UNIVERSITY OF THESSALY

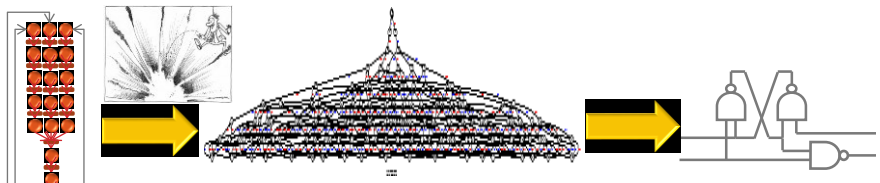
TECHNION Israel Institute of Technology

USC Viterbi School of Engineering University of Southern California

Asynchronous Design Seminar at University of Verona – Lecture Notes 6

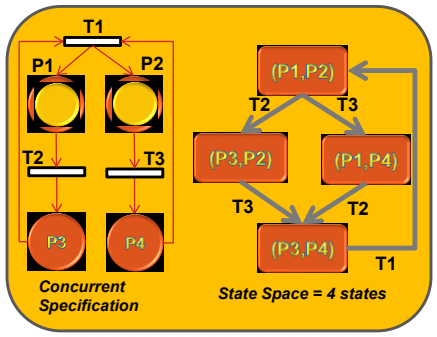
Asynchronous Control Circuits – PTnet to MSFSM Decomposition Flow

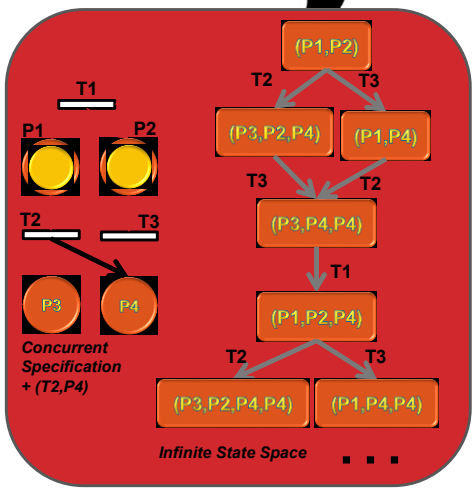
State Explosion Revisited



- *Logic Synthesis and Verification require specification's full state space*
  - *State space size is exponential compared to the specification*

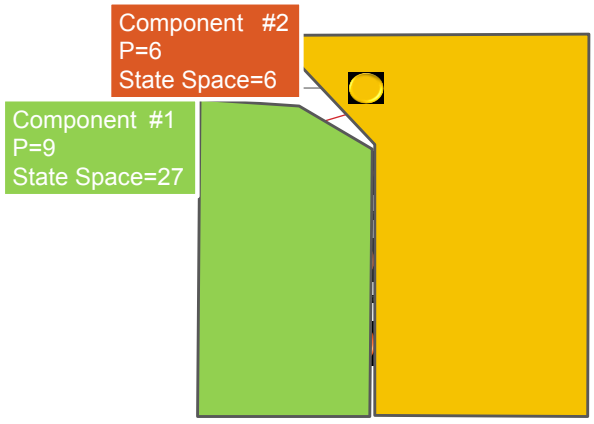
Specification and Confluence





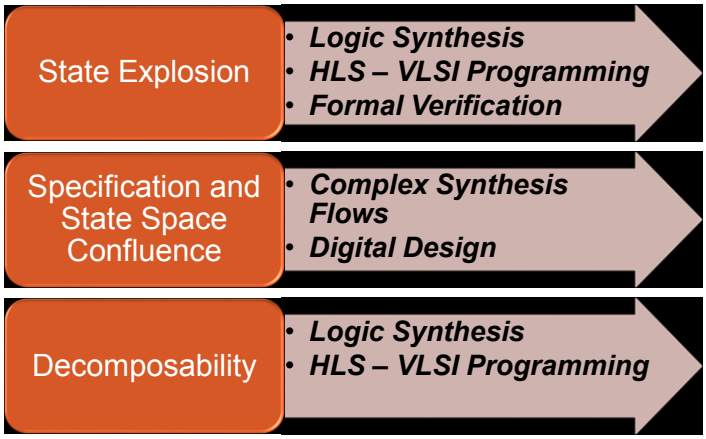
- ***Changes at the Concurrent Specification Level Have Unpredictable Results at the State Space***

PTnet Decomposability

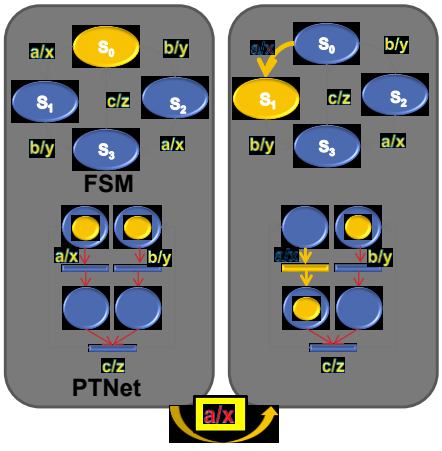


- ***Concurrent Specification Decomposition is Unpredictable as it is Evaluated with Specification Metrics, e.g. number of places.***

