

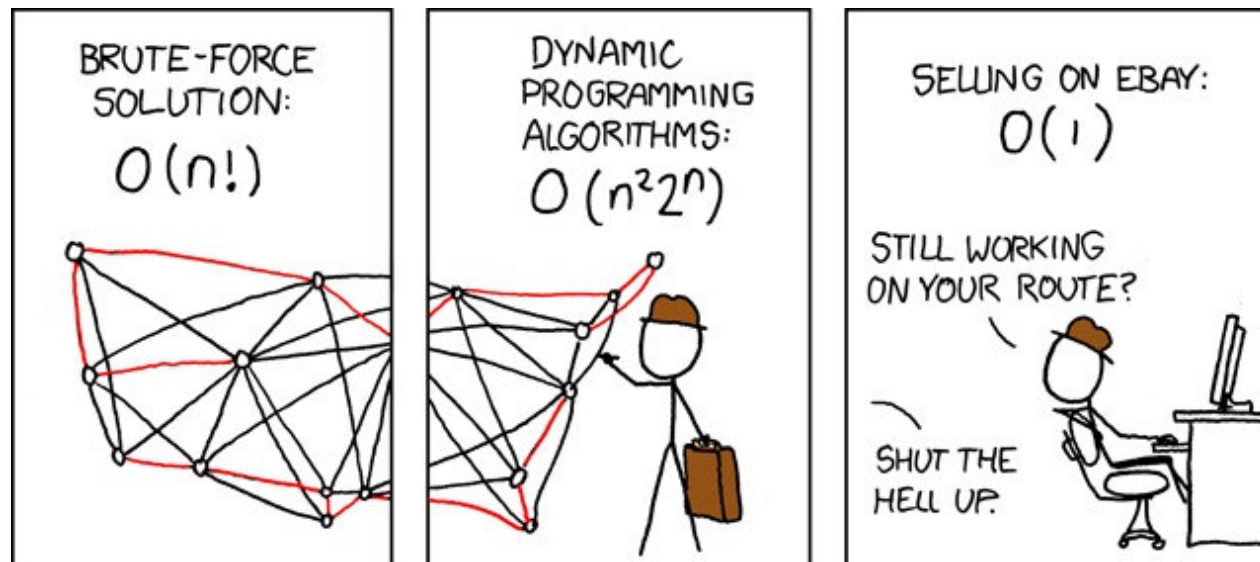


Enhanced Verification by Temporal Decomposition

Mike Case, Hari Mony, Jason Baumgartner, Bob Kanzelman
FMCAD 2009, Nov. 16, 2009

Introduction

- Domain: gate-level property checking (and SEC)
- Problem: simplify the design; remove irrelevant detail



[XKCD]

Outline

- Transient Signals
 - What are they and where do they come from?
 - How to eliminate them

- Initialization Inputs
 - What are they and where do they come from?
 - How to eliminate them

- Experimental Results

Outline

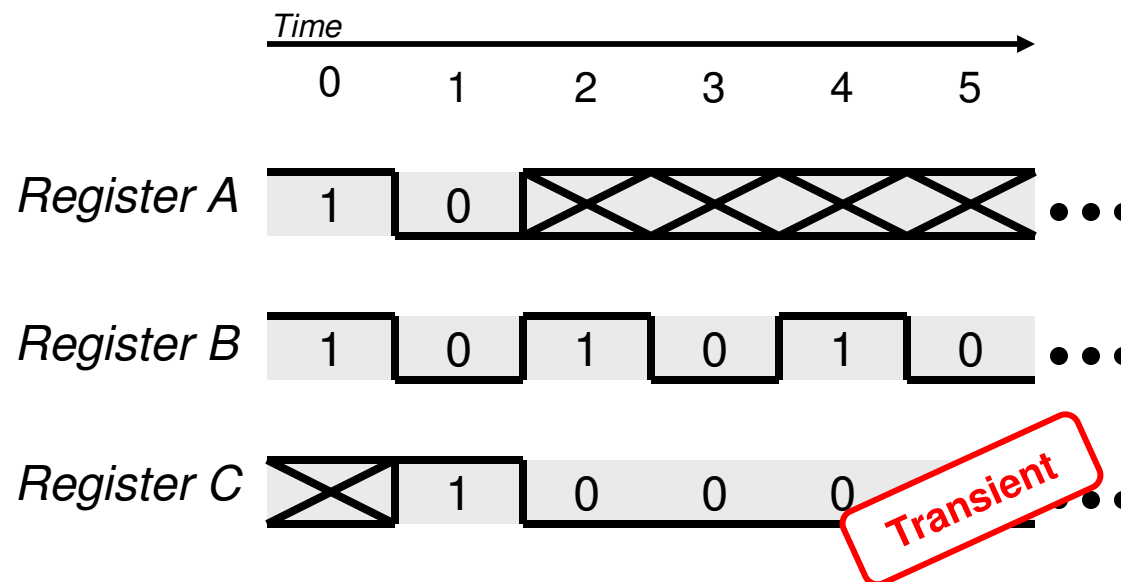
- Transient Signals
 - What are they and where do they come from?
 - How to eliminate them

