RTレベルデータパスにおけるコントロールスキュースケジューリング

小畑　貴之[†]　　金子　峰雄[†]

[†] 北陸先端科学技術大学院大学 情報科学科　〒923–1292 石川県能美郡辰口町旭台 1–1
E-mail: †{t-obata,mkaneko}@jaist.ac.jp

あらまし　高位合成におけるスケジューリングは通常演算をコントロールステップへ割り当てる問題として捕らえられることが多いが，配線遅延の影響増大と高速化への要求により，レジスタやマルチプレクサの動作タイミングを直接的にスケジュールする必要が生じている．また回路の動作速度の向上のために，順序回路においてフリップフロップへ入力されるクロック信号にスキューを導入する研究が行われてきたが，RTデータパスにおけるレジスタやマルチプレクサのコントロール信号のタイミングもまたクロックに同期しており，コントロール信号にスキューを導入することにより，より高速に動作する回路が得られるものと期待される．本稿ではスケジュール制約下でのスキュー最適化問題を扱うが，これは従来の順序回路に対するクロックスキュー問題と比較して二つの特徴がある．一つはクロックスキュー問題ではフリップフロップはクロック信号に同期して毎回動作するのに対して，コントロールスキュー問題ではレジスタ・マルチプレクサは指定されたコントロールステップでのみ動作することであり，もう一つはクロックスキュー問題ではフリップフロップ間の信号の伝搬は一クロック周期内で行われなければならないのに対してコントロールスキュー問題ではマルチサイクルの伝達が許されることである．後者の特徴によりRTデータパスでは最小動作クロック周期だけでなく最大動作クロック周期をも制限されることになる．本稿では，CDFGとマルチプレクサやレジスタ間の遅延及びスケジュールから制約マルチグラフを構成し，正サイクルの有無を判定して最小のクロック周期とそれを実現するスキューを得る手法を提案する．

キーワード　データパス，タイミングスキュー，スケジューリング

Control Signal Skew Scheduling for RT Level Datapaths

Takayuki OBATA[†] and Mineo KANEKO[†]

[†] School of Information Science, Japan Advanced Institute of Science And Technology 1–1 Asahidai
Tatunokuti, Ishikawa, 923–1292 Japan
E-mail: †{t-obata,mkaneko}@jaist.ac.jp

Abstract　To synthesize high performance VLSI systems, the importance of exploiting interconnection delay information at higher level design is recognized. To improve system performance further in terms of total computation time and/or robustness to delay variation, we are going to introduce appropriate delays which differentiate the arrival times of control signals to registers. A similar technique to the skew of flip-flop control has been proposed for sequential circuits, and significant effort has been devoted to so–called clock scheduling. There is a difference that, in general, clock signal in a sequential circuit is fed to each register every time, while register control signal in RT level datapath is fed to a register only in selected (i.e., scheduled) control steps. When we comprehend the behavior of a datapath with data flows instead of executions of operations, a multiplexer in a datapath behaves as a gate starting and stopping data flow. Introduction of the skew in the arrival times of control signals to multiplexers will contribute, as well as the one for registers, to improve system performance. In this paper, we propose optimization algorithm of skew for both registers and multiplexers in placed RT level datapath.

Key words　datapath, timing skew, scheduling

1. Introduction

In deep sub-micrometer or nanometer technology, not only logic delay but also interconnection delay become dominant factors for the speed of VLSI systems, because interconnection delay can not take advantage of the scaling of feature dimensions. To synthesize high performance VLSI systems, the importance of exploiting interconnection delay information at higher level design is recognized, and several synthesis systems, which combine tasks in high level synthesis and floorplan, have been proposed[1]-[7].

In the past synthesis systems which treat only logic delay, the timing of the start and the end of each elementary operation was major concerns as the timing issue in the high level synthesis. It is because the timing of control signals to the hardware components, such as registers, multiplexers, can be fixed without ambiguity from a schedule of the execution of elementary operations. The situation changes when we consider not only logic delay but also interconnection delay, in which we need to care about the timing of data flow as well as the timing of execution of operations. We can see one typical example of it in [8], where control signals to registers and multiplexers are treated as subjects to be scheduled in the high level synthesis.

