

Simultaneous Clock Skew Scheduling and Power-Gated Module Selection for Standby Leakage Minimization*

SHIH-HSU HUANG, CHUN-HUA CHENG AND DA-CHEN TZENG

Department of Electronic Engineering

Chung Yuan Christian University

Chungli, 320 Taiwan

E-mail: {shhuang; g9402601; g9476017}@cycu.edu.tw

Leakage current minimization is an important topic for event driven applications that spend most of their times in standby mode. Power gating technique is one of the most effective ways to reduce the standby leakage current. However, when power gating technique is applied to a functional unit, there exists a delay-power tradeoff, which can be characterized with the widths of sleep transistors. In this paper, we point out that: under the same target clock period, there are many feasible clock skew schedules; since different clock skew schedules impose different timing constraints to functional units, different clock skew schedules may lead to different standby leakage currents. Based on that observation, we present an MILP (mixed integer linear programming) approach to formally formulate the problem of simultaneous application of optimal clock skew scheduling and power-gated module selection (*i.e.*, sleep transistor width selection) in high-level synthesis stage. Experimental data show that: compared with the existing design flow, our standby leakage current reduction achieves 29.3%.

Keywords: electronic design automation, clock skew scheduling, high-level synthesis, power gating, mixed integer linear programming

1. INTRODUCTION

High performance and low power are the two important concerns in modern circuit design. For event driven applications, like a processor running X-server, spend most of their times in standby mode while no computation was performed, and therefore standby leakage current will account for a large fraction of total power consumption. Thus, modern event driven application designs face the following two challenges: the first challenge is to reduce the clock period for high performance (in active mode), and the second challenge is to reduce the standby leakage current for low power.

For the first challenge, the clock skew is a manageable resource to reduce the clock period [1-7]. By properly scheduling the clock arrival times of registers, the clock period of a nonzero clock skew circuit can be shorter than the longest combinational delay. The optimal clock skew scheduling problem [1-6] is to obtain the smallest feasible clock period and the clock arrival time of each register. Several graph-based algorithms [2-5] have been proposed to solve the optimal clock skew scheduling efficiently. Recently, Huang *et al.* [7] point out that the register binding in high-level synthesis has a significant impact on the design of a nonzero clock skew circuit. Therefore, the utilization of

Received March 7, 2008; revised July 15 & December 3, 2008; accepted December 11, 2008.

Communicated by Yao-Wen Chang.

* This work was supported in part by the National Science Council of Taiwan, R.O.C., under contract No. NSC 96-2628-E-033-004-MY3.

clock skew should be considered starting from the stage of high-level synthesis.

For the second challenge, one technique, called multi-threshold CMOS (MTCMOS), is becoming more popular [8-15]. As shown in Fig. 1 (a), the technique utilizes a high V_{th} (threshold voltage) transistor (called sleep transistor or power gate) to gate the power supply lines for the entire functional unit when the circuit is in standby mode. Note that the determination of sleep transistor width has two opposing criteria. On the one hand, in the standby mode (sleep = 1), the sleep transistor is turned off. The standby leakage current of the functional unit is proportional to the width of the sleep transistor. On the other hand, in the active mode (sleep = 0), the sleep transistor is turned on and works as a resistor as shown in Fig. 1 (b). The normal current flowing through the sleep transistor produces a voltage drop that degrades the speed of the functional unit. Therefore, in high-level synthesis stage, we can construct many different delay-power characteristic power-gated modules for a same type of functional unit by changing the width of sleep transistor.

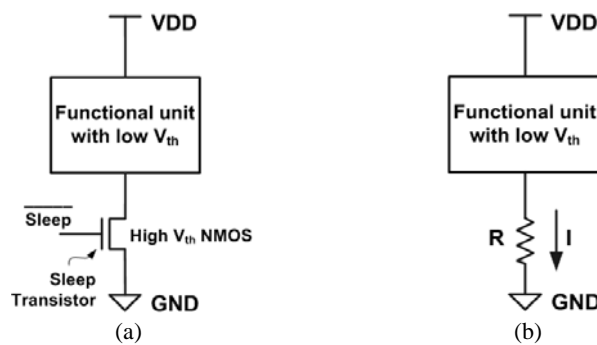


Fig. 1. (a) Functional unit with power gating; (b) Sleep transistor is modeled as a resistor in active mode.

From the above discussions, there is a demand to design a nonzero clock skew circuit with power gating. However, in the existing design flow, optimal clock skew scheduling and power gating are two independent processes. Up to now, no attention has been paid to the interaction between optimal clock skew scheduling and power gating. In this paper, we point out that: under the same target clock period, there are many feasible clock skew schedules; since different clock skew schedules impose different timing constraints to functional units, different clock skew schedules may lead to different standby leakage currents. Therefore, we have the motivation to study the power gating of nonzero clock skew circuits.

In this paper, we study the simultaneous application of optimal clock skew scheduling and power-gated module selection (*i.e.*, sleep transistor width selection) in high-level synthesis stage. Note that our paper is the first work to deal with the problem. We conjecture that the problem is NP-hard. Therefore, an MILP (mixed integer linear programming) approach is proposed to solve the problem optimally. Compared with the existing design flow, benchmark data show that our approach can save 29.3% standby leakage current.

The rest of this paper is organized as follows. Section 2 revisits the optimal clock skew scheduling. Section 3 studies the functional unit library with power gating considered (in high-level synthesis stage). Section 4 demonstrates our motivation. Then, in section 5, we present our MILP approach. The experimental results are given in section 6. Finally, in section 7, we provide some concluding remarks.

2. OPTIMAL CLOCK SKEW SCHEDULING

In this section, we borrow the materials from [7] to address the optimal clock skew scheduling in high-level synthesis. A data path from register R_i to register R_j is defined as the combinational logic from register R_i to register R_j . Thus, if the input variable of operation O_k is assigned to register R_i and the output variable of operation O_k is assigned to register R_j , the data path from register R_i to register R_j includes the functional unit that executes operation O_k . Since a data path may perform different operations at different control steps, a data path may include several functional units. As a result, the minimum delay (maximum delay) of a data path is the minimum delay (maximum delay) among all the functional units included in the data path.