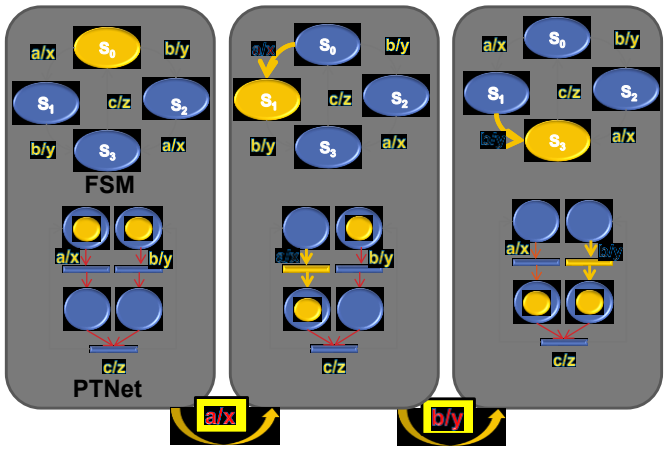
Applications



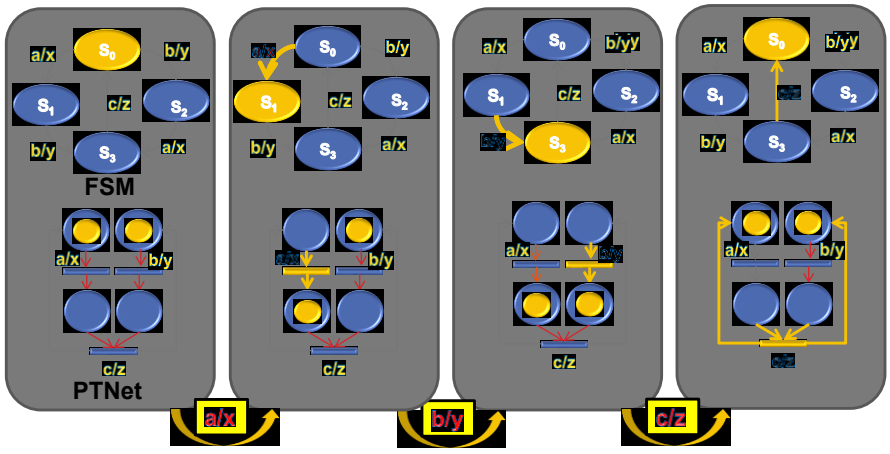
PTnet Operation



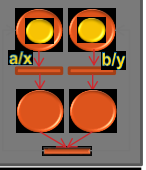
PTnet Operation



PTnet Operation

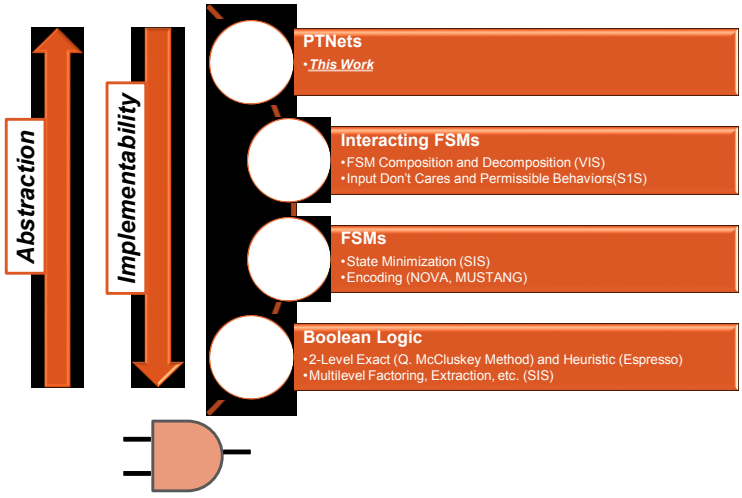


Taxonomy of Control Model Expressiveness

Algebra	FSM	Inter. FSMs	PTNet
$S_0 = S_1.b + S_2.a + S_0.(a+b)'$ $S_1 = S_0.a + S_1.b'$ $S_2 = S_0.b + S_2.a'$ $x = S_0.a + S_1 + S_2.a$			

Control Model	State Space	Choice	Concurrency	Synchronization
Algebra	Implicit	Implicit	Implicit	Implicit
FSM	Explicit	Explicit	Implicit	Implicit
Inter. FSMs	Implicit	Explicit	Explicit	Implicit
PTNet	Implicit	Explicit	Explicit	Explicit

Synchronous Control Model Implementation



Asynchronous Control Model Implementation

(A,BM)FSM Synthesis[1]	PTNet Synthesis[2]	Decomposition[4]	Structural Synthesis[5]	Direct Mapping[3]
• Conventional FSM Heuristics • Exponential Concurrent Specifications	• Compact Concurrent Specifications • Exponential Synthesis Time	• Lower Synthesis Time • Infeasible Implementation • Worst Case Exponential	• Lower Synthesis Time • Worst Case Exponential	• Linear Synthesis Time • Suboptimal Results

[1]K. Y. Yun and D. L. Dill, *"Automatic synthesis of extended burst-mode circuits"*, IEEE TCAD, 1999

[2]J. Cortadella et al., *"Logic Synthesis of Asynchronous Controllers and Interfaces"*, Springer-Verlag, 2002.

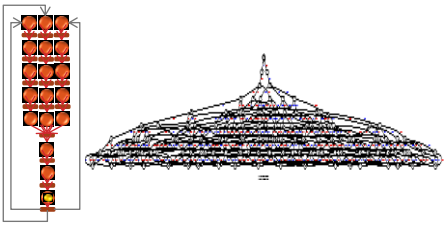
[3]D. Sokolov et al., *"Direct Mapping of Low-Latency Asynchronous Controllers from STGs"*, IEEE TCAD, 2007

[4]D. Wist et al., *"Signal transition graph decomposition: internal communication for speed independent circuit implementation"*, IET Computers & Digital Techniques, 2011

[5]E. Pastor et al. *"Structural Methods for the Synthesis of Speed-Independent Circuits"*, IEEE TCAD, 1998

Commonly used PTnet Implementation Flows

Direct Mapping*	Synthesis#
• Linear Complexity • Suboptimal Result	• Globally Optimal Result • Exponential Complexity

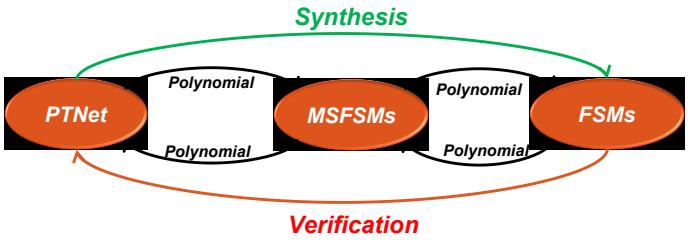


*D. Sokolov, A. V. Bystrov, and A. Yakovlev, "Direct Mapping of Low-Latency Asynchronous Controllers from STGs," IEEE TCAD, 2007.

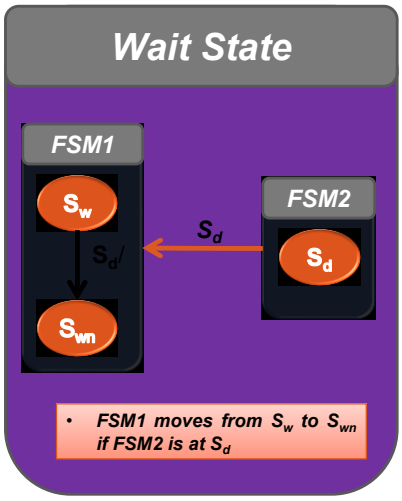
#J. Cortadella et al., Logic Synthesis of Asynchronous Controllers and Interfaces. Springer-Verlag, 2002.

