LOGIC SYNTHESIS OF CONCURRENT CONTROLLER SPECIFICATIONS

PAVLOS M. MATTHEAKIS

SUPERVISOR: CHRISTOS P. SOTIRIOU
OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• Optimization
• Results
• Conclusions and Future Work
STATE EXPLOSION

• Logic Synthesis requires specification’s complete state space
• State space’s size is exponential compared to the specification
SPECIFICATION AND STATE CONFLUENCE

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

Concurrent Specification

State Space = 4 states

Concurrent Specification + (T2,P4)

Infinite State Space

Concurrent Specification

(P1,P2)

(P3,P2)

(P1,P4)

(P3,P4)

(P1,P2)

(P3,P2,P4)

(P1,P4)

(P3,P4,P4)

(P1,P2,P4)

(P3,P4,P4)

(P1,P4,P4)
DECOMPOSABILITY

Component #1
P=9
State Space=27

Component #2
P=6
State Space=6

Complete Specification
P=15
State Space=33

• Concurrent Specification Decomposition is Unpredictable as it is Evaluated with Specification Level Metrics, e.g. number of places.
IMPORTANCE

State Explosion
- Logic Synthesis
- HLS – VLSI Programming
- Formal Verification

Specification and State Space Confluence
- Complex Synthesis Flows
- Digital Design

Decomposability
- Logic Synthesis
- HLS – VLSI Programming
OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• Optimization
• Results
• Conclusions and Future Work
PTNET OPERATION

FSM

PTNet
PTNET OPERATION

FSM

PTNet
PTNET OPERATION

FSM

PTNet

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z

a/x
b/y
c/z
CONTROL MODELS EXPRESSABILITY

**Algebra**

\[ S_0 = S_1.b + S_2.a + S_0.(a+b)' \]
\[ S_1 = S_0.a + S_1.b' \]
\[ S_2 = S_0.b + S_2.a' \]
\[ x = S_0.a + S_1 + S_2.a \]

**FSM**

**Inter. FSMs**

**PTNet**

<table>
<thead>
<tr>
<th>Control Model</th>
<th>State Space</th>
<th>Choice</th>
<th>Concurrency</th>
<th>Synchronization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algebra</td>
<td>Implicit</td>
<td>Implicit</td>
<td>Implicit</td>
<td>Implicit</td>
</tr>
<tr>
<td>FSM</td>
<td>Explicit</td>
<td>Explicit</td>
<td>Implicit</td>
<td>Implicit</td>
</tr>
<tr>
<td>Inter. FSMs</td>
<td>Implicit</td>
<td>Explicit</td>
<td>Explicit</td>
<td>Implicit</td>
</tr>
<tr>
<td>PTNet</td>
<td>Implicit</td>
<td>Explicit</td>
<td>Explicit</td>
<td>Explicit</td>
</tr>
</tbody>
</table>
CONTROL MODELS
IMPLEMENTABILITY TO
CLOCKED LOGIC

Abstraction

Implementability

PTNets
- This Work

Interacting FSMs
- FSM Composition and Decomposition (VIS)
- Input Don’t Cares and Permissible Behaviors (S1S)

FSMs
- State Minimization (SIS)
- Encoding (NOVA, MUSTANG)

Boolean Logic
- 2-Level Exact (Q. McCluskey Method) and Heuristic (Espresso)
- Multilevel Factoring, Extraction, etc. (SIS)
CONTROL MODELS IMPLEMENTABILITY TO SELF-TIMED LOGIC

(A,BM)FSM Synthesis\(^1\)
- Conventional FSM Heuristics
- Exponential Concurrent Specifications

PTNet Synthesis\(^2\)
- Compact Concurrent Specifications
- Exponential Synthesis Time

Decomposition\(^4\)
- Lower Synthesis Time
- Infeasible Implementation
- Worst Case Exponential

Structural Synthesis\(^5\)
- Lower Synthesis Time
- Worst Case Exponential

Direct Mapping\(^3\)
- Linear Synthesis Time
- Suboptimal Results

---

\(^1\) K. Y. Yun and D. L. Dill, "Automatic synthesis of extended burst-mode circuits", IEEE TCAD, 1999


\(^3\) D. Sokolov et al., “Direct Mapping of Low-Latency Asynchronous Controllers from STGs”, IEEE TCAD, 2007


OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• Optimization
• Results
• Conclusions and Future Work
MULTIPLE SYNCHRONIZED FSMS

