A Bipartition-Codec Architecture to Reduce Power in Pipelined Circuits

Shanq-Jang Ruan, Rung-Ji Shang, Feipei Lai, and Kun-Lin Tsai

***Abstract*—This paper proposes a new approach to synthesize pipelined circuits with low-power consideration. We treat each output value of a combinational circuit as one state of a finite-state machine (FSM). If the output of a combinational circuit transits mainly among some few states, we could extract those states (output) and the corresponding input to build a subcircuit. After bipartitioning the circuit, we apply the encoding technique to the highly active subcircuit for further power reduction. In this paper, we formulate the bipartition problem and present a probabilistic-driven algorithm to bipartition a circuit so as to minimize the power dissipation. Our experimental results show that an average power reduction on several Microelectronic Center of North Carolina (MCNC) benchmarks of 31.6% is achievable.**

***Index Terms*—Bipartition, low power, synthesis.**

I. Introduction

The objective of this paper is to address the problem of optimizing logic-level pipelined circuits for low power. Since the switching component accounts for over 90% of the total power dissipation, a lot of research activities focused on reducing the switching activity to save power [1]. Some research proposed new state encoding algorithms to reduce the Hamming distances between the binary representations of frequent transition pairs of states (e.g., [2]–[4]). Benini and Micheli [5] proposed gated clock to build low-power finite-state machine (FSM). They exploited the concept of self loop, an idle condition for a Moore machine. When the machine is in a self-loop condition, the next state is the same as the present state and the output does not change in the next cycle. Therefore, clocking the machine only wastes power. The technique obtains power saving by stopping the clock of the circuit during the self-loop period. FSMs with a lot of self loops can perform well using the gated-clock approach, but some FSMs that transit mainly among some states cannot benefit from these methods. Chow *et al.* [6] decomposed an FSM into a number of coupled submachines so that some state transitions of high probability will be confined to the smaller submachines most of the time. Alidina *et al.* [7] proposed two precomputation architectures for low-power sequential logic. Their approaches have much power saving effect if few input pins can determine the correct output values. However, finding an appropriate observability "don't care" (ODC) for the precomputation logic is not always possible. Choi and Hwang [8] partitioned a circuit into multiple subcircuits through recursive application of Shannon expansion with respect to the selected input variables. The power improvement of this scheme is determined by properly selecting input variables. However, the most frequently active subcircuit is not necessarily the smallest one.

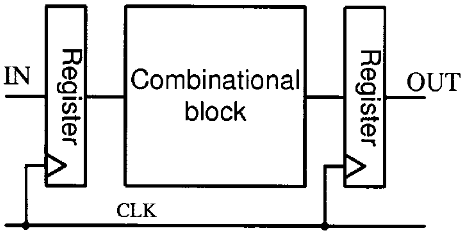


Fig. 1. Combinational pipelined circuit.

Previous work in the area of partition for low power has been focused on the logic variable analysis [7], [8]. Contrary to these approaches, we bipartition a circuit in terms of output probability. By observing the behavior of combinational circuits, we found that only a few output often appear in the circuit. We treat each different output as a corresponding state of an FSM and transform the combinational circuit into a state transition graph (STG). Then the bipartition algorithm is exploited to partition the STG into two STGs: one containing few states of high activity and the other containing many states of low activity. Obviously, the former will be active at most of the time and dominate the power dissipation. Furthermore, we use the codec architecture to reduce the power dissipation in the most active circuit block. The power dissipation benefit is due to the following three reasons. First, the lengths of the registers, which are used to store the output of each stage, are reduced after encoding. Second, the Hamming distance of the register switching is smaller than before. Finally, the circuit switching activity of the combinational block is also reduced. We have shown in [9] that the power consumption of our bipartition-codec architecture can be reduced. In this paper, we demonstrate the bipartition and bipartiton-codec approaches for power saving.

The remainder of this paper is organized as follows. In Section II, we briefly describe the bipartition and bipartition-codec architectures. Our bipartition algorithm is described in Section III, where the power dissipation model of bipartition-codec architecture is described and we formulate the bipartition problem so as to propose the algorithm for achieving power reduction. In Section IV, the experimental results are presented, which confirms the effectiveness of our new architecture. Finally, we summarize the conclusion in Section V.