Given a scheduled DFG and a resource binding solution (including functional unit binding and register binding), we can model the hardware as a circuit graph, in which each vertex denotes a register and each directed edge denotes a data path. A special vertex called the *host* is introduced for the synchronization with primary inputs and primary outputs. Each directed edge $R_i \rightarrow R_j$ is associated with a weight $(\min(R_i, R_j), \max(R_i, R_j))$, where $\min(R_i, R_j)$ and $\max(R_i, R_j)$ are the minimum delay and the maximum delay of the data path from register R_i to register R_j , respectively.

Let T_i denote the clock arrival time of register R_i . For a data path from register R_i to register R_j , there are two types of timing constraints: setup constraint and hold constraint. To prevent the data reaching a register too late relative to the following clock pulse, the clock skew must satisfy the following setup constraint: $T_i - T_j \leq P - \max(R_i, R_j)$, where P is the target clock period. To prevent the same clock pulse triggering the same data into two adjacent registers, the clock skew must satisfy the following hold constraint: $T_j - T_i \leq \min(R_i, R_j)$. We say that a circuit graph works with the target clock period P , if and only if there is a clock skew schedule (*i.e.*, a solution of clock arrival times of registers) that satisfies all the timing constraints.

The *optimal clock skew scheduling* problem [1-6] is to find the smallest feasible clock period of a circuit graph and the clock skew schedule for the circuit graph. Conventionally, a constraint graph is used to model all the timing constraints of a circuit graph for solving the clock skew scheduling problem. In the constraint graph, each vertex represents a register and each directed edge $R_i \rightarrow R_j$ associated with a weight $w_{i,j}$ corresponds to the constraint $T_j - T_i \leq w_{i,j}$. Therefore, each data path from register R_i to register R_j in the circuit graph G has the following two directed edges in the constraint graph $G_{cg}(G)$: the setup constraint is modeled as a directed edge $R_j \rightarrow R_i$ associated with a weight $w_{j,i} = P - \max(R_i, R_j)$, and the hold constraint is modeled as a directed edge $R_i \rightarrow R_j$ associated with a weight $w_{i,j} = \min(R_i, R_j)$. Note that, there is a feasible clock skew schedule for the circuit graph G to work with the target clock period P , if and only if the constraint graph $G_{cg}(G)$ contains no negative cycle when the clock period is P . Based on

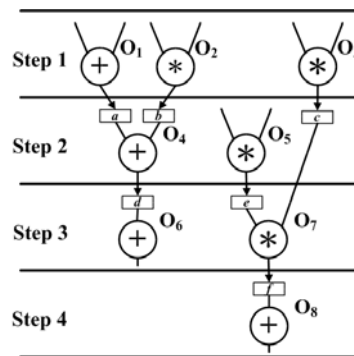


Fig. 2. A scheduled DFG.

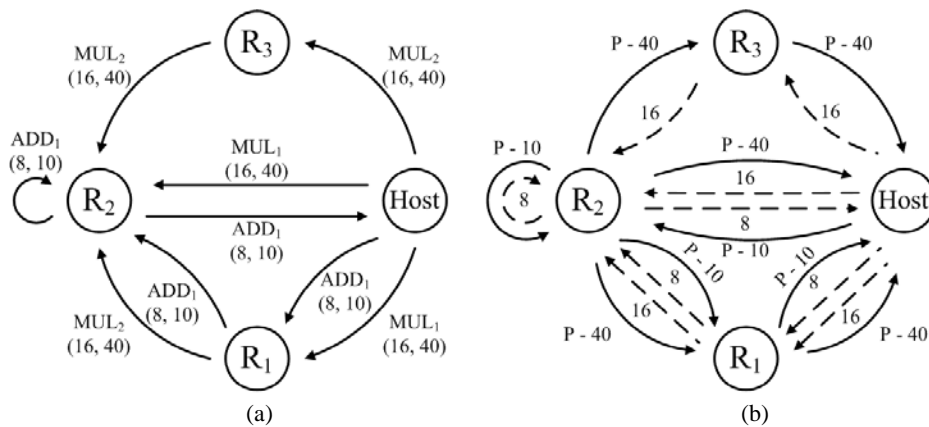


Fig. 3. (a) Circuit graph G_1 ; (b) Constraint graph $G_{cg}(G_1)$.

this property, several algorithms, including the binary search strategy [2], the shortest path approach [3], and the cycle detection method [4], have been proposed to solve the optimal clock skew scheduling problem efficiently.

Let's use the scheduled DFG shown in Fig. 2 for illustration. Suppose that we are given two multipliers (MUL_1 and MUL_2), one adder (ADD_1), and three registers (R_1 , R_2 , and R_3), and the resource binding solution is $MUL_1 = \{O_2, O_5\}$, $MUL_2 = \{O_3, O_7\}$, $ADD_1 = \{O_1, O_4, O_6, O_8\}$, $R_1 = \{a, e\}$, $R_2 = \{b, d, f\}$, and $R_3 = \{c\}$ ¹. Suppose that the minimum delay and the maximum delay of the multiplier MUL_1 are 16 and 40, respectively, the minimum delay and the maximum delay of the multiplier MUL_2 are 16 and 40, respectively, and the minimum delay and the maximum delay of the adder ADD_1 are 8 and 10, respectively. As a result, we can derive a circuit graph G_1 as show in Fig. 3 (a). The corresponding constraint graph $G_{cg}(G_1)$ is displayed in Fig. 3 (b). After the optimal clock skew scheduling is applied, we find that the smallest feasible clock period is 32 under $T_{host} = 0$, $T_1 = 8$, $T_2 = 16$, and $T_3 = 8$.

¹ The notation $MUL_1 = \{O_2, O_5\}$ means that operations O_2 and O_5 are assigned to multiplier MUL_1 .

3. FUNCTIONAL UNIT LIBRARY WITH POWER GATING CONSIDERED

Up to now, in high-level synthesis stage, no attention is paid to construct the functional unit library with power gating considered. In this section, we study this problem.

