A Low Energy Full TMR Design Method with Optimized Selection of Time/Space TMR Mode and Supply Voltage

Kazuhito Ito Yuki Hayashi

Department of Electrical and Electronic Systems
Saitama University
255 Shimookubo, Sakura-ku, Saitama, 338-8570, Japan
{hayashi,kazuhito}@elc.ees.saitama-u.ac.jp

Abstract— Triple modular redundancy (TMR) is to execute an operation three times and obtain the correct result by taking the majority of the three outputs. While TMR is effective in eliminating soft errors in LSIs, the overhead of area as well as the energy consumption is the problem. In addition to the space TMR mode, where the three copies of an operation are actually executed, the time TMR mode is available, where only two copies of an operation are executed and the results are compared, then if the results differ, the third copy is executed to get the correct result. With the time TMR mode, the penalty of energy consumption can be reduced. The drawback of time TMR is that it requires longer time duration. Appropriately selecting the power supply voltage is also an effective technique to reduce the energy consumption. In this paper, a method to derive a TMR design is proposed which selects the TMR mode and supply voltage for each operation to minimize the energy consumption within the time and area constraints.

I. Introduction

When a lay with large energy hits a semiconductor LSI, depending on the position of the hit, the value of a combinational circuit is inverted temporarily (a single event transition, SET), or the erroneous value is memorized in latches, registers, or memory cells (a single event upset, SEU). The possibility of occurrence of these soft error is increasing because more and more devices are integrated on a LSI with the advance of LSI manufacturing technology [1, 2, 3, 4, 5, 6].

To mitigate the soft error, triple modular redundancy (TMR) is well known where operations and data storages are tripled and the correct outcome is obtained by taking the majority of those tripled operations or storages [7, 8]. The drawback of TMR is the overhead of the area and the energy consumption for executing operations and storing data three times. The methods to reduce the overhead have been proposed. In [9], TMR is stopped to reduce the energy consumption when no error occurred. In [10], TMR is applied only to a part of the circuit so that the error immunity is maximized within the constrained overhead.

TMR is implemented in two ways. In the space TMR, three copies of an operation are actually executed and the majority is taken on these three copies. In the time TMR, only two copies of an operation are executed and the results are compared, then if the results differ, the third copy is executed to get the correct result. With the time TMR, the penalty of energy consumption can be reduced. The drawback of the time TMR is that it requires longer time duration.

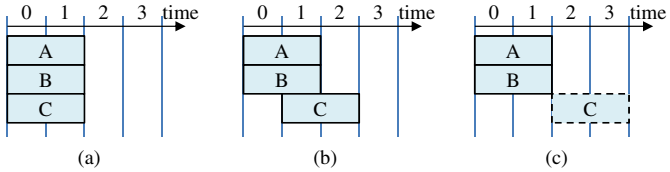

Fig. 1. (a)(b) schedules of executions A, B, and C of an operation where space TMR is necessary. (c) a schedule where time TMR is possible.

In CMOS technologies, the energy consumption of a circuit is reduced by lowering the supply voltage at the expense of increased delay [11, 12]. Appropriately selecting the power supply voltage is also an effective technique to reduce the energy consumption.

In this paper, a method to derive a TMR design is proposed which appropriately selects the TMR mode and supply voltage for each operation to minimize the energy consumption with respect to the given time and resource constraints.

II. Preliminary

A. Triple modular redundancy

TMR is to execute three copies of an operation and then the majority of the three results is taken. Let i_m denote the execution $m(=\{A,B,C\})$ of an operation i. When the result produced by i_A contains an error but the results by i_B and i_C are correct, the majority of these results is the correct one. Hence the error has been mitigated. Figure 1 shows the time chart of the execution of an operation i, where A, B, and C respectively denote the execution of i_A, i_B, and i_C. When all of i_A, i_B, and i_C are executed simultaneously as shown in Figs. 1(a) and 1(b), three FUs, one for each execution, are used. This is called the *space TMR (S-TMR) mode*. The S-TMR mode operation is shown in Fig. 2(a). To consider the error in the majority operation, the majority unit is also tripled. i_m receives data from a respective register, the result is fed to three majority units, and the output is stored in a respective register.