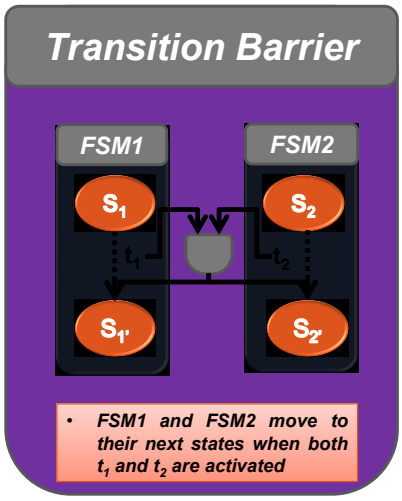
Multiple Synchronised FSMs (MSFSMs) Model

- Novel Model for Concurrent Control Specifications
  - Parallel Tasks Described with FSMs
  - Synchronization explicitly described based on two primitives
    - **Transition Barriers, Wait States**
  - Polynomial Synthesis and Verification Paths

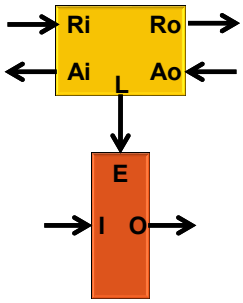


MSFSM Synchronisation Primitives

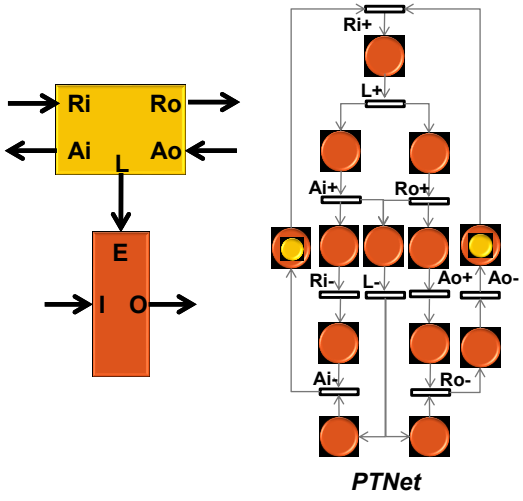




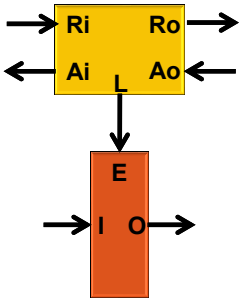
MSFSMs Example – 4-phase Latch Controller

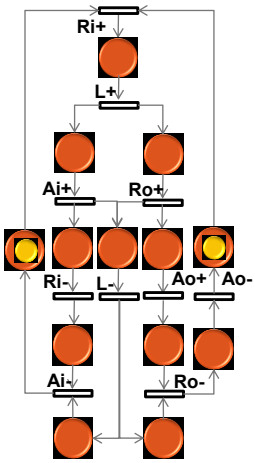


MSFSMs Example – 4-phase Latch Controller

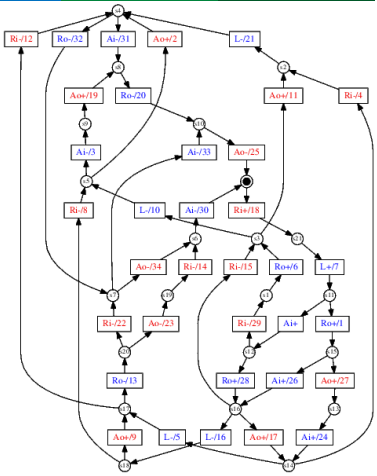


MSFSMs Example – 4-phase Latch Controller



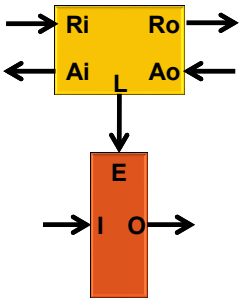


PTNet

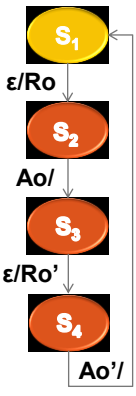


State Graph (FSM)

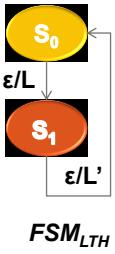
MSFSMs Example – 4-phase Latch Controller