We assume a single sleep transistor is employed to support the power gating of a functional unit.² Note that, for each functional unit, determining the width of sleep transistor faces two but opposing design criteria. On the one hand, during active mode (*i.e.*, the sleep transistor is turned on), the sleep transistor acts as a resistor (whose resistance value is R), as shown in Fig. 1 (b), which causes a voltage drop at virtual ground line and the voltage drop is equal to $I \times R$, where I is the current flowing through the sleep transistor. Because of the voltage drop, the operating speed of the functional unit degrades more when the width of sleep transistor shrinks. To reduce the performance penalty, the value R should be as small as possible, which implies the width of sleep transistor should be as large as possible. On the other hand, during the standby mode (*i.e.*, the sleep transistor is turned off), the leakage current flowing through the sleep transistor is proportional to the width of the sleep transistor. To minimize the standby leakage current of the functional unit, the width of sleep transistor should be designed as small as possible.

Since there is a delay-power tradeoff, in high-level synthesis stage, a same type of functional unit should be characterized with different power-gated modules (*i.e.*, different sleep transistor widths). In the following, we use the functional unit library shown in the Table 1 for illustration. The column *Functional Type* denotes the type of functional unit. The column *Module Name* denotes the names of power-gated modules. The column *Transistor Width* denotes the widths of sleep transistors. The multiplier type and adder type are both characterized with two different sleep transistor widths. The column *Delay* is a two-tuple (min, max), in which *min* denotes the minimum delay and *max* denotes the maximum delay. For example, the minimum delay and the maximum delay of functional unit ADD_fast is 8 and 10, respectively. The column *Leakage Current* denotes the standby leakage current. For the convenience of presentation, in the following, we use the form $MUL_1 \leftarrow MUL_fast$ to represent that we use the module MUL_fast to implement the multiplier MUL_1 .

Table 1. Delay-power characterization of adder and multiplier.

Functional Type	Module Name	Transistor Width	Delay (min, max)	Leakage Current
Adder	ADD_fast	Large	(8, 10)	80
	ADD_slow	Small	(10, 12)	40
Multiplier	MUL_fast	Large	(16, 40)	100
	MUL_slow	Small	(20, 42)	50

4. MOTIVATION

In this section, we demonstrate our motivation. Section 4.1 describes the existing design flow. Section 4.2 points out our observation: the existing design flow cannot minimize the standby leakage current.

² In this paper, we do not consider the distributed sleep transistor network [12, 13, 15].

4.1 Existing Design Flow

Velenis *et al.* [6] present a two-step process to design a nonzero clock skew circuit for both speed and power enhancement: in the first step, optimal clock skew scheduling is applied for clock period minimization; then, in the second step, low power techniques, such as supply voltage scaling and gate sizing, are applied to reduce the power consumptions of non-critical data paths. Note that Velenis *et al.* [6] do not mention the power gating. However, the application of power gating in the second step is straightforward.

Therefore, intuitively, we can use the following design flow to implement the power gating of a nonzero clock skew circuit. In high-level synthesis stage, the fastest power-gated modules are selected for all functional units. Then, after high-level synthesis, the two-step process presented in [6] is adopted for speed and power enhancement. We elaborate the details as below.

Step 1: Clock skew scheduling for clock period minimization.

By selecting the fastest power-gated modules for all functional units, we derive a circuit graph. Based on the circuit graph, the optimal clock skew scheduling is applied to obtain the smallest feasible clock period and the clock arrival time of each register.

Step 2: Power-gated module selection for standby leakage current minimization.

According to the clock arrival time of each register (which is obtained in step 1), we minimize the standby leakage current of each functional unit by choosing the slowest power-gated module that can satisfy the timing constraints.

Let's use the scheduled DFG shown in Fig. 2 for illustration. Suppose that we are given two multipliers (MUL_1 and MUL_2), one adder (ADD_1), and three registers (R_1 , R_2 , and R_3), and the resource binding solution is $MUL_1 = \{O_2, O_5\}$, $MUL_2 = \{O_3, O_7\}$, $ADD_1 = \{O_1, O_4, O_6, O_8\}$, $R_1 = \{a, e\}$, $R_2 = \{b, d, f\}$, and $R_3 = \{c\}$. In addition, suppose that we use the functional unit library as shown in Table 1. Then, in the existing design flow, we can use the two-step process presented in [6] to implement the power gating of nonzero clock skew circuits.

Step 1: Clock skew scheduling for clock period minimization.

We select the fastest power-gated module to implement each functional unit; *i.e.*, $MUL_1 \leftarrow MUL_fast$, $MUL_2 \leftarrow MUL_fast$, and $ADD_1 \leftarrow ADD_fast$. As a result, we can derive a circuit graph $G1$ as show in Fig. 3 (a). The corresponding constraint graph $G_{cg}(G1)$ is displayed in Fig. 3 (b). After the optimal clock skew scheduling is applied, we find that the smallest feasible clock period is 32 under $T_{host} = 0$, $T_1 = 8$, $T_2 = 16$, and $T_3 = 8$.

Step 2: Power-gated module selection for standby leakage current minimization.

According to the clock arrival time of each register (which is obtained in step 1), we implement each functional unit with the slowest power-gated module that can satisfy the timing constraints. We analyze each functional unit as below.

- Consider the multiplier MUL_1 . The data path from host to register R_1 includes the multiplier MUL_1 . According to the setup constraint, the maximum delay of multiplier MUL_1

- could not exceed 40 (*i.e.* $P + T_1 - T_{host} = 32 + 8 = 40$). Therefore, we only can use fastest power-gated module to implement MUL_1 .
- Consider the multiplier MUL_2 . The data path from host to register R_3 includes the multiplier MUL_2 . According to the setup constraint, the maximum delay of multiplier MUL_2 could not exceed 40 (*i.e.* $P + T_3 - T_{host} = 32 + 8 = 40$). Therefore, we only can use fastest power-gated module to implement MUL_2 .
 - Consider the adder ADD_1 , which is included in the data path from host to register R_1 , the data path from register R_1 to register R_2 , and the data path from register R_2 to host. Since $T_{host} = 0$ and $T_1 = 8$, and $T_2 = 16$, even if we implement adder ADD_1 with the module ADD_slow , the timing constraints are still satisfied. Therefore, we can use module ADD_slow to implement adder ADD_1 .

