A Fast Probability-Based Algorithm for Leakage Current Reduction Considering Controller Cost

Kuang-Yao Chen

Department of Electronic Engineering

National Changhua University of Education

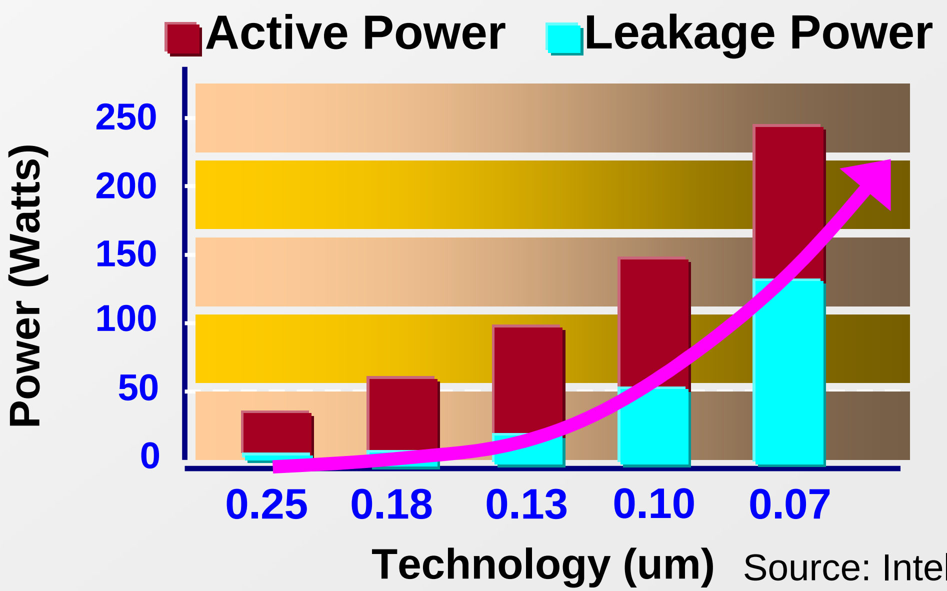
Taiwan

Outline

- **Introduction**
- **Problem Definition**
- **Cost Function**
- **Algorithm**
- **Experiments**
- **Conclusions and Future Work**

Introduction

- **Ultra Deep Sub-Micron Chips Have High Leakage Currents.**
- **We Can Reduce Leakage Current in**
  - ✶ **Normal mode**
  - ✶ **Sleep mode (focused in our research)**



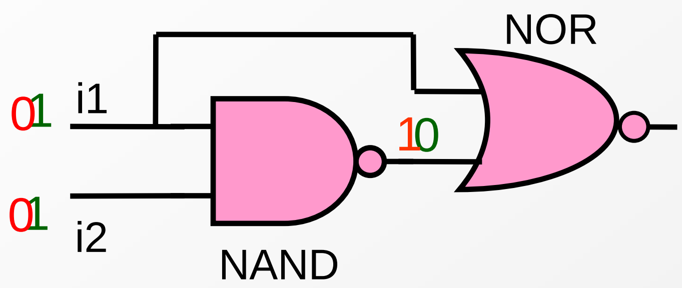
Source: Intel

Introduction

- **Many Algorithms Have Been Proposed to Construct** ***Minimum Leakage Vector*** **(MLV)**
  - In order to reduce the leakage current of sleep mode
- **Issue of the Published Techniques**
  - **They omit to count the leakage current overhead of a newborn MLV controller.**

Introduction

- **MLV and MLV Problem**
  - **For sleep mode**



Input State	Leakage (nA)	
	NAND	NOR
00	0.8	7.7
01	5.4	2.4
10	3.0	4.8
11	7.2	0.7

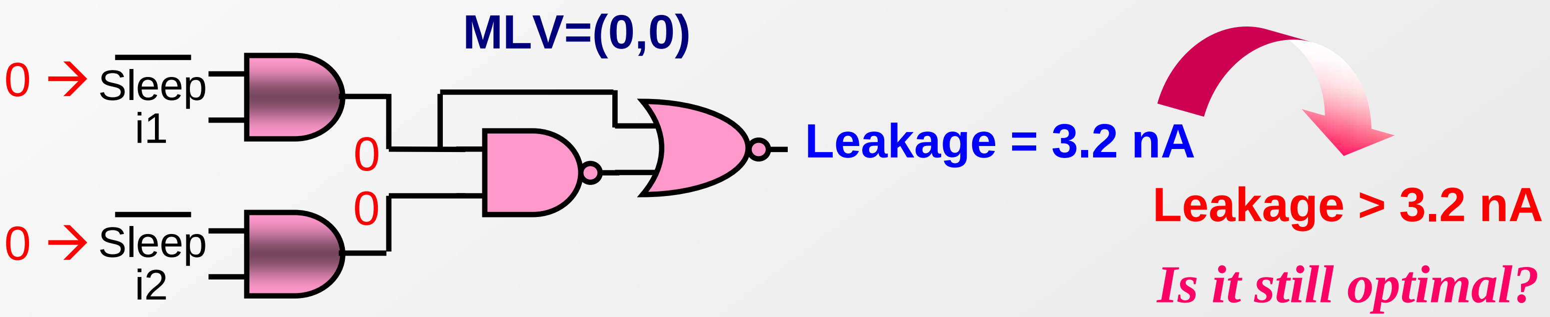
MLV is (0, 0)!

