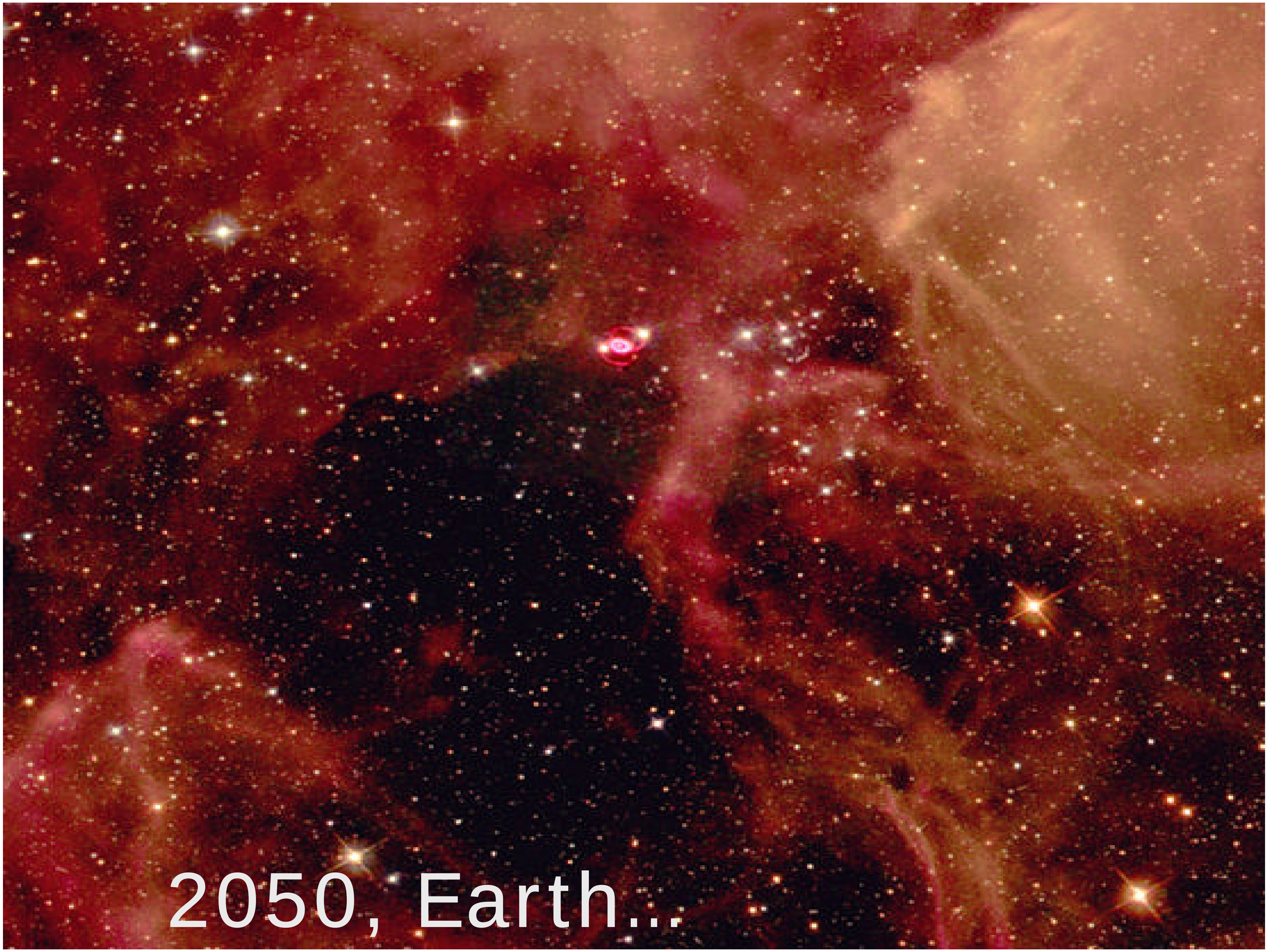

2050, Earth...

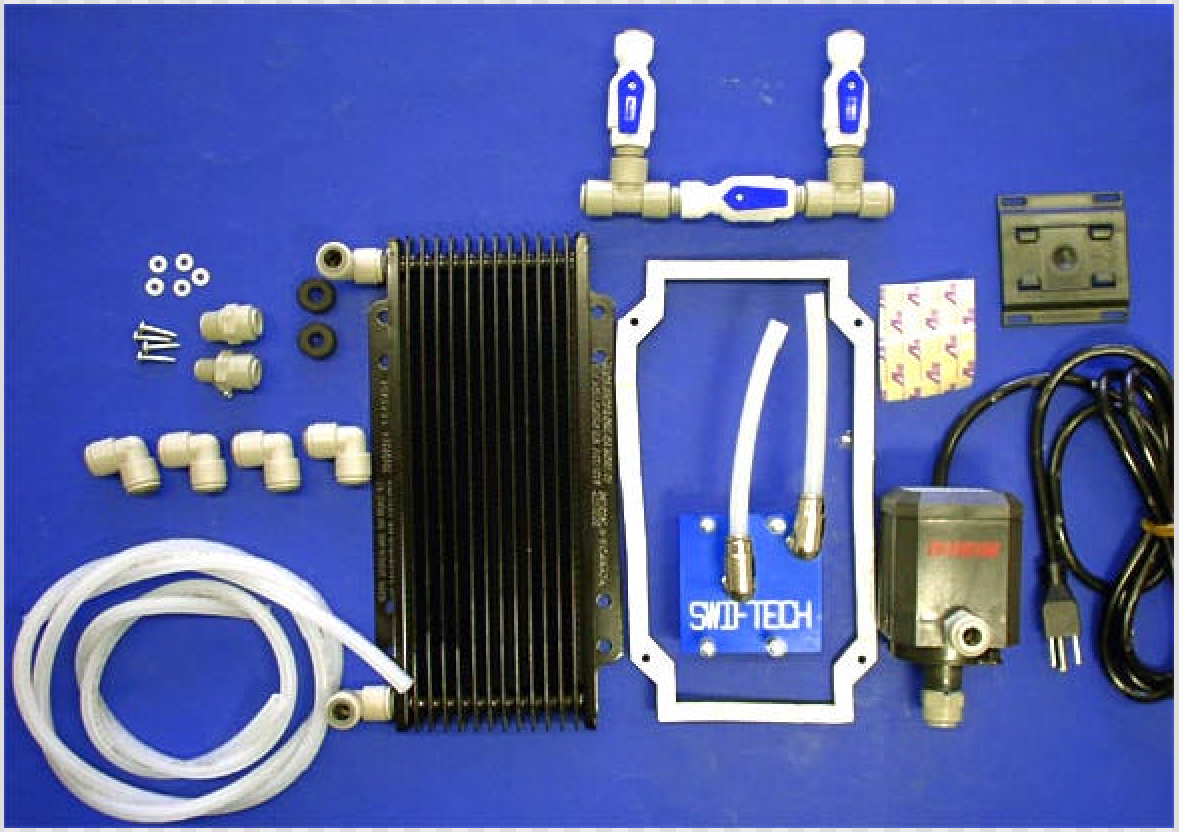
Using Simulator to Analyze Power Consumption

