



NII



# Ultra Fine-Grained Run-Time Power Gating of On-Chip Routers for CMPs

Hiroki Matsutani (Univ Tokyo, Japan)

Michihiro Koibuchi (NII, Japan)

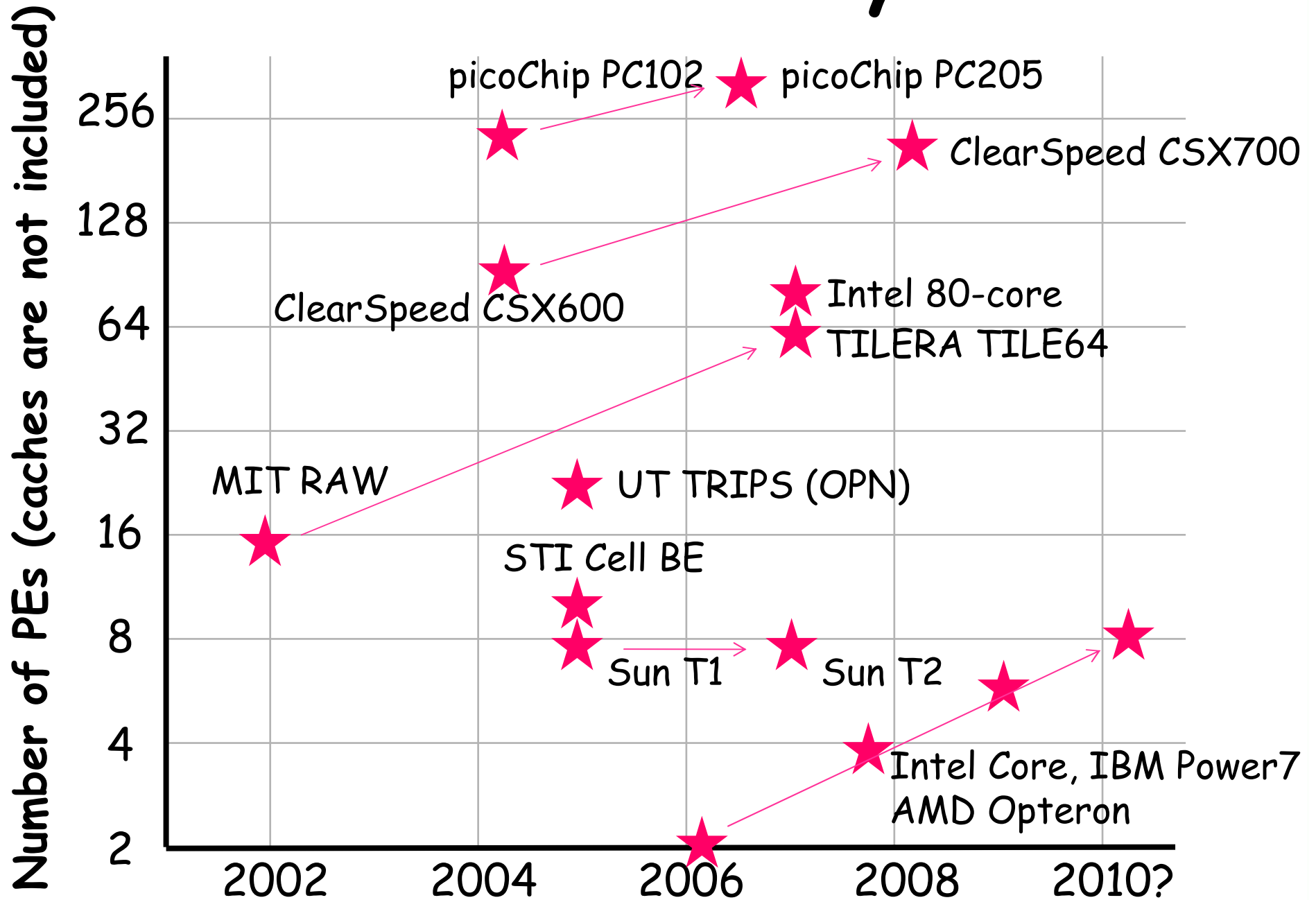
Daisuke Ikebuchi (Keio Univ, Japan)

Kimiyoshi Usami (Shibaura IT, Japan)

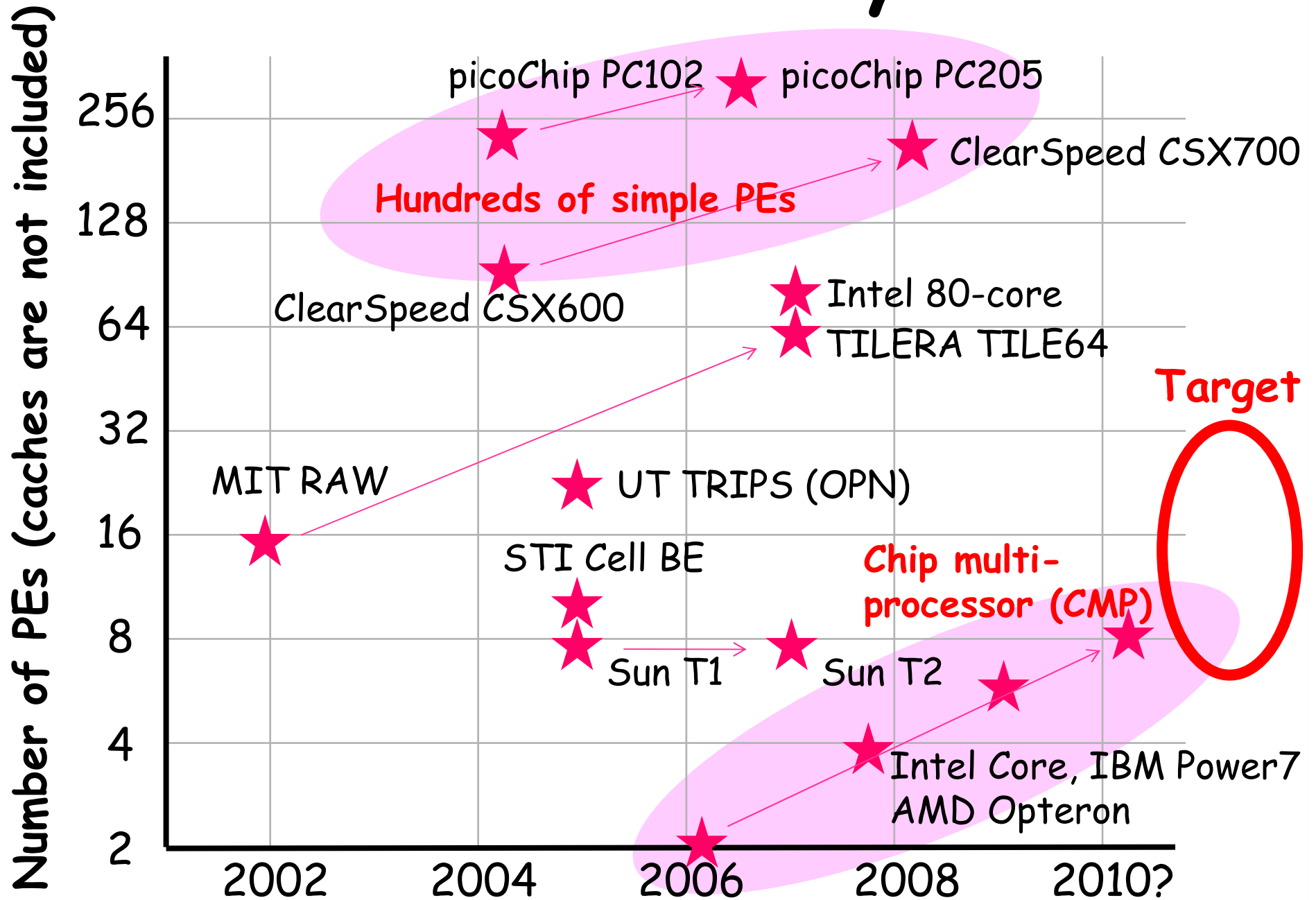
Hiroshi Nakamura (Univ Tokyo, Japan)

Hideharu Amano (Keio Univ, Japan)

# Multi-Core & Many-Core

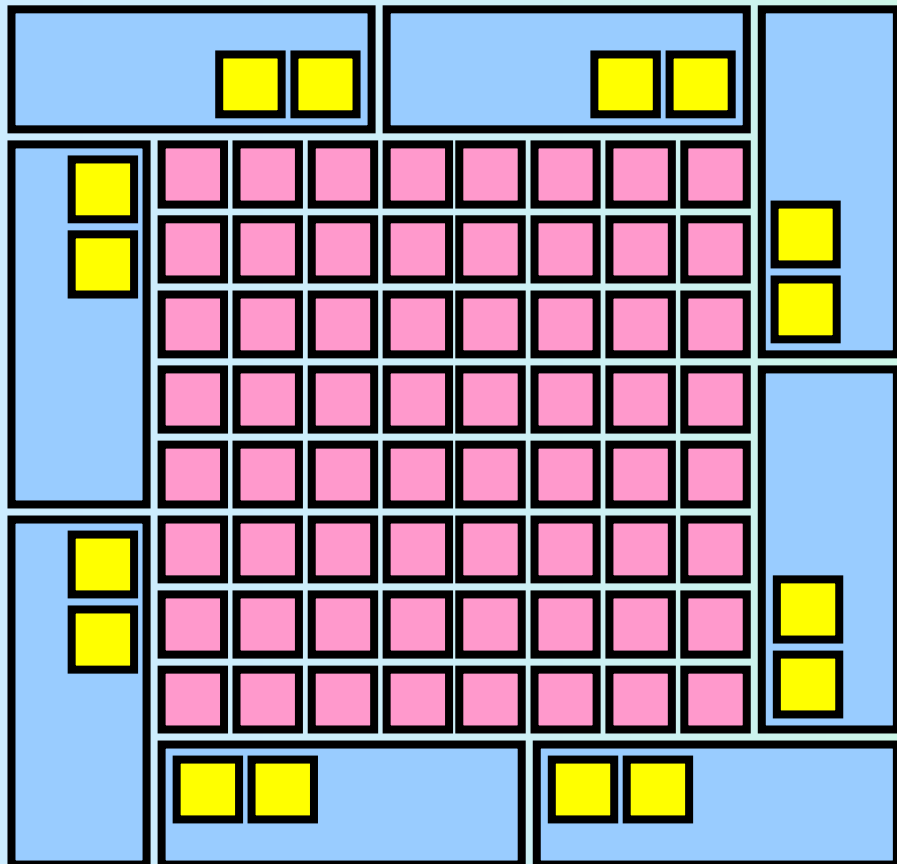


# Multi-Core & Many-Core

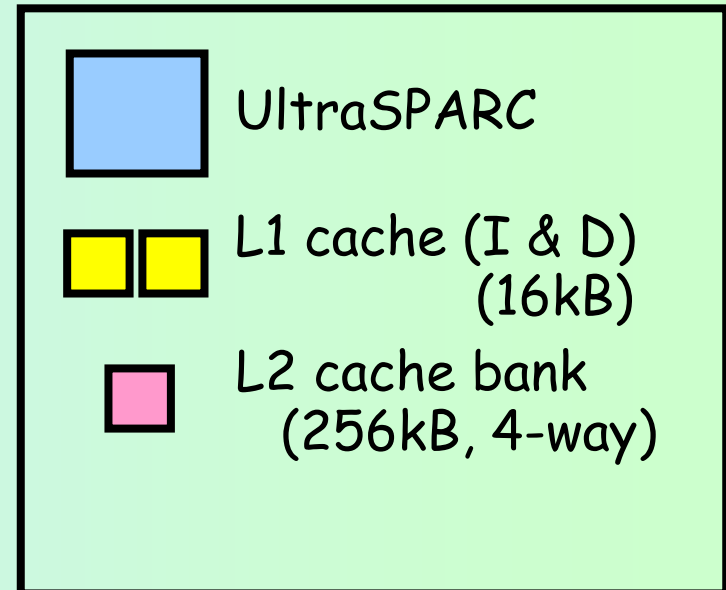


# Our target: NoC for future CMPs

- 8-CPU CMP example
  - 8 CPUs (each has a private L1 cache)
  - Shared L2 cache (divided into 64 banks)



[Beckmann, MICRO'04]



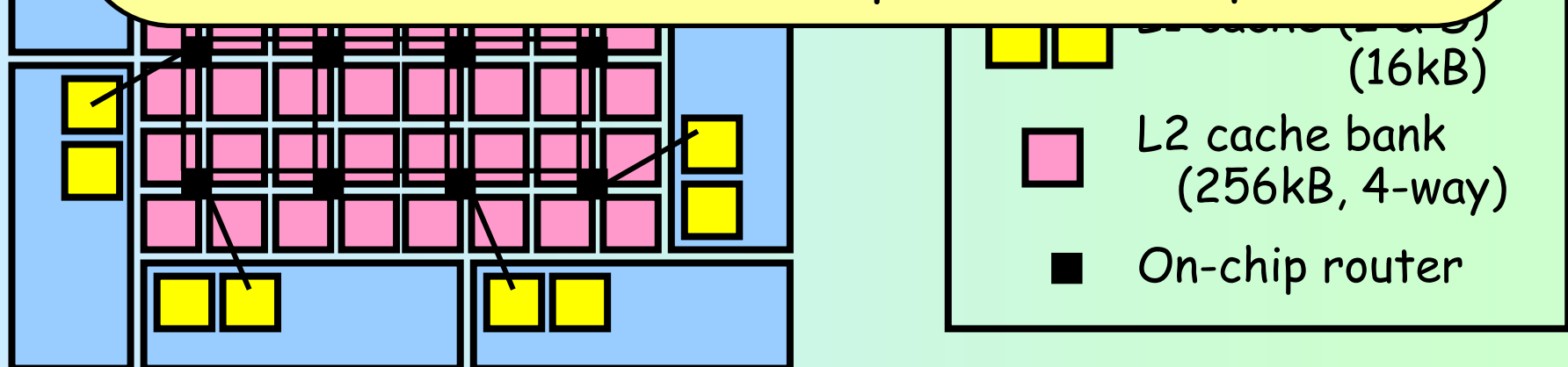
# Our target: NoC for future CMPs

- 8-CPU CMP example
  - 8 CPUs (each has a private L1 cache)

On-chip network is an infrastructure of CMPs.  
It must be always ready for the packet transfers.

→ It consumes leakage power at any time!

Run-time power gating that stops the power supply to the routers whenever possible is required.



# Outline: Fine-grain power gating router

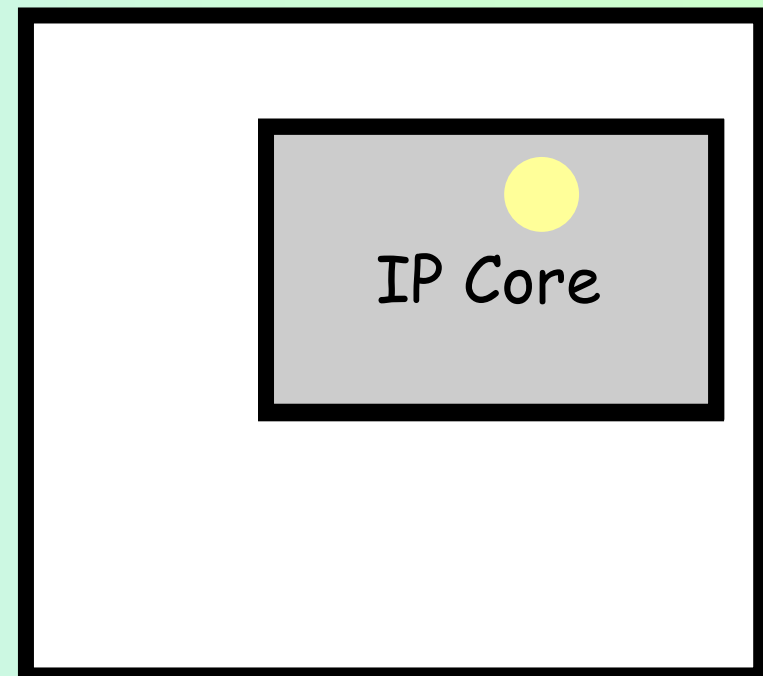
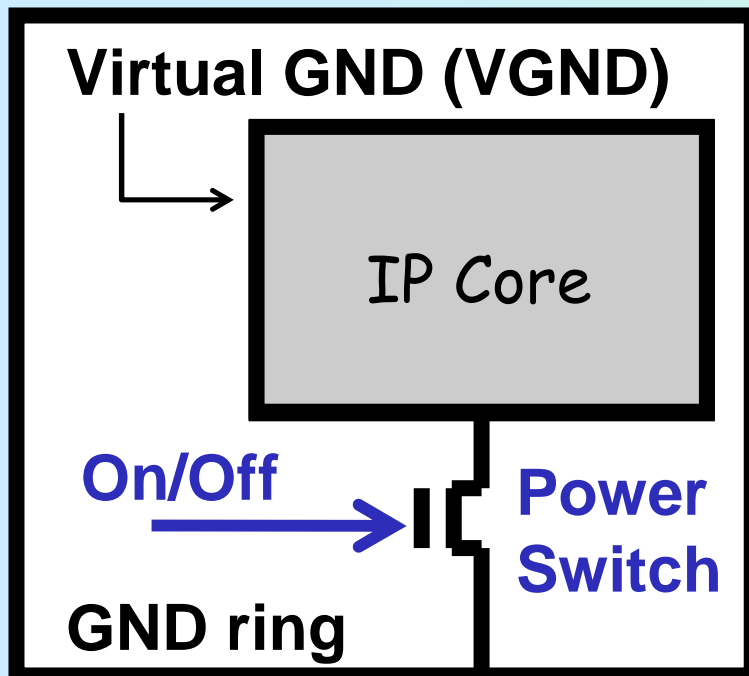
- Fine-grained power gating router
  - Input VC buffers
  - Crossbar MUXes, VC MUXes
  - Output latches

} 35 power domains in each router
- Power domain implementation @ 65nm
  - Design flow
  - Wakeup latency estimation and its impact
- Three early wakeup methods
- Evaluation results
  - Application performance w/ early wakeup
  - Leakage power reduction

# Power gating: Coarse- vs. fine-grain

- Coarse-grain approach
  - IP core (module) level
  - Surrounded by VGND
  - Power switch between VGND and GND
- Fine-grain approach
  - Standard cell level
  - Each cell has VGND port
  - All cells in a domain share the same VGND line