Case 1:
If (**(i1,i2)==(1,1)**) Then
leakage = 7.2 + 4.8
= 12.0 (nA)

Case 2:
If (**(i1, i2)==(0,0)**) Then
leakage = 0.8 + 2.4
= 3.2 (nA)

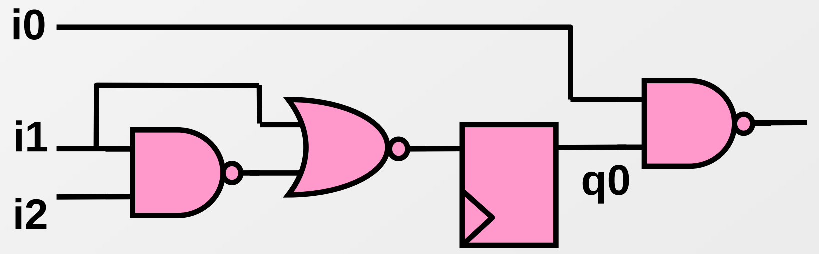
Introduction

- **Our Technique for Solving MLV Problem**
  - ✶ **Taking MLV controller cost into account**
  - ✶ **Using fast probability-based algorithm**



Problem Definition

- **Control-Point**
  - ✶ **Primary inputs**
  - ✶ **F/F outputs**
- **Control-Point Vector (CPV)**
- **Inherent CPV (*CPV$_I$*)**
  - ✶ **Given by designers for sleep mode**
  - ✶ **The elements**
    - Always ONE (1)
    - Always ZERO (0)
    - Unfixed (X)

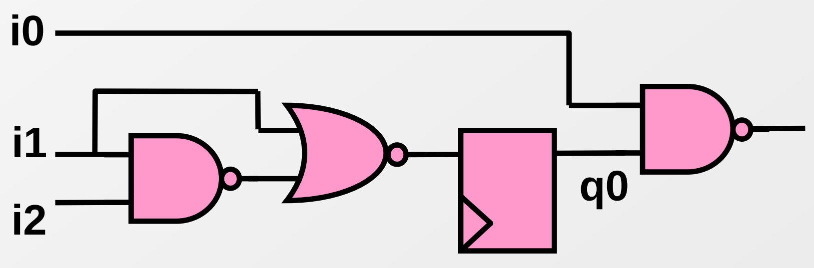


i0, i1, i2, and q0 are control-points.

Inherent CPV = (X, 0, 0, X)

Problem Definition

- **Minimum Leakage CPV (*CPV_M*)**
  - **Minimum Leakage Vector**
  - **The elements in *CPV_M***
    - Always ONE
    - Always ZERO
  - **Example**
    - (0, 0, 0, 0)



CPV = (i0, i1, i2, q0)

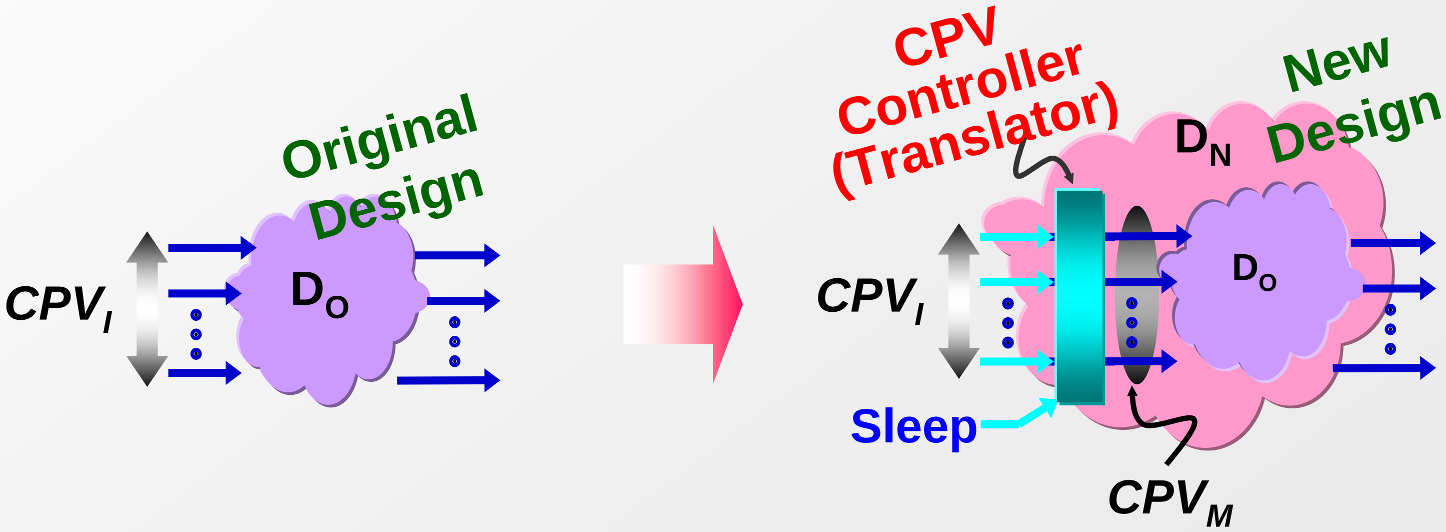
CPV_I = (X, 0, 0, X)

CPV_M = (0, 0, 0, 0)

Problem Definition

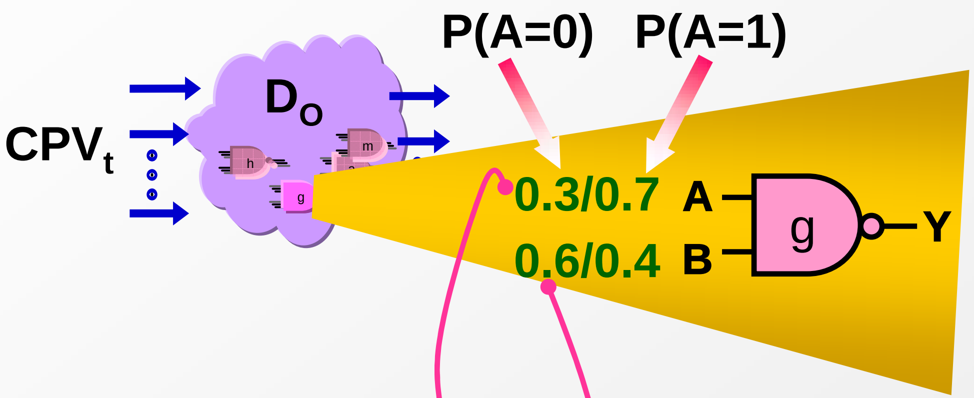
- **Our Objective**
  - **Finding a CPV_M for the given CPV_I**
  - **Counting the cost of the CPV controller**
  - **Post-processing the design by pin-reordering**

Illustration for Combinational Circuit



Cost Function

- **The Cost Function of Our Algorithm**
  - **Expected Value of Leakage Current**
- **Expected Value of Leakage Current of a Gate**



