Precomputation-based Guarding for Dynamic and Leakage Power Reduction

Afshin Abdollahi, Massoud Pedram
University of Southern California

Farzan Fallah, Indradeep Ghosh
Fujitsu Labs of America

The 21st International Conference on Computer Design
October 13-15, 2003
San Jose, California

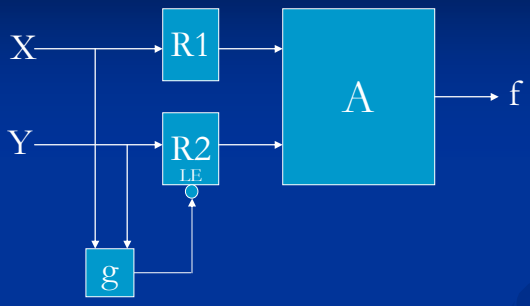
Outline

- Introduction
- Power reduction techniques
  - Power Supply/Ground Gating
  - Clock Gating
    - Guarded Evaluation
    - Precomputation
  - Operand isolation
- Precomputation-based Guarding
- Results
- Conclusions and Future Work

Introduction

- Sources of Power Consumption
  - Switching Power
  - Leakage Currents
  - Short Circuit Currents
- Power Reduction Techniques
  - Clock Gating (for Switching Power)
  - Power Supply Gating (for Leakage Current)

Precomputation



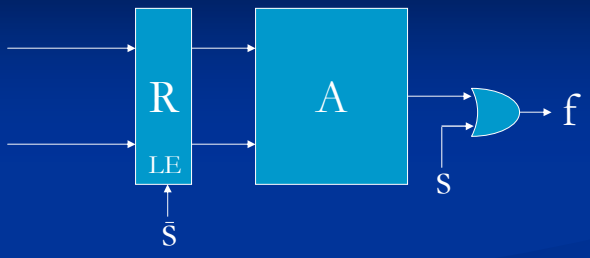
Basic idea:

Freeze some of the inputs when a specific condition on input values holds.

If f is independent of Y, then freeze Y.

Goals: Minimize size of g, maximize |Y|, and maximize the likelihood of the condition happening.

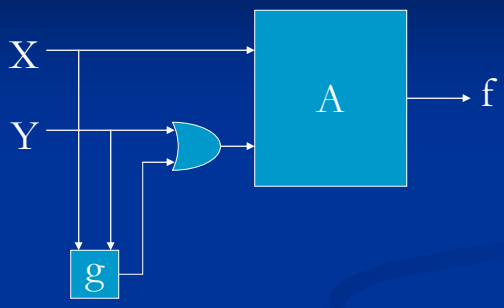
Guarded Evaluation



$(s=1) \Rightarrow (f=1)$

- Disabling gates performing redundant computation.
- Selecting an existing signal s instead of generating a new signal.

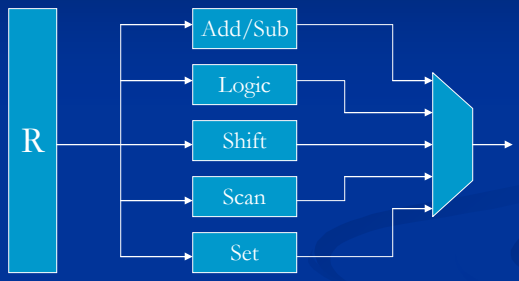
Operand Isolation



If f is independent of Y, then set g to 1.

- Can freeze Y for several consecutive cycles.
- Alternatively, an AND gate may be used.

Input Sharing Problem

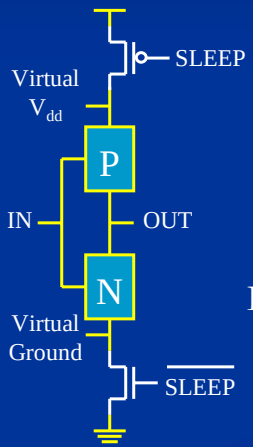


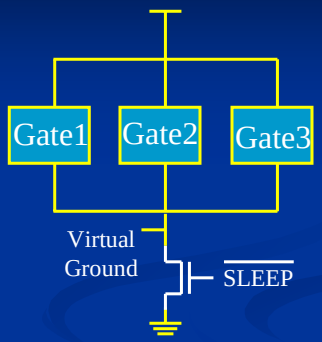
Integer unit of FR500, a commercial VLIW processor

All modules are driven by the same set of registers.

Power Supply/Ground Gating

- Low Threshold
- High Threshold



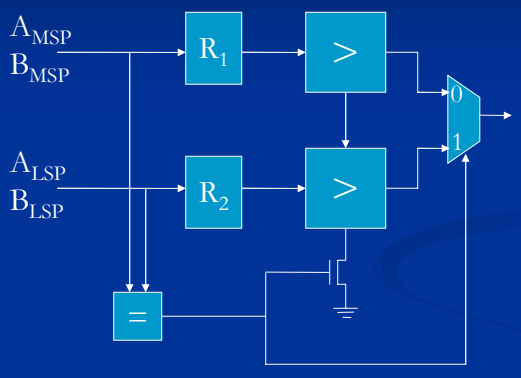


Reducing leakage and switching power

Our Approach

- Combines precomputation and ground gating
- Reduces both switching and leakage power
- Solves the input sharing problem
- Can disable registers as well if input sharing problem does not exist.

Comparators



If $A_{MSP} > B_{MSP}$, then A>B
If $A_{MSP} < B_{MSP}$, then A<B

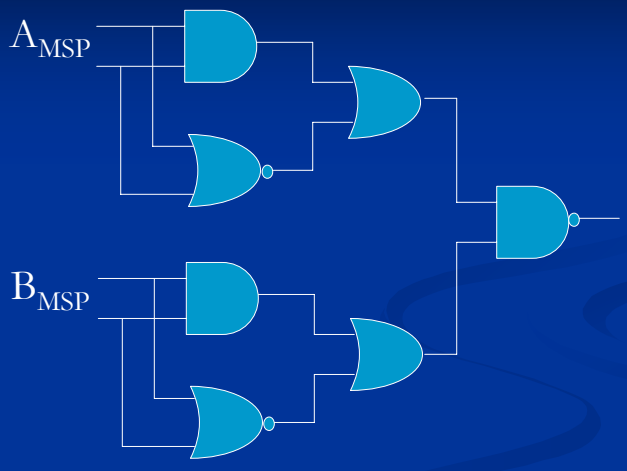
A = 1001,1010 0101,0100
B = 0001,1010 1101,0100

Adders

- Partition the adder into MSP and LSP.
- The goal is to disable the MSP when it is not necessary to operate.
- If the sign extension part of operands exceed the MSP range, the MSP is disabled.

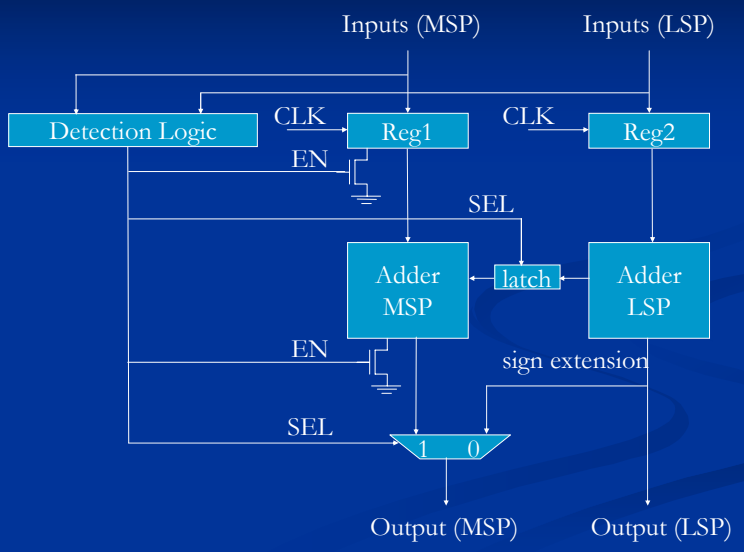
	MSP	LSP
A	= 1111,1111	1101,0011
B	= 0000,0000	0011,0110
Sum	= 0000,0000	0000,1001

Detection Logic

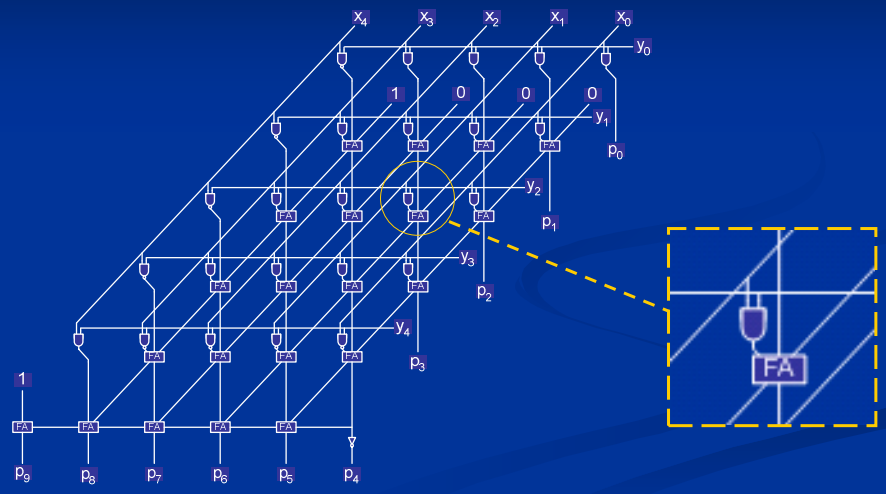


Output is low if MSPs have sign information only.

Adders

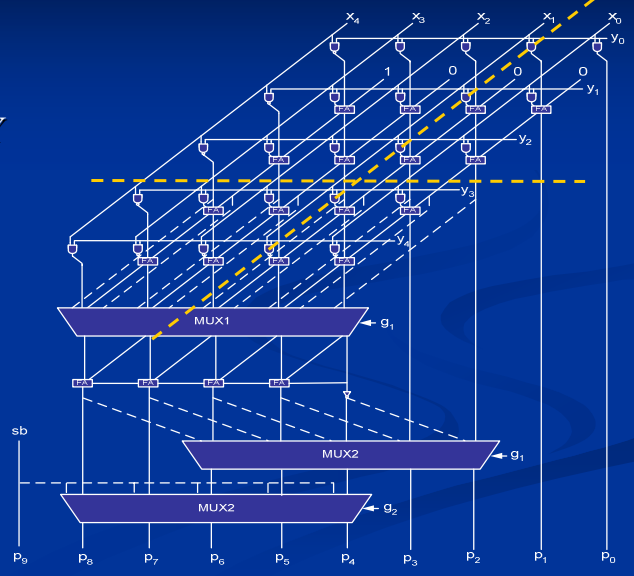


Multipliers



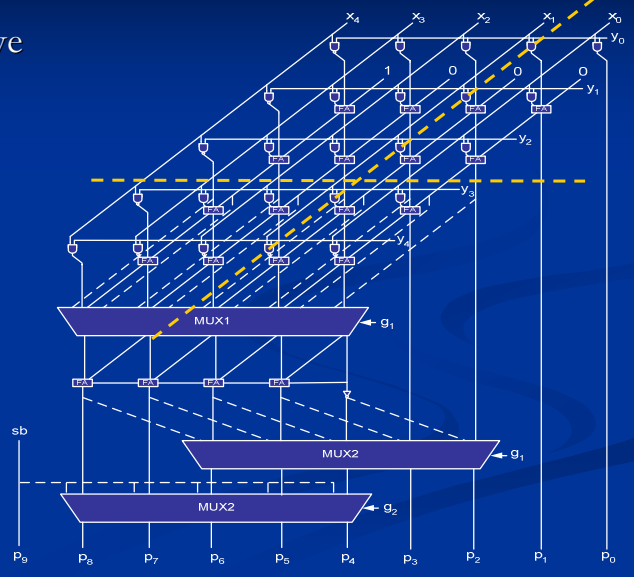
Two Dimensional Guarding

Detection Logic Generates two signals for X and Y directions.



Two Dimensional Guarding

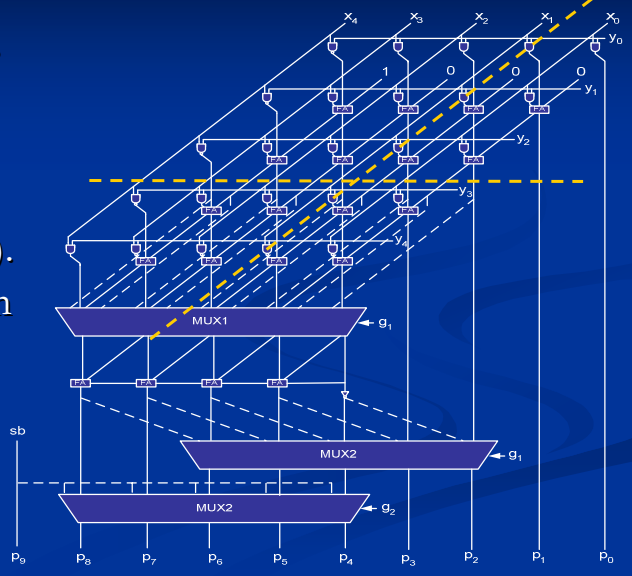
The CSA (carry save adder) array is partitioned to four regions.
One region (right upper) is always active.



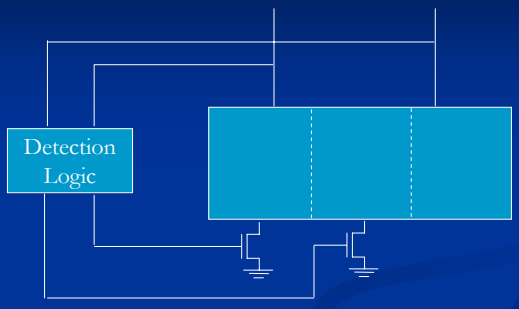
Two Dimensional Guarding

Sign extension is easy to generate (XOR of the most significant bits of operands).

sb = An ⊕ Bn

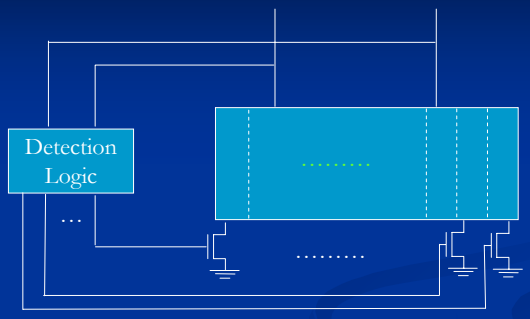


Hybrid Guarding



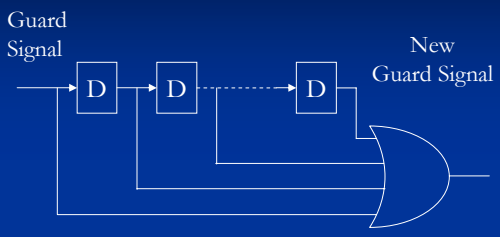
- Using two or more thresholds for sign extension length.
- Select the threshold based on input data

Dynamic Guarding

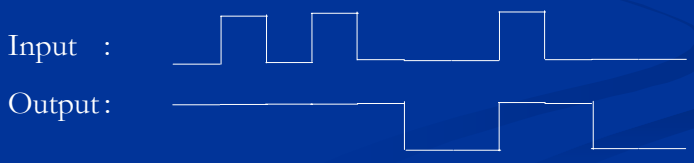


- Dynamically detect the sign extension length based on input data.
- Disable the corresponding part of the circuit.