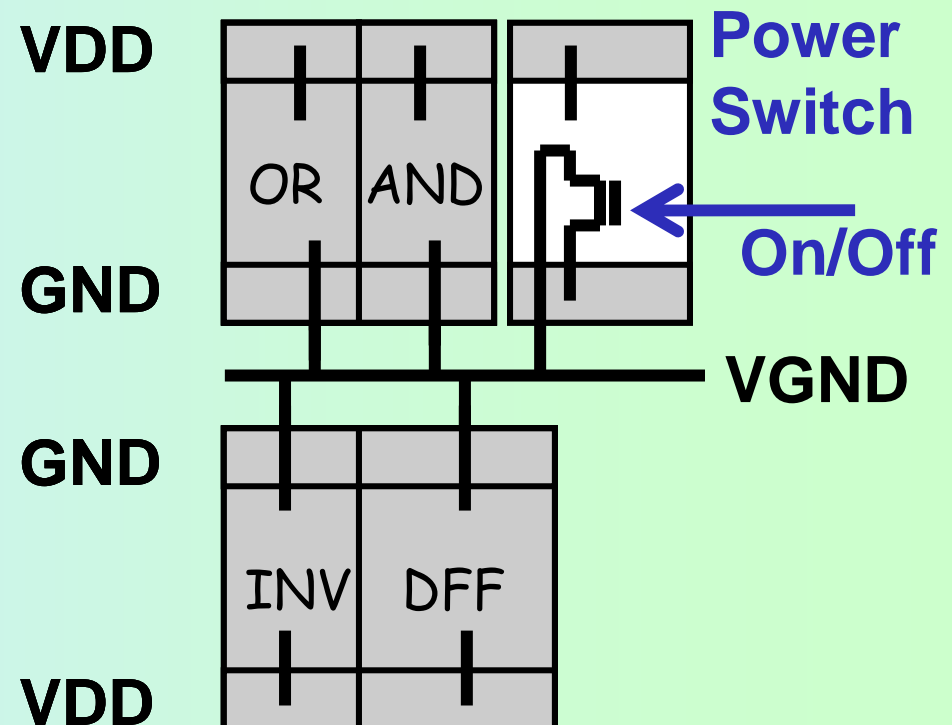
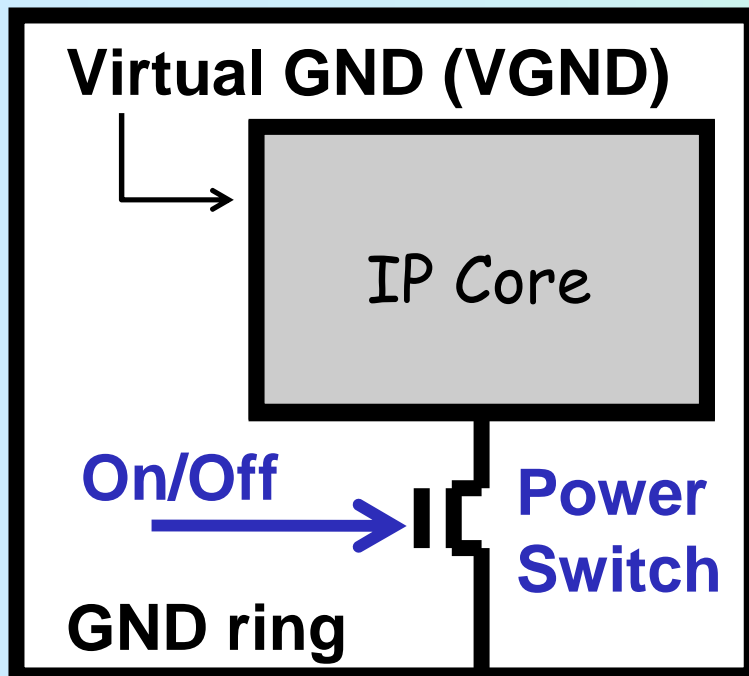
[Usami, ICCD'06]



# Power gating: Coarse- vs. fine-grain

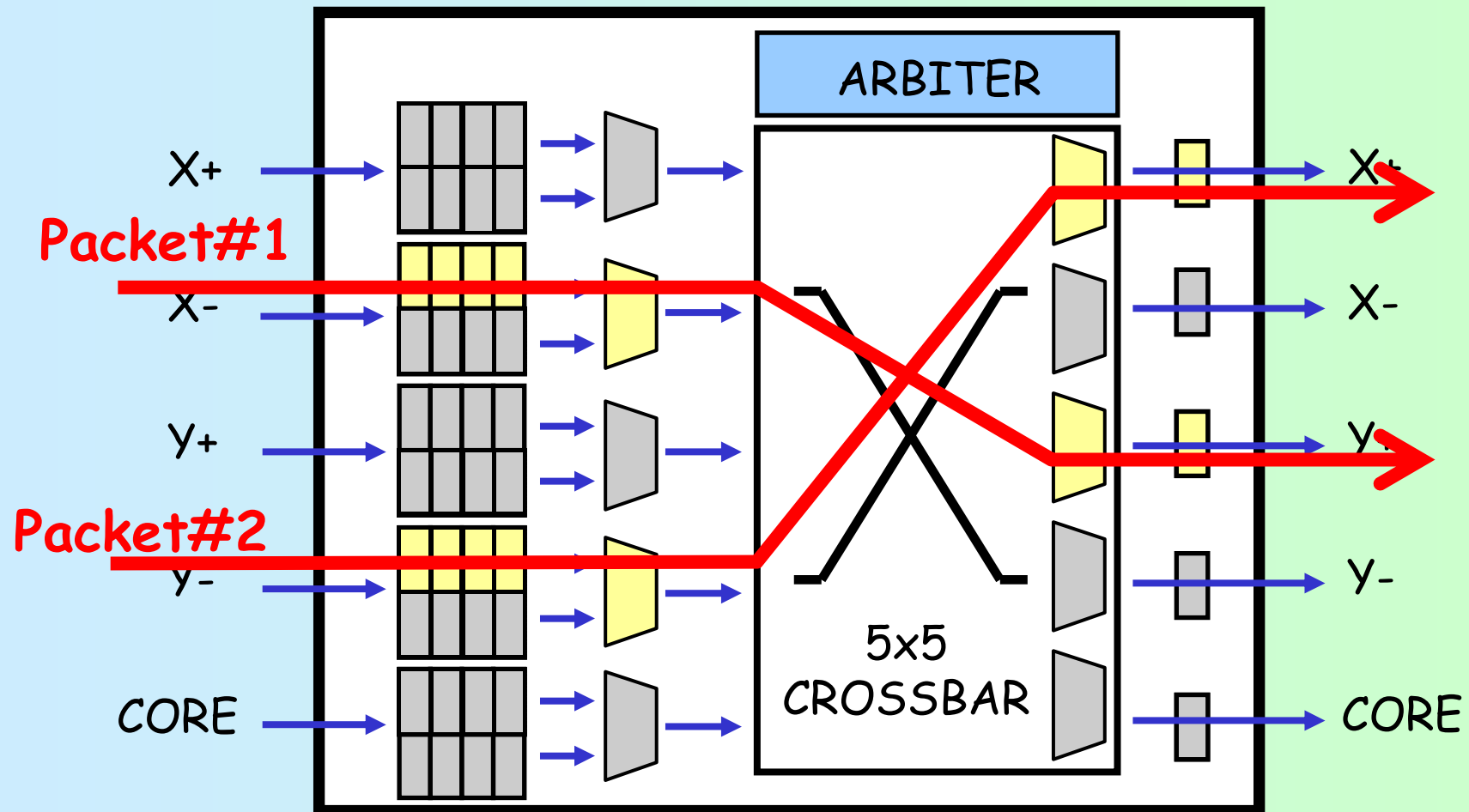
- Coarse-grain approach
  - IP core (module) level
  - Surrounded by VGND
  - Power switch between VGND and GND
- Fine-grain approach
  - Standard cell level
  - Each cell has VGND port
  - All cells in a domain share the same VGND line

[Usami, ICCD'06]



# Which is better? Coarse or Fine

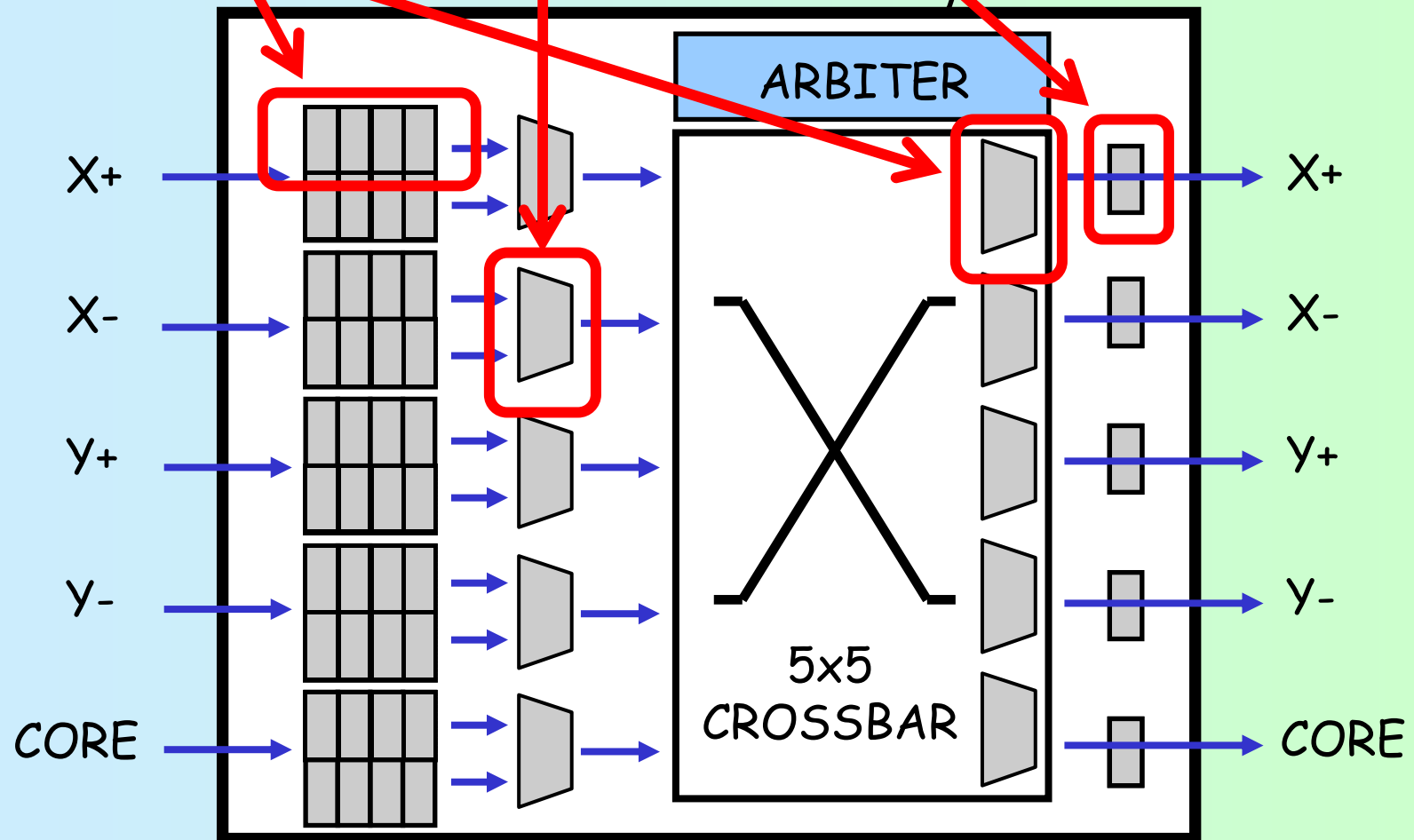
- Each router component (e.g., input port) works independently each other
  - Fine-grain approach has more opportunities to sleep



# Fine-grain run-time PG router

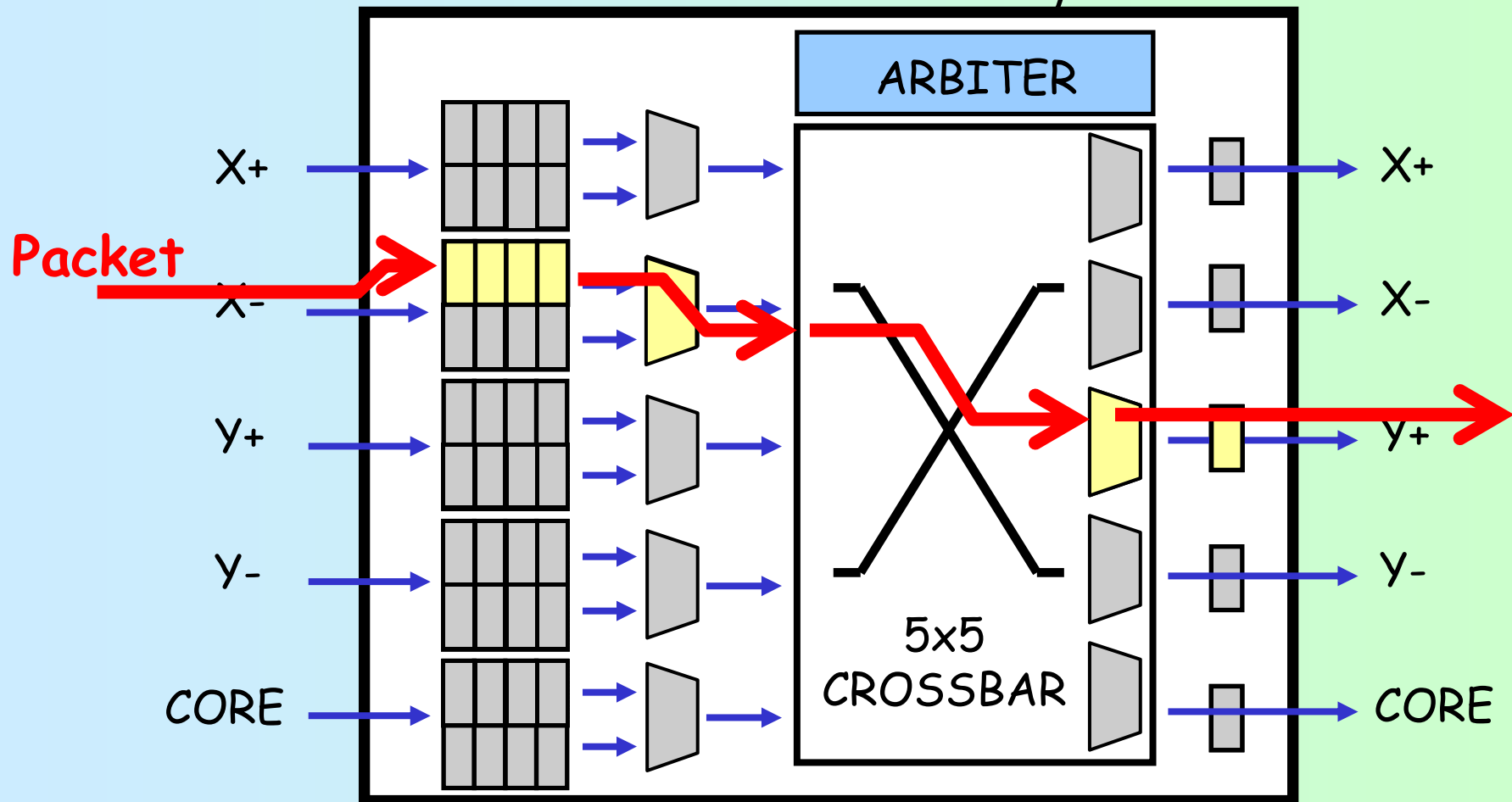
- Router is divided into many Micro-power-domains
  - Input VC buffers, Output latches
  - Crossbar MUXes, VC MUXes

35 power domains  
in a 5-port router



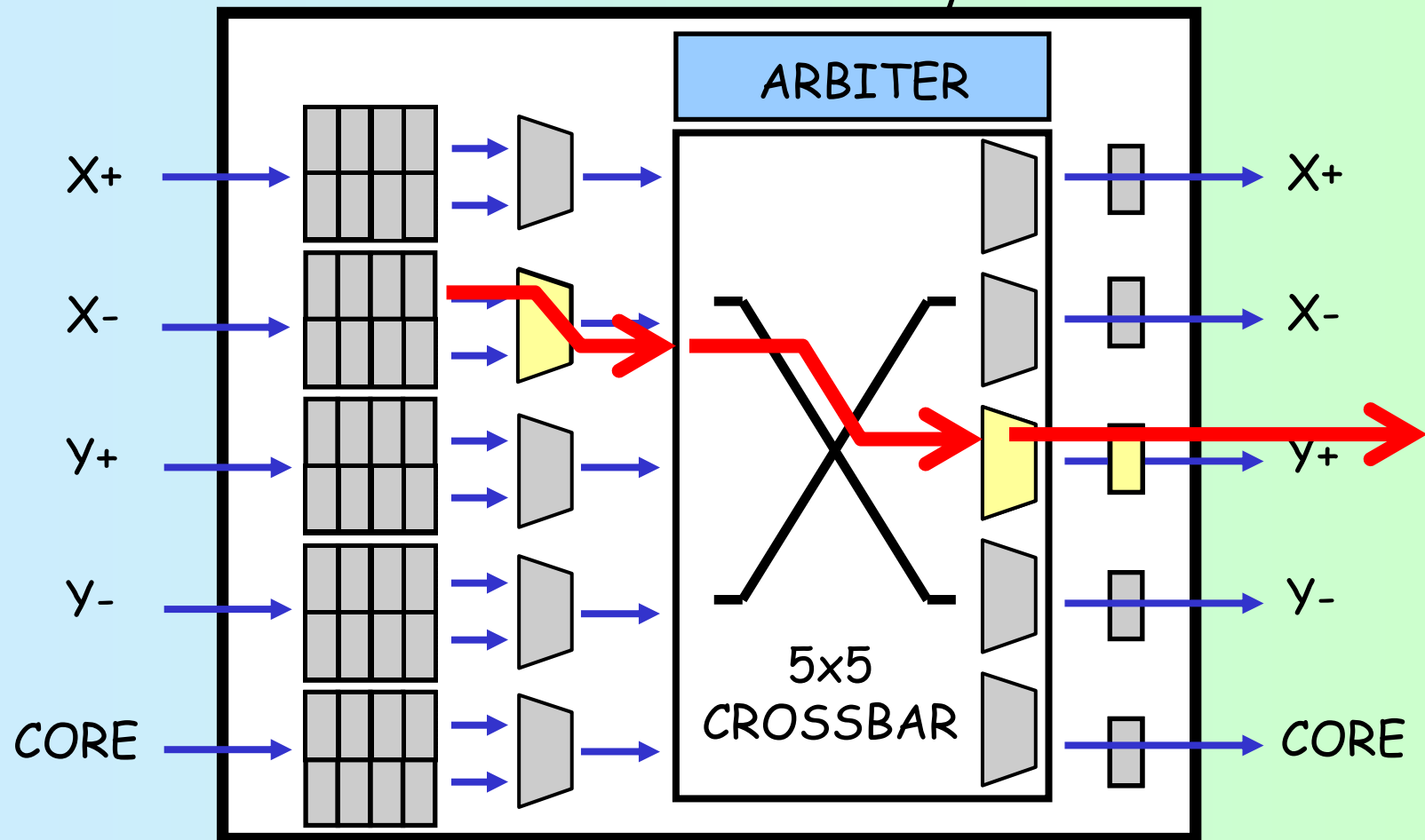
# Fine-grain run-time PG router

- Router is divided into many Micro-power-domains
    - Input VC buffers, Output latches
    - Crossbar MUXes, VC MUXes
- 35 power domains in a 5-port router