Song Li
Zizhen Yao

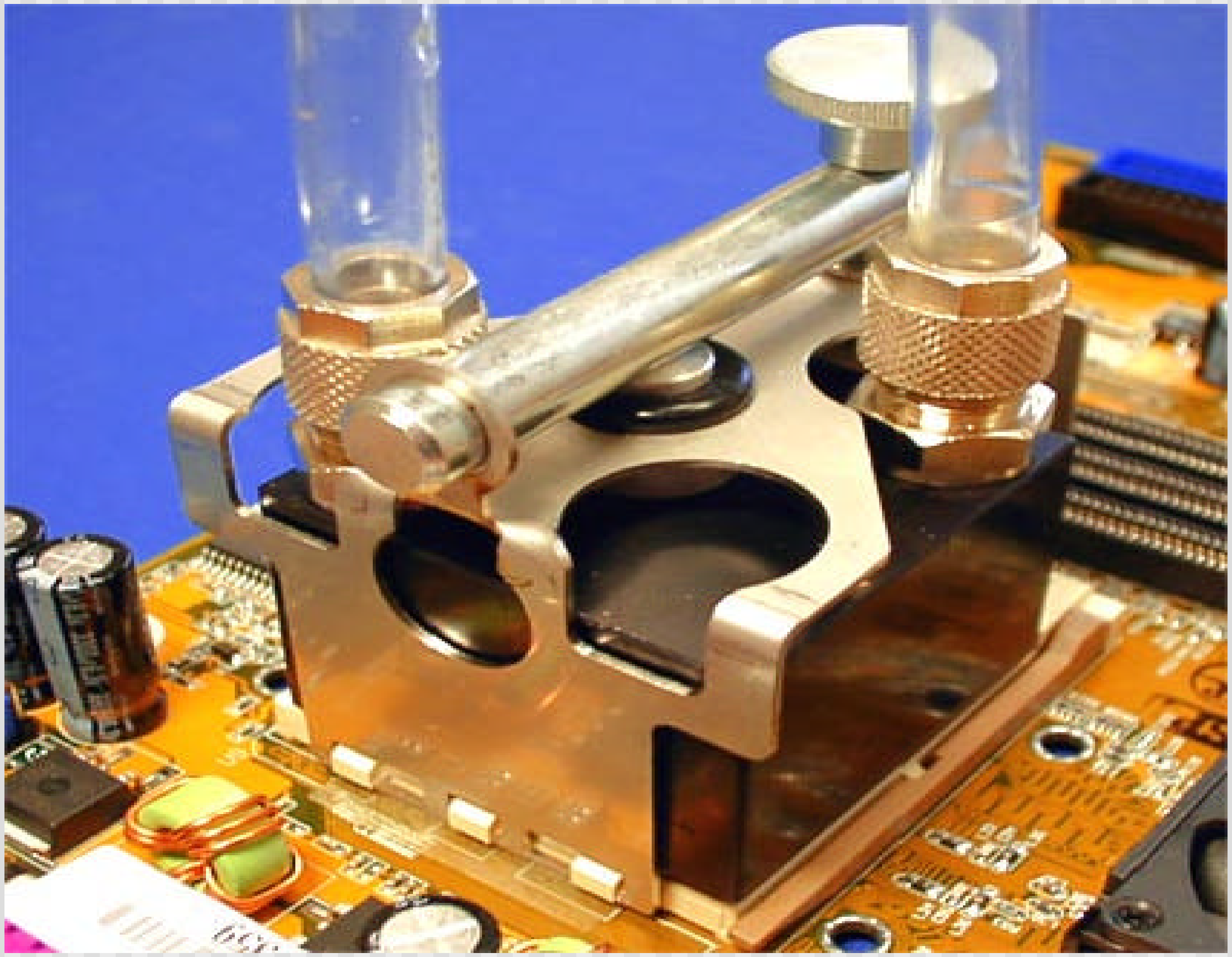
Outline

- Introduction to Power dissipation problems and solutions.
- Power Modeling and Simulation (Previous research)
- Our work
- Future plan