II. Bipartition-Codec Architecture

To illustrate this architecture, let us consider a combinational block that is separated by distinct registers as shown in Fig. 1. Fig. 2 is the bipartition architecture without codec, where the global control block (GCB) selects which group is enabled and only one group is active in each cycle. After bipartitioning a circuit, we apply the encoding technique to the highly active subcircuit (Group_1). We replace the small block with a codec architecture, which consists of an encoder and a decoder, to reduce the internal switching activity of the block, as shown in Fig. 3. Notice that the Encoder not only encodes the output with minimal Hamming distance, but generates the selection (SEL) signal for alternative path. The principle of bipatition-codec architecture can be described as follows. The input IN feeds into the Encoder and the registers of Group_2. The SEL signal and encoding output become valid before the rising edge of the global clock CLK. If SEL = 1, the gated clock CLK_1 will be activated and CLK_2 stopped at the next rising edge of CLK. At this moment the encoding output passes through register R_1 and propagates into the decoder. The input IN will not feed through

Manuscript received December 1, 1999; revised June 13, 2000. This work was supported in part by the National Science Council of Taiwan under Grant 89-2213-E-002-138. This paper was recommended by Associate Editor E. Macii.

S.-J. Ruan is with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C.

R.-J. Shang and K.-L. Tsai are with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, R.O.C..

F. Lai is with the Department of Electrical Engineering and the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, R.O.C.

Publisher Item Identifier S 0278-0070(01)00951-4.

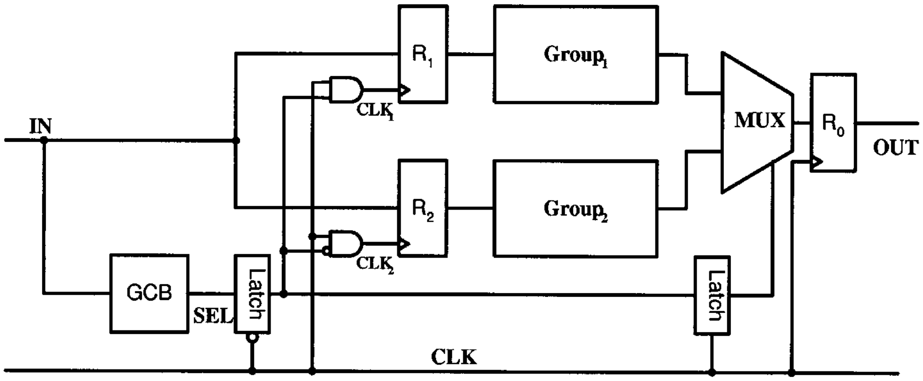


Fig. 2. Bipartition architecture.

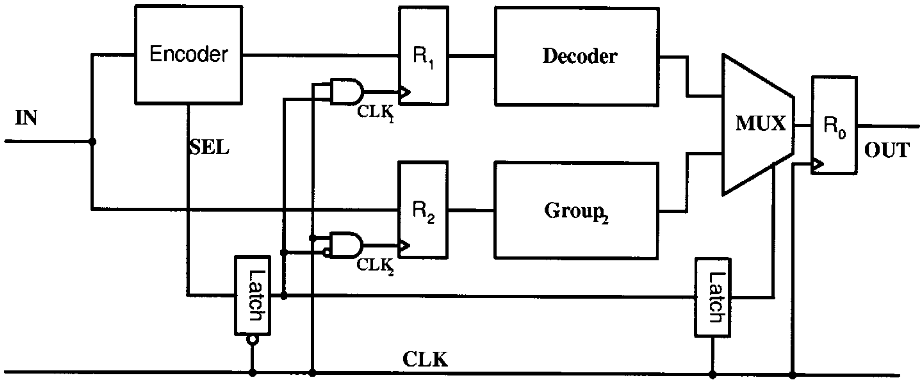


Fig. 3. Bipartition-codec architecture.

R_2. Hence, the Decoder is selected to compute output while $Group_2$ is idle. The interested reader may refer to [9].