Novel Model for Concurrent Control Specifications

- Parallel Tasks Described with FSMs
- Synchronization explicitly described
  - Transition Barriers, Wait States
- Polynomial Synthesis and Verification Paths
**MSFSMS SYNCHRONIZATION PRIMITIVES**

### Wait State

- FSM1 moves from $S_w$ to $S_{wn}$ if FSM2 is at $S_d$

### Transition Barrier

- FSM1 and FSM2 move to their next states when both $t_1$ and $t_2$ are activated
MSFSMS EXAMPLE: 4-PHASE LATCH CTRL
MSFSMS EXAMPLE: 4-PHASE LATCH CTRL
MSFSMS EXAMPLE: 4-PHASE LATCH CTRL

PTNet

State Graph (FSM)
MSFSMS EXAMPLE: 4-PHASE LATCH CTRL
MSFSMS EXAMPLE:
4-PHASE LATCH CTRL

FSM_{LHS}

FSM_{LTH}

FSM_{RHS}
OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• Optimization
• Results
• Conclusions and Future Work
OVERVIEW

PTNet Synthesis Flow to Synchronous Circuit

PTNet Decomposition to FSMs

- PTNet Decomposition to S-Components
- S-Components Transformation to FSMs

FSMs Synchronization

- Synchronization Primitives Extraction
- Synchronization Integration
STEPS (1/4)

PTNet

SC₁

PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
STEPS (1/4)

PTNet

SC₁

SC₂

PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
STEPS (1/4)

PTNet

SC₁

SC₂

SC₃

PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
STEPS (1/4)

PTNet

SCover

PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
STEPS (2/4)

PTNet to S-Component Decomposition
S-Component to FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
STEPS (3/4)

PTNet to S-Component Decomposition

S-Component to FSM Mapping

Synchronization Primitive Extraction

Synchronization Integration
STEPS (4/4)

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

FSM

1

a.S

47/

b/

c/

ε/x

S1

S2

S3

S4/

y∙z

ε/y·z

d/

MSFSM

FSMs

Synchronization
COMPLEXITY

- $O(PT + P^2)$
  - PTNet to S-Component Decomposition

- $O(P^2T^2)$
  - S-Component to FSM Mapping

- $O(P^3T^2)$
  - Synchronization Primitive Extraction

- $O(P^2T)$
  - Synchronization Integration

$O(P^3T^2)$
OVERVIEW

PTNet Synthesis Flow to Asynchronous Circuit

PTNet Decomposition to BMFSMs

BMFSMs Synchronization

PTNet Decomposition to S-Components

S-Components Transformation to BMFSMs

Synchronization Primitives Extraction

Synchronization Integration
1. Choose a signal (R_i) and form all the consecutive transition pairs (R_i+, R_i-) (R_i-, R_i+)
1. Choose a signal ($R_i$) and form all the consecutive transition pairs ($R_i^+, R_i^-$) ($R_i^-, R_i^+$).

2. For each pair ($R_i^+, R_i^-$),
   (i) remove all concurrent PTNet components
   (ii) DFS to connect pair’s transitions

PTNet to S-Component Decomposition
S-Component to BM-FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
**STEPS 1/3**

1. Choose a signal (Ri) and form all the consecutive transition pairs (Ri+, Ri-) (Ri-, Ri+)

2. For each pair (Ri+, Ri-)
   (i) remove all concurrent PTNet components
   (ii) DFS to connect pair’s transitions

3. Remove all unmarked PTNet components and their corresponding arcs
STEPS 1/3

PTNet

PTNet to S-Component Decomposition
S-Component to BM-FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
STEPS 2/3

PTNet to S-Component Decomposition
S-Component to BM-FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
1. Form the Signal Support for each BMFSM

<table>
<thead>
<tr>
<th>$FSM1$</th>
<th>$FSM2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Ai$</td>
<td>$Ro$</td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>1</td>
</tr>
<tr>
<td>S4</td>
<td>1</td>
</tr>
</tbody>
</table>

PTNet to S-Component Decomposition
S-Component to BM-FSM Mapping
Synchronization Primitive Extraction
Synchronization Integration
1. Form the Signal Support for each BMFSM

2. Insert Synchronizing BMFSMs

<table>
<thead>
<tr>
<th>FSM1</th>
<th>FSM2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ai</td>
<td>Ro</td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>1</td>
</tr>
<tr>
<td>S4</td>
<td>1</td>
</tr>
</tbody>
</table>
1. Form the Signal Support for each BMFSM