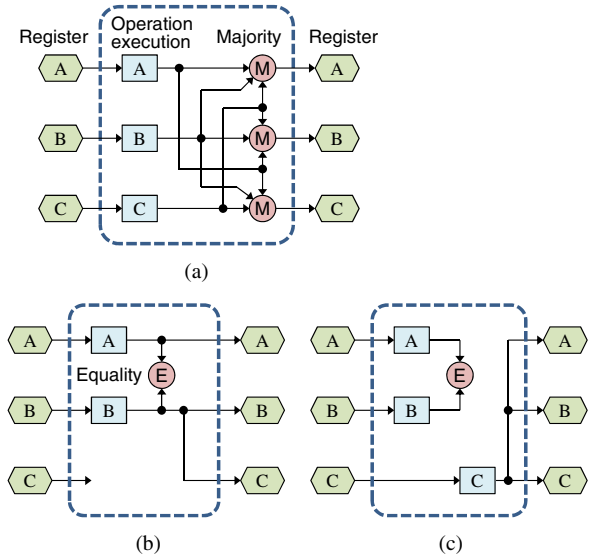

Fig. 2. TMR operations. (a) space TMR. (b) time TMR without error. (c) time TMR when an error ocurred.

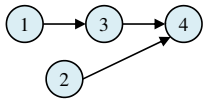

Fig. 3. An example DFG.

The three executions i_A, i_B, and i_C need not be executed in parallel. Figure 1(c) shows that one execution (i_C) starts after the other two executions (i_A and i_B) finished. In this situation, equality of the results of i_A and i_B can be checked before i_C starts. If the results are identical, those are correct. Thus the results are stored directly in a respective register. In addition, the result, either by i_A or by i_B, is used as the result of i_C and stored in a register corresponding to i_C as shown in Fig. 2(b). If the results of i_A and i_B differ, then i_C is executed and, assuming the result of i_C is correct, it is stored in registers as shown in Fig. 2(c). This is called the *time TMR (T-TMR) mode*. It is important to note that i_C is executed only when an error occurs in either i_A, i_B, or the comparator. Therefore the energy consumption for i_C can be saved in the T-TMR mode. A dotted box as shown in Fig. 1(c) indicates an execution which is necessary only in case of error.

B. *Motivating example*

While T-TMR mode saves energy consumption, it requires longer execution time than S-TMR mode because one execution of an operation must wait until other two executions of the operation finish. Therefore it is important to select which operation is in T-TMR mode to minimize the energy consumption of executing a processing algorithm.

Figure 3 shows a DFG of an example processing algorithm. There are 4 operations, 1, 2, 3, and 4, as denoted by nodes in the DFG. Data dependencies among operations are denoted by arcs. Assume these operations are of the same type for simplicity. First suppose an FU named PH is available which requires

TABLE I
FU SPECIFICATION FOR EXAMPLE

Name	Supply voltage	Duration	Energy
PH	VddH	1	1
PL	VddL	2	0.5

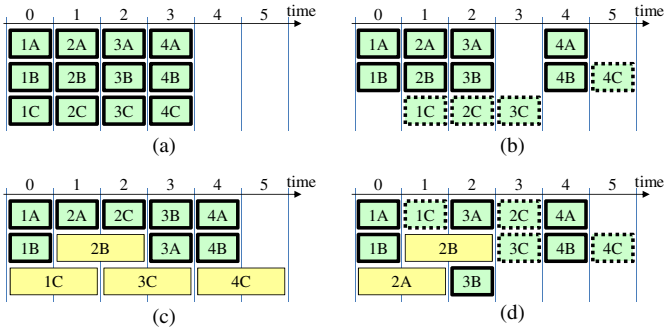

Fig. 4. TMR designs of Fig. 3. (a) time TMR with single (high) voltage. (b) time and space TMR with single voltage. (c) space TMR with dual voltages. (d) time and space TMR with dual voltage.

