AN ABSTRACT OF THE THESIS OF

Hsien-Yen Lee for the degree of Master of Science in Electrical and Computer Engineering presented on August 22, 2002.

Title: Implementation and Comparison of Two Wakeup Logic for Out-of-order Superscalar Microprocessors

Abstract approved: Redacted for privacy

Shih-Lien Lu

The wakeup logic in out-of-order superscalar microprocessors is responsible for resolving the data dependency hazard between instructions. Its performance is critical because it may prevent the processor to have deeper pipelines or to achieve the highest IPC (Instructions Per Cycle) possible.

In this thesis, we implemented the circuit and layout for two types of wakeup logic (CAM-type and RAM-type) used in the modern microprocessors. These two implementations are simulated extensively using a circuit level simulator – HSPICE, with full parasitic loads. We, then, made comparison between the CAM-type and RAM-type wakeup circuits.

From the simulation results, the CAM-type wakeup logic has a better performance than the RAM-type wakeup logic if a larger number of physical registers is employed by the processor. The performance impacts caused by varying the other superscalar design parameters, such as instruction window size and issue width, are not much different for both types of wakeup logic implementations.

Implementation and Comparison of Two Wakeup Logic for Out-of-order Superscalar Microprocessors

by

Hsien-Yen Lee

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements of the
degree of

Master of Science

Presented August 22, 2002
Commencement June 2003

Master of Science thesis of Hsien-Yen Lee presented on August 22, 2002.

APPROVED:

Redacted for privacy

Major Professor, representing Electrical and Computer Engineering

Redacted for privacy

Head of the Department of Electrical and Computer Engineering

Redacted for privacy

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for privacy

Hsien-Yen Lee, Author

ACKNOWLEDGMENTS

I would like to express sincere appreciation to Professor Lu for his assistance in the completion of the thesis. I also would like to thank my wife, Hsing-Ying, for companying me walking through many hard times. In addition, thank to my daughter, Iling, who brought me many joys when I was down.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

Figure Page

LIST OF APPENDIX TABLES

IMPLEMENTATION AND COMPARISON OF TWO WAKEUP LOGIC FOR OUT-OF-ORDER SUPERSCALAR MICROPROCESSORS

1 INTRODUCTION

1.1 MOTIVATION

Microprocessor performance has increased a hundredfold over the past decade. Much of the increase can be attributed to the utilization of pipelining and superscalar parallel execution of instructions mechanisms. Wider pipelines exploit the instruction parallelism to achieve higher IPC (Instruction Per Cycle). Deeper pipelines temporally provide better throughput by exploiting parallelism. Over the past decade, the number of pipeline stages has grown from 5 (Intel 486) to 20 (Intel Pentium 4) [1,2]. This growth trend will continue because it a good approach to exploit more instruction parallelism. However, any critical logic that may be a barrier to the deeper pipeline must be reviewed to determine if it meets the timing requirement (setup time and hold time) for a certain clock frequency. One of the critical logic is the instruction scheduling logic, also named issue logic, which resides in the issue window of superscalar microprocessors. Issue logic, which is composed of wakeup logic and select logic, is responsible for resolving the true data dependence between the instructions in the instruction queue, and issuing the instructions to execution units. Wakeup logic will mark a queued instruction ready if the required operand values for this instruction

are available. Select logic picks ready instructions for execution by evaluating the availability of functional units. Palacharla, Jouppi and Smith [3] have mentioned: "Wakeup and select together constitute what appears to be an atomic operation. That is, if they are divided into multiple pipeline stages, dependent instructions cannot issue in consecutive cycles." This means that if dependent instructions are going to be executed in consecutive cycles, issue logic must be performed within one cycle time.

From the discussion above we know that it's extremely important to design a high-performance issue logic if we want to obtain the highest IPC. The performance of wakeup logic or select logic is critical. There are two different implementations of wakeup logic in the modern microprocessors [4,5,6,7]. One is CAM-type wakeup logic and the other is RAM-type wakeup logic. Palacharla [4] has studied