Retiming for High-performance Superconductive Circuits with Register Energy Minimization

Ting-Ru Lin and Massoud Pedram
University of Southern California, Los Angeles CA
<tingruli,pedram>@usc.edu

ABSTRACT

Retiming, which is a circuit transformation whereby registers are relocated to optimize performance, area, or energy consumption, has reached a high level of maturity in CMOS designs. However, the recent emergence and rapid rise of non-CMOS technologies are introducing new and important variants of the standard retiming problems. This paper presents a path-balancing retiming transformation, taking superconductive designs as an evaluation case study, where the retiming solution must achieve full path balancing of the circuit while simultaneously minimizing the energy consumption of inserted registers with performance constraints. This optimization problem, which is called a constrained register energy minimization (CREM) problem, is precisely formulated and polynomially solved. Next, the CREM problem formulation is extended to retime the circuit under a dual clocking architecture requiring partially (bounded depth difference) path balanced (PPB) characteristics only. It is shown that the PPB-CREM problem is NP-complete. We thus propose a polynomial-time approximation algorithm with a bounded error to solve this retiming variant. Compared to prior work, our approach reduces 38% of register count and 50% of register energy consumption of 14 benchmark circuits on average. Moreover, the competitive ratio of the register energy consumption between our approximate solution and the optimal solution is on average only 1.08.

1 INTRODUCTION

Retiming is a circuit transformation in which registers are relocated for the optimization of performance, area, or energy consumption in such a way that the functional behavior of the circuit remains the same [1]. Over the past decades, retiming has attained a significant level of maturity in CMOS designs for diverse applications including but not limited to testability [2, 3], logic re-synthesis [4, 5], circuit partitioning [6, 7], and physical planning [8]. However, with the approaching end of Moore's law, researchers have started exploring promising non-CMOS technologies for building high-performance and energy-efficient systems. The circuits built with non-CMOS technologies generally encompass an enormous number of registers or buffer cells due to the physical limitations of fundamental devices [9–11]. This unique characteristic motivates researchers to reevaluate the possibility of generalizing or even advancing retiming for evolving non-CMOS applications [12, 13].

The superconductive electronic (SCE) technology is one of the rapidly evolving non-CMOS technologies, which promises high performance and ultra-high energy efficiency. Josephson junctions (JJs), active switch devices in SCE, propagate single-flux-quantum (SFQ) pulses through logic cells in about 1 ps and dissipate only about 10^{-19} J per JJ switching [9]. The implementation of passive transmission lines (PTLs) for connecting SFQ cells promises a propagation speed 100 $\mu m/ps$ (a PTL connection requires an impedance-matched PTL driver on the source side and an impedance-matched PTL receiver on the sink side). For short connections, we can use Josephson Transmission lines (JTLs) with even faster propagation speeds. Higher than 20 GHz working frequencies for SFQ designs is thus ubiquitous, posing an advantage over CMOS designs [14]. Furthermore, the dynamic energy dissipation per SFQ INV cell is about 10^{-18} J whereas a minimum-size CMOS INV driving another identical INV in an industrial 12nm FinFET technology consumes about 10^{-15} J. We can attain zero static power consumption by implementing the design of energy-efficient rapid single flux quantum (ERSFQ) [15] or efficient single flux quantum (eSFQ) [16].

Standard logic cells in SFQ designs are known as clocked cells because they, unlike CMOS logic cells, need input clock signals to generate data signals at their outputs. Since the input data signals of any clocked cell must arrive within a target clock period (input signals that arrive in previous clock periods will be “consumed and forgotten" by the receiving cell), many registers are inserted between pairs of clocked cells to fully balance the path delays of the inputs to any SFQ logic cell. An SFQ circuit built with clocked cells and path-balancing registers may be considered as a fully wave-pipelined circuit with a large number of pipeline stages. To pursue optimal designs, we generalize the retiming transformation to tackle a constrained register energy minimization (CREM) problem, whose objective is to build a wave-pipelined circuit with minimal register energy consumption while meeting performance constraints. Many retiming algorithms have been proposed to address similar problems [1, 12, 14, 17–21], while none of them are tailored to the CREM problem. An obvious limitation of these algorithms is that they can only be performed after initial registers are inserted either by human effort or by some algorithms. However, the existing algorithms for initial register insertions for SFQ designs generally lack optimality guarantees for final synthesis results.

The research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government.

ICCAD '20, November 2–5, 2020, Virtual Event, USA

ACM ISBN 978-1-6654-2324-3/20/11…$15.00
https://doi.org/10.1145/3400302.3415659

This paper describes a path-balancing retiming transformation which integrates register insertions and a conventional retiming transformation to optimally solve the CREM problem. We abstract an arbitrary synthesized circuit without path-balancing registers as a directed graph model and then formulate the CREM problem as an integer linear programming (ILP) problem based on the graph model. The formulated ILP problem is shown to correspond to a well-known minimum cost flow problem with polynomial algorithms through dual problem transformation.

Next, we extend the CREM problem for building an arbitrary circuit under an advanced dual clocking architecture with an imbalance bound (i.e., converging paths should be path balanced only within the said imbalance bound – we can think of this factor as a positive slack on relative sequential depths of the said paths). We refer to such circuits as partially path balanced (PPB) circuits, and we denote this problem as PPB-CREM problem.

We prove the extended problem to be NP-complete by reducing the vertex cover problem to it in polynomial time and space. Thus, we present a polynomial-time approximation algorithm with a proven bounded error to efficiently solve the PPB-CREM problem. Given 14 benchmark circuits, our experiment results demonstrate that our approach reduces 38% of register count and 50% of register energy consumption on average compared to the prior work [13]. Moreover, the solution acquired by our approximation algorithm is on average only 1.08X away from the optimal solution.

The remainder of this paper is organized as follows. Section 2 provides background on SFQ circuits including design architectures; Section 3 describes system graph models; Section 4 and Section 5 detail path-balancing retiming for SFQ designs; Section 6 provides experimental results; Section 7 concludes.

2 BACKGROUND

2.1 Standard SFQ Cell Library

The standard SFQ cell library used in this paper is the one developed by Sunmagnetics [22] and adheres to the MIT-LL SFQ5ee process technology rules [23]. There are different types of cells in this library: DC2SFQ, SFQ2DC, PTL driver/receiver, NDRO (non-destructive readout flip-flop), Splitter, NOT, DFF, two-input AND, two-input OR, and two-input XOR. An SFQ cell can only drive one other cell because of the physical limitations of Josephson junctions (JJs) [9–11]. The first input of the NDRO cell is "set" whereas the second input is "reset". The splitter, a clockless cell, only receives data signals. Logic cells such as AND, NOT, OR, and XOR, known as clocked cells, receive both data signals and clock signals. Data signals appropriately change the internal state of the cell upon arrival at inputs of an SFQ cell. The clock signal produces an appropriate output signal while resetting the internal state of the cell back to its default state.