- Initialization Inputs
 - What are they and where do they come from?
 - How to eliminate them

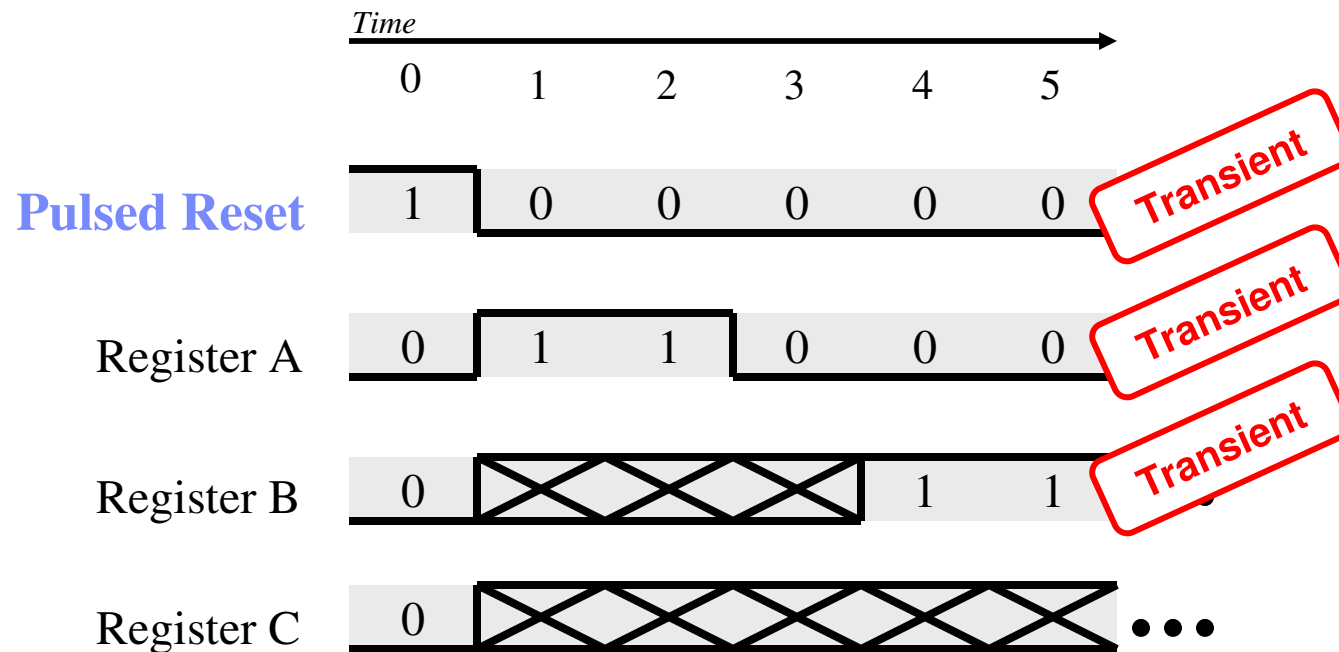
- Experimental Results

Transient Signals

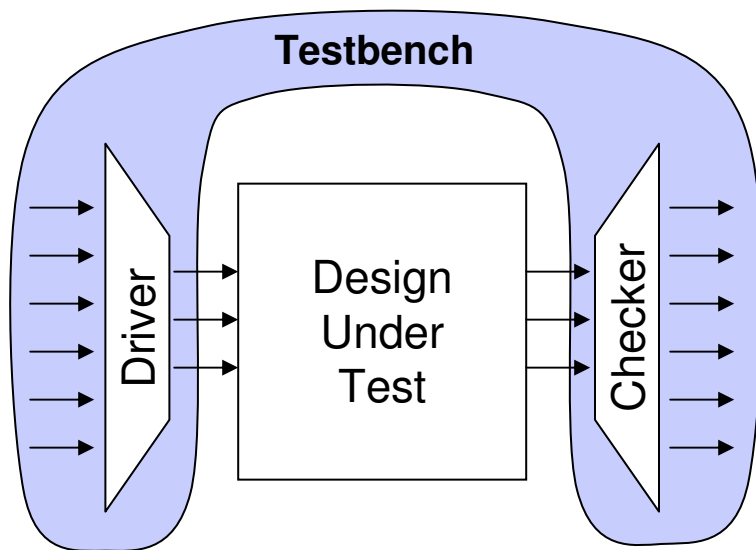
- Definition: a **transient signal** takes arbitrary values for a finite number of clock cycles and then assumes a fixed constant value



Why Do They Occur? – Initialization Sequences



Why Do They Occur? – Verification Testbenches



FPU Pipeline

Time 0:	MUL	NOP	NOP	NOP
Time 1:	NOP	MUL	NOP	NOP
Time 2:	NOP	NOP	MUL	NOP
Time 3:	NOP	NOP	NOP	MUL
Time >3:	NOP	NOP	NOP	NOP

- Testbench → all FPU signals are transient

How Common Are Transient Signals?

