# AN EFFECTIVE OUTPUT-ORIENTED ALGORITHM FOR LOW POWER MULTIPARTITION ARCHITECTURE

Shanq-Jang Ruan, Jen-Chiun Lin, Po-Hung Chen, Feipei Lai, *Senior Member, IEEE,*
Kun-Lin Tsai and Chung-Wei Yu

Dept. of Electrical Engineering &
Dept.of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan
flai@cc.ee.ntu.edu.tw

## ABSTRACT

Circuit partition for low power is one of the useful techniques which reduce power dissipation by confining the switching activity to a subcircuit. In this paper, we propose an effective output-oriented partition algorithm for low power combinational logic circuit. Unlike previous study, we discuss the relationship among power dissipation, area complexity and input/output behavior of a combinational logic circuit rather than inspect its logic function. Experimental results show that our algorithm can obtain sizable power saving over a wide range of MCNC benchmarks.

## 1. INTRODUCTION

In CMOS design, the dynamic power of circuits accounts for over 90% of the total power consumption [1]. It implies the power dissipation is directly proportional to the average switching activity of a circuit. Based on the result, many power optimization techniques which aimed at minimizing switching activity at architectural and logic levels are proposed in recent years. [2] is a good survey of previous work.

Recently, there are two major low power techniques in logic level synthesis: *precomputation* and *gated-clock*. Luca and Giovanni proposed gated-clock to build low power FSMs [3]. The technique stops the useless circuit glitches in the idle states of FSMs. In [4], [5], an FSM being decomposed 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.* first proposed precomputation-base scheme, which selectively disables the inputs of a sequential logic circuit to achieve low power [6]. On the other hand, another variant precomputation scheme is circuit partition. In 1998, Chen proposed a bipartiton architecture which treated each output value of a combinational circuit as one state of an FSM, the most transitive states will be extracted to build a small subcircuit [7]. The power saving is based on the observation that most time the small block is at work and the big one is idle. However, bipartition approach is limited if the probability distribution of the output vectors is uniform in the circuit, moreover the precomputation logic which used to select the subcircuit to compute always offsets the power saving. Choi and Hwang partition a combinational circuit into multiple subcircuits through recursive application of Shannon expansion with respect to the selected input variables [8]. The advantage of this architecture is that it simplify the precomputation logic as an "encoder". Nevertheless, the algorithm to select the best input is quite complex.

In this paper, we propose an *output-oriented partition algorithm* for optimizing the power of logic circuits. By surveying the theoretical models of power dissipation and area complexity, we know that the circuit area depends on the output entropy and grows exponentially with input number [9], [10], [11], [2]. Therefore, based on evenly partitioning the output vectors by Shannon expansion scheme, where the area of each partition is significantly smaller than the original. Since the different output vector of each partition may be reduced, of course, the entropy of each partition is also reduced.

## 2. MULTIPARTITION ARCHITECTURE

According to the Shannon expansion, a combinational circuit can be partitioned into $2^n$ blocks by recursively applying the Shannon expansion, where $n$ is the number of the input variables. Here the selection logic is constructed in the form of a decoder. For example, if we select any two of the input variables in a combinational logic, the circuit will be partitioned into four blocks as in Fig. 1. Depending on the selection of input variables, the selection logic activates one of the four cofactor circuits *f1,f2,f3,f4* by enabling its input latch while the others is disabled.

However, a large amount of duplicated latches for turning on/off each cofactor circuit is a main disadvantage of this architecture. In addition, we also need a *m*-to-1 multiplexors for each output pin, where *m* is the number of partitions.

0-7803-6542-9/00/$10.00 © 2000 IEEE

## 3. PARTITION ALGORITHM

As we mentioned in [9], the area complexity depends on the number of input, in which, the average area complexity of an $n$-input Boolean function varies exponentially with the number of different output. On the other hand, in a random input logic, if we select $n$ input logic as selection variables, the activated probability of each partition can be regarded as $1/2^n$. Therefore, evenly partitioning the output vectors by its input variables leads to better result on theoretical view.