In this library, the bias DC current of each JJ is approximately 100 μA. When extra current flows into a biased JJ, the summation of its DC bias current and the extra current may exceed the critical current level of the JJ, causing the JJ to leave its superconductive state, emanating a quantum flux pulse of fixed magnetic flux value $\phi_0 = 2.0678 \times 10^{-15}$ Wb=Volt-second=Ampere-Henry. The JJ subsequently returns to its superconductive state. A JJ undergoing such a "leap" is called an active JJ. Given different input signals and a clock signal, the dynamic energy consumption of a cell is proportional to the number of active JJs in the cell during any clock period. More details are provided in Table 1.

Table 1: Standard SFQ Cell Library

Cells	Height (μm)	Width (μm)	#Inputs (+Clk)	#Outputs	Prop. Delay (ps)	Input Signals	Energy ($\times 10^{-19} J$)
Splitter	50	40	1 (+0)	2	5.7	(0) (1)	0 6.21
NOT	50	70	1 (+1)	1	13.0	(0) (1)	8.28 8.28
DFF	50	60	1 (+1)	1	6.8	(0) (1)	6.21 12.42
NDRO	50	90	2 (+1)	1	10.0	(0,0) (0,1) (1,0)	6.21 12.42 14.49
AND	50	70	2 (+1)	1	8.7	(0,0) (0,1) (1,0) (1,1)	8.28 10.35 10.35 16.56
OR	50	70	2 (+1)	1	6.0	(0,0) (0,1) (1,0) (1,1)	4.14 10.35 10.35 16.56
XOR	50	70	2 (+1)	1	6.3	(0,0) (0,1) (1,0) (1,1)	8.28 14.49 14.49 22.77

2.2 Single Clock Architecture

SFQ designs in a single clock architecture refer to the SFQ circuits with a single global clock input for all cells. In these circuits, clock signals are a steady periodic sequence of SFQ pulses with the clock period being defined as the inter arrival time of two consecutive pulses on the clock signal input. A pipeline stage is simply a clocked cell plus any signal splitters and JTL connections (or PTL connections with required PTL driver and receiver pairs). With this view of the a pipeline stage, SFQ circuits follow the same timing rule as pipeline CMOS circuits in that all input signals of any cell must arrive in the same clock period for correct operation. Take the operation of an SFQ AND cell as an example. A operation error occurs in an AND cell if its inputs with logic 1 value arrive in two different clock periods because the internal state of the AND cell is reset after every clock pulse. We describe the circuit built to operate correctly under the single clock architecture as a fully path balanced (FPB) circuit and define it formally as follows.

Definition 1. *A combinational circuit is a fully path balanced (FPB) circuit if the inputs for any clocked logic cell in the circuit arrive within the same clock period. In other words, the difference between the number of clocked cells on the shortest path and that on the longest path from the primary inputs to a clocked cell is zero and this condition holds for all clocked cells in the circuit.*

This definition suggests that any clocked cell in a FPB circuit can only receive output signals from the cells in its previous pipeline stage. Thus, many registers are typically inserted between pairs of clocked cells to attain correct logic behaviors. The number of these registers can approach the number of all other cells [12, 13]. We elaborate on the register requirement of a FPB circuit using Figure

1(a). Let the pipeline stage of *INV1* and *OR1* receiving *Inputs* be 1. As the inputs of *AND1* are provided by *INV1* and *OR1* (after passing through splitter *S1*), the pipeline stage of *AND1* is 2. Similarly, the pipeline stage of *INV2* is 3. $R1$ and $R2$ are inserted to balance the delays of the output signal from $S1$. Therefore, all signals from *Outputs* are generated by the cells in the 3^{rd} pipeline stage.

2.3 Dual Clock Architecture

As shown in [13], an SFQ circuit can be realized with fewer path-balancing registers by adopting a dual clock architecture with two clock sources: a fast clock and a slow clock. The frequency of the fast clock is Θ+1 times that of the slow clock (where $\Theta \in \mathbb{Z}^+ \cup \{0\}$.) The architecture requires that each set of primary inputs is repetitively fed for additional Θ cycles of a fast clock. As a result, each set of primary outputs can only be acquired after Θ+1 fast clock cycles which correspond to one cycle of a slow clock. This result suggests that the throughput of an SFQ circuit in the single clock architecture is Θ+1 times higher than the throughput of the circuit in the dual clock architecture. However, this is not a critical factor in many applications because the data processing throughput is generally limited by other micro-architectural considerations or data dependencies. The advantage of repeating inputs for Θ cycles is that the circuit can accommodate an *imbalance bound* of Θ which denotes the maximum difference between the number of clocked cells on the shortest path and that on the longest path from the primary inputs to any clocked cell. We describe the circuit built in a dual clock architecture as a partially path balanced (PPB) circuit and define it as follows:

Definition 2. *A combinational circuit is a partially path balanced (PPB) circuit if the input signals for any clocked logic cell in the circuit arrive within a window of* $\Theta + 1$ *clock cycles. In other words, the imbalance bound of the circuit is* Θ. *Evidently, the primary inputs to PPB circuits must be persistently present for* $\Theta + 1$ *clock cycles, and the output can be read only every* $\Theta + 1$ *clock cycles.*

Notice that a FPB circuit is a special case of a PPB circuit where $\Theta = 0$. We explain the details of a PPB circuit using Figure 1(b). The signal of the fast clock is sent to the clock input of all clocked cells. As shown in [13], we can partition an arbitrary PPB circuit into modules and each module consists of a repeat band, a logic block, and a mask band. The inputs of the logic block are repetitively fed by the outputs of the repeat band which consists of only NDRO cells. The reset and set inputs of each NDRO are fed by the slow clock and the output of the previous module, respectively. The logic block decides the functional behaviors of the circuit and we use the circuit in Figure 1(a) to represent the logic block. The mask band is formed by 2-input AND cells whose data inputs are fed by the slow clock and the outputs of the logic block. Thus, the outputs of a module are updated at the rate of the slow clock and therefore, they are the correct outputs of the whole circuit. Given Figure 1(b), if we repeat the inputs to the logic block for additional two times ($\Theta = 2$), we can remove all registers in the logic block.

Although the design approach for FPB circuits has matured, the design approach for PPB circuits is still in its infancy. Pasandi and Pedram [13] proposed a heuristic algorithm which is able to generate PPB circuits in time complexity of $O(|E|+|V|)$. Using their approach, the register count can be reduced by more than 50% for many benchmark circuits given Θ=4. However, the reductions may be as low as 10% in other benchmark circuits. These inconsistent reductions point to the necessity of developing a more rigorous approach to building optimal PPB circuits.

Since this paper addresses the optimization problem, the cost of the repeat and mask bands will not be considered since it is a fixed cost. Moreover, details of SFQ clock delivery networks will not be discussed here due to page limitation but they can be found in [24].

3 SYSTEM MODELS

This section defines the notation and terminology used in this paper and elaborates on path-balancing retiming models.