From above analyses, we obtain the following power-gated module selection solution: $MUL_1 \leftarrow MUL_fast$, $MUL_2 \leftarrow MUL_fast$, and $ADD_1 \leftarrow ADD_slow$. For the convenience of readers, Fig. 4 (a) provides the new circuit graph G_2 , and Fig. 4 (b) provides the new constraint graph $G_{cg}(G_2)$. Note that, when $T_{host} = 0$, $T_1 = 8$, $T_2 = 16$, $T_3 = 8$, and the target clock period is 32, all the timing constraints in the constraint graph $G_{cg}(G_2)$ are satisfied. According to the power-gated module selection solution, *i.e.*, $MUL_1 \leftarrow MUL_fast$, $MUL_2 \leftarrow MUL_fast$, and $ADD_1 \leftarrow ADD_slow$, the standby leakage current of the circuit is 240 ($100 + 100 + 40 = 240$).

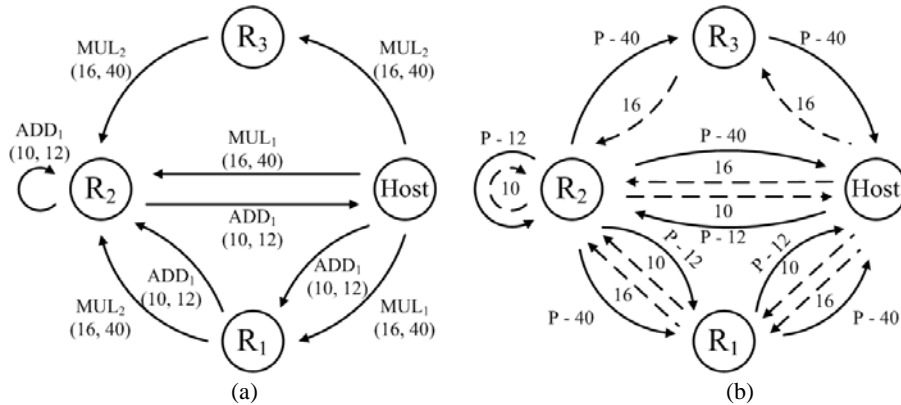


Fig. 4. (a) Circuit graph G_2 ; (b) Constraint graph $G_{cg}(G_2)$.

4.2 Our Observation

In fact, in this example, there exists a solution, in which the standby leakage current is only 140 under the same target clock period (*i.e.*, the target clock period is 32). Consider the following solution: $T_{host} = 0$, $T_1 = 10$, $T_2 = 20$, $T_3 = 10$, $MUL_1 \leftarrow MUL_slow$, $MUL_2 \leftarrow MUL_slow$, and $ADD_1 \leftarrow ADD_slow$. Fig. 5 (a) gives the corresponding circuit graph G_3 . Fig. 5 (b) give the corresponding constraint graph $G_{cg}(G_3)$. When the target clock period is 32, all the timing constraints in the constraint graph $G_{cg}(G_3)$ are met. Since each functional unit uses the slowest power-gated module, the standby leakage current of the circuit is only 140 ($50 + 50 + 40$).

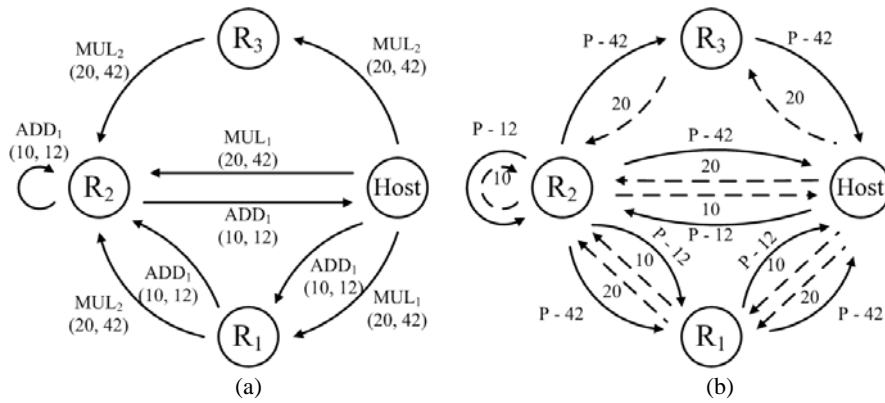


Fig. 5. (a) Circuit graph G_3 ; (b) Constraint graph $G_{cg}(G_3)$.

From this example, we find that the standby leakage current is not minimized in the existing design flow (*i.e.*, the two-step process presented in [6]). The reason is that: in the existing design flow, optimal clock skew scheduling and power gating are two independent processes. Therefore, in the existing design flow, the clock skew schedule is derived without the consideration of power-gated module selection. However, under the same target clock period, there are many feasible clock skew schedules; since different clock skew schedules impose different timing constraints to functional units, different clock skew schedules may lead to different standby leakage currents. As a result, in order to minimize the standby leakage current, there is a demand to study the simultaneous application of optimal clock skew scheduling and power-gated module selection.

5. THE PROPOSED MILP APPROACH

In this section, we propose an MILP approach to formally formulate the problem of simultaneous application of optimal clock skew scheduling and power-gated module selection. Note that, under the target clock period, our MILP approach guarantees minimizing the standby leakage current.

First, we introduce the constants, notations, and variables used in our MILP approach as below.

- For each register R_i , we define a real-value variable T_i , which denotes its clock arrival time.
- The notation $c(t)$ denotes the set of functional units in the type t . For example, if the number of multiplier and adder is 2 and 1 respectively, we have $c(mul) = \{MUL_1, MUL_2\}$ and $c(add) = \{ADD_1\}$ (note that, here, multiplier and adder are abbreviated as *mul* and *add*, respectively).
- The notation $h(t)$ denotes the set of sleep transistor widths characterized for the functional unit in the type t . Take the functional unit library given in Table 1 as an example. The set $h(\text{multiplier})$ is {large, small}.
- The notation $\langle t, w \rangle$ denotes the following power-gated module selection: the type is t and the sleep transistor width is w .