- On 105 “hard” IBM designs: 49% had transients
- On 27 “hard” HWMCC designs: 25% had transients

Design	Design Size			T. Found	
	ANDs	Regs	Inputs	Num.	Dur.
IBM0	3039	291	45	8	1
IBM6	77313	9829	544	1	1
IBM8	3011	396	86	2	2
IBM22	11218	908	924	23	10
IBM23	8713	477	6	1	1
IBM28	116322	21607	1316	1150	74
IBM31	7291	696	61	4	3
IBM32	8177	698	65	4	3
nusmvbrp	464	52	11	2	1
nusmvguidancep2	1748	86	84	2	1
nusmvqueue	2376	84	82	2	1
nusmvreactorp2	1242	76	74	2	1
r17b	4087	322	613	18	2

Outline

- Transient Signals
 - What are they and where do they come from?
 - How to eliminate them

- Initialization Inputs
 - What are they and where do they come from?
 - How to eliminate them

- Experimental Results

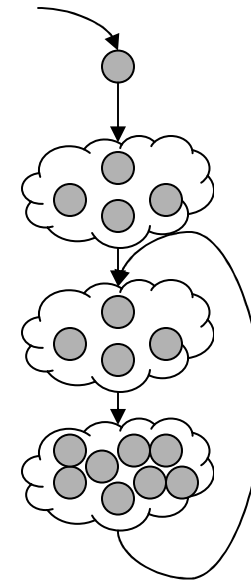
Detecting Transients With Ternary Simulation

Ternary Sim. Execution

Time 0: State=0000, Inputs=XX
 ↓
 Time 1: State=0X1X, Inputs=XX
 ↓
 Time 2: State=X00X, Inputs=XX
 ↓
 Time 3: State=X0XX, Inputs=XX
 ↓
 Time 4: State=X00X, Inputs=XX

Transient

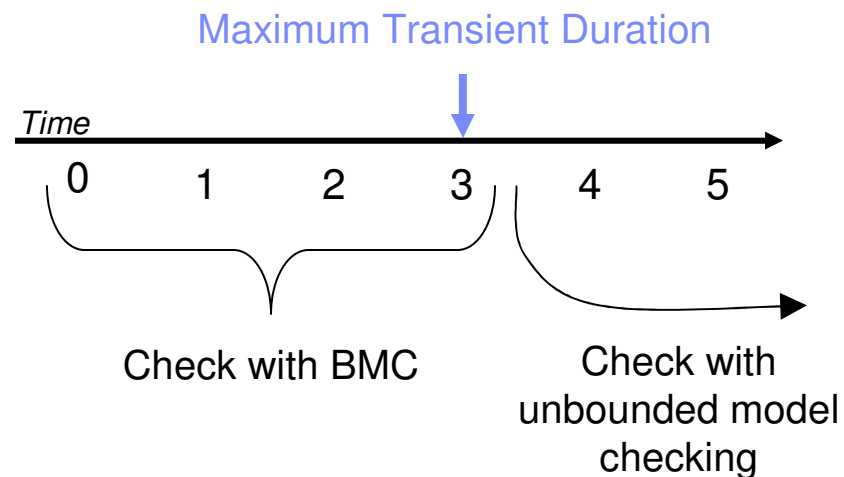
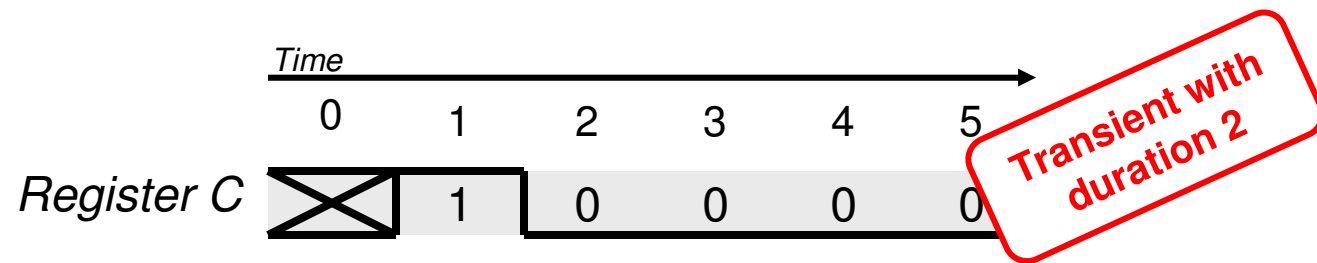
Abstract State Space