FSM_{LHS}

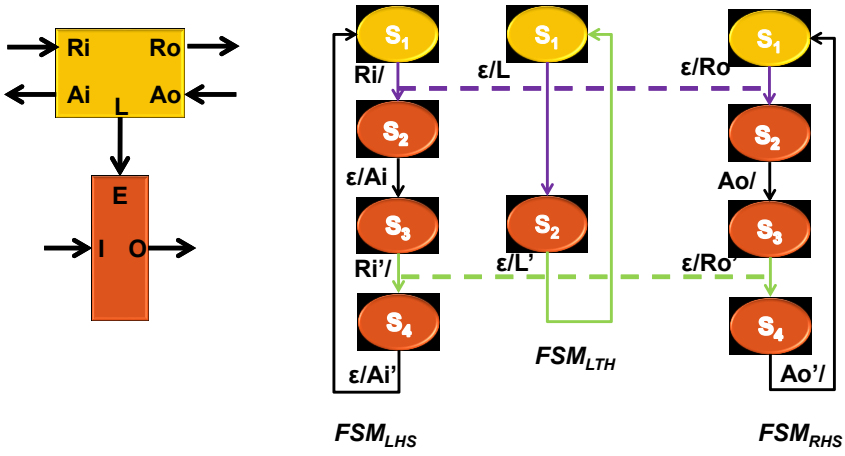


FSM_{RHS}

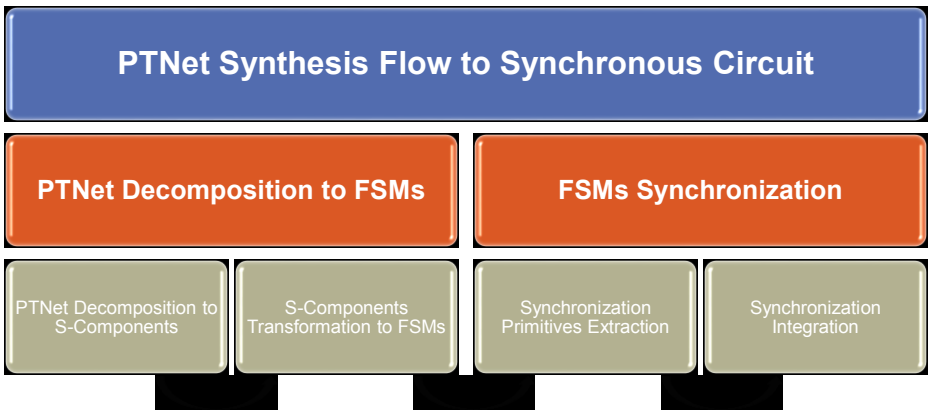


FSM_{LTH}

MSFSMs Example – 4-phase Latch Controller

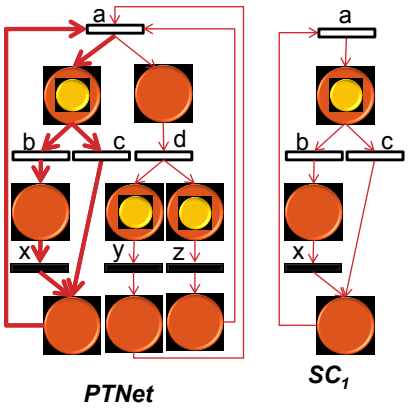


Synchronous Synthesis Flow



PTnet to S-Component Decomposition

- Step 1/4



PTNet to S-Component Decomposition

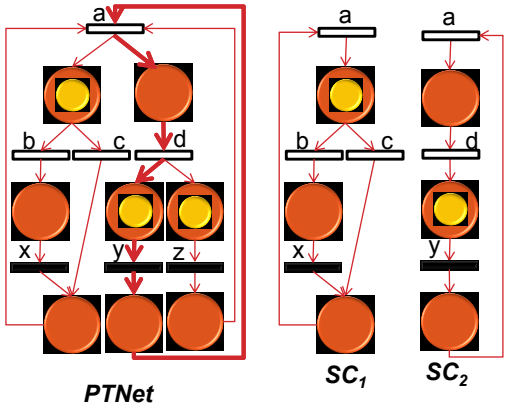
S-Component to FSM Mapping

Synchronization Primitive Extraction

Synchronization Integration

PTnet to S-Component Decomposition

- Step 1/4



PTNet to S-Component Decomposition

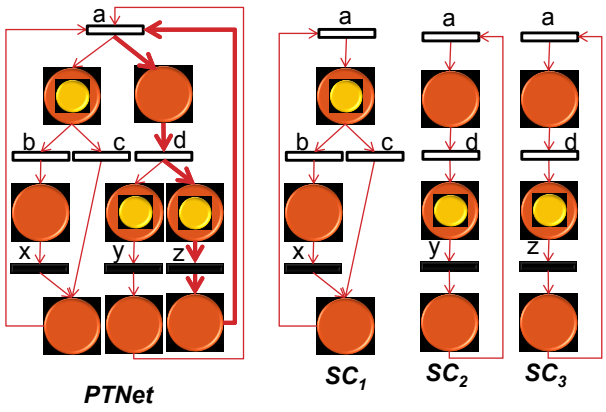
S-Component to FSM Mapping

Synchronization Primitive Extraction

Synchronization Integration

PTnet to S-Component Decomposition

- Step 1/4



PTNet to S-Component Decomposition

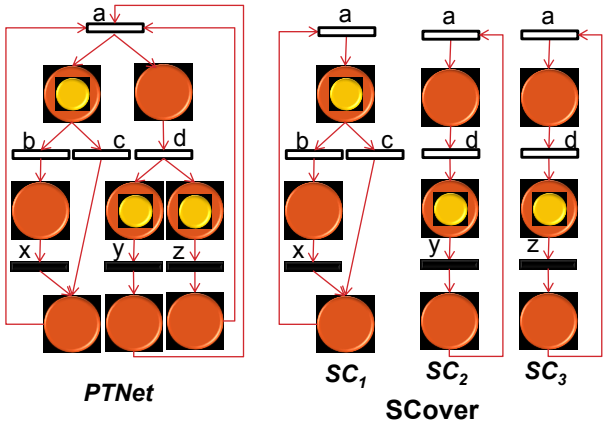
S-Component to FSM Mapping

Synchronization Primitive Extraction

Synchronization Integration

PTnet to S-Component Decomposition

- Step 1/4



PTNet to S-Component Decomposition

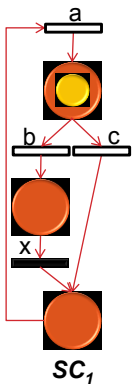
S-Component to FSM Mapping

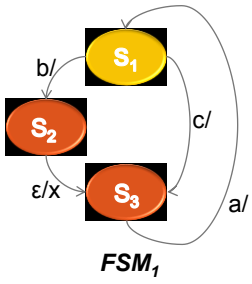
Synchronization Primitive Extraction

Synchronization Integration

S-Component to FSM Mapping

- Step 2/4

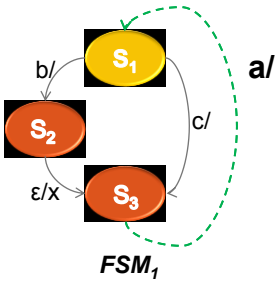


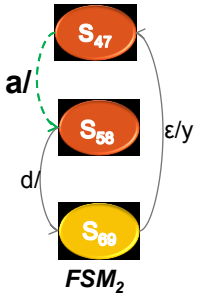


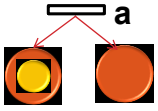
PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration

Synchronisation Primitive Extraction

- Step 3/4



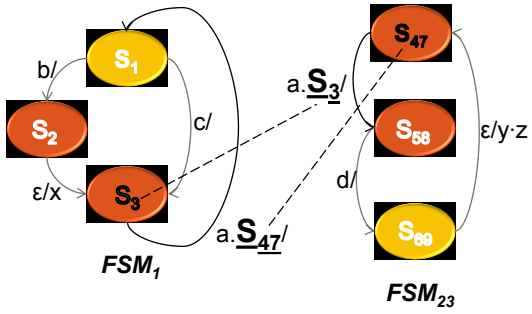




PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration

Synchronisation Integration

- Step 4/4



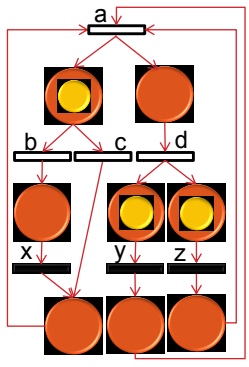
PTNet to S-Component Decomposition

S-Component to FSM Mapping

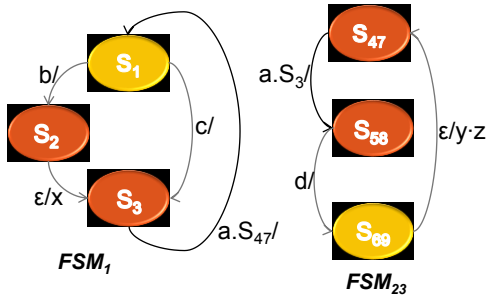
Synchronization Primitive Extraction

Synchronization Integration

Resultant MSFSM Decomposition



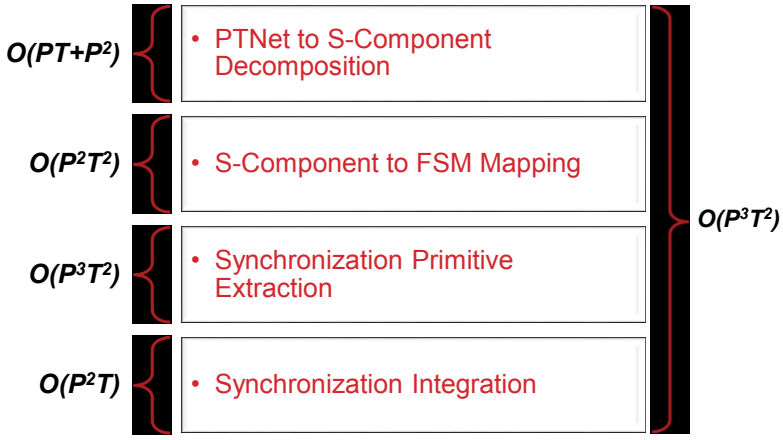
PTNet



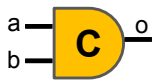
MSFSM

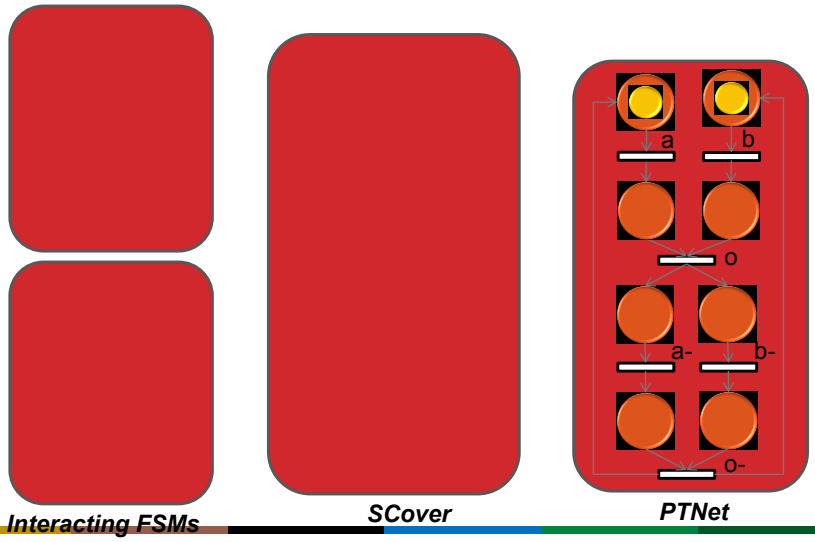
FSMs Synchronization

Complexity Analysis

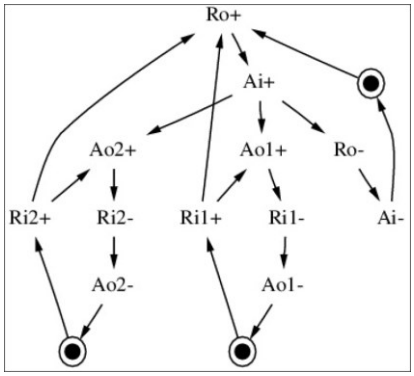


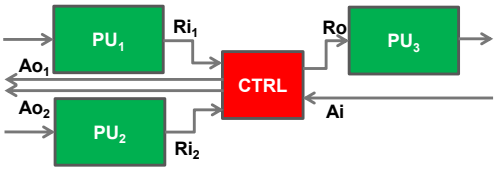
From MSFSMs to PTnets





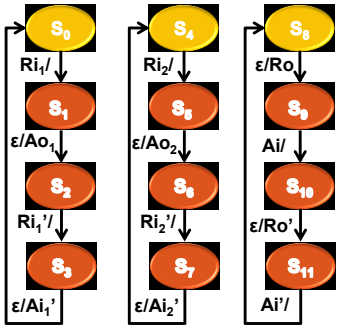
MFSM-based Verification

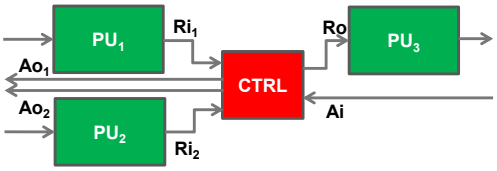




Is the Concurrent System Deadlock-Free?

MFSM-based Verification

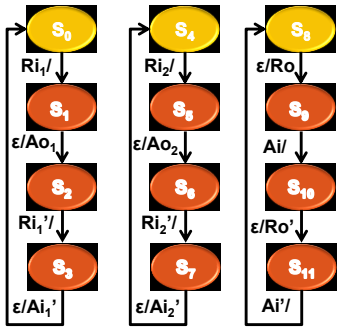




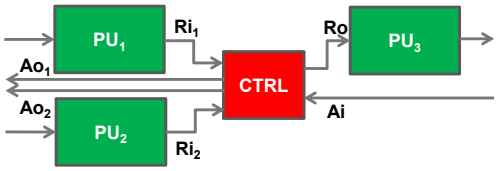
Synchronization Points	
W	*Dependent States*
S8	*S1, S5*
S1	*S10*
S5	*S10*

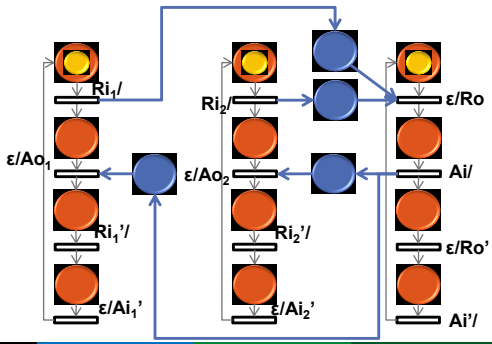
Is the Concurrent System Deadlock-Free?

MFSM-based Verification

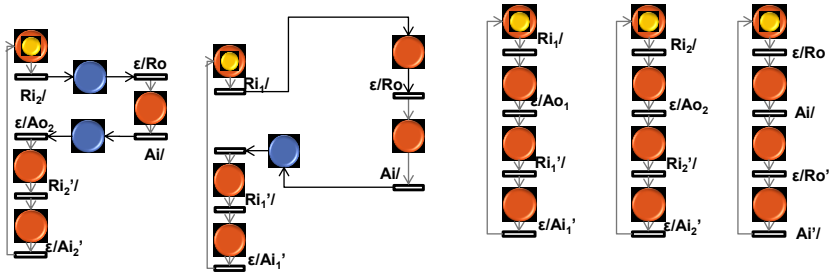


Synchronization Points	
W	*Dependent States*
S8	*S1, S5*
S1	*S10*
S5	*S10*





MFSM-based Verification

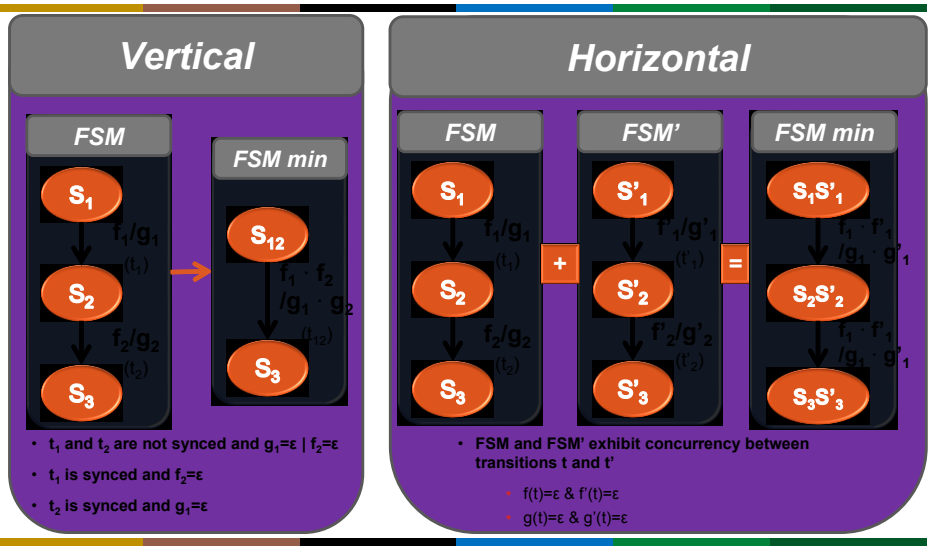


The system is covered by 5 initially marked minimal siphons.