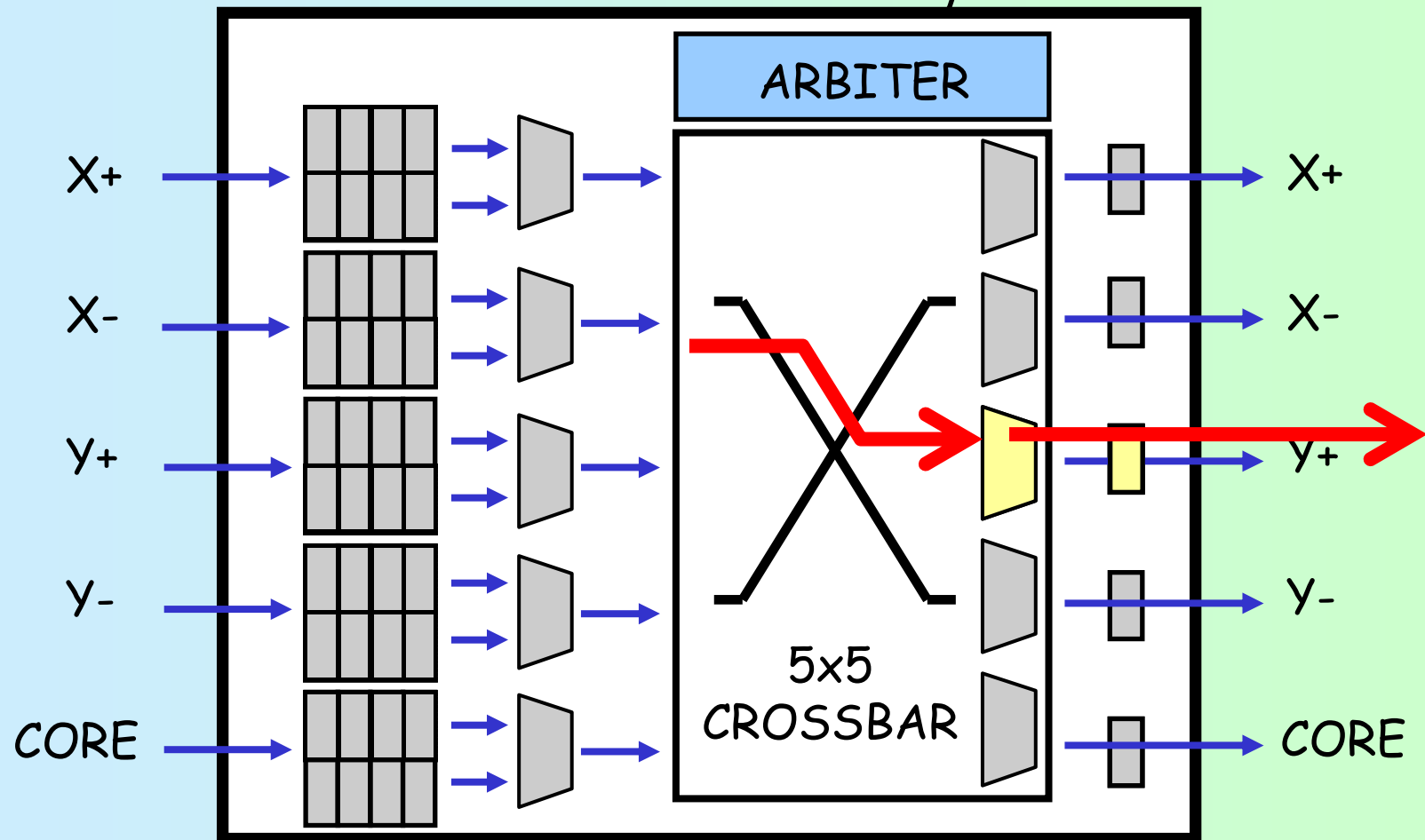
# Fine-grain run-time PG router

- Router is divided into many Micro-power-domains
    - Input VC buffers, Output latches
    - Crossbar MUXes, VC MUXes
- } 35 power domains in a 5-port router



# Fine-grain run-time PG router

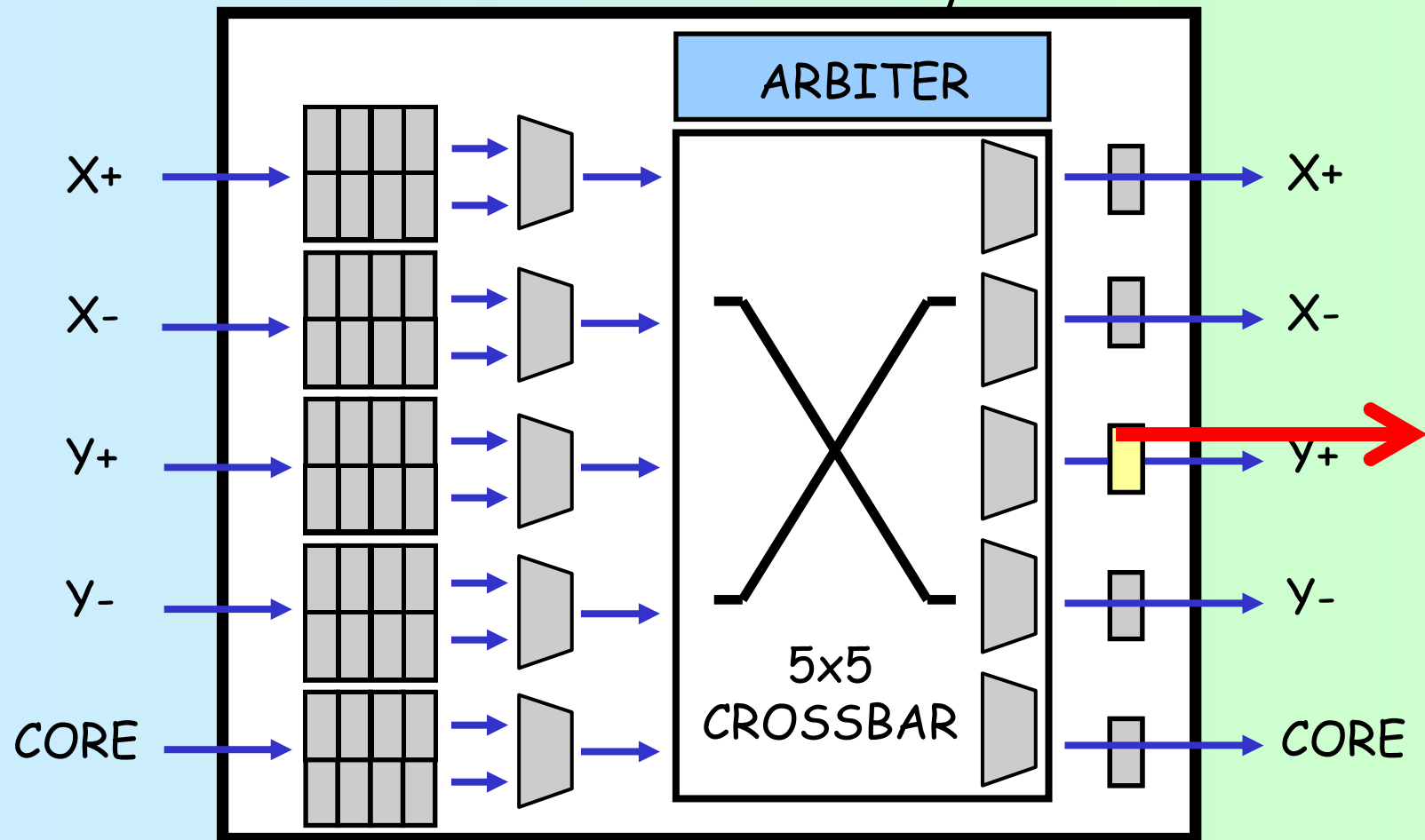
- Router is divided into many Micro-power-domains
    - Input VC buffers, Output latches
    - Crossbar MUXes, VC MUXes
- 35 power domains in a 5-port router



# Fine-grain run-time PG router

- Router is divided into many Micro-power-domains
  - Input VC buffers, Output latches
  - Crossbar MUXes, VC MUXes

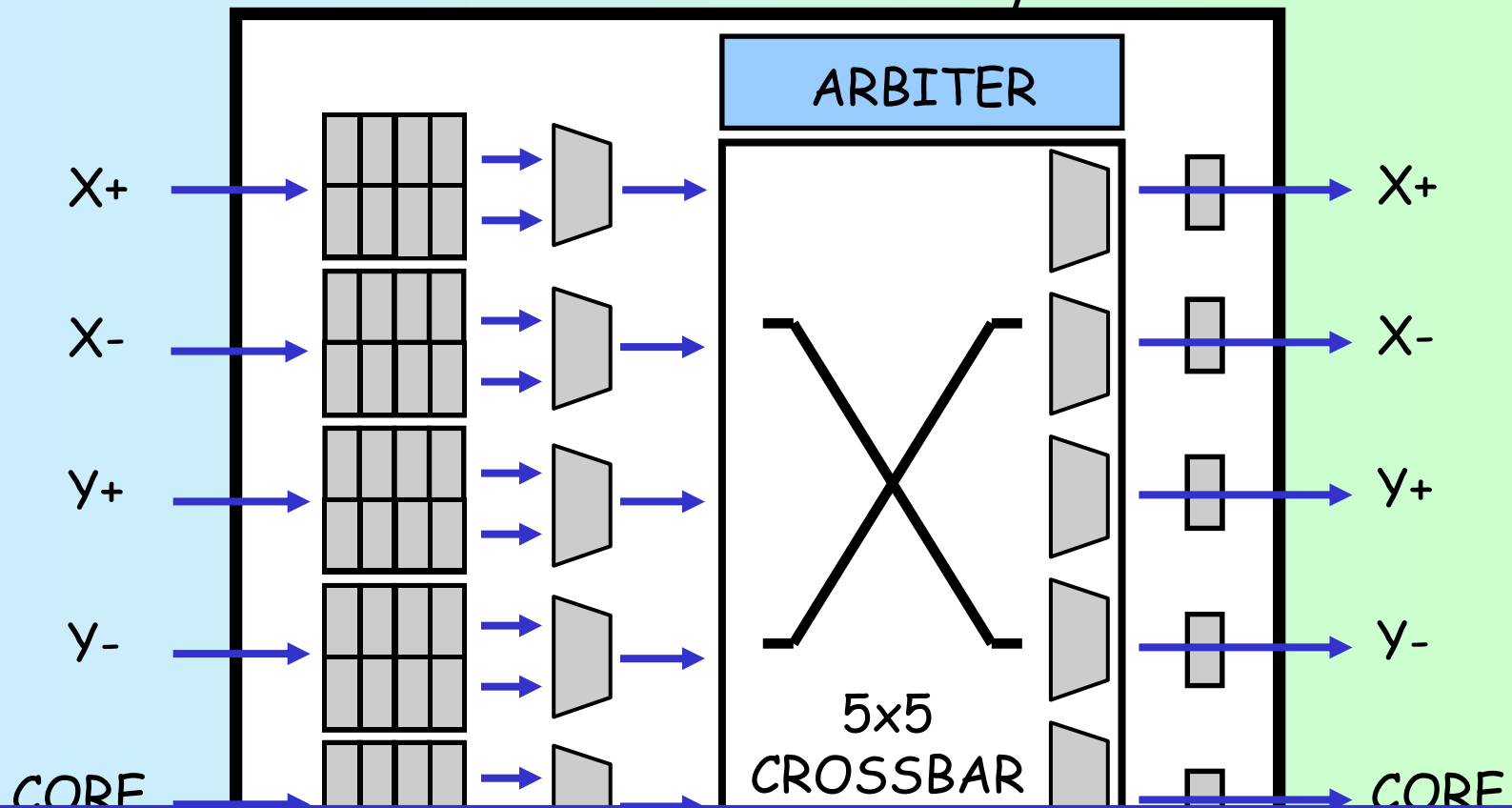
35 power domains  
in a 5-port router



# Fine-grain run-time PG router

- Router is divided into many Micro-power-domains
  - Input VC buffers, Output latches
  - Crossbar MUXes, VC MUXes

35 power domains  
in a 5-port router



Each power domain is activated only when it is "really" used

# Outline: Fine-grain power gating router

- Fine-grained power gating router
  - Input VC buffers
  - Crossbar MUXes, VC MUXes
  - Output latches

} 35 power domains in each router
- Power domain implementation @ 65nm
  - Design flow
  - Wakeup latency estimation and its impact
- Three early wakeup methods
- Evaluation results
  - Application performance w/ early wakeup
  - Leakage power reduction

# Power domain design: Design flow

- Verilog netlist  
Synopsys DesignCompiler
- HOLD cell insertion  
not to propagate "X"  
By hand
- Place-and-route  
Synopsys Astro
- Power switch insertion  
Sequence Design CoolPower
- Place-and-route again  
Synopsys Astro
- RC extraction  
Cadence Assura (QRC)
- SPICE simulation  
Synopsys HSIM

```
module FIFO (in, out);  
  input [127:0] in;  
  output [127:0] out;  
  DFF reg0 (in0, out0, clk);  
  DFF reg1 (in1, out1, clk);  
  ....  
endmodule
```

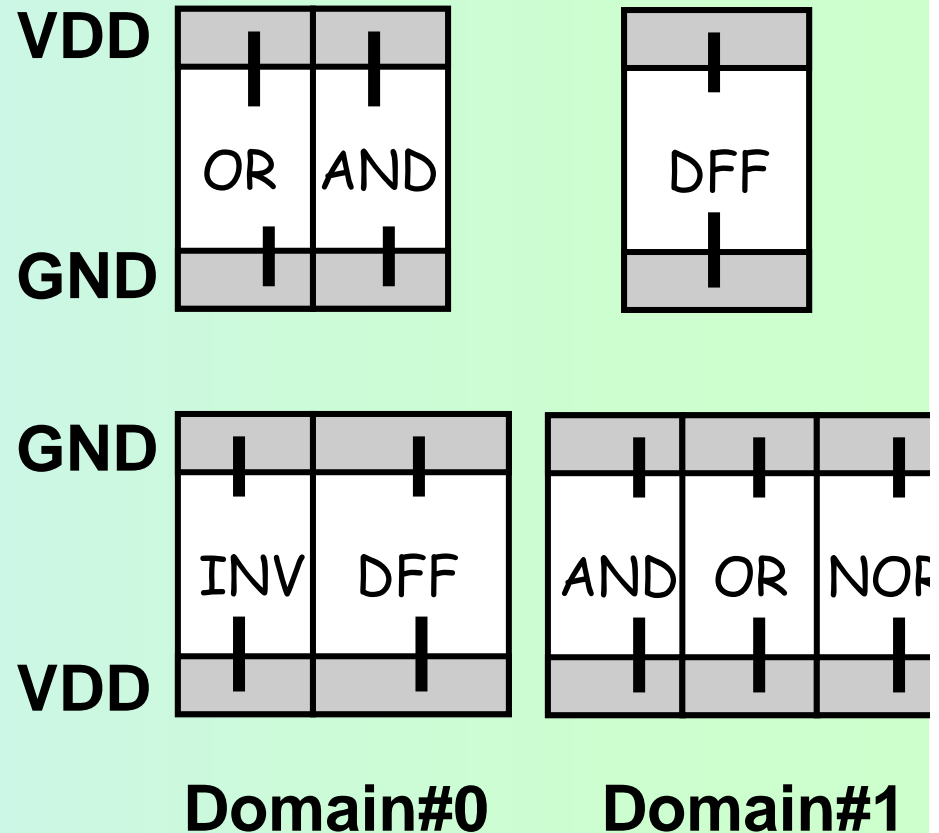
# Power domain design: Design flow

- Verilog netlist  
*Synopsys DesignCompiler*
- HOLD cell insertion  
not to propagate "X"  
*By hand*
- Place-and-route  
*Synopsys Astro*
- Power switch insertion  
*Sequence Design CoolPower*
- Place-and-route again  
*Synopsys Astro*
- RC extraction  
*Cadence Assura (QRC)*
- SPICE simulation  
*Synopsys HSIM*

```
module FIFO (in, out);  
  input [127:0] in;  
  output [127:0] out;  
  DFF reg0 (in0, out0, clk);  
  DFF reg1 (in1, out1, clk);  
  ....  
  HOLD (out0);  
  HOLD (out1);  
endmodule
```