TABLE II
TMR DESIGNS OF FIG. 3

TMR mode	Supply voltage	Energy
space	VddH	12(100%)
space & time	VddH	8 (67%)
space	VddH & VddL	10 (83%)
space & time	VddH & VddL	7 (58%)

1 unit of time (u.t.) and consumes 1 unit of energy (u.e.) to execute an operation at the supply voltage VddH. Also assume the scheduling constraint such that all the operations complete within 6 u.t. (time constraint) and at most three FUs can be used (resource constraint). An operation schedule in Fig. 4(a) is obtained when only S-TMR mode is employed. The energy consumption is 12 u.e. Fig. 4(b) shows the schedule where T-TMR mode is considered. In Fig. 4(b), the execution C of an operation in T-TMR mode is shown as a dotted box. All the operations can be in T-TMR mode with the given time constraint and the energy consumption is 8 u.e. In CMOS technologies, the energy consumption of a circuit is reduced by lowering the supply voltage at the penalty of increased circuit delay. If an FU named PL with the lower supply voltage VddL, whose specification is shown in Table I, is available in addition to PH, the schedule in Fig. 4(c) is obtained with respect to the same resource constraint of 3 FUs (2 PHs and 1 PL) and only S-TMR mode is considered. The execution allocated to PL is shown as a thin box in Fig. 4(c). The energy consumption is 10 u.e. When both T-TMR mode and dual supply voltage are considered, the schedule shown in Fig. 4(d) is obtained with the energy consumption of 7 u.e. These designs are summarized in Table II.

This example suggests that the energy consumption of the design with TMR can be further minimized by considering both selection of S-TMR/T-TMR mode and multiple supply voltages.

III. Problem Formulation as MIP model

The energy consumption minimization problem for selecting TMR mode and the power supply voltage is formalized as a mixed integer programming (MIP) model. The following variables and parameters are employed. The objective is to minimize the energy consumption in Eq. (15) with respect to the constraints in Eqs. (1)–(3), (8), (9), and (10)–(14).

- DFG (N,E): a given processing algorithm is denoted as a data-flow graph (DFG) with the set of nodes N representing operations and the set of edges E representing data dependencies among operations.
- $X_{i,m}^{t,v}$: a binary variable and becomes 1 when i_m starts at time t and is assigned the voltage v.
- U_i^v: a binary variable and becomes 1 when i_C must be executed since the operation i is in S-TMR mode and the voltage v is assigned to i_C.
- $M_{i,m}^v$: a binary variable and becomes 1 when the operation i is in the S-TMR mode and a majority unit of the voltage v is used to generate the correct output of i_m.
- $LO_{i,m}^{vl,vh}$: a binary variable and becomes 1 when the level conversion from the lower voltage vl to the higher voltage vh is needed at the output of i_m.
- $LI_{i,j,m}^{vl,vh}$: a binary variable and becomes 1 when the level conversion from the lower voltage vl to the higher voltage vh is needed for the data from the output of operation i to the input of j_m.
- P_i: a binary variable and becomes 1 when the operation i is in the T-TMR mode and therefore a comparison is needed to compare the results of i_A and i_B.
- q_i^v: a constant indicating the operation duration of the operation i when it is assigned the voltage v.
- d_{ij}: a constant indicating the delay count on the data dependency edge from the operation i to the operation j.
- K_f^v: the maximum number of FUs of the operation type f and the voltage v.
- N_f: the set of operations of the operation type f.
- L_f^v: a constant indicating the duration for which an execution of an operation occupies an FU of the operation type f and the voltage v.
- T: a constant indicating the iteration period of the given processing algorithm.

The start time $T_{i,m}^S$ and the end time $T_{i,m}^E$ of i_m satisfy $T_{i,m}^E = T_{i,m}^S + q_i^v$ if a voltage v is assigned to i_m. Assume that, among the three executions i_A, i_B, and i_C of operation i, i_C ends last. That is, $T_{i,C}^E \geq T_{i,A}^E$ and $T_{i,C}^E \geq T_{i,B}^E$.

Operation execution

$$\sum_v \sum_t X_{i,m}^{t,v} = 1 \qquad \forall i, m = \{A,B,C\} \tag{1}$$

i_m is executed exactly once at some time at one of the voltages.

Precedence constraint

$$\sum_v \sum_t t X_{j,m}^{t,v} \geq \sum_v \sum_t (t + q_i^v) X_{i,C}^{t,v} - d_{ij}T \qquad \forall (i,j) \in E, m = \{A,B,C\} \tag{2}$$

If there exists a data dependency from an operation i to an operation j, then the precedence relation $T_{j,m}^S \geq T_{i,C}^E - d_{ij}T$ must be satisfied. $T_{j,m}^S$ is given as $\sum_v \sum_t t X_{j,m}^{t,v}$ and $T_{i,C}^E$ is given as $\sum_v \sum_t (t+q_i^v) X_{i,m}^{t,v}$. The term $d_{ij}T$ is introduced to take the inter-iteration data dependencies into account. It is sufficient to consider $T_{i,C}^E$ only because i_C is assume to end last among i_A, i_B, and i_C.

