Low Power Design: Current Practice and Opportunities

Gang Qu
University of Maryland, College Park
ASPDAC Tutorial, Incheon, South Korea
January 22, 2024

MeshSec Lab

1

Contents of this Tutorial

- Low power basics and scope of the tutorial
- Current low power design methodologies and techniques
  - Device level low power and leakage reduction
  - Dynamic power reduction with fixed Vdd
  - Dynamic voltage and frequency scaling
- Approximate computing for low power
  - Opportunities in neural network models
- Low power and security

2

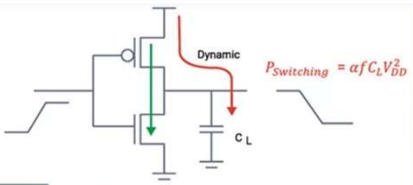
Low Power Basics

- Need of low power design
  - Longer battery life (fewer charging)
  - Less package cost
  - More reliable circuitry (and the system)
- Power vs energy
- Average power vs peak power
- Low power/energy design vs power/energy aware computing

Sources of Power Dissipation

- Dynamic power
- Short circuit power
- Leakage (static) power
  - Gate-oxide leakage
  - Subthreshold leakage
- Low power design principle
  - Simply reduce the parameters
  - But …

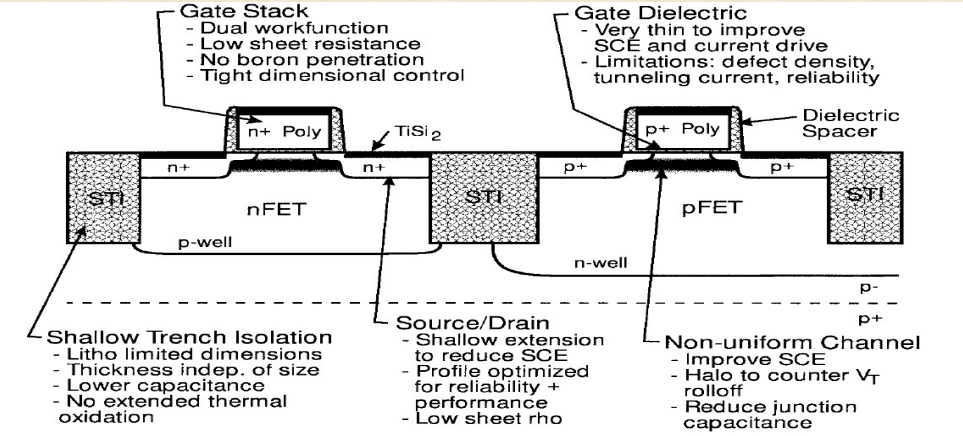
$$P_{Total} = \alpha f C_L V_{DD}^2 + t_{sc} V_{DD} I_{peak} + V_{DD} I_{leakage}$$



α = activity factor (0 to 1)
f = frequency
t_{sc} = transition time
C_L = capacitive load
V_{DD} = supply voltage
$I_{leakage}$ = leakage current
I_{peak} = peak current

https://www.synopsys.com/glossary/what-is-low-power-design.html

Bulk CMOS Cross-Section



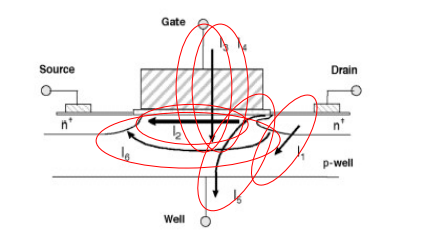
Wong et al. ***Nanoscale CMOS.*** Proc. IEEE, 87, pp. 537-570, 1999.

Low Power at Device Level

- Body- and Back-Gate Bias
  - Makes body bias to be adjustable
  - Allows the threshold voltages to be adjusted after manufacture
  - Used in commercial SRAM/DRAM chips
- Strained Si MOSFET
  - Use biaxially tensile strained Si in the channel
  - Caused higher drive current: good for speed or voltage and power
- Fully-Depleted SOI
  - Si layer becomes much thinner
- Double-gate FET
  - uses a second gate below the channel
  - Screens the drain field to reduce short channel effect

Sources of Leakage Current

- I_1: Junction leakage
- I_2: Subthreshold
- I_3: Gate oxide tunneling
- I_4: Hot-carrier injection
- I_5: Gate-induced drain leakage
- I_6: Punchthrough leakage
- Long channel (L>1μm) — very small
- Short channel (L>180nm, T_{ox}>30A°) — I_2
- Very short channel (L>90nm, T_{ox}>20A°) — I_2+ I_3+ I_4
- Nano-scaled (L<90nm, T_{ox}<20A°) — all



Roy et al. ***Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-Submicrometer CMOS Circuits***. Proc. IEEE, 91, pp. 305-327, 2003.

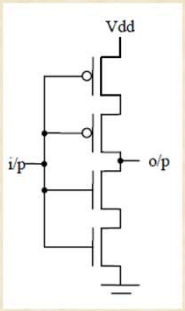
Leakage Reduction Techniques

- Process level techniques
  - control dimensions (gate oxide thickness, junction depth scaling)
  - well engineering (retrograde doping, halo doping)
- Circuit level techniques
  - transistor stacking and input vector control
  - Power gating
  - multiple threshold voltage V_{th}
  - dual threshold voltage V_{th}
  - supply voltage scaling

$$I_{sub} = \frac{W}{L}\mu_e v_T^2 C_{sth}\, e^{\frac{V_{GS}-V_{th}+\eta V_{DS}}{nv_T}}\left(1-e^{\frac{-V_{DS}}{nv_T}}\right)$$

Transistor Stacking

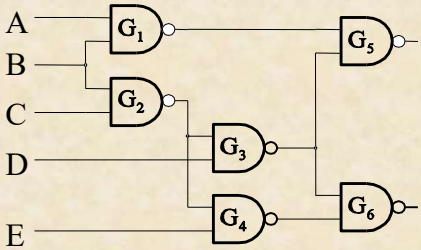
- Stack effect: threshold voltage of a transistor decreases as the channel length decreases and the electric field in the channel region increases.
- Transistor stacking: replacing a transistor with a series combination of two transistors.
  - Leakage current can be reduced significantly.
  - Delay will be increased (performance issue), reliability and lifespan of the integrated circuit will be reduced.



Input Vector Control

Input	Leakage(*nA*)
00	37.84
01	100.30
10	95.17
11	454.50

Leakage current in a 2-input NAND gate



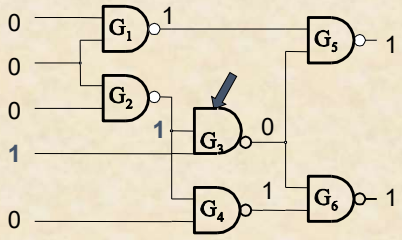
- Random input vector
  - 00000: 1175.02 nA
  - 11111: 1463.80 nA
- Minimum leakage vector
  - 00010: 831.08 nA

- Idea: find an MLV (NP hard) and apply it when the circuit is idle.
  - Exact methods: branch and bound, ILP, Pseudo Boolean SAT…
  - Heuristics: random search, genetic algorithm …

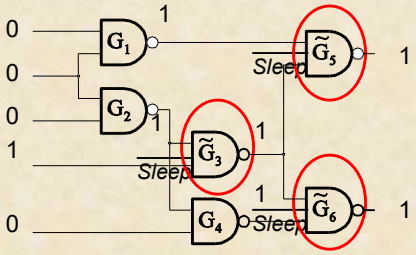
Input Vector Control + Gate Replacement

Input	Leakage(*nA*)
00	37.84
01	100.30
10	95.17
11	454.50

Leakage current in a 2-input NAND gate



- Minimum leakage vector
  - 00010: 831.08 nA
  - G3 only: 454.50 nA
- 3-input NAND
  - 011: 100.30 nA

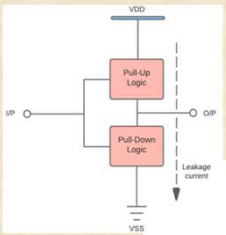


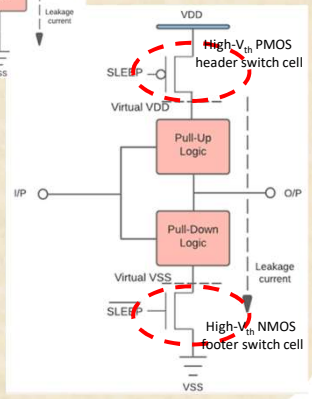
- Minimum leakage vector
  - 00010: 831.08 nA
- Gate replacement
  - 00010: 476.88 nA

Yuan and Qu, **A combined gate replacement and input vector control approach for leakage current reduction.** TVLSI, Vol. 14, No. pp 173-182, 2006.

Power Gating

- Idea: turn off inactive parts of circuit to reduce power consumption.
- Implementation: leakage current flows from VDD to the ground, disconnect the path by adding a power gating circuit.
  - Introduce the SLEEP signal that indicates the active or inactive mode of operation.
  - Use SLEEP to control the sleep transistors, also called switch cells, to turn the circuit on/off.