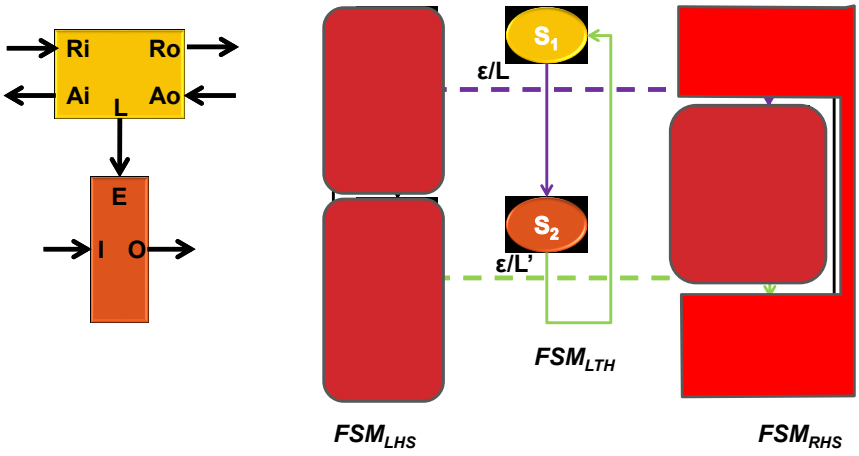
Deadlock-Free

MSFSM Optimisations

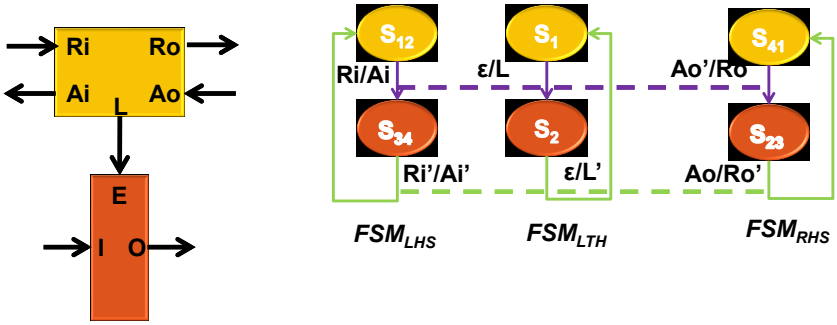
Horizontal and Vertical State Collapsing, State Minimisation



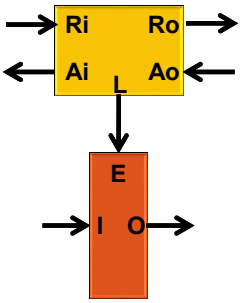
Horizontal and Vertical State Collapsing, State Minimisation

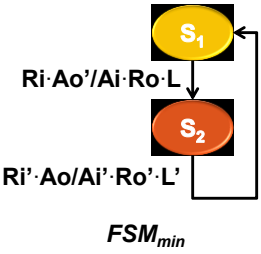


Horizontal and Vertical State Collapsing, State Minimisation



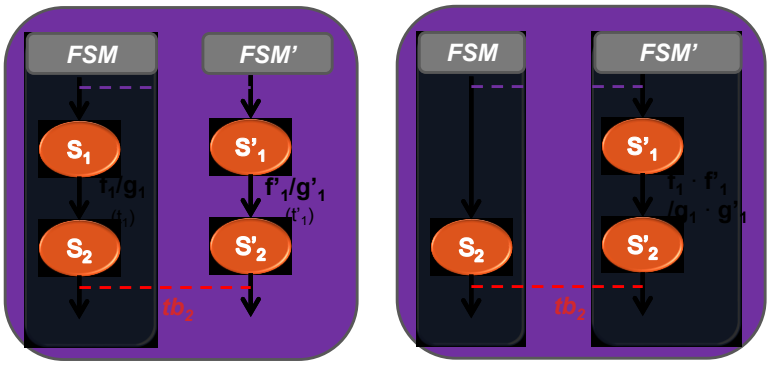
Horizontal and Vertical State Collapsing, State Minimisation