Judge TMR mode and ensure i_C ends last

$$\sum_v \sum_t (W - t - q_i^v) X_{i,m}^{t,v} + q_i^{v1} U_i^{v1} \geq \sum_t (W - t) X_{i,C}^{t,v1} \qquad \forall i, v1, m = \{A,B\} \tag{3}$$

If i_C starts before i_A and i_B finish, i.e., either $T_{i,C}^S < T_{i,A}^E$ or $T_{i,C}^S < T_{i,B}^E$, then the operation i is in S-TMR mode. On the other hand, if both i_A and i_B finish before i_C starts, i.e., both $T_{i,C}^S \geq T_{i,A}^E$ and $T_{i,C}^S \geq T_{i,B}^E$ hold, then the operation i can be in T-TMR mode. In that case, i_C is to be executed only when the results of i_A and i_B are not identical because of an error.

A binary variable $U_i^{v1} = 1$ indicates that i_C is to be executed at the voltage $v1$ when i is in S-TMR mode. When the voltage $v1$ is assigned to i_C, the relation

$$T_{i,C}^S \geq T_{i,m}^E - q_i^{v1} U_i^{v1} \tag{4}$$

must be satisfied for $m = \{A,B\}$. This relation is translated as follows: (1) if $T_{i,C}^S \geq T_{i,m}^E$ (the condition for T-TMR mode is met), U_i^{v1} can be 0; (2) if $T_{i,m}^E - q_i^{v1} \leq T_{i,C}^S < T_{i,m}^E$ (i_C starts before i_m ends and hence S-TMR mode is required), U_i^{v1} has to be 1; and (3) $T_{i,C}^S + q_i^{v1} < T_{i,m}^E$ (i_C ends before i_m ends) is prohibited.

With the binary variables defined above, the values are obtained as follows when the voltage $v1$ is assigned to i_C.

$$T_{i,C}^S = \sum_t t X_{i,C}^{t,v1} \tag{5}$$

$$T_{i,m}^E = \sum_v \sum_t (t + q_i^v) X_{i,m}^{t,v} \tag{6}$$

Then the relation in Eq. (4) is rewritten as

$$-\sum_v \sum_t (t + q_i^v) X_{i,m}^{t,v} + q_i^{v1} U_i^{v1} \geq -\sum_t t X_{i,C}^{t,v1}. \tag{7}$$

To take into account the condition that $U_i^{v1} = 1$ is required only when the voltage $v1$ is assigned to i_C, Eq. (7) is further rewritten as Eq. (3) where W is a sufficiently large positive constant. Eq. (3) imposes the same constraint as Eq. (4) when the voltage $v1$ is assigned to i_C. When the voltage other than $v1$ is assigned to i_C, the right hand side of Eq. (3) is 0 and therefore U_i^{v1} needs not be 1.

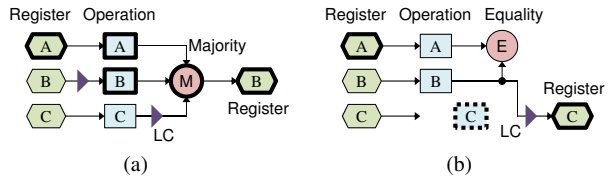

Fig. 5. Examples of need for level conversions. (a) In S-TMR mode. (b) In T-TMR mode.

Need for a majority unit for S-TMR mode

$$M^v_{i,m} \geq \sum_t X^{t,v}_{i,m} + \sum_v U^v_i - 1 \qquad \forall i,m,v \tag{8}$$

If an operation i is in space TRM mode, which is indicated by $\sum_v U^v_i = 1$, and the voltage v is assigned to i_m, a majority unit of the voltage v is needed to obtain the correct output of i_m for each of $m = \{A,B,C\}$.

Need for an output level conversion in S-TMR mode

$$LO^{vl,vh}_{i,m} \geq M^{vh}_{i,m1} + \sum_t X^{t,vl}_{i,m} - 1 \qquad \forall i,m,m1 \neq m, vh, vl < vh \tag{9}$$

The majority operation to get a correct result for i_m is assigned the same voltage as i_m. Therefore when the higher voltage is assigned to the majority unit for i_{m1} ($m1 \neq m$), a level conversion of the result of i_m is needed before it is input to the majority unit. For example in Fig. 5(a), the majority unit for the execution B is assigned the same higher supply voltage as the operation execution B and the operation execution C is assigned the the lower supply voltage. Thus a level conversion is needed from the operation execution C to the majority unit.