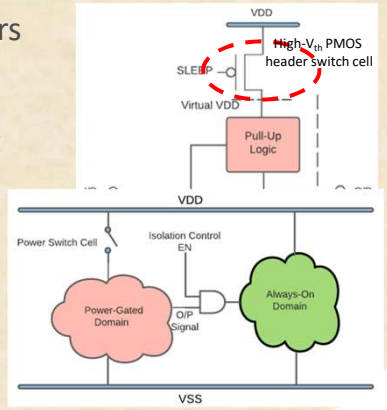




https://anysilicon.com/power-gating

Power Gating

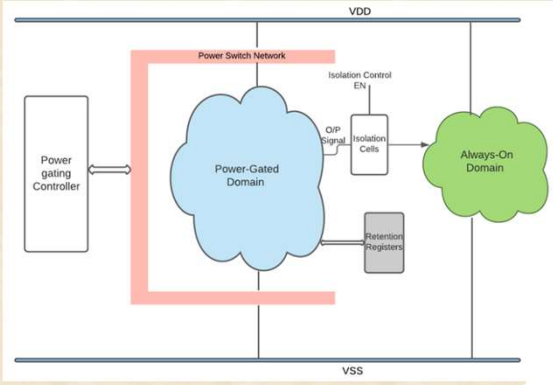
- Implementation:
  - SLEEP = 0: both PMOS and NMOS sleep transistors are on; both logic are connected to virtual VDD and VSS, working in normal mode.
  - SLEEP = 1: PMOS and NMOS sleep transistors are off; VDD to VSS path is disconnected; leakage is reduced.
- Isolation cell:
  - Normal mode: buffer
  - Low power mode: a constant logic (0 or 1)
  - Implemented with an OR or AND gate



https://anysilicon.com/power-gating

Power Gating

- effective in reducing leakage
- Trade-off:
  - cost of the added logic (power control logic, isolation cells, retention cells, etc.)
  - power and delay to enter/exit the low power mode
  - added complexity to other stages of the design flow (synthesis, DFT, verification, physical design, etc.)



https://anysilicon.com/power-gating

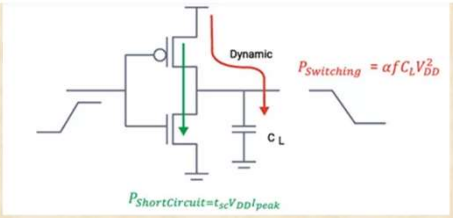
Multi-threshold and Dual Threshold

- Principle: high V_{th} device has low leakage but high delay.
- Multi-threoshold CMOS (MTCMOS):
  - Use high V_{th} MOS device for sleep transistor to reduce leakage in standby mode.
  - Use low V_{th} MOS device for logic circuit to maintain performance in normal active mode.
- Dual Threshold MOS (DTCMOS)
  - Use high V_{th} MOS on non-critical path to reduce leakage
  - Use low V_{th} MOS on critical path to maintain performance

Dynamic Power Reduction with Fixed Vdd

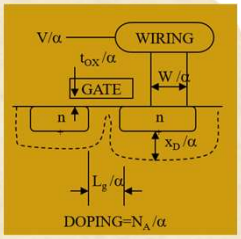
- Transistor sizing
- Device scaling
- Frequency scaling
- Glitching and path balancing
- Clock gating
- Optimization:
  - don't care conditions
  - Decomposition
  - Re-timing
  - Technology mapping
  - Pre-computation
  - Finite state machine synthesis

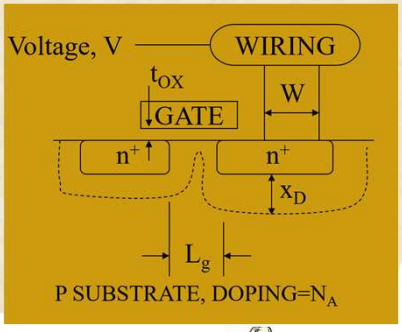
$$P_{Total} = \alpha f C_L V_{DD}^2 + t_{sc} V_{DD} I_{peak} + V_{DD} I_{leakage}$$



Scaling of Silicon Device Technology

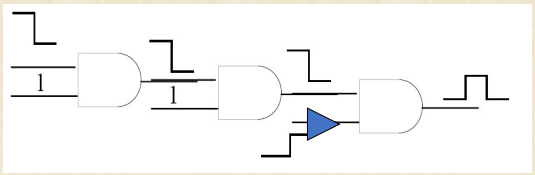
- α:constant-electric field scaling factor
- ε: generalized scaling factor ($\varepsilon > 1$)
- Generalized selective scaling factors
  - α_d: gate length and device vertical scaling
  - α_w: device width and wiring scaling ($\alpha_d > \alpha_w$)
- Area: $1/\alpha^2$, $1/\alpha^2$, $1/\alpha_w{}^2$
- Gate delay: $1/\alpha$, $1/\alpha$, $1/\alpha_d$
- Power dissipation: $1/\alpha^2$, ε^2/α^2, $\varepsilon^2/\alpha_d\,\alpha_w$
- Power density: 1, ε^2, $\varepsilon^2\,\alpha_w/\alpha_d$

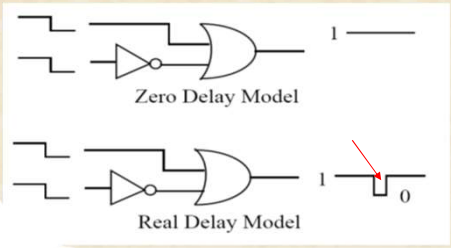




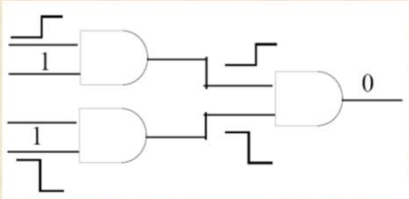
Glitching and Path Balancing

- What is glitching
- Why it matters
  - Reliability
  - Switching power





- Path balancing
  - Logic restructure
  - Buffer insertion

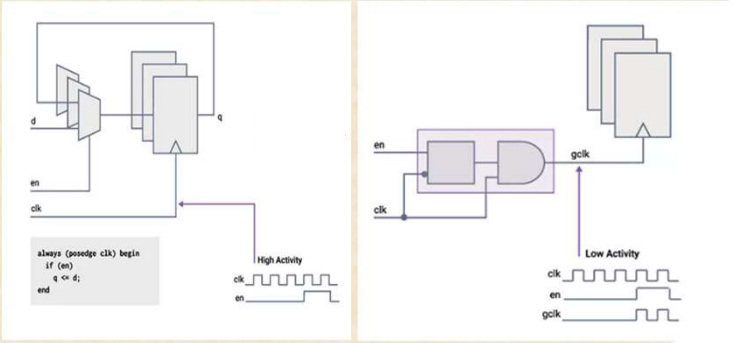


Pedram. **Power Minimization in IC Design: Principles and Applications**, TODAES, Vol. 1, No. 1, pp. 3-56, 1996.

Clock Gating

$$P_{Total} = \alpha f C_L V_{DD}^2 + t_{sc} V_{DD} I_{peak} + V_{DD} I_{leakage}$$

- Idea: disable idle logic blocks
- Implementation in synchronous design: turn off clock
- Cost: control circuitry
- Efficient and effective
- Commercial tools available, e.g. grouping circuit and clock routing



https://www.synopsys.com/glossary/what-is-low-power-design.html

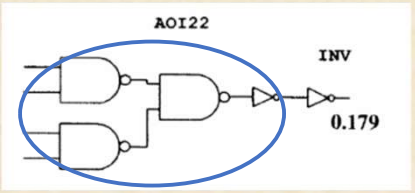
Don't Care Condition Optimization

$$P_{Total} = \alpha f C_L V_{DD}^2 + t_{sc} V_{DD} I_{peak} + V_{DD} I_{leakage}$$

- Probabilities:
  - Static probability: p(x), p(x')
  - Transition probability: $p^{ij}(x)$
- Switching probability of a gate with output signal x
  - $p^{01}(x)+p^{10}(x) = 2p(x)p(x')$
  - When will this be minimized? p(x) or p(x') = 0
- Idea of don't care optimization
  - If p(x) < 0.5, set don't care to output 0
  - If p(x) > 0.5, set don't care to output 1

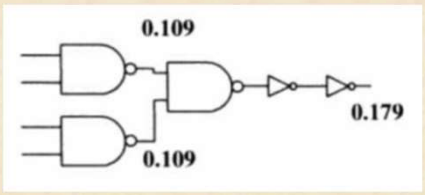
Technology Mapping: Objective Function

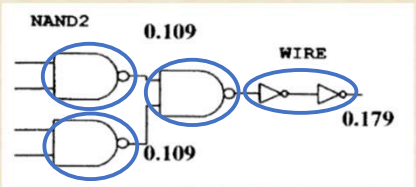
Gate	Area	Intrinsic Capacitance	Input Load Capacitance
INV	928	0.1029	0.0514
NAND2	1392	0.1421	0.0747
AOI22	2320	0.3410	0.1033



Area: 2320+928=3248

αc:
0.179(0.3410+0.0514)+
0.179X0.1029 = 0.0887





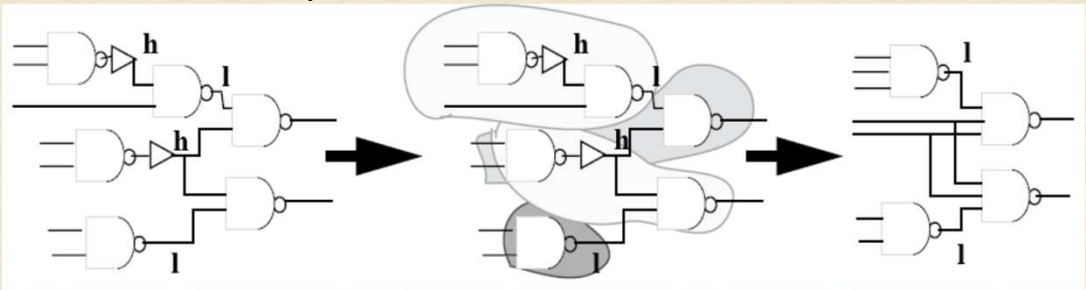
Area: 1392x3 = 4176

αc:
0.109(0.1421+0.0747)X2
+ 0.179X0.1421 = 0.0726

Monteiro and Devadas, ***Computer-Aided Design Techniques for Low Power Sequential Logic Circuits***, KAP, 1997.