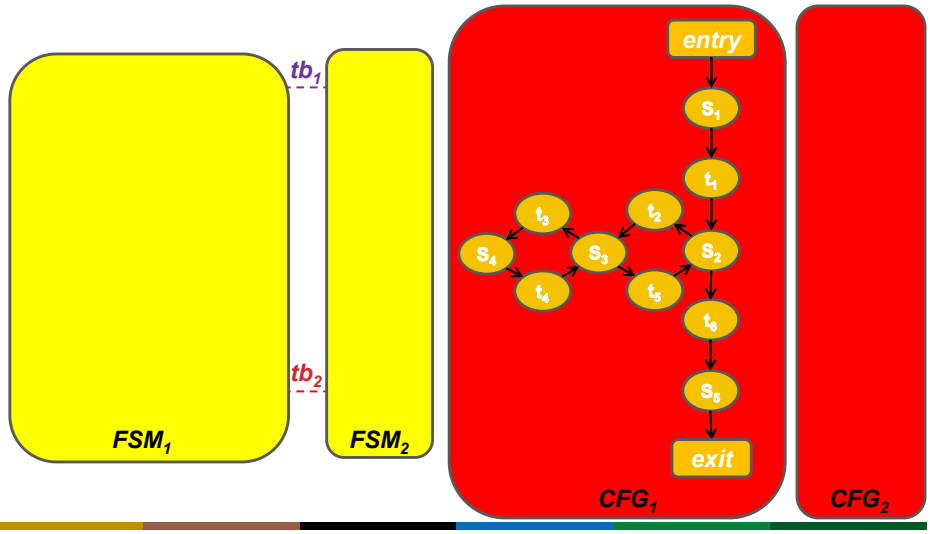
FSM_{min}

The notion of MSFSM Cross(X)-Compatibles

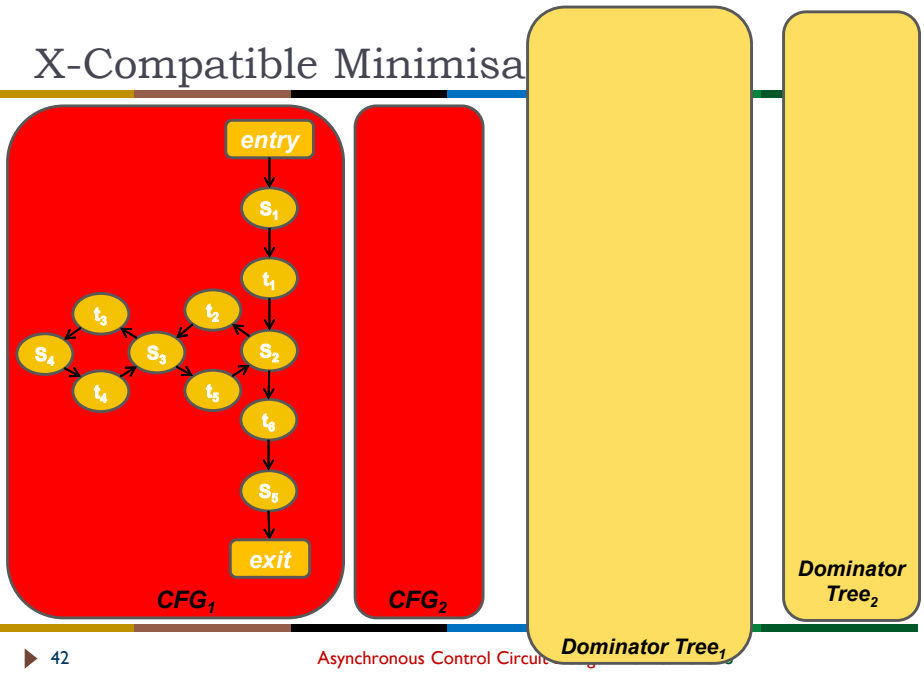


- ***Two transitions are X-Compatible if they are concurrent and mutually inclusive.***
  - *t_1 and t_1' are X-Compatible*

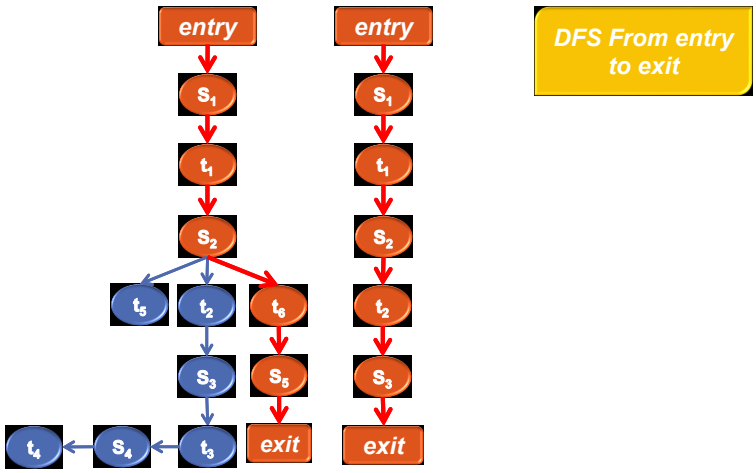
X-Compatible Extraction



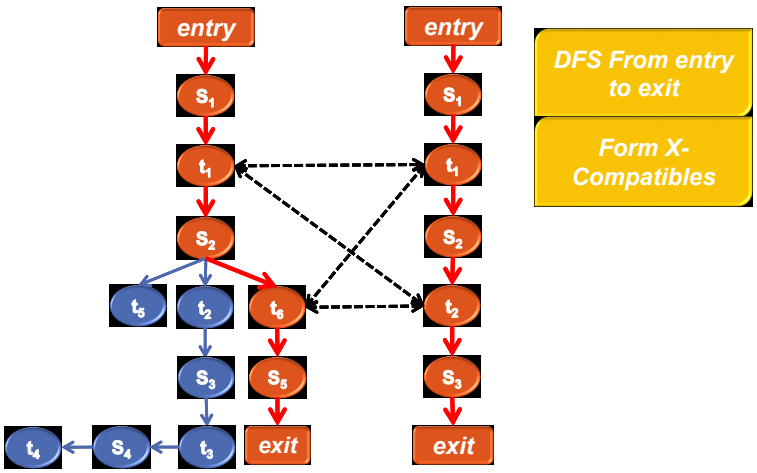
X-Compatible Minimisa



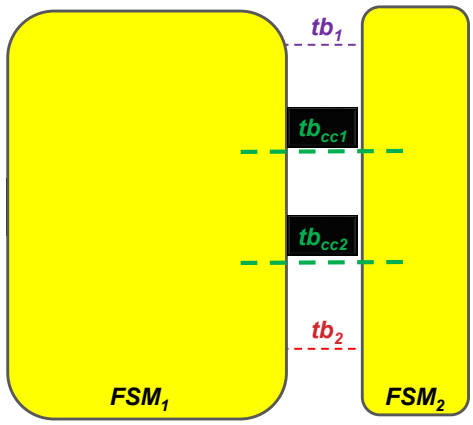
X-Compatible Minimisation



X-Compatible Minimisation

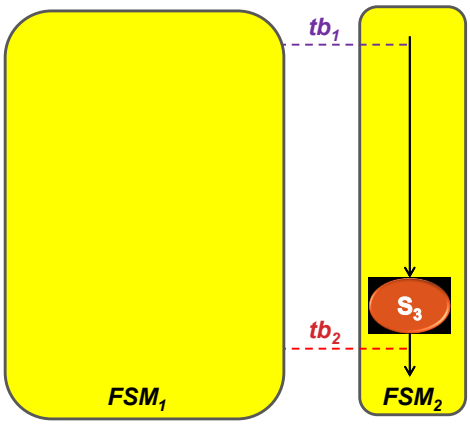


X-Compatible Minimisation



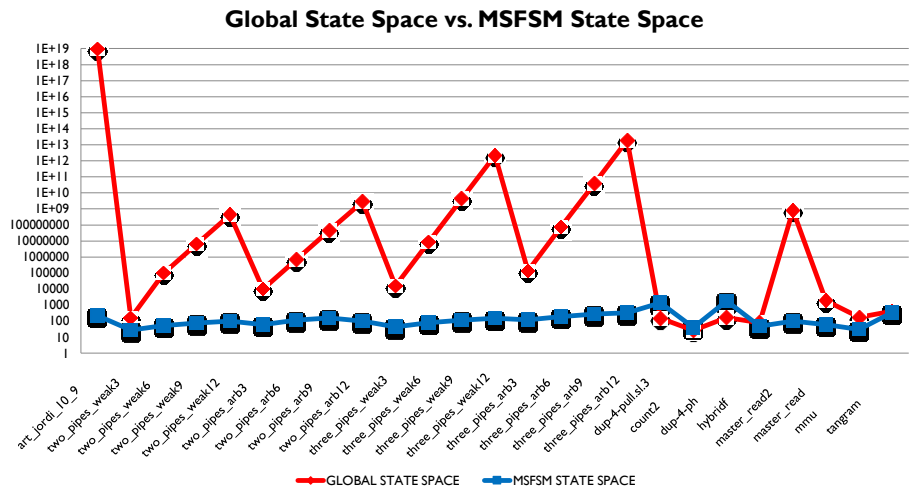
- DFS From entry to exit
- Form X-Compatibles
- Synchronize