Need for an input level conversion

$$LI^{vl,vh}_{i,j,m} \geq \sum_t X^{t,vl}_{i,m} + \sum_t X^{t,vh}_{j,m} - 1 \qquad \forall (i,j) \in E, m = \{A,B\}, vh, vl < vh \tag{10}$$

$$LI^{vl,vh}_{i,j,C} \geq \sum_t X^{t,vl}_{i,C} + U^{vh}_j - 1 \qquad \forall (i,j) \in E, vh, vl < vh \tag{11}$$

The result of i_m is stored in a register at the same voltage as i_m. Therefore, when a data dependency (i,j) exists and the voltage of j_m is higher than the voltage of i_m, a level conversion is needed for $m = \{A,B\}$ For example in Fig. 5(a), the operation execution B is assigned the higher voltage and the input register B are assigned the the lower voltage. Thus a level conversion is needed from the input register B to the operation execution B. For the case of j_C, the level conversion is needed only when the operation j is to be executed in S-TMR mode (indicated by $U^{vh}_j = 1$). Hence Eq. (11) is used for $m = C$.

Need for a comparator in T-TMR mode

$$P_i \geq 1 - \sum_v U^v_i \qquad \forall i \tag{12}$$

In T-TMR mode, the results of i_A and i_B are compared with a comparator at the lowest voltage. When an operation i is in T-TMR mode, i.e., $\sum_v U^v_i = 0$, a comparator is needed for i.

Need for an output level conversion in T-TMR mode

$$LO^{vl,vh}_{i,B} \geq \sum_t X^{t,vl}_{i,A} + \sum_t X^{t,vl}_{i,B} + \sum_t X^{t,vh}_{i,C} - U^{vh}_i - 2 \qquad \forall i,m,m1 \neq m, vh, vl < vh \tag{13}$$

In T-TMR mode, if the results of i_A and i_B are identical, the result is also used as the result of i_C. If the voltage of i_C is higher than the voltages of i_A and i_B, then a level conversion is needed before copying the result. For example in Fig. 5(b), a level conversion is needed from the operation execution B at the lower supply voltage to the output register C at the higher supply voltage.

FU constraint

$$\sum_{i \in N_f} \sum_{m=\{A,B,C\}} \sum_{t'=0}^{L^v_f - 1} X^{t-t',v}_{i,m} \leq K^v_f \qquad \forall t,f,v \tag{14}$$

For each time t, operation type f, and voltage v, the number of operations executed simultaneously at time t should not exceed the specified constraint K^v_f.

Objective The objective is to minimize the energy consumption EC calculated as follows.

$$\begin{aligned} EC \;=\; & \sum_f \sum_v ECQ^v_f \sum_{i \in N_f} \left(\sum_{m=\{A,B\}} \sum_t X^{t,v}_{i,m} + U^v_i \right) \\ & + ECC \sum_i P_i + \sum_v ECM^v \sum_i \sum_{m=\{A,B,C\}} M^v_{i,m} \\ & + \sum_{vl,vh} ECL^{vl,vh} \sum_i \sum_{m=\{A,B,C\}} LO^{vl,vh}_{i,m} \\ & + \sum_{vl,vh} ECL^{vl,vh} \sum_{(i,j)} \sum_{m=\{A,B,C\}} LI^{vl,vh}_{i,j,m} \end{aligned} \tag{15}$$

ECQ^v_f, ECC, ECM^v, and $ECL^{vl,vh}$ are the energy consumption of one execution of an operation of type f at the voltage v, a comparison at the lowest voltage for T-TMR mode, a majority operation at the voltage v, and a level conversion from the voltage vl to vh, respectively.

IV. Solution with Scheduling Exploration

While the MIP model guarantees the solution optimality when it is solved, the CPU time to solve the problem increases as the problem size becomes larger. Sometimes even obtaining the first integer solution candidate consumes very long CPU time. The energy minimization problem for TMR design considered in this paper includes the scheduling of operations. In the MIP model, scheduling is modeled by allocating one binary variable to each possible clock cycle (CC) of operations. Therefore, the wider the possible CC ranges are, the more binary variables the MIP model employs and the longer CPU time is consumed.

