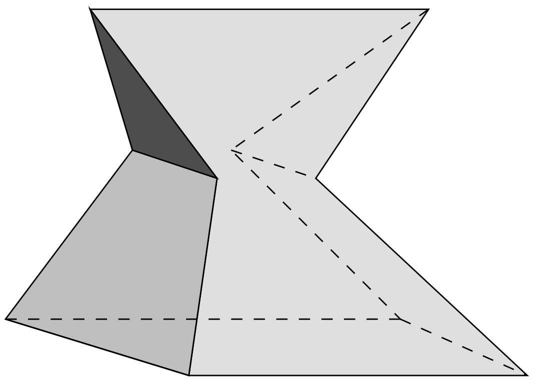
Representing the Reachable Space

- Coho: Projectagons
  - Project high dimensional polyhedron onto two-dimensional subspaces.
  - A point is in the projectagon iff its projections are contained in the corresponding polygons.
  - Projectagons are efficiently manipulated using two-dimensional geometry computation algorithms.
  - Each edge of the polygon corresponds to a face of the high-dimensional polyhedron.



Representing the Reachable Space

- Coho: Projectagons
  - Project high dimensional polyhedron onto two-dimensional subspaces.
  - A point is in the projectagon iff its projections are contained in the corresponding polygons.
  - Projectagons are efficiently manipulated using two-dimensional geometry computation algorithms.
  - Each edge of the polygon corresponds to a face of the high-dimensional polyhedron.

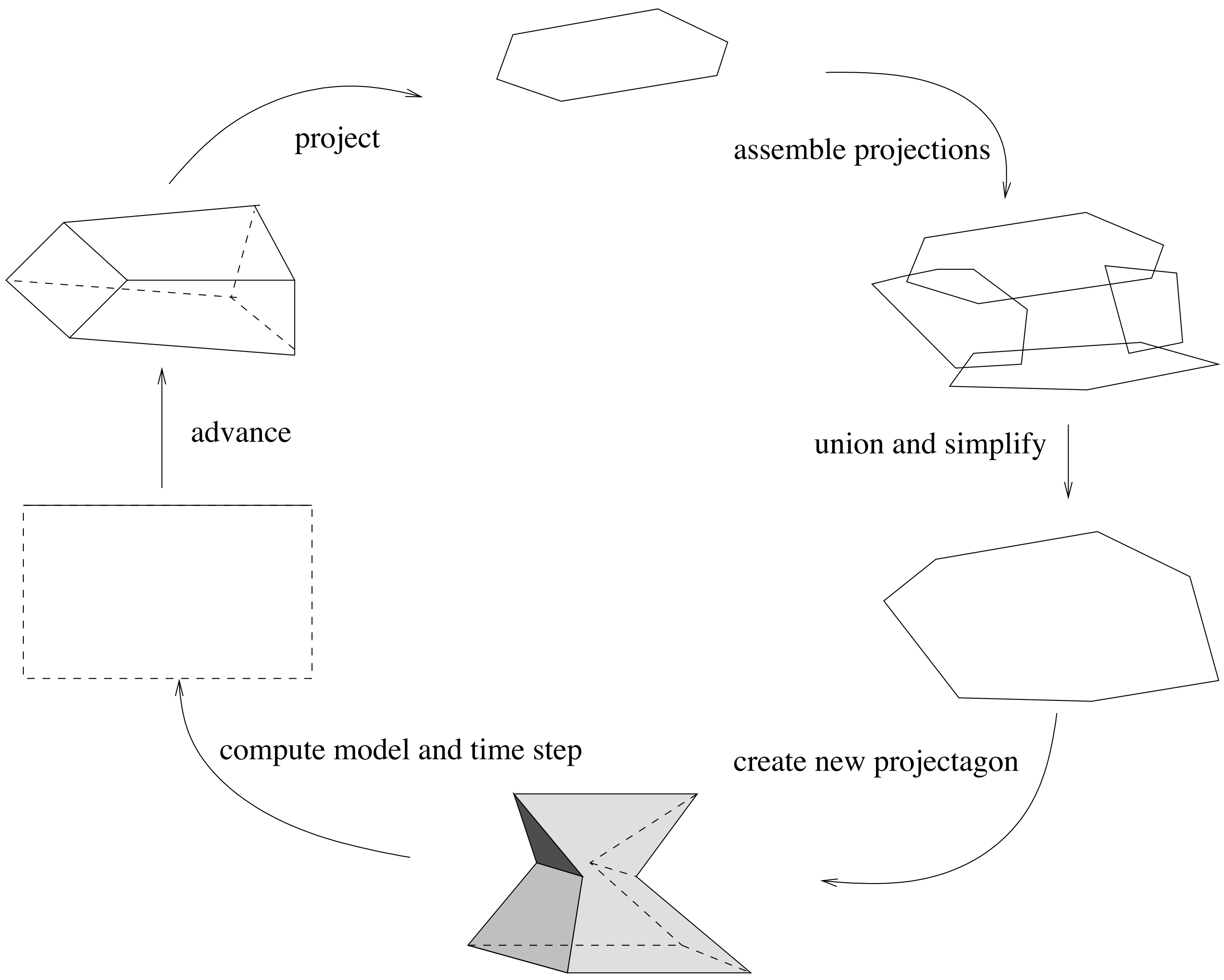


- Other approaches:
  - symbolic hyper-rectangles (HyTech)
  - convex polyhedra (CheckMate)
  - orthogonal polyhedra (d/dt)

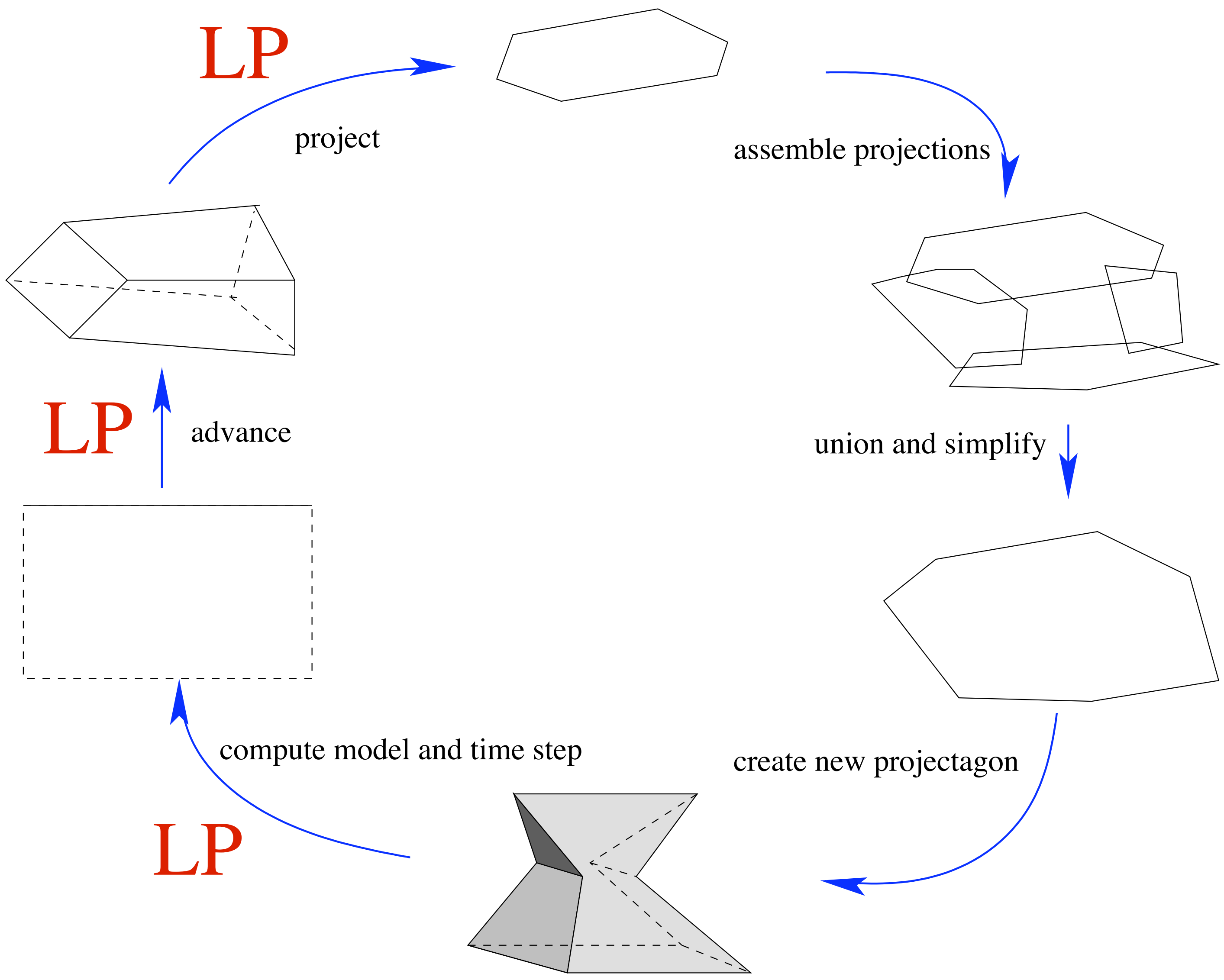
Reachability for Projectagons

- Extremal trajectories original from projectagon faces.
- Projectagon faces correspond to projection polygon edges.
- Coho computes time-advanced projectagons by working on one edge at a time.