We are now considering synchronous circuits, and the timing of control signals to registers and multiplexers is governed by a periodic clock signal. So, the scheduling problem is basically an assignment problem of control signals to discrete time, which is usually called "control step", "state", etc. To improve system performance in terms of total computation time and/or robustness to delay variation, we are going to introduce appropriate delays which differentiate the arrival times of control signals to registers and multiplexers.

A similar technique to the skew of register control has been proposed for sequential circuits, and significant effort has been devoted to so-called clock scheduling [9],[11]. There is a difference that, in general, clock signal in a sequential circuit is fed to each register every time, while register control signal in RT level datapath is fed to a register only in selected (i.e., scheduled) control steps.

When we comprehend the behavior of a datapath with data flows instead of executions of operations, a multiplexer in a datapath behaves as a gate starting and stopping data flow, and the timing of the arrival of control signals to multiplexers is not trivial under interconnection delay dominant situation. Introduction of the skew in the arrival times of control signals to multiplexers will contribute, as well as the one for registers, to improve system performance.

In this paper, we propose optimization algorithm of skew for both registers and multiplexers in placed RT Level datapath.

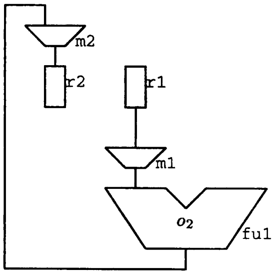

Fig. 1 Datapath relevant to the execution of o_2

2. Preliminaries

In this paper, we consider multiplexer-based RT level datapaths which consist of functional units, registers, multiplexers and nets. Application algorithm to be implemented is given with a set of operations ($\mathcal{O}$), data dependencies between operations (subset of $\mathcal{O}^2$) and control dependencies. The topological structure of a datapath will be fixed depending on the assignment of operations to functional units and data to registers, and the behavior of the datapath will be controlled by a controller (FSM) through control signals which are fed to registers and multiplexers. Note that, associated with each operation $o \in \mathcal{O}$, the following control signals will be considered; (1) control of two multiplexers which are located at input terminals of a functional unit to which o is assigned, and select input data for o, (2) control of a multiplexer which is located at input terminal of a register to which the output data of o is assigned, and selects the output of o as a input to the register, and (3) control of the timing for the register to latch the output of o.

c^o_m: The control signal to module (register or multiplexer) m for the execution of operation o.

σ^o_m: The control step assigned to c^o_m.

D^o_{x-y} / d^o_{x-y}: Maximum / Minimum delay between module x and y, which occurs on data flow related to the execution of the operation o.

s: Setup time of a register.

h: Hold time of a register.

clk: Clock period.

τ_m: Control skew for module m. $0 \leq \tau_m < clk$.

$next(m,o)$: If m is a multiplexer, $next(m,o)$ is the first operation whose control signal reaches m after data related to the execution of o passes through m. If m is a register, $next(m,o)$ is the first operation whose output is written to m after the output of o.

3. Constraints for Feasible Schedule

We now consider timing constraints for control signals rel-

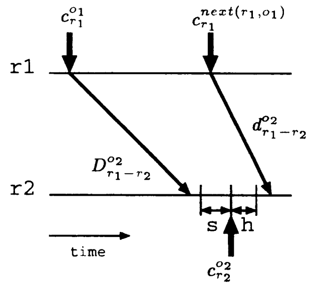

Fig. 2 Setup / Hold conditions

evant to the execution of an operation o_2. We assume that o_2 uses the output of another operation o_1 as an input. The output of o_1 is stored in a register r_1 and is sent to a functional unit which performs o_2 through multiplexer m_1. Let r_2 be a register in which the output of o_2 is written. This situation is illustrated in Fig.1. The first operation which is performed in ful after o_2 is $next(m_1, o_2)$. The first operation whose output is written to r_2 after the output of o_2 is $next(m_2, o_2)$. The operation whose output overwrite the output of o_1 is $next(r_1, o_1)$.