3.1 Preliminaries

We model a combinational circuit by a graph $G(V, E)$. Each vertex $v_i \in V$ represents a cell where its propagation delay is denoted by $d(v_i)$. Each direct edge $e_{i,j} \in E$ corresponds to a connection from the output of v_i to the input of v_j. The weight $w(e_{i,j})$ of $e_{i,j}$ denotes the number of registers on the connection. Note that we view each clocked cell as a composition of an "immobile" (permanently attached) register and a clockless cell [13]. Thus, if $e_{i,j}$ ends at a vertex which represents a clocked cell, value of $w(e_{i,j})$ is increased by 1. The expected energy consumption of a register along $e_{i,j}$ is calculated as follows:

$$\alpha_{i,j} = Pr^1_{i,j} E^1_r + Pr^0_{i,j} E^0_r, \tag{1}$$

where $Pr^1_{i,j}$ and $Pr^0_{i,j}$ are the probabilities of forwarding logic '1' and '0' values from v_i to v_j through $e_{i,j}$. E^1_r and E^0_r are the internal register energy consumptions of generating '1' and '0', respectively.

A path p from v_i to v_j is symbolized as $v_i \xrightarrow{p} v_j$. We use $w(p)$ to denote the sum of edge weights of path p. Similarly, we use $d(p)$ to denote the the sum of vertex delays of path p (inclusive of delays of v_i and v_j). Evidently, if there are two or more paths from v_i to v_j, all such paths are considered independently of each other. The clock period T of $G(V, E)$ is calculated as follows:

$$T = \max_{p: w(p)=0} d(p). \tag{2}$$

One can define $W(v_i, v_j) = \min_p w(p)$ for any path $p : v_i \rightarrow v_j$ and $D(v_i, v_j) = \max_p d(p)$ for any path p such that $w(p) = W(v_i, v_j)$. The former denotes the minimum of register counts along any path from v_i to v_j whereas the latter denotes the maximum (vertex) delay among all paths from v_i to v_j (inclusive of endpoints of the path) that have weight $W(v_i, v_j)$.

3.2 Path-Balancing Retiming Models

In the context of SFQ circuits, we define $s(v_j)$ as the pipeline stage (sequential depth) of v_j. If all inputs of v_j are fed by primary signals, $s(v_j)$ is 0 if v_j is a clockless cell; and $s(v_j)$ is 1 if v_j is a clocked cell. Otherwise, $s(v_j)$ is calculated as follows:

$$s(v_j) = \max_{v_i \in V,\ e_{i,j} \in E} s(v_i) + w(e_{i,j}). \tag{3}$$

We can interpret $s(v_j)$ as how many clock cycles v_j needs to wait for its immediate inputs to be ready after a set of primary input values is launched into the circuit. In our work, we denote a path-balancing retiming operation as an integer-valued edge-labelling function

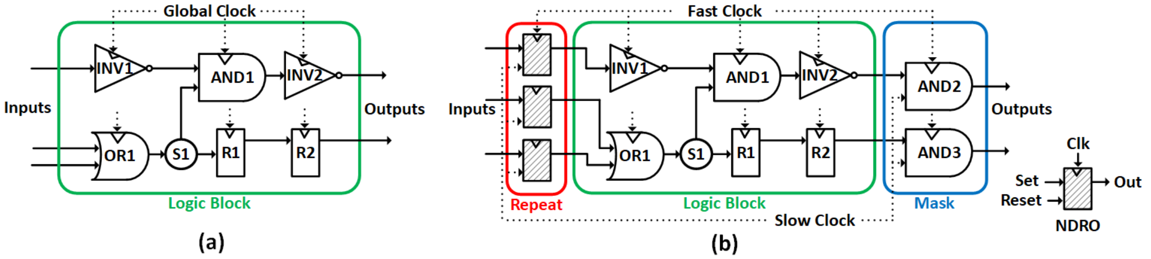

Figure 1: SFQ circuits with clocked cells. (a) a circuit in the single clock architecture. (b) a circuit in the dual clock architecture. S: splitter; R: register; and Gray rectangle: NDRO.

$w_r : E \rightarrow \mathbb{Z}^+ \cup \{0\}$ on $G(V,E)$. More precisely, the path-balancing retiming replaces $w(e_{i,j})$ in $G(V,E)$ by $w_r(e_{i,j})$ ($\geq w(e_{i,j})$) to form a new (fully or partially) path-balancing graph $G_r(V,E)$. See Lemma 1 for a formal exposition of the edge weight update function. The lower script "r" on some symbols (e.g. $s_r(v_j)$) indicates that the referred symbols are analogously defined on or derived from $G_r(V,E)$. Given an arbitrary synthesized circuit without any path-balancing registers, we aim to build the corresponding FPB/PPB circuit with minimal register energy while meeting performance constraints using path-balancing retiming. This problem is called the *constrained register energy minimization* (CREM) problem.

4 FULLY PATH BALANCED CIRCUITS

We provide details of building a FPB circuit by a path-balancing retiming transformation, CREM problem formulation, CREM problem complexity, and solution methods in this section.

4.1 Fully Path-Balancing Retiming

In this subsection, we prove the critical properties of FPB circuits and verify the validity of the proposed path-balancing retiming transformation following the definition in [26].

LEMMA 1. *Let $G_r(V,E)$ be a transformed version of $G(V,E)$ obtained by a path-balancing retiming transformation. $G_r(V,E)$ represents a FPB circuit if and only if $s_r(v_j) - s_r(v_i) = w_r(e_{i,j}),\ \forall e_{i,j} \in E$.*

PROOF. If $G_r(V,E)$ is a FPB circuit, all input signals of any vertex v_j must arrive within the $s_r(v_j)^{th}$ clock cycle. Since these input signals are generated from the previous vertices, we have $s_r(v_j) = s_r(v_i) + w_r(e_{i,j}),\ \forall e_{i,j} \in E$. Conversely, a circuit where $s_r(v_j) = s_r(v_i) + w_r(e_{i,j}),\ \forall e_{i,j} \in E$ exhibits the property that an input signal arriving at v_i within the $s_r(v_i)^{th}$ clock period propagates on $e_{i,j}$ and arrives at v_j within the $s_r(v_j)^{th}$ clock period where $s_r(v_i) + w_r(e_{i,j}) = s_r(v_j)$. This hold true for all inputs of any v_i's. Therefore, the circuit is a FPB circuit. □

LEMMA 2. *Let $G_r(V,E)$ be the FPB version of $G(V,E)$. The clock period of $G_r(V,E)$ is less than or equal to T_r if and only if $\forall v_i, v_j \in V$ such that $D_r(v_i,v_j) > T_r$, we have $W_r(v_i,v_j) = s_r(v_i) - s_r(v_j) \leq -1$.*

PROOF. If $G_r(V,E)$ is a FPB circuit with its clock period being less than or equal to T_r, then, from Equation 2, we have $w_r(p) \geq 1$ for any path $p : v_i \rightarrow v_j$ that satisfies $d_r(p) > T_r$. Since $G_r(V,E)$ represents a FPB circuit, $\forall p : v_i \rightarrow v_j$ we have $w_r(p) = W_r(v_i,v_j) = s_r(v_j) - s_r(v_i)$ based on Lemma 1, ensuring $d_r(p) = D_r(v_i,v_j)$. We thus have $s_r(v_i) - s_r(v_j) \leq -1,\ \forall v_i, v_j \in V$ such that $D_r(v_i,v_j) > T_r$. Conversely, if $G_r(V,E)$ is a FPB with $s_r(v_i) - s_r(v_j) \leq -1,\ \forall e_{i,j} \in E$ such that $D_r(v_i,v_j) > T_r$, the path weight of any path with $D_r(v_i,v_j) > T_r$ must be at least 1. Since there is no zero-weight path with delay larger than T_r, the clock period of $G_r(V,E)$ is less than or equal to T_r. □