Low Power-Driven Technology Mapping

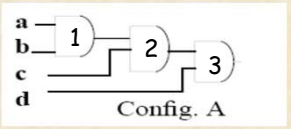
- Idea: hide nodes with high switching activity inside the library cells that have small load capacitances.
- Cost: area, delay, or other



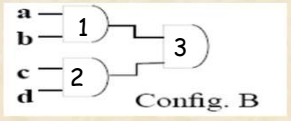
Pedram. **Power Minimization in IC Design: Principles and Applications**, TODAES, Vol. 1, No. 1, pp. 3-56, 1996.

Decomposition

- Example 1. F = ab+ac+bc
  - Three different decompositions: a(b+c) + bc, b(a+c) + ac, c(a+b) + ab
  - They, and ab+ac+bc, may have different total switching activities.
- Example 2. F(a,b,c,d) = abcd
  - p(a) = 0.3, p(b) = 0.4, p(c) = 0.7, p(d) = 0.5
  - p(1) + p(2) + p(3) = ?
  - Config. A: p(a)p(b)+p(1)p(c)+p(2)p(d) = 0.246
  - Config. B: p(a)p(b)+p(c)p(d)+p(1)p(2) = 0.512



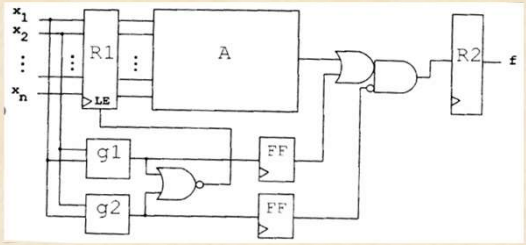
Config. A



Config. B

Pre-computation

- Basic idea
  - Add precomputation logic to compute the output for a subset of input cases (adds power, area, delay, etc.)
    - Given $f(x_1,\ldots,x_n)$, determine $g_1(x_1,\ldots,x_m)$ and $g_2(x_1,\ldots,x_m)$ s.t.
    - $g_1(x_1,\ldots,x_m) = 1 \rightarrow f(x_1,\ldots,x_n) = 1$
    - $g_2(x_1,\ldots,x_m) = 1 \rightarrow f(x_1,\ldots,x_n) = 0$
  - Turn off the original circuit or part of it for the next clock cycle, no switching on the off circuit (saves power)



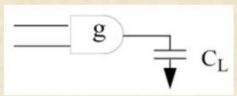
- $g_1 = g_2 = 0$, same as before with extra delay
- $g_1 = 1$, $g_2 = 0$, A is disabled to save power, f=1
- $g_1 = 0$, $g_2 = 1$, A is disabled to save power, f=0

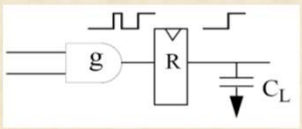
Retiming for Low Power

- Idea: reducing $\Sigma\alpha C_L$

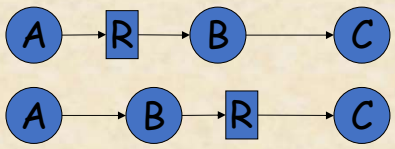
$$P_{Total} = \alpha f C_L V_{DD}^2 + t_{sc} V_{DD} I_{peak} + V_{DD} I_{leakage}$$

- Add a flip flop
  - $\alpha_g C_L$
  - $\alpha_g C_R + \alpha_R C_L$



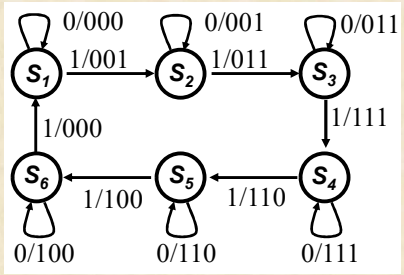


- Move a flip flop
  - $\alpha_0 C_R + \alpha_1 C_B + \alpha_2 C_C$
  - $\alpha'_0 C_B + \alpha'_1 C_R + \alpha'_2 C_C$



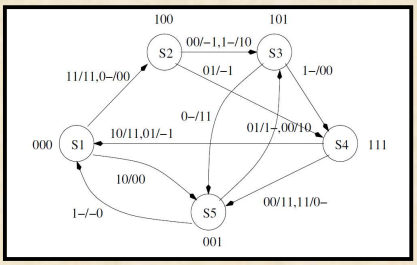
State Encoding in Finite State Machine

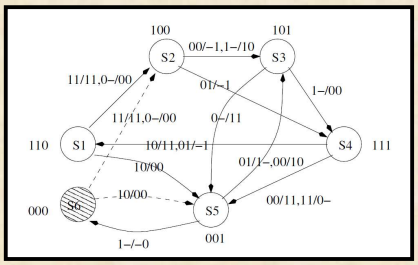
- Sequential binary assignment:
  - S_1=001, S_2=010, S_3=011, S_4=100, S_5=101, S_6=110
  - Average bits to be changed:
    [(0+2)+(0+1)+(0+3)+(0+1)+(0+2)+(0+3)]/12 = 1
- Ad hoc binary assignment:
  - S_1=000, S_2=001, S_3=011, S_4=111, S_5=110, S_6=100
  - Average bits to be changed:
    [(0+1)+(0+1)+(0+1)+(0+1)+(0+1)+(0+1)]/12 = 0.5



- One-hot encoding:
  - S_1=000001, S_2=000010, S_3=000100, S_4=001000, S_5=010000, S_6=100000

Finite State Machine Re-engineering





# of states n = 5

# of bits k = 3

design space: $\frac{2^k!}{(2^k-n)!}$ = 6920

switching activity: **1.27**

switching activity: **1.17**

# of states n = 6

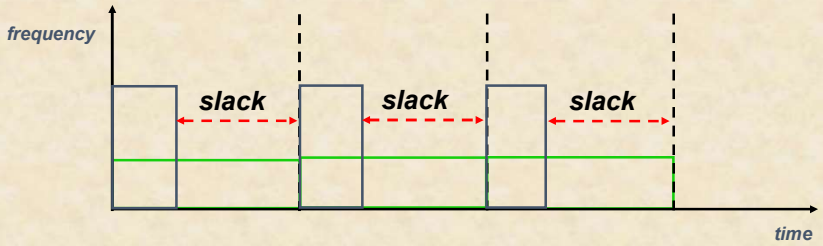
# of bits k = 3

design space: $\frac{2^k!}{(2^k-n)!}$ = 20160

Yuan et al. ***FSM Re-Engineering: A Novel Approach to Sequential Circuit Synthesis***, TCAD, Vol. 27, No. 6, pp. 1159-1164, 2008.

Dynamic Voltage and Frequency Scaling

- Suppose that a data sample comes every 1 ms
- Requires processing time of 250 μs at 600MHz
- DVFS: reduce voltage such that clock slows down to 150MHz



DVS System Modeling

If s(t) and p(t) are the DVS processor's processing speed and power consumption at time t, then

the work completed from T_1 to T_2 is $\int_{T_1}^{T_2} s(t)\,dt$,

the energy consumption is $\int_{T_1}^{T_2} p(t)\,dt$.

- Ideal: whenever, whatever, no switching delay
- Multiple: checkpoint, discrete, no switching delay
- Feasible: $[v_{min}, v_{max}]$, maximal change rate, CPU processes at $v(t)$ during voltage transition
- Practical: stop processing until the steady state is reached

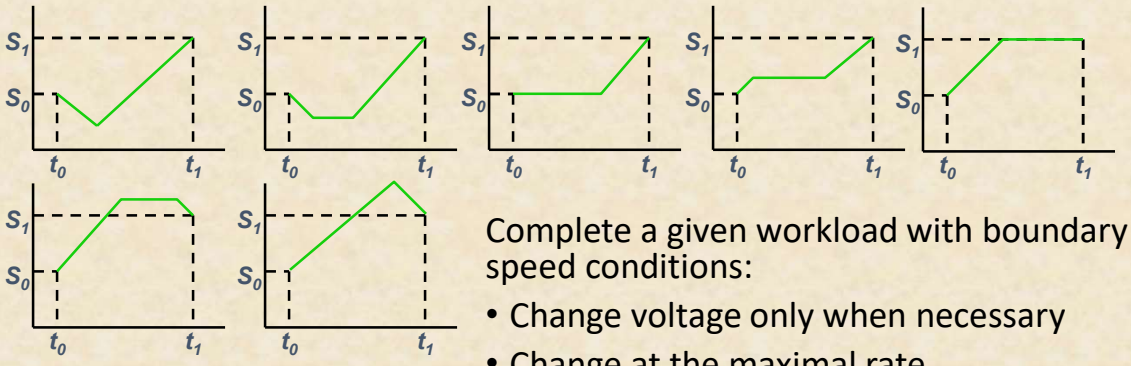
DVS System Modeling

	Ideal	Feasible	Practical	Multiple
Voltage change continuously	√	√	√	×
Maximal and minimal voltages	×	√	√	√
Voltage change rate	×	√	√	×
Need time to reach steady state	×	√	√	×
Processing during voltage change		√	×	

Energy saving

G. Qu. "Scheduling Problems for Reducing Energy on Variable Voltage Systems", M.S. Thesis, UCLA 1996.

Optimal Solution for Feasible DVS System



Complete a given workload with boundary speed conditions:

- Change voltage only when necessary
- Change at the maximal rate
- Time(s) when voltage changes is calculable

Yuan and Qu. "What Is the Limit of Energy Saving by Dynamic Voltage Scaling", ICCAD'2001 .

