A Practical Method for Multi-Domain Clock Skew Optimization

Yanling Zhi[1] Hai Zhou[2,1] Xuan Zeng[1]

[1]State Key Lab. of ASIC & System, Fudan University, China

[2]Department of EECS, Nowthwestern University, U.S.A.

16th Asia and South Pacific Design Automation Conference

Outline

Outline

Conventional Clock Skew Scheduling

Without clock skew scheduling

- Performance determined by the longest combinational path.

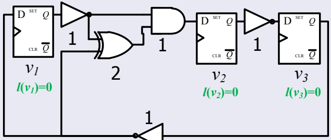

Figure: $T_{\min} = 4$

With clock skew scheduling

- "Steal" time from paths with larger slacks and bestow it to more critical ones.

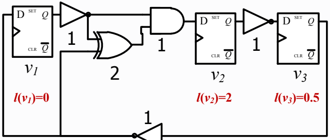

Figure: $T_{\min} = 2.5$

Conventional Clock Skew Scheduling

Without clock skew scheduling

- Performance determined by the longest combinational path.

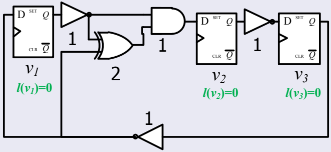

Figure: $T_{\text{min}} = 4$

With clock skew scheduling

- "Steal" time from paths with larger slacks and bestow it to more critical ones.

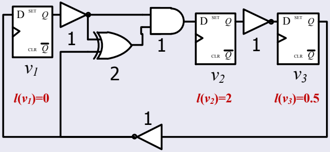

Figure: $T_{\text{min}} = 2.5$

Multi-Domain Clock Skew Scheduling

Conventional clock skew scheduling

- Impractical in reliably implementing a large set of arbitrary clock latencies.

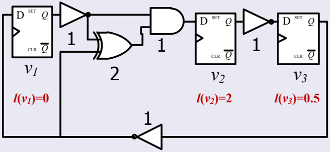

Figure: $T_{\min} = 2.5$

Multi-domain clock skew scheduling

- Overcome the implementation difficulty by constraining the number of possible clock latencies.

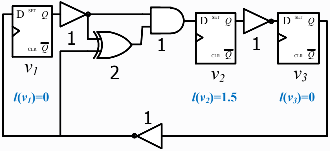

Figure: $T_{\min} = 2.5$

Multi-Domain Clock Skew Scheduling

Conventional clock skew scheduling

- Impractical in reliably implementing a large set of arbitrary clock latencies.

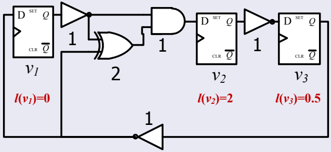

Figure: $T_{\min} = 2.5$

Multi-domain clock skew scheduling

- Overcome the implementation difficulty by constraining the number of possible clock latencies.

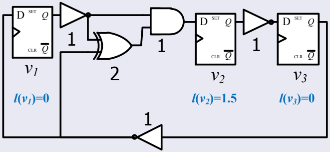

Figure: $T_{\min} = 2.5$

Previous Works

Ravindran et al.: **SAT-based** algorithm

- Uses SAT solver to enumerate the assignment of clocking domains to registers.
- Obtains good results at a high computational cost due to the large overhead of SAT solver.

Casanova et al.: **Multi-level Clustering** algorithm

- Progressively clusters half of the registers at each level.
- Much faster, but no guarantee on the solution quality.

Our Contributions

1. A new framework based on branch-and-bound to search for the optimal domain assignment:
    - **Concise enough**, thus avoiding the large overhead of SAT solver.
    - **Effective search strategies**.
2. A greedy clustering algorithm to efficiently estimate the upper bound of a branch:
    - No multi-level process.
    - **Greedily clusters** registers according to their *skew affinity*.

Outline

Timing Constraints

- *Setup time constraints*:
  the signal from u to v has enough time to stabilize its value before storing:
  $l(u) + d_{\max}(u, v) \leq T + l(v) - d_s(v)$.
- *Hold time constraints*:
  the signal from u does not overwrite the previous data in v: $l(u) + d_{\min}(u, v) \geq l(v) + d_h(v)$.

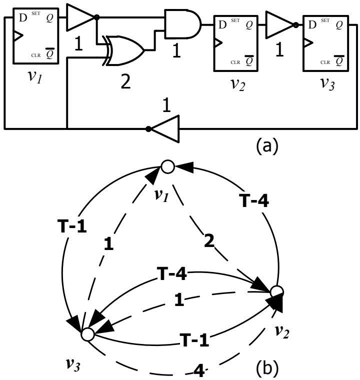

Figure: Timing constraint graph

Multi-Domain Clock Skew Optimization Problem

- Minimize the cycle period while satisfying setup and hold time constraints, and the additional constraints on clock latencies:

$$
\begin{array}{ll}
\min & T \\
s.t. & l(u) + T - d_{\max}(v,u) - d_s(u) \geq l(v), \forall (u,v) \in E_s \\
& l(u) + d_{\min}(u,v) - d_h(v) \geq l(v), \forall (u,v) \in E_h \\
& l(u) \in \{d_1, d_2, ..., d_n\}, \forall u \in V \\
& d_i \in (-T, 0], i = 1, \cdots, n.
\end{array}
$$

The Complexity

- The complexity of multi-domain clock skew optimization problem is not known yet in existing works.
- An upcoming study from our group has shown that the problem is NP-Hard if the number of domains is not constant.