III. Bipartition Algorithm

A. Power Dissipation Model for Bipartition-Codec Architecture

Consider the proposed architecture in Fig. 3. Whenever the input signals switch, the Encoder encodes the output and computes a new value SEL for selecting which one of the two groups will be active. The other additional components are needed for a correct behavior of the circuit. Thus, the average power dissipation of bipartition-codec architecture can be expressed as the following:

$$\begin{aligned} Power_{\text{avg}} &= Power(Encoder) \\ &\quad + Prob(Group_1) * Power(Decoder) \\ &\quad + Prob(Group_2) * Power(Group_2) \\ &\quad + Power(control) \end{aligned} \tag{1}$$

with constraint

$$Prob(Group_1) + Prob(Group_2) = 1$$

where $Power_{\text{avg}}$ stands for the average power dissipation, $Prob(Group_1)$ *and* $Prob(Group_2)$ are the sums of state probabilities in $Group_1$ and $Group_2$, respectively, *Power*(*Encoder*) and *Power*(*Decoder*) are the average power dissipation of Encoder and Decoder, and *Power*(*control*) is the average power dissipation of the two latches, two AND gates, and multiplexers in Fig. 3, which can be regarded as a constant here. *Prob*(*Group*$_1$) and *Prob*(*Group*$_2$) depend on the probabilities of selected states. *Power*(*Encoder*), *Power*(*Decoder*) and *Power*(*Group*$_2$) depend on both the logic function being performed and the primary input. Clearly, the logic function and primary input can also be determined by the states we selected. It is possible that adding states increases the power dissipation. Therefore, the power dissipation of our architecture strongly depends on the choice of states.

B. Problem Formulation

The problem of bipartition can be regarded as to bipartite a circuit such that it dissipates the minimum power. Suppose we have an FSM of n different states. Then there are $2^n - 2$ different bipartitions. Hence, developing a simple model for solving the bipartition problem is important for our architecture.

According to the previous discussion of our power model, it is a complex tradeoff in moving states between $Group_1$ and $Group_2$ because moving a state will change the areas and probabilities of both groups. In order to reduce the size of the problem, we ignore the power effect of $Group_2$ since the probability of $Group_2$ is significantly smaller than that of $Group_1$, as suggested by the experimental results presented in [10]. Therefore, we simplify the problem as follows. First, we wish to include as many states of high probabilities as we can in $Group_1$. Second, the gate counts of $Group_1$ after synthesis should be only a small fraction of the circuit. Some important questions arise: 1) how large a state probability can be treated as a “high-state probability” and 2) the gate count of a circuit is hard to estimate before synthesis. For selecting the proper states, we employ an average probability as a threshold for deciding the state probability as high or low. The average probability is defined as

$$\text{Average probability} = 1/n$$

where n is the total state number of the given FSM. If a state probability is larger than the average probability, it is worth adding it to $Group_1$.

```
Bipartition ( S = {S1, S2, ...,Sn})
{
        MinProb = 0.5;
        AvgProb = $1/n$;
        MaxStateNo = $\lceil n/2 \rceil$;
        $G_1$ = NULL; $PG_1$ = 0; G1_State_No = 0;
        while (G1_State_No $\leq$ MaxStateNo ){
              $S_i$ = sel_max_Prob(S);
              if (Prob($S_i$) $\geq$ AvgProb){
                    S = S - $S_i$;
                    $G_1$ = $G_1 \cup S_i$;
                    $PG_1$ = $PG_1$ + Prob($S_i$);
                    G1_State_No = G1_State_No + 1;
              }
              else
                    break;
        }
        while($PG_1$ < MinProb){
              NewState = sel_max_Prob(S);
              $G_1$ = $G_1 \cup$ NewState;
              $PG_1$ = $PG_1$ + Prob(NewState);
        }
        return $G_1$;
}
```

Fig. 4. Bipartition algorithm.

The strategy implies that the newly added states have to contribute substantial probability gains for Group$_1$.

Since the gate count of Group$_1$ is hard to estimate before synthesis, we assume that there is a correlation between the number of states and the power dissipated in the final implementation. The less number of states we select, the more *don't care* conditions we have. The size of a cover is the number of its implicants. *Don't care* conditions can be effectively used to reduce the size of a cover of an incompletely specified function [11]. The experimental results of [10] suggest the assumption by adjusting α for including states. Consequently, the correlation of the gate counts and the numbers of states is assumed in proportion.