DEFINITION 3. *Let C and C_r be two combinational circuits. Suppose that for every configuration c of C, there exists a configuration c_r of C_r such that when C is started in c and C_r is started in c_r, the two circuits exhibit the same behavior. The two circuits C and C_r are said to be equivalent.*

LEMMA 3. *Given $G(V,E)$, let C be a correct FPB circuit of G and C_r be a transformed FPB circuit obtained by using a fully path-balancing retiming transformation. C and C_r are equivalent.*

PROOF. We prove this lemma by an induction argument similar to that used in [26]. Let t_o be the maximum difference of the largest stage delays from a primary input to any vertex between C and C_r which may be represented as follows:

$$t_0 = \max_{v_i \in V} |s(v_i)T - s_r(v_i)T_r|,$$

where T and T_r denote the clock periods of C and C_r, respectively. Suppose both C and C_r start at time zero and run with an arbitrary sequence of inputs. For all $t \geq t_0$, we can find that the operation performed by a vertex v_i in C at time t is the same as that performed by v_i in C_r at time $t - s(v)T + s_r(v)T_r$ based on Lemma 1. Thus, the behaviors of C and C_r are indistinguishable from $t \geq t_0$. □

4.2 FPB-CREM Problem Formulation

We perform the path-balancing retiming for a combinational circuit modeled by $G(V,E)$ for solving the CREM problem with performance constraints which is formulated as follows.

Minimize:

$$\sum_{e_{i,j} \in E} \alpha_{i,j} w_r(e_{i,j}). \tag{4}$$

Subject to:

$$\begin{aligned}
&s_r(v_i) - s_r(v_j) \leq -1, && \forall v_i, v_j \in V,\ D_r(v_i,v_j) > T_r,\\
&s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) = 0, && \forall e_{i,j} \in E,\\
&- w_r(e_{i,j}) \leq -c_{i,j}, && \forall e_{i,j} \in E,\\
&s_r(v_i), w_r(e_{i,j}) \in \mathbb{Z}^+ \cup \{0\}, && \forall v_i \in V, e_{i,j} \in E.
\end{aligned}$$

The objective equation is the sum of the expected energy consumption of registers on all edges. The energy consumption of immobile registers is also included in the objective equation but is not reported because it is a constant. The first condition ensures that the clock period of the transformed circuit is equal to or less than T_r based on Lemma 2. The second condition guarantees that the transformed circuit is a correct FPB circuit given Lemma 1 and Lemma 3. The third condition sets the minimum number of the registers on edges.

4.3 FPB-CREM Problem Complexity

Herein, we prove that the described CREM problem is polynomially solvable and propose a solution method.

Theorem 1. *The CREM problem for building a FPB circuit with its clock period equal or less than T_r is a polynomially solvable problem.*

Proof. The second and third conditions of the CREM problem can be replaced by $s_r(v_i) - s_r(v_j) \leq -c_{i,j}, \ \forall v_i, v_j \in V, e_{i,j} \in E$ because $s_r(v_i) - s_r(v_j) = -w_r(e_{i,j}) \leq -c_{i,j}$. We then replace the objective function of $\sum_{e_{i,j} \in E} \alpha_{i,j} w_r(e_{i,j})$ with $\sum_{v_j \in V} [\sum_{v_i \in FI(v_j)} \alpha_{i,j} - \sum_{v_k \in FO(v_j)} \alpha_{j,k}] s_r(v_j)$ where FI/FO stands for fanin/fanout. The CREM problem is thus reformulated as follows.

Minimize:

$$\sum_{v_j \in V} [\sum_{v_i \in FI(v_j)} \alpha_{i,j} - \sum_{v_k \in FO(v_j)} \alpha_{j,k}] s_r(v_j). \tag{5}$$

Subject to:

$$\begin{aligned} & s_r(v_i) - s_r(v_j) \leq -1, && \forall v_i, v_j \in V : \ D_r(v_i, v_j) > T_r, \\ & s_r(v_i) - s_r(v_j) \leq -c_{i,j}, && \forall e_{i,j} \in E, \\ & s_r(v_i) \in \mathbb{Z}^+ \cup \{0\}, && \forall v_i \in V. \end{aligned}$$

The dual form of the reformulated problem is a minimum cost network flow problem with polynomial algorithms [1]. □

4.4 FPB-CREM Solution Method

The CREM problem can be solved by any solver for a minimum cost network flow problem. Values of pipeline stages are then derived from values returned by the solver. The dominant cost of this method is for solving the minimum cost network flow where the number of performance constraints is bounded by $O(|V|^2)$. Therefore, the CREM problem can be solved in $O((|E| + |V|^2)^2 log(|V|) + |V|(|E| + |V|^2) log^2(|V|))$ time [27], which is the same complexity as that of conventional constrained retiming algorithms [17].

5 PARTIALLY PATH BALANCED CIRCUITS

As the previous section, we elaborate on path-balancing retiming for building a PPB circuit before formulate the corresponding optimization problem and describe our solution methods.

5.1 Partially Path-Balancing Retiming

We prove a critical property of PPB circuits and verify the equivalence between FPB and PPB circuits.

Lemma 4. *Let $G_r(V, E)$ be the transformed version of $G(V, E)$ obtained by using a path-balancing retiming transformation. $G_r(V, E)$ is a PPB circuit with an imbalance bound of Θ if and only if $s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) \leq \Theta, \ \forall e_{i,j} \in E$.*

Proof. If $G_r(V, E)$ is a PPB circuit, the input signals of a vertex v_j are ready in the $s_r(v_j)^{th}$ clock period. Thus the arrival time of any input signals from v_i to v_j connected by $e_{i,j}$ must be after the $(s_r(v_j) - \Theta - 1)^{st}$ clock period. Thus, we have $s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) \leq \Theta, \ \forall e_{i,j} \in E$. Conversely, the graph model with $s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) \leq \Theta, \ \forall e_{i,j} \in E$ guarantees that the arrival time of all input signals of v_j from its previous vertices is equal to or larger than the $(s_r(v_j) - \Theta - 1)^{st}$ clock. The model is a PPB circuit. □

Lemma 5. *Given $G(V, E)$, let C be its corresponding correct FPB circuit and C_r be the transformed circuit with an imbalance bound of Θ. Then C and C_r are equivalent.*

Proof. We prove this lemma in the same way as what we did for Lemma 3. Let t_o be the maximum difference of the largest stage delay from a primary input to any vertex between C and C_r and is represented as follows:

$$t_0 = \max_{v_i \in V} |s(v_i)T - (s_r(v_i) + \Theta)T_r|.$$

Suppose C and C_r start at time zero and run with an arbitrary sequence of inputs. For all $t \geq t_0$, we find that the operation performed by a vertex v_i in C at time t is the same as the operation performed by v_i in C_r at time $t - s(v)T + (s_r(v) + \Theta)T_r$ based on Lemma 4. Thus, C and C_r behaviors are indistinguishable for $t \geq t_0$. □

5.2 PPB-CREM Problem Formulation

We extend the CREM problem for building a PPB circuit with minimal energy consumption while meeting performance constraints. Given an imbalance bound of Θ, the PPB-CREM problem is formulated as follows.

Minimize:

$$\sum_{e_{i,j} \in E} \alpha_{i,j} w_r(e_{i,j}). \tag{6}$$

Subject to:

$$\begin{aligned} & -W_r(v_i, v_j) \leq -1, && \forall v_i, v_j \in V : D_r(v_i, v_j) > T_r, \\ & s_r(v_i) + w_r(e_{i,j}) - s_r(v_j) \leq 0, && \forall e_{i,j} \in E, \\ & s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) \leq b_r(e_{i,j})\Theta, && \forall e_{i,j} \in E, \\ & \sum_{e_{i,j} \in E} b_r(e_{i,j}) \leq |FI(v_j)| - 1, && \forall v_j \in V, \\ & -w_r(e_{i,j}) \leq -c_{i,j}, && \forall e_{i,j} \in E, \\ & s_r(v_i), w_r(e_{i,j}) \in \mathbb{Z}^+ \cup \{0\}, && \forall v_i \in V, e_{i,j} \in E, \\ & b_r(e_{i,j}) \in \{0, 1\}, && \forall e_{i,j} \in E. \end{aligned}$$

The objective equation is the summation of the expected energy consumption of the registers on all edges. Similarly, the energy consumption of immobile registers is included in the objective equation but is not reported.

The first condition ensures that if the worst delay of the path(s) with the minimum path weight is larger than T_r, the path(s) starting from v_i to v_j have at least one register between their source and sink vertices. Note that the condition $W_r(v_i, v_j) = s_r(v_j) - s_r(v_i)$ only holds for FPB circuits. The second condition suggests that the pipeline stage of a vertex is equal to or larger than than that of its input vertex. The third condition indicates the input signals of a vertex can arrive within a window of $\Theta + 1$ clock cycles. $b_r(e_{i,j})$

is a binary variable that is set to 1 if the connection from v_i to v_j is not on any longest path from primary input to v_j. The fourth condition thus captures the fact that $s_r(v_j) - s_r(v_i) = w_r(e_{i,j})$ when the connection from v_i to v_j is along any longest path from primary input to v_j through the circuit; otherwise, we may have $s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) \leq \Theta$.

5.3 PPB-CREM Problem Complexity

THEOREM 2. *The proposed PPB-CREM problem for building a PPB circuit is NP-complete.*

PROOF. We prove that the PPB-CREM problem is NP-complete by a polynomial-time reduction of a NP-complete vertex cover problem (VCP). Given an undirected graph $G(V, E)$, VCP asks for a subset $S \subseteq V$ so every edge has at least one connecting vertex in S and is formulated as follows.

$$\begin{aligned} \min \quad & \sum_{v_i \in V} \beta_i y(v_i) \quad \text{subject to} \\ & y(v_i) + y(v_j) \geq 1, \qquad \forall e_{i,j} \in E, \\ & y(v_i) \in \{0, 1\}, \qquad \forall v_i \in V. \end{aligned}$$

$y(v_i) = 1$ if the vertex is in S. Otherwise, $y(v_i) = 0$.

Problem reduction: We create $G_r(V_r, E_r)$ as shown in Figure 2 to show how an arbitrary vertex cover problem is reduced to the PPB-CREM problem. In $G_r(V_r, E_r)$, we create a super-source vertex v_S and a super-terminal vertex v_T, the pair being connected by $e_{S,T}$ with a condition of $w_r(e_{S,T}) \geq 2$. For each vertex $v_i \in V$, we create $e^i_{k,l} \in E_r$ with its tail/head vertices and assign the register energy consumption along this edge as β_i. Such edge is marked as a solid edge in Figure 2. For each condition of $y(v_i) + y(v_j) \geq 1$ (let $i < j$ always be true), three edges are created in $G_r(V_r, E_r)$: one from v_S to the tail vertex of $e^i_{k,l}$, one from the head vertex of $e^i_{k,l}$ to the tail vertex of e^j, and the last one from the head vertex of $e^j_{k,l}$ to v_T. We assign the register energy consumption along these three edges as $\sum_{v_i \in V} \beta_i$. The edges created for the VCP condition are marked as the dotted edge in Figure 2. As a result, we reduce an arbitrary vertex cover problem based on $G(V, E)$ to the PPB-CREM problem with Θ=1 based on $G_r(V_r, E_r)$ without performance constraints.

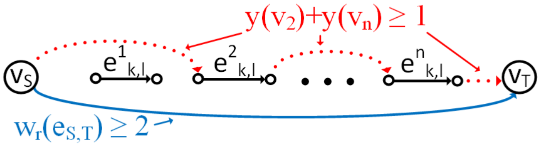

Figure 2: A directed retiming graph, $G_r(V_r, E_r)$. The edges related to the condition of $y(v_2) + y(v_n) \geq 1$ are marked as red dotted lines in this figure.

Reduction time analysis: The described reduction is done in $O(|V| + |E|)$ time and the solution acquired by solving the PPB-CREM problem can be mapped back to the VCP in $O(|V|)$ time. Thus, our problem is NP-complete. □

5.4 PPB-CREM Solution Method

We resort to an approximation algorithm for the PPB-CREM problem by replacing $-W_r(v_i, v_j) \leq -1$ with $s_r(v_i) - s_r(v_j) \leq -1$ and assigning value 1 to all $b_r(e_{i,j})$. The simplified problem is as follows.

Minimize:

$$\sum_{e_{i,j} \in E} \alpha_{i,j}(s_r(v_i) - s_r(v_i) + w_r(e_{i,j})). \tag{7}$$

Subject to:

$$\begin{aligned} & s_r(v_i) - s_r(v_j) \leq -1, \quad \forall v_i, v_j \in V,\ D_r(v_i, v_j) > T_r, \\ & s_r(v_i) + w_r(e_{i,j}) - s_r(v_j) \leq 0, \qquad \forall e_{i,j} \in E, \\ & s_r(v_j) - s_r(v_i) - w_r(e_{i,j}) \leq \Theta, \qquad \forall e_{i,j} \in E, \\ & s_r(v_i) - s_r(v_i) - w_r(e_{i,j}) \leq -c_{i,j}, \quad \forall e_{i,j} \in E, \\ & s_r(v_i) \in \mathbb{Z}^+ \cup \{0\}, \ \forall v_i \in V, e_{i,j} \in E, \end{aligned}$$

By treating $s_r(v_i) + w_r(e_{i,j})$ as one variable for each $e_{i,j}$, we can show that the dual form of the problem is a minimum cost flow problem which can be solved in $O((|E| + |V|^2)^2 log(|V|) + |V|(|E| + |V|^2)log^2(|V|))$ time [27].

For an arbitrary pair of v_i and v_j with $D_r(v_i, v_j) > T_r$, the first condition of the simplified problem cannot guarantee $W_r(v_i, v_j) \geq 1$. If $W_r(v_i, v_j) = 0$ but $s_r(v_i) - s_r(v_j) \leq -1$ (which is guaranteed by the first condition), we can locate at least one edge along any path from v_i to v_j such that the weight of the located edge can be increased by one while the pipeline stage values of its connecting vertices remain the same. To meet the performance constraints, we perform a graph search to increase the edge weight, if necessary. As a result, we can have $W_r(v_i, v_j) \geq 1, \forall v_i, v_j \in V, D_r(v_i, v_j) > T_r$ by adding no more than $|E|$ registers in $G_r(V, E)$ while keeping the existing pipeline stage of all vertices intact. Based on the second and the third conditions of the simplified problem, the value of $s_r(v_j)$ is in the range $[max_{e_{i,j}} s_r(v_i) + w_r(e_{i,j}), max_{e_{i,j}} s_r(v_i) + w_r(e_{i,j}) + \Theta]$. To apply the solutions to the PPB-CREM problem, we need to check if $s_r(v_j) = max_{e_{i,j}} s_r(v_i) + w_r(e_{i,j})$ holds for every vertex by performing a graph search again. If not, we evaluate the energy overhead of increasing the weight of different edges connected to v_j to achieve $s_r(v_j) = max_{e_{i,j}} s_r(v_i) + w_r(e_{i,j})$. We then choose the best action. No more than $\Theta|V|$ registers will be added to $G_r(V, E)$ to ensure correct stage values. Proofs for above statements are straightforward and hence omitted here.

The graph search for changing the edge weight for specified clock delays and setting correct stage values can be done in $O(|V||E|)$ time. Thus, the time complexity of the whole approximation algorithm is $O((|E| + |V|^2)^2 log(|V|) + |V|(|E| + |V|^2)log^2(|V|))$. The bounded error of our algorithm is $O(|E| + \Theta|V|)$ since the solution value of the objective function of the reformulated problem is less than or equal to that of the PPB-CREM problem.

6 EXPERIMENTAL RESULTS

To evaluate the proposed optimization framework, we synthesized Kogge-Stone adders (KSA), array multipliers (Mul), and integer dividers (IntDiv) using the SFQ logic synthesis tools from [12, 14]. Some of the ISCAS c-series and EPFL benchmarks were also synthesized. We used the best clock period acquired by prior approach of [12, 14] as our performance constraints. More details are specified in Table 2. The number of performance constraints for SFQ circuits is actually far less than $|V|^2$ because we just need to consider the paths which start from a cell to another cell without passing any clocked cells in between. Our framework is written in Python with the python optimization tool library and the hardware environment for generating the experimental results is a Linux machine with an Intel(R) Core(TM) CPU i7-6700 @3.40 GHz and a 16.0 GB RAM.

Table 2: Circuit Specification

Circuits	#Clocked Cells	#Clockless Cells	Cell Area (mm^2)	Cell Energy ($\times 10^{-19}$)	Clock (ps)
16-bit KSA	194	178	1.04	2335	20.1
32-bit KSA	469	437	2.52	5415	20.1
8-bit Mul	320	320	1.76	4025	14.4
16-bit Mul	1408	1408	7.74	18117	14.4
8-bit IntDiv	601	491	3.09	7963	25.8
16-bit IntDiv	2095	1684	10.7	27970	25.8
c499	350	309	1.84	5414	23.4
c1908	436	348	2.22	6304	23.1
c3540	1356	1054	6.85	16964	25.8
c6288	2121	1690	10.8	29119	22.8
Coding-cavlc	841	708	4.35	10921	22.8
Int2Float	274	227	1.41	3509	22.8
Sin	8283	7053	43.10	123654	25.8
Voter	10334	6980	50.00	134952	20.1

We start by comparing our approach to the state-of-the-art approach proposed in [12, 14] for building FPP circuits. Their approach resorts to a two-step method for achieving cell count minimization with performance optimization. Specifically, they utilize dynamic programming algorithms for minimizing the number of initial path-balancing registers as well as other cells. They then perform conventional constrained retiming algorithms for meeting performance constraints while minimizing the number of the path-balancing registers in the end [17]. Note that although each step guarantees the optimal result, the final result may not be optimal. In contrast to their approach, our retiming can perform register count minimization while meeting performance constraints without reliance on initial path-balancing registers.

Table 3 reports the number of the balancing registers and the register energy consumption. We do not report the running time because the time complexity of the prior approach [12, 14] is the same as our approach due to the constrained retiming algorithms (the running time of all circuits is less than 4 minutes and is just a few seconds for most circuits). As expected, all values are lower except *#Register* of 16-bit Mul because the lowest register energy can be acquired given different values of *#Register*. Based on the average ratios, we can see that the two-step procedure cannot promise optimal results. The main reason is that the first step of the dynamic programming algorithms restricts the number of movable path-balancing registers for the conventional retiming algorithms. We would like to emphasize that even though the energy reduction of about 5% compared to the reference is not that significant, it is still noteworthy because of the ultra-high operating frequency of SFQ circuits (e.g., 20 GHz and higher).

Table 3: Results for Building Fully Path Balanced Circuits

Circuits	#Registers		Register Energy ($\times 10^{-19}$)	
	FPB [12] +[14]	FPB	FPB [12] +[14]	FPB
16-bit KSA	220	206 (0.93)	1894	1806 (0.95)
32-bit KSA	580	522 (0.90)	4915	4550 (0.92)
16-bit Mul	3390	3391 (1.00)	30621	30621 (1.00)
16-bit IntDiv	15418	15140 (0.98)	134147	130904 (0.97)
c3540	1174	1117 (0.95)	10625	9978 (0.93)
c6288	3426	3393 (0.99)	31288	30645 (0.97)
Coding-cavlc	556	549 (0.98)	5059	4977 (0.98)
Sin	11954	11445 (0.96)	112030	106083 (0.94)
Average Ratio	-	0.96	-	0.95

Table 4 reports the results of building PPB circuits with Θ=1 using our approach and the state-of-the-art approach described in [13]. The state-of-the-art approach is a greedy-based approach with the time complexity of $O(|E|+|V|)$. The optimal results generated by ILP solvers are marked by PPB^{opt} as reference values. The optimal results of Sin and Voter benchmarks are not provided because they could not be generated in 24 hours. Although PPB circuits are developed for reducing register count, the prior approach could still result in a large number of registers when Θ is small because it is unaware of the influence of each register insertion on final results. Take circuit *Sin* as an example. *#Register* of *Sin* in PPB is more than that of *Sin* in FPB. The reason is that although many registers are removed in the early stage of this approach, far more registers are added in the late stage of the approach. If we compare the values of *#Register* and *Register Energy* between Table 3 and Table 4, we will find that the energy reduction reaches 22% on average after the implementation of a dual clock architecture using our approach. So does the reduction of *#Register*. No less than 30% reduction in *#Register* can even be attained for some circuits including 16-bit KSA, 32-bit KSA, c3540, and Coding-cavlc.

Based on the reported average ratios in Table 4, our approach reduces 38% of register count and 50% of register energy consumption compared to the prior approach [13]. Moreover, the register energy consumptions of 16-bit Mul and c6288 produced by the prior approach are more than 3X those of the optimal results. In contrast, our approximation algorithm produces results that are within 10% of the optimal results. These improvements demonstrate the benefits of developing rigorous approximation algorithms for building PPB circuits. Notice that our approximation algorithm has rather larger running times on some large circuits (e.g. 16-bit IntDiv) compared to the prior art approach. However, these running times are still acceptable because they are no more than a few minutes on the largest benchmarks. It is also worth stating that conventional retiming algorithms for area minimization with performance constraints have the same time complexity as our algorithm [1, 17].

We study the values in the fifth and the seventh columns in Table 4 to compare our results and optimal results when Θ=1. The register energy acquired by our approach is only 1.08X the optimal results on average even though the error is theoretically bounded by $O(|E| + \Theta|V|)$. Comparing the result acquired by our approach against the optimal result for the 16-bit IntDiv, we confirm that our proposed approach is feasible for large circuits which need more than 14,000 registers for path balancing. A relatively large ratio is observed for Coding-calvc, a part of a context adaptive variable-length coding (CAVLC) video encoder. This circuit contains look-up tables for coefficients, total zeros, trailing ones, and other signals, resulting in a relatively complex and non-repetitive circuit design [28]. Thus, many values of pipeline stages assigned by our minimum

Table 4: Results for Building Partially Path Balanced Circuits (Θ = 1)

Circuits	#Registers			Register Energy ($\times 10^{-19}$ J)			Run Time (s)		
	PPB^{opt}	PPB[13]	PPB	PPB^{opt}	PPB[13]	PPB	PPB^{opt}	PPB[13]	PPB
16-bit KSA	105	127 (1.20)	117 (1.11)	971	1131 (1.16)	1068 (1.09)	4.17	0.01	2.84
32-bit KSA	277	369 (1.33)	325 (1.17)	2563	3165 (1.23)	2915 (1.13)	10.3	0.02	6.66
8-bit Mul	630	1153 (1.83)	669 (1.06)	5663	11843 (2.09)	5936 (1.04)	5.86	0.02	5.48
16-bit Mul	3047	10427 (3.42)	3175 (1.04)	27937	110180 (3.94)	28899 (1.03)	162.4	0.11	97.2
8-bit IntDiv	1808	2712 (1.50)	1842 (1.01)	15692	35766 (2.27)	15950 (1.01)	11.2	0.04	10.3
16-bit IntDiv	14384	25551 (1.77)	14490 (1.00)	123971	412116 (3.32)	124793 (1.00)	648.2	0.18	245.6
c499	168	176 (1.04)	171 (1.01)	1564	2434 (1.55)	1592 (1.01)	4.90	0.01	4.59
c1908	540	622 (1.15)	594 (1.10)	5122	7366 (1.43)	5517 (1.07)	5.73	0.02	5.46
c3540	703	948 (1.34)	817 (1.16)	6234	11092 (1.77)	7059 (1.13)	800.5	0.05	18.7
c6288	2681	8757 (3.26)	2960 (1.10)	24327	131960 (5.42)	26549 (1.09)	65.5	0.11	50.0
Coding-cavlc	304	456 (1.50)	368 (1.21)	2749	4890 (1.77)	3258 (1.18)	21.5	0.03	17.6
Int2Float	139	192 (1.38)	175 (1.25)	1303	2028 (1.55)	1584 (1.21)	3.97	0.01	3.66
Sin*	-	22781 (-)	10055 (-)	-	306565 (-)	92034 (-)	>86400	0.52	5.36
Voter*	-	4335 (-)	4034 (-)	-	50456 (-)	33607 (-)	>86400	0.46	4.65
Average Ratio	-	1.72	1.10	-	2.29	1.08	-	-	-

*: The experiments were run on an Intel Xeon E5-2450 v2 CPU with 32GB RAM.

Table 5: Results for Building Partially Path Balanced Circuits (Θ = 2)

Circuits	#Registers			Register Energy ($\times 10^{-19}$ J)			Run Time (s)		
	PPB^{opt}	PPB[13]	PPB	PPB^{opt}	PPB[13]	PPB	PPB^{opt}	PPB[13]	PPB
16-bit KSA	78	83 (1.06)	95 (1.21)	726	763 (1.05)	863 (1.18)	5.01	0.01	3.28
32-bit KSA	217	252 (1.16)	277 (1.27)	2020	2257 (1.11)	2468 (1.22)	7.32	0.04	6.38
8-bit Mul	574	1050 (1.83)	635 (1.10)	5174	10802 (2.08)	5582 (1.07)	5.45	0.01	5.15
16-bit Mul	2926	9964 (3.40)	3172 (1.08)	26881	105484 (3.92)	28660 (1.06)	128.5	0.10	91.1
8-bit IntDiv	1668	2501 (1.50)	1721 (1.03)	14434	33393 (2.31)	14850 (1.02)	11.0	0.04	9.60
16-bit IntDiv	13923	24708 (1.77)	14132 (1.01)	119757	401869 (3.35)	121471 (1.01)	485.9	0.17	236.9
c499	136	144 (1.05)	136 (1.00)	1266	1937 (1.53)	1266 (1.00)	4.72	0.01	4.40
c1908	441	622 (1.41)	508 (1.18)	4171	6101 (1.46)	4758 (1.14)	7.51	0.01	4.98
c3540	506	706 (1.39)	639 (1.26)	4483	8140 (1.81)	5506 (1.22)	2074	0.05	17.7
c6288	2388	8228 (3.44)	2783 (1.16)	21727	124518 (5.73)	24914 (1.14)	13297	0.11	47.5
Coding-cavlc	219	323 (1.47)	285 (1.30)	1969	3433 (1.74)	2522 (1.28)	2741	0.03	16.64
Int2Float	99	132 (1.33)	124 (1.25)	900	1365 (1.51)	1111 (1.23)	6.12	0.01	3.65
Sin*	-	21341 (-)	9563 (-)	-	288708 (-)	87886 (-)	>86400	0.54	5.76
Voter*	-	2757 (-)	2874 (-)	-	33723 (-)	24813 (-)	>86400	0.36	5.66
Average Ratio	-	1.73	1.15	-	2.30	1.13	-	-	-

*: The experiments were run on an Intel Xeon E5-2450 v2 CPU with 32GB RAM.

cost flow solver have large errors. A similar circuit design is also found in Int2Float.

We further examine the bounded error of our approximation algorithm by setting Θ=2 and provide our results in Table 5. By comparing the values in Table 4 and Table 5 generated by our approach, we observe a further reduction in register count and register energy after the increase of Θ. The reduction achieved by our algorithm remains far more than that reported by the prior approach [13]. However, the register energy ratio between our results and optimal results increases from 1.08 to 1.13 when Θ increases from 1 to 2, which justifies the derived bounded error of $O(|E| + \Theta|V|)$. While the average competitive ratio between our results and optimal results does increase with the increase of Θ, we believe that this does not represent a significant drawback because the substantial throughput drop (recall that the throughput is inversely proportional to Θ+1) is generally not considered for most designs even when the peak target throughput is set by other design constraints.

7 CONCLUSION

We presented a constrained register energy minimization (CREM) problem in which register insertions are performed in the post-synthesis step for building a wave-pipelined circuit with minimal register energy consumption. We formulated this problem as a mathematical optimization problem and proved that it is polynomially solvable. Moreover, the CREM problem was extended to an advanced dual clock architecture as a PPB-CREM problem, which was proven to be a NP-complete problem. To tackle the PPB-CREM problem, we presented a polynomial-time algorithm with a proven bounded error. We evaluated the feasibility and robustness of our algorithm on 14 benchmark circuits. Experimental results showed that our approach reduces 38% of the register count and 50% of the register energy consumption compared to the state-of-the-art. The average ratio of the register energy consumption between our solutions and optimal solutions is only 1.08. The proposed formulations and algorithms have general applicability e.g., SFQ design, CMOS wave-pipelined circuits, and other (emerging) non-CMOS circuits.

REFERENCES

[1] C. E. Leiserson, and J. B. Saxe, “Retiming Synchronous Circuitry,” *Algorithmica*, vol. 6, pp. 5–35, June 1991.

[2] A. El-Maleh, T. E. Marchok, J. Rajski, and W. Maly, “Behavior and Testability Preservation under the Retiming Transformation,” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 16, no. 5, pp. 528–542, May 1997.

[3] S. Dey and S. Chakradhar, “Retiming Sequential Circuits to Enhance Testability,” in *Proc. IEEE VLSI Test Symposium*, pp. 28–33, Apr. 1994.

[4] R. K. Ranjan, V. Singhal, F. Somenzi, and R. K. Brayton, “On the Optimization Power of Retiming and Resynthesis Transformation,” in *IEEE Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 402–407, Nov. 1998.

[5] S. Malik, E. M. Sentovich, R. K. Brayton, and A. Sangiovanni-Vincentelli, “Retiming and Resynthesis: Optimizing Sequential Network with Combinational Techniques,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 10, no. 1, pp. 74–84, Jan. 1991.

[6] P. Peichen, Arvind K. Karandikar, and C. L. Liu, “Optimal Clock Period Clustering for Sequential Circuits with Retiming,” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 17, no. 6, pp. 489–498, June 1998.

[7] J. Cong, H. Li, and C. Wu, “Simultaneous Circuit Partitioning/Clustering with Retiming for Performance Optimization,” in *Proc. DAC*, pp. 460–465, June 1999.

[8] J. Cong and S. K. Lim, “Physical Planning with Retiming,” in *IEEE Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 2–7, Nov. 2000.

[9] O. A. Mukhanov, “Energy-Efficient Single Flux Quantum Technology,” *IEEE Trans. Appl. Supercond.*, vol. 21, no. 3, pp. 760–769, June 2011.

[10] O. Zografos, A. De Meester, E. Testa, M. Soeken, P.-E. Gaillardon, G. De Micheli, L. Amarù, P. Raghavan, “Wave pipelining for majority-based beyond-CMOS technologies,” in *Design, Automation and Test in Europe Conferene and Exhibition (DATE)*, pp. 1306–1311, May 2017.

[11] E. Testa, O. Zografos, M. Soeken, A. Vaysset, M. Manfrini, and Rudy Lauwereins, “Inverter Propagation and Fan-Out Constraints for Beyond-CMOS Majority-Based Technologies,” IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 164–169, July 2017.

[12] G. Pasandi and M. Pedram, “PBMap: A Path Balancing Technology Mapping Algorithm for Single Flux Quantum Logic Circuits,” *IEEE Trans. Appl. Supercond.*, June 2019.

[13] G. Pasandi and M. Pedram, “An Efficient Pipelined Architecture for Superconducting Single Flux Quantum Logic Circuits Utilizing Dual Clocks,” *IEEE Trans. Appl. Supercond.*, Mar. 2020.

[14] G. Pasandi and M. Pedram, “A Dynamic Programming-Based, Path Balancing Technology Mapping Algorithm Targeting Area Minimization,” in *IEEE Int. Conf. on Computer-Aided Design (ICCAD)*, Nov. 2019.

[15] A. F. Kirichenko, I. V. Vernik, J. A. Vivalda, R. T. Hunt, and D. T. Yohannes, "ERSFQ 8-Bit Parallel Adders as a Process Benchmark," *IEEE Trans. Appl. Supercond.*, vol. 25, no. 3, pp. 5–35, June 2015.

[16] M. H. Volkmann, I. V. Vernik, and O. A. Mukhanov, "Wave-Pipelined eSFQ Circuits," *Supercond. Sci. Technol.*, vol. 25, no. 3, June 2015, Art. no 1301005.

[17] N. Shenoy, “Retiming: Theory and practice,” *Integration, the VLSI Journal*, vol. 22, no. 1-2, pp. 1–21, Jan. 1997.

[18] P. Pan, “Continuous retiming: algorithms and applications,” in *Int. Conf. on Computer Design (ICCD)*, pp. 116–121, Oct. 1997.

[19] C. Chu, E. F. Y. Young, D. K. Y. Tong, and S. Dechu, “Retiming with Interconnect and Gate Delay,” in *IEEE Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 221–226, Jan. 2003.

[20] J. Monteiro, S. Devadas, and A. Ghosh, “Retiming Sequential Circuits for Low Power,” in *IEEE Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 398–402, Nov. 1993.

[21] A. P. Hurst, A. Mishchenko, and R. K. Brayton, “Fast Minimum-Register Retiming via Binary Maximum-Flow,” in *Formal Methods in Computer Aided Design (FMCAD)*, pp. 181–187, Nov. 2007.

[22] SUN Magnetics, RSFQlib, 2020. [Online]. Available: https://github.com/sunmagnetics/RSFQlib

[23] S. K. Tolpygo, V. B. Bolkhovsky, T. J. Weir, L. M. Johnson, Mark A. Gouker, and Willioam. D. Oliver, “Fabrication Process and Properties of Fully-Planarized Deep-Submicro Nb/Al-AlO$_x$/Nb Josephson Junctions for VLSI Circuits,” *IEEE Trans. Appl. Supercond.*, vol. 25, no. 3, June 2015.

[24] C. J. Fourie, “Digital Superconducting Electronics Design Tools—Status and Roadmap," *IEEE Trans. Appl. Supercond.*, vol. 28, no. 5, Aug. 2018, Art. no. 1300412.

[25] K. Gaj, E. G. Friedman, and M. J. Feldman, “Timing of multi-gigahertz rapid single flux quantum digital circuits," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 16, no. 2–3, pp. 247–276, 1997.

[26] C. E. Leiserson and J. B. Saxe, “Optimizing Synchronous Systems,” in *Proc. 22nd Annu. Symp. Foundations Comput. Sci. (FOCS)*, pp. 23–36, Oct. 1981.

[27] J. B. Orlin, “A Faster Strongly Polynomial Minimum Cost Flow Algorithm,” *J. Oper. Res.*, vol. 41, pp. 338–350, May 1993.

[28] L. Amaru, P.-E. Gaillardon, and G. D. Micheli, “The EPFL Combinational Benchmark Suite,” Int’l Workshop on Logic Synth. (IWLS), 2015.