Input State	Leakage (nA)
00	0.8
01	5.4
10	3.0
11	7.2

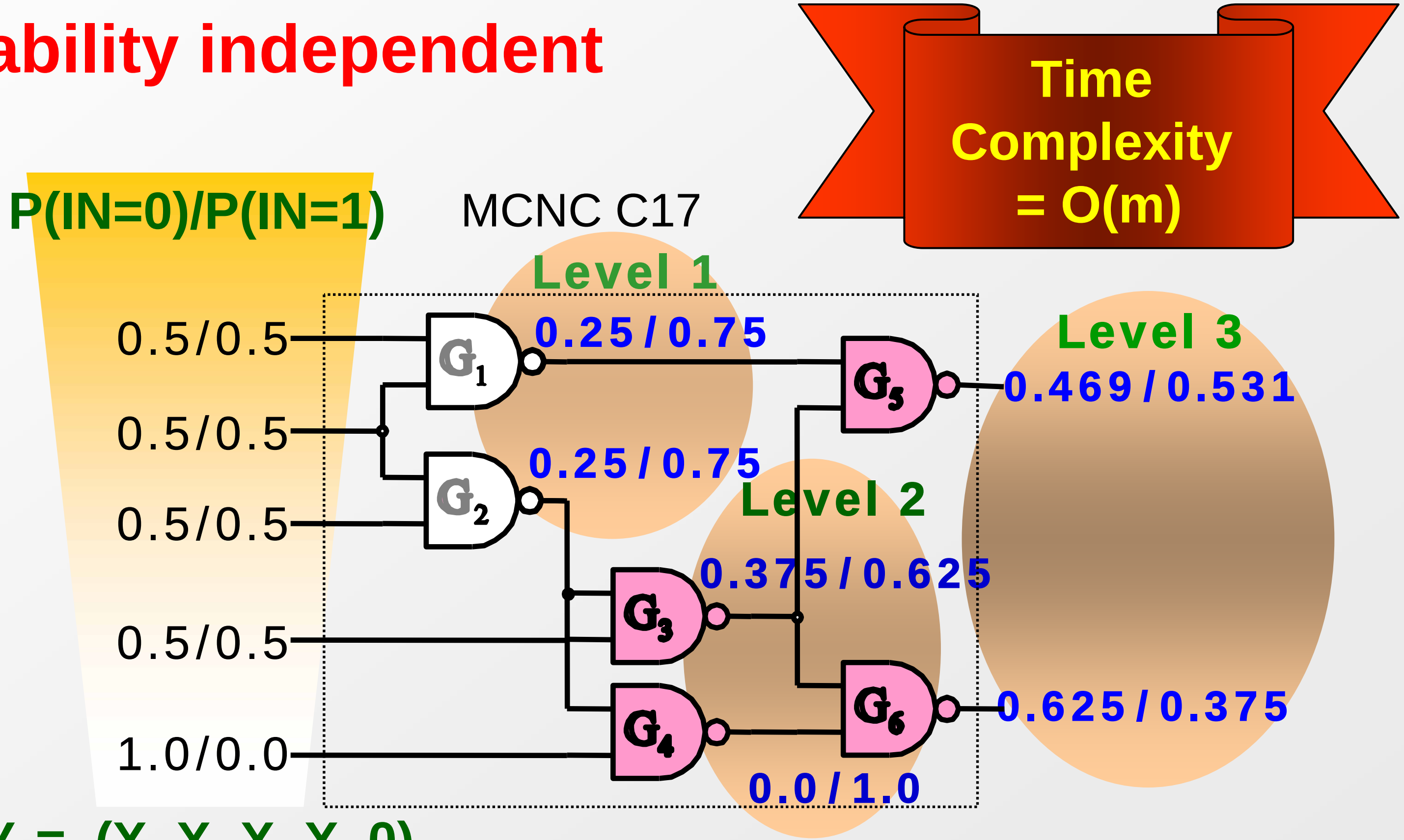
Probability of Each State:

P(A=0, B=0) = 0.3 * 0.6 = 0.18
P(A=0, B=1) = 0.3 * 0.4 = 0.12
P(A=1, B=0) = 0.7 * 0.6 = 0.42
P(A=1, B=1) = 0.7 * 0.4 = 0.28

E[*Lkg_of_gate(g, D_O, CPV_t)*]
= (0.18 * 0.8) + (0.12 * 5.4) + (0.42 * 3.0) + (0.28 * 7.2)
= 4.07 (nA)

Cost Function

- **Probability Calculation**
  - ★ **Assume that each pin is** **probability independent**



$CPV_I = (X, X, X, X, 0)$

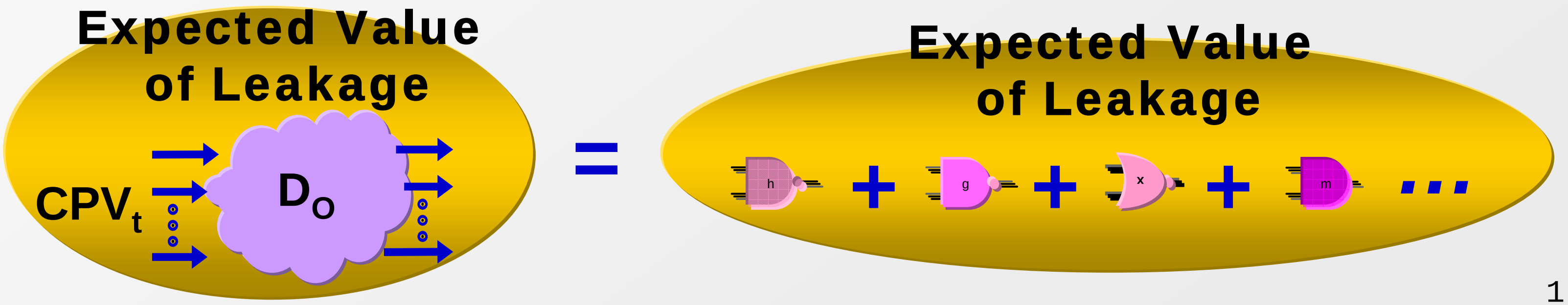
Cost Function

- **Theorem**

$$E[Lkg\_of\_dsgn(D_O, CPV_t)] = \sum_{i=1}^{m} E[Lkg\_of\_gate(g_i, D_O, CPV_t)]$$

(m is the total gate number of design D_O)