C. Heuristic Algorithm

The previous algorithm proposed in [9] would include more than half of the total state number in Group$_1$. The area of Group$_1$ may be larger than Group$_2$. This implies that the highly active subcircuit consumes much power. Consequently, for example, the architecture will not have any power-saving effect. Although the phenomenon did not happen in the previous publication, we found this situation in the later simulation and did modification accordingly, as shown in Fig. 4. The input probabilities of the current stage are derived from the previous stage. The input of our algorithm is a set of individual states $S_1, S_2, \ldots, S_n$. As *MinProb* is set to 0.5, it forces the probability of Group$_1$ to be greater than 0.5 after applying the algorithm. This guarantees Group$_1$ to be active during more than the half of the execution time. *AvgProb* is the ratio $1/n$ that we discussed in previous subsection. *MaxStateNo* is the maximum number of states in Group$_1$ and is set to $\lceil n/2 \rceil$. Function *sel_max_Prob* returns the state in S with the highest probability. We initialize the set G_1 to null and the probability of G_1, PG_1 and the state number of Group$_1$ $G1$*_State_*No to zero. The first while loop expands the G_1 set by adding the states whose probabilities are greater than *AvgProb*. Note this while loop exits on the following conditions: first, all states that satisfy the *AvgProb* constraint have been selected, but $G1$*_State_*No is less than *MaxStateNo*. Further, the loop exits if all states left in S have a state probability less than *AvgProb*. In our experiments, the first while loop always exits due to the high cluster characteristic. The second while loop continues to select states until the constraint on *MinProb* is satisfied. Otherwise, a circuit could not obtain good power-saving effect due to its small group probability. In most cases, the second while loop will be skipped since most benchmarks have cluster and self-loop characteristic. The algorithm runs in linear time in the number of states.

TABLE I
PROBABILITIES OF BIPARTITION

Circuits	Input	Output	Group1		Group2	
	No.	No.	State No.	PG1	State No.	PG2
sao2	10	4	3	0.966	7	0.034
misex3	14	14	2	0.808	1039	0.192
table3	14	14	4	0.925	66	0.075
misex1	8	7	2	0.625	9	0.375
table5	17	15	4	0.906	56	0.094
sqrt8	8	4	8	0.75	8	0.25
con1	7	2	2	0.688	2	0.312
rd84	8	4	3	0.711	6	0.289
bw	5	8	7	0.5	16	0.5
b12	15	9	6	0.53	74	0.47

It should be noted that the bipartition algorithm has two scenarios. Consider the first while loop. When the algorithm selects states up to the *MaxStateNo*, the probability of G_1 would be greater than or equal to 0.5 in this situation. The second while loop expands the G_1 until the state probability of G_1 is greater than *MinProb*. However, the state number will always be less than $\lceil n/2 \rceil$ after exiting this loop.

D. State Encoding

At the gate level of abstraction, the average power dissipation is proportional to the average switching activity. Hence, a good state assignment algorithm should significantly reduce the switching activity of Group$_1$. Many state assignment algorithms, such as [2]–[4], minimize the Hamming distance among the codes of the states of high-transition probability. We adopted the heuristic algorithm proposed by Benini and Micheli [4] to encode the states in our architecture .

IV. Experimental Results

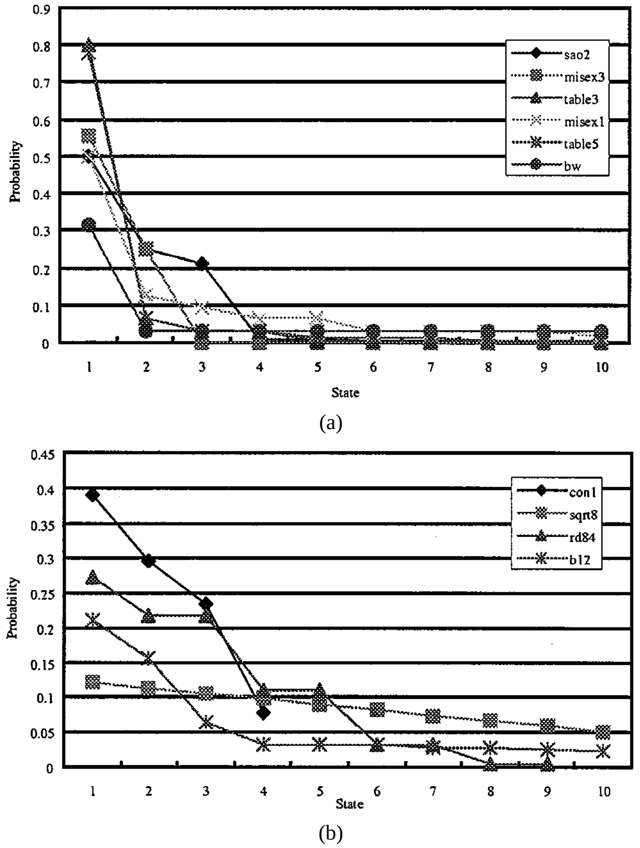
The bipartition and codec algorithms have been implemented in C++ on a SUN Sparc station. We used SIS [12] to synthesize our partition results and estimated the power by PowerMill [12]. The random logic circuits taken from the Microelectronics Center of North Carolina (MCNC) programmable logic arrays (PLAs) are used to demonstrate our algorithms. In the experiment, 5-V supply voltage and a clock frequency of 20 MHz were assumed. The rugged script of SIS was used to optimize the benchmarks. We assumed uniform probability distribution for the primary input, but this assumption is not restrictive. For example, in a pipelined circuit, the input probability distribution can be computed from the output probability of the previous stage as the primary input. The registers of the output part are unchanged in our architectures. Therefore, we do not consider the effect of these registers on area and power dissipation. The area unit and power unit are 128 μm^2 and 1 μW in our experiments, respectively.