Voltage Set-up on Multiple DVS System

- For a multiple-voltage DVS system to serve a set of applications $\{(e_i, d_i, p_i): i=1, 2, \ldots, n\}$ without missing their deadlines, where e_i: execution time d_i: deadline, p_i: probability d_i occurs.
  - if the system has m voltages $\{v_1, v_2, \ldots, v_m\}$, determine the value of each v_i to minimize the average energy consumption.
  - determine m and the value of each v_i.

Hua and Qu. "Approaching the Maximum Energy Saving on Embedded Systems with Multiple Voltages", ICCAD, 2003 .

Information of Two Applications

Application	Deadline	Execution Time	Probability	(V)
A	10	9	0.03	3.0564
		4	0.18	1.8124
		3	0.39	1.5516
B	8	6	0.04	2.6888
		4	0.10	2.0669
		3	0.12	1.7479
		2	0.14	1.4176

V_{ref} = 3.3v

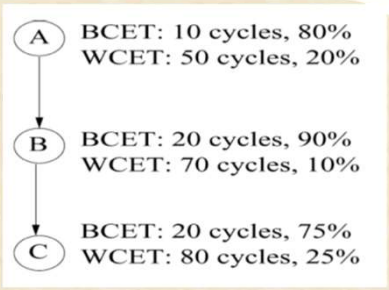
DVS Systems	Voltages	Energy	vs. fixed-voltage	vs. Ideal
fixed-voltage	3.0564	2.9536	--	+151.1%
dual-voltage	3.0564 1.8124	1.3833	- 53.2%	+17.6%
3-voltage	3.0564 2.0688 1.5514	1.2337	- 58.2%	+4.9%
4-voltage	3.0564 2.0768 1.8119 1.5509	1.2071	- 59.1%	+2.6%
ideal	--	1.1763	--	--

For most embedded systems, a couple of well-selected voltage levels will be sufficient in energy saving.

Hua and Qu. "Approaching the Maximum Energy Saving on Embedded Systems with Multiple Voltages", ICCAD, 2003 .

Probabilistic Design

- Design of Real time systems:
  - Performance guarantee
  - WCET-based design
- New applications:
  - Iterative execution of streamline data
  - Soft real time
  - Execution time varies, but not "pure" random
- Over designed systems
  - More CPU
  - More energy/power



	A	B	C	RET	Prob
1	10	20	20	50	54.0%
2	50	20	20	90	13.5%
3	10	70	20	100	6.0%
4	10	20	80	110	18.0%
5	50	70	20	140	1.5%
6	50	20	80	150	4.5%
7	10	70	80	160	2.0%
8	50	70	80	200	0.5%

Scheduling for Best Energy Saving

I: run at V_1 to completion or deadline 10.

II: run at V_1 for A; use V_1 or the slowest to complete B by 5, terminate if not done by 8; use the slowest to complete C by 10.

III: allocates 1, 7, 2 to A,B,C. If cannot be completed at V_1, terminates; otherwise use the slowest speed.

Task	BCET	WCET
$\mathcal{A}$	(1, 80%)	(6, 20%)
$\mathcal{B}$	(2, 90%)	(7, 10%)
$\mathcal{C}$	(2, 75%)	(5, 25%)

Voltage (V)	Power	Delay
$v_1 = 3.3$	1	1
$v_2 = 2.4$	0.30	1.8
$v_3 = 1.8$	0.09	3.4

	$\mathcal{Q}(\%)$	$t@v_1$	$t@v_2$	$t@v_3$	E	E@($\mathcal{Q} = 60\%$)
I	91.5	6.94	0	0	6.94	4.55
II	91.5	4.21	4.54	0	5.57	3.65
III	60	2.56	0	4.90	3.00	3.00

	Naive	BEEM		OOME			
Benchmark	energy	energy	Saving over Naive	energy	Saving over Naive	Saving over BEEM	Q
FFT1	699.69	539.02	22.96%	371.00	46.98%	31.17%	0.804
FFT2	441.91	259.08	41.37%	199.14	54.94%	23.14%	0.804
Laplace	1322.2	724.39	45.21%	571.42	56.78%	21.12%	0.822
qmf4	147.65	72.84	50.67%	67.64	54.19%	7.13%	0.832
karp10	547.03	382.83	30.02%	264.57	51.64%	30.89%	0.823
meas	260.10	145.82	43.94%	115.37	55.64%	20.88%	0.803
sum1	80.30	57.08	28.91%	39.60	50.69%	30.63%	0.819
almu	66.03	35.89	45.65%	28.60	56.68%	20.31%	0.813
DSC-7-7	12.18	8.19	32.77%	5.32	56.29%	34.99%	0.809
DSC-7-8	12.18	7.65	37.14%	5.18	57.43%	32.28%	0.809
average saving			37.86%	---	54.12%	26.16%	---

Naïve: I

BEEM: II
Best effort energy minimization

OOME: III
Online offline minimal effort

Hua et al. "**Energy Reduction Techniques for Multimedia Applications with Tolerance to Deadline Misses**", DAC, 2003.

Approximate Computing

- A broad range of techniques that trade computation quality for power/energy efficiency.
- Applicable applications:
  - No need to use accurate computation
  - Error tolerable
  - IoT, AI/ML, image/signal processing, …
- Approximation at different levels
  - Arithmetic, software, compiler, architecture, memory, circuit, …
  - Could be combined with DVFS and other techniques

Arithmetic Units with Different Precisions

	cells	area	power(nW)
8-bit checker	10	23.93	61475.68
16-bit checker	17	39.89	132145.68
8-bit adder	91	212.12	752614.84
16-bit adder	230	322.33	2234652.24
32-bit adder	498	1116.93	4818821.94
8-bit multiplier	377	1037.62	2829745.1
16-bit multiplier	1406	4208.68	10815807.06
32-bit multiplier	4916	15126.01	34033690.3

Approximate Arithmetic

- Example: Compute S = A + B

$A = 0011\ 1010\ 0001\ 1000_2$

$B = 0101\ 1011\ 1011\ 1000_2$

$0000\ 0000\ 0111\ 1001_2 = 121$

$+\ 0000\ 0000\ 1011\ 1000_2 = 184$

$=\ 0000\ 0000\ 1111\ 1001_2 = 249$

$1\ 0011\ 0001_2 = 305$

Error = 18.4% !

- Build a 16-bit approximate adder that:
  - 8-bit accurate adder for high 8 bits
  - 8 OR gates for low 8 bits

$0111\ 1010\ 0111\ 1001_2 = 31353$

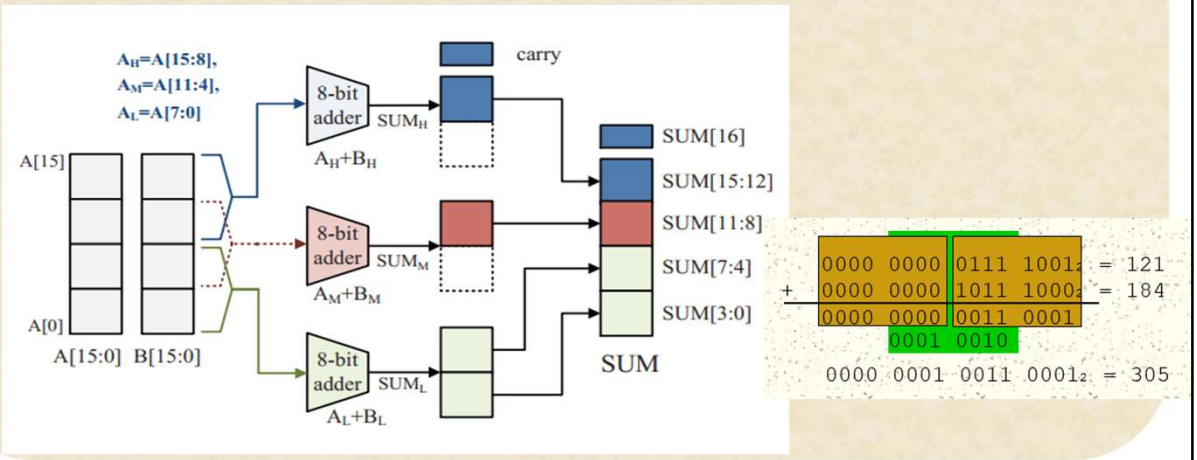
$+\ 0101\ 1011\ 1011\ 1000_2 = 23480$

$=\ 1101\ 0101\ 1111\ 1001_2 = 54777$

Error = 0.102% !

Mahdiani et al, "**Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications**," TCAS-I, vol. 57, no. 4, pp. 850-862, 2010.