Fast, but incomplete

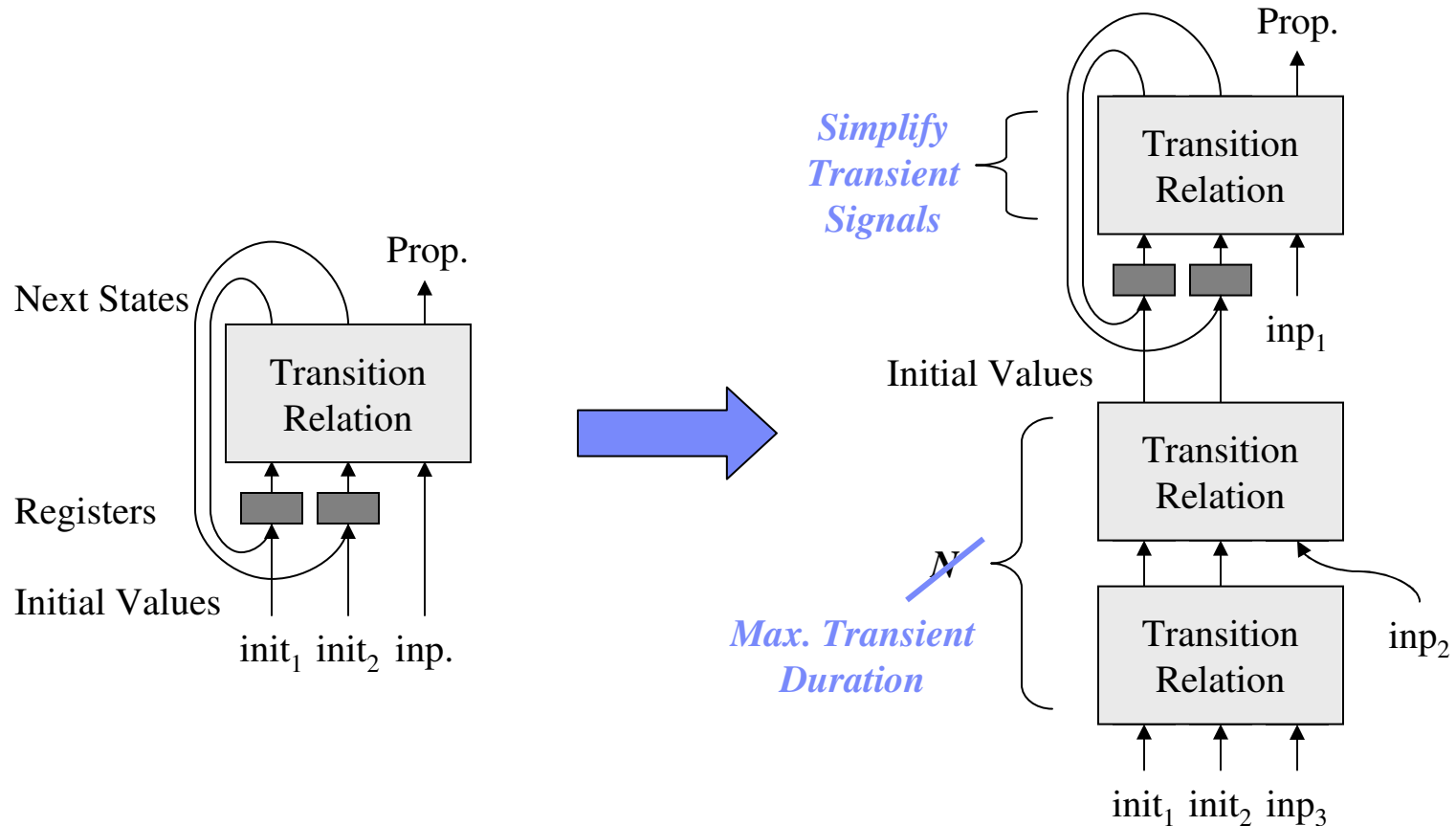
Temporal Decomposition For Verification

- Definition: the **transient duration** is the number of time steps before a transient settles to its constant value



Time Shifting for Unbounded Verification

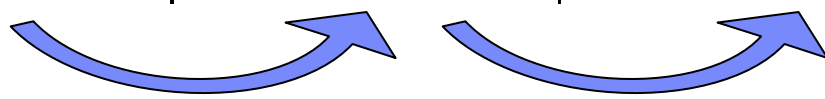
- Modeling: a register's **initial values** is an arbitrary combinational function



- New design starts in any state reachable in N steps.