Table I presents the performance of our bipartition algorithm on a subset of the MCNC PLAs. It includes the state numbers of Group$_1$ and Group$_2$ and their group probabilities PG_1 and PG_2. In all of the ten benchmarks in Table I, the number of states that cluster in Group$_1$ is less than that of Group$_2$. However, the probabilities of Group$_1$ are far bigger than Group$_2$'s except for the benchmark bw. These results indicate that Group$_1$ is small but will be activated most of the time.

The area and power dissipation for the original and bipartition architectures are tabulated in Table II. The original column shows the total area TA, power dissipation of registers FF, and total power TP implemented by conventional architecture. In the bipartition architecture column, the area size column includes G_1 (Group$_1$), G_2 (Group$_2$),

TABLE II
SIMULATION RESULTS OF ORIGINAL CIRCUIT AND BIPARTITION ARCHITECTURES

Circuits	Original			Bipartition Architecture											
				Area					Power						
	TA	FF	TP	G1	G2	GCB	TA	AI%	G1	G2	GCB	FF	TP	PF%	PR%
sao2	571	2117	3361	75	249	117	944	65	201	20	315	1915	2967	9.6	12
misex3	2557	3467	9041	106	1069	829	2765	8	178	651	2795	2771	7251	20.1	20
table3	2842	3509	8620	72	2580	913	4326	52	127	727	3348	1330	6122	62.1	29
misex1	340	1615	2238	22	72	60	600	76	35	123	206	892	2000	44.8	11
table5	3203	3761	7946	72	2225	1067	4260	33	121	434	4003	1388	6575	63.1	17
bw	694	1144	4443	163	351	29	872	26	247	885	87	1010	4055	11.7	9
con1	209	1331	1506	58	0	41	458	119	175	0	107	1131	1882	32.1	-25
sqrt8	371	1667	2541	137	79	8	643	73	480	70	18	1432	2660	-7.6	-5
rd84	531	1653	3176	218	208	29	1168	120	611	219	1332	1503	4400	9.1	-39
b12	581	2919	3780	114	246	129	1152	98	184	387	369	2377	4222	18.6	-12
Average	1190	2318	4665	103	707	322	1718	44.4	235	351	1258	1575	4213	26	9.7



(a)

(b)

Fig. 5. (a) Probability distribution of circuits I. (b) Probability distribution of circuits II.

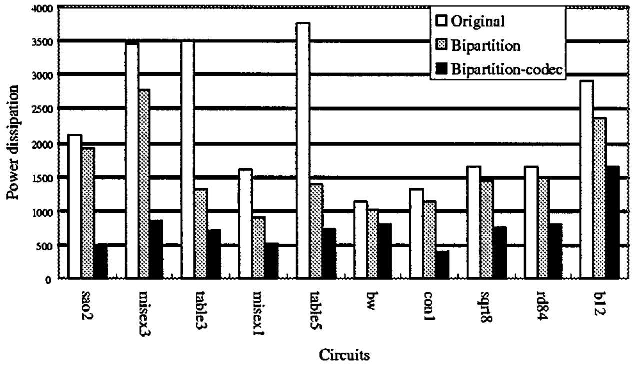


Fig. 6. Power dissipation of the registers in the original, bipartition, and bipartition-codec architectures.