$\sigma_{r_2}^{o_2}$ must be sufficiently later than $\sigma_{r_1}^{o_1}$ to assure that r_2 receives control signal $c_{r_2}^{o_2}$ after arrival of the right result of o_2. This condition is called *setup condition* and is formulated as below.

$$\sigma_{r_1}^{o_1} \cdot clk + D_{r_1-r_2}^{o_2} + s \leqq \sigma_{r_2}^{o_2} \cdot clk \qquad (1)$$

On the other hand, $\sigma_{r_2}^{o_2}$ must be sufficiently earlier than $\sigma_{r_1}^{next(r_1,o_1)}$ to assure that the right result of o_2 is kept at the input port of r_2 until result is surely stored in the register. This condition is called *hold condition* and is formulated as below.

$$\sigma_{r_2}^{o_2} \cdot clk \leqq \sigma_{r_1}^{next(r_1,o_1)} \cdot clk + d_{r_1-r_2}^{o_2} - h \qquad (2)$$

For the above two conditions, please refer to Fig.2

In additing to register-to-register constraints, with respect to m_1, m_2 and r_2, the following setup and hold conditions must be satisfied related to execution of o_2.

$$\sigma_{m1}^{o2} \cdot clk + D_{m1-r2}^{o2} + s \leqq \sigma_{r2}^{o2} \cdot clk \qquad (3)$$

$$\sigma_{m2}^{o2} \cdot clk + D_{m2-r2}^{o2} + s \leqq \sigma_{r2}^{o2} \cdot clk \qquad (4)$$

$$\sigma_{r2}^{o2} \cdot clk \leqq \sigma_{m1}^{next(m_1,o_2)} \cdot clk + d_{m1-r2}^{o2} - h \qquad (5)$$

$$\sigma_{r2}^{o2} \cdot clk \leqq \sigma_{m2}^{next(m_2,o_2)} \cdot clk + d_{m2-r2}^{o2} - h \qquad (6)$$

Note that the timing constraint from register to register and the one from multiplexer to register have a similar formulation, which indicates us that those two are essentially the same problem.

4. Introduction of Control Skew

Even though every control signal is dispatched with being synchronized with a common clock signal, we can make use of skew of arrival time of control signals to improve system performance. Especially in this paper, we will consider a new problem; skew of arrival time of both register control signals and multiplexer control signals in datapath architecture. Fig.3 shows an illustrative example. As we we can see from fig.3(b), we can reduce clk from fig.3(a) by introduction of skew for control signals of module m_j.

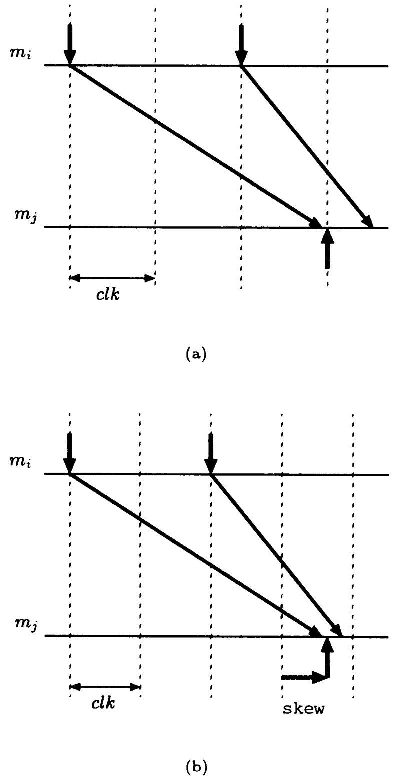

Fig. 3 *clk* reduction with skew

Ideally, the control skew is an assignment of real values to registers and multiplexers, but, in the implementation phase, skew may vary from its nominal value. So we define τ_m as the minimum control skew for module m, and $\tau_m + \tau_{err}$ as the maximum control skew.