**Example 1:** In Fig. 2, we select different input variables to partition the truth table based on the Shannon expansion. If we select $x_2$ as the partition variable, the output vector will be {00, 10} when $x_2 = 0$, and it will be {11, 10} when $x_2 = 1$. Hence the number of different output vectors in both partitions is two. However, if we select $x_3$ to partition the same table, the number of different output vectors are three and two corresponding to $x_3 = 0$ and $x_3 = 1$, respectively. However, for instance, if we want to partition circuit into $2^k$ subcircuits by using the Shannon expansion, there will be different combinations. It is very inefficient to try all these combinations for finding an optimal solution. Thus we develop a heuristic approach to satisfy the *minimal* number of output for each partition. The basic operation of it is selecting an input variable which yields the even distribution with minimal number of different output. A new input set is given without the selecting variable for partitioning. A $2^k$ partition will be achieved by applying the operation $k$ times. The following heuristic partition algorithm illustrates the procedure.

```
Bit-Vector: Vector of {0,1}
Pattern: (Bit-Vector Input-Vector,
          Bit-Vector Output-Vector)
Partition-Vector: Ordered-Set of Integer
Pattern-Set: Ordered-Set of Pattern
Partition-vector-Set: Ordered-Set of Partition-Vector
Bit-Vector-Set: Order-Set of Bit-Vector
PAPARTITION(Pattern-Set PS, Integer depth)
  Integer min-value ←+∞
  Partition-Vector SV ←{0,1,...,|Input-Vector|-1
  Partition-Vector PV ←Φ
  FOR Integer I From 1 TO depth
  DO
    Partition-Vector-Set PVS ←Φ
    FOR EACH Integer p∈ SV-PV
      DO
        Partition-Vector New-PV ←PV ∪ {p}
        Pattern-Set(PS_0, PS_1, ..., PS_2^i-1) ←
                  DO-PARTITION(PS, NEW-PV)
        Integer max-value ●
                  MAX_{j=0}^{2^i-1} {COUNT − OUTPUT(PS_J)}
        IF max-value < min-value
        THEN
          PVS ←{New-PV}
          min-value ←max-value
        ELSE IF max-value = min-value
        THEN
          PVS ←PV ∪ {New-PV}
        PV randomly select an element from PVS
    RETURN PV

  DO-PARTITION(Pattern-Set PS, Partition-Vector PV)
    Pattern-Set(PS_0, PS_1, ..., PS_2^i-1) ←(Φ, Φ, ..., Φ)
    FOR EACH Pattern P∈ PS
    DO
      Integer i ←0
      FOR EACH Integer j∈ PV
      DO
        i ←i×2+P.Input-Vector[j]
      PS_i ∪ {P}
  RETURN(PS_0, PS_1, ..., PS_2^i-1)
COUNT-OUTPUT(Pattern-Set PS)
Bit-Vector-Set OS ←Φ
FOR EACH Pattern P∈ PS
DO
  OS ←OS ∪ {P. Output-Vector}
RETURN |OS|
```

The algorithm partitions a circuit into $2^{depth}$ parts according to the *depth* bits of its input. The input bits selected for partition form the partition vector. During each iteration, we add one more input bit to the partition vector such that the number of different output of partitions by the partition vector is minimal, until we reached depth bits. Note that if there exist more than one bit to do so, we randomly select one in our algorithm.

## 4. EXPERIMENTAL RESULT

The output-oriented partition algorithm has been implemented in C++ on a SUN Sparc station. We used the SIS to synthesize our partition results and estimated the power with cell library, *MCNC.genlib* and *MCNC_latch.genlib* (including in SIS) [12]. The power dissipation of the circuits including selection logic, flip-flops, input latches of each partition, partitions and multiplexor. 19 random logic circuits taken from MCNC PLAs are used to verify our algorithm. In the experiment, 5v supply voltage and a clock frequency of 20MHz were assumed. The rugged script of SIS was used to optimize the benchmarks. All power estimates are in micro-Watts and areas are the relative cost (which defined by MCNC.genlib and MCNC_latch.genlib of each cell in SIS). Table 1 shows the experimental results over wide range of MCNC PLAs. The *Original* column stands for the benchmarks implemented by conventional architecture. The benchmarks implemented by the proposed algorithm are shown in *multipartition architecture* column. The #P represents the partition number of the multipartition architecture. The power and area reductions are computed as $\frac{100(Original - Multipartition)}{original}$. Thus, the negative value of the columns mean the power/area reduction in multipartition architecture. In all benchmarks, two-way partition obtains power saving effect, as well as area decreased in most

cases. However, the highest power reduction rate does not always perform on two-way partition. For example, the highest power reduction rate of *9symml* and *sao2* are four-way partition. The remarkable area overhead of eight-way partition in most benchmarks are due to the duplicated input latches and multiplexors.