2. Insert Synchronizing BMFSMs

3. Insert Synchronization Logic

STEPS 3/3

FSM1

<table>
<thead>
<tr>
<th></th>
<th>Ai</th>
<th>Ro</th>
<th>S1</th>
<th>S3</th>
<th>sync1</th>
<th>sync2</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

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

\[ O(P\cdot T^3 + P^2 \cdot T^2) \]
- PTNet to S-Component Decomposition

\[ O(P^2 \cdot T^2) \]
- S-Component to BM-FSM Mapping

\[ O(P^3 \cdot T^2) \]
- Synchronization Primitive Extraction

\[ O(P^2 \cdot T) \]
- Synchronization Integration

\[ O(P^3 \cdot T^2 + P \cdot T^3) \]
OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• Optimization
• Results
• Conclusions and Future Work
INTERACTING FSMS TO PTNETS

Interacting FSMs

\[ S_1 = 1 \]
\[ a.S_3/o \rightarrow S_2 = 1 \]
\[ a'.S_4/o' \]

\[ FSM_a \]

\[ S_3 = 1 \]
\[ b.S_1/ \rightarrow S_4 = 1 \]
\[ b'.S_2/ \]

\[ FSM_b \]

SCover

\[ SC_a \]
\[ a- \]
\[ o \]
\[ a- \]
\[ b- \]
\[ SC_b \]

PTNet

\[ PTNet \]
\[ C \]
\[ a \]
\[ b \]
\[ o \]
MSFSM VERIFICATION

Synchronization Points

<table>
<thead>
<tr>
<th>W</th>
<th>Dependent States</th>
</tr>
</thead>
<tbody>
<tr>
<td>S8</td>
<td>S1, S5</td>
</tr>
<tr>
<td>S1</td>
<td>S10</td>
</tr>
<tr>
<td>S5</td>
<td>S10</td>
</tr>
</tbody>
</table>

Is the Concurrent System Deadlock-Free?
MSFSM VERIFICATION

Synchronization Points

<table>
<thead>
<tr>
<th>W</th>
<th>Dependent States</th>
</tr>
</thead>
<tbody>
<tr>
<td>S8</td>
<td>$S_1$, $S_5$</td>
</tr>
<tr>
<td>S1</td>
<td>$S_{10}$</td>
</tr>
<tr>
<td>S5</td>
<td>$S_{10}$</td>
</tr>
</tbody>
</table>
The system is covered by 5 initially marked minimal siphons.

Deadlock-Free
OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• **Optimization**
• Results
• Conclusions and Future Work
**HORIZONTAL AND VERTICAL COLLAPSING**

**Vertical**

- **FSM**
  - $S_1$ → $S_2$ (with $f_1/g_1$ (t_1))
  - $S_2$ → $S_3$ (with $f_2/g_2$ (t_2))

- **FSM min**
  - $S_1$ → $S_{12}$
  - $S_{12}$ → $S_3$ (with $f_1 \cdot f_2/g_1 \cdot g_2$ (t_{12}))

- $t_1$ and $t_2$ are not synced and $g_1=\varepsilon$ | $f_2=\varepsilon$
- $t_1$ is synced and $f_2=\varepsilon$
- $t_2$ is synced and $g_1=\varepsilon$

**Horizontal**

- **FSM**
  - $S_1$ → $S_2$ (with $f/g$ (t_1))

- **FSM’**
  - $S'_1$ → $S'_2$ (with $f'/g'$ (t'_1))

- **FSM min**
  - $S_1S'_1$ → $S_2S'_2$

- FSM and FSM’ exhibit concurrency between transitions t and t’
  - $f(t)=\varepsilon$ & $f'(t)=\varepsilon$
  - $g(t)=\varepsilon$ & $g'(t)=\varepsilon$
HORIZONTAL AND VERTICAL COLLAPSING

\[ \text{HFSM} \]

\[ \text{LHS} \]

\[ \text{Ri} / \varepsilon / \text{Ai} \]

\[ \text{Ri'} / \varepsilon / \text{Ai'} \]

\[ \text{L} / \varepsilon / \text{L} \]

\[ \varepsilon / \text{Ro} \]

\[ \varepsilon / \text{Ro'} \]

\[ \text{FSM}_{LTH} \]

\[ \text{FSM}_{LHS} \]

\[ \text{FSM}_{RHS} \]
HORIZONTAL AND VERTICAL COLLAPSING

\[ \text{FSM}_{LHS} \quad \text{FSM}_{LTH} \quad \text{FSM}_{RHS} \]
HORIZONTAL AND VERTICAL COLLAPSING

$FSM_{min}$
Two transitions are X-Compatible if they are concurrent and mutually inclusive.