Configurable Adder



Kahng and Kang. "**Accuracy-configurable adder for approximate arithmetic designs**". *DAC 2012.*

A Dynamic Approximate Integer Format

- Given a 4-block operand A= $b_3b_2b_1b_0$.
  - Only five possible values of A's sentinel bits st_a: 0000, 0001, 0011, 0111, 1111.
  - For the first four cases, the data A will be stored in following format:

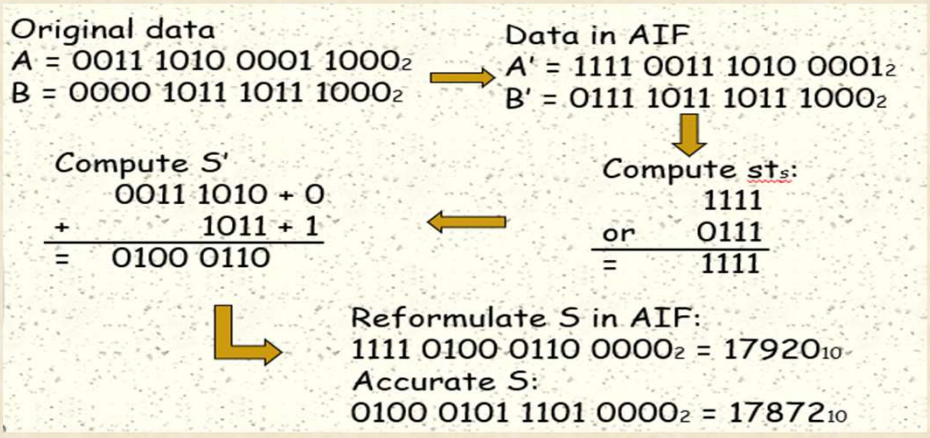
st_a	b_2	b_1	b_0

  - For the last case of 1111, A will be stored as:

st_a	b_3	b_2	b_1

Gao et al, "**A Novel Data Format for Approximate Arithmetic Computing**", *ASPDAC 2017.*

Example: Addition Using AIF



Gao et al, "**A Novel Data Format for Approximate Arithmetic Computing**", *ASPDAC* 2017.

Results on Fibonacci Sequence

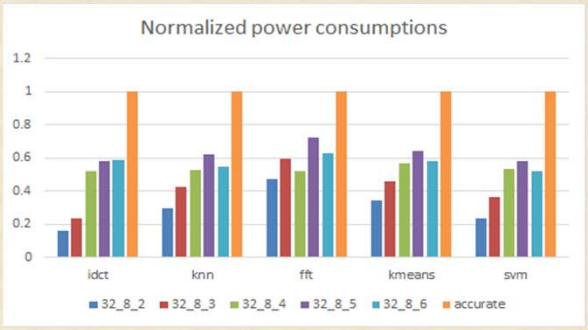
- Fibonacci sequence: a number is the sum of its previous two: 1, 1, 2, 3, 5, 8, 13, 21, 34, …

#	pc=2	pc=3	pc=4	#	pc=2	pc=3	pc=4	#	pc=2	pc=3	pc=4
1~13	0	0	0	22	-0.02732	3.49E-05	0	31	-0.02291	-0.00035	2.30E-06
14	-0.00984	0	0	23	-0.02692	0	0	32	-0.02267	-0.00059	-8.51E-06
15	-0.01317	0	0	24	-0.02707	1.33E-05	0	33	-0.02276	-0.0005	-4.38E-06
16	-0.01691	0	0	25	-0.02912	0.000404	8.24E-06	34	-0.02273	-0.00053	-5.96E-06
17	-0.01858	0	0	26	-0.03225	0.000336	1.02E-05	35	-0.02274	-0.00052	-5.36E-06
18	-0.01985	0	0	27	-0.0375	0.000362	9.44E-06	36	-0.02274	-0.00052	-5.59E-06
19	-0.02173	-0.00044	0	28	-0.03947	0.000352	9.72E-06	37	-0.0328	-0.0001	1.05E-06
20	-0.02905	0.000183	0	29	-0.04118	0.000356	9.61E-06	38	-0.03621	-0.00046	-5.53E-06
21	-0.02625	-5.65E-05	0	30	-0.04205	0.000354	9.66E-06	39	-0.04003	-0.00064	-3.02E-06
								40	-0.04174	-0.00077	-3.98E-06

Gao et al, "**A Novel Data Format for Approximate Arithmetic Computing**", *ASPDAC* 2017.

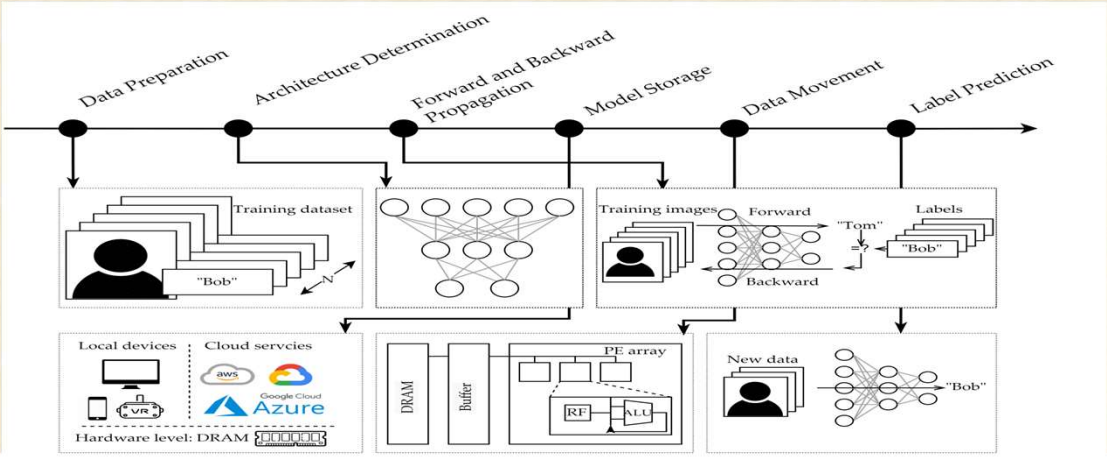
Results on Real Life Applications

AIF modules	IDCT	KNN	FFT	Kmeans	SVM
32_8_2	41.7579	92.9%	0.0081	1.125%	60.94%
32_8_3	64.6398	93.2%	2.79E-04	0.125%	85.11%
32_8_4	89.8324	93.1%	1.61E-05	0	85.98%
32_8_5	112.0872	93.2%	3.02E-06	0	85.59%
32_8_6	116.2944	93.2%	7.11E-08	0	86.42%
baseline	116.2949	93.2%	0	0	86.42%
Error Metric	PSNR	Classification accuracy	ARES	miss-clustered%	Classification accuracy

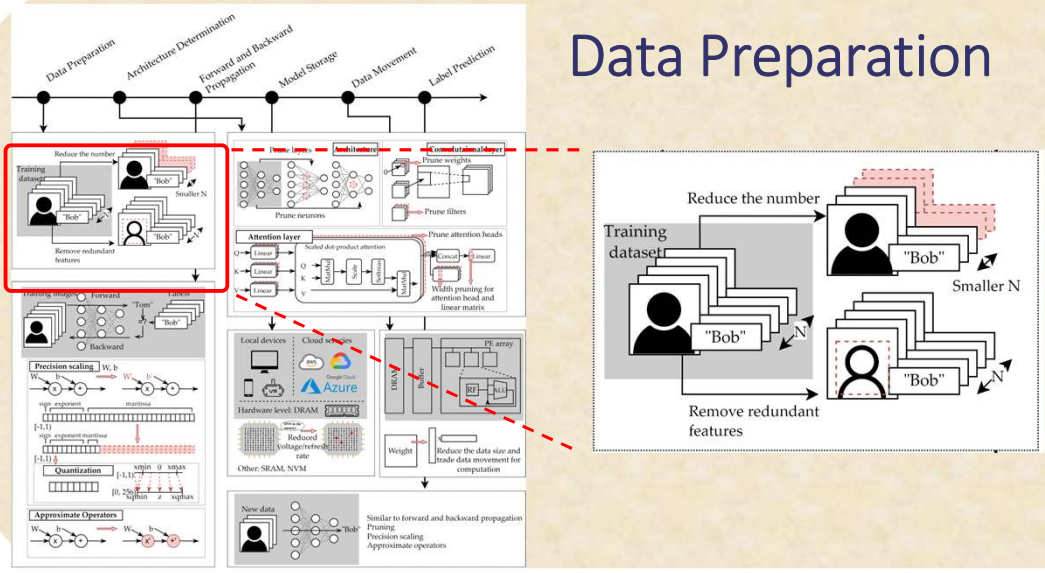


Gao et al, "**A Novel Data Format for Approximate Arithmetic Computing**", *ASPDAC* 2017.