In summary, the power savings can be obtained over most benchmarks for multipartition architecture by synthesizing with our output-oriented algorithm. As shown in Table 1, in the case of the *9symml*, up to 53.7% substantial power reduction can be obtained. However, as the partition number increased, the area overhead increased due to the control logic which consists of duplicated latches, multiplexor and complex selection logic. This also offsets the power saving effect we get by the multipartition architecture.

## 5. CONCLUSION

An output-oriented partition algorithm for low power multipartition architecture was presented. Unlike previous studies, we partition a combinational logic circuit by the output occurrence rather than analyze its logic function. Based on the Shannon expansion, we select the input variables in term of minimizing and evenly distributing the distinct output among partitions. Therefore, reduce the area complexity and limit switching activity of each partition.

While partitioning a circuit by applying the Shannon expansion, this paper offers two important contributions. First, we can obtain power saving with area decreased by using our algorithm in multipartition architecture. The limitation is unavoidable to all partition schemes based on the Shannon expansion. Our experimental results suggests that two or four-way partition achieved good tradeoff in power and area. Second, output occurrence is another possibility for partitioning circuit to get power reduction. Besides, the multiplexor of multipartition architecture determines the amount of power dissipation, area and delay. As proposed in [8], if we replace the output multiplexor by a transmission gate in a wired-OR form, our method can produce better results in power, area and delay.

## 6. ACKNOWLEDGMENT

This work was supported by R.O.C. NSC under grant NSC 89-2213-E-002-138.

## 7. REFERENCE

[1] A. P. Chandrakasan, S. Sheng, and R.W. Brodersen, "Low power CMOS digital design," *IEEE J. Solid-State Circuits*, vol.27, pp.473-484, Apr. 1992.

[2] M. Pedram, "Power minimization in IC design: Principles and applications," *ACM Tran. Design Automation of Electronic Systems*, vol.1, pp.3-56, Jan 1996.

[3] L. Benini and G. D. Micheli, "Automatic synthesis of low-power gated clock finite-state machine," *IEEE Trans. Computer-Aided Design*, vol.15, pp.630-643, Sep. 1996.

[4] J.C. Monterio and A.L. Oliveira, "Finite state machine decomposition for low-power," in *Proceeding Design Automation Conf.*, pp.758-763, 1998.

[5] S.-H. Chow, Y.-C. Ho, T. Hwang, and C.L. Liu, "Low power realization of finite state machines-a decomposition approach," *ACM Tran. Design Automation of Electronic Systems*, vol.1, pp.315-340, Jul. 1996.

[6] 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.

[7] 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 of Electronics, Communication and Computer Science*, vol. E81-A, pp.664-671, Apr. 1998.

[8] I.-S. Choi and S.-Y. Hwang, "Circuit partition algorithm for low-power design under area constraint using simulated annealing," *IEE Proc. Circuit Devices Syst.*, vol.146, pp.8-15, Feb. 1999.

[9] K.-T. Cheng and V.D. Agrawal, "An entropy measure for the complexity of multi-output boolean functions," in *proceedings of 27th ACM/IEEE Design Automation Conference*, pp.302-305, 1990.

[10] M. Nemani and F. N. Najm, "High-level area and power estimation for VLSI circuits," *IEEE Trans. Computer-Aided Design*, vol.18, pp. 697-713, Jun. 1999.

[11] E. Macii, M. Pedram, and F. Somenzi, "High-level power modeling, estimation, and optimization," *IEEE Trans. Computer-Aided Design*, vol. 17, pp.1061-1079, Nov. 1998.

[12] Ellen M. Sentovich *et. al.*, "SIS: A system for sequential circuit synthesis," tech. rep., Department of Electrical Engineering and Computer Science, University of California, Berkely, May 1992.