GCB, total area TA, and the percentage of area increase $AI\%$ [computed as $100(\mathit{Area}_{\text{bipartition}} - \mathit{Area}_{\text{original}})/\mathit{Area}_{\text{original}}$]. In the power dissipation column, G_1, G_2, and FF have the same meaning as described before. TP is the total power dissipation. $PF\%$ and $PR\%$ are the power improvement of input registers and total power computed as $100(\mathit{Power}_{\text{original}} - \mathit{Power}_{\text{bipartition}})/\mathit{Power}_{\text{original}}$. For all of these circuits, the area and power dissipation of G_1 and G_2 are much smaller compared to the original architecture. These improvements could be achieved by exploiting the increased *don't care sets* after bipartitioning. We also want to point out that Group_2 consumes small power dissipation because it is often idle. Moreover, the power reduction of registers $PF\%$ is also understandable. Because there are less states contained in Group_1 and Group_2, it is possible that the output of each group can be determined by less input variables. This implies that the switching activity on the input registers is reduced. However, the results show that the bipartition architecture can only obtain power saving for the first six circuits. To explain the results, we analyzed the cluster characteristic of the benchmark circuits. The first ten state probabilities of each circuit in descending order are shown in Fig. 5 (*con1* and *sqrt1* only have four and nine states, respectively). The first state probabilities in Fig. 5(a) are above 0.5 except for bw, after which they decreased dramatically. In contrast, the highest state probability of the circuits in Fig. 5(b) is less than 0.4 and the probability distributions are more evenly distributed than Fig. 5(a). This implies that Group_1 can get high probability with less states for the circuits plotted in Fig. 5(a) than in Fig. 5(b). Further, the power dissipation of the overheads (i.e., two latches, two AND gates, and multilplexers) and registers are a relatively large fraction of the total power dissipation in small circuits and, thus, offsets the power saving. Thus, in small circuits, the power reduction is limited.

Table III presents the area and power dissipation of the ten benchmarks implemented by the bipartition-codec architecture. The columns in this table have the same meaning as in Table II except that "E" and "D" represent the Encoder and Decoder, respectively, and "*Ctrl*" stands for the power dissipation of additional hardware (i.e., two latches, two AND gates, and Multiplexers). The results show that bipartition codec gets much more power improvement than bipartition architecture does. Notice that the Encoder does not only generate the SEL function similar to the GCB, but also encodes the output code. However, the power dissipation of the Encoder is just a little greater than the GCB of bipartition architecture. Because of self-loop effect and the small area of the Decoder, the Decoder hardly dissipates power. Moreover, the power dissipation of the input registers is smaller than that of bipartition architecture. Fig. 6 shows that the bipartition-codec architecture is more

TABLE III
SIMULATION RESULTS OF BIPARTITION-CODEC ARCHITECTURE

Circuits	Area size				Power							
	E	D	Area	AI%	E	D	G2	FF	Ctrl	TP	PF%	PR%
sao2	130	7	725	27	342	14	20	467	379	1255	77.9	63.7
misex3	799	3	2305	-10	2800	4	744	858	669	5075	75.3	43.9
table3	959	36	4084	44	3207	28	663	701	527	5127	80.0	40.5
misex1	72	0	441	30	193	0	132	523	681	1555	67.6	31.7
table5	1164	31	4312	35	4117	22	344	740	527	5748	80.3	27.6
bw	89	155	880	27	238	191	880	804	1795	3908	69.3	12
con1	88	0	342	64	278	0	16	408	480	1182	29.7	21.5
sqrt8	141	3	542	46	526	5	78	748	586	1942	55.2	23.6
rd84	263	10	774	46	969	15	225	805	632	2646	51.3	16.7
b12	233	29	1014	75	529	13	514	1674	839	3495	43.5	5.2
Average	394	27	1542	29.6	1320	29	362	773	712	3193	63.0	31.6

TABLE IV
AVERAGE AREA AND POWER COMPARISON AMONG DIFFERENT ARCHITECTURES

	Precomputation	Choi's algorithm	Choi's area constraint	Bipartition Codec
Area (%)	-0.5	99	-0.3	29.6
Power (%)	-9.6	-32	-11.6	-31.6