Neural Network Flow

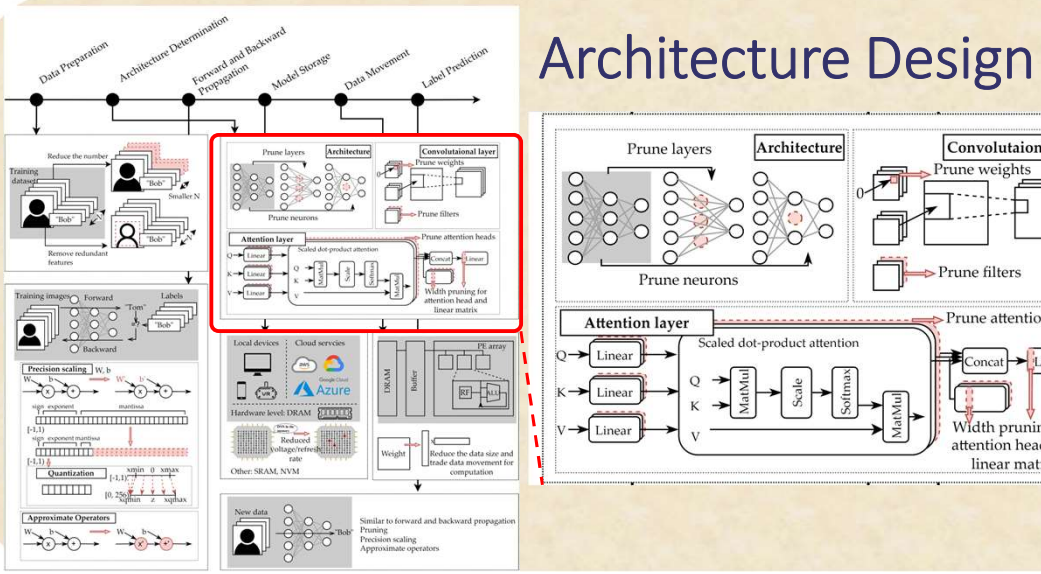


Data Preparation



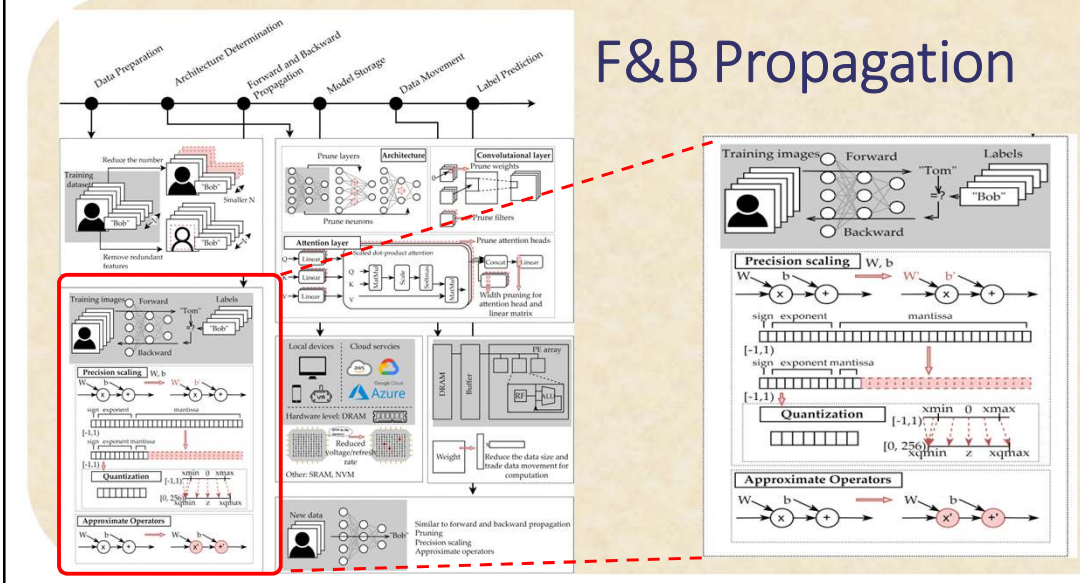
47

Architecture Design

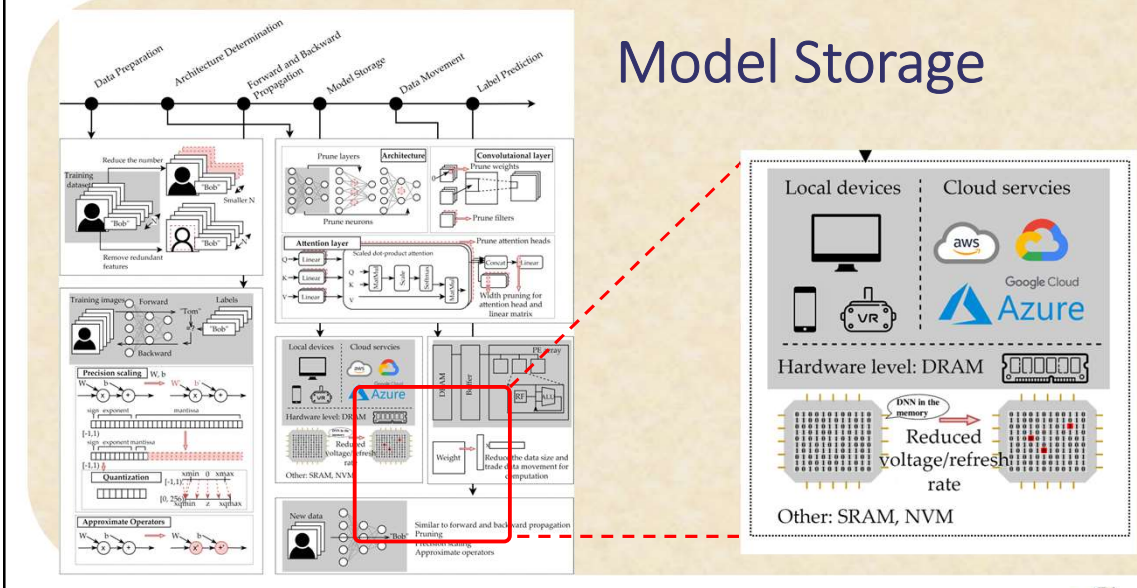


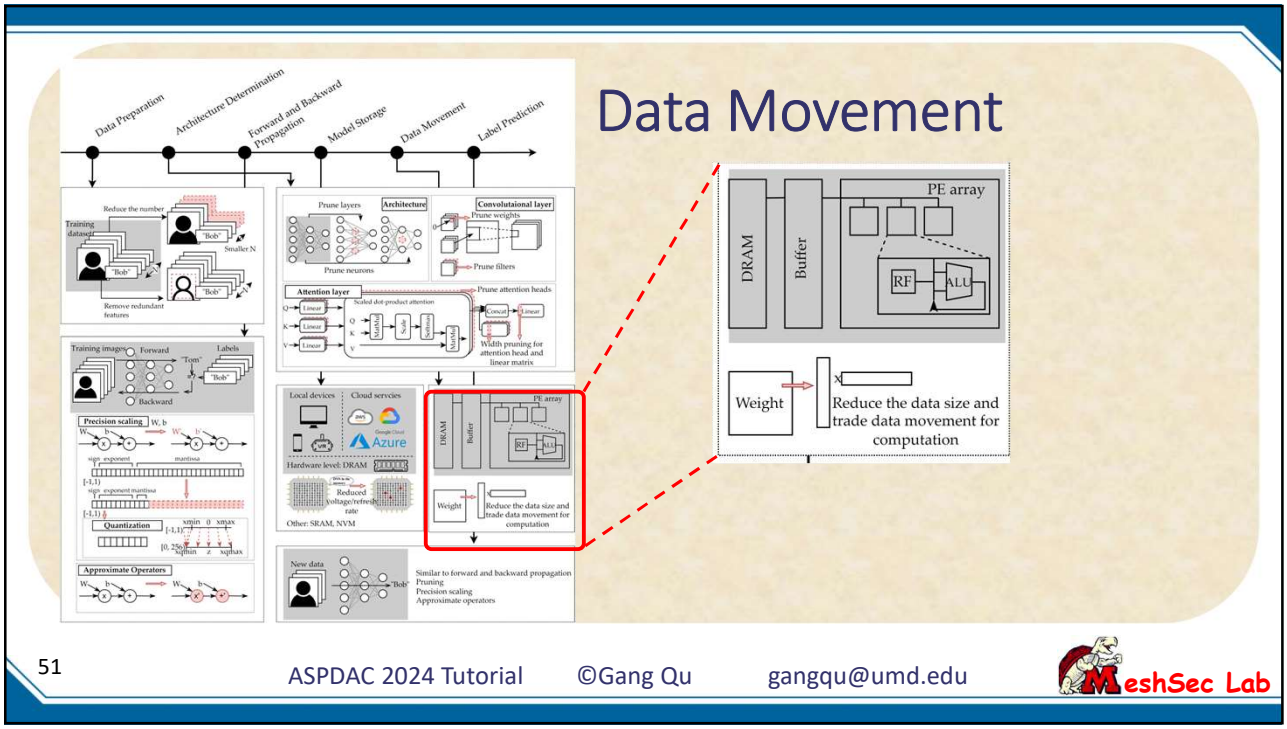
48

F&B Propagation

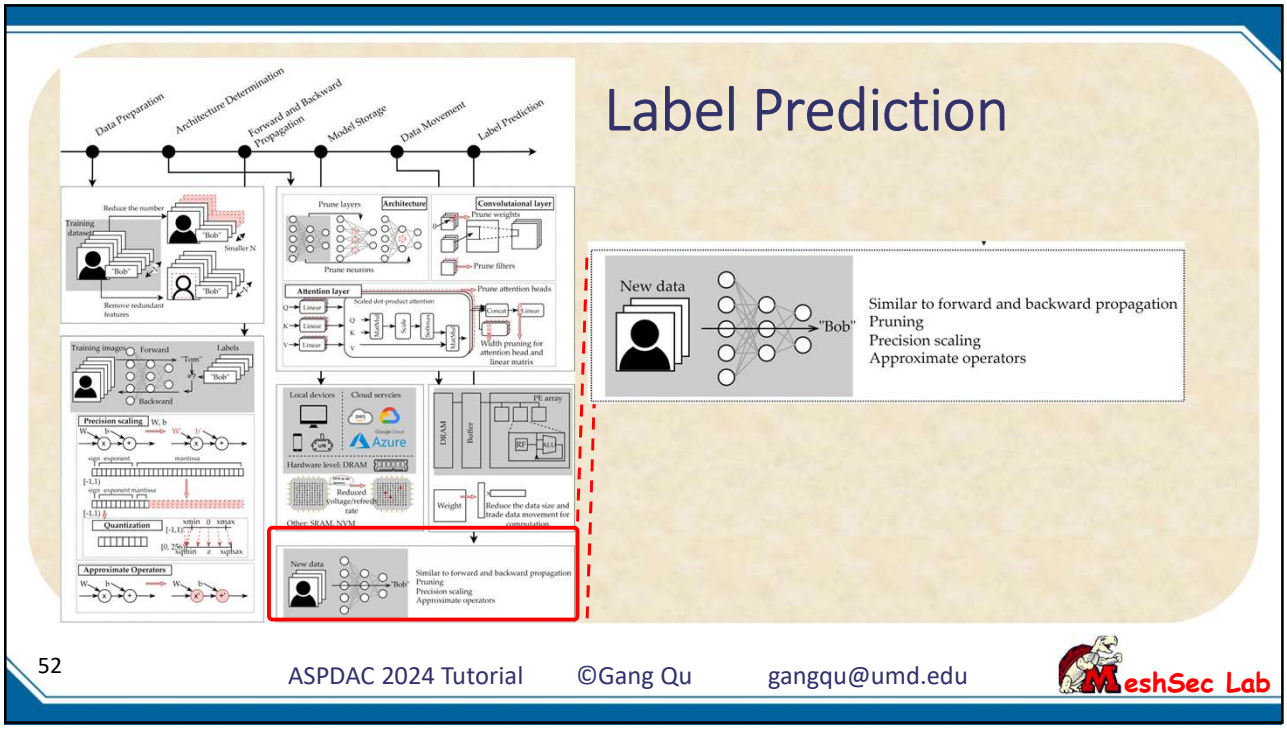


Model Storage




Data Movement
Data Preparation
Architecture Determination
Forward and Backward Propagation
Model Storage
Data Movement
Label Prediction
PE array
DRAM
Buffer
RF
ALU
Weight
Reduce the data size and trade data movement for computation
51
ASPDAC 2024 Tutorial
©Gang Qu
gangqu@umd.edu
MeshSec Lab