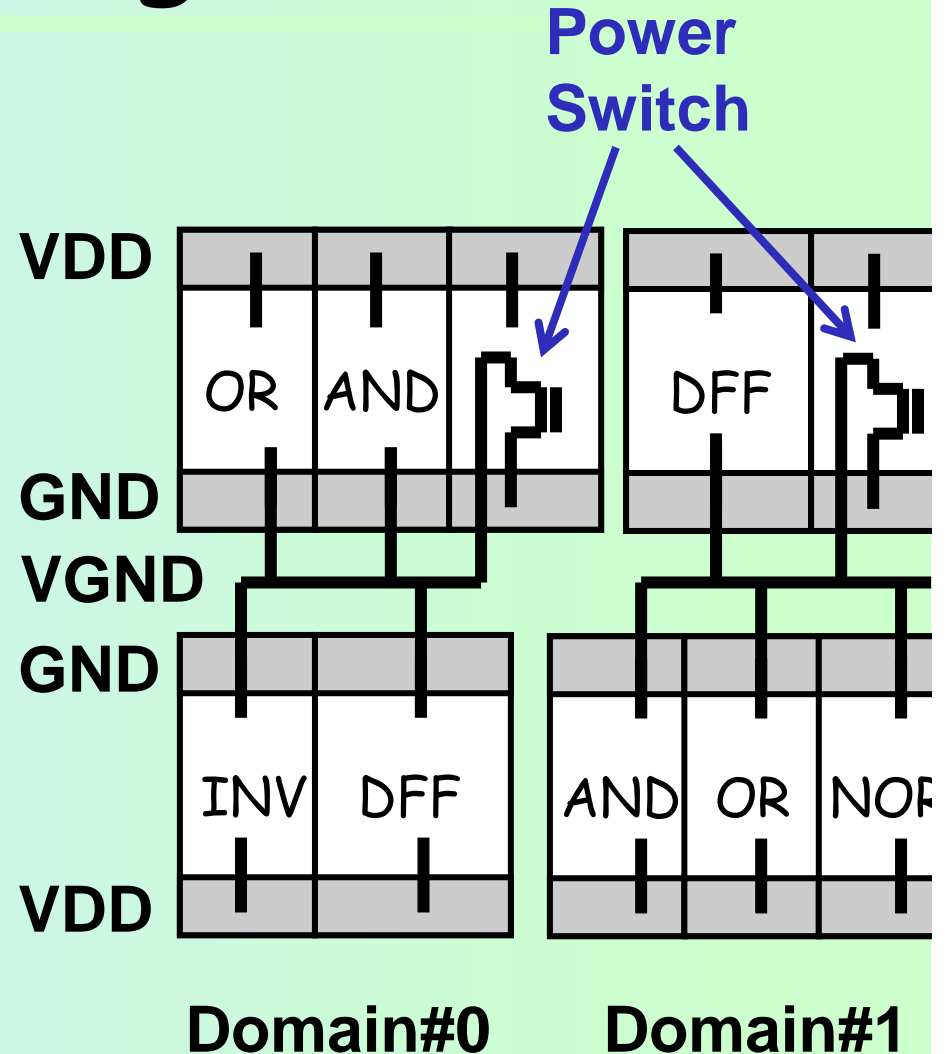
# Power domain design: Design flow

- Verilog netlist  
*Synopsys DesignCompiler*
- HOLD cell insertion  
not to propagate "X"  
*By hand*
- Place-and-route  
*Synopsys Astro*
- Power switch insertion  
*Sequence Design CoolPower*
- Place-and-route again  
*Synopsys Astro*
- RC extraction  
*Cadence Assura (QRC)*
- SPICE simulation  
*Synopsys HSIM*



# Power domain design: Design flow

- Verilog netlist  
*Synopsys DesignCompiler*
- HOLD cell insertion  
not to propagate "X"  
*By hand*
- Place-and-route  
*Synopsys Astro*
- **Power switch insertion**  
*Sequence Design CoolPower*
- Place-and-route again  
*Synopsys Astro*
- RC extraction  
*Cadence Assura (QRC)*
- SPICE simulation  
*Synopsys HSIM*



## Area overhead:

Power switch and hold cells: 4.3%

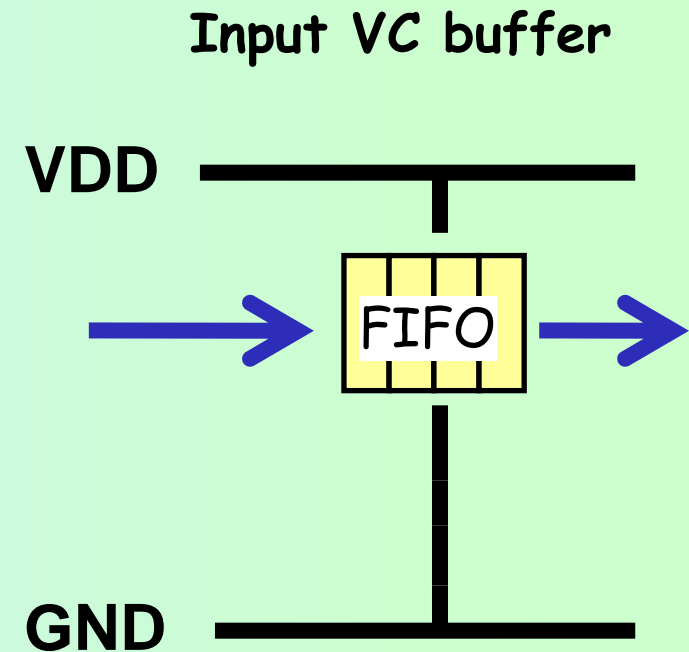
But, we need larger cells: 15.9% in total

# Power domain design: Design flow

- Verilog netlist  
*Synopsys DesignCompiler*
- HOLD cell insertion  
not to propagate "X"  
*By hand*
- Place-and-route  
*Synopsys Astro*
- Power switch insertion  
*Sequence Design CoolPower*
- Place-and-route again  
*Synopsys Astro*
- RC extraction  
*Cadence Assura (QRC)*
- **SPICE simulation**  
*Synopsys HSIM*

## On/Off control

- Power On when packet comes
- Power Off when packet leaves

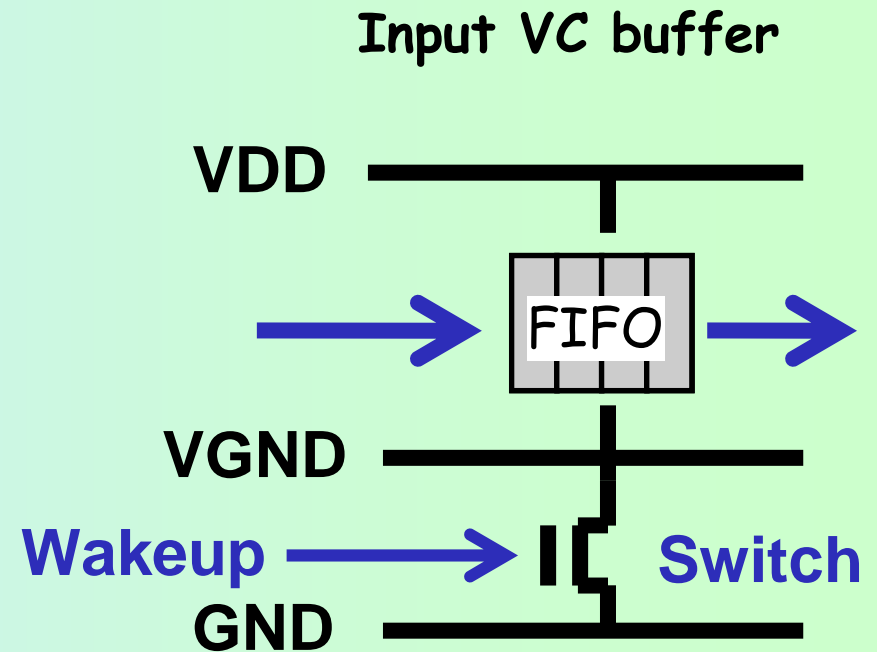


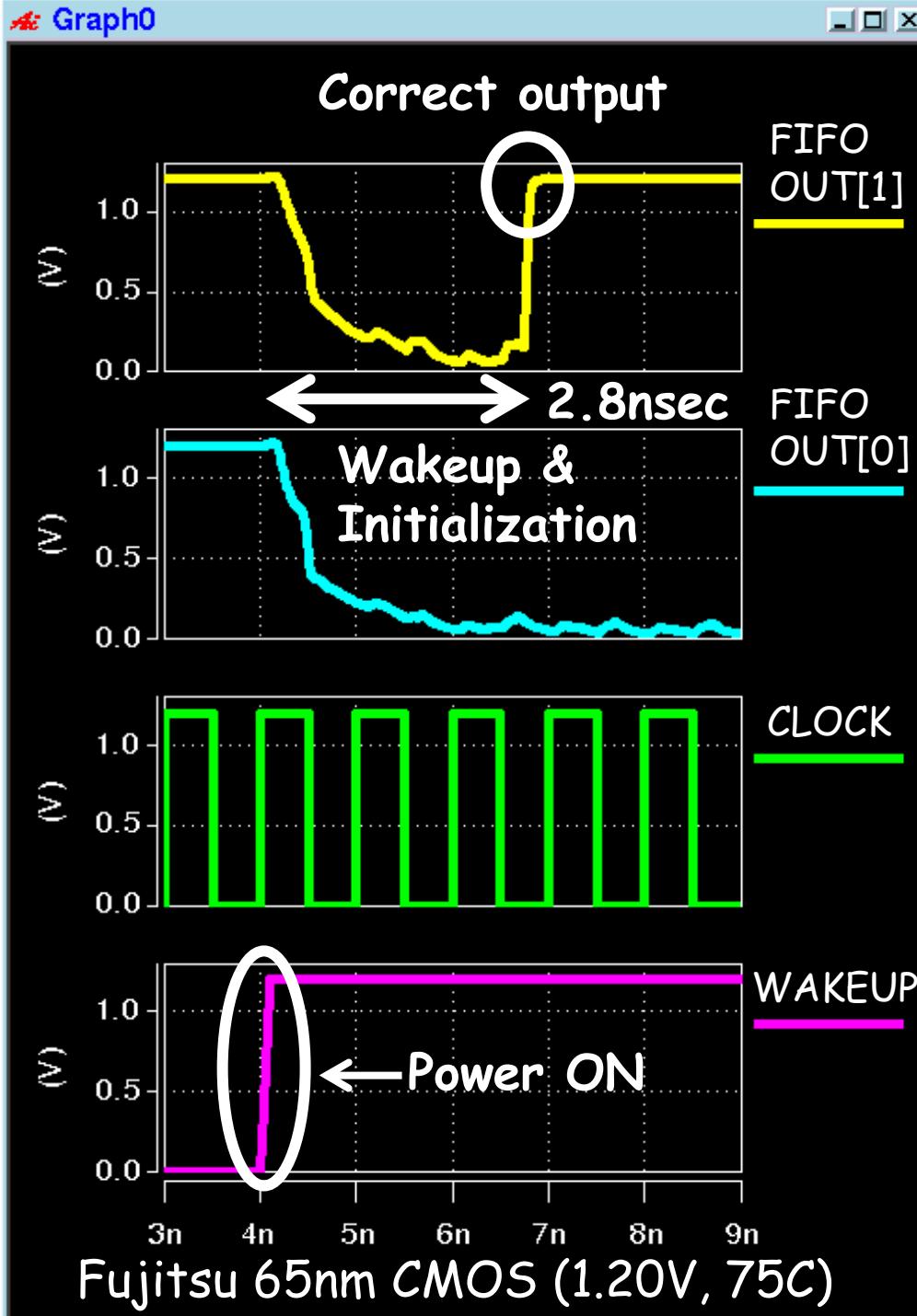
# Power domain design: Design flow

- Verilog netlist  
*Synopsys DesignCompiler*
- HOLD cell insertion  
not to propagate "X"  
*By hand*
- Place-and-route  
*Synopsys Astro*
- Power switch insertion  
*Sequence Design CoolPower*
- Place-and-route again  
*Synopsys Astro*
- RC extraction  
*Cadence Assura (QRC)*
- **SPICE simulation**  
*Synopsys HSIM*

## On/Off control

- Power On when packet comes
- Power Off when packet leaves



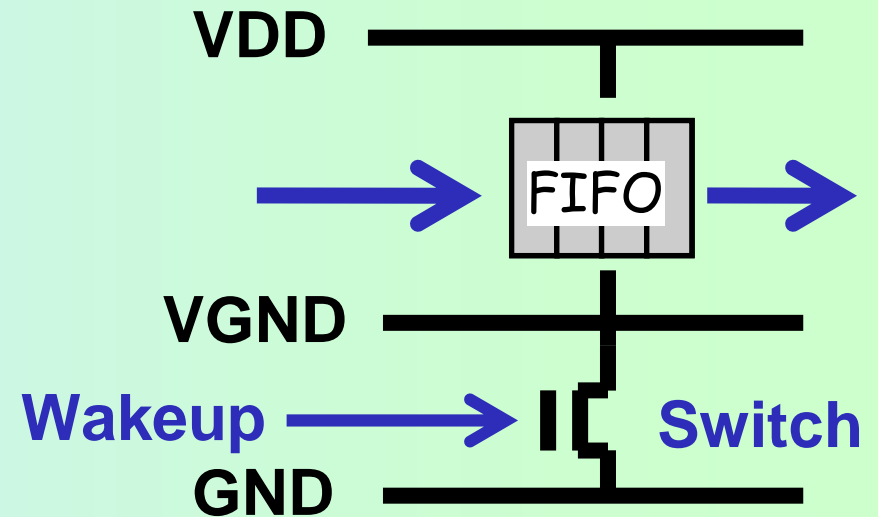


# Design: Design flow

## On/Off control

- Power On when packet comes
- Power Off when packet leaves

## Input VC buffer



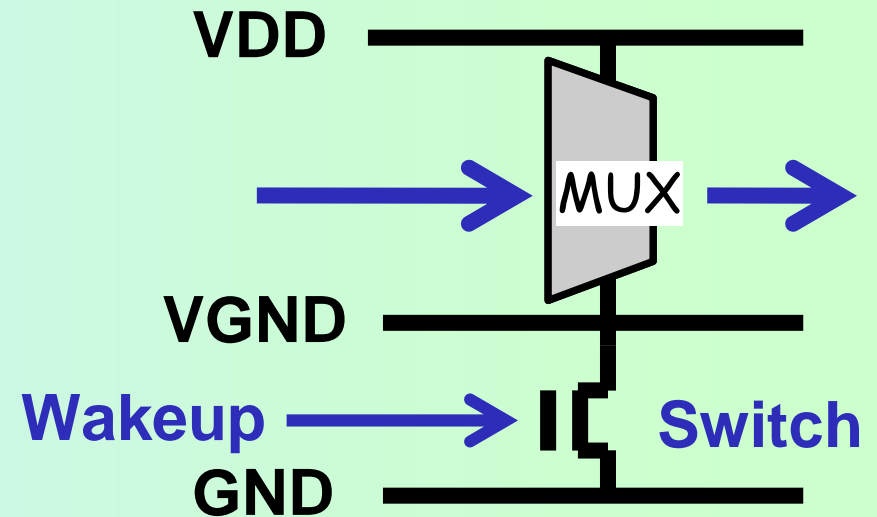
# Power domain design: Design flow

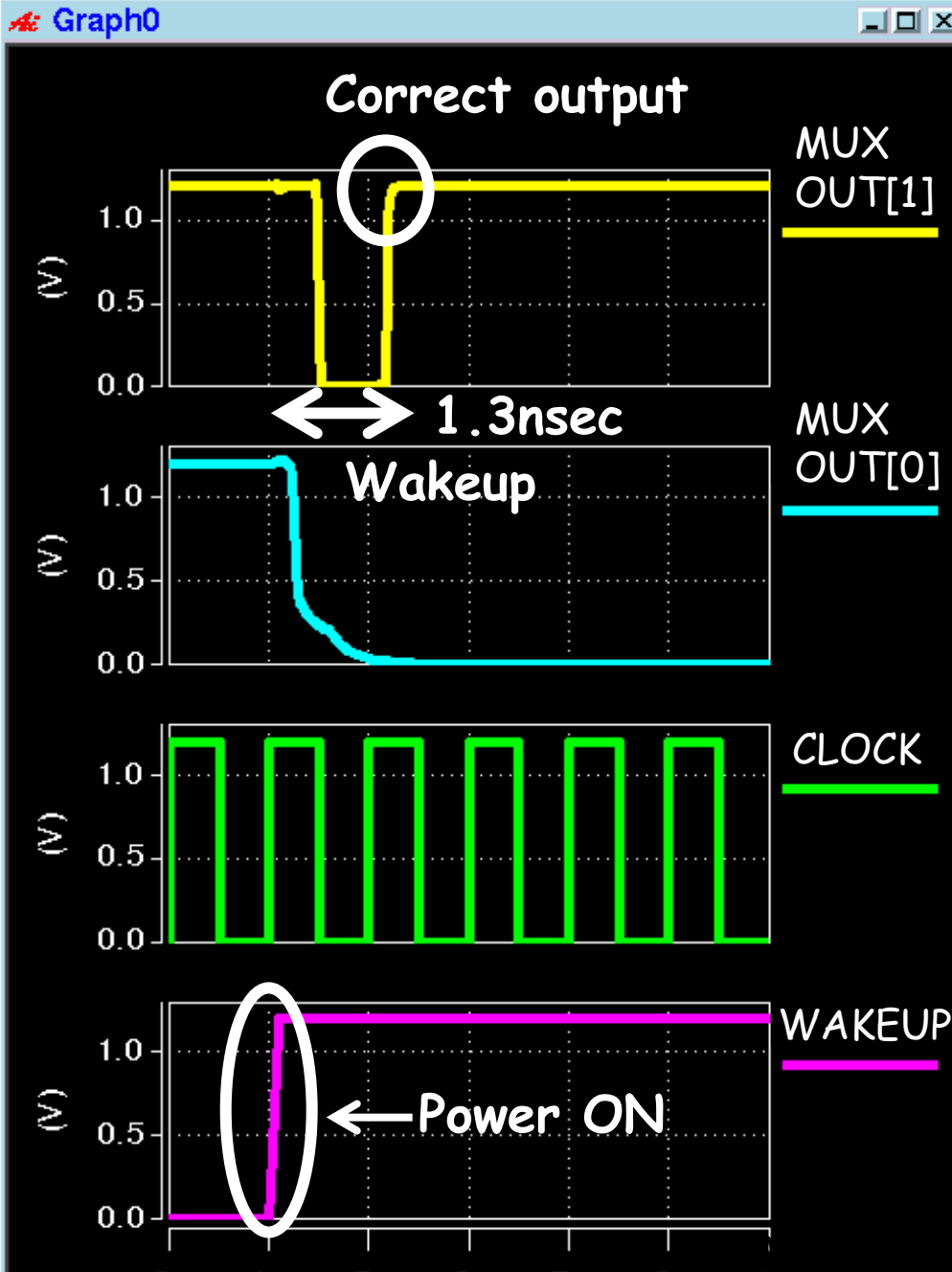
- Verilog netlist  
*Synopsys DesignCompiler*
- HOLD cell insertion  
not to propagate "X"  
*By hand*
- Place-and-route  
*Synopsys Astro*
- Power switch insertion  
*Sequence Design CoolPower*
- Place-and-route again  
*Synopsys Astro*
- RC extraction  
*Cadence Assura (QRC)*
- SPICE simulation  
*Synopsys HSIM*

## On/Off control

- Power On when packet comes
- Power Off when packet leaves

## Crossbar multiplexer



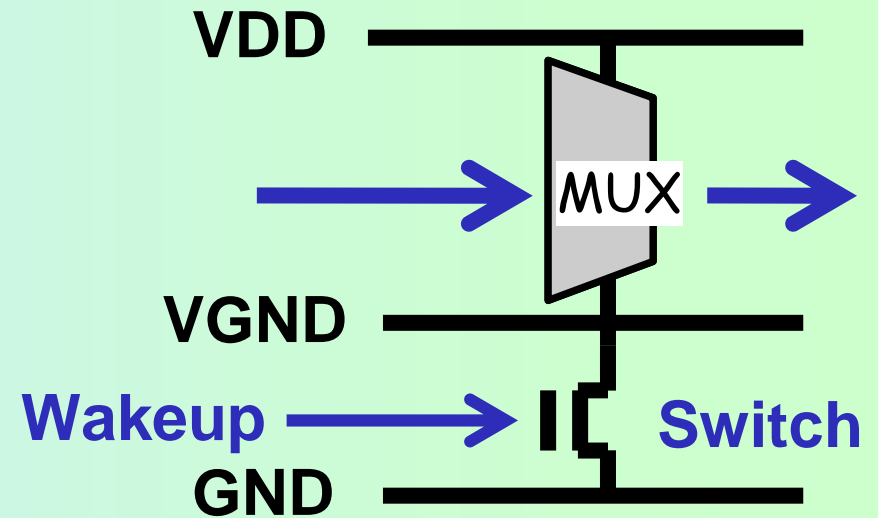


# Design: Design flow

## On/Off control

- Power On when packet comes
- Power Off when packet leaves

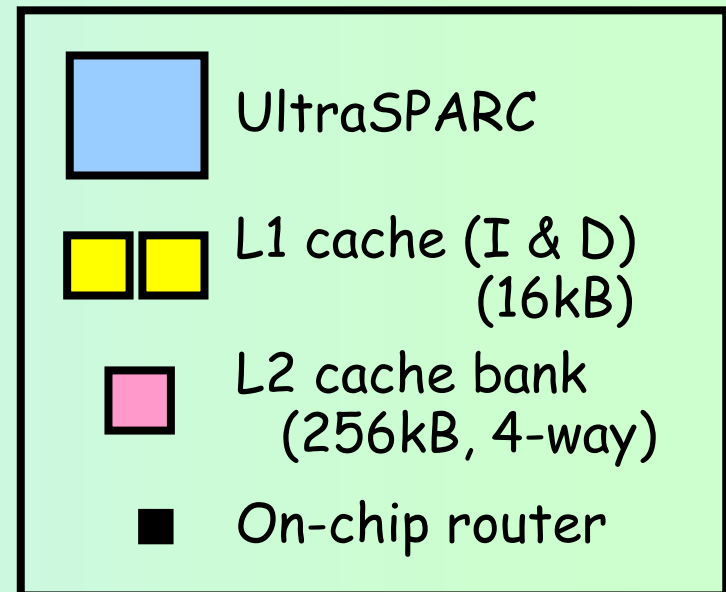
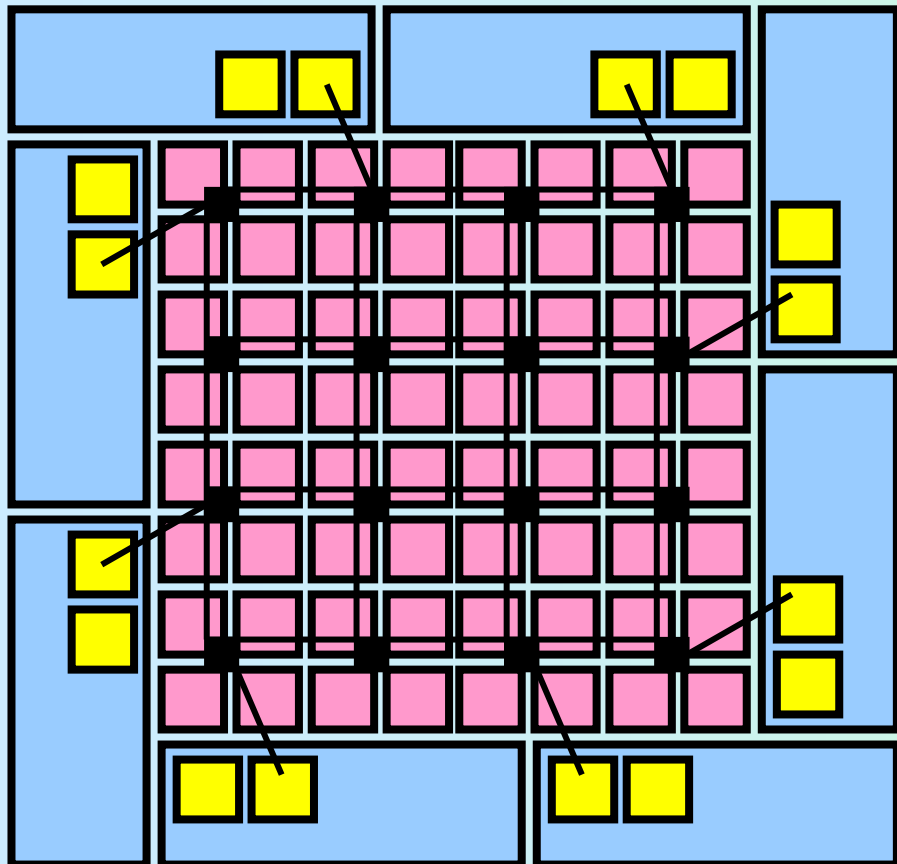
## Crossbar multiplexer



All power domains in this router can be activated within

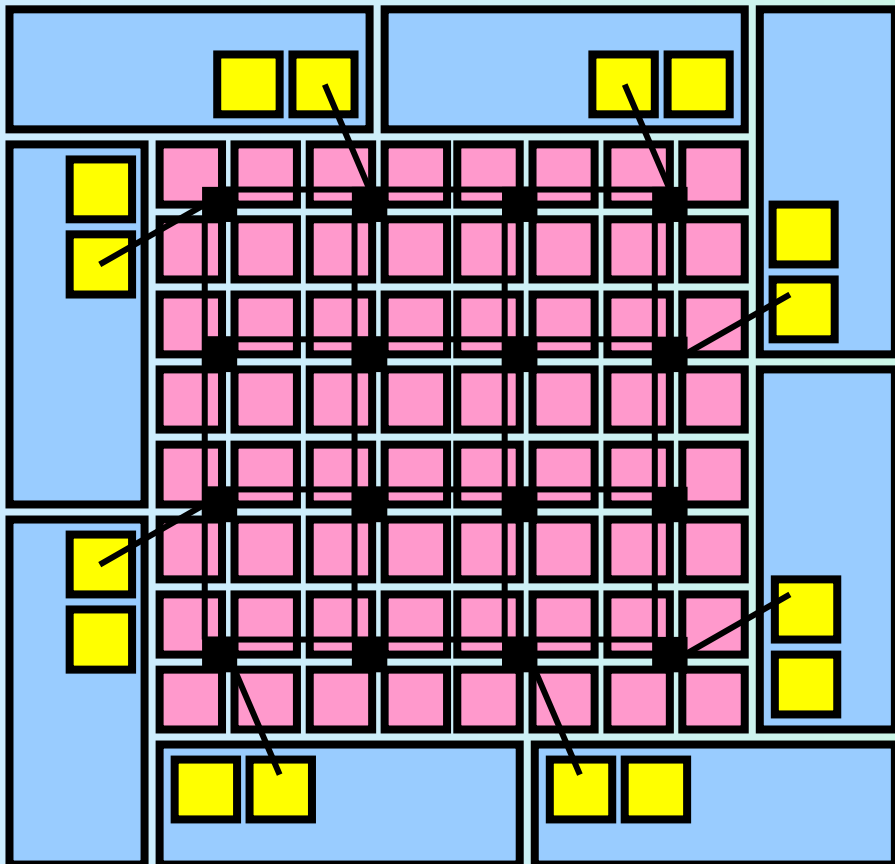
# Wake up latency impact on CMPs

- Full system CMP simulator: GEMS/Simics [Martin, CAN'05]
  - 3-cycle router [RC] [VSA] [ST] radix, lu, fft, barnes, ocean, raytrace, volrend, water-ns, water-sp, fmm (10 applications)
  - Wakeup latencies: 2, 3, 4 cycles
  - SPLASH-2 benchmark (8 threads)



# Wake up latency impact on CMPs

- Full system CMP simulator: GEMS/Simics [Martin, CAN'05]
  - 3-cycle router [RC][VSA][ST] radix, lu, fft, barnes,
  - Wakeup latencies: 2, 3, 4 cycles ocean, raytrace, volrend,
  - SPLASH-2 benchmark (8 threads) water-ns, water-sp, fmm (10 applications)

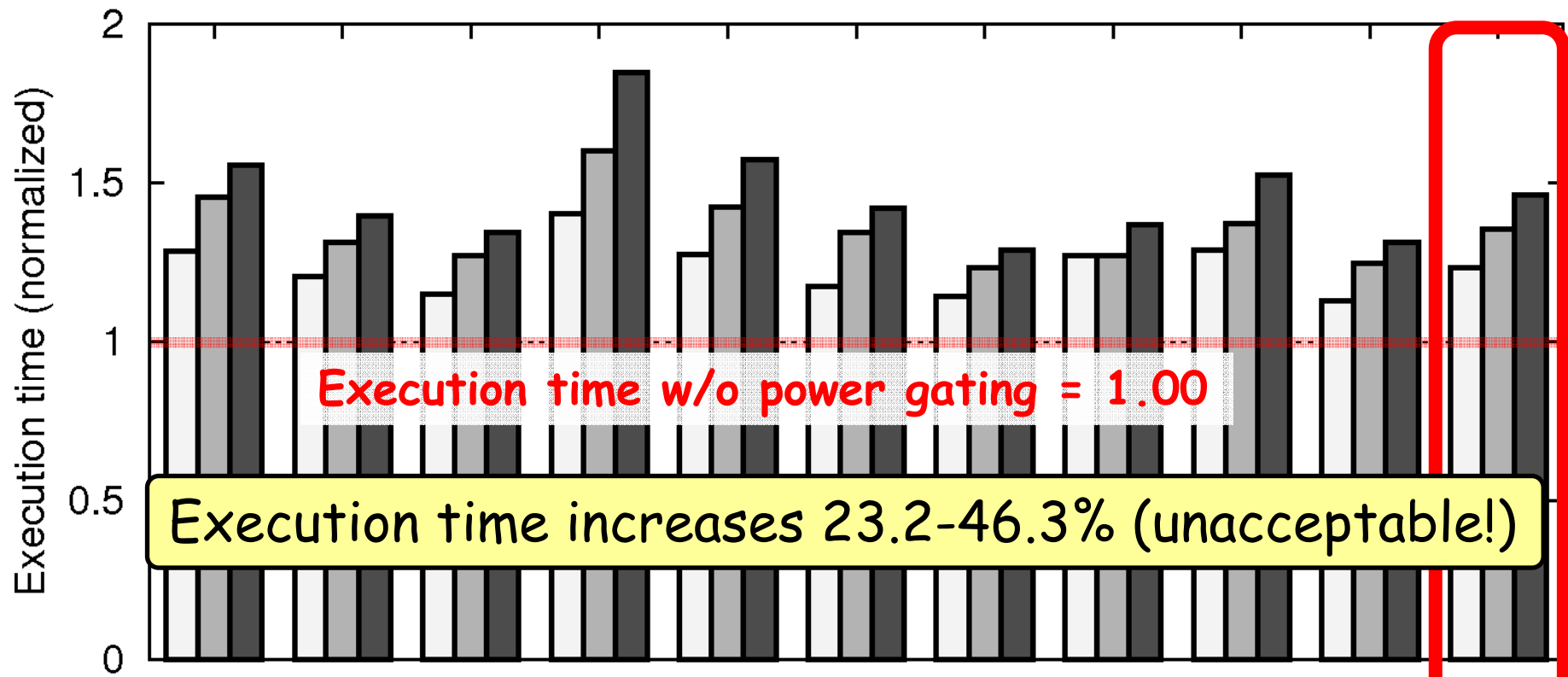


## Token coherence protocol

- VC0 [Martin, ISCA'03]
  - Request msg (L1 ↔ L2)
- VC1
  - Request msg (L2 ↔ Mem)
- VC2
  - Reply msg (All ↔ All)
- VC3
  - Persistent request msg

# Wakeup latency impact: Results

- Execution times of SPLASH-2 (10 applications)



Longer execution consumes more power; Early wakeup required

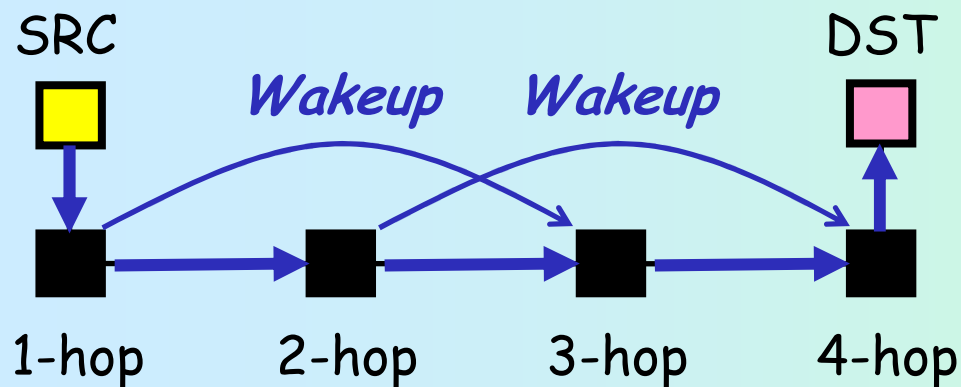
# Outline: Fine-grain power gating router

- Fine-grained power gating router
  - Input VC buffers
  - Crossbar MUXes, VC MUXes
  - Output latches

} 35 power domains in each router
- Power domain implementation @ 65nm
  - Design flow
  - Wakeup latency estimation and its impact
- Three early wakeup methods
- Evaluation results
  - Application performance w/ early wakeup
  - Leakage power reduction

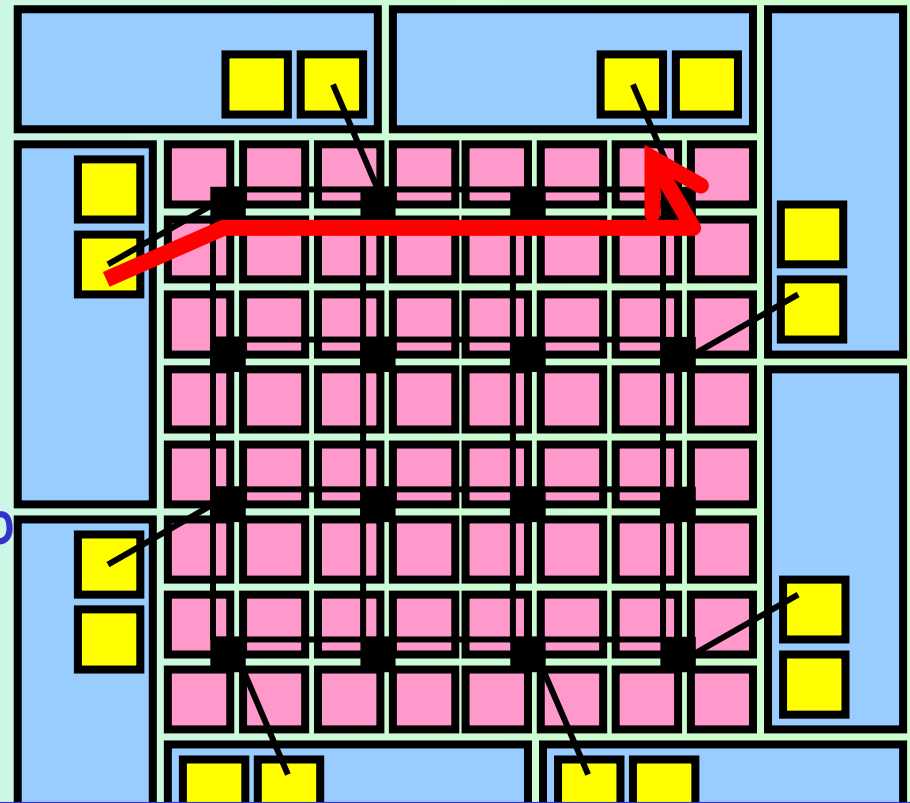
# Early wakeup: Look-ahead method

- Router modules in 2-hop away activated in advance
  - Look-ahead routing is used [Matsutani, ASPDAC'08]
  - Wakeup procedure starts 4-cycle in advance



- Problems
  - Wakeup signals span 2-hop
  - Cannot wakeup the 1-hop router in advance

(\*) Average hop is 2.67 @ 4x4 mesh

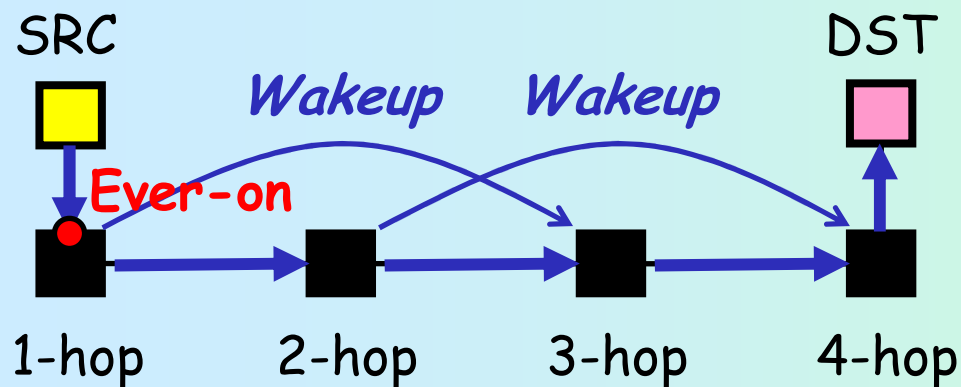


Suffers first-hop wakeup latency; Small benefit of look-ahead

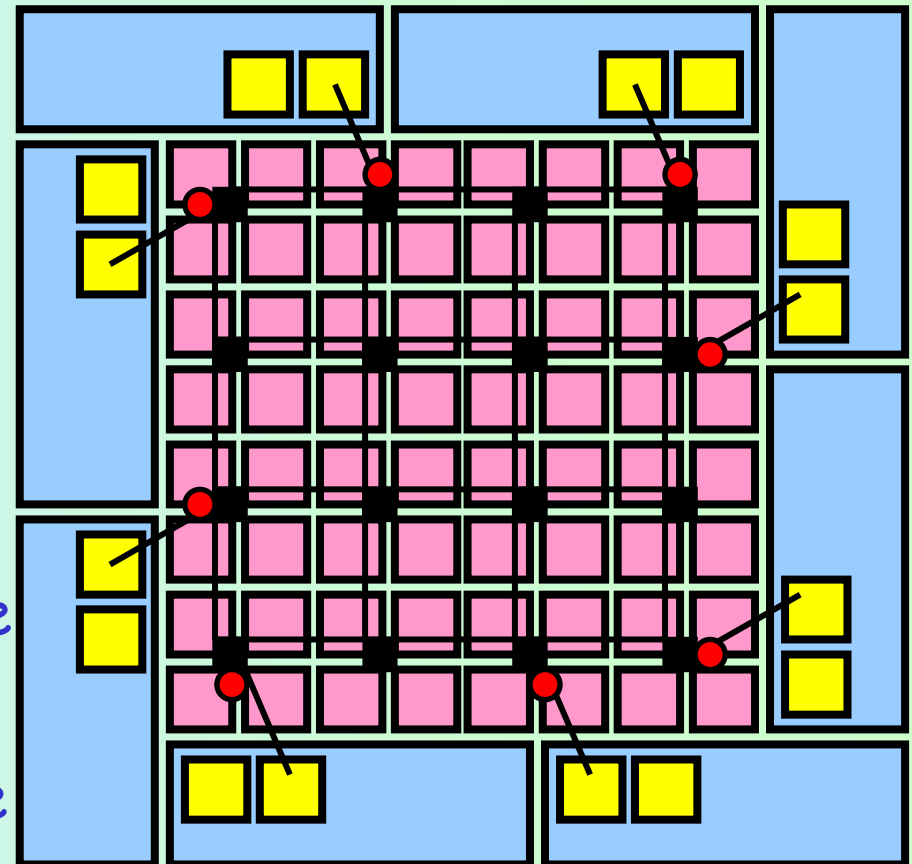
# Early wakeup: Look-ahead method

+ CPU ever-on

- Router modules in 2-hop away activated in advance
  - Look-ahead routing is used
  - Wakeup procedure starts 4-cycle in advance



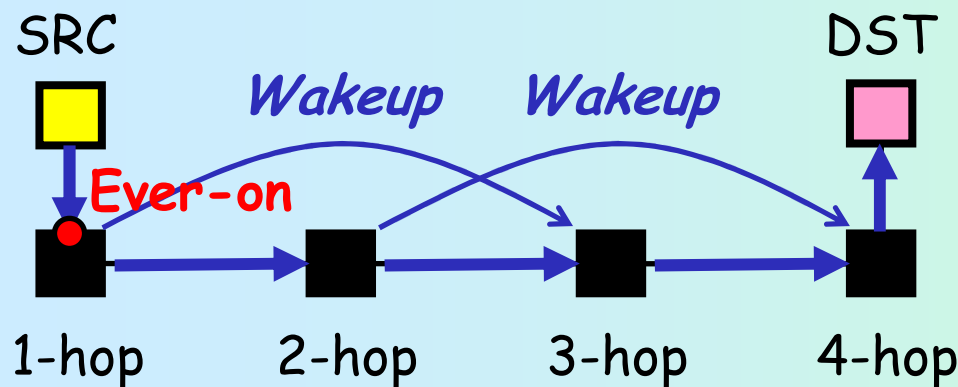
- Ever-on domain
  - VC buffers connected from CPU are always active
  - No wait for the first-hop
  - But, they consume leakage



# Early wakeup: Look-ahead method

+ CPU ever-on

- Router modules in 2-hop away activated in advance
  - Look-ahead routing is used
  - Wakeup procedure starts 4-cycle in advance



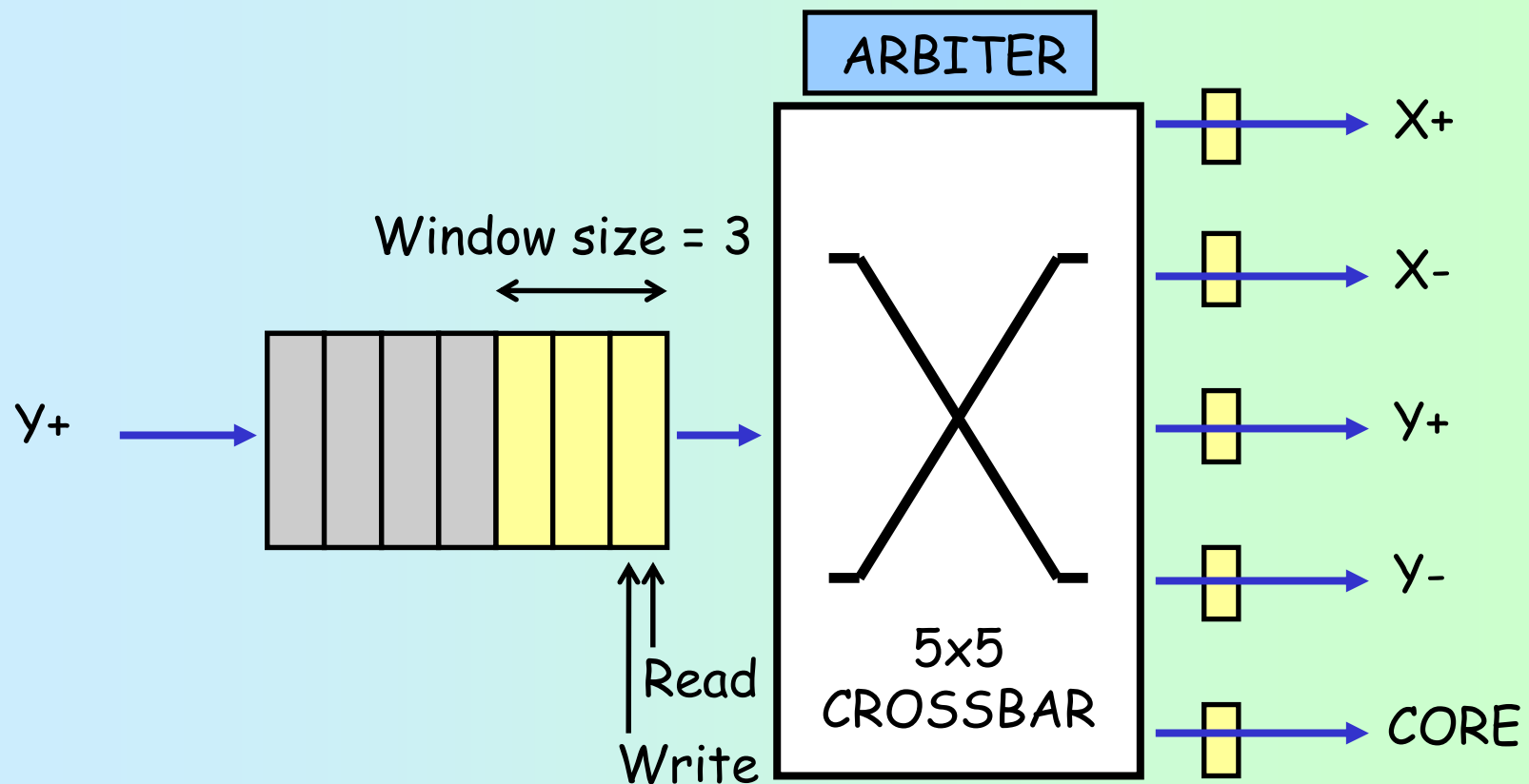
- Ever-on domain
    - VC0 and VC2 connected from CPU are always active
    - No wait for the first-hop
- (\*) VC0 and VC2 are heavily loaded

- VC0
  - Request msg (L1 ↔ L2)
- VC1
  - Request msg (L2 ↔ Mem)
- VC2
  - Reply msg (All ↔ All)
- VC3
  - Persistent request msg

Ever-on domain is only 4.7%; Minimum leakage power

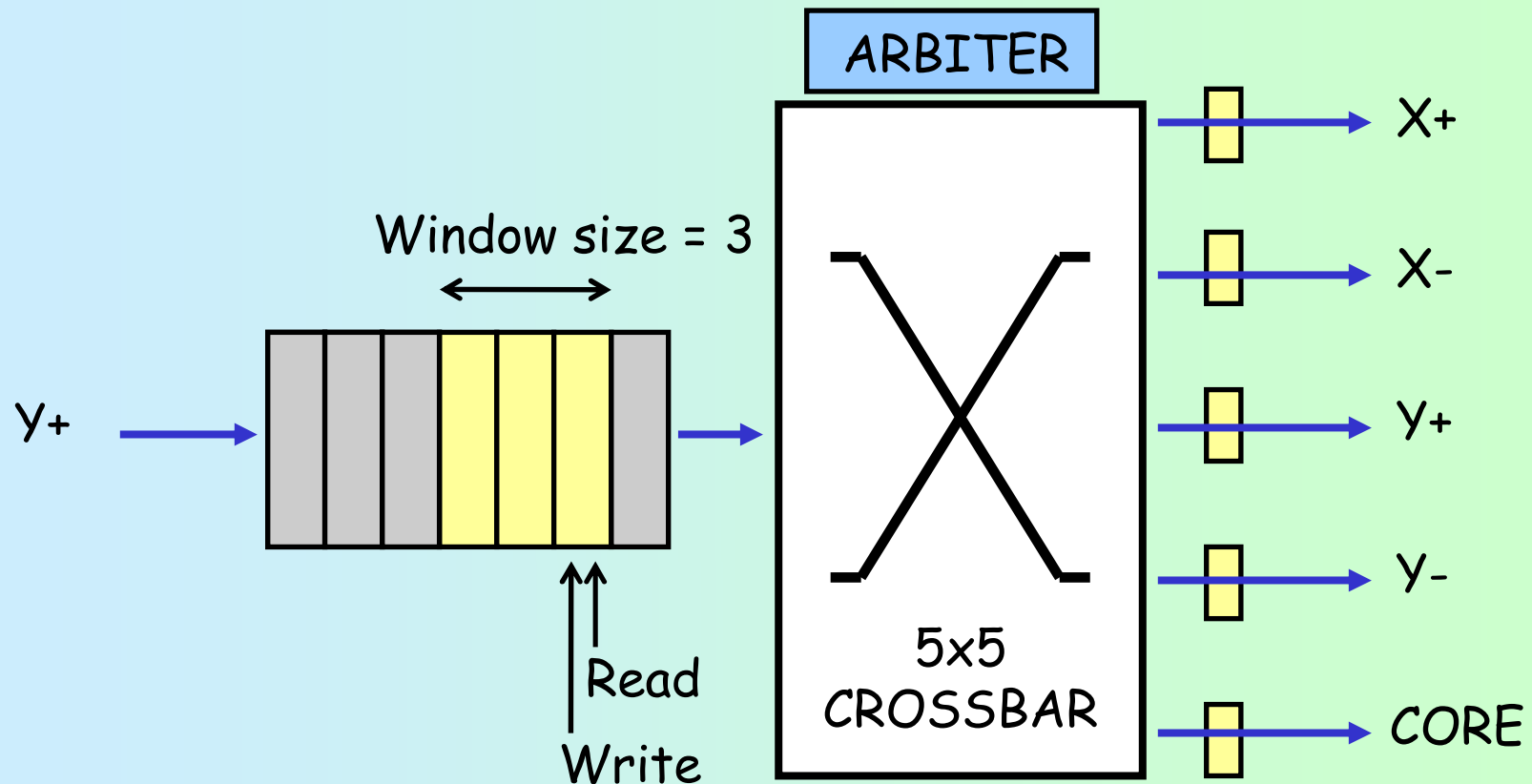
# Early wakeup: Active buffer window

- Active buffer window in each VC buffer
  - A part of the buffer is always activated [Chen,ISLPED'03]
  - Active buffer window shifts when it receives/sends flit
  - Short packets (less than window size) → No wait



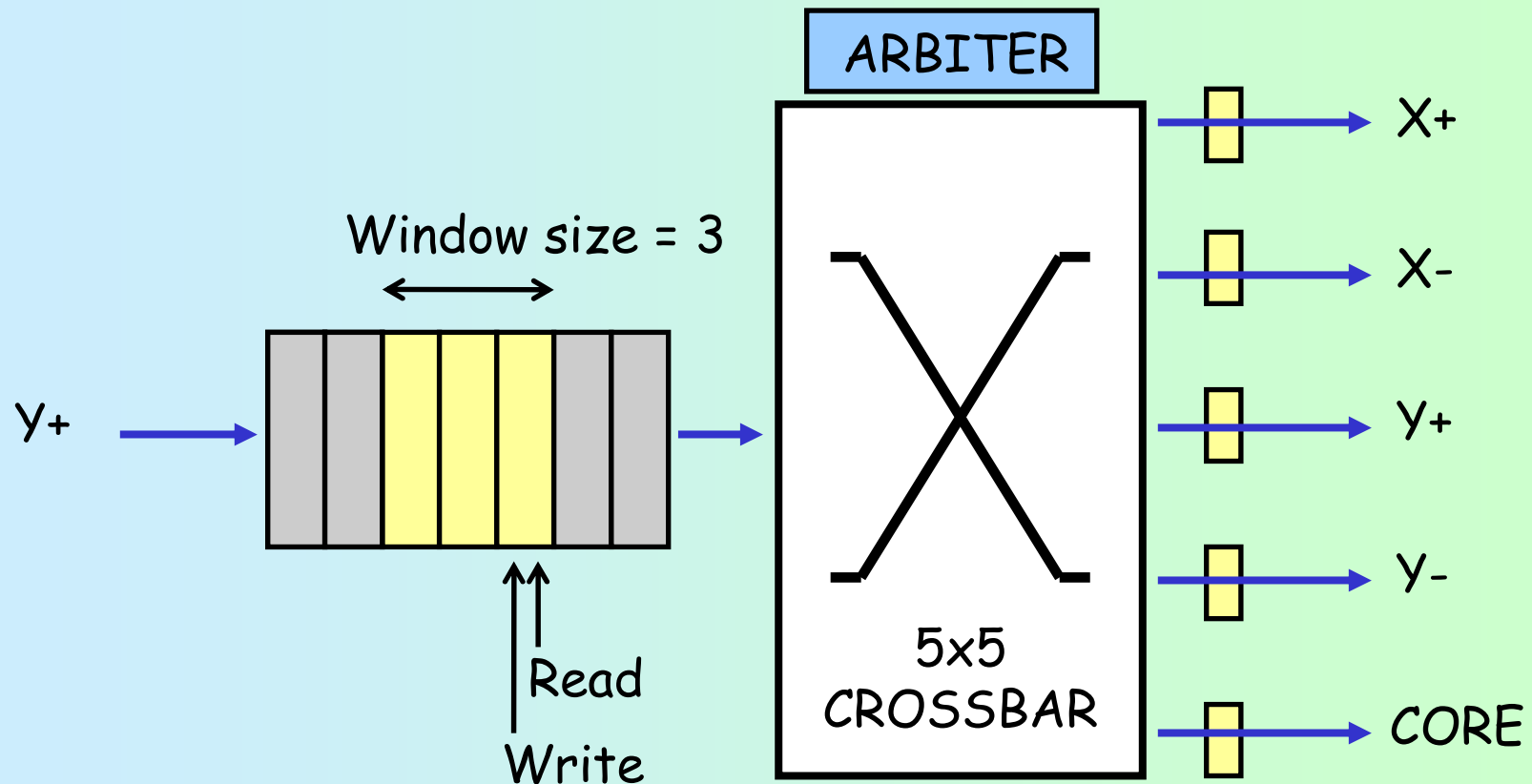
# Early wakeup: Active buffer window

- Active buffer window in each VC buffer
  - A part of the buffer is always activated [Chen,ISLPED'03]
  - Active buffer window shifts when it receives/sends flit
  - Short packets (less than window size) → No wait



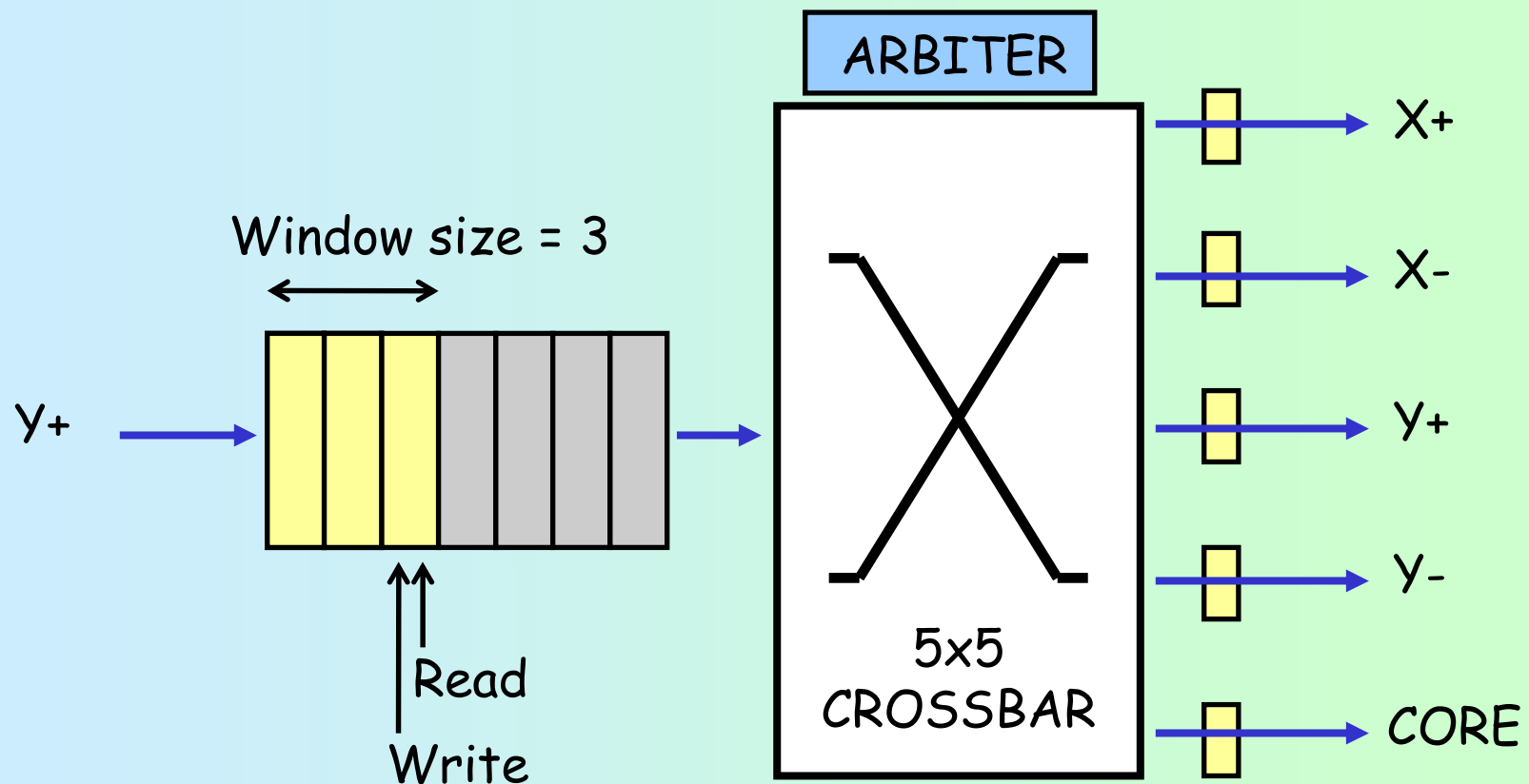
# Early wakeup: Active buffer window

- Active buffer window in each VC buffer
  - A part of the buffer is always activated [Chen,ISLPED'03]
  - Active buffer window shifts when it receives/sends flit
  - Short packets (less than window size) → No wait



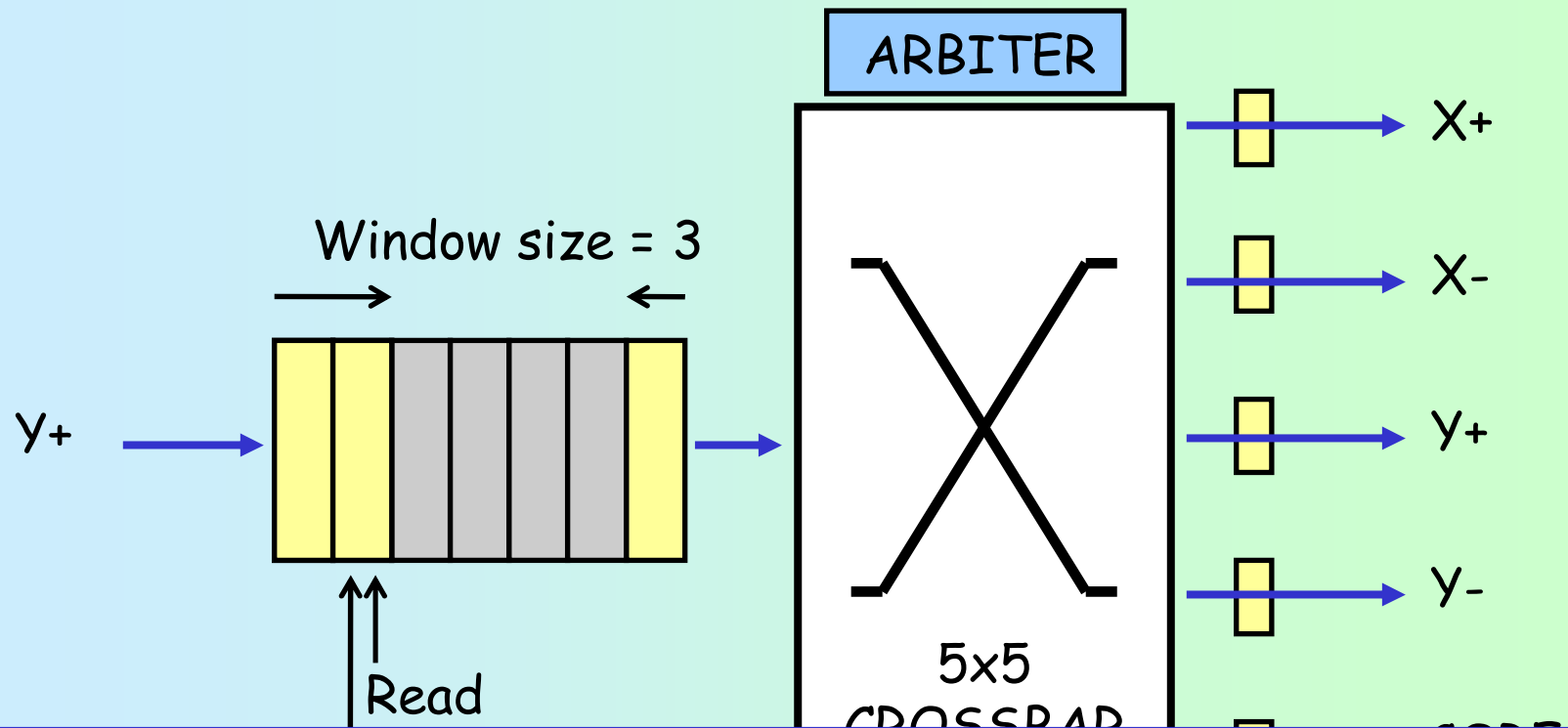
# Early wakeup: Active buffer window

- Active buffer window in each VC buffer
  - A part of the buffer is always activated [Chen,ISLPED'03]
  - Active buffer window shifts when it receives/sends flit
  - Short packets (less than window size) → No wait



# Early wakeup: Active buffer window

- Active buffer window in each VC buffer
  - A part of the buffer is always activated [Chen,ISLPED'03]
  - Active buffer window shifts when it receives/sends flit
  - Short packets (less than window size) → No wait



No wait, but the window consumes leakage → Small PG benefit

# Outline: Fine-grain power gating router

- Fine-grained power gating router
  - Input VC buffers
  - Crossbar MUXes, VC MUXes
  - Output latches

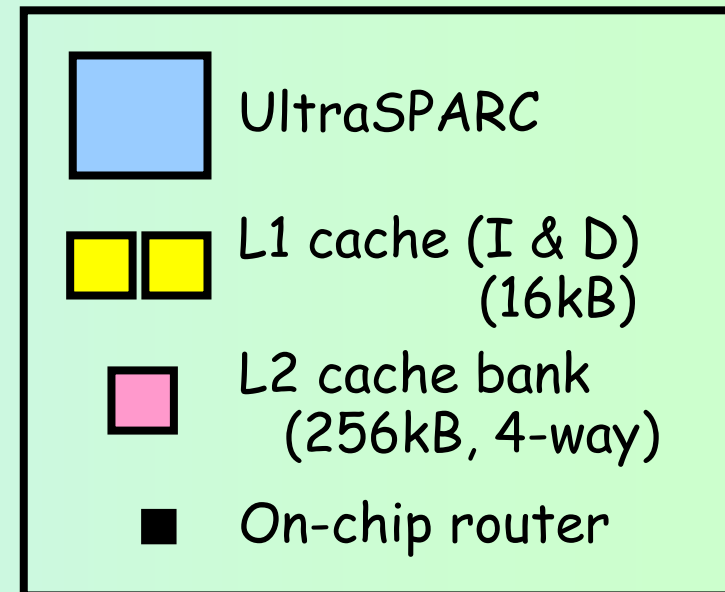
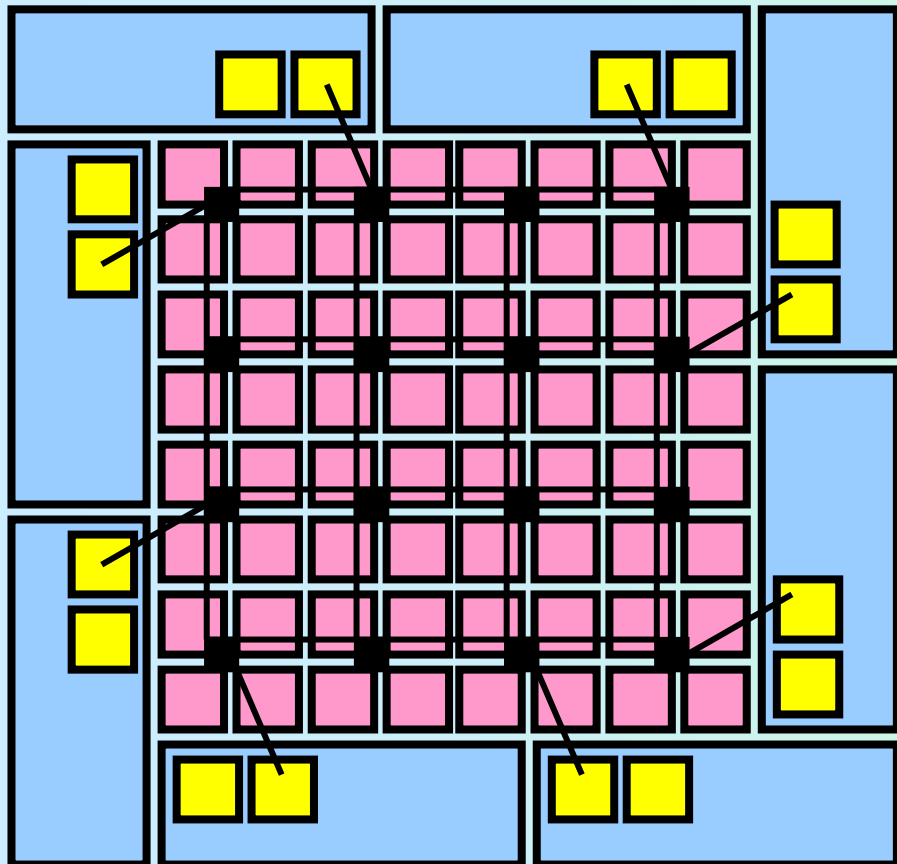
} 35 power domains in each router
- Power domain implementation @ 65nm
  - Design flow
  - Wakeup latency estimation and its impact
- Three early wakeup methods
- Evaluation results
  - Application performance w/ early wakeup
  - Leakage power reduction

# CMP simulator: GEMS/Simics

[Martin,CAN'05]

- Full system CMP simulation
  - 8 CPUs, 64 L2 banks, 4x4 mesh
  - Sun Solaris 9; Sun Studio 12
  - SPLASH-2 benchmark (8 threads)

radix, lu, fft, barnes,  
ocean, raytrace, volrend,  
water-ns, water-sp, fmm  
(10 applications)

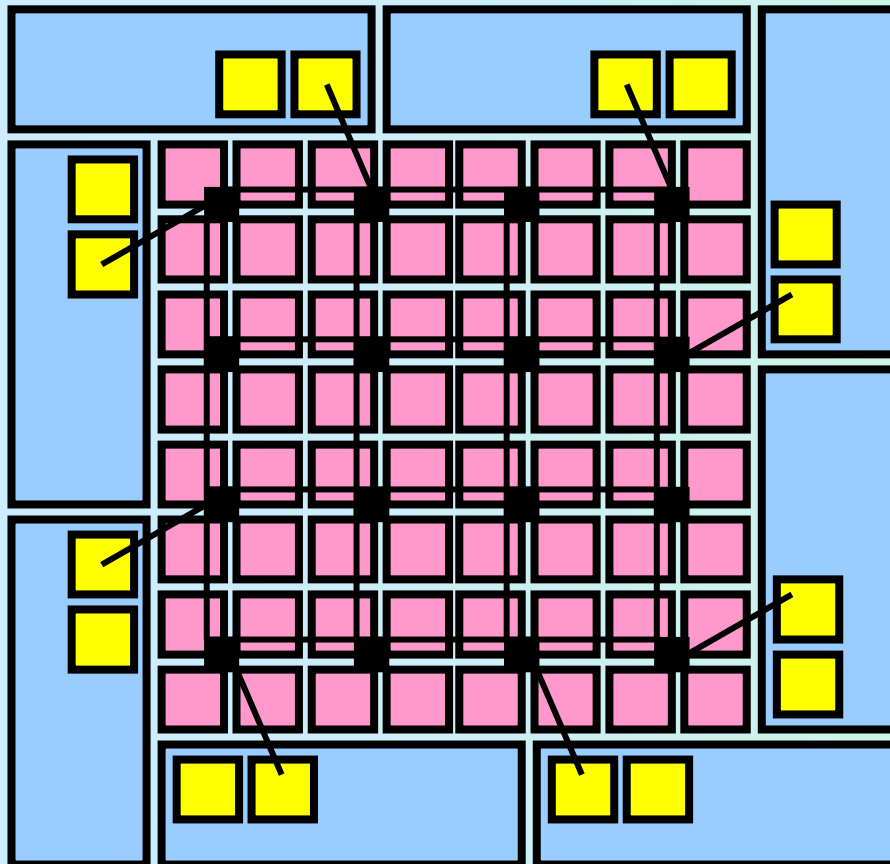


# CMP simulator: GEMS/Simics

[Martin,CAN'05]

- Full system CMP simulation
  - 8 CPUs, 64 L2 banks, 4x4 mesh
  - Sun Solaris 9; Sun Studio 12
  - SPLASH-2 benchmark (8 threads)

radix, lu, fft, barnes,  
ocean, raytrace, volrend,  
water-ns, water-sp, fmm  
(10 applications)



## Token coherence protocol

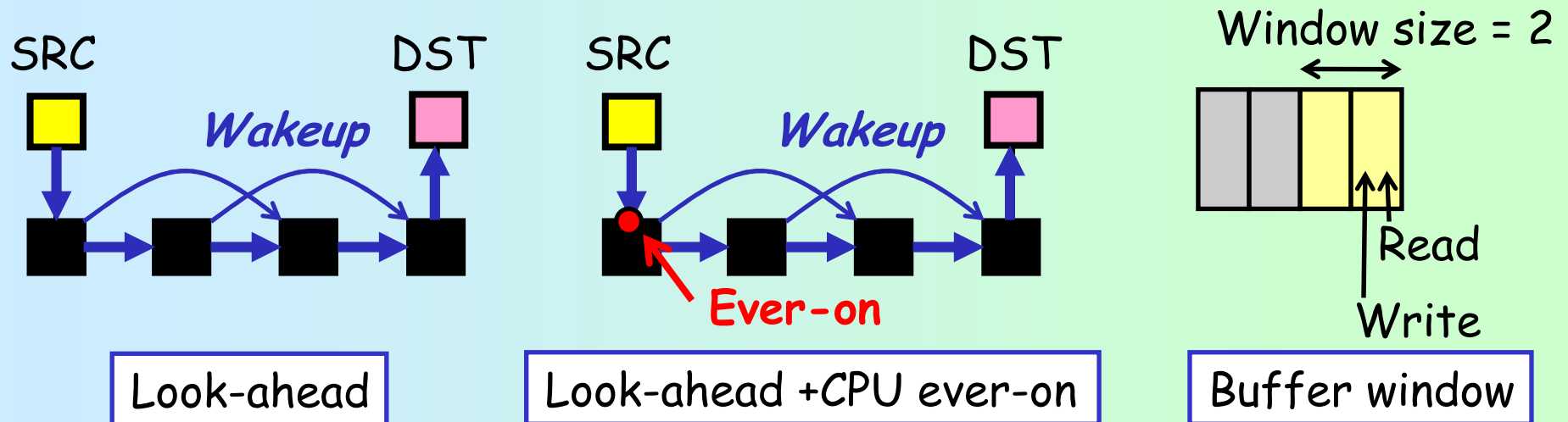
[Martin,ISCA'03]

- VC0
  - Request msg (L1 ↔ L2)
- VC1
  - Request msg (L2 ↔ Mem)
- VC2
  - Reply msg (All ↔ All)
- VC3
  - Persistent request msg

# CMP simulator: GEMS/Simics

- Full system CMP simulation
    - 8 CPUs, 64 L2 banks, 4x4 mesh
    - Sun Solaris 9; Sun Studio 12
    - SPLASH-2 benchmark (8 threads)
- radix, lu, fft, barnes, ocean, raytrace, volrend, water-ns, water-sp, fmm (10 applications)

- Three early wakeup methods are compared



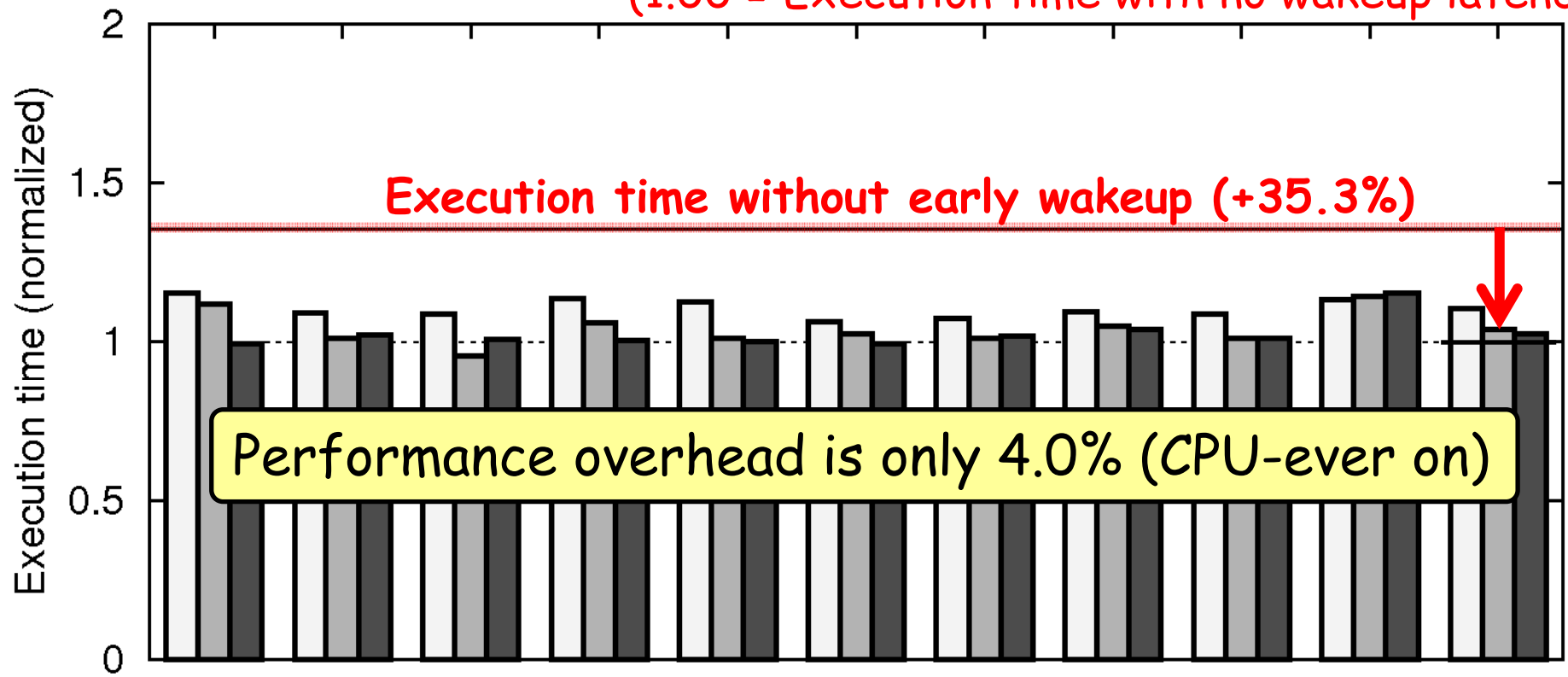
- Wakeup latency: 3nsec (3-cycle wakeup @ 1GHz)