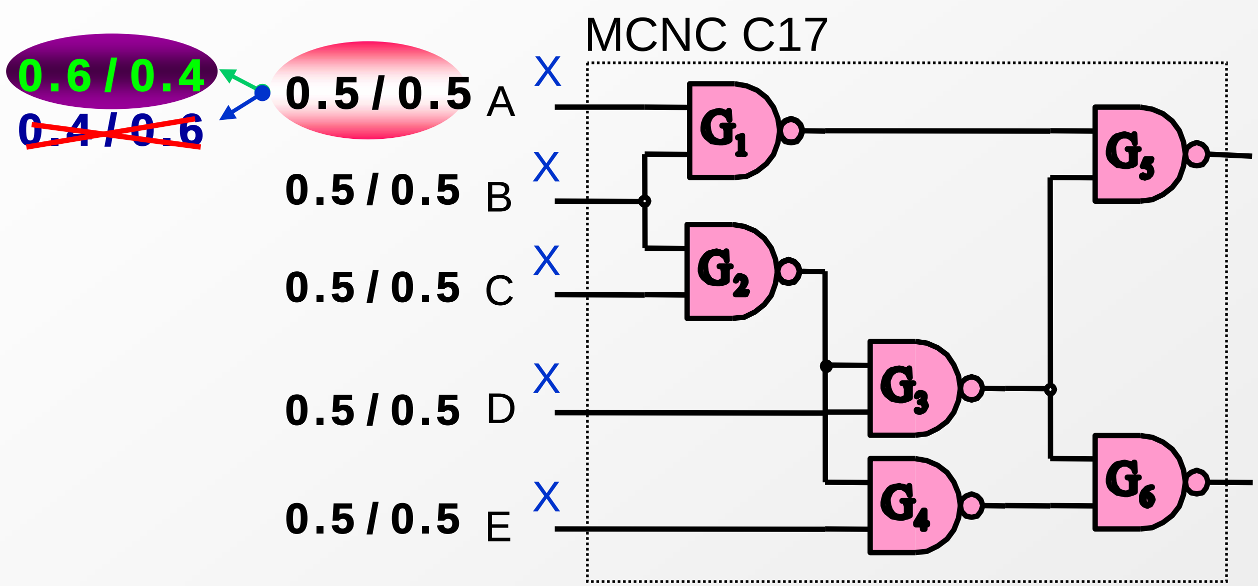
- **Time Complexity of Calculating $E[Lkg\_of\_dsgn(D_O, CPV_t)]$ Is O(m)**



Algorithm

- **Illustration (Ignoring Controller Cost)**

Rt = 0.1



CPV$_I$ = (X, X, X, X, X)