Simplification Results

Design	T. Found		Transients Simplified		Logic Size Change	
	Num.	Dur.	Num.	Dur.	TR	Total
IBM0	8	1	8	1	-1.10%	1.83%
IBM6	1	1	1	1	-1.44%	2.05%
IBM8	2	2	2	2	-0.27%	2.76%
IBM22	23	10	23	10	-17.17%	19.48%
IBM23	1	1	1	1	-9.57%	0.00%
IBM28	1150	74	0	0	0.00%	0.00%
IBM31	4	3	4	3	-2.23%	7.26%
IBM32	4	3	4	3	-1.18%	7.18%
nusmvbrp	2	1	2	1	-8.18%	3.18%
nusmvguidancep2	2	1	2	1	-1.89%	15.14%
nusmvqueue	2	1	2	1	-4.10%	11.79%
nusmvreactorp2	2	1	2	1	-5.52%	15.82%
r17b	18	2	16	1	-30.38%	44.03%

- 
- Resources limit the transients that can be simplified
 - TR decreases
 - Initialization logic increases

Outline

- Transient Signals
 - What are they and where do they come from?
 - How to eliminate them

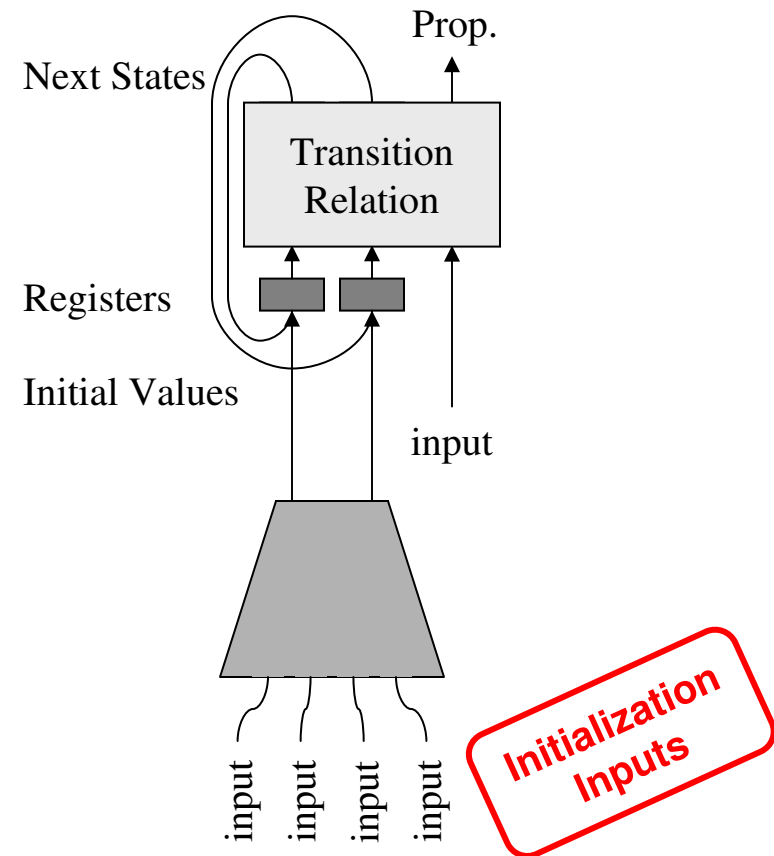
- Initialization Inputs
 - What are they and where do they come from?
 - How to eliminate them

- Experimental Results

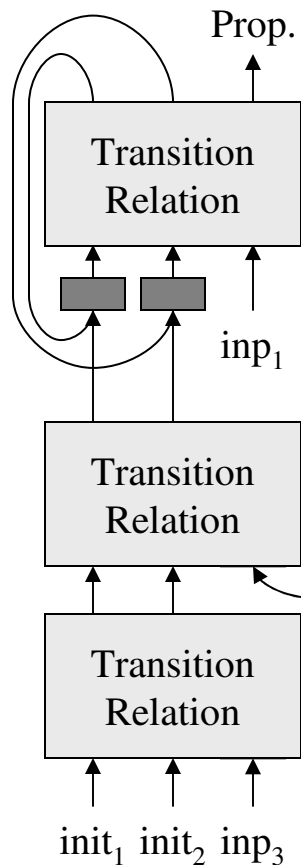
We have lots of initialization inputs

- Definition: an **initialization input** is a primary input that is only needed to compute the initial state