A method to explore the scheduling of operations is proposed [13]. Every schedule must satisfy precedence constraints among operations in a given processing algorithm. The

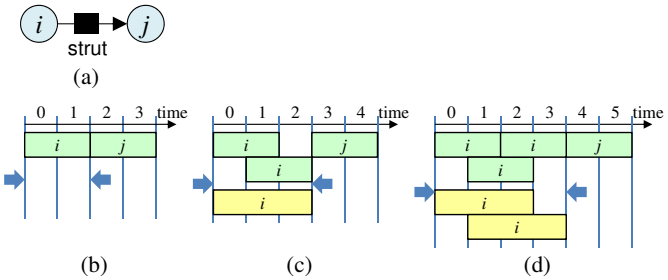

Fig. 6. Strut values and schedules. (a) a strut St on the edge (i, j). Possible schedules for (a) $St = 0$, (c) $St = 1$, and (d) $St = 2$.

'as soon as possible' (ASAP) schedule is obtained by solving the longest path problem on the DFG describing the given processing algorithm and is known to satisfy all the precedence constraints. The method [13] imposes appropriate delay time (called *strut*) onto edges between operations and then obtains the ASAP schedule. Different schedules are derived from different combinations of struts. Thus the optimal combination of struts is explored to obtain a schedule which results in the optimal solution to the given problem.

A. Applying Scheduling exploration to TMR design

Figure 6(a) shows a strut given to the edge (i, j) between operations i and j. Assume that the shortest operation duration of i is 2 u.t. and (i, j) is only the edge outgoing from i or incoming to j. The start time T_j^S of j is determined as $T_i^S + 2 + St$ by ASAP scheduling where St is the value of the strut on (i, j). Further assume that $T_i^S = 0$. When $St = 0$, the obtained schedule is as shown in Fig. 6(b). When $St = 1$, $T_j^S = 3$. If there exists an FU to execute i with the longer duration of 3 u.t. (at the lower supply voltage), then the FU can be used without violating the precedence constraint to j as shown in Fig. 6(c). In addition, i can start at time 1 as long as i is executed with the duration of 2 u.t. without violating precedence constraints. In the case of $St \leq 1$, only S-TMR mode is available to i, because i_A, i_B, and i_C have to be executed at the same time. When $St = 2$, the start time $T_j^S = 4$. The allowed start time of i is 0, 1, and 2 if i is allocated to a FU of the duration of 2 u.t. as shown in Fig. 6(d). In this case, T-TMR mode is possible if i_A and i_B are scheduled to start at time 0 and i_C to start at time 2.

The consequence is that, by considering the schedule obtained by the ASAP scheduling with struts as the lower bound of the operation start time, the strut value controls the assignment of the supply voltage and selection of TMR mode for operations. The lower supply voltage and/or T-TMR mode is allowed when the related strut is sufficiently large. Generally speaking, the larger the strut is, the wider the possible scheduling range of the operation is. The wider scheduling range results in more binary variable for the start time in MIP models. In time constrained scheduling, increasing a strut usually requires the other strut(s) to be decreased. Therefore, the total number of binary variables is kept small for any combination of struts for the moderate time constraints.

B. Scheduling exploration by simulated annealing

The simulated annealing (SA) algorithm is used to explore the combinations of the struts [13]. As a solution candidate, a combination of the struts is generated by the SA framework. The scheduling ranges of operations are computed based on the struts and an MIP model described in Sect. III is generated with respect to the scheduling ranges. The MIP model is solved by an MIP solver and its optimized solution cost is used as the cost function of the solution candidate.

For some combinations of struts, only the schedules might be derived where more operations than the constrained FU count K_f^v are executed at the same time. Although such combinations of struts are not feasible, these combination of struts should not be avoided in the exploration because accepting such combinations could be the escape from local optima. Thus, instead of prohibiting the solution candidates with excess FUs, a large penalty is given to the FUs exceeding K_f^v to allow such a solution being accepted. With integer variables Y_f^v, the constraint in Eq. (14) is rewritten as

$$\sum_{i \in N_f} \sum_{m=\{A,B,C\}} \sum_{t'=0}^{L_f^v - 1} X_{i,m}^{t-t',v} - Y_f^v \leq K_f^v. \qquad \forall t, f, v \qquad (16)$$

When FUs are required exceeding the constraint K_f^v, Y_f^v is set as the number of FUs exceeding K_f^v. In addition, the objective in Eq. (15) is changed as

$$EC' = EC + G \sum_f \sum_v Y_f^v \qquad (17)$$

where G is a sufficiently large positive constant. It is expected that minimizing the objective EC' in the exploration forces Y_f^v to zero in the final solution, which means the required number of FUs is within the constraint.