Considering control skew, constraints between register m_j and register or multiplexer m_i relevant to operation o_j presented in (1)-(6) are modified as follows

$$\sigma_{m_i}^{o_i} \cdot clk + \tau_{m_i} + \tau_{err} + D_{m_i-m_j}^{o_j} + s \leqq \sigma_{m_j}^{o_j} \cdot clk + \tau_{m_j} \qquad (7)$$

$$\sigma_{m_j}^{o_j} \cdot clk + \tau_{m_j} + \tau_{err} \leqq \sigma_{m_i}^{next(m_i,o_i)} \cdot clk + \tau_{m_i} + d_{m_i-m_j}^{m_j} - h \qquad (8)$$

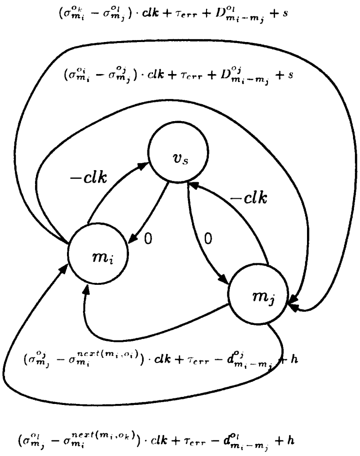
$(\sigma^{o_k}_{m_i} - \sigma^{o_l}_{m_j}) \cdot clk + \tau_{err} + D^{o_l}_{m_i-m_j} + s$
$(\sigma^{o_i}_{m_i} - \sigma^{o_j}_{m_j}) \cdot clk + \tau_{err} + D^{o_j}_{m_i-m_j} + s$
v_s
$-clk$
$-clk$
0
0
m_i
m_j
$(\sigma^{o_j}_{m_j} - \sigma^{next(m_i,o_i)}_{m_i}) \cdot clk + \tau_{err} - d^{o_j}_{m_i-m_j} + h$
$(\sigma^{o_l}_{m_j} - \sigma^{next(m_i,o_k)}_{m_i}) \cdot clk + \tau_{err} - d^{o_l}_{m_i-m_j} + h$

Fig. 4 Skew constraint multigraph

where o_i is an operation whose output is an input of o_j when m_i is a register or $i = j$ when m_i is a multiplexer.

5. Skew Scheduling for Minimizing Clock Period

From (7)-(8), we have

$$\tau_{m_j} - \tau_{m_i} \geq (\sigma^{o_i}_{m_i} - \sigma^{o_j}_{m_j}) \cdot clk + \tau_{err} + D^{o_j}_{m_i-m_j} + s \quad (9)$$

$$\tau_{m_i} - \tau_{m_j} \geq (\sigma^{o_j}_{m_j} - \sigma^{next(m_i,o_i)}_{m_i}) \cdot clk + \tau_{err} - d^{o_j}_{m_i-m_j} + h \quad (10)$$

We generate skew constraint multigraph $G_s = (V, E)$ from (9-10) as shown in Fig.4. Vertices V is a set of multiplexers, registers and one auxiliary source node v_s. Weighted edges E are the union of edges corresponding to (9)-(10) over all operations, and $\{(v, v_s)|v \in V \setminus v_s\} \cup \{(v_s, v)|v \in V \setminus v_s\}$. Edge weights for $\{(v, v_s)|v \in V \setminus v_s\}$ and $\{(v_s, v)|v \in V \setminus v_s\}$ are $-clk$ and 0. Then, skew assignment (skew schedule) problem is now considered as the problem to assign real values to vertices in G_s, and maximum path lengths from v_s to other vertices gives us a solution, i.e., skew of registers and multiplexers. If G_s has a positive cycle, feasible skew schedule dose not exist. Skew schedule for minimizing clock period is, then, the problem to find minimum clk such that the resultant G_s includes no positive cycle.

Once we assign a value to clk, multi edges between a pair of vertices can be reduced to one edge which has maximum weight (since only maximum path length is our concern), and hence we can use Ford algorithm to compute maximum path length.