Design	Design Size			Init. Inputs
	ANDs	Regs	Inputs	
IBM0	3039	291	45	0
IBM6	77313	9829	544	0
IBM8	3011	396	86	1
IBM22	11218	908	924	275
IBM23	8713	477	6	0
IBM28	116322	21607	1316	232
IBM31	7291	696	61	0
IBM32	8177	698	65	0



Transient Simplification Creates Initialization Inputs



Design	Design Size			Init. Inputs	Transient Simp.		Init. Inputs
	ANDs	Regs	Inputs		Num.	Dur	
IBM0	3039	291	45	0	8	1	25
IBM6	77313	9829	544	0	1	1	409
IBM8	3011	396	86	1	2	2	58
IBM22	11218	908	924	275	23	10	368
IBM23	8713	477	6	0	1	1	0
IBM28	116322	21607	1316	232	0	0	0
IBM31	7291	696	61	0	4	3	178
IBM32	8177	698	65	0	4	3	178

- We created initialization inputs!

Outline

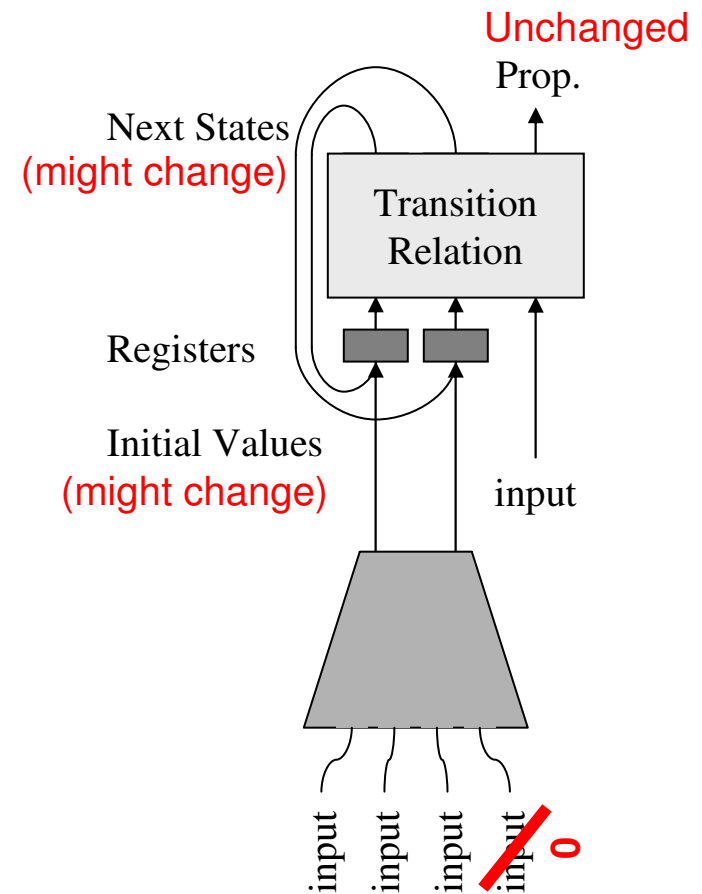
- Transient Signals
 - What are they and where do they come from?
 - How to eliminate them

- Initialization Inputs
 - What are they and where do they come from?
 - How to eliminate them