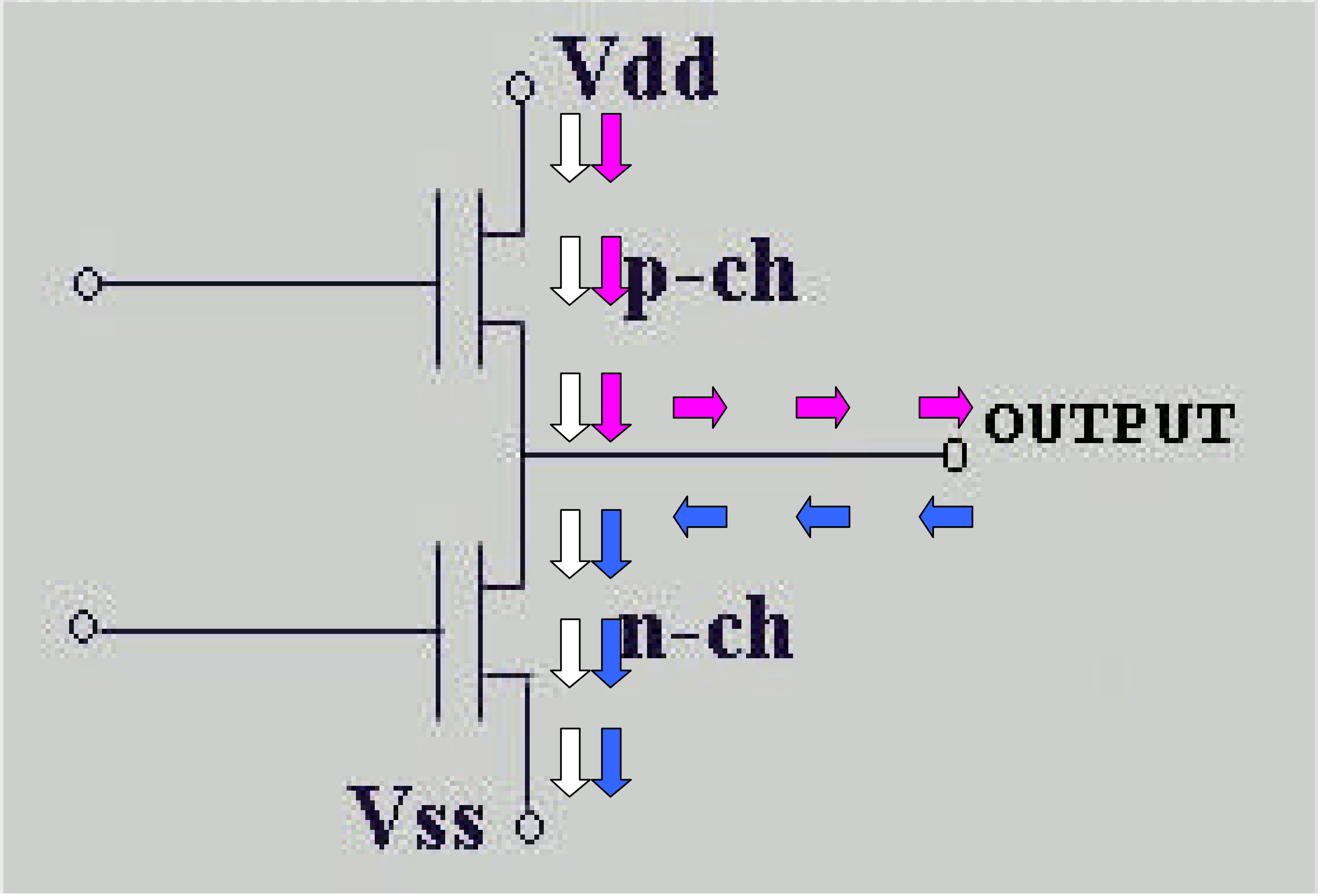
What you need for a cool chip



Cooling techniques



Source of Power Dissipation



Power Saving Techniques

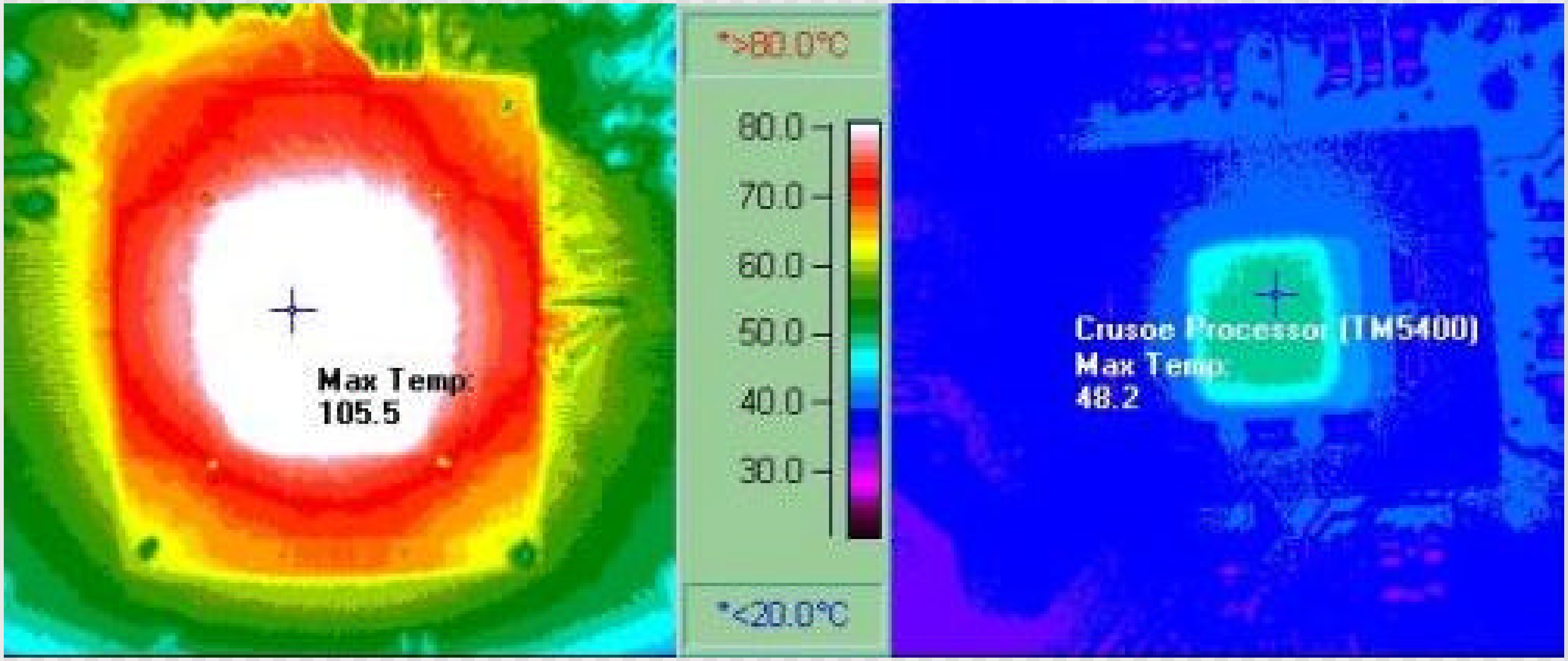
- Voltage Scaling: ultimate limitation
  - The voltage to tell data from noise
  - The threshold of a P-N node: (Si ~ 0.7V, Ge ~ 0.3V)
- Logic:
  - Logic layout optimization – put functional units closer to each other if they need to communicate a lot
- Conditional clocking
  - Certain functional units are idle when not in use
  - i.e. FPU
  - Wakeup time is critical in performance impact

Power Saving Examples

- Alpha:
  - Lower VDD (2.2V for 21264 vs. 3.3 for 21064 and 21164)
  - Conditional clocking
  - Low power bus

Power Saving Examples Cont'd

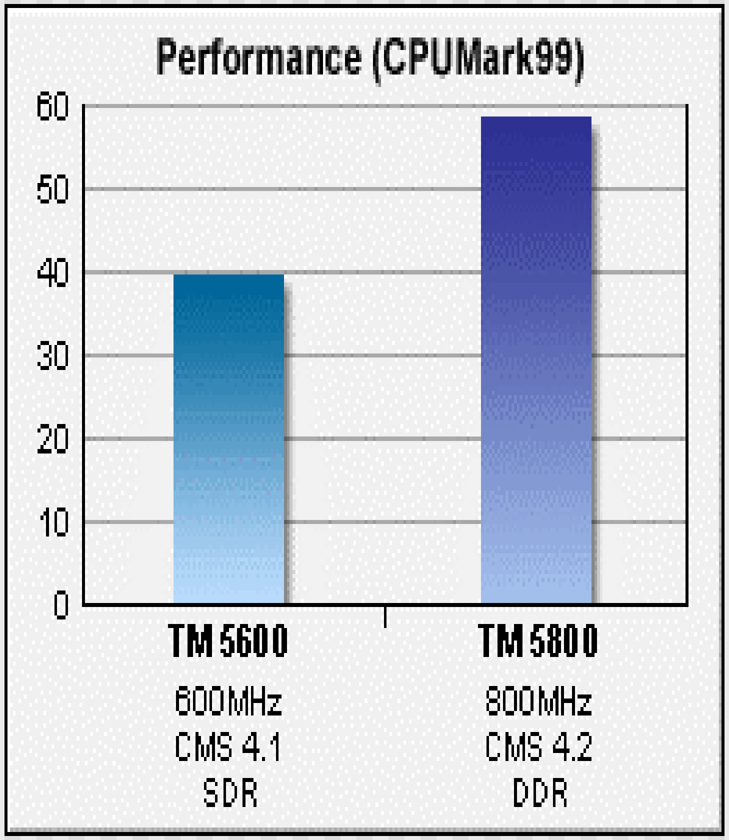
- PIII vs. Crusoe

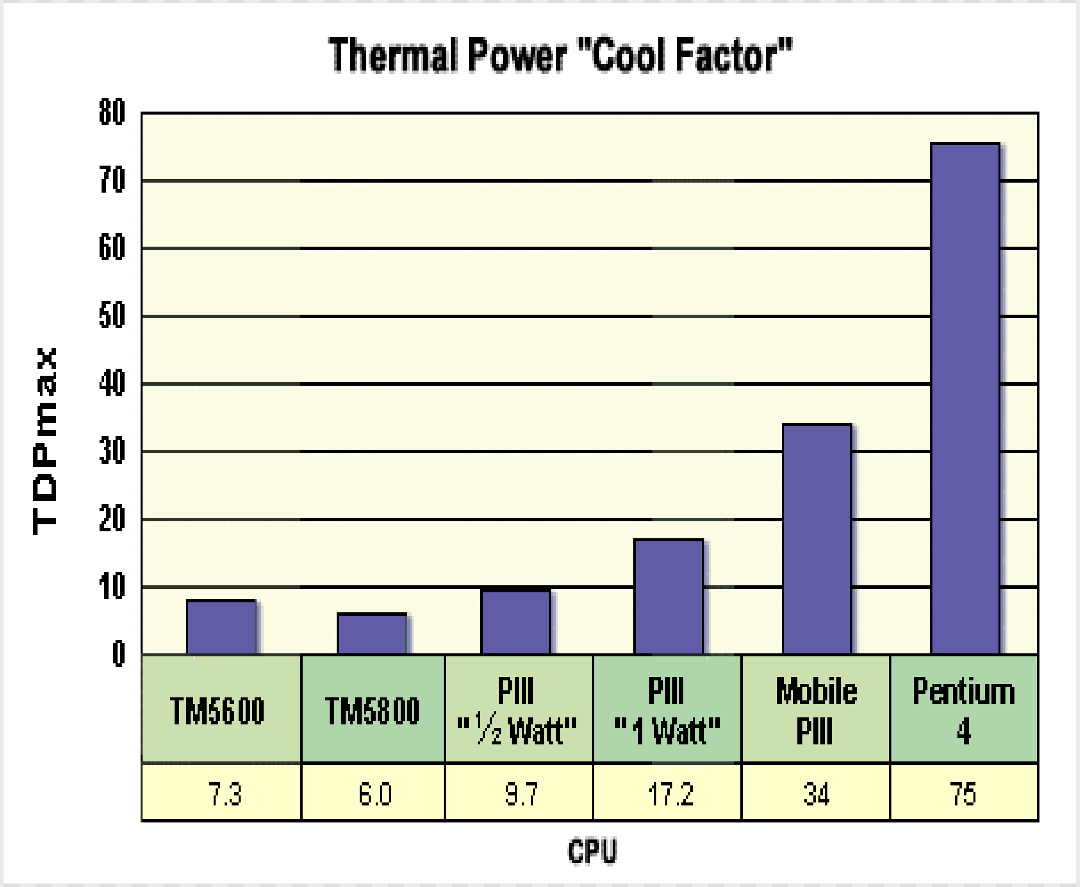


Power Saving Examples Cont’d

- Crusoe:
  - Scalable VDD
  - The power of code morphing
    - No architectural backward-compatibility requirement
    - So that instructions can be optimized to achieve better performance / power ratio
- Die size:
  - Crusoe:TM5600: 88mm^2 TM5800: 55mm^2
  - Alpha: 21164: 299mm^2 21264: 314mm^2

Power Saving Examples Cont’d





TM5600	TM5800	PIII "½ Watt"	PIII "1 Watt"	Mobile PIII	Pentium 4
7.3	6.0	9.7	17.2	34	75

Power Analysis Tool

- Circuit level
  - PowerMill, QuickPower
  - Can be applied only at the late stage of architecture design.
  - Accurate, but expensive.
- Architecture level
  - Wattch
  - Can analyze high level architecture design
  - Tradeoff accuracy for efficiency.

Power Model

- Computation of power

$$P = CV^2 f$$

- Computation of C (capacitance)
  - Diffusion capacitance (transient state).
  - Gate capacitance (gate charge and discharge).
  - Wire capacitance (wire to substrate, wire to wire)

Alpha Pipeline

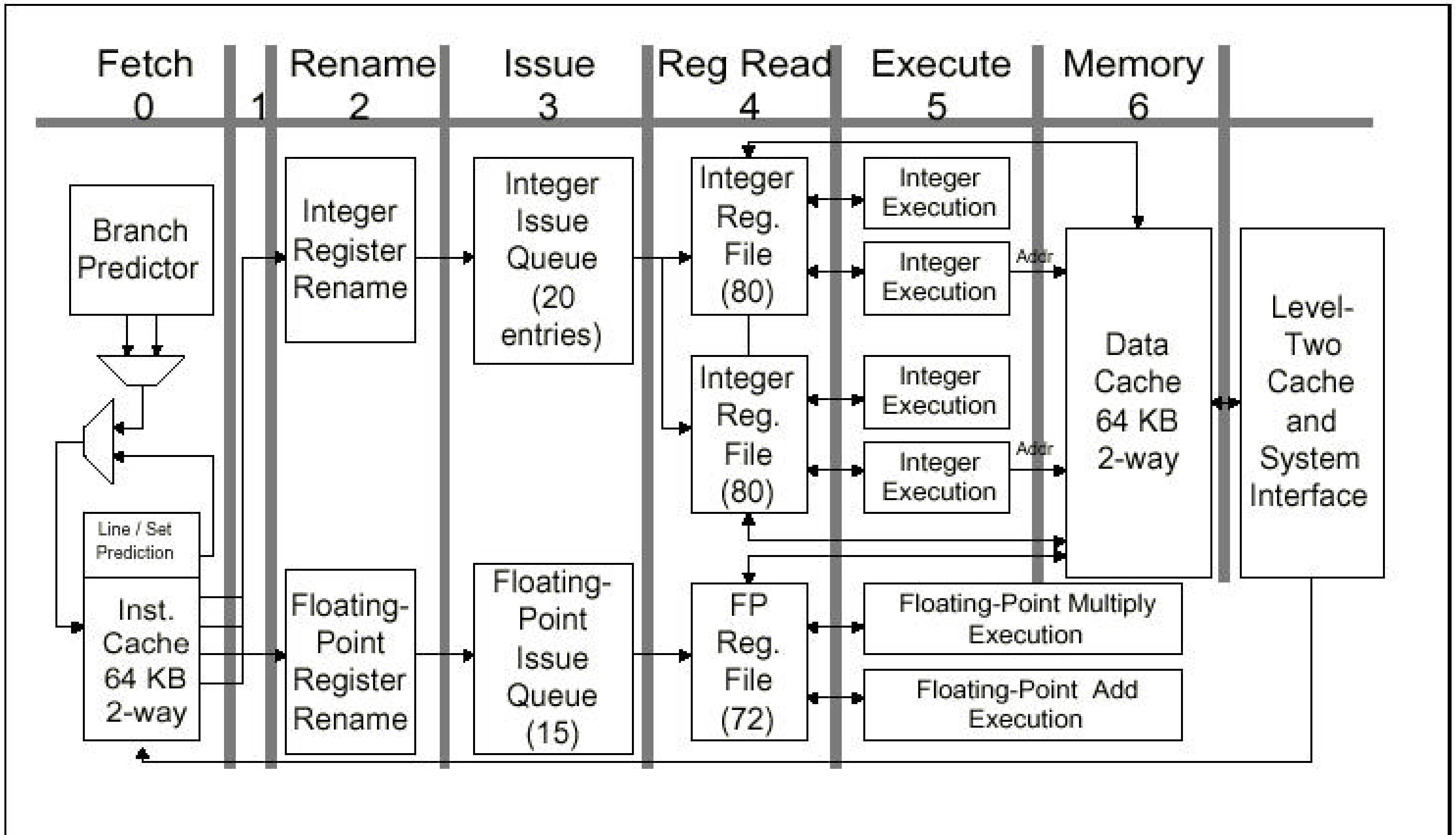


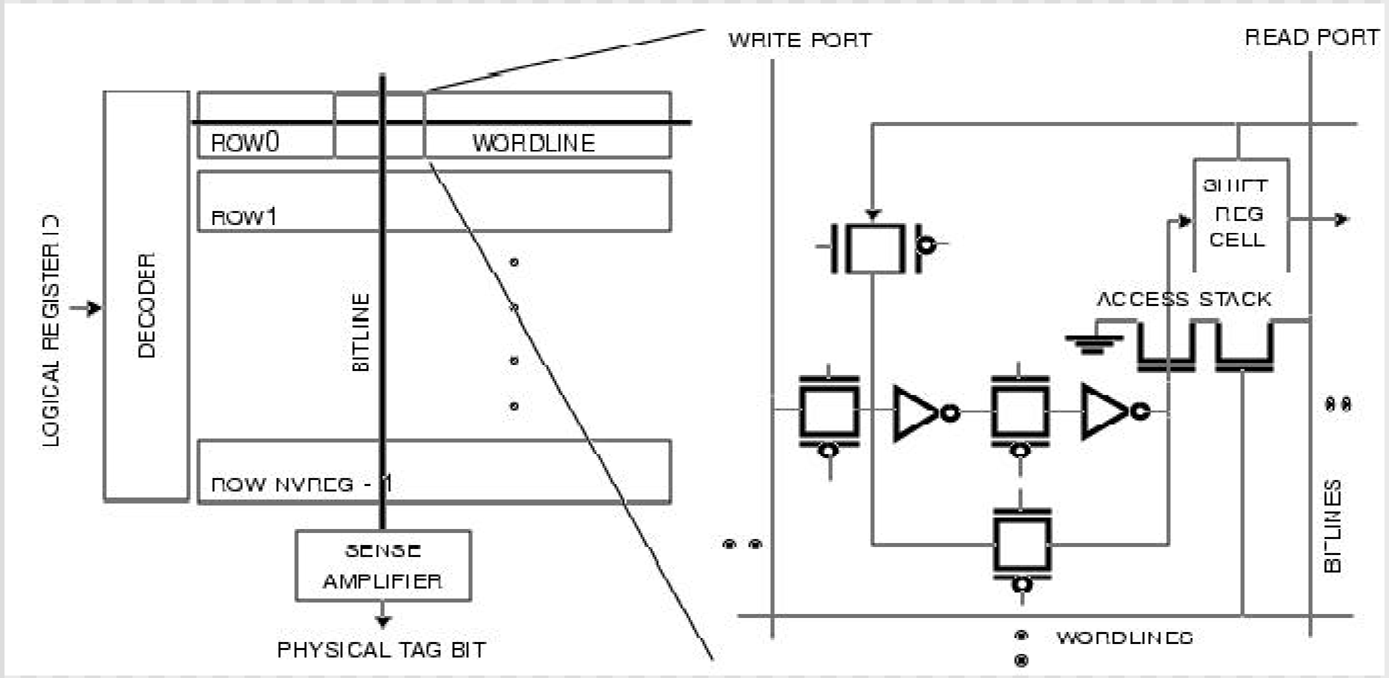
Figure 1 The Basic 21264 Pipeline

Types of logic Structure

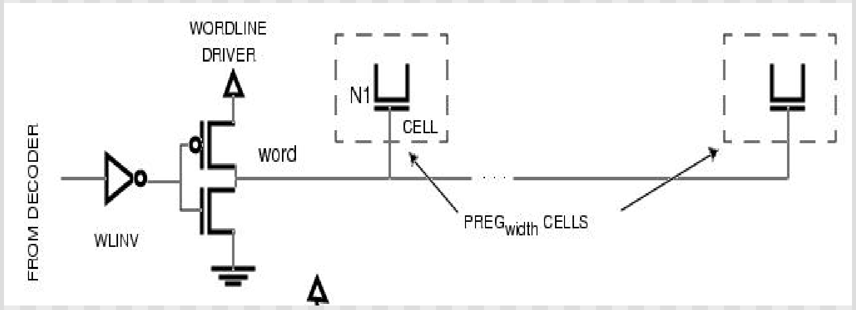
- Array Structure (RAM)
  - Data and instruction caches
  - Register file
  - Branch predictor
  - Renaming table
- Fully Associative Content-Addressable Memories (CAM)
  - Wakeup logic, TLB
  - dependency check
    - Renaming
    - load store queue:
- Combinatoric Logic (ALU) and Clock

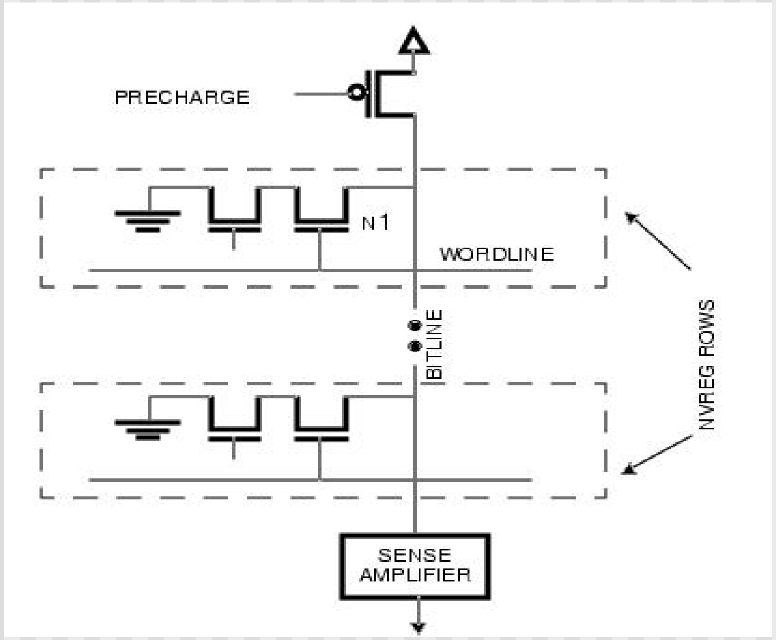
Capacitance Modeling (RAM)

Decoder, wordline drive, bitline discharge, and output drive.



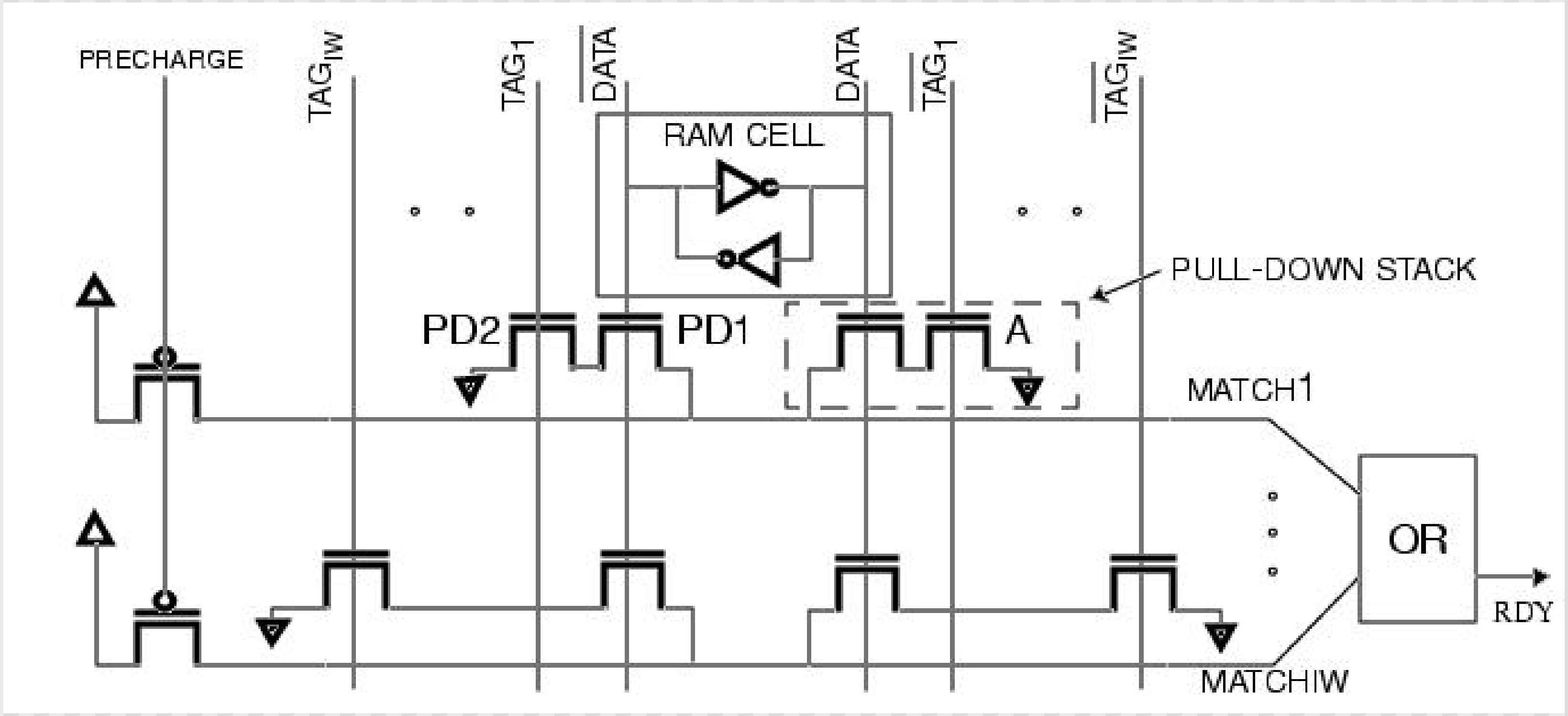
Wordline and Bitline





- Cwordline = C_{diff}(WordlineDriver) + C_{gate}(CellAccess) * NumBitlines + C_{metal} * WordlineLength
- Cbitline = C_{diff}(Precharge) + C_{gate}(CellAccess) * NumWordlines + C_{metal} * BitlineLength

CAM

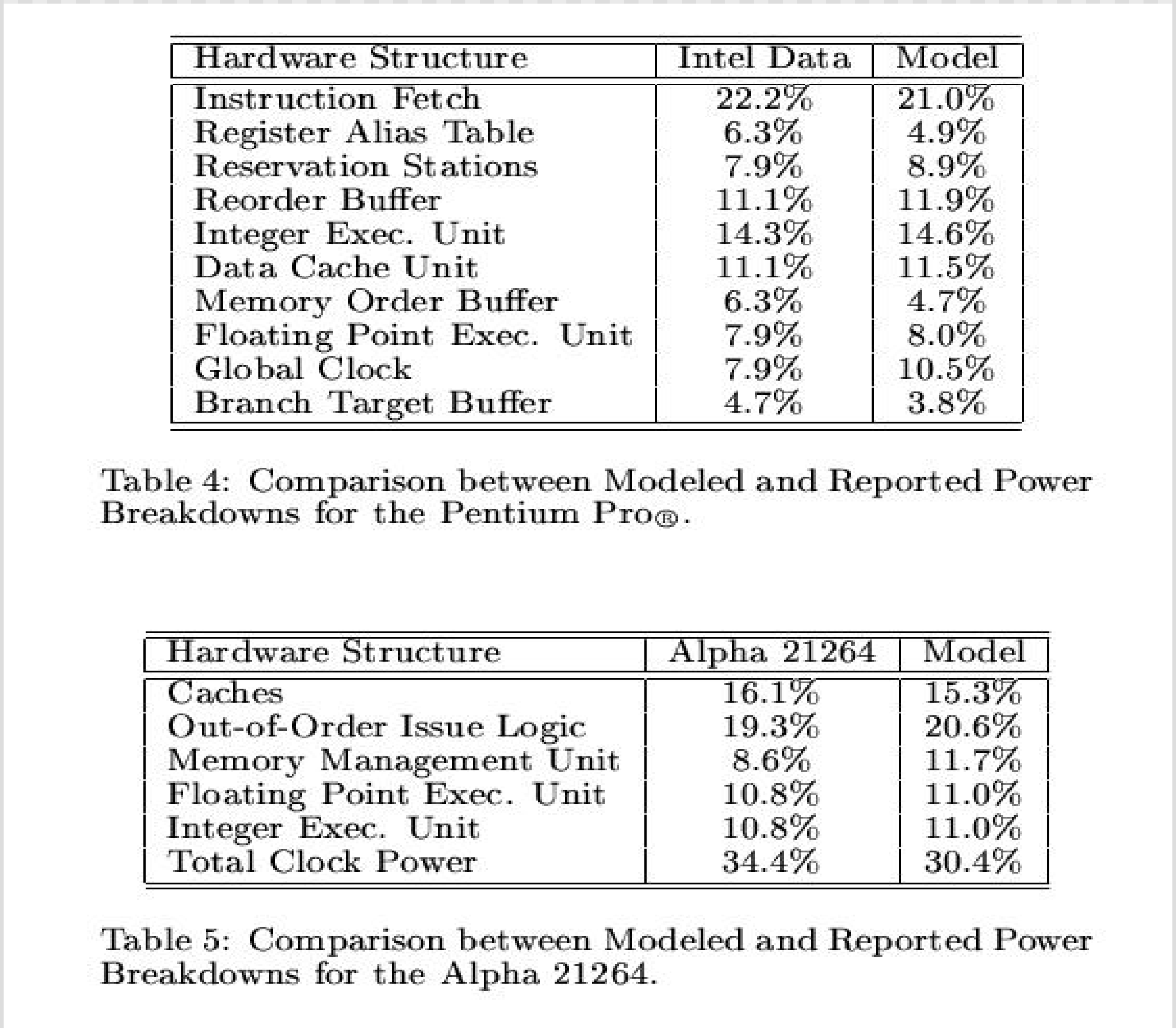


Ctagline= C_{gate}(CompareEn) *NumberTags+ C_{diff}(CompareDriver) + C_{meta}l*TaglineLength
Cmatchline= 2*C_{diff}(CompareEn) *TagSize+ C_{diff}(MatchPreCharge) + C_{diff}(MatchOR)+C_{metl}*MatchlineLength

Wattch

- Using Power model to compute the power consumption at each access of each device
- Extend execution-driven simulator to keep the access counter of each device during execution
- Conditional Clocking is modeled
  1. All or nothing clock gating
  2. Linear clock gating
  3. Linear scale with port or unit usage, plus unused units dissipates 10% of their maximum power.

Verification of Wattch



Hardware Structure	Intel Data	Model
Instruction Fetch	22.2%	21.0%
Register Alias Table	6.3%	4.9%
Reservation Stations	7.9%	8.9%
Reorder Buffer	11.1%	11.9%
Integer Exec. Unit	14.3%	14.6%
Data Cache Unit	11.1%	11.5%
Memory Order Buffer	6.3%	4.7%
Floating Point Exec. Unit	7.9%	8.0%
Global Clock	7.9%	10.5%
Branch Target Buffer	4.7%	3.8%

Table 4: Comparison between Modeled and Reported Power Breakdowns for the Pentium Pro®.

Hardware Structure	Alpha 21264	Model
Caches	16.1%	15.3%
Out-of-Order Issue Logic	19.3%	20.6%
Memory Management Unit	8.6%	11.7%
Floating Point Exec. Unit	10.8%	11.0%
Integer Exec. Unit	10.8%	11.0%
Total Clock Power	34.4%	30.4%

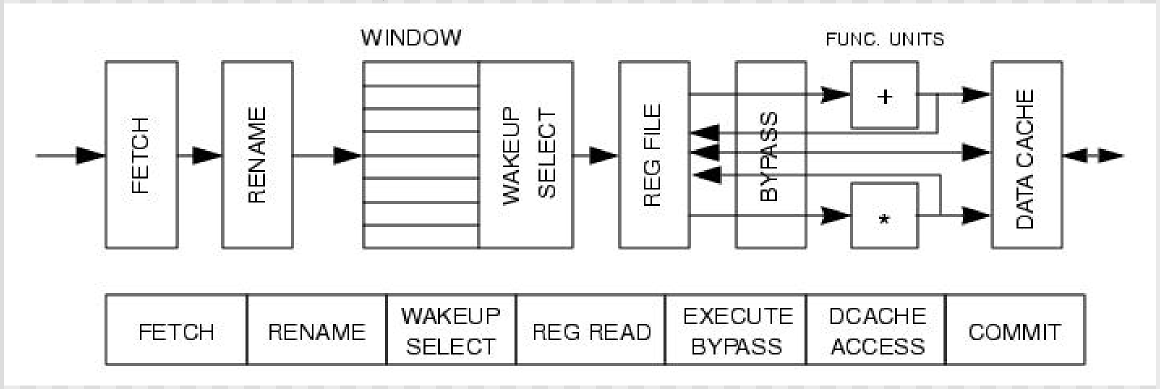
Table 5: Comparison between Modeled and Reported Power Breakdowns for the Alpha 21264.

Our work

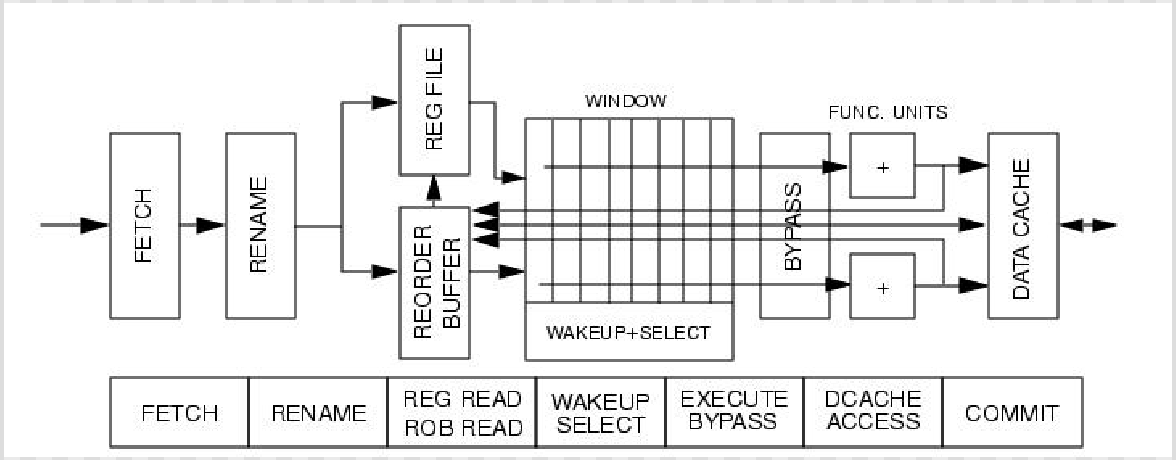
- Port Wattch to Sim-Alpha
- Use Wattch-Alpha to analyze power consumption of architecture designs

Diffence between SimpleScalar and Sim-Alpha (1)

Sim-Alpha (MIPS R10000, DEC 21264)



SimpleScalar (Intel Pentium Pro, PowerPC)



Diffence between SimpleScalar and Sim-Alpha (2)

- There is no slot stage in SimpleScalar
- Instruction fetch
  - Sim-alpha: fetches by block
  - SimpleScalar: fetches each instruction.
- Line and set prediction.
  - Each fetch block of four instructions includes a line and set prediction, which indicates where is the next block of instructions.
  - Combines the speed advantage of a direct-mapped cache with the lower miss ratio of a two-way set-associative cache.
  - Only simulated by Sim-alpha simulates this design.
- Memory system
  - Bus and MSHR are not simulated in SimpleScalar.

Our Work: different from Wattch

- Icache access:
  - In Wattch Icache access is per instruction
  - In SimAlpha Icache access is per block
- ALU access
  - More instructions in Alpha's ISA
  - i.e. ITOF: IALU
  - i.e. Addressing mode: DISP, RR
- Rename access
  - Wattch only considers the power for reservation station allocation.
  - Rename table-lookup for operands is neglected.

Experiments to perform

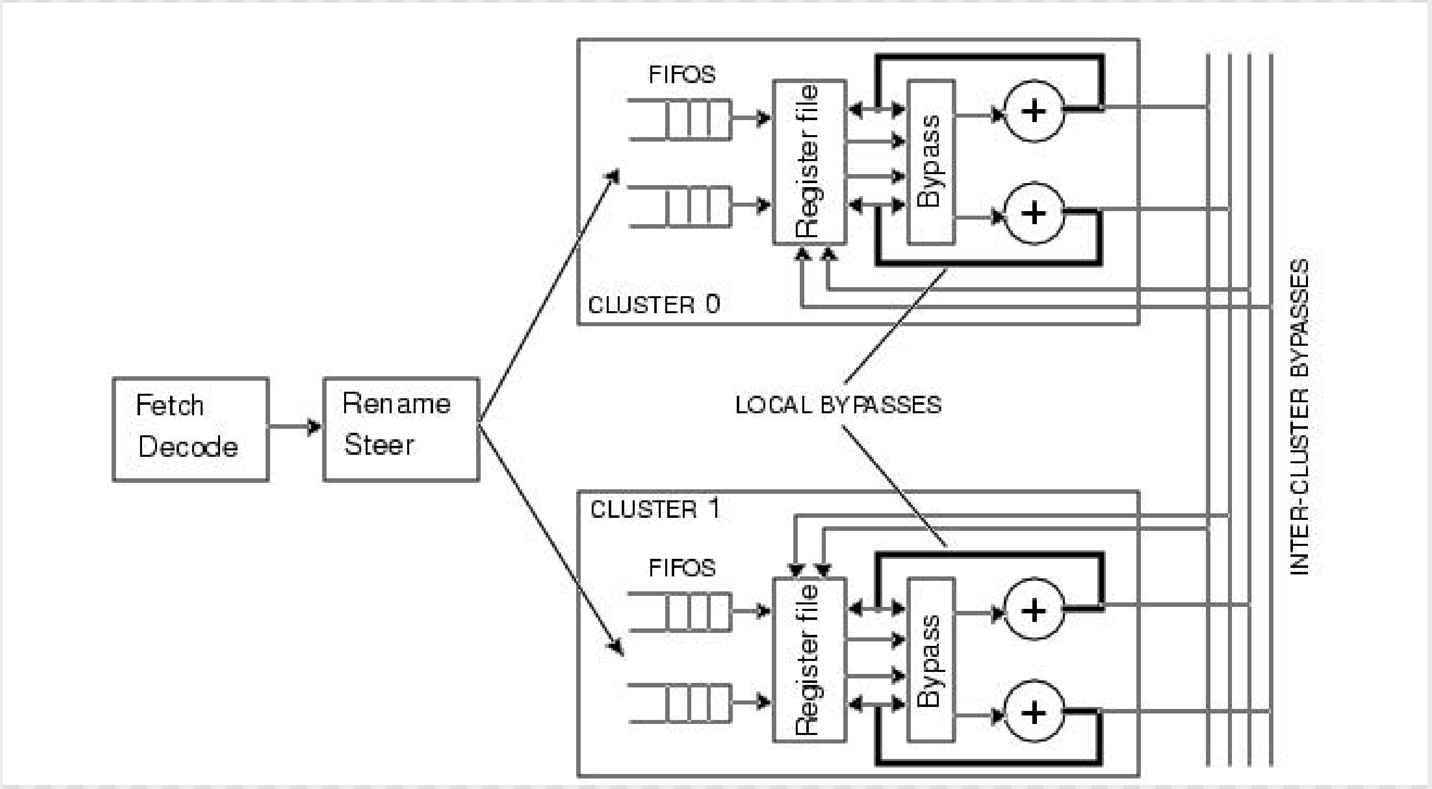
- How is the associativity of CAM device effect the performance and power consumption
  - Issue Queue
  - Set associative cache
- Microarchitecture that reduce the associativity
  - Fifo-based design for issue queue

Fifo-based microarchitecture

- Motivation
  - Decrease associativity of logic device without performance degradation.
  - Exploit the natural dependences among the instructions.
- The issue window is replaced by several fifo queues. Each queue
  - Contains dependent instructions
  - Inorder issue
- Selection instructions from head of queues.
- Result broadcast to head of queues

Fifo-based microarchitecture

Further Reduction of associativity by clustering


FIFOS
Register file
Bypass
+
CLUSTER 0
Fetch Decode
Rename Steer
LOCAL BYPASSES
CLUSTER 1
FIFOS
Register file
Bypass
+
INTER-CLUSTER BYPASSES