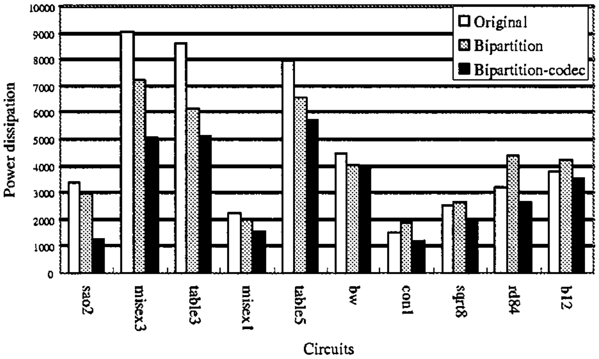


Fig. 7. Total power dissipation of the original, bipartition, and bipartition-codec architectures.

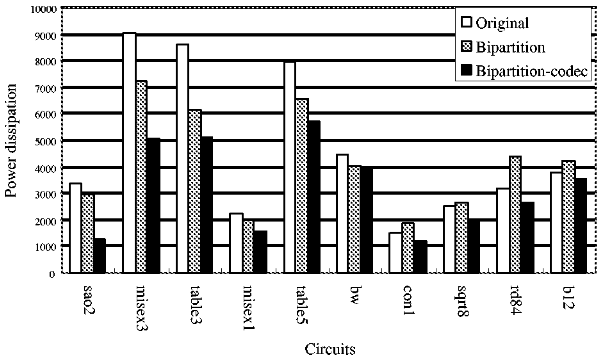


Fig. 8. Total area of the original, bipartition, and bipartition-codec architectures.

successful in reducing the power dissipation of the registers than the bipartition architecture. As a result, the codec architecture (encoder, input registers, and decoder) significantly reduces the power dissipation compared to the appropriate blocks (GCB registers and Group$_1$) of the bipartition architecture. We show the total power dissipation and area of the original, bipartition, and bipartition-codec architectures in Fig. 7 and Fig. 8 for comparison. The bipartition-codec architecture can obtain power reduction not only for the high-cluster probability circuits [listed in Fig. 5(a)], but also for the low-cluster probability circuits [listed in Fig. 5(b)]. The area penalty of bipartition-codec architecture is smaller than the bipartition architecture in most benchmarks. The reader can observe that the relation between area overhead and power reduction is irrelevant. For example, the power reduction for *sao2* benchmark is 63.7%, while the area increased by 27%. In contrast, the power reduction achieved for *b12* is only 5.2%, but the area can be increased by 75%.

For our experiments, the threshold of the group probability, *MinProb* was set to 0.5 in Fig. 4. It should be noted that it may also be possible to use other values to produce larger power reduction. For example, if we set the thresholds of benchmark *con1* as 0.95, the power reduction $PR\%$ increases from 21.5% to 29%. However, *MinProb* = 0.5 leads to the best results for most of the circuits.

Table IV compares the average area and power reduction of previous schemes with our proposed architecture. The precomputation column represents the circuits implemented by precomputation-based method [7]. Choi's algorihm and Choi's area constraint columns represent the circuits implemented by Choi's algorithm and Choi's algorithm with area constraint [8], respectively. The data of thre above three columns are cited from [8]. The bipartition-codec column is the proposed architecture. As shown in the table, our proposed architecture obtained almost the same power saving and smaller area overhead than Choi's algorithm. Although precomputation and Choi's area constraint algorithms can achieve power reduction and reduce the area by a little bit, the power reduction is limited. Consequently, our proposed architecture obtained significant power saving with less area overhead than Choi's method.

In summary, the bipartition approach obtains power reductions from the circuits with few high-probability states. The bipartition circuit consumes less power than the original one. The power dissipation of the input registers is also reduced for the sake of less switching of input variables. The codec architecture further reduces the switching activity of the input registers, as well as the internal switching activity of the Decoder. As a result, our new bipartition-codec approach produces very promising results. If the power reduction is the essential consideration of a combinational circuit, the designer can simply predict the power-saving effects achieved by our architecture based only on the PLA description of the circuit function. It is very helpful for a designer to determine the initial architecture without the unnecessary effort in synthesizing and analyzing the logic structure for low power.