The following lemma is a key to derive a binary search algorithm for finding minimum clock period.

Step1. Generate G_v. Calculate clk_{min}.

Step2. If $clk_{max} - clk_{min} < \epsilon$, go to Step7.

Step3. Set $clk = (clk_{max} - clk_{min})/2$

Step4. Reduce multi edges of G_v. Apply modified Ford algorithm to compute longest path lengths on G_v.

Step5. If there is no positive cycle, set $clk_{max} = clk$ and go to Step2.

Step6. Extract a positive cycle. If its coefficient is positive, set $clk_{max} = clk$, If not, set $clk_{min} = clk$. Go to Step2.

Step7. Set $clk = clk_{max}$. Reduce multi edges of G_v and apply Ford algorithm. If there exist a positive cycle, given instance has no solution. If not, maximum path lengths to vertices are optimized control skews.

Fig. 5 Algorithm CSO (Control Skew Optimization)

[Lemma 1] There is at most one continuous clk region in which feasible skew assignment exists.

We assume that there is a positive cycle in G_s with $clk = clk^*, clk^* \in R_+$, the cycle weight can be denoted as $x \cdot clk^* + y$. (We call x the "coefficient" of the cycle.) If the coefficient x is positive, the cycle is also positive for all $clk > clk^*$. If the coefficient is negative, the cycle is also positive for all $clk < clk^*$. So we can derive the above lemma. (In conventional skew problem in a sequential circuit, $\sigma^{o_i}_{m_i} - \sigma^{o_j}_{m_j} = -1$ and $\sigma^{o_j}_{m_j} - \sigma^{next(m_i,o_i)}_{m_i} = 0$, and all cycles have negative or zero coefficients.)

If upper and lower bound of clk is given, we can use binary search to find minimum clock period by identifying a positive cycle if it exists and extracting the polarity of its coefficient. Extraction of a cycle coefficient is performed using modified Ford algorithm whose complexity is the same with the original one.

The lower bound of clk "clk_{min}" is calculated from (9)-(10). clk_{min} must satisfy the following inequality so that $\tau_{m_j} - \tau_{m_i}$ has feasible range of value.

$$clk_{min} \geq \frac{D^{o_j}_{m_i-m_j} - d^{o_j}_{m_i-m_j} + s + h + 2\tau_{err}}{\sigma^{next(m_i,o_i)}_{m_i} - \sigma^{o_i}_{m_i}} \quad (11)$$

The upper bound "clk_{max}" is assumed to be given by a user.

The algorithm is shown in Fig.5. "ϵ" is the resolution of clock period, and it is also given by a user.

[Theorem 1] Algorithm CSO computes minimum clock period and feasible control skew for registers and multiplexers if they exist, and says "no solution" if they not, with $O(|\mathcal{O}|^2 \cdot log_2((clk_{max} - clk_{min})/\epsilon))$ computation time.

First, $|V| = O(|\mathcal{O}|)$ and $|E| = O(|\mathcal{O}|)$. The computational complexity for constructing G_s is $O(|\mathcal{O}|)$. Computation of clk_{min} and clk_{max} take $O(|E|)$ time. Let ϵ be time unit for clk. Binary search recurs $O(log_2((clk_{max} - clk_{min})/\epsilon))$ times. In each recurrence takes $O(|E|)$ time to remove multi edges and $O(V \cdot E)$ time for ford algorithm. Finally, proposed algorithm takes $O(|\mathcal{O}|^2 \cdot log_2((clk_{max} - clk_{min})/\epsilon))$ time.