- The constants $d_{\langle t, w \rangle}$, $D_{\langle t, w \rangle}$, and $I_{\langle t, w \rangle}$ denote the minimum delay, the maximum delay, and the standby leakage current of the power-gated module $\langle t, w \rangle$, respectively. Take the functional unit library given in Table 1 as an example. For the module `MUL_fast`, we have $d_{\langle mul, large \rangle} = 16$, $D_{\langle mul, large \rangle} = 40$, and $I_{\langle mul, large \rangle} = 100$.
- The notation $e(z)$ denotes the functional type of power-gated module z .
- For each combination of functional unit z and the power-gated module selection $\langle e(z), w \rangle$, we define a binary variable $f_{z, \langle e(z), w \rangle}$. If functional unit z is implemented by $\langle e(z), w \rangle$, then the value of $f_{z, \langle e(z), w \rangle}$ is 1; otherwise, the value of $f_{z, \langle e(z), w \rangle}$ is 0. For example, if $MUL_1 \leftarrow MUL_fast$, then the value of $f_{MUL_1, \langle mul, large \rangle}$ is 1; otherwise, the value of $f_{MUL_1, \langle mul, large \rangle}$ is 0.

Next, we present the objective function and the constraints used in our MILP approach. The objective function is:

$$\text{Minimize } \sum_{z \in Q} \sum_{w \in h(e(z))} f_{z, \langle e(z), w \rangle} \times I_{\langle e(z), w \rangle}. \quad (1)$$

The constraints are as below. Each functional unit must be assigned to a power-gated module. Therefore, for each functional unit z , we have the following constraint:

$$\sum_{w \in h(e(z))} f_{z, \langle e(z), w \rangle} = 1. \quad (2)$$

Let P be a constant that denotes the target clock period. Suppose that the input of operation O_k is variable u , the output of operation O_k is variable v , variable u is assigned to register R_i , and variable v is assigned to register R_j . Then, for the data path from register R_i to register R_j , we have the following setup constraint:

$$T_i - T_j \leq P - \sum_{w \in h(e(z))} f_{z, \langle e(z), w \rangle} \times D_{\langle e(z), w \rangle}. \quad (3)$$

Suppose that the input of operation O_k is variable u , the output of operation O_k is variable v , variable u is assigned to register R_i , and variable v is assigned to register R_j . Then, for the data path from register R_i to register R_j , we have the following hold constraint:

$$T_j - T_i \leq \sum_{w \in h(e(z))} f_{z, \langle e(z), w \rangle} \times d_{\langle e(z), w \rangle}. \quad (4)$$

Take the scheduled DFG shown in Fig. 2 as example. Suppose that the resource binding solution is $MUL_1 = \{O_2, O_5\}$, $MUL_2 = \{O_3, O_7\}$, $ADD_1 = \{O_1, O_4, O_6, O_8\}$, $R_1 = \{a, e\}$, $R_2 = \{b, d, f\}$, and $R_3 = \{c\}$, the target clock period is 32, and the functional unit library is as shown in Table 1. Then, our MILP formulation is as below.

Due to Formula (1), the objective function is:

$$\begin{aligned} \text{Minimize } & f_{MUL_1, \langle mul, large \rangle} \times 100 + f_{MUL_1, \langle mul, small \rangle} \times 50 + f_{MUL_2, \langle mul, large \rangle} \times 100 \\ & + f_{MUL_2, \langle mul, small \rangle} \times 50 + f_{ADD_1, \langle add, large \rangle} \times 80 + f_{ADD_1, \langle add, small \rangle} \times 40. \end{aligned}$$

Due to Formula (2), we have the following constraints:

$$\begin{aligned} f_{MUL_1, \langle mul, large \rangle} + f_{MUL_1, \langle mul, small \rangle} &= 1; \\ f_{MUL_2, \langle mul, large \rangle} + f_{MUL_2, \langle mul, small \rangle} &= 1; \\ f_{ADD_1, \langle add, large \rangle} + f_{ADD_1, \langle add, small \rangle} &= 1. \end{aligned}$$

Due to Formula (3), we have the following setup constraints:

$$\begin{aligned} T_{host} - T_1 &\leq 32 - (f_{ADD_1, \langle add, large \rangle} \times 10 + f_{ADD_1, \langle add, small \rangle} \times 12); \\ T_1 - T_2 &\leq 32 - (f_{ADD_1, \langle add, large \rangle} \times 10 + f_{ADD_1, \langle add, small \rangle} \times 12); \\ T_2 - T_{host} &\leq 32 - (f_{ADD_1, \langle add, large \rangle} \times 10 + f_{ADD_1, \langle add, small \rangle} \times 12); \\ T_2 - T_2 &\leq 32 - (f_{ADD_1, \langle add, large \rangle} \times 10 + f_{ADD_1, \langle add, small \rangle} \times 12); \\ T_{host} - T_1 &\leq 32 - (f_{MUL_1, \langle mul, large \rangle} \times 40 + f_{MUL_1, \langle mul, small \rangle} \times 42); \\ T_1 - T_2 &\leq 32 - (f_{MUL_2, \langle mul, large \rangle} \times 40 + f_{MUL_2, \langle mul, small \rangle} \times 42); \\ T_3 - T_2 &\leq 32 - (f_{MUL_2, \langle mul, large \rangle} \times 40 + f_{MUL_2, \langle mul, small \rangle} \times 42); \\ T_{host} - T_3 &\leq 32 - (f_{MUL_2, \langle mul, large \rangle} \times 40 + f_{MUL_2, \langle mul, small \rangle} \times 42); \\ T_{host} - T_2 &\leq 32 - (f_{MUL_1, \langle mul, large \rangle} \times 40 + f_{MUL_1, \langle mul, small \rangle} \times 42). \end{aligned}$$