V. Experimental Results

A multi-threaded MIP solver IBM ILOG CPLEX 12.5.0.0 [14] was used to solve MIP models. A parallel SA method [15] was employed in implementing the schedule exploration. All the experiments were done on a PC with a 2.2GHz microprocessor running 4 threads on 4 physical cores and 8 GB of main memory. The processing algorithms used were the 5th order wave elliptic filter (WEF) [16] consisting of 8 multiplications and 26 additions, the WEF unfolded [18] by factor 3 (WEF3, 24 multiplications and 78 additions), and 8-point 1D DCT [17] (DCT) consisting of 11 multiplications and 29 additions.

Assuming a 0.18μm CMOS process as the target, the characteristics of the 16-bit functional units are summarized in Table III. In the experiments, only two levels of supply voltages were employed. The higher voltage is 1.8 V and the lower voltage is 1.2 V. Shown in Table III are from left to right, the name of the functional unit, the operation type (f), the power supply voltage, the operation duration in clock cycle (q), the duration for which an execution of the operation occupy an FU (L), and the average energy consumption in pJ. The operation duration

TABLE III
SPECIFICATION OF RESOURCES

ID	f	Vdd [V]	q	L	ECQ [pJ]
AH	addition	1.8(H)	1	1	3.9626
AL	addition	1.2(L)	2	2	1.7611
MH	multiplication	1.8(H)	2	1	44.949
ML	multiplication	1.2(L)	3	1	19.977
MaH	majority	1.8(H)	—	—	0.1607
MaL	majority	1.2(L)	—	—	0.0714
CL	comparison	1.2(L)	—	—	0.1210
LV	level conversion	1.2/1.8	—	—	0.5638

of the addition AH is 1 clock cycle. The addition at the lower voltage requires 2 clock cycles to execute and occupies an FU for 2 clock cycles. The multipliers are pipelined. The operation duration of MH is 2 clock cycles but the same FU is used for the next execution after 1 clock cycle. The multiplication ML is pipelined into 3 clock cycles. Thus the operation duration is 3 clock cycles and an FU (a multiplier at the lower voltage) is occupied for 1 clock cycle.

The results by solving the full MIP model are summarized in Table IV. The table shows the processing algorithm, the iteration period T specified as the time constraint, the resource constraint for FUs, the optimum energy consumption, and the CPU time. In the cases of WEF with $T = 25$ and $T = 30$ and WEF3, the solver was terminated manually when a sufficiently long time elapsed. The minimized energy consumption shown in the table might not be the optimum. For other cases, the MIP solver finished and the optimum solutions had been obtained.

Table V shows the results obtained by the exploration of the schedule. The penalty G for FUs exceeding the constraint was 200. In SA, the start and end temperatures were 100 and 1. The temperature was decreased by multiplying $\alpha = 0.95$ every 500 solution candidates were generated. For the case of the smaller DFG, the MIP solver successfully found the optimum solutions. For larger DFG and larger T, the exploration can find a better solution than MIP in a shorter CPU time.

TABLE IV
RESULTS BY FULL MIP SOLUTION

DFG	T	AH	AL	MH	ML	EC [pJ]	CPU [s]
WEF	21	5	4	4	4	793.7	495
WEF	25	5	4	2	2	676.0	(6261)
WEF	30	3	3	2	2	573.1	(6384)
WEF3	55	5	4	4	4	2816	(3909)
WEF3	60	6	4	3	2	3010	(3700)
DCT	8	12	4	6	4	1371	0.45
DCT	9	10	4	4	4	1228	125.6
DCT	10	8	4	4	4	1027	1001