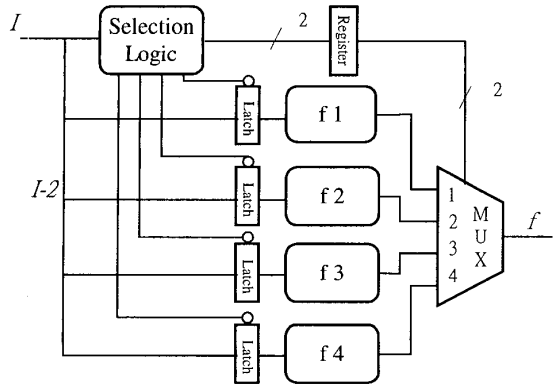


**Fig. 1 4-partition architecture**

**Table 1 Experimental result**

| | Original | | Multipartition Architecture | | | Reduction | |
|---|---|---|---|---|---|---|---|
| benchmark | Power_orig | Area_orig | # of P | Power_total | Area_total | Power % | Area % |
| 9symml | 1229.2 | 358 | 2 | 767.1 | 437 | -37.6 | 22.1 |
| | | | 4 | 624.6 | 612 | -49.2 | 70.9 |
| | | | 8 | 569.3 | 846 | -53.7 | 136.3 |
| sqrt8 | 635.5 | 348 | 2 | 523.8 | 279 | -17.6 | -19.8 |
| | | | 4 | 606 | 428 | -4.6 | 23.0 |
| | | | 8 | 611.1 | 588 | -3.8 | 69.0 |
| cm152a | 560.7 | 142 | 2 | 528.7 | 275 | -5.7 | 93.7 |
| | | | 4 | 482.9 | 443 | -13.9 | 212.0 |
| | | | 8 | 516.7 | 739 | -7.8 | 420.4 |
| cmb | 751 | 428 | 2 | 723.9 | 387 | -3.6 | -9.6 |
| | | | 4 | 772.9 | 664 | 2.9 | 55.1 |
| | | | 8 | 804.5 | 1205 | 7.1 | 181.5 |
| con1 | 355 | 338 | 2 | 346.1 | 184 | -2.5 | -45.6 |
| | | | 4 | 325.4 | 280 | -8.3 | -17.2 |
| | | | 8 | 389.5 | 471 | 9.7 | 39.3 |
| rd73 | 546 | 338 | 2 | 559.7 | 311 | 2.5 | -8.0 |
| | | | 4 | 535.2 | 471 | -2.0 | 39.3 |
| | | | 8 | 608.6 | 698 | 11.5 | 106.5 |
| rd84 | 814.2 | 348 | 2 | 680 | 396 | -16.5 | 13.8 |
| | | | 4 | 707.7 | 645 | -13.1 | 85.3 |
| | | | 8 | 734.2 | 954 | -9.8 | 174.1 |
| C17 | 222.2 | 318 | 2 | 212.3 | 108 | -4.5 | -66.0 |
| | | | 4 | 248.4 | 181 | 11.8 | -43.1 |
| | | | 8 | 329.1 | 243 | 48.1 | -23.6 |
| t | 222.2 | 318 | 2 | 212.3 | 108 | -4.5 | -66.0 |
| | | | 4 | 238.3 | 180 | 7.2 | -43.4 |
| | | | 8 | 340 | 247 | 53.0 | -22.3 |
| sao2 | 999.2 | 368 | 2 | 716.3 | 403 | -28.3 | 9.5 |
| | | | 4 | 624.5 | 573 | -37.5 | 55.7 |
| | | | 8 | 708.4 | 870 | -29.1 | 136.4 |
| cm85a | 605.6 | 378 | 2 | 604.7 | 319 | -0.1 | -15.6 |
| | | | 4 | 656 | 573 | 8.3 | 51.6 |
| | | | 8 | 604.1 | 861 | -0.2 | 127.8 |
| clip | 966.7 | 300 | 2 | 845.5 | 485 | -12.5 | 61.7 |
| | | | 4 | 921 | 810 | -4.7 | 170.0 |
| | | | 8 | 822 | 982 | -15.0 | 227.3 |
| x2 | 604.4 | 173 | 2 | 581.7 | 274 | -3.8 | 58.4 |

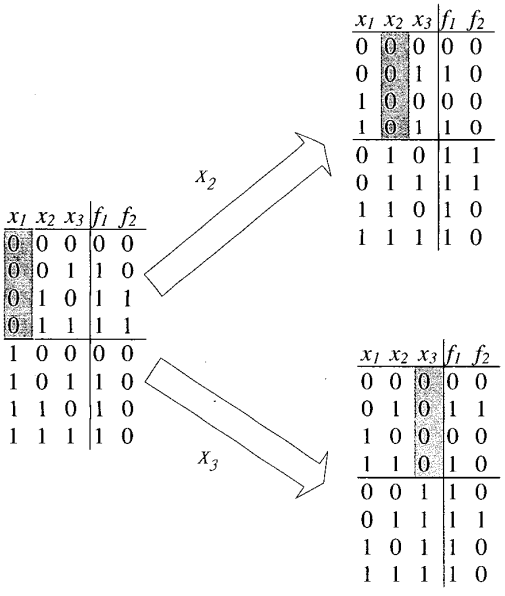


**Fig. 2 Example of partition**