Due to Formula (4), we have the following hold constraints:

$$\begin{aligned} T_1 - T_{host} &\leq (f_{ADD_1, \langle add, large \rangle} \times 8 + f_{ADD_1, \langle add, small \rangle} \times 10); \\ T_2 - T_1 &\leq (f_{ADD_1, \langle add, large \rangle} \times 8 + f_{ADD_1, \langle add, small \rangle} \times 10); \\ T_{host} - T_2 &\leq (f_{ADD_1, \langle add, large \rangle} \times 8 + f_{ADD_1, \langle add, small \rangle} \times 10); \\ T_2 - T_2 &\leq (f_{ADD_1, \langle add, large \rangle} \times 8 + f_{ADD_1, \langle add, small \rangle} \times 10); \\ T_1 - T_{host} &\leq (f_{MUL_1, \langle mul, large \rangle} \times 16 + f_{MUL_1, \langle mul, small \rangle} \times 20); \\ T_2 - T_1 &\leq (f_{MUL_2, \langle mul, large \rangle} \times 16 + f_{MUL_2, \langle mul, small \rangle} \times 20); \\ T_2 - T_3 &\leq (f_{MUL_2, \langle mul, large \rangle} \times 16 + f_{MUL_2, \langle mul, small \rangle} \times 20); \\ T_3 - T_{host} &\leq (f_{MUL_2, \langle mul, large \rangle} \times 16 + f_{MUL_2, \langle mul, small \rangle} \times 20); \\ T_2 - T_{host} &\leq (f_{MUL_1, \langle mul, large \rangle} \times 16 + f_{MUL_1, \langle mul, small \rangle} \times 20). \end{aligned}$$

After solving the MILP formulation, we find that: $f_{MUL_1, \langle mul, large \rangle} = 0$, $f_{MUL_2, \langle mul, large \rangle} = 0$, $f_{ADD_1, \langle mul, large \rangle} = 0$, $f_{MUL_1, \langle mul, small \rangle} = 1$, $f_{MUL_2, \langle mul, small \rangle} = 1$, $f_{ADD_1, \langle mul, small \rangle} = 1$, $T_{host} = 0$, $T_1 = 10$, $T_2 = 20$, and $T_3 = 10$. Therefore, we have $MUL_1 \leftarrow MUL_slow$, $MUL_2 \leftarrow MUL_slow$, and $ADD_1 \leftarrow ADD_slow$. Note that the standby leakage current of the circuit is only 140.

6. EXPERIMENTAL RESULTS

In our experiment, we use synthesizable intellectual properties provided in Synopsys DesignWare library to implement the following types of functional units: ALU, multiplier, divisor, selector, and comparator. Without loss of generality, these functional units are assumed to be 16-bit designs and they are targeted to TSMC 0.18 μ m process technology. The logic synthesis tool is Synopsys Design Compiler, and the placement and routing tool is Synopsys Astro.

Note TSMC 0.18 μ m process technology does not support MTCMOS. The standard threshold voltage in TSMC 0.18 μ m process technology is 0.52V. In our experiment, we

assume the threshold voltage of functional unit is 0.52V (*i.e.*, low V_{th} is 0.52V) and the threshold voltage of sleep transistor is 0.61V (*i.e.*, high V_{th} is 0.61V).

Furthermore, we do not force all types of functional units to have the same sleep transistor length. The reason is that: according to [14], we know the sleep transistor length has an impact on the sleep transistor efficiency³. Therefore, for each type of functional unit, we use the following two steps to determine its sleep transistor length: first, we perform circuit-level simulation by using Synopsys EPIC tool with respect to many combinations of sleep transistor lengths and sleep transistor widths; second, we choose the sleep transistor length with the consideration of sleep transistor efficiency. Table 2 gives the sleep transistor length of each type of functional unit used in our experiment.

Table 2. Sleep transistor length of each type of functional unit.

	ALU	Multiplier	Divisor	Selector	Comparator
Sleep Transistor Length	0.80 μm	0.60 μm	0.80 μm	0.60 μm	0.80 μm

Table 3. Sleep transistor widths of each type of functional unit.

Sleep Transistor Width	ALU	Multiplier	Divisor	Selector	Comparator
Largest	4.80 μm	3.00 μm	1.20 μm	4.20 μm	4.80 μm
Large	3.20 μm	2.40 μm	1.00 μm	3.00 μm	3.20 μm
Medium	1.60 μm	1.20 μm	0.80 μm	1.80 μm	1.60 μm
Small	0.80 μm	0.60 μm	0.60 μm	1.20 μm	0.80 μm
Smallest	0.40 μm	0.30 μm	0.40 μm	0.60 μm	0.40 μm

Next, we report the sleep transistor widths of each type of functional unit used in our experiment. Note, for the power-gated modules that are in the same type, we suppose they have the same sleep transistor length. Thus, for the power-gated modules that are in the same type, their differences are only in their sleep transistor widths. In our functional unit library, each type of functional unit has five different power-gated modules (*i.e.*, five different sleep transistor widths). Table 3 gives the five sleep transistor widths of each type of functional unit. For the convenience of presentation, we also use the five terms *Largest*, *Large*, *Medium*, *Small*, and *Smallest* to name these five sleep transistor widths.

Table 4 tabulates the delay and the standby leakage current of each power-gated module. We perform circuit-level simulation by using Synopsys EPIC tool to measure these values. The detailed methods are as below.

- Delay measurement. We use the following two steps to measure the delays. In the first step, we do not consider the sleep transistor. We use Synopsys PrimeTime to find the minimum delay path and the maximum delay path. Then, we use the pattern generation method [16] to derive the patterns for sensitizing these two paths. In the second step, we suppose that the sleep transistor is present. We make the following assumption: even if the sleep transistor is present, these patterns (derived in the first step) still cause the minimum delay and the maximum delay. Thus, by feeding these patterns, we can use circuit-level simulation to measure the minimum delay and the maximum delay.

³ In [14], the sleep transistor efficiency is defined as I_{ON}/I_{OFF} , where I_{ON} denotes the drain current when the sleep transistor is turned on, and I_{OFF} denotes the drain current when the sleep transistor is turned off.

Table 4. Functional unit library used in our experiment.