Basic Step of Coho



Basic Step of Coho



Coho Linear Program Solver

- Coho Linear Program

$$\begin{array}{rll} & \min & c^T x \\ s.t. & Ax & \leq b \end{array}$$

$$A^T_{block} = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ & \alpha_2 & \beta_2 & & \mathbf{0} \\ & \mathbf{0} & \ddots & & \\ & & & \ddots & \\ \beta_2 & & & & \alpha_2 \end{bmatrix}$$

- Each inequality constraint corresponds to a face of the projectagon.
- One or two non-zero elements on each row of A.
- Dual is a standard form LP: $A^T u = c$.
- Efficient linear system solver in $O(n)$ time.

Coho Linear Program Solver

- Coho Linear Program

$$A^T_{block} = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ & \alpha_2 & \beta_2 & 0 \\ & 0 & \ddots & \\ \beta_2 & & & \alpha_2 \end{bmatrix}$$

  - Each inequality constraint corresponds to a face of the projectagon.
  - One or two non-zero elements on each row of A.
  - Dual is a standard form LP: $A^T u = c$.
  - Efficient linear system solver in $O(n)$ time.
- Simplex-based linear program solver:
  - Reduce accumulated error by computing tableau matrix directly from input data.
  - Use Interval Arithmetic for well-conditioned problems.
  - Use Arbitrary Precision Rational Computation for ill-conditioned problems.

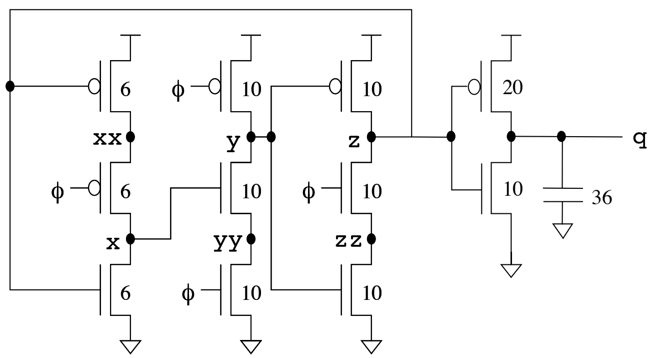
Summary of Coho

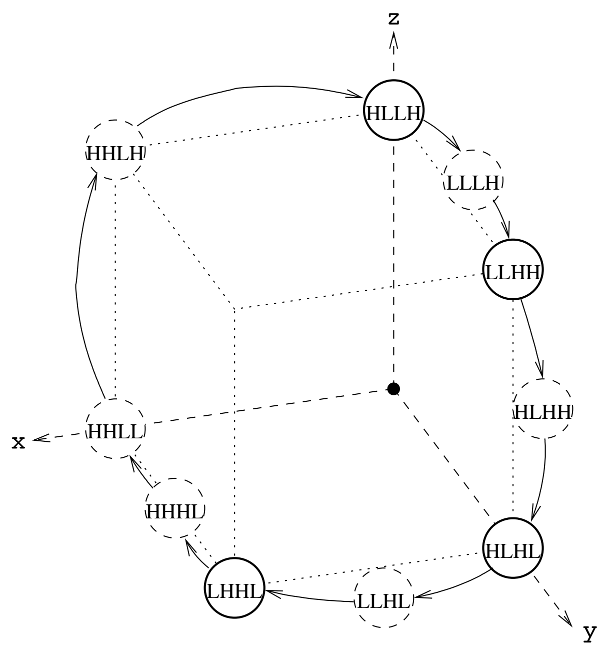
- Summary of Basic Algorithm:
  - The ODE model of circuit is approximated by linear differential inclusions.
  - Use projectagons to represent reachable set.
  - Coho is sound: all approximations overestimate the reachable space.
  - Extensive use of linear programming.
- Numerical Problems:
  - Ill-conditioned linear programs
    - Exploit LP structure of Coho's LPs.
    - Use hybrid approach of interval and arbitrary-precsion arithmetic.
  - Polygon intersection/union difficult with nearly-parallel edges.
    - Use hybrid approach of interval and arbitrary-precsion arithmetic already implemented for LPs.

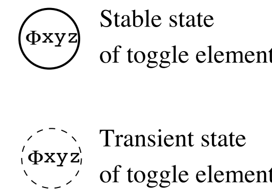
Circuit Verification

- Circuit description
  - Use MSPICE – a Matlab package that provides simple spice-like functionality.
  - Allows us to use same model for simulation and verification.
  - Simulate first:
    - Do not attempt to verify a incorrect system.
    - Have a rough idea of the reachable space to guide the verification.
    - Helps explain verification failures.
- Compute reachable set by Coho
- Verify design specification using reachable set

Toggle Circuit



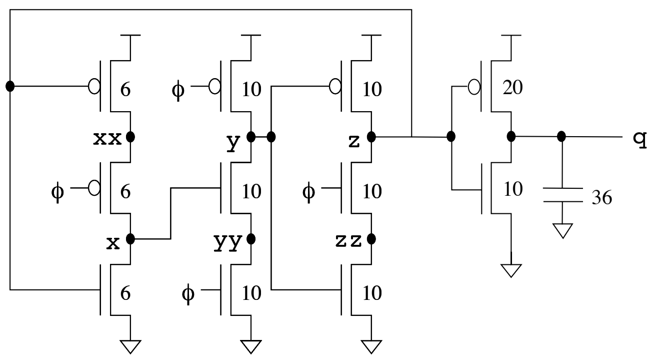


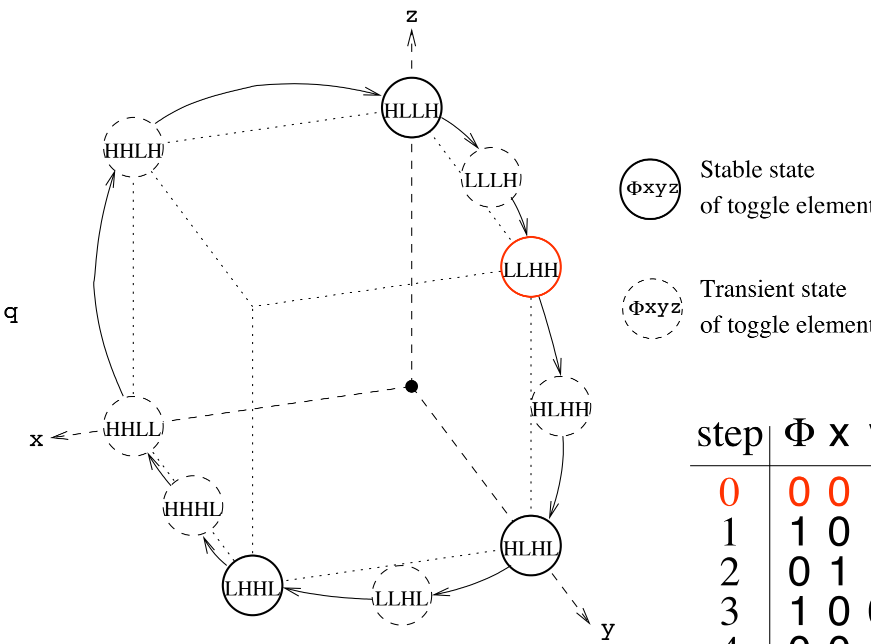


step	Φ	x	y	z
0	0	0	1	1
1	1	0	1	0
2	0	1	1	0
3	1	0	0	1
4	0	0	1	1

- Start from the state where Φ is low, x is low, y, z are high
- Φ rises to high, z falls to low
- Φ falls to low, x rises to high
- Φ rises to high, y falls to low, z rises to high, x falls to low
- Φ falls to low, y rises to high