6. Experiment

Table 1 Experimental Results

Jaumann

instance ID	n/s	w/s	ratio	#fu	#reg
1	47	31	0.66	9	10
2	51	32	0.62	9	9
3	51	41	0.80	9	10
4	51	36	0.70	9	11
5	66	52	0.79	9	10
6	68	60	0.88	9	12
7	69	59	0.86	9	12
8	70	59	0.84	10	11

Lattice

instance ID	n/s	w/s	ratio	#fu	#reg
1	64	40	0.63	6	8
2	42	27	0.64	6	8
3	56	38	0.68	6	8
4	66	44	0.67	6	8
5	56	40	0.71	6	8
6	64	57	0.89	6	8
7	59	56	0.95	6	8
8	72	58	0.81	5	8

Proposed method was implemented using C and tested on AMD Athlon based system. Our experimental instance are two DAG algorithms based on Jaumann wave filter and all-pole lattice filter. For each algorithm, eight different control step schedule and binding are tested. Experimental results are shown in table 1. A user defined parameter clock resolution is 2. The first column n/s is the minimum clock period with zero skew. The second column w/s is the minimum clock period with optimized skew obtained from our proposed algorithm. The third column is the reduction ratio of clock period. Last two columns show the number of functional units and registers. The results show the effectiveness of the skew optimization.

7. Conclusion

In this paper, we have focused on a new problem, the skew of the arrival time of control signals to multiplexers and registers in a RT level datapath. In interconnection delay dominant VLSI systems, the timing of the arrival of control signals at multiplexers should be carefully controlled as much as the timing of the arrival of control signals at registers.

We have formulated the control skew schedule problem. An algorithm for this problem based on binary search has been developed, and that can be applied also to clock skew optimization of sequential circuits which have multi cycle paths.

Future work is to develop a unified algorithm which optimizes control step schedule considering control signal skew.

Acknowledgment

This research is conducted as a program for the " 21st Century COE Program " by Ministry of Education, Culture, Sports, Science and Technology. This work is also partly supported by the Ministry of Education, Culture, Sports, Science, and Technology under Grant-in-Aid for Scientific Research (C), No. 16560294, 2004-2005.

References

[1] J. P. Weng, A. C. Parker, “3D scheduling: high-level synthesis with floorplanning,” Proc. Design Automation Conf., pp.668-673, 1991.

[2] Y. M. Fang, D. F. Wong, “Simultaneous functional unit binding and floorplanning,” Proc. Int. Conf. on Computer Aided Design, pp.317-321, 1994.

[3] V. G. Moshnyaga, K. Tamaru, “A placement driven methodology for high-level synthesis of sub-micron ASIC's,” Proc. Int. Symp. on Circuits and Systems, vol. 4, pp572-575, 1996.

[4] S. Tarafdar, M. Leeser, Z. Yin, “Integrating floorplanning in data-transfer based high-level synthesis,” Proc. Int. Conf. on Computer Aided Design, pp.412–417, 1998.

[5] P. Prabhakaran, P. Banerjee, “Parallel algorithm for simultaneous scheduling, binding and floorplanning in high-level synthesis,” Proc. Int. Symp. on Circuits and Systems, vol. 6, pp372-376, 1998.

[6] K. Ohashi, M. Kaneko, S. Tayu, “Assignment-space exploration approach to concurrent datapath/floorplan synthesis,” Proc. Int. Conf. on Computer Design, pp.370–375, 2000.

[7] D. Kim, J. Jung, S. Lee, J. Jeon, K. Choi, “Behavior-to-placed RTL synthesis with performance-driven placement,” Proc. Int. Conf. on Computer Aided Design, 2001.

[8] . Kaneko, K. Ohashi, “Assignment Constrained Scheduling under Max/Min Logic/Interconnect Delays for Placed Datapath”, Proc. APCCAS2004, vol.2, pp.545-548, 2004.

[9] John P. Fishburn, “Clock Skew Optimization”, IEEE Trans. on Computers, pp. 945–951, Vol.39, No. 7, 1990.

[10] Xun Liu, Marios C.Papaefthymiou, Eby G. Friedman, “Retiming and Clock Scheduling for Digital Circuit Optimization”, IEEE Trans. on Computer Aided Design, pp.184–203, Vol.21, No. 2, 2002.

[11] R. B. Deokar and S. S. Sapatnekar, “A graphtheoretic approach to clock skew optimization,” Proc. IEEE Int. Symp. Circuits and Systems, pp. 1.407-1.410, 1994.