X-Compatible Minimisation

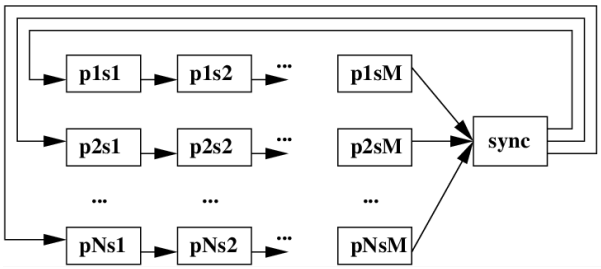


- DFS From entry to exit
- Form X-Compatibles
- Synchronize
- Minimize

Results

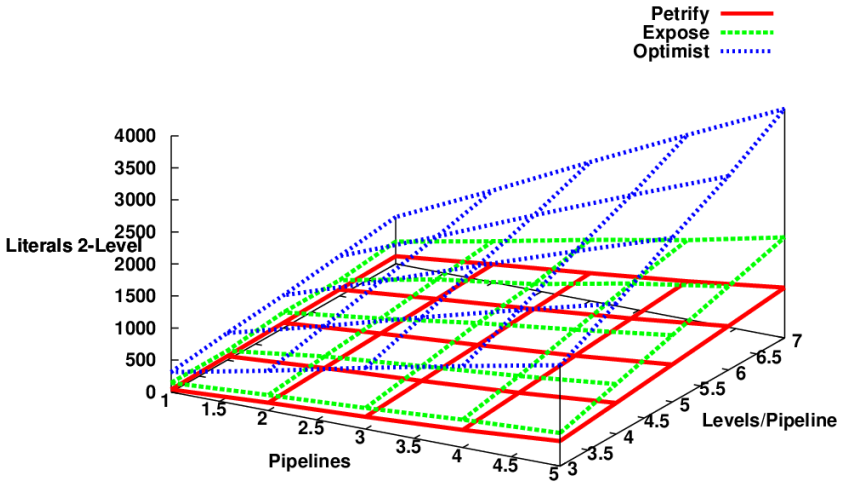


Asynchronous Synthesis Synthetic Benchmark

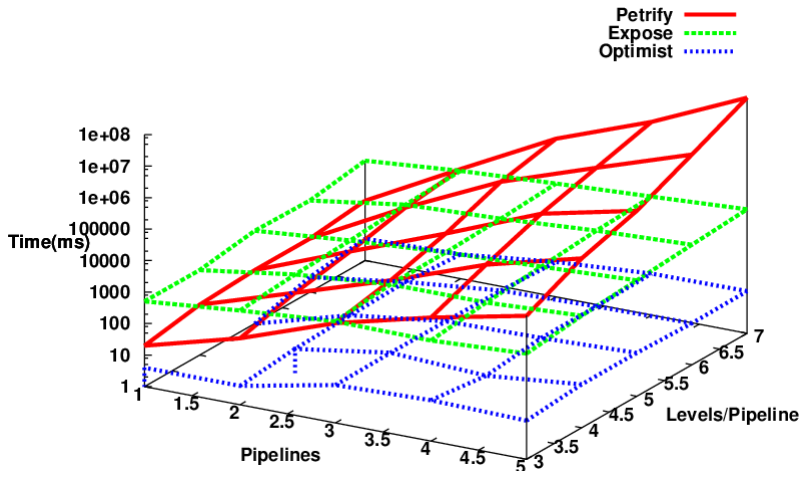


- N Parallel Handshake Controllers
- M Sequential Controllers Per Pipeline
- Synchronized at the M stage
  - Sync+ phase is a join, but Sync- is free ("loosely" synchronised)

Area Analysis



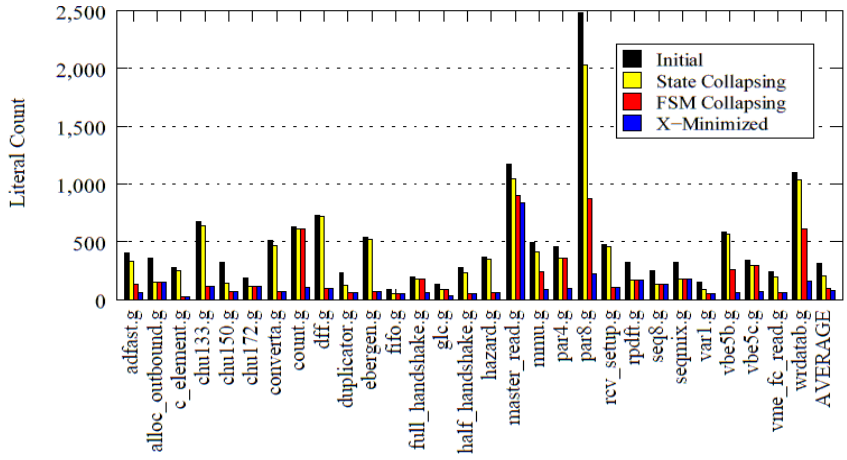
Execution Time



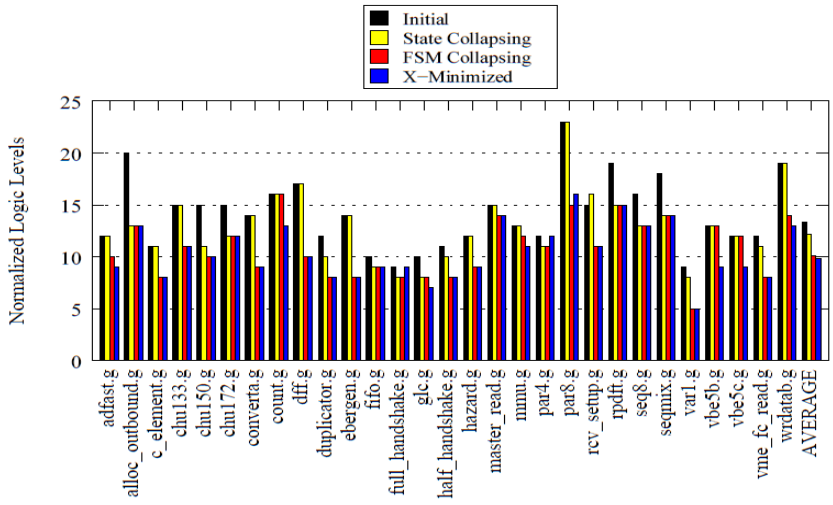
Asynchronous Control Circuit Area

Benchmark	Petrify	Optimist	Expose
alloc-outbound	*66*	*258*	*30*
c3	*12*	*198*	*13*
count2	*Irresolvable CSC*	*302*	*178*
dff	*44*	*304*	*20*
duplicator	*93*	*228*	*111*
full	*44*	*264*	*102*
half	*43*	*198*	*148*
monkey	*N/A*	*1148*	*181*
rpdft	*34*	*214*	*25*
semi-decoupled	*86*	*242*	*208*
vbe6a	*132*	*732*	*237*

Optimisation – Literal Count (LC)



Optimisation – Logic Depth



Future Work

- Logic Synthesis of Mixed Synchronous and Asynchronous Circuits
  - Unified flow for Concurrent Control
- Examine Concurrency / Area Trade-Off possible by X-Compatible State/Segment Optimization
- Support more Asynchronous Timing Models
  - Speed Independent (SI)
    - State Encoding Requirements (Race-Free, Hazard-Free)