# Evaluations: Application performance

Execution times of SPLASH-2 (3-cycle wakeup @ 1GHz)

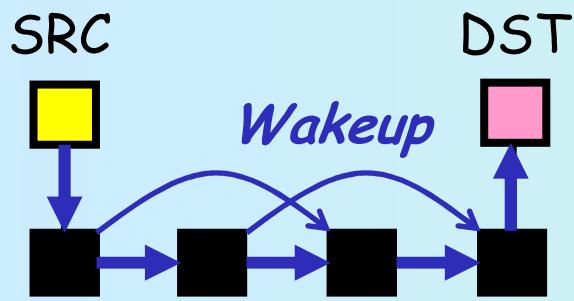
Look-ahead      Look-ahead with CPU ever-on      Buffer window

(1.00 = Execution time with no wakeup latency)

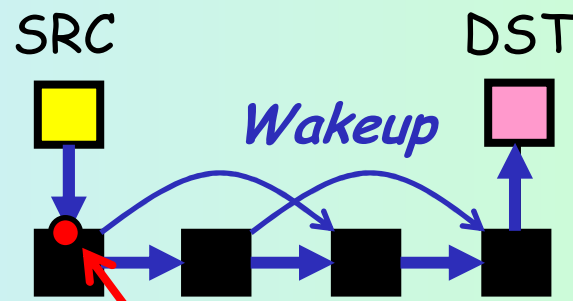


Early wakeup significantly mitigates the performance

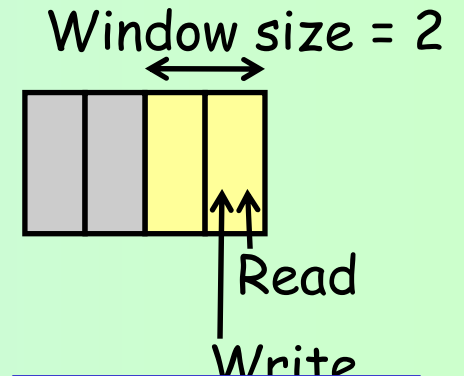
# Evaluations: Leakage power reduction



Look-ahead



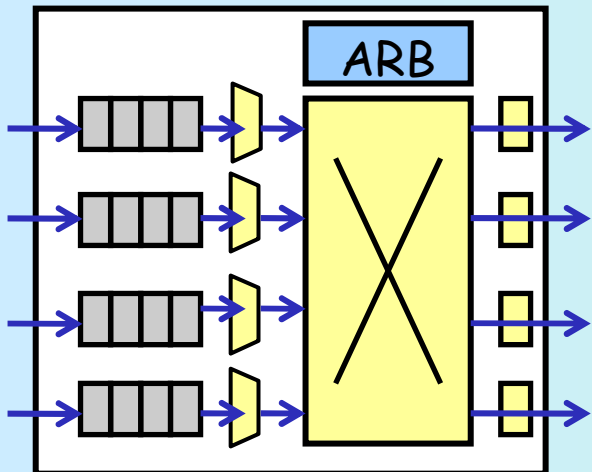
Look-ahead + CPU ever-on



Buffer window

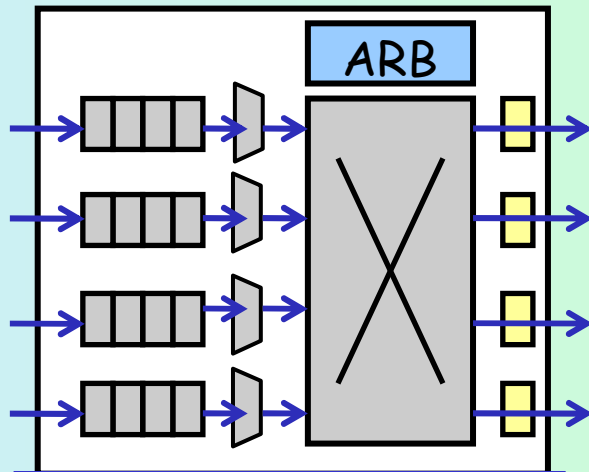
- Power gating is applied to the router with 3 steps

Level-1 power gating



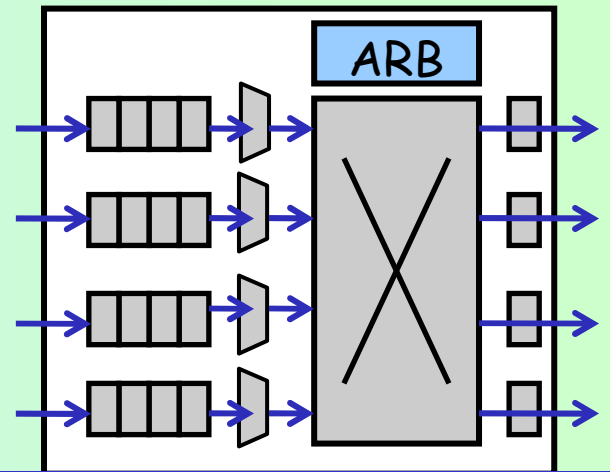
Input buffer only

Level-2 power gating



Input buffer + Xbar

Level-3 power gating

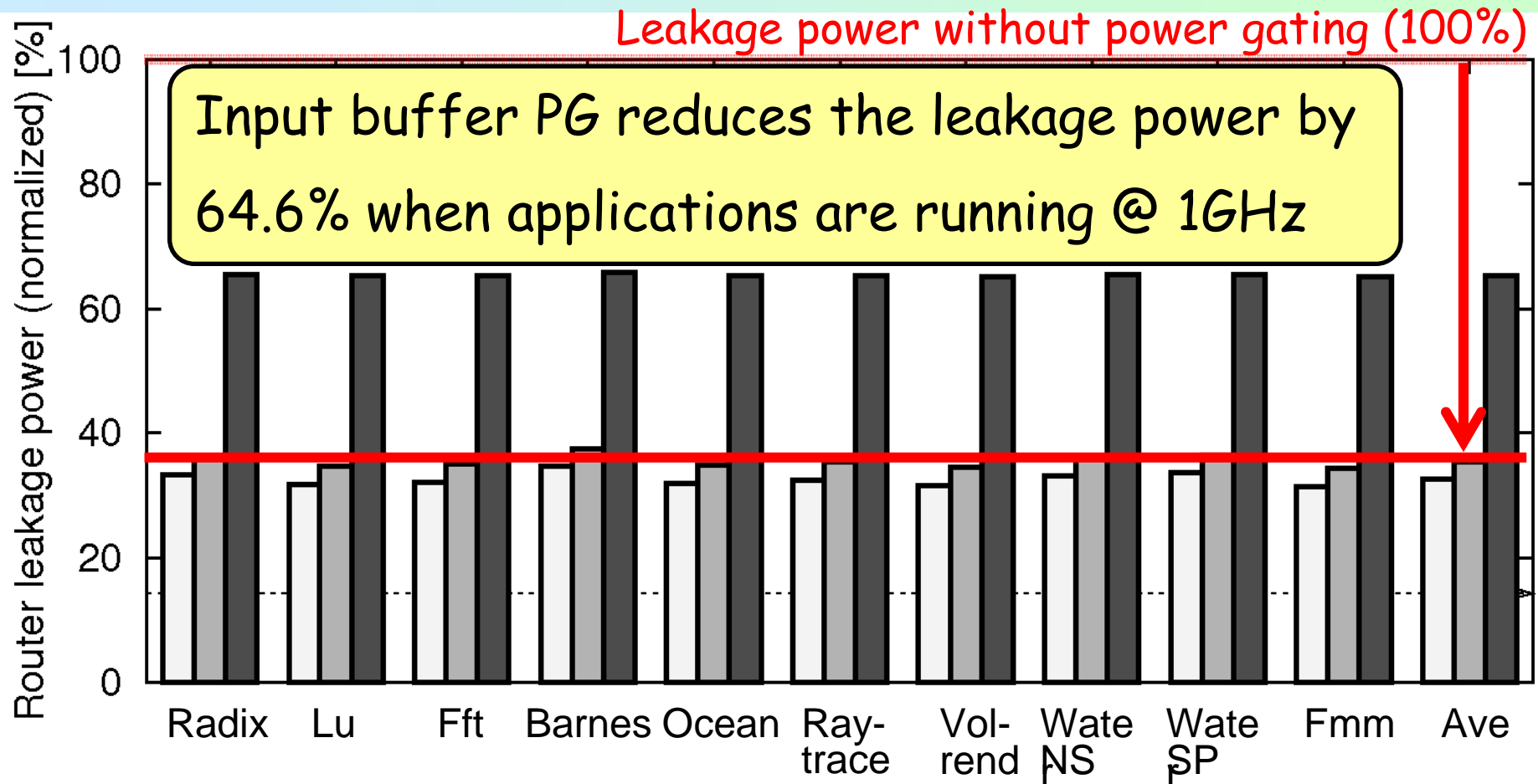


In/output buffer + Xbar

# Evaluations: Leakage power reduction

Level-1 PG: Input buffer only (3-cycle wakeup)

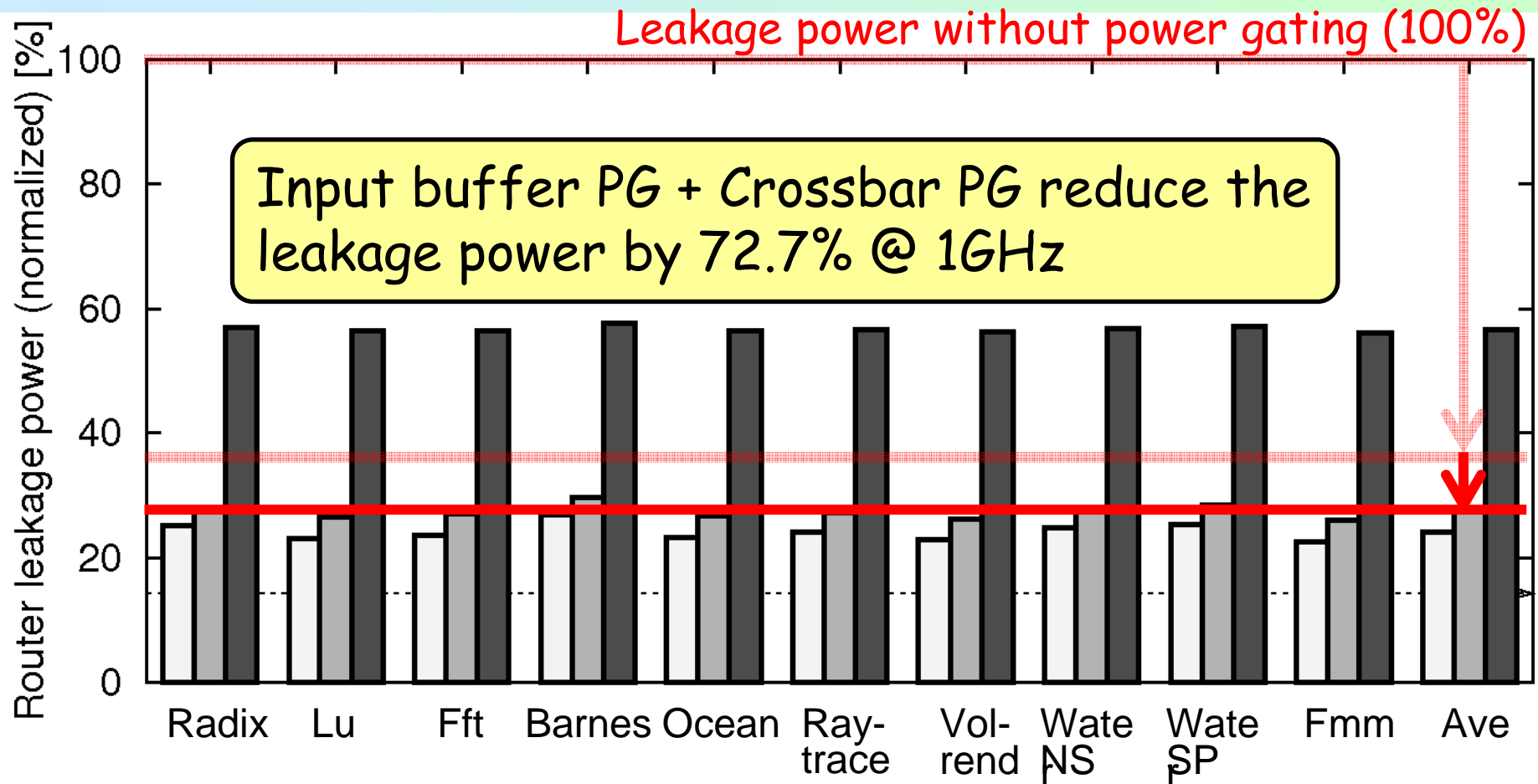
Look-ahead      Look-ahead with CPU ever-on      Buffer window



# Evaluations: Leakage power reduction

Level-2 PG: Input buffer + Crossbar (3-cycle wakeup)

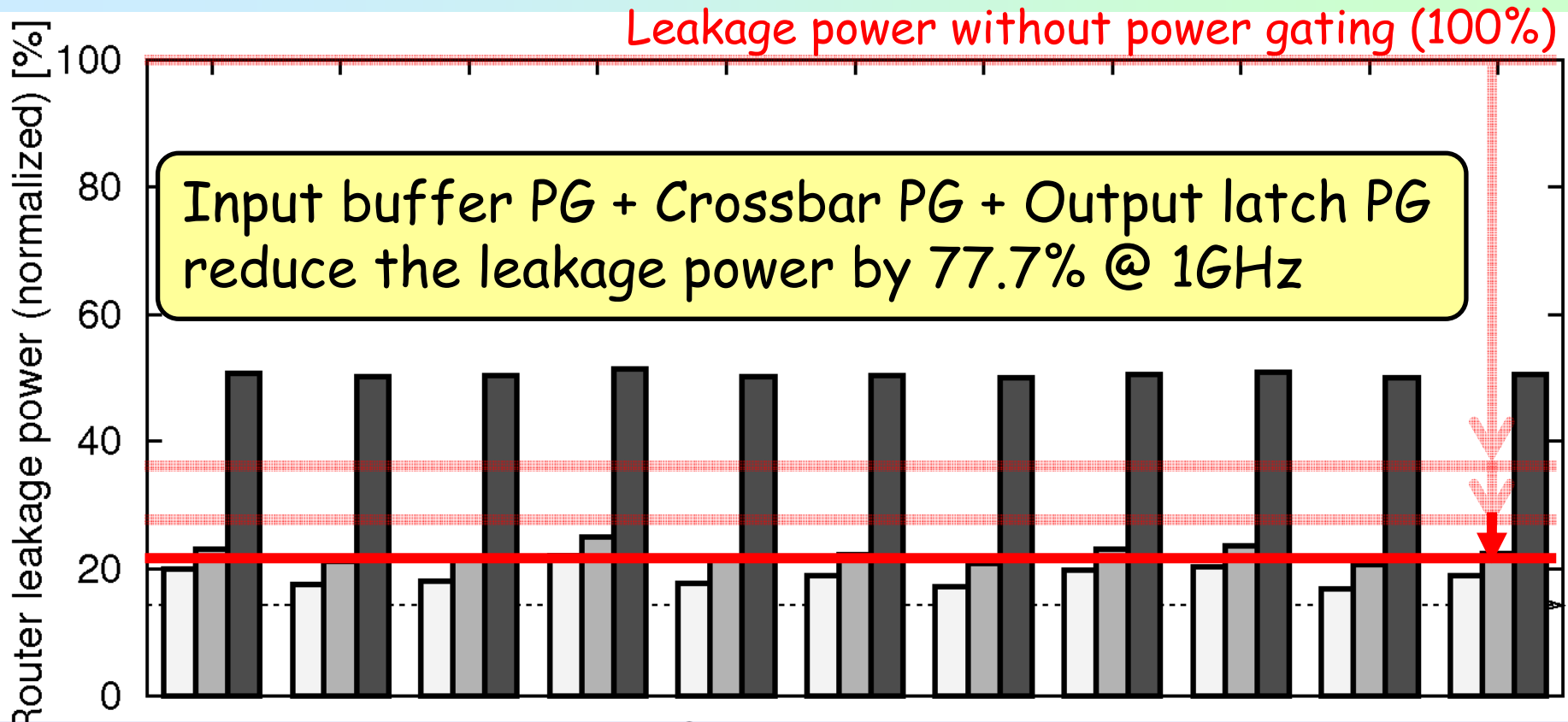
Look-ahead      Look-ahead with CPU ever-on      Buffer window



# Evaluations: Leakage power reduction

Level-3 PG: Input buffer + Crossbar + Output latch

Look-ahead      Look-ahead with CPU ever-on      Buffer window



Fine-grain PG with early wakeup reduces the leakage by 77.7%

# Summary: Run-time PG router for CMPs

- Power gating for router components
  - Input VC buffers
  - Crossbar MUXes, VC MUXes
  - Output latches
  - Wakeup latency is at most 3nsec
- Three early wakeup methods
  - Look-ahead
  - Look-ahead with CPU ever-on
  - Look-ahead with active buffer window
- Evaluation results
  - Performance overhead is less than 4.0% @ 1GHz
  - Leakage power is reduced by 77.7%

} 35 power domains  
in each router

**Thank you for your attention**

It would be very helpful if you would speak slowly. Thank you in advance.