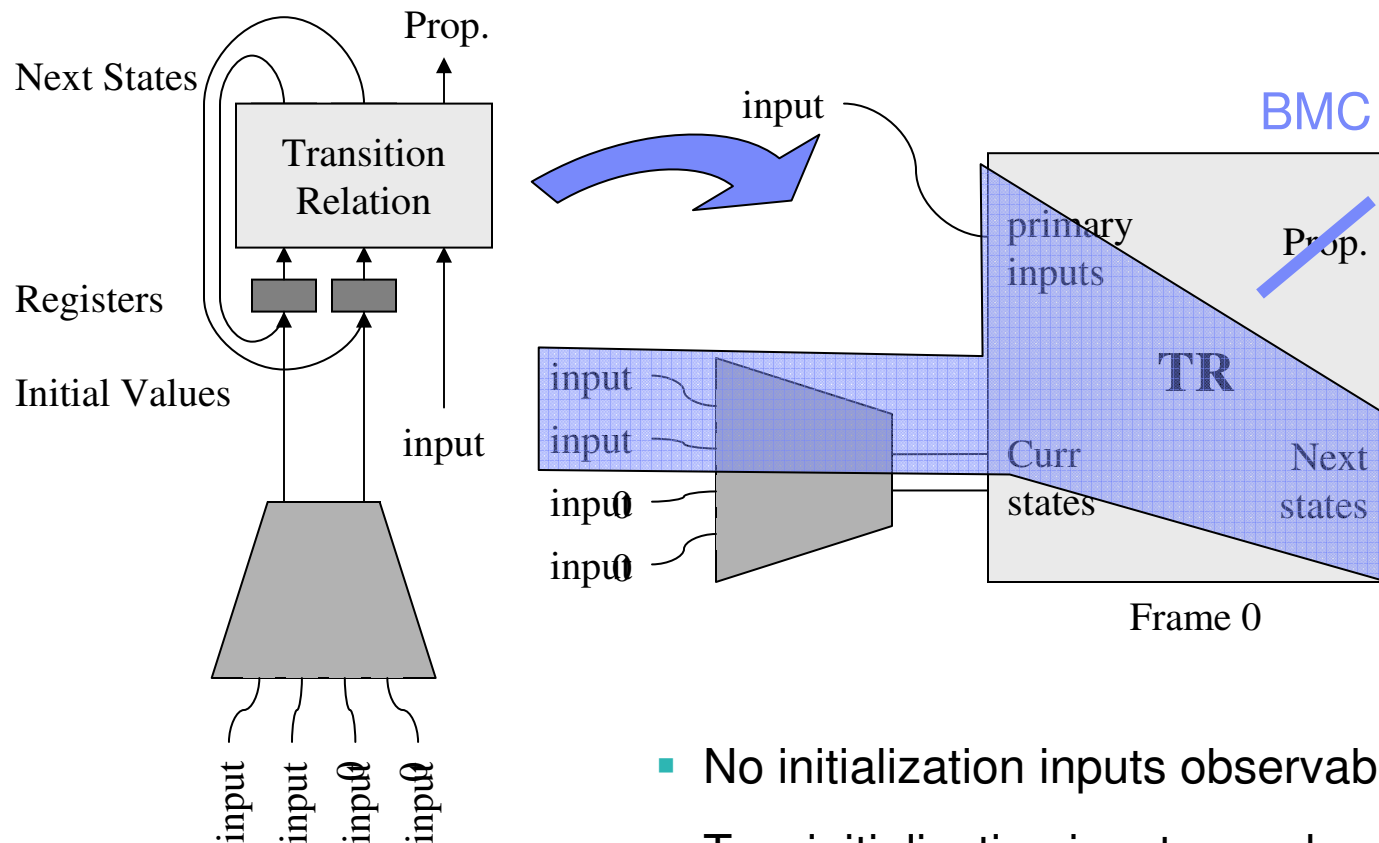
- Experimental Results

Initialization and Observability

- Definition: a signal X is **observable** at signal Y if a toggle at X can cause a toggle at Y
 - Approximate with structural analysis
- Simplify initialization inputs that are never observable at any property

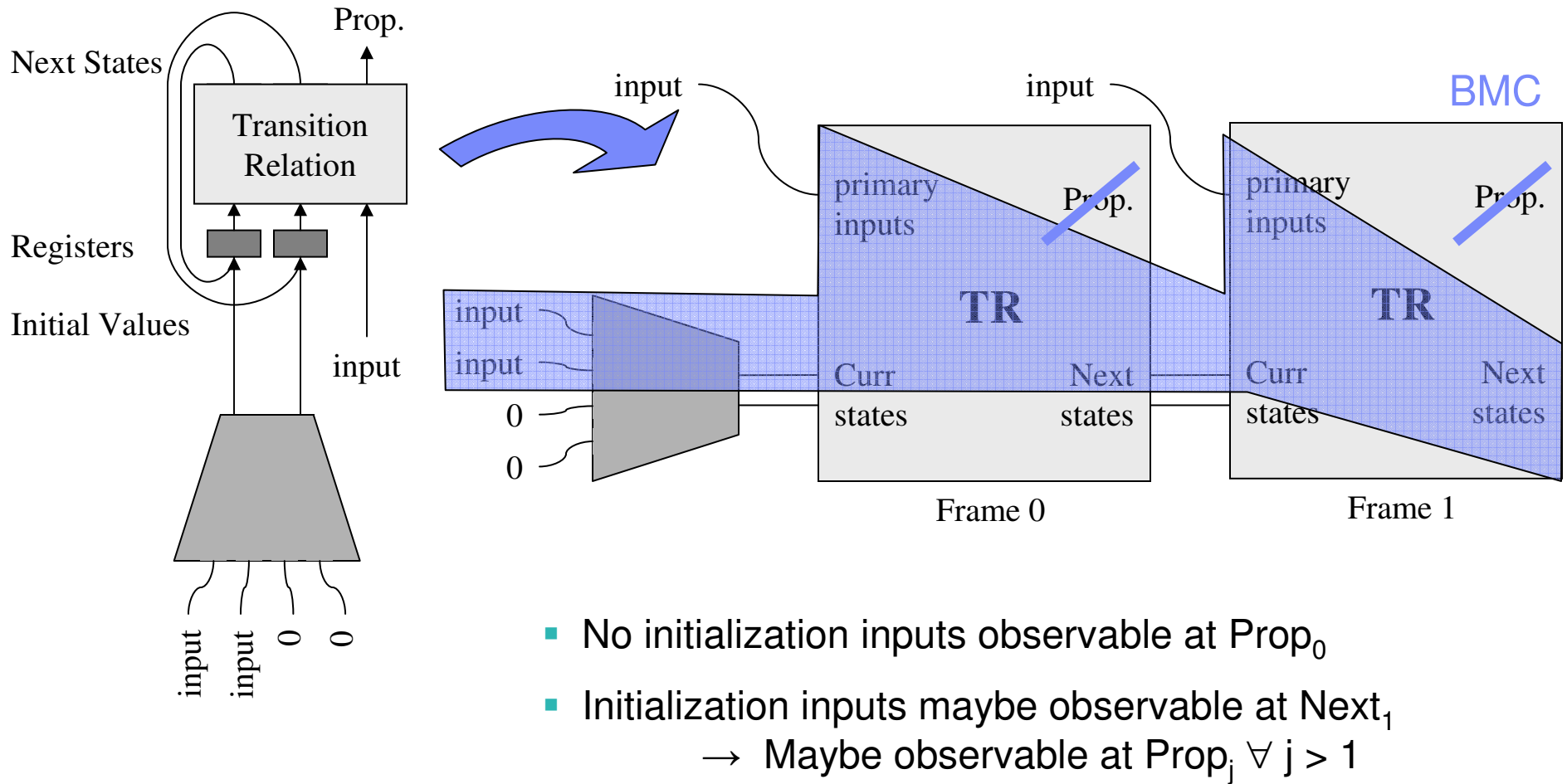


Simplification of Initialization Inputs: High Level



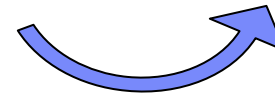
- No initialization inputs observable at Prop_0
- Two initialization inputs un-observable at Next_0
 \rightarrow Un-observable at $\text{Prop}_j \forall j > 0$

Simplification of Initialization Inputs: High Level



Simplification Results

Design	Init. Inputs	Transient Simp.		COI-Based Input Simp., Alg. 3		
		Num.	Dur	Init. Inputs	Removed	Time
IBM0	0	8	1	25	0	1.59 s
IBM6	0	1	1	409	1	9.55 s
IBM8	1	2	2	58	12	0.67 s
IBM22	275	23	10	368	214	1.61 s
IBM23	0	1	1	0	0	0.00 s
IBM28	232	0	0	0	4	0.34 s
IBM31	0	4	3	178	3	0.36 s
IBM32	0	4	3	178	3	0.37 s



Fast, but incomplete

Outline

- Transient Signals
 - What are they and where do they come from?
 - How to eliminate them

- Initialization Inputs
 - What are they and where do they come from?
 - How to eliminate them

- Experimental Results

Runtime of our Methods + BMC Results

- Two methods presented:
 - Transient Simplification – Runtime capped at 10 seconds
 - Initialization Input Simplification – Runtime capped at 10 seconds
- Tested on 105 “hard” industrial designs and 27 “hard” academic designs
 - Largest was 632K ands and 97K registers

TABLE IV. BMC Depth Comparison

Benchmark Set	30-min BMC	Trans. Simp. + Input Simp. + 29-min BMC
IBM (135)	100.00%	102.20%
HWMCC (28)	100.00%	103.68%

Note: this is a systematic improvement on a large benchmark suite, not a collection of anecdotal examples

Synthesis Results

TABLE VII. Design Size after Retiming, On 163 Designs

Flow	Ands	Regs.	Inputs
Baseline (after Comb. Synth.)	100.00%	100.00%	100.00%
Retiming + Retiming	116.81%	91.06%	174.56%
Retiming + Algorithms 2,3	108.04%	90.73%	163.65%

TABLE VI. Design Size after Signal Correspondence, On 163 Designs

Flow	Ands	Regs.
Baseline (after Comb. Synth.)	100.00%	100.00%
Signal Correspondence	76.23%	73.91%
Trans. Simp. + Input Simp. + Signal Correspondence	75.39%	72.21%

Verification Results

TABLE V. Interpolation Comparison

Benchmark Set	Transient + Input Simp.	
	Enables Proof	Breaks Proof
IBM (135)	10	3
HWMMC (28)	0	1

- Induction results:
 - Transient logic often breaks induction
 - Our methods can render problems inductive
 - Design 1: 22k Ands and 800 registers, solvable in 42 sec
 - Design 2: 20k Ands and 3k registers, solvable in 56 sec

Conclusion

- Transient and Initialization Input Simplification
- Fast running
- Benefits synthesis and verification

Thank You