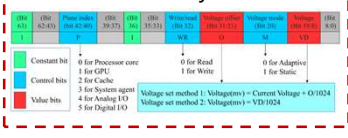
51


Label Prediction
Data Preparation
Architecture Determination
Forward and Backward Propagation
Model Storage
Data Movement
Label Prediction
New data
"Bob"
Similar to forward and backward propagation
Pruning
Precision scaling
Approximate operators
52
ASPDAC 2024 Tutorial
©Gang Qu
gangqu@umd.edu
MeshSec Lab

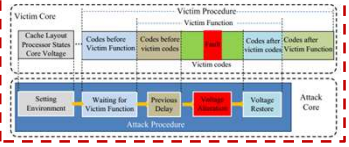
52

VoltJockey Vulnerability

Show low-power technologies have security issues



Prove secure processors are not secure



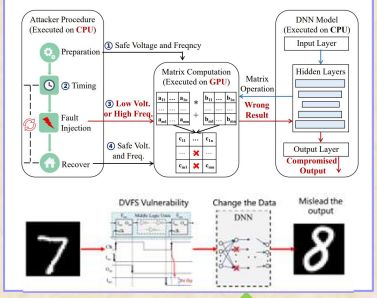
Trigger International Thinking and Research on the Safety of Low-Power Technologies

V0LT PWN

Plunder Volt

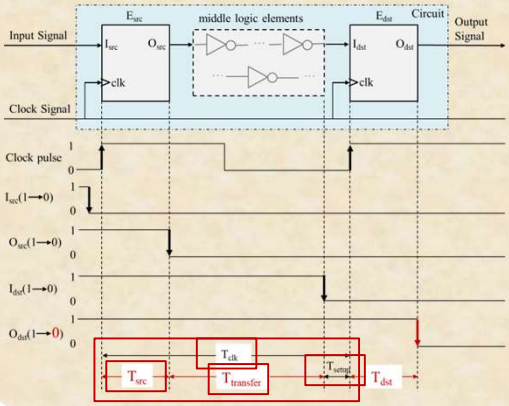
Platy Pus

Lightning Vulnerability



2018 2019 2020 2021 2023

Timing Constraint on IC Design



- T_{clk}: Clock Cycle
- T_{src}: Latency of the first Flip-Flop
- $T_{transfer}$: Transfer time
- T_{setup}: Setup time of the last Flip-Flop

The timing constraint for the IC

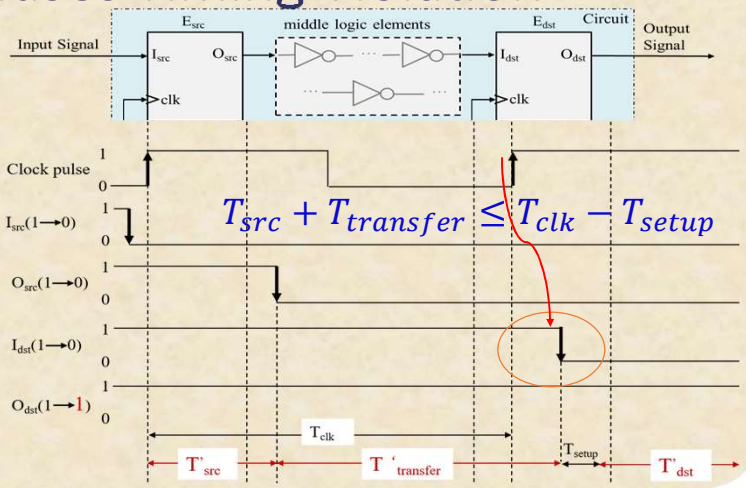
$$T_{src} + T_{transfer} \leq T_{clk} - T_{setup} - T_{\varepsilon}$$

Low Voltage Causes Timing Violation

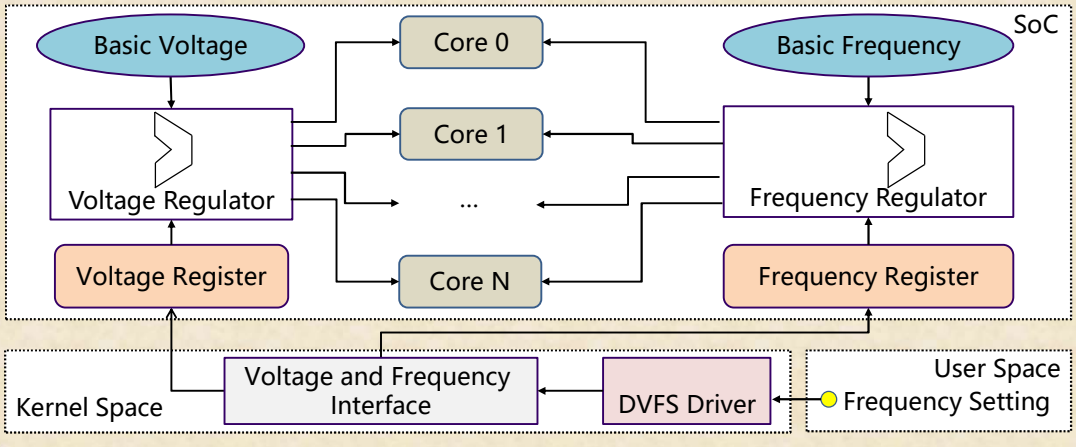
- Gate latency

  $G_t = k(V_t/(V_t - V_r)^2)$
- Reduce voltage
  - T_{src} increases
  - $T_{transfer}$ increases

Reducing voltage may inject hardware fault

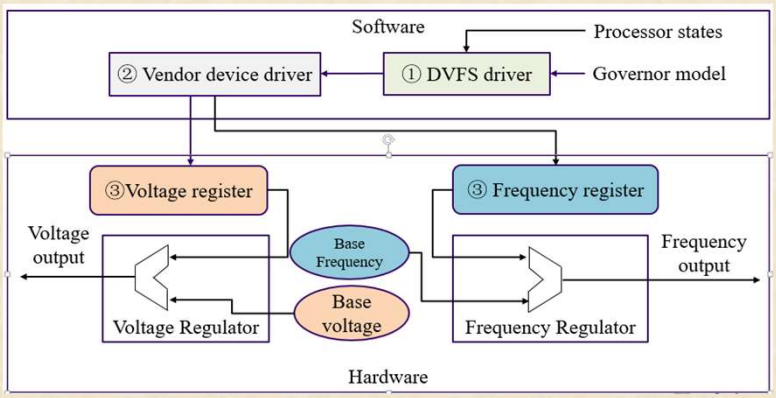


DVFS on Multi-core Processors



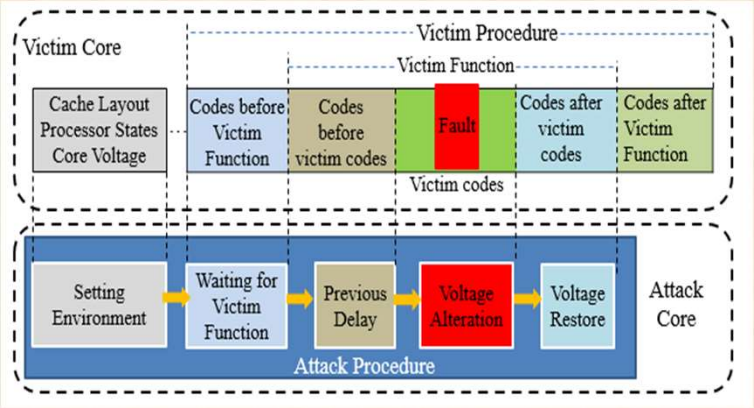
DVFS Working Flow

- DVFS driver selects proper V and F
- Vendor device driver changes V and F registers
- V and F registers alter the regulator outputs



Overview of VoltJockey

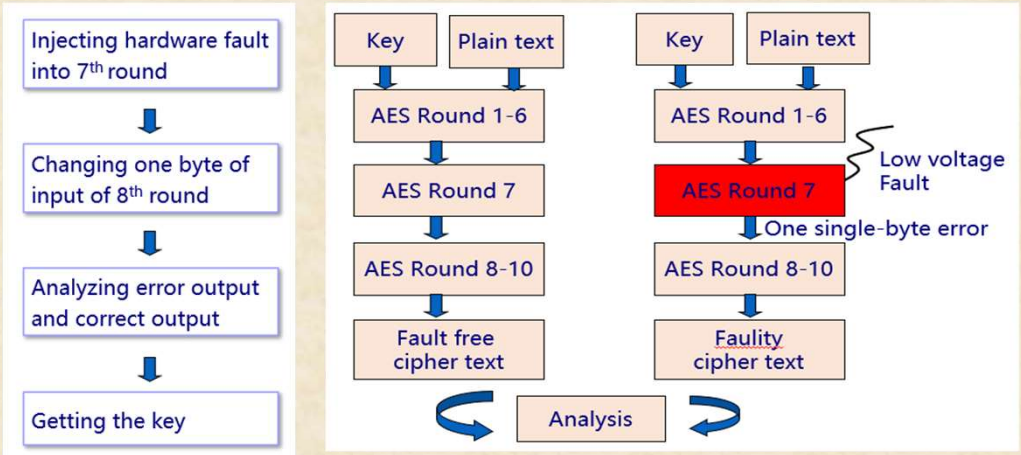
- The attacker procedure and victim procedure are executed on different cores.
- Victim core has a high frequency, but all other cores have a low frequency.