P(A=0)=0.6→ Expected Value: 27.33 nA

P(A=1)=0.6→ Expected Value: 27.53 nA

0.5 / 0.5

Increase P(A=0) by Rt

0.6 / 0.4

Update Probability of Each Nets

Calculate Expected Value of Leakage

Expected Value: 27.33 nA

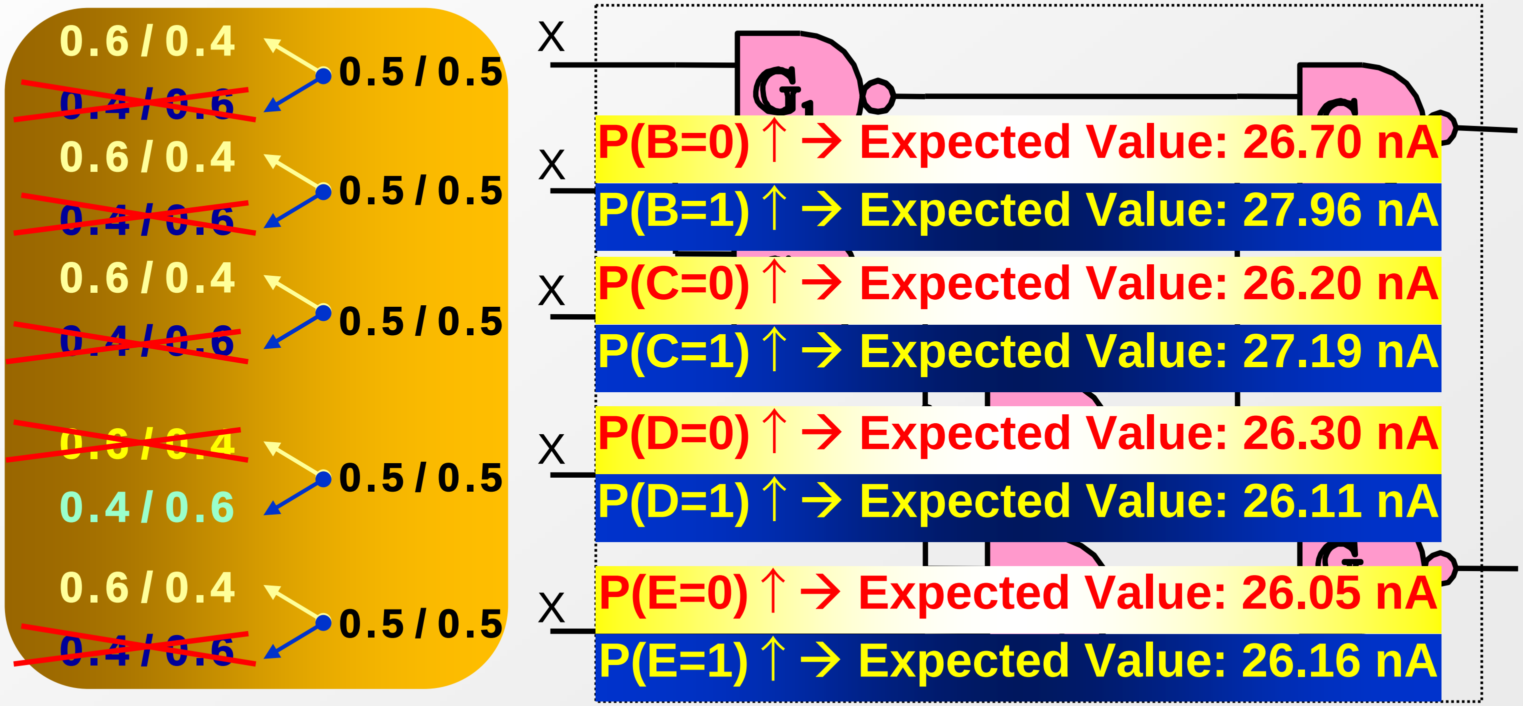
Update Probability of CPV

Algorithm

- **Illustration**
  - ★ **First iteration**

Rt = 0.1

MCNC C17

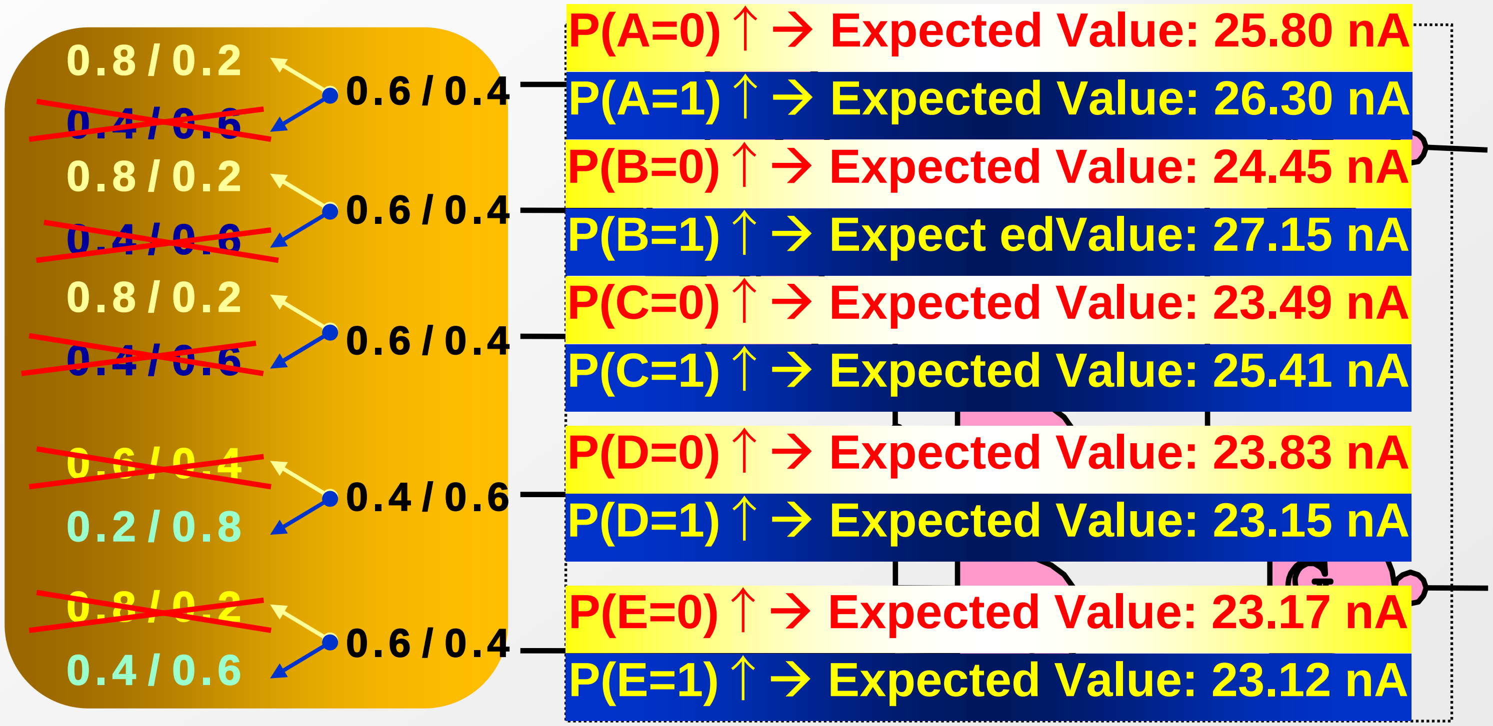


Algorithm

- **Illustration**
  - ★ **Second iteration (updating Rt)**

Rt = Rt + 0.1 = **0.2**

MCNC C17

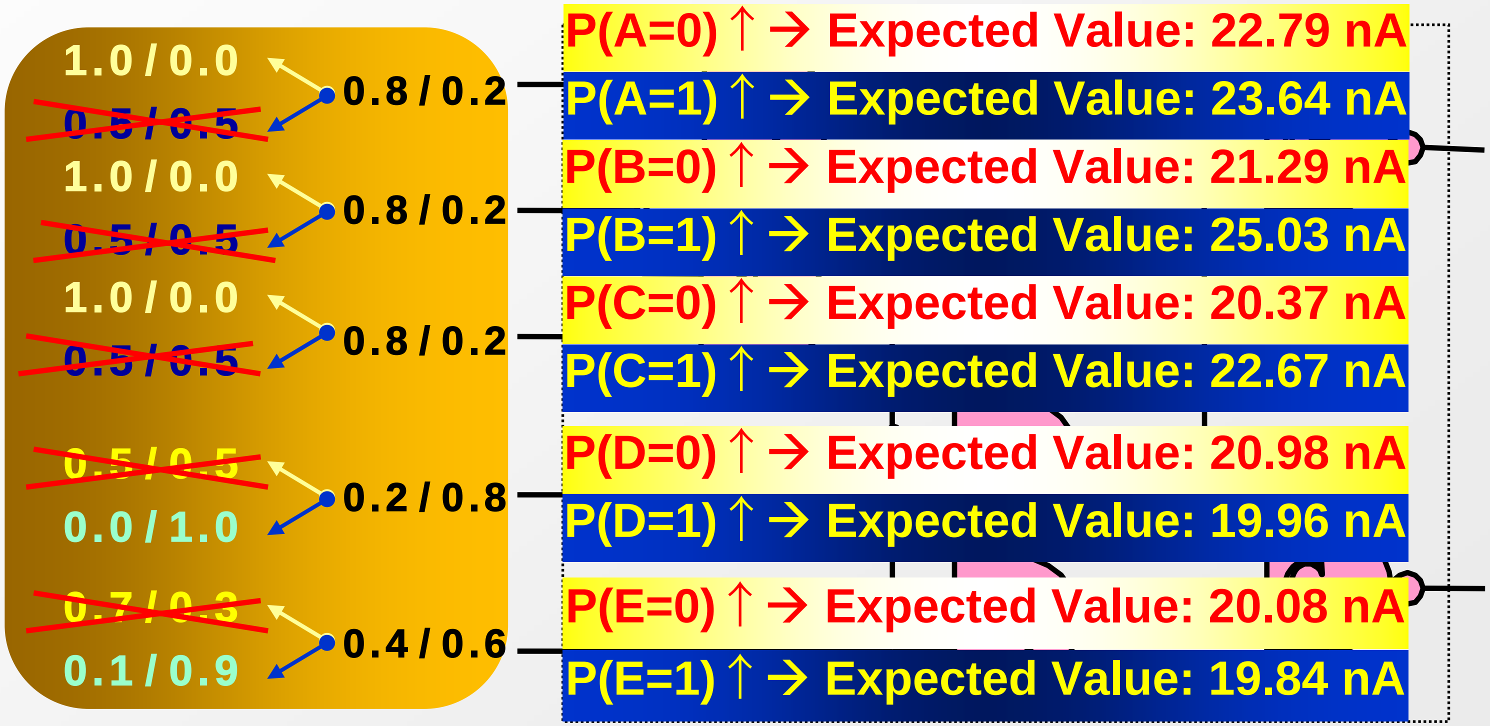


Algorithm

- **Illustration**
  - **Third iteration**

Rt = Rt + 0.1 **= 0.3**

MCNC C17



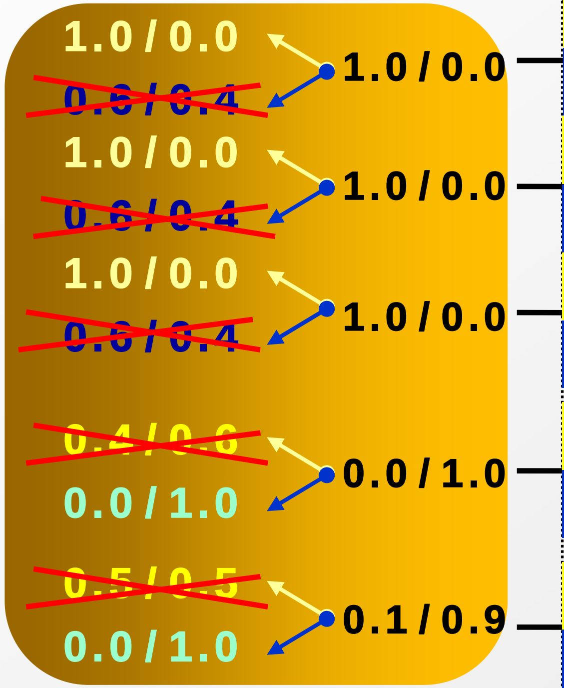
Algorithm