Outline

Preliminaries

1. Slack Interval

- Uniform form of setup and hold time constraints:
$$l(v) - l(u) \leq w(u, v).$$
- Slack of an edge (u, v): the margin for skew increment without violating the constraint:
$$s(u, v) = w(u, v) - (l(v) - l(u)).$$
- *Slack Interval* of a register: the latency range it can have without violating any time constraints.

Preliminaries (Cont'd)

2. Calculation of Slack Intervals under a given cycle period

- **Parametric shortest path algorithm** for slack optimization problem (Albrecht et. al).
- Obtain as large as possible slack intervals for all registers.
- Complexity: $O(|V||E| + |V|^2 \log |V|)$. Time-consuming for large circuits.

Preliminaries (Cont'd)

3. Merge gain

- The more overlap the slack intervals of two registers have, the less impact on performance clustering them causes.
- *Merge gain* (Casanova et. al):

  $gain(u, v) = 2 \times overlap(u, v) - (range(u, v) - overlap(u, v))$

Branch-and-Bound Search Tree

Branch-and-Bound Search Tree

- *Internal nodes* ⇔ partial domain assignments of registers.
- *leaf nodes* ⇔ complete domain assignments.
- *Register to branch*: internal Nodes on the same depth have the same branching registers.

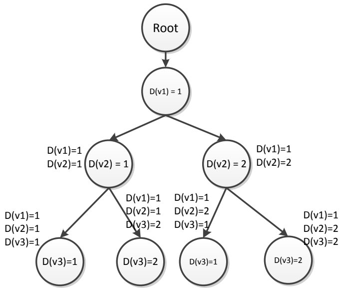

Figure: An example search tree for the previous circuit

Critical Issues for the Search

- Order of registers to branch: determines the order of solution spaces to visit.
- Selection of branch to process: determines the search path to the optimal solution.
- Lower and upper bound computation: important in both branch selection and pruning “bad” branches.

Order of registers to branch

- For registers: smaller slack interval $\Rightarrow$ more critical $\Rightarrow$ easier to determine its domain assignment $\Rightarrow$ branch earlier.
- How to determine:
  1. calculate the optimal cycle period T^* without domain constraints.
  2. calculate the slack intervals under T^*;
  3. sort the registers according their slack interval size.

Selection of branch to process

- Minimum-cost-first strategy. A priority queue for branches is maintained, where

$$prio(b) = \alpha \times lb(b) + (1 - \alpha) \times ub(b) - \beta \times dep(b).$$

- The depth of branches is considered.
  - Compensate the increase of lower and upper bounds when more registers are domains-assigned.

Lower and upper bounds computation for branches

- Lower bound: solve the clock skew scheduling under partial domain assignment.
- Upper bound: an efficient greedy clustering algorithm is developed.

Algorithm (CluBrB) Overview

1. Determine the order of registers to branch; calculate the upper bound T and lower bound T^*.
2. Initialize priority queue *pq*.
3. Process the branch *b* with minimum priority:
    1. Branch *b*.
    2. Calculate the lower bound and upper bound of each child branch.
    3. Update *pq* and T.
4. repeat 3 until $T = T^*$ or *pq* is empty.

Greedy Clustering Algorithm for Upper Bound

- Cluster registers in a bottom-up fashion.
- Always cluster the register pair with the largest merge gain.
- Re-calculate slack intervals (time-consuming) only when the slack interval overlap of the register pair to be clustered is negative.
  - Worst case: $|V| - n$, but often much less than $log_2|V|$ in practice.

Greedy Clustering Algorithm Flow

1. Construct merging priority queue, where the priority is the negative of merge gain of register pairs.
2. Cluster the register pair with minimum priority. If their slack interval overlap is negative, re-construct the merging priority queue.
3. Repeat 2 until the number of remaining registers is n.

Improvement

- $O(|V|^2)$ candidate register pairs for clustering.
- Improves to $O(|V|)$:
  1. Sorting the registers by their slack intervals.
  2. For each register, search the best register to cluster in constant nearest neighbors.

Outline

Experimental Results on ISCAS89 Benchmarks

	SAT-based algorithm	Multi-level Clustering Algorithm	CluBrB (ours)
Accuracy	Optimal	22 of the 60 tests have degradation (up to 7%)	Optimal
Performance	27 circuits: $\leq$ 1 minute, others: slightly longer.	$\leq$ 2 seconds.	27 circuits: $\leq$ 2 seconds, others: slightly longer. Mostly the number of b&b iterations is very small.

Approximation Characteristics

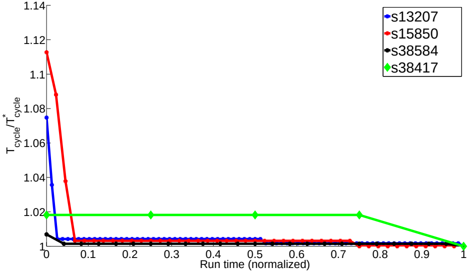

Figure: Track of the search progress for large circuits

Outline

Conclusion

- A practical method for multi-domain clock skew optimization based on branch-and-bound framework with effective search strategy.
- A greedy clustering algorithm to efficiently estimate the upper bound of branches.
- The optimality and efficiency were validated on ISCAS89 benchmarks.