Sleep Transistor Width	ALU		Multiplier		Divisor		Selector		Comparator	
	Delay (ns) (min, max)	Leakage (nA)	Delay (ns) (min, max)	Leakage (nA)	Delay (ns) (min, max)	Leakage (nA)	Delay (ns) (min, max)	Leakage (nA)	Delay (ns) (min, max)	Leakage (nA)
Largest	(0.31, 3.88)	1.064	(0.13, 7.08)	0.821	(2.49, 8.40)	1.551	(0.16, 0.33)	1.201	(0.14, 1.91)	1.146
Large	(0.35, 3.92)	0.403	(0.34, 7.29)	0.412	(2.82, 38.73)	0.781	(0.19, 0.36)	0.758	(0.14, 1.91)	0.741
Medium	(0.49, 4.06)	0.093	(0.79, 7.74)	0.211	(4.79, 40.70)	0.215	(0.22, 0.39)	0.421	(0.17, 1.94)	0.221
Small	(0.71, 4.28)	0.040	(1.70, 8.65)	0.110	(7.81, 43.72)	0.095	(0.25, 0.42)	0.222	(0.31, 2.07)	0.042
Smallest	(1.21, 4.78)	0.013	(3.24, 10.19)	0.051	(15.05, 50.96)	0.039	(0.33, 0.49)	0.103	(0.50, 2.27)	0.014

- Standby leakage current measurement. We assume the value of each input is 0 in the standby mode. Thus, the standby leakage current can be measured through circuit-level simulation.

Nine benchmark circuits, including HAL, Autoregressive Filter (AR), Bandpass Filter (BF), Elliptic Wave Filter (EWF), R1, R2, IDCT1, IDCT2 and Motion, are used to test the effectiveness of our approach. Benchmark circuit HAL is adopted from [17]; benchmark circuit AR is adopted from [18]; benchmark circuit BF is adopted from [19]; benchmark circuit EWF is adopted from [20]; benchmark circuits R1 and R2 are adopted from [21]; and benchmark circuits IDCT1, IDCT2, and Motion are the representative functions adopted from the MediaBench suite [22]. For each benchmark circuit, the scheduled DFG is derived by the scheduling approach proposed in [17], the functional unit binding solution is derived by the left edge algorithm [23], and the register binding solution is derived by the approach proposed in [7].

Table 5 tabulates the characteristics of benchmark circuits. The column *#ops* gives the number of operations. The column *#vars* gives the number of variables. The column *#steps* gives the number of control steps. The column *Resource* gives 6-tuple (*#alus*, *#muls*, *#divs*, *#sels*, *#comps*, *#regs*), where *#alus*, *#muls*, *#divs*, *#sels*, *#comps*, and *#regs* are the number of ALUs, the number of multipliers, the number of divisors, the number of selectors, the number of comparators, and the number of registers, respectively. The column *Period* gives the target clock period.

Table 5. Characteristics of benchmark circuits.

Circuit	#ops	#vars	#steps	Resources	Period (ns)
HAL	11	8	4	(2, 2, 0, 0, 1, 4)	6.955
AR	28	30	8	(4, 4, 0, 0, 0, 8)	6.955
BF	29	30	8	(3, 2, 0, 0, 0, 6)	6.955
EWF	34	47	14	(4, 2, 0, 0, 0, 11)	6.955
R1	82	91	11	(7, 7, 0, 2, 3, 45)	6.955
R2	108	114	12	(8, 10, 0, 2, 2, 62)	6.955
IDCT1	48	60	12	(6, 3, 2, 0, 0, 24)	38.270
IDCT2	96	200	24	(9, 8, 2, 0, 0, 46)	38.090
Motion	290	430	42	(12, 15, 8, 2, 0, 190)	38.090

Table 6. Our experimental results and comparisons.

Circuit	Leakage (nA)			CPU Time (s)	
	Existing	Ours	Imp	Existing	Ours
HAL	1.682	1.273	24.3%	< 1	< 1
AR	3.336	2.927	12.3%	< 1	< 1
BF	1.681	1.272	24.3%	< 1	< 1
EWf	1.694	1.285	24.1%	< 1	< 1
R1	6.086	3.776	38.0%	< 1	< 1
R2	8.548	6.503	23.9%	< 1	< 1
IDCT1	3.333	1.997	40.1%	< 1	< 1
IDCT2	3.627	2.291	36.8%	< 1	< 1
Motion	13.535	8.191	39.5%	< 1	< 1

The platform of our experiment is a personal computer with AMD K8-4200+ CPU. We use Extended LINGO Release 10.0 as the MILP solver. Table 6 tabulates our experimental results. For the purpose of the comparisons, we also report the results of the existing design flow (*i.e.*, the two-step process presented in section 4.1). The column *Leakage* denotes the standby leakage current of the circuit. The column *Existing* denotes the existing design flow. The column *Ours* denotes our MILP approach. Benchmark data show that our approach can greatly reduce the standby leakage current. The column *Imp* denotes the relative improvement of our MILP approach over the existing design flow. Compared with the existing design flow, the average improvement of our approach achieves 29.3%. The column *CPU Time* denotes the CPU time in seconds. Both the CPU time of existing design flow and the CPU times of our approach are within 1 second.

7. CONCLUSIONS

In this paper, we present the first work to deal with the power gating of nonzero clock skew circuits. Given a target clock period, our objective is to minimize the standby leakage current of a circuit. We propose an MILP approach to formally formulate the simultaneous application of optimal clock skew scheduling and power-gated module selection. Compared with the existing design flow, experimental data show that the improvement of our approach achieves 29.3%.

The main limitation of our paper is that we assume the power gating of functional block is employed by a single sleep transistor. Our future work will extend our approach to the distributed sleep transistor network for further power reduction.