- **Illustration**
  - ★ **Forth iteration**

Rt = Rt + 0.1 = **0.4**

Optimal CPV (0,0,0,1,1)

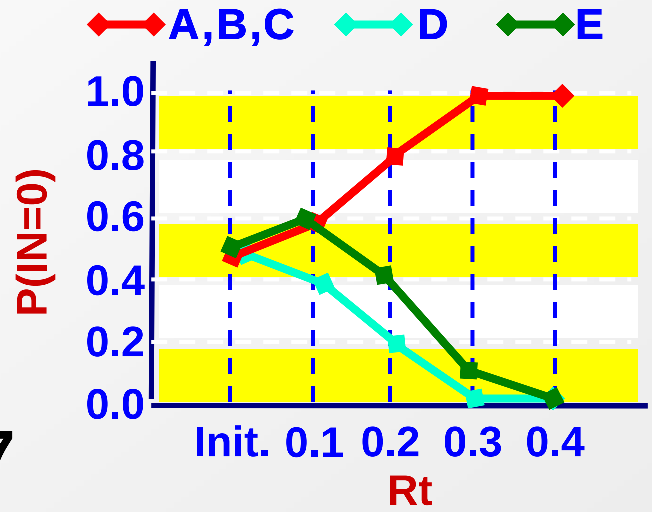
Time Complexity < O(m*n)



P(A=0)↑→ Expected Value: 19.84nA
P(A=0)↑→ Expected Value: 20.72 nA
P(B=0)↑→ Expected Value: 19.84 nA
P(B=1)↑→ Expected Value: 22.56 nA
P(C=0)↑→ Expected Value: 19.84 nA
P(C=1)↑→ Expected Value: 21.68 nA
P(D=0)↑→ Expected Value: 20.70 nA
P(D=1)↑→ Expected Value: 19.84 nA
P(E=0)↑→ Expected Value: 20.00 nA
P(E=1)↑→ Expected Value: 19.80 nA

Algorithm

- **Analysis and Discussion**
  - ✶ **Convergence rate of the algorithm is controlled by Rt**
  - ✶ **Convergence of MCNC C17**
    - A, B, C, D are monotonic.
    - E is NOT monotonic.