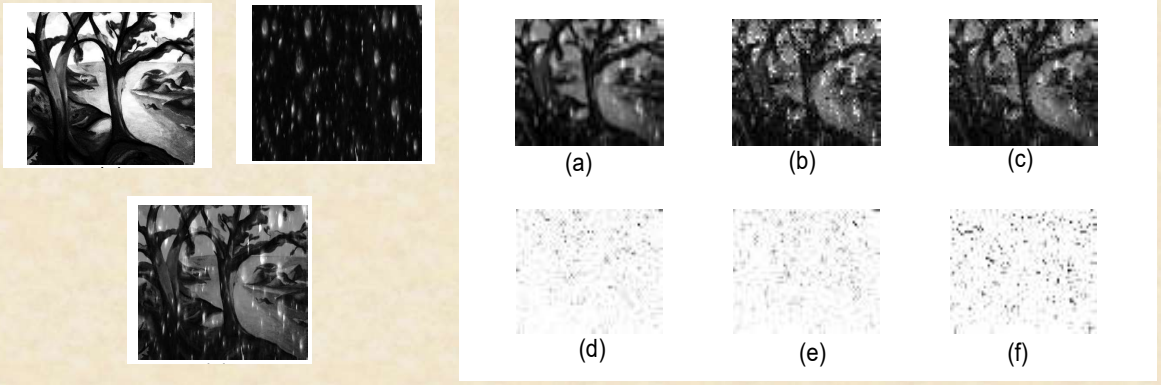
Fault Injection Attacks

- Introducing hardware faults into the execution of cryptographic algorithms and obtaining secret data by analyzing the faulty output
- Traditional fault injection methods: Using special hardware devices
  - Clock Glitch; Voltage Glitch; Electromagnetic Glitch; Optical; Laser
  - Hard to implement remotely

Fault Injection on AES

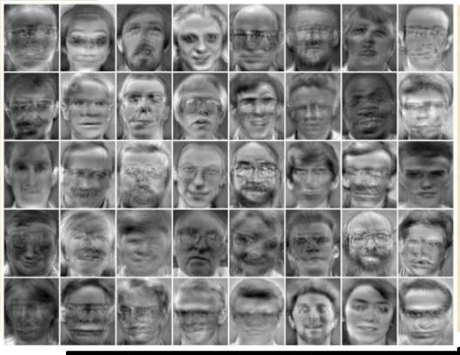


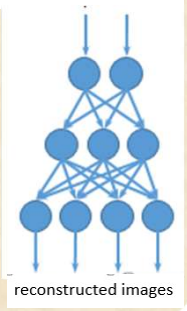
DVFS for Device Authentication



61

Model Inversion Attack



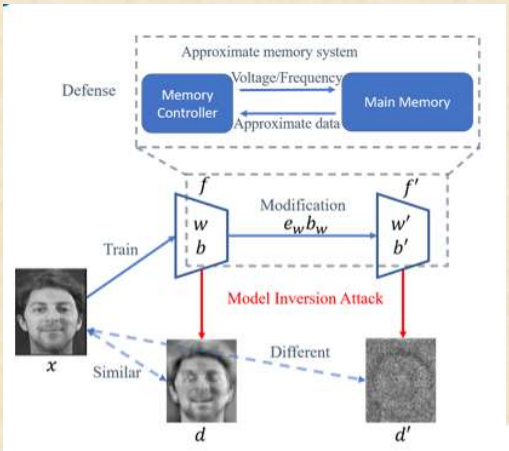


Fredrikson et al, Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures, CCS, 2015.

62

MIDAS Against Model Inversion Attack

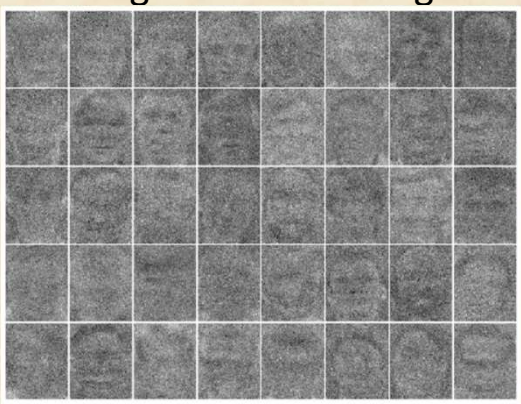
- Use approximate memory system to store model parameters

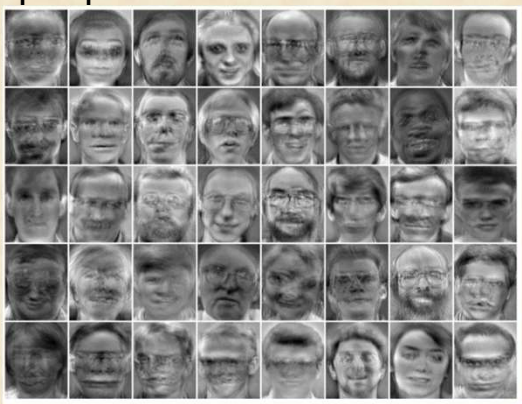


Arafin et al, "MIDAS: Model Inversion Defenses using an Approximate Memory System", AsianHOST'2020.

Protection with MIDAS

- Training data: b&w images of 40 people





Arafin et al, "MIDAS: Model Inversion Defenses using an Approximate Memory System", AsianHOST'2020.

Conclusions

Technology will evolve, but low power will not die

- More applications, devices, greedy human nature → higher power/energy demand
- Other important things: security, privacy, …
- Holistic approach is needed:
  - Circuit, memory, architecture, OS, compiler, application, communication, networking, human, …

Thank you!

Acknowledgments:
Miodrag Potkonjak
Colleagues
Students

References and Readings

- What is Low Power Design? – Techniques, Methodology & Tools | Synopsys https://www.synopsys.com/glossary/what-is-low-power-design.html
- The Ultimate Guide to Power Gating – AnySilicon https://anysilicon.com/power-gating/
- Wong et al. ***Nanoscale CMOS.*** Proc. IEEE, 87, pp. 537-570, 1999.
- Roy et al. ***Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-Submicrometer CMOS Circuits*** *Proceedings of the IEEE*, Vol.91, No.2, pp. 305-327, 2003.
- Pedram. **Power Minimization in IC Design: Principles and Applications**, TODAES, Vol. 1, No. 1, pp. 3-56, 1996.
- Yuan et al., ***FSM Re-Engineering: A Novel Approach to Sequential Circuit Synthesis***, TCAD, Vol. 27, No. 6, pp. 1159-1164, 2008.
- Gu et al. "**Improving Dual Vt Technology by Simultaneous Gate Sizing and Mechanical Stress Optimization**", *ICCAD,* 2011.
- Yuan and Qu, "**A Combined Gate Replacement and Input Vector Control Approach for Leakage Current Reduction**", *TVLSI*, Vol.14, No. 2, pp. 173-182, 2006.
- Qu, "**What Is the Limit of Energy Saving by Dynamic Voltage Scaling**", ICCAD, pp. 560-563, 2001 .
- Hua and Qu. "**Approaching the Maximum Energy Saving on Embedded Systems with Multiple Voltages**", ICCAD, 2003.
- Hua et al. "**Energy Reduction Techniques for Multimedia Applications with Tolerance to Deadline Misses**", DAC, 2003.
- Mahdiani et al, "**Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications**," TCAS-I., vol. 57, no. 4, pp. 850–862, 2010.
- Kahng and Kang. "**Accuracy-configurable adder for approximate arithmetic designs**". *DAC 2012.*
- Gao et al, **"A Novel Data Format for Approximate Arithmetic Computing"**, *ASPDAC* 2017.

DVFS on Security

- CLKSCREW: Exposing the Perils of Security-Oblivious Energy Management, Usenix Security **2017**
- VOLtA: Voltage over-scaling based lightweight authentication for IoT applications, ASPDAC **2017**
- VoltJockey: Breaching TrustZone by software-controlled voltage manipulation over multi-core frequencies, CCS, **2019**
- VoltJockey: Breaking SGX by software-controlled voltage-induced hardware faults, AsianHOST **2019**
- MIDAS: Model Inversion Defenses using an Approximate Memory System", AsianHOST **2020**, TETC **2022**
- Plundervolt: How a Little Bit of Undervolting Can Create a Lot of Trouble, S&P **2020**
- V0LTpwn: Attacking x86 Processor Integrity from Software, Usenix Security **2020**
- Lightning: Striking the Secure Isolation on GPU Clouds with Transient Hardware Faults", arXiv **2021**.
- DVFSspy: Using Dynamic Voltage and Frequency Scaling as a Covert Channel for Multiple Procedures, ASPDAC **2022**