TABLE V
RESULTS BY SCHEDULE EXPLORATION

DFG	T	AH	AL	MH	ML	EC [pJ]	CPU [s]
WEF	21	5	4	4	4	827.3	1610
WEF	25	5	4	2	2	685.3	1837
WEF	30	5	4	4	4	573.6	3574
WEF3	55	6	4	4	4	2765	1428
WEF3	60	6	4	3	2	2538	1874
DCT	8	12	4	6	4	1371	1184
DCT	9	10	4	4	4	1228	3046
DCT	10	8	4	4	4	1027	10949

VI. CONCLUSIONS

In this paper, a method to derive a TMR design is proposed which appropriately selects the TMR mode and supply voltage for each operation to minimize the energy consumption with respect to the given time and area constraints. The design problem is formalized as the MIP model. To support larger DFG and wider scheduling ranges, the exploration of schedule by simulated annealing is applied to solve the problem in a reasonable CPU time. Developing a heuristic algorithm for the problem remains as future work.

REFERENCES

[1] P. Hazucha and C. Svensson, "Impact of CMOS technology scaling on the atmospheric neutron soft error rate," IEEE Trans. Nucl. Sci., vol.47, no.6, pp.2586–2594, 2000.

[2] P. Hazucha and C. Svensson, "Cosmic ray neutron multiple-upset measurements in a 0.6-μm CMOS process," IEEE Trans. Nucl. Sci., vol.47, no.6, pp.2995–2602, 2000.

[3] J.A. Maestro and P. Reviriego, "Study of the effects of MBUs on the reliability of a 150 nm SRAM device," Proc. DAC 2008, pp.930–935, 2008.

[4] M.P. Baze, S.P. Buchner, and D. McMorrow, "A digital CMOS design technique for SEU hardening," IEEE Trans. Nucl. Sci., vol.47, pp.2603–2608, 2000.

[5] R. Garg and N. Jayakumar, "A design approach for radiation-hard digital electronics," Proc. DAC 2006, pp.773–778, 2006.

[6] R. Garg, C. Nagpal, and S.P. Khatri, "A fast, analytical estimator for the SEU-induced pulse width in combinational designs," Proc. DAC 2008, pp.918–923, 2008.

[7] S.W. Kwak and B.K. Kim, "Task-scheduling strategies for reliable TMR controllers using task grouping and assignment," IEEE Trans. Reliability, vol.49, pp.355–362, 2000.

[8] S. Golshan and E. Bozorgzadeh, "SEU-aware resource binding for modular redundancy based designs on FPGAs," Proc. DATE 2009, pp.1124–119, 2009.

[9] B.J. LaMeres and C. Gauer, "A power-efficient design approach to radiation hardened digital circuitry using dynamically selectable triple modulo redundancy," Military and Aerospace Programmable Logic Devices (MAPLD) Conference, pp.1–5, 2008.

[10] D. Tsuruta, M. Wakizaka, Y. Hara-Azumi, and S. Yamashita, "A TMR-based soft error mitigation technique with less area overhead in high-level synthesis," Proc. SASIMI 2012, pp.396–401, 2012.

[11] T. Kuroda and T. Sakurai, "Overview of low-power ULSI circuit techniques," IEICE Trans. Electron., vol.E78-C, no.4, pp.334–344, 1995.

[12] D. Chen, J. Cong, Y. Fan, and J. Xu, "Optimality study of resource binding with multi-Vdds," Proc. DAC 2006, pp.580–585, 2006.

[13] K. Ito and H. Seto, "Reducing power dissipation of data communications on LSI with scheduling exploration," IPSJ Trans. System Level Design Methodology, vol. 2, pp. 53-63, 2009.

[14] IBM ILOG CPLEX, http://www.ilog.com/.

[15] Y. Ota and K. Ito, "A parallel simulated annealing algorithm with look-ahead neighbor solution generation," Proc. SASIMI 2013.

[16] S.M. Heemstra de Groot, S.H. Gerez, and O.E. Herrmann, "Range-chart-guided iterative data-flow graph scheduling," IEEE Trans. Circuits Syst.-I: Fund. Theory & Appl., vol.39, pp.351–364, 1992.

[17] C. Loeffler, A. Ligtenberg, and G.S. Moschytz, "Practical fast 1-D DCT algorithms with 11 multiplications," Proc. IEEE ICASSP '89, pp.988–991, 1989.

[18] K.K. Parhi and D.G. Messerschmitt, "Static rate-optimal scheduling of iterative data-flow programs via optimum unfolding," IEEE Trans. Computers, vol.40, pp.178–195, 1991.