REFERENCES

1. J. P. Fishburn, "Clock skew optimization," *IEEE Transactions on Computers*, Vol. 39, 1990, pp. 945-951.
2. S. M. Burns, "Performance analysis and optimization of asynchronous circuits," Ph.D. Thesis, California Institute of Technology, Pasadena, California, U.S.A., 1991.
3. R. B. Deokar and S. S. Sapatnekar, "A graph-theoretic approach to clock skew op-

- timization,” in *Proceedings of IEEE International Symposium on Circuits and Systems*, Vol. 1, 1994, pp. 407-410.
4. C. Albrecht, B. Korte, J. Schietke, and J. Vygen, “Cycle time and slack optimization for VLSI chips,” in *Proceedings of IEEE/ACM International Conference on Computer Aided Design*, 1999, pp. 232-238.
 5. N. Maheshwari and S. S. Sapatnekar, *Timing Analysis and Optimization of Sequential Circuits*, Kluwer Academic Publishers, Boston, MA, U.S.A., 1999.
 6. D. Velenis, K. T. Tang, I. S. Kourtev, V. Adler, F. Baez, and E. G. Friedman, “Demonstration of speed and power enhancements on an industrial circuit through application of clock skew scheduling,” *Journal of Circuits, Systems and Computers*, Vol. 11, 2002, pp. 231-245.
 7. S. H. Huang, C. H. Cheng, Y. T. Nieh, and W. C. Yu, “Register binding for clock period minimization,” in *Proceedings of IEEE/ACM Design Automation Conference*, 2006, pp. 439-444.
 8. S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada, “1-V power supply high-speed digital circuit technology with multi-threshold voltage CMOS,” *IEEE Journal of Solid-State Circuits*, Vol. 30, 1995, pp. 847-854.
 9. J. Kao, S. Narendra, and A. Chandrakasan, “Transistor sizing issues and tool for multi-threshold CMOS technology,” in *Proceedings of IEEE/ACM Design Automation Conference*, 1997, pp. 409-414.
 10. J. Kao, S. Narendra, and A. Chandrakasan, “MTCMOS hierarchical sizing based on mutual exclusive discharge patterns,” in *Proceedings of IEEE/ACM Design Automation Conference*, 1998, pp. 495-500.
 11. M. Anis, S. Areibi, M. Mahmoud, and M. Elmasry, “Dynamic and leakage power reduction using an automated efficient gate clustering technique,” in *Proceedings of IEEE/ACM Design Automation Conference*, 2002, pp. 480-485.
 12. C. Long and L. He, “Distributed sleep transistor network for power reduction,” in *Proceedings of IEEE/ACM Design Automation Conference*, 2003, pp. 181-186.
 13. D. S. Chiou, S. H. Chen, S. C. Chang, and C. Yeh, “Timing driven power gating,” in *Proceedings of IEEE/ACM Design Automation Conference*, 2006, pp. 121-124.
 14. S. Kaijian and D. Howard, “Challenges in sleep transistor design and implementation in low-power designs,” in *Proceedings of IEEE/ACM Design Automation Conference*, 2006, pp. 113-116.
 15. D. S. Chiou, D. C. Juan, Y. T. Chen, and S. C. Chang, “Fine-grain sleep transistor sizing algorithm for leakage power minimization,” in *Proceedings of IEEE/ACM Design Automation Conference*, 2007, pp. 81-86.
 16. A. Krstic, Y. M. Jiang, and K. T. Cheng, “Pattern generation for delay testing and dynamic timing analysis considering power-supply noise effects,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 20, 2001, pp. 416-425.
 17. S. H. Huang and C. H. Cheng, “A formal approach to the slack driven scheduling problem in high level synthesis,” in *Proceedings of IEEE International Symposium on Circuits and Systems*, 2005, pp. 5633-5636.
 18. J. Ramanujam, S. Deshpande, J. Hong, and M. Kandemir, “A heuristic for clock selection in high-level synthesis,” in *Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference*, 2002, pp. 414-419.

19. C. A. Papachristou and H. Konuk, "A linear program driven scheduling and allocation method followed by an interconnect optimization algorithm," in *Proceedings of IEEE/ACM Design Automation Conference*, 1990, pp. 77-83.
20. M. Balakrishnan and P. Marwedel, "Integrated scheduling and binding: A synthesis approach for design space exploration," in *Proceedings of IEEE/ACM Design Automation Conference*, 1989, pp. 68-74.
21. S. H. Huang and C. H. Cheng, "An ILP approach to the simultaneous application of operation scheduling and power management," *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences*, Vol. E91-A, 2008, pp. 375-382.
22. C. Lee, M. Potkonjak, and W. H. Maggione-Smith, "MediaBench: A tool for evaluating and synthesizing multimedia and communications systems," in *Proceedings of IEEE International Symposium on Microarchitecture*, 1997, pp. 330-335.
23. F. J. Kurdahi and A. C. Parker, "REAL: A program for register allocation," in *Proceedings of IEEE/ACM Design Automation Conference*, 1987, pp. 210-215.



Shih-Hsu Huang (黃世旭) received the B.S. degree in Computer Science and Information Engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C., in 1989, the M.S. degree in Computer Science from National Tsing Hua University, Hsinchu, in 1991, and the Ph.D. degree in Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in 1995. From 1995 to 2000, he was with Computer and Communications Research Laboratories, Industrial Technology Research Institute, Hsinchu, rising to the position of deputy manager of IC design department, responsible for the design of high performance IC's. In 2000, he joined the department of Electronic Engineering, Chung Yuan Christian University, Chungli, Taiwan, as a faculty member, where he is currently a full Professor. Dr. Huang co-received the Most Popular Paper Award from the 18th VLSI Design/CAD Symposium, Taiwan, in 2007. His research interests include high-level synthesis, timing optimization, and clock tree synthesis.



Chun-Hua Cheng (程駿華) received the B.S. degree in Electronic Engineering from Chun Yuan Christian University, Chungli, Taiwan, R.O.C., in 2003, and the M.S. degree in Electronic Engineering from Chung Yuan Christian University, Chungli, Taiwan, in 2005. He is presently working toward the Ph.D. degree in Electronic Engineering at Chung Yuan Christian University, Chungli, Taiwan. Mr. Cheng co-received the Most Popular Paper Award from the 18th VLSI Design/CAD Symposium, Taiwan, in 2007. His research interests include timing optimization and high-level synthesis.



Da-Chen Tzeng (曾大誠) received the B.S. degree in Electrical Engineering from National Taiwan Ocean University, Keelung, Taiwan, R.O.C., in 2005, and the M.S. degree in Electronic Engineering from Chung Yuan Christian University, Chungli, Taiwan, R.O.C., in 2007. Mr. Tzeng co-received the Most Popular Paper Award from the 18th VLSI Design/CAD Symposium, Taiwan, in 2007. His research interests include low power design and high-level synthesis.