- $t_1$ and $t'_1$ are X-Compatibility
X-COMPATIBLES
EXTRACTION

FSM₁

FSM₂

CFG₁

CFG₂
X-COMPATIBLES MINIMIZATION

$\text{CFG}_1$

$\text{CFG}_2$

$\text{Dominator Tree}_1$

$\text{Dominator Tree}_2$
X-COMPATIBLES
MINIMIZATION

DFS From entry to exit
X-COMPATIBLES MINIMIZATION

DFS From entry to exit
Form X-Compatibles
X-COMPATIBLES MINIMIZATION

DFS From entry to exit
Form X-Compatibles
Synchronize

FSM₁

S₁
S₂
S₃
S₄
S₅

(t₁)
(t₂)
(t₃)
(t₄)
(t₅)
(t₆)

FSM₂

S₆
S₇
S₈

(t₇)
(t₈)

Synchronization

tb₁

tb₂

tb_{cc₁}

tb_{cc₂}
X-COMPATIBLES
MINIMIZATION

DFS From entry
to exit

Form X-
Compatibles

Synchronize

Minimize

\begin{align*}
&\text{tb}_1 \\
&\text{tb}_2
\end{align*}

\text{S}_1 \quad \text{S}_2 \quad \text{S}_3 \quad \text{S}_4 \quad \text{S}_5

\text{FSM}_1 \quad \text{FSM}_2
OVERVIEW

- Motivation
- Background
- MSFSMs
- Synthesis
- Verification
- Optimization
- Results
- Conclusions and Future Work
Transforming a PTNet to MSFSMs and then to a synthesizable equivalent does not suffer from the state explosion issue.
N Parallel Handshake Controllers
M Sequential Controllers Per Pipeline
Synchronized at the M stage
Exposé unveils the solution space between the direct mapping approaches and the ones which require the complete state space.
• **Expose** execution time scales with the size of the initial specification
## CONTROL CIRCUITS AREA

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Petrify</th>
<th>Optimist</th>
<th>Expose</th>
</tr>
</thead>
<tbody>
<tr>
<td>alloc-outbound</td>
<td>66</td>
<td>258</td>
<td>30</td>
</tr>
<tr>
<td>c3</td>
<td>12</td>
<td>198</td>
<td>13</td>
</tr>
<tr>
<td>count2</td>
<td>Irresolvable CSC</td>
<td>302</td>
<td>178</td>
</tr>
<tr>
<td>dff</td>
<td>44</td>
<td>304</td>
<td>20</td>
</tr>
<tr>
<td>duplicator</td>
<td>93</td>
<td>228</td>
<td>111</td>
</tr>
<tr>
<td>full</td>
<td>44</td>
<td>264</td>
<td>102</td>
</tr>
<tr>
<td>half</td>
<td>43</td>
<td>198</td>
<td>148</td>
</tr>
<tr>
<td>monkey</td>
<td>N/A</td>
<td>1148</td>
<td>181</td>
</tr>
<tr>
<td>rpdf</td>
<td>34</td>
<td>214</td>
<td>25</td>
</tr>
<tr>
<td>semi-decoupled</td>
<td>86</td>
<td>242</td>
<td>208</td>
</tr>
<tr>
<td>vbe6a</td>
<td>132</td>
<td>732</td>
<td>237</td>
</tr>
</tbody>
</table>
• Applying the introduced transformations at the MSFSM level predictably reduces area.
• Applying the introduced transformations at the MSFSM level typically increases performance.
OVERVIEW

• Motivation
• Background
• MSFSMs
• Synthesis
• Verification
• Optimization
• Results
• *Conclusions and Future Work*
CONCLUSIONS

Novel Control Specification Model, MSFSMs
• Key for Concurrent Specifications Synthesis and Verification

Logic Synthesis Tool, Expose, Targeting Synchronous and Asynchronous Logic

Optimization Operations with Predictable QoR
FUTURE WORK

Logic Synthesis of Mixed Synchronous and Asynchronous Circuits

Examination of the Concurrency / Area Trade-Off Unveiled by the X-Compatibles Optimization

Expansion of the supported Asynchronous Timing Models
• Speed Independent