Reducing the Switching Activity



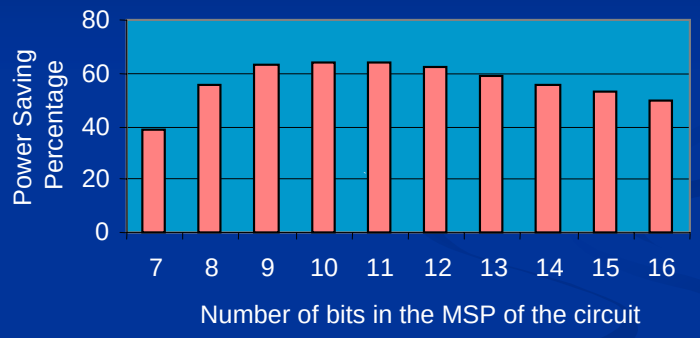
Disabling the circuit only if Guard Signal has been active several cycles.



Results

- 32-bit functional units
- Process
  - 70nm technology
  - Supply voltage = 0.9V
  - NMOS transistors' threshold voltage = 0.2V
  - PMOS transistors' threshold voltage = -0.22V
  - Sleep transistors' threshold voltage = 0.5V
- Simulation Tools
  - PowerMill to estimate the power in transistor-level
  - SPICE to estimate the delay of circuits
- Test Bench
  - Trace of 1000 vectors corresponding to ALU unit in the data-path of a processor executing a JPEG decoder program

Results (Comparator)



Results (Adder)

Method	Regs	Adder	Total Saving
12-bit signal-gating	32%	25%	26%
12-bit operand-isolation	0%	21%	17%
12-bit guarding	44%	25%	38%
14-bit signal-gating	34%	35%	27%
14-bit operand-isolation	0%	28%	15%
14-bit guarding	50%	28%	41%
16-bit signal-gating	35%	45%	27%
16-bit operand-isolation	0%	35%	15%
16-bit guarding	55%	52%	48%
18-bit signal-gating	8%	49%	1%
18-bit operand-isolation	0%	36%	2%
18-bit guarding	12%	23%	3%

Results (Multiplier)

Method	Regs	Multiplier	Total Saving
22_bit_input1 Gating 16_bit_input2 Gating	41%	58%	49%
22_bit_input1 Guarding 16_bit_input2 Guarding	43%	60%	52%
22_bit_input1 Guarding input2_dynamic	49%	67%	56%
22 &16_bit_input1 Hybrid Guarding input2_dynamic Guarding	51%	71%	60%

Delay and Area Overheads

Circuit	Guarding Method	Delay Overhead	Area Overhead
Comparator	10-bit Guarding	25%	30%
Adder	18-bit Guarding (reduced switching activity)	15%	10%
Multiplier	22&16_bit_input1 hybrid guarding input2_dynamic guarding	9%	6%

Results (FR500 VLIW processor)

- Technology: 0.18um
- Supply voltage: 1.8V
- Used an instruction set simulator to generate a trace.
- Used the precomputation-based guarding method for the ADD/SUB module.
- Used full guarding for other modules to disconnect them from the ground when they were not used to execute any instruction.

Results (FR500 VLIW processor)

- Power reduction = 81%
- Area overhead = 9%
- Delay overhead = 12%
- Operand isolation:
  - Power reduction = 58%
  - Area overhead = 11%
- Precomputation combined with operand isolation:
  - Power reduction = 61%
  - Area overhead = 14%

Conclusions and Future Work

- Conclusions
  - Combining precomputation and power supply gating
  - Reducing switching and leakage power
  - Solving the input sharing problem or disabling the registers when the input sharing problem does not exist
  - At least 20% higher power saving in compare to existing methods
- Future work
  - Developing hierarchical precomputation architecture
  - Developing an algorithm for performing near optimum dynamic guarding for every circuit