V. Conclusion

In this paper, a new bipartition-codec architecture for pipelined circuits with low-power consideration is proposed. The experimental results confirm that our architecture can effectively save the power for a large class of random logic circuits. We treat each output vector of a combinational block in a pipelined circuit as a state in an FSM. When the output vectors of some benchmarks cluster only around a few special output vectors, our algorithm could effectively extract the small group that includes these few output vectors. We applied a state encoding technique to reduce the switching activity in the highly active Group_1. Although the encoder must be active at any time, it dissipates little power. Group_2 is often idle and dissipates little power on average. Therefore, huge power reduction is obtained for large pipelined circuits, where the probability of Group_1 being active is high. Although circuit partition and gated clock for low power are not new approaches, this paper provides two novel investigations for attaining power reduction of combinational circuits. First, we analyzed the output behavior of a combinational circuit and modeled it as an FSM, bipartitioning it with probability consideration. Specifically, we only need a high-level description of the circuits for our bipartition algorithm such as a truth table; no Boolean equations or detailed gate-level descriptions are necessary. Second, as in some circuits, the power dissipation of the registers is significantly higher than the power dissipated in the combinational block; we incorporated an encoding concept called codec architecture to further optimize power consumption. The gain of power efficiency compensates for the offset caused by the duplicated registers and overhead.

When some small number of output states dominates most of the state transitions, our solution will have the best performance. For example, we can get the best power reduction for the circuits in Fig. 5(a). However, there are some circuits not suitable for our architecture. If the probability distribution of output vectors is uniform, then the circuits will not benefit from our architecture. For example, in Table I, the probability of Group_1 and Group_2 in bw and *b12* are more uniform than the others; the power reduction of both circuits shown in Table III are limited. In addition, the power consumption of the additional hardware (i.e., two latches, two AND gates, and multiplexers) of our architecture is necessary. The power reduction in small benchmarks is limited, such as *sqrt8* and *con1* in which the overhead offsets the power saving.

Acknowledgment

The authors would like to thank Dr. E. Naroska for his helpful feedback in preparing this paper.

References

[1] K. Roy and S. C. Prasad, *Low-Power CMOS VLSI Circuit Design*, New York: Wiley, 2000.

[2] E. Olson and S. M. Kang, “Low-power state assignment for finite state machine,” in *Proc. Int. Workshop Low-Power Design*, Napa, CA, Apr. 1994, pp. 69–74.

[3] G. Hachtel *et al.*, “Re-encoding sequential circuits to reduce power dissipation,” in *Proc. Int. Workshop Low-Power Design*, Napa, CA, Apr. 1994, pp. 63–68.

[4] L. Benini and G. D. Micheli, “State assignment for low-power dissipation,” *IEEE J. Solid-State Circuits*, vol. 30, pp. 258–268, Mar. 1995.

[5] ——, “Automatic synthesis of low-power gated-clock finite-state machine,” *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 630–643, Sept. 1996.

[6] S.-H. Chow, Y.-C. Ho, T. Hwang, and C. L. Liu, “Low power realization of finite state machines—A decomposition approach,” *ACM Trans. Design Automat. Electromagn. Syst.*, vol. 1, pp. 315–340, July 1996.

[7] M. Alidina, J. Monterio, S. Devadas, S. Devadas, A. Ghosh, and M. Papaefthymiou, “Precomputation-based sequential logic optimization for low power,” *IEEE Trans. VLSI Syst.*, vol. 2, pp. 426–436, Dec. 1994.

[8] I.-S. Choi and S.-Y. Hwang, “Circuit partition algorithm for low-power design under area constraint using simulated annealing,” in *Inst. Elect. Eng. Proc. Circuit Devices Syst.*, vol. 146, Feb. 1999, pp. 8–15.

[9] S.-J. Ruan, R.-J. Shang, F. Lai, S.-J. Chen, and X.-J. Huang, “A bipartition-codec architecture to reduce power in pipelined circuits,” in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1999, pp. 84–89.

[10] S.-J. Chen, R.-J. Shang, X.-J. Huang, S.-J. Ruan, and F. Lai, “Bipartition and synthesis in low power pipelined circuits,” *IEICE Trans. Fundamentals Electron., Commun., Comput. Sci.*, vol. E81-A, pp. 664–671, Apr. 1998.

[11] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*, New York: McGraw-Hill, 1994.

[12] “SIS: A system for sequential circuit synthesis,” Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, CA, 1992.