$CPV_I = (X, X, X, X, X)$

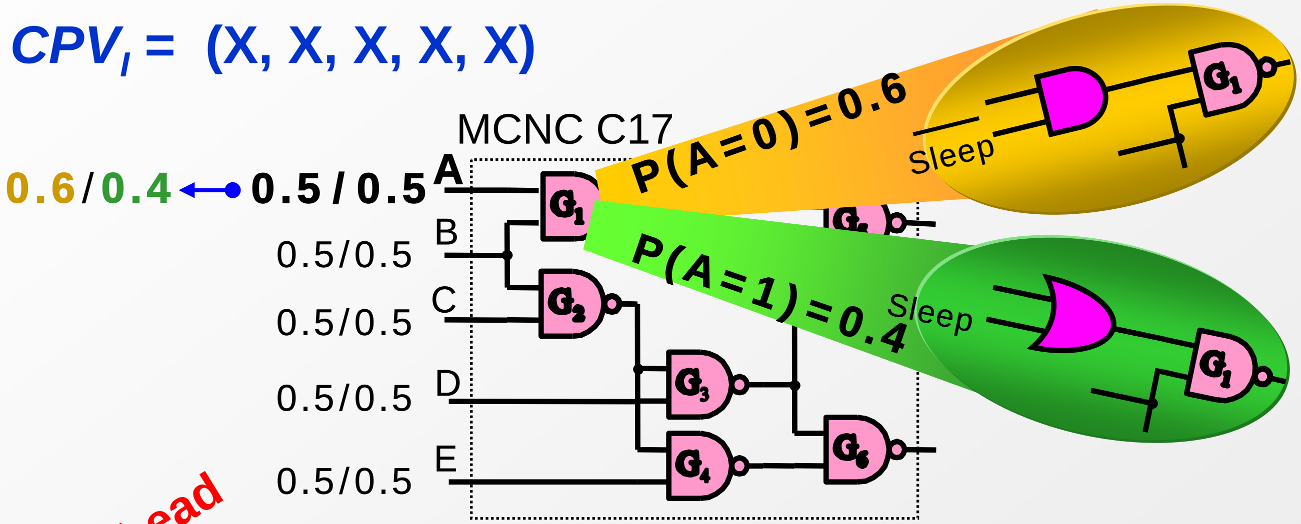
$CPV_M = (0, 0, 0, 1, 1)$

  - ✶ **Execute 10 iterations at most.**
  - ✶ **The initial Rt is set as 0.1 rather than 0.5**
    - For reducing the effect of decision order of control-points on the quality of CPV_M

Algorithm

- **Considering Controller Cost**

CPV_I = (X, X, X, X, X)



Controller Overhead

	w/o Controller Cost	w/ Controller Cost
0.6 * AND's Leakage	0	0.6 * 5.2 = 3.1
0.4 * OR's Leakage	0	0.4 * 4.5 = 1.8
Expected Value of Leakage of C17	27.3	27.3 + 3.1 + 1.8 = 32.2

Experiments

- **All Benchmark Circuits Are Synthesized by Design Compiler**
  - **Use 90nm CMOS standard cell library**
- **26 Small MCNC Benchmark Circuits**
  - **Our algorithm can find optimal solutions on 22 benchmark circuits**
  - **Average CPU time of our algorithm is 0.04 second**

Bench-mark	#PI	Leakage	
		Exhaustive Search	Our Algorithm
b1	3	2.6	2.6
cm42a	4	4.7	4.7
C17	5	1.5	1.5
cm82a	5	5.6	5.6
decod	5	4.7	4.7
cm138a	6	3.4	3.4
:	:	:	:
cmb	16	4.7	4.7
parity	16	16.4	16.4
pm1	16	7.0	7.0
t481	16	8.1	8.1
tcon	17	6.4	6.4
pcle	19	14.8	14.8
sct	19	15.4	15.4
cc	21	10.4	10.4
cm150a	21	11.6	11.6
Avg.	12	20.4	20.5

Experiments

- **12 Large Benchmark Circuits**
  - **CPU time of our program is averagely less than Random Search Program by 93%.**
  - **Average Controller Overhead**
    - Area is 6.5%; timing is 6.4%

Bench-mark	Random Search (w/o pin-reordering)		Our Algorithm			Reduction		
	Min. leakage (A)	Max. leakage (B)	w/o CPV Controller Leakage (C)	w/ CPV Controller Leakage (D)	CPU Time (s)	(A – C) / A	(B – C) / B	(B – D) / B
c3540	269	295	244	248	9.8	9%	17%	16%
c6288	578	684	450	453	31.0	22%	34%	34%
c7552	527	567	467	484	65.9	11%	18%	15%
i6	103	127	87	94	2.2	16%	31%	26%
:	:	:	:	:	:	:	:	:
Avg.	276	310	243	249	52.9	**11%**	**23%**	**20%**

Conclusions and Future Work

- **Conclusions**
  - ✶ **Presented a fast probability-based algorithm for constructing a minimum leakage CPV (*CPV_M*) used in the sleep mode.**
  - ✶ **Our algorithm can take the newborn controller into account.**
- **Future Work**
  - ✶ **Allow unfixed (X) elements appearing in *CPV_M*.**

Thank you

Q&A

Backup Slides

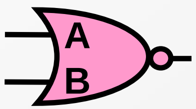
Backup Slides

Introduction

- **Related Algorithms/Techniques Used for Solving MLV Problems**
  - **Exact algorithms/techniques for small ckts.**
    - SAT solver
    - Integer linear programming
    - Branch and bound
  - **Heuristic algorithms/techniques for large ckts.**
    - Mixed-integer linear programming
    - Input controllability
    - Gate replacement
    - Our Algorithm
    - Etc.

Problem Definition

- **Pin Reordering**
  - **In fact, our algorithm employs a special leakage library for calculating the leakage current cost**
  - **Post-process the new design by pin reordering technique**



Original Leakage Current Library (NOR Gate)

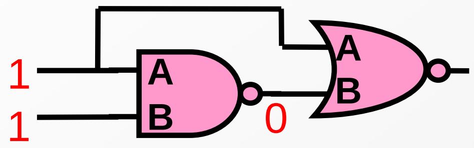
Input State (A B)	Leakage (nA)
00	7.7
01	2.4
10	4.8
11	0.7

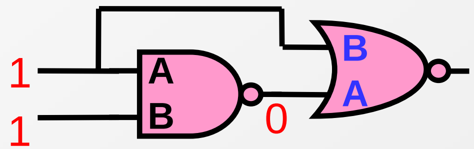
Special Leakage Current Library (NOR Gate)