Toggle Circuit

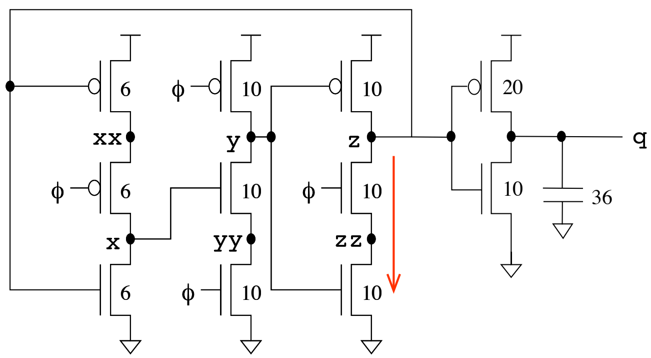


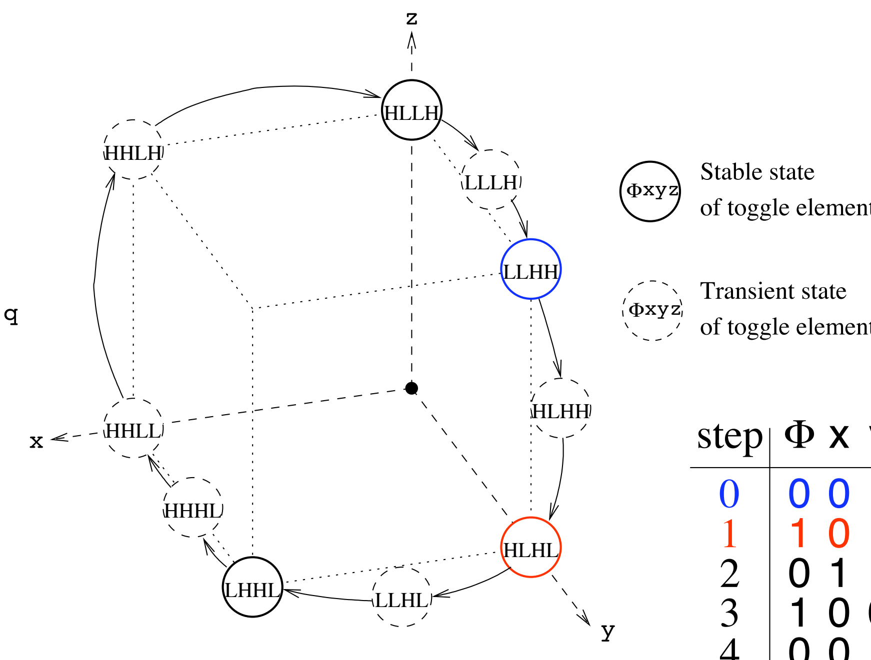


step	Φ	x	y	z
0	0	0	1	1
1	1	0	1	0
2	0	1	1	0
3	1	0	0	1
4	0	0	1	1

- Start from the state where Φ is low, x is low, y, z are high
- Φ rises to high, z falls to low
- Φ falls to low, x rises to high
- Φ rises to high, y falls to low, z rises to high, x falls to low
- Φ falls to low, y rises to high

Toggle Circuit

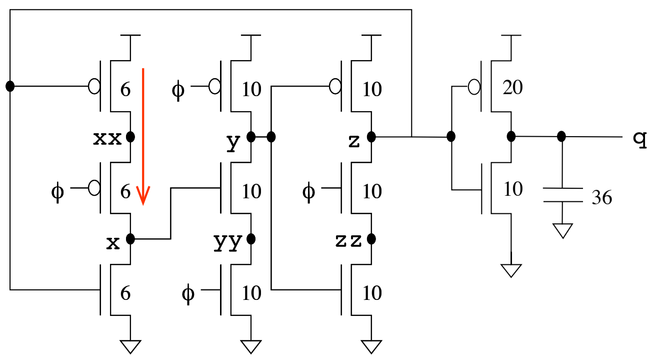


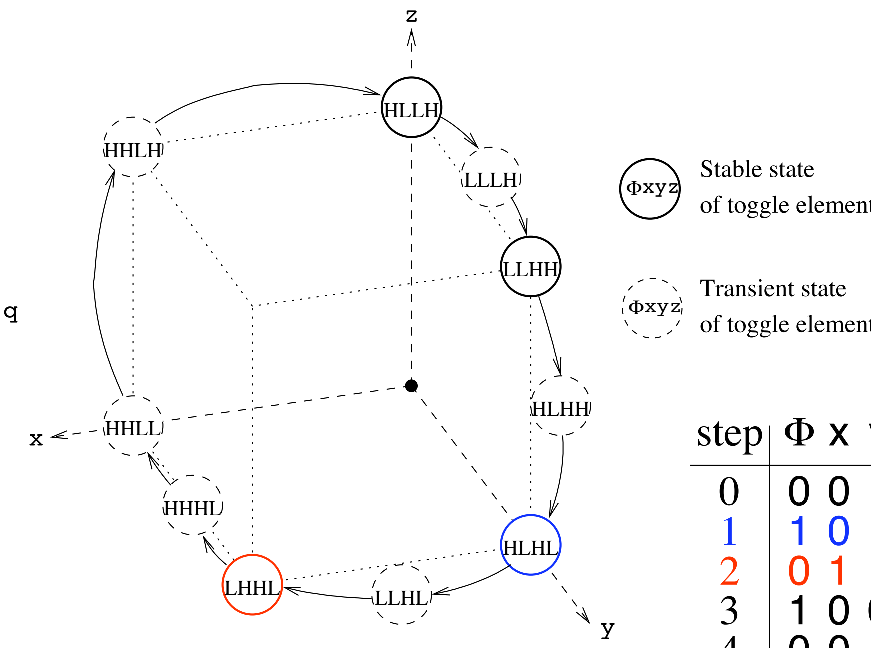


step	Φ	x	y	z
0	0	0	1	1
1	1	0	1	0
2	0	1	1	0
3	1	0	0	1
4	0	0	1	1

- Start from the state where Φ is low, x is low, y, z are high
- Φ rises to high, z falls to low
- Φ falls to low, x rises to high
- Φ rises to high, y falls to low, z rises to high, x falls to low
- Φ falls to low, y rises to high

Toggle Circuit

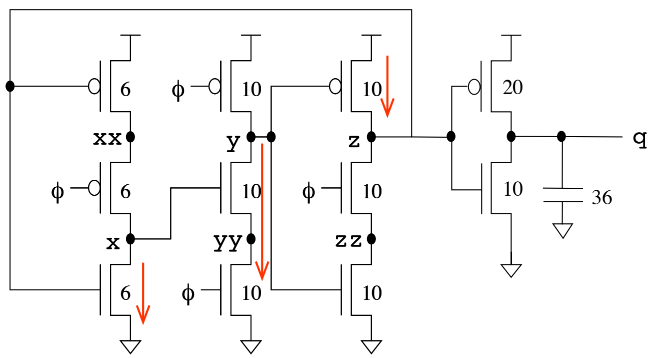


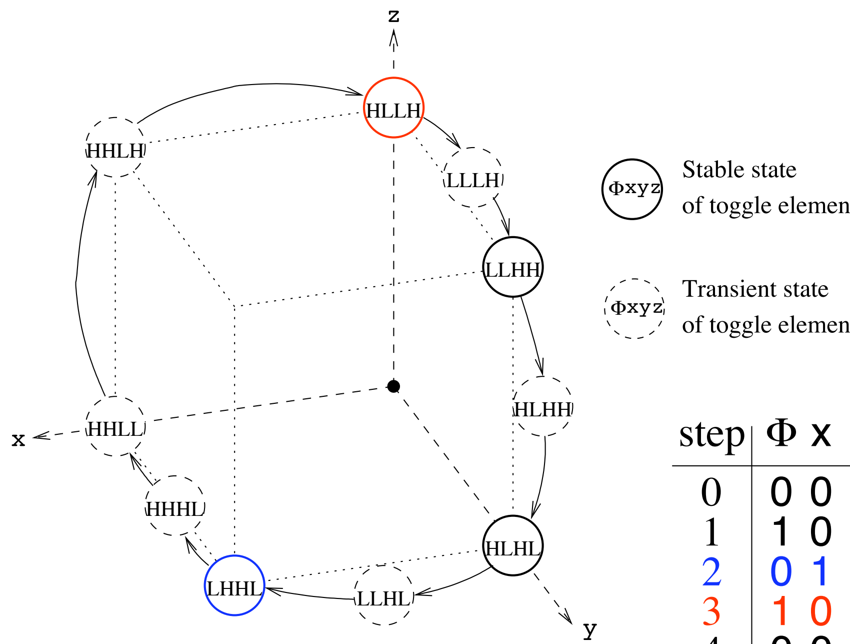


step	Φ	x	y	z
0	0	0	1	1
1	1	0	1	0
2	0	1	1	0
3	1	0	0	1
4	0	0	1	1

- Start from the state where Φ is low, x is low, y, z are high
- Φ rises to high, z falls to low
- ➔ Φ falls to low, x rises to high
- Φ rises to high, y falls to low, z rises to high, x falls to low
- Φ falls to low, y rises to high

Toggle Circuit

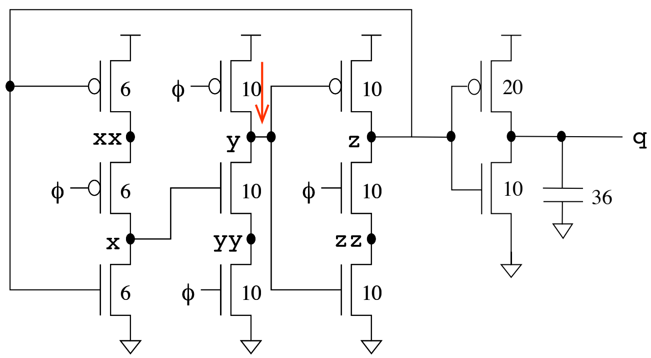


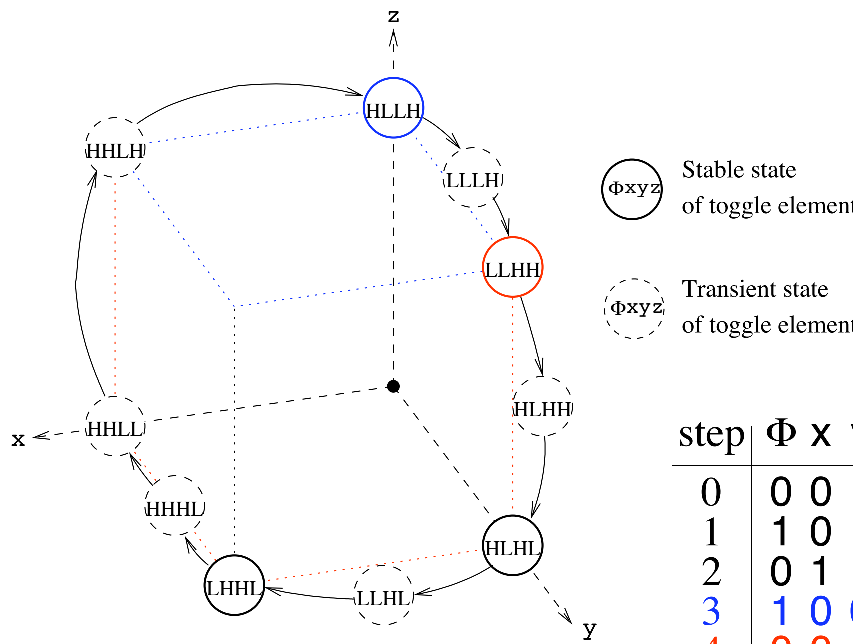


step	Φ	x	y	z
0	0	0	1	1
1	1	0	1	0
2	0	1	1	0
3	1	0	0	1
4	0	0	1	1

- Start from the state where Φ is low, x is low, y, z are high
- Φ rises to high, z falls to low
- Φ falls to low, x rises to high
- Φ rises to high, y falls to low, z rises to high, x falls to low
- Φ falls to low, y rises to high

Toggle Circuit

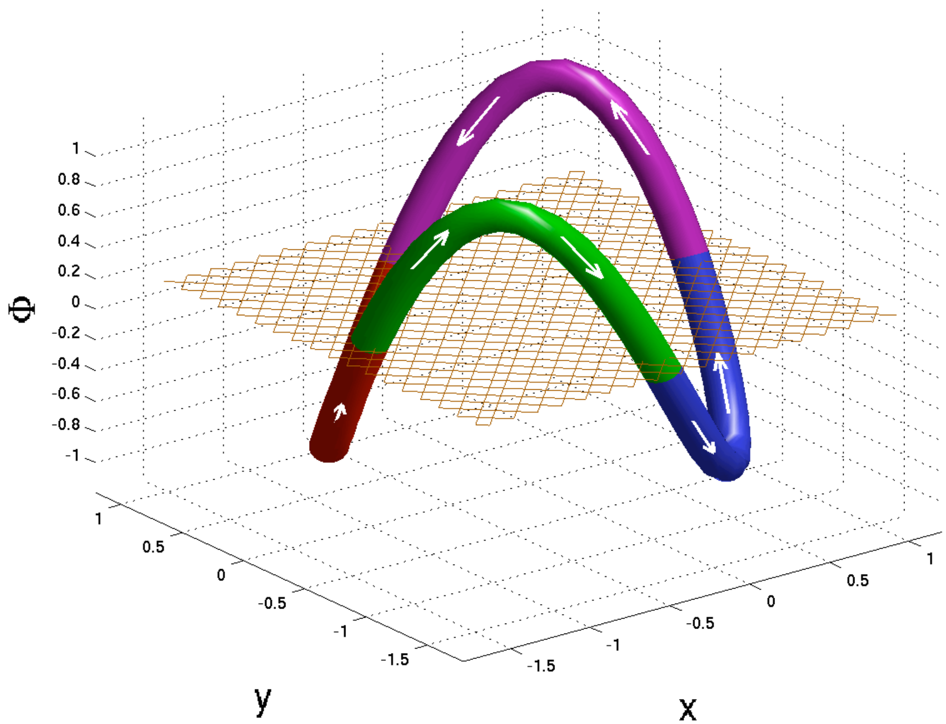


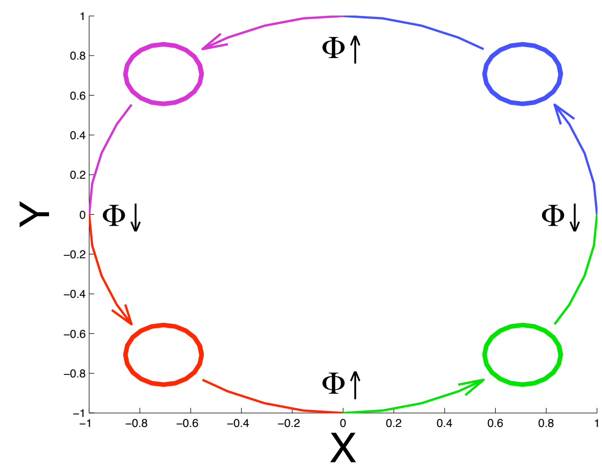


step	Φ	x	y	z
0	0	0	1	1
1	1	0	1	0
2	0	1	1	0
3	1	0	0	1
4	0	0	1	1

- Start from the state where Φ is low, x is low, y, z are high
- Φ rises to high, z falls to low
- Φ falls to low, x rises to high
- Φ rises to high, y falls to low, z rises to high, x falls to low
- Φ falls to low, y rises to high

Specification

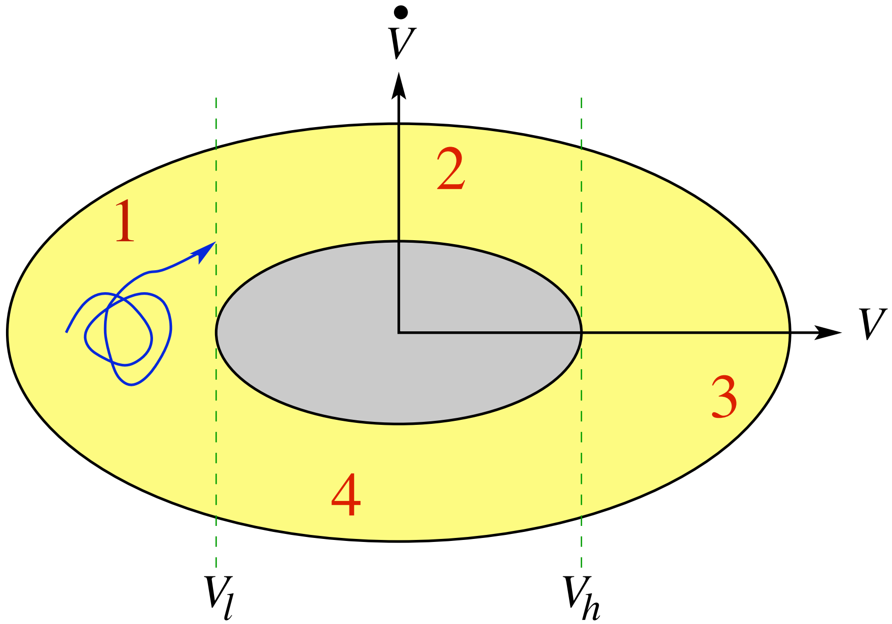


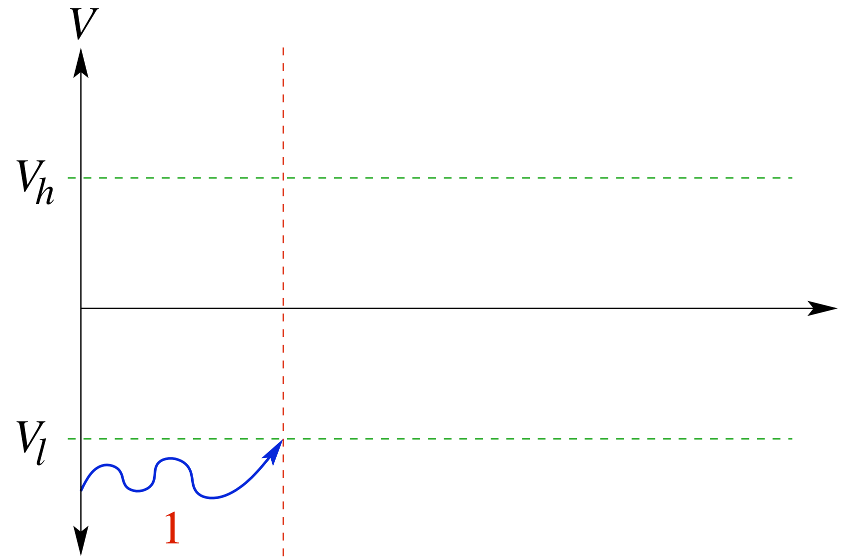


With a "well-behaved" clock input:

- The reachable space is a collection of trajectories whose period is twice that of the clock;
- The output is "well-behaved" like the clock.

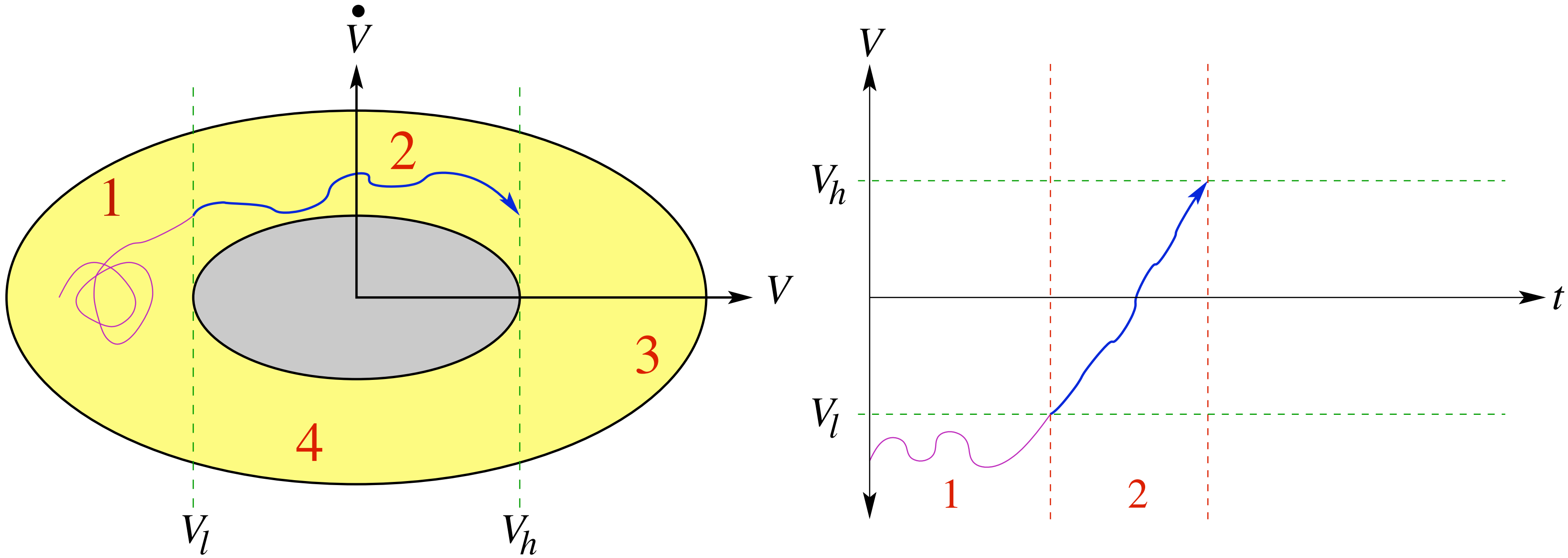
Brockett's Annulus





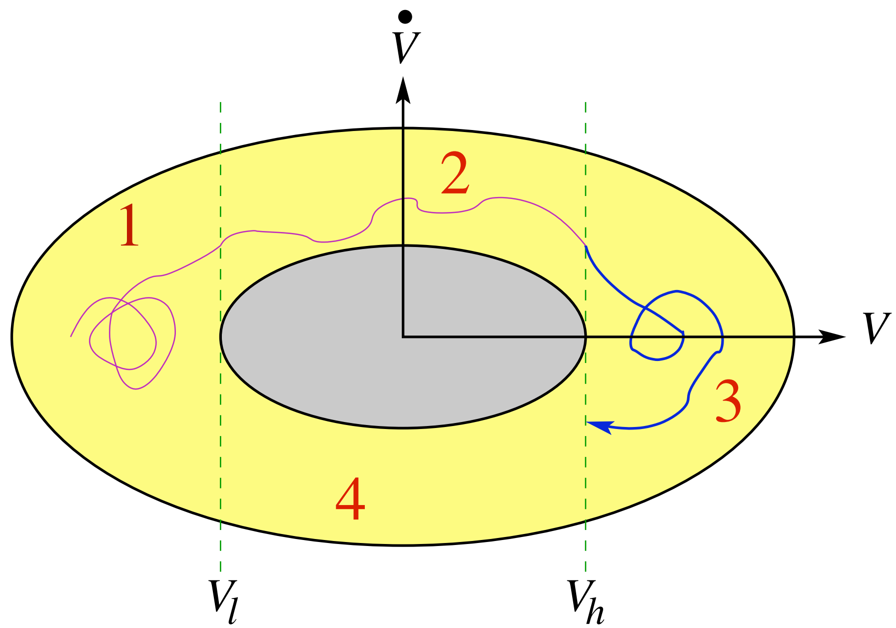
- Region 1 represents a logical low signal. The signal may wander in a small interval.
- Region 2 represents a monotonically rising signal.
- Region 3 represents a logical high signal.
- Region 4 represents a monotonically falling signal.
- Brockett's annulus allows entire families of signals to be specified.

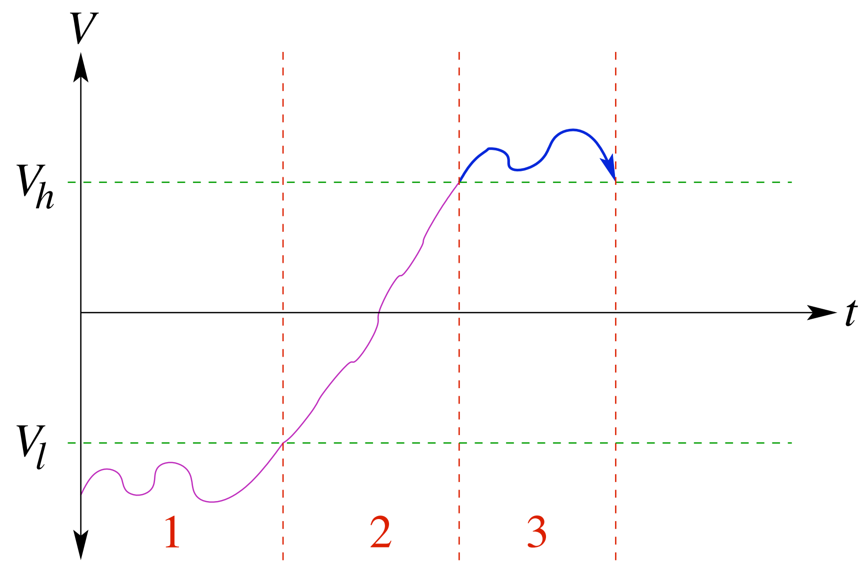
Brockett's Annulus



- Region 1 represents a logical low signal. The signal may wander in a small interval.
- Region 2 represents a monotonically rising signal.
- Region 3 represents a logical high signal.
- Region 4 represents a monotonically falling signal.
- Brockett's annulus allows entire families of signals to be specified.

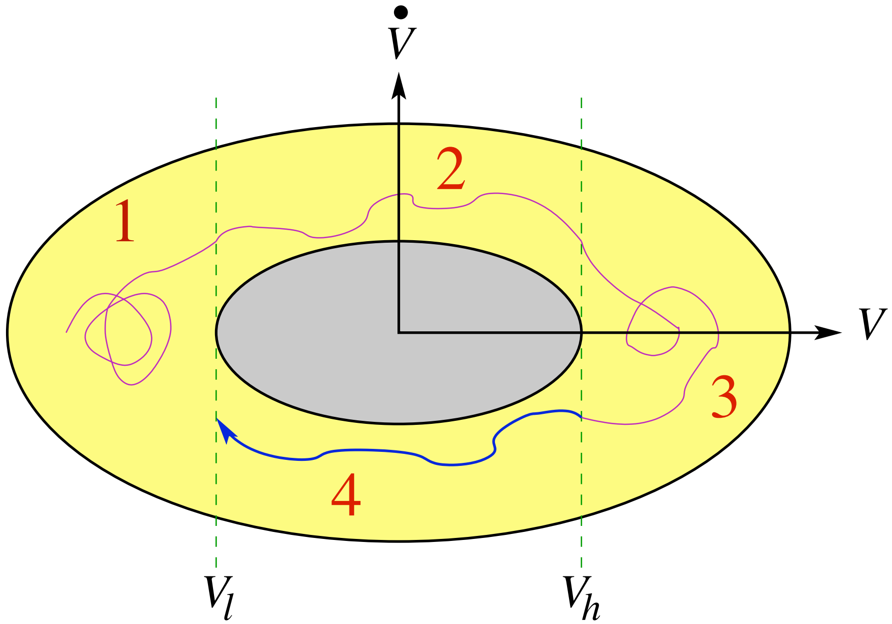
Brockett's Annulus

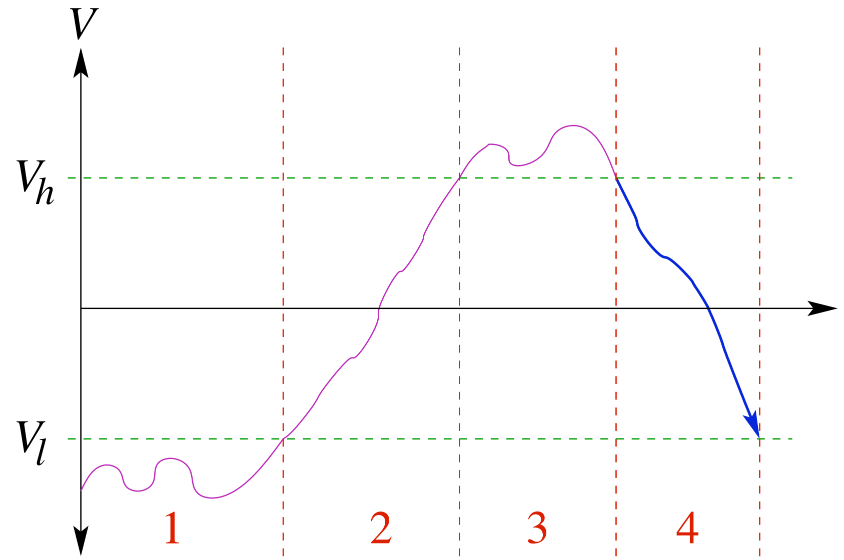




- Region 1 represents a logical low signal. The signal may wander in a small interval.
- Region 2 represents a monotonically rising signal.
- Region 3 represents a logical high signal.
- Region 4 represents a monotonically falling signal.
- Brockett's annulus allows entire families of signals to be specified.

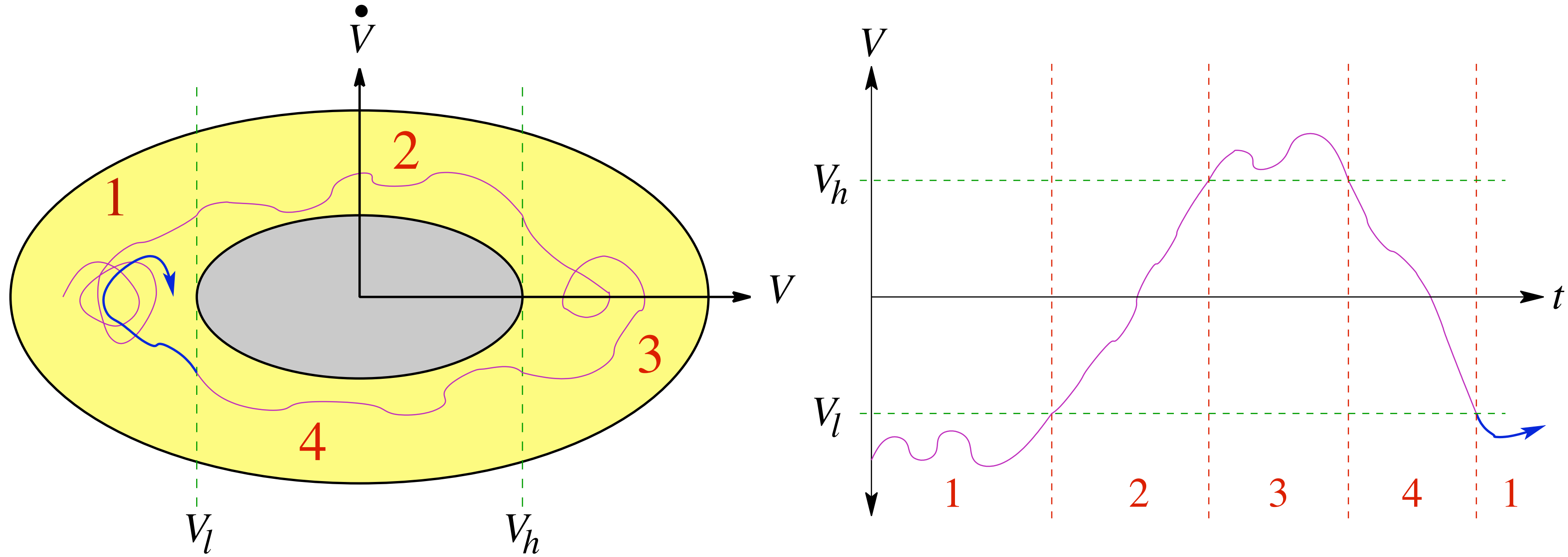
Brockett’s Annulus





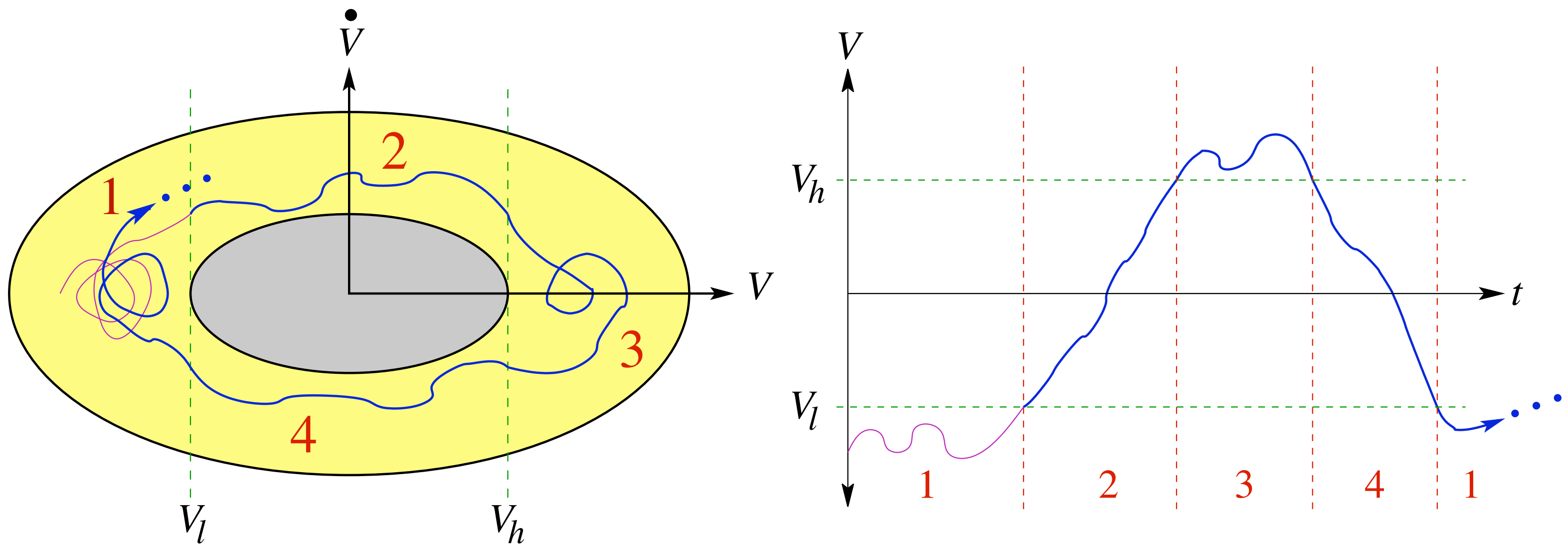
- Region 1 represents a logical low signal. The signal may wander in a small interval.
- Region 2 represents a monotonically rising signal.
- Region 3 represents a logical high signal.
- Region 4 represents a monotonically falling signal.
- Brockett’s annulus allows entire families of signals to be specified.

Brockett's Annulus



- Region 1 represents a logical low signal. The signal may wander in a small interval.
- Region 2 represents a monotonically rising signal.
- Region 3 represents a logical high signal.
- Region 4 represents a monotonically falling signal.
- Brockett's annulus allows entire families of signals to be specified.

Brockett Data Sheets

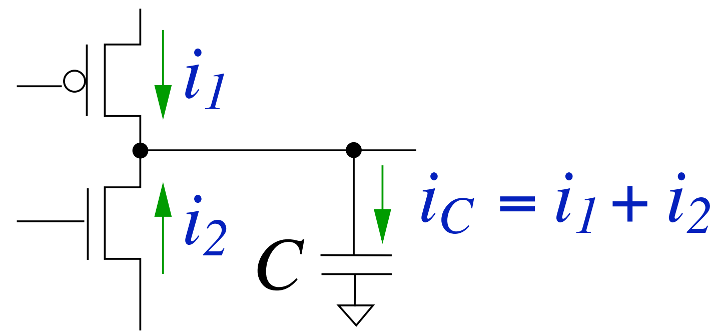


- The left and right boundaries of region 1 give min and max logical low level.
- The left and right boundaries of region 3 give min and max logical high level.
- The upper boundary of region 2 gives the minimum rise time.
- The lower boundary of region 2 gives the maximum rise time.
- The upper and lower boundaries of region 4 give the maximum and minimum fall times respectively.

Circuit Models

- Model MOS circuits as a collection of voltage controlled current sources
- Current function is obtained by simulating TSMC 180nm, 1.8 volt, bulk CMOS process
- Linearize the current function

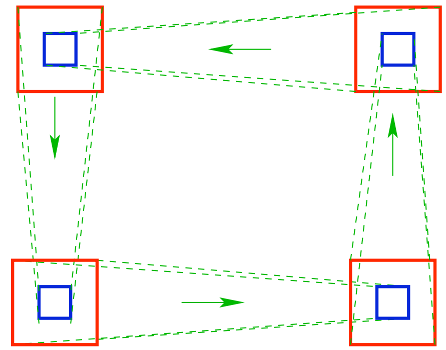
$$AV + b - u \leq ids(V) \leq AV + b + u$$



- Time derivative of voltage

$$\dot{V} \;=\; C^{-1} \cdot I_c$$

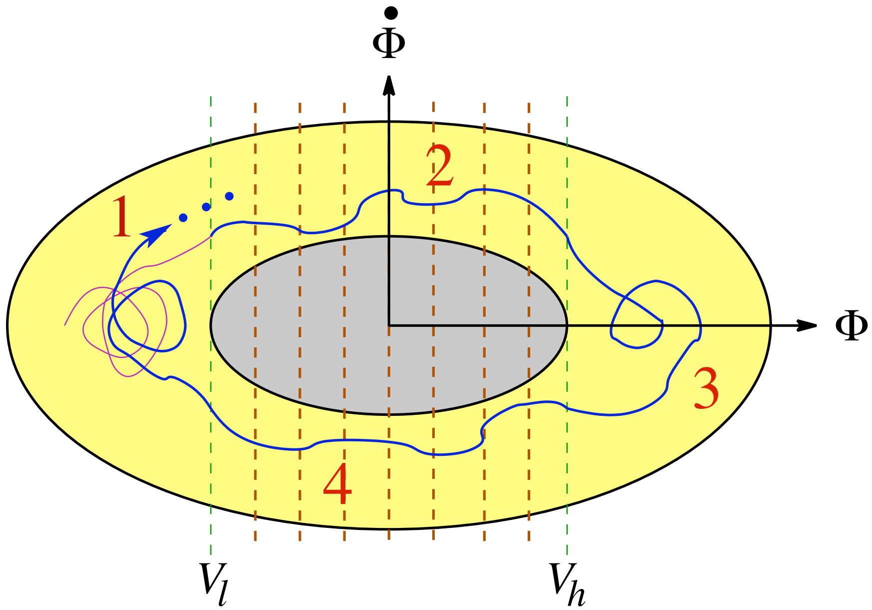
Verification Strategy

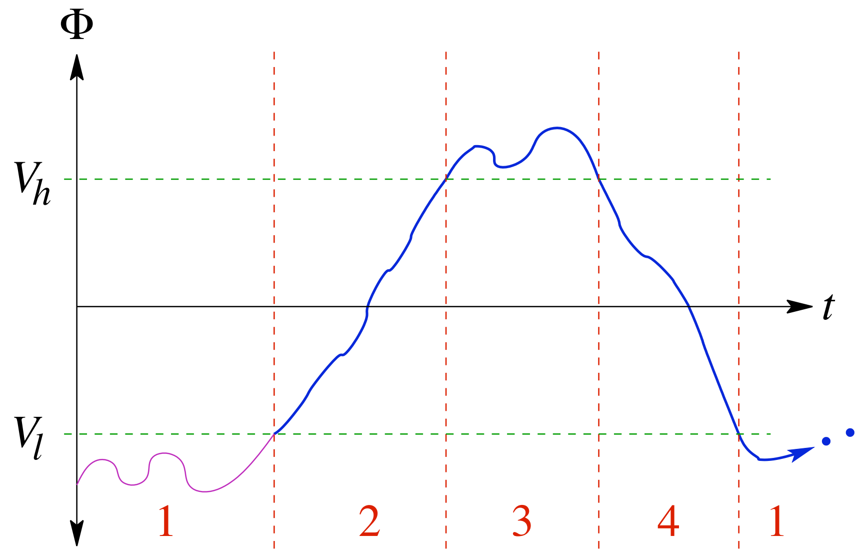


- Separate verification into four phases
  - One phase for each transition of Φ.
  - Assume bounding hyperrectangle for start of phase.
  - Establish bounding hyperrectangle at end of phase.
  - Containment establishes invariant set.
  - Allows parallel execution and parallel debugging.
- Use invariant set to show that Brockett-ring at input implies Brockett-ring at output.
- BUT overapproximation errors need to be managed
  - Slicing
  - Multiple models

Slicing

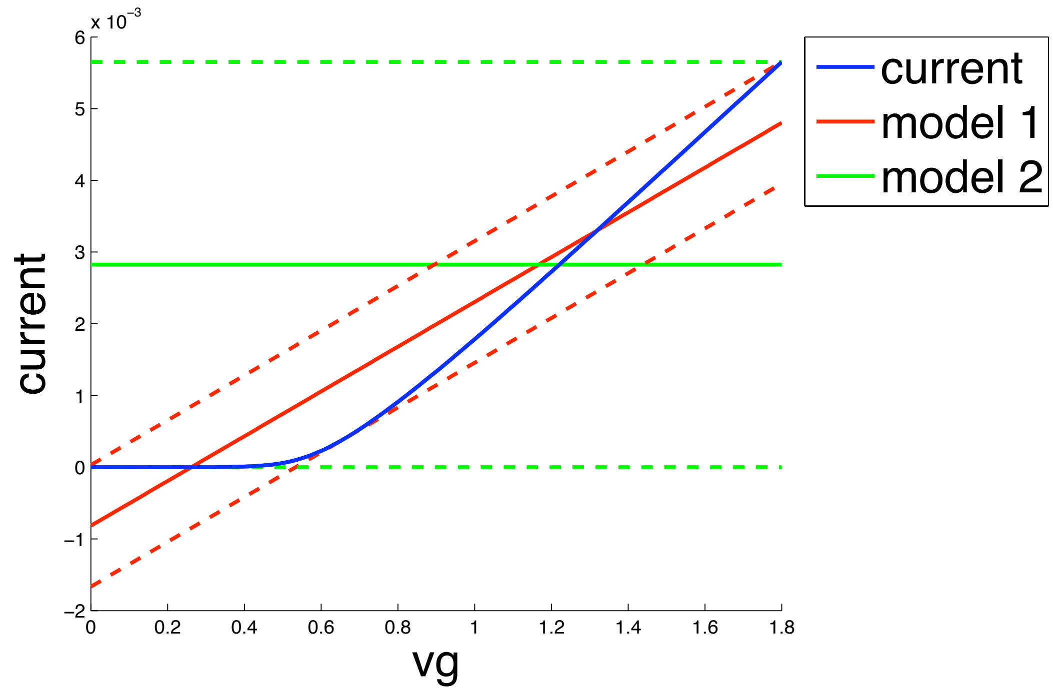
Partition the reachable space along a critical variable:





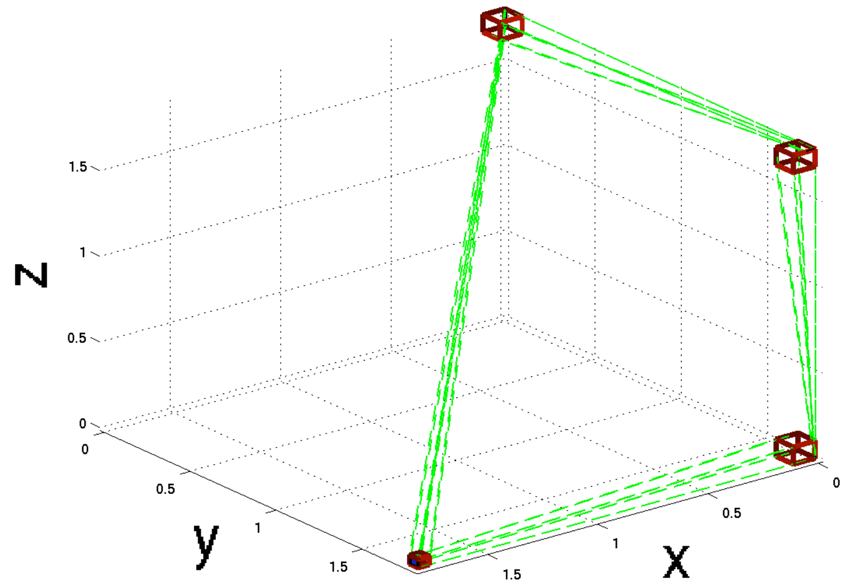
- Large range of Φ leads to large approximation error in linear model.
- Partition reachable space by intervals of Φ.

Multiple Models



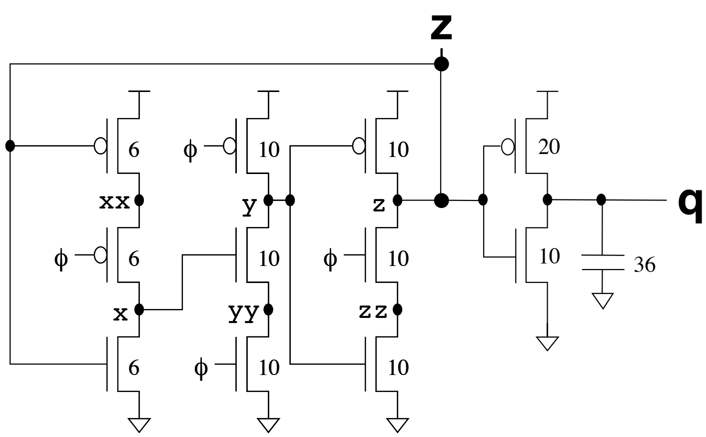
- *Negative current* from source to drain caused by overapproximation of linearization.
- Use multiple models to reduce error.
- Intersect the projectagons to eliminate non-physical states.

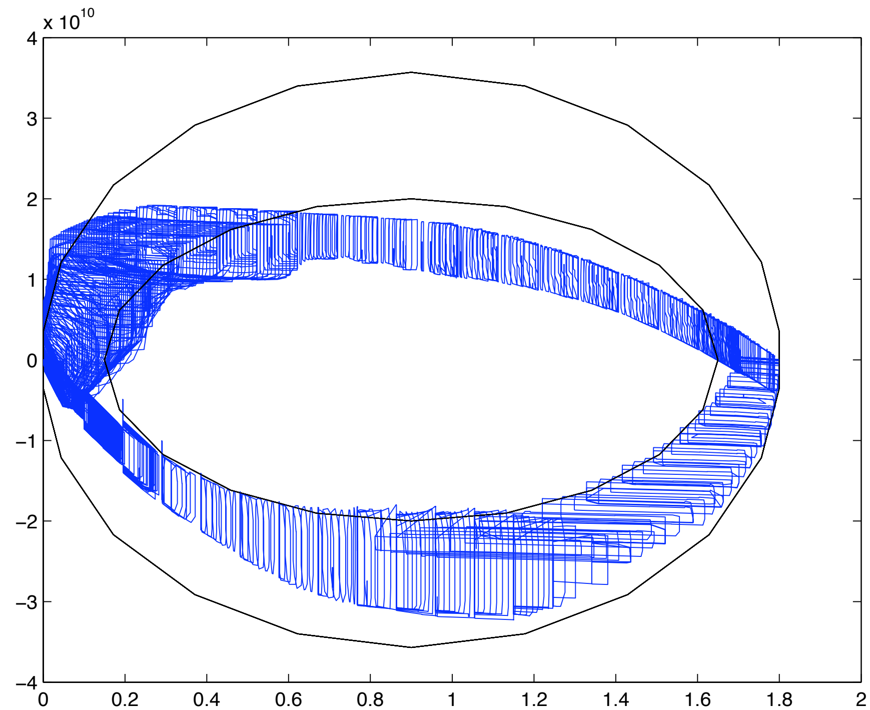
The Invariant Set



- Red: Hyperrectangles at beginning of each phase.
- Blue: Hyperrectangles at end of each phase.
- An invariant set with twice the period of the clock has been established.

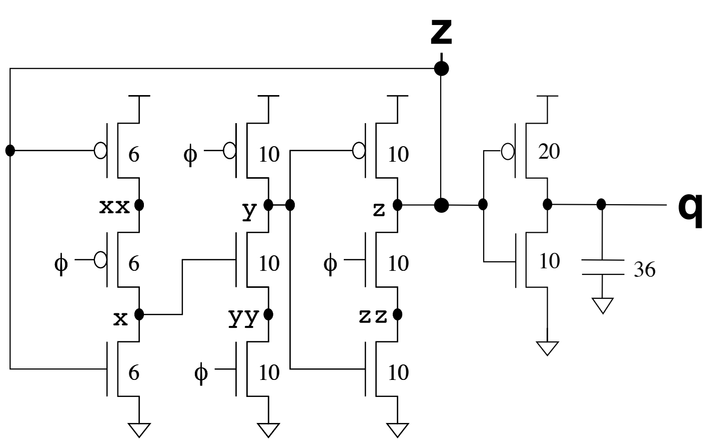
Brockett Ring at z

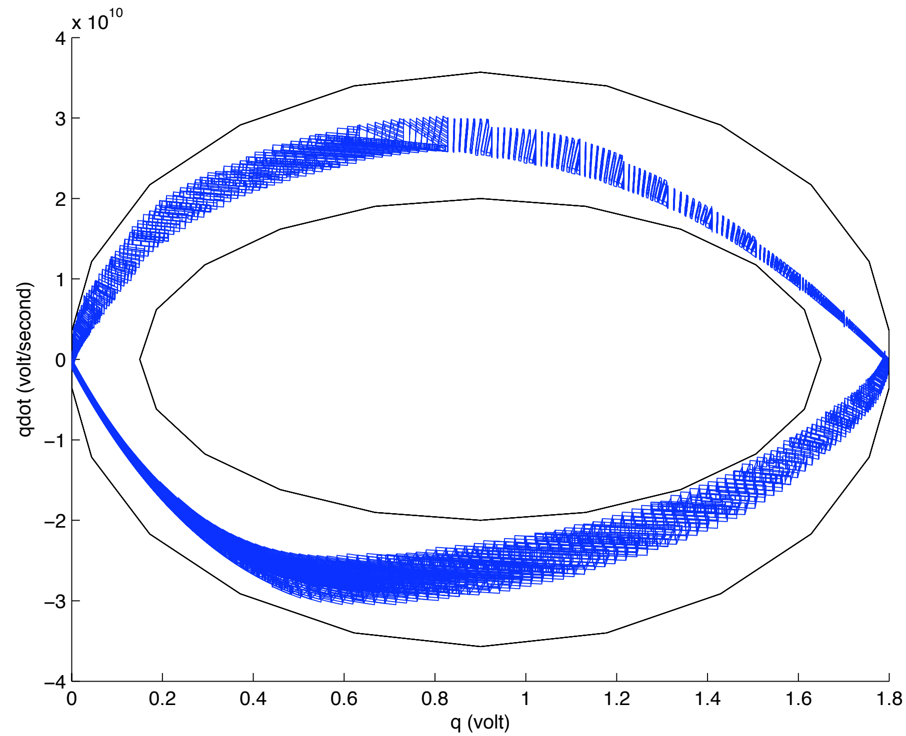




- Construct the brockett annulus for z, ignoring the inverter
- Perform a separate reachability analysis for the output inverter
- Arbitrary ripple counter

Brockett Ring at q





- Construct the brockett annulus for z, ignoring the inverter
- Perform a separate reachability analysis for the output inverter
- Arbitrary ripple counter

Experience from the Toggle

- Coho works for moderate dimensional systems.
- Topological properties provide a mathematically rigorous abstraction from continuous to discrete models.
- Leakage current included in the circuit model.
  - We found that we needed to add keepers to the circuit.
- Slicing and multiple models improved accuracy of linearization to enable successful verification.
- Seven-dimensional record sets a record –
  looks like we have headroom for more.
- Verification process currently involves substantial manual effort – more automation needed before useful in practice.

Conclusion and Future Work

- Conclusion
  - Demonstrate a new reachability method to verify a real circuit
  - Model the circuit with SPICE-level, non-linear differential equations.
  - Projection based representation of reachable space
  - Digital behavior corresponds to topological properties of invariant sets in the continuous space
- Future Work
  - Improve performance
  - Exploit parallelism
  - Develop more accurate circuit model
  - Verify more circuits
  - Apply Coho to hybrid systems
  - Compare with other tools