Clock Skew Scheduling with Delay Padding for Prescribed Skew Domains

Chuan Lin and Hai Zhou

Jan 25, 2007

MAGMA®

NORTHWESTERN UNIVERSITY

Outline

- Useful skew scheduling
- Motivation and problem formulation
- Algorithms
- Experimental results
- Conclusions

Introduction

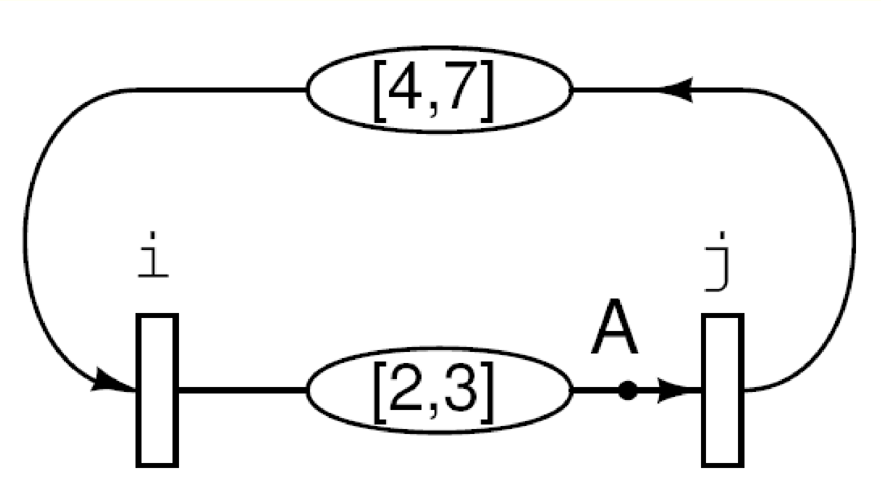
- Clock skew
  - Difference in clock arrival times due to clock distribution delays
- Zero skew [Tsay ICCAD91]
- Skew scheduling [Fishburn TC90]
  - Assigns skews to FFs to improve performance
  - Arbitrary skews cannot be implemented reliably

Introduction

- Multi-domain skew scheduling
  [Ravindran *et al*. ICCAD03]
  - Given number of domains
  - Algorithms to compute optimal schedule

- No control on the domain distribution
- Not consider the flexibility of delay padding
  [Shenoy *et al*. ICCAD93]
  - Insert extra delays on short paths
  - Keep delays on long paths unchanged

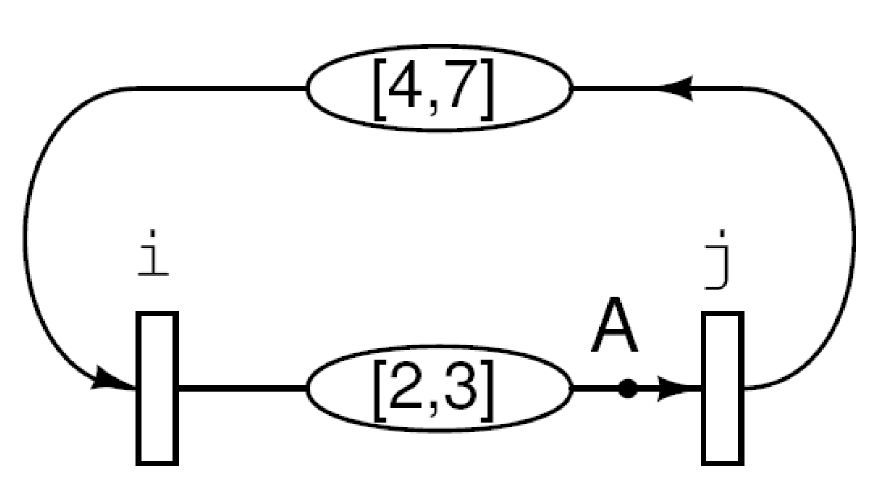
Motivation Example



$$skew(i) + 3 - T \leq skew(j)$$
$$skew(j) + 7 - T \leq skew(i)$$
$$skew(i) + 2 \geq skew(j)$$
$$skew(j) + 4 \geq skew(i)$$

- Case 1: *skew(i)=skew(j),* then $T \geqslant 7$

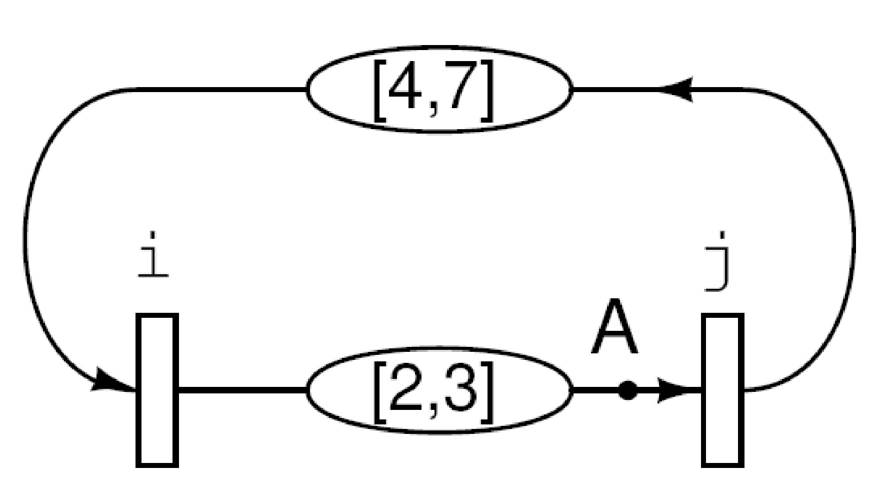
Motivation Example



$$skew(i) + 3 - T \le skew(j)$$
$$skew(j) + 7 - T \le skew(i)$$
$$skew(i) + 2 \ge skew(j)$$
$$skew(j) + 4 \ge skew(i)$$

- Case 2: *skew(i)=T/2, skew(j)=0,* then $6 \leqslant T \leqslant 8$
  - The mini *T* is reduced to 6 by skew scheduling

Motivation Example



$$skew(i) + 3 - T \leq skew(j)$$
$$skew(j) + 7 - T \leq skew(i)$$
$$skew(i) + 2 \geq skew(j)$$
$$skew(j) + 4 \geq skew(i)$$

- Case 3: *skew(i)=0, skew(j)=T/2,* then infeasible
  - If delay 5 is inserted at A, then T = 14

Motivation Example

- Observations
  - Skew scheduling may help to reduce period
  - Caution should be taken to avoid infeasibility
  - Delay padding can remedy a skew schedule

Problem Formulation

[Optimal Skew Scheduling]

- Given
  - A sequential circuit
  - A finite set of prescribed skew domains
- Find
  - A domain assignment for each FF
  - Setup & hold constraints satisfied with padding
  - Clock period is minimized

Notations

- N skew domains
  - $s_0T, s_1T, \ldots, s_NT$ w.r.t. the clock period T
  - $0 = s_0 < s_1 < \ldots < s_N < 1$
- Setup period X, hold period H
- A directed graph *G=(V,E)*

  V: gates and FFs

  E: interconnects

Notations

- Each gate *v :* max delay *D(v)*, min delay *d(v)*
- Each combinational path *p* from FF *i* to FF *j*
  - max delay *D(p)*, min delay *d(p)* without padding
  - max delay *ᛕ(p)*, min delay *(p)* with padding

$$D(i,j) \triangleq \max_{p \in G: i \rightsquigarrow j} D(p), \qquad \Delta(i,j) \triangleq \max_{p \in G: i \rightsquigarrow j} \Delta(p)$$

$$d(i,j) \triangleq \min_{p \in G: i \rightsquigarrow j} d(p), \qquad \delta(i,j) \triangleq \min_{p \in G: i \rightsquigarrow j} \delta(p)$$

Constraints

- Feasibility of a given period *T*
  - Domain index *l(i)* of FF *i*
  - Setup constraints
  - Hold constraints

$$0 \leq l(i) \leq N-1, \quad \forall i \in V_t$$
$$s_{l(i)}T + \Delta(i,j) - s_{l(j)}T \leq T - X(j), \quad \forall (i,j) \in E_t$$
$$s_{l(i)}T + \delta(i,j) - s_{l(j)}T \geq H(j), \quad \forall (i,j) \in E_t$$

Algorithm Overview

- Find an optimal domain assignment
  - Minimizes period under setup constraints only
- Compute an efficient padding solution by network flow
  - There is always a padding solution under the minimum period

Lower Bound for Feasible Period

- For any combinational path *p*

$$s_{l(i)}T + \Delta(p) - s_{l(j)}T \leq T - X(j)$$
$$s_{l(i)}T + \delta(p) - s_{l(j)}T \geq H(j)$$

- Subtracting the first by the second yields

  T-X(j)-H(j) ⩾ ʎ(p)-(p) = D(p)-d(p)

Lower Bound for Feasible Period

- *(**Lemma**) A feasible period T must satisfy*

$$T \geq T_{lb} \triangleq \max_{(i,j)\in E_t, p\in E: i \leadsto j} \big(D(p) - d(p) + X(j) + H(j)\big)$$

- We can compute T_{lb} by a longest path computation in O(|E|+|V|log|V|) time

Minimum Period under Setup Constraints

$$0 \le l(i) \le N-1, \quad \forall i \in V_t$$
$$s_{l(i)}T + D(i,j) - s_{l(j)}T \le T - X(j), \quad \forall (i,j) \in E_t$$

- If we know an optimal domain assignment ***l****
  - $T_S = max\ (D(i,j)+X(j))\ /\ (1+s_{l^*(i)}-s_{l^*(j)})$
  - Optimal period $T^* = max\ (T_{lb}, T_S)$
- But, how to compute ***l**** ?

Algorithm to Compute l^*

- Start with zero domain for all FFs
  - $l(i) = 0$, for all i
- $T = max\,(D(i,j)+X(j))\,/\,(1+s_{l(i)}-s_{l(j)})$
- If $T \leqslant T_{lb}$, the current l is an l^*
- Else, suppose (x,y) is the edge determining T
  - Show that $s_{l^*(y)}-s_{l^*(x)} > s_{l(y)}-s_{l(x)}$
  - Increase $l(y)$ by 1
- Iterate until there exists some i s.t. $l(i) > N$

Algorithm to Compute l^*

- *(**Lemma**)* $l \leqslant l^*$ *is kept before we reach an* l^*

- The algorithm terminates in $O(N|V_t|B_t \log|E_t|)$ with an optimal l^* and the minimal period T^* under setup constraints only

Padding for Hold Constraints

- Is there always a padding solution under *l** and *T**?
  - Yes. (Shenoy *et al*. [ICCAD93])
- Minimum padding is LP

$$
\begin{aligned}
MP: \quad & \text{Minimize} \sum_{(u,v)\in G_c} p(u,v) \\
a(i) &= s_{l^*(i)} T^*, \quad \forall i \in PI \\
a(v) &\leq a(u) + d(v) + w(u,v) + p(u,v), \quad \forall (u,v) \in G_c \\
a(j) &\geq H(j) + s_{l^*(j)} T^*, \quad \forall j \in PO \\
A(i) &= s_{l^*(i)} T^*, \quad \forall i \in PI \\
A(v) &\geq A(u) + D(v) + w(u,v) + p(u,v), \quad \forall (u,v) \in G_c \\
A(j) &\leq T^* - X(j) + s_{l^*(j)} T^*, \quad \forall j \in PO \\
p(u,v) &\geq 0, \quad \forall (u,v) \in G_c
\end{aligned}
$$

Network Flow Algorithm for Padding

- Find a “good” padding in much less time than LP
  - Find a feasible padding such that the setup slack is zero on each edge
    - Min-cost flow problem
    - Any padding less than the obtained padding solution satisfies setup constraints
  - Tighten the obtained padding to meet hold
    - Convex-cost flow problem

Experimental Setup

- ISCAS-89 benchmark
  - ASTRA [Sapatnekar TCAD'96]
  - Assign $D(v) = d(v)$ between 2 and 100
  - $X = 2, H = 2$
- MOSEK solver for min-padding
- 4 evenly distributed skew domains

Experimental Results (Skew Scheduling)

- Reduction in period
  - 16.9% average
- Efficiency
  - 0.59 seconds for the largest test

Circuit	T_{ub}	T^*	impr%	t(sec)
s838	96.0	65.6	31.6%	0.01
s1196	102.0	100.0	2.0%	0.00
s1423	334.0	256.0	23.4%	0.07
s5378	94.0	94.0	0.0%	0.01
s9234	180.0	164.0	8.9%	0.04
s9234.1	180.0	164.0	8.9%	0.05
s13207.1	288.0	272.0	5.6%	0.02
s15850	374.0	213.7	42.9%	0.29
s15850.1	374.0	292.0	21.9%	0.28
s35932	140.0	126.0	10.0%	0.16
s38417	222.0	128.0	42.3%	0.28
s38584	308.0	290.0	5.8%	0.59
arith			16.9%	
geo			18.4%	

Experimental Results (Delay Padding)

- Padding solution is good
  - 1.5X min-padding
  - Amortized area overhead is small
- Efficiency
  - 16X faster than MOSEK solver

Circuit	padding		time(sec)	
	MOSEK	ours	MOSEK	ours
s838	56.8	56.8	0.20	0.01
s1196	25.0	87.0	0.45	0.01
s1423	966.0	1425.4	0.49	0.03
s5378	0.0	0.0	2.65	0.09
s9234	38.0	75.0	5.57	0.37
s9234.1	38.0	75.0	5.72	0.43
s13207.1	546.0	562.0	9.25	0.88
s15850	12441.4	14305.6	11.34	1.28
s15850.1	1862.0	2829.0	13.63	1.49
s35932	15840.0	22896.0	13.21	2.83
s38417	6612.0	8224.0	38.55	3.44
s38584	22932.5	23920.0	19.19	5.14
arith	1	1.53X	15.6X	1
geo	1	1.42X	12.5X	1

Conclusions

- Prescribed skews are a better trade-off between schedule flexibility and clock implementation
- Polynomial-time skew scheduling algorithm
  - Minimizes clock period
  - Provable optimality
  - Practical efficiency
- Delay padding under min-period
  - Existence guaranteed
  - Solved by efficient network flow techniques

Thank You !