Input State (A B)	Leakage (nA)
00	7.7
01	2.4
10	2.4
11	0.7

Problem Definition

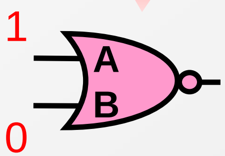
- **Pin Reordering**



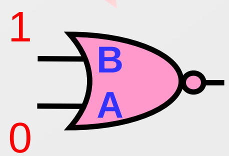


Leakage = 12.01 nA > **Leakage = 9.61 nA**

No such gate in our final design

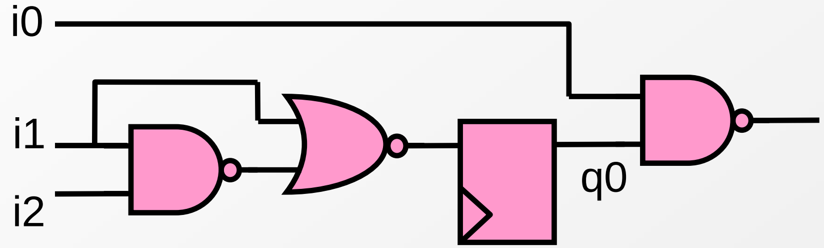


It can appear in our final design



Problem Definition

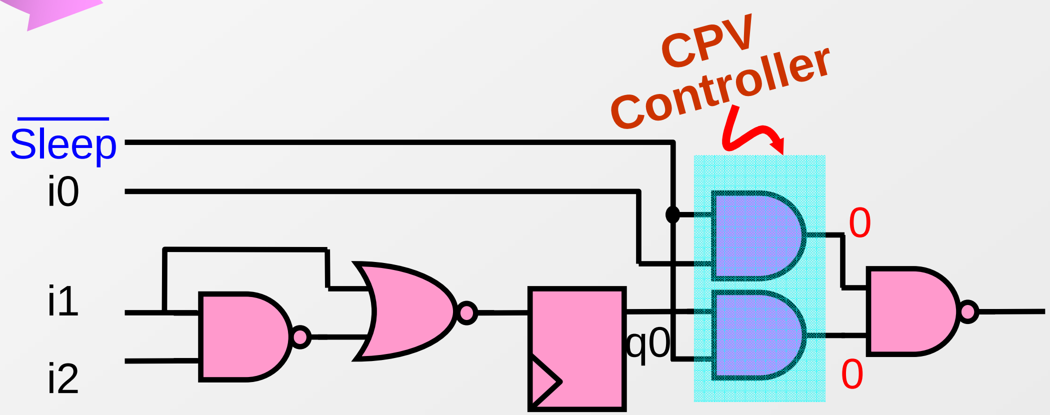
- **CPV Controller**



CPV = (i0, i1, i2, q0)

Inherent CPV = (X, 0, 0, X)

Minimum Leakage CPV = (0, 0, 0, 0)

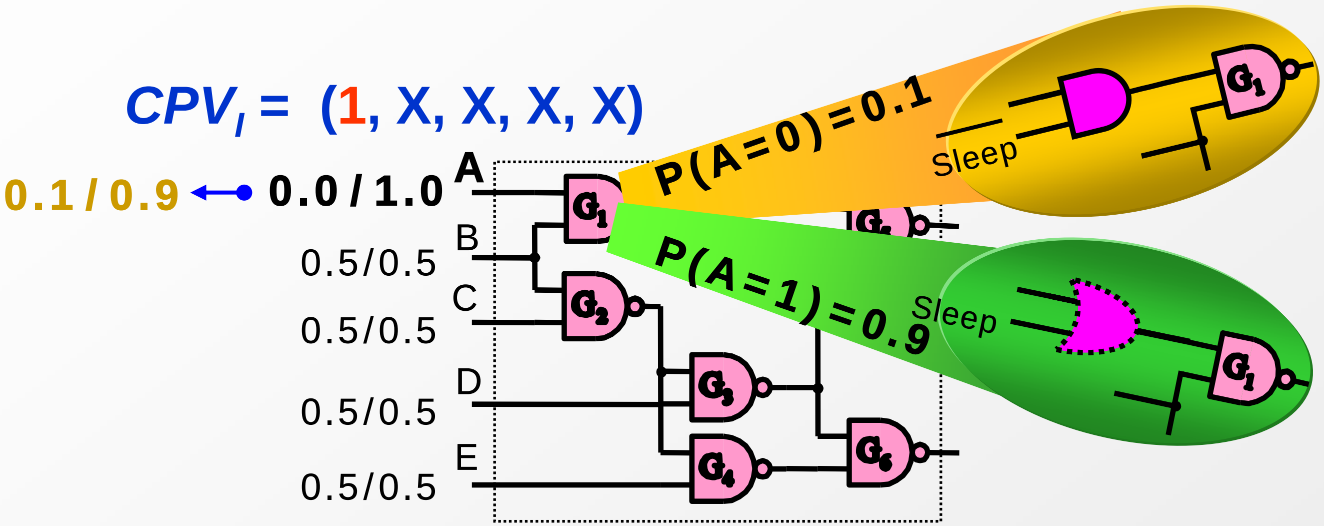


Cost Function

- **Probability Independent**
  - **The assumption of *probability independent* is necessary in our research!**
    - Without this assumption, the calculation of our cost function becomes an NP hard problem.
  - **E. Acar et al., "Leakage and Leakage Sensitivity Computation for Combinational Circuits," *ISLPED'03***
    - The paper had demonstrated that the accuracy error of expected leakage current is small under the assumption of probability independent

Algorithm

- **Issue of Controller Cost**



	w/o Controller Cost	w/ Controller Cost
AND's Leakage	0	0.1 * 5.2 = 0.5
OR's Leakage	0	0.9 * 0 = 0
Expected Value of Leakage	28.0	28.0 + 0.5 + 0 = 28.5

Experiments

- **Number of CPV generated by Random Search Program**
  - **10K ~ 100K**

Experiments

- **W/O Considering Controller Cost vs. W/ Considering Controller Cost**

Bench-mark	D_O's leak. (uA) (La)	Our Algorithm (Using Deterministic CPV_i)						Reduction (%)		
		cost function without considering *controller* (traditional techniques)			cost function with *considering controller*					
		D_N's leak. (Lb)	overhead (%)		D_N's leak. (Lc)	overhead (%)		$\frac{La-Lb}{La}$	$\frac{Lb-Lc}{Lb}$	a1–a2/ t1–t2
			area (a1)	timing (t1)		area (a2)	timing (t2)			
C499	36.40	31.74	11.3	2.2	30.82	1.4	2.2	12.8	4.3	9.9/0.0
C880	29.57	25.59	10.1	2.3	24.94	5.4	0	13.5	2.5	4.7/2.3
C7552	134.60	126.49	7.5	1.2	123.98	3.9	1.2	6.0	2.0	3.6/0.0
i6	32.41	23.88	22.2	3.4	23.13	12.9	0	26.3	3.1	9.3/3.4
i7	39.91	29.08	25.2	2.8	28.36	8.7	2.8	27.1	2.5	16.5/0.0
i9	48.85	34.85	8.0	3.1	33.85	6.0	3.1	28.7	2.9	2.0/0.0
:	:	:	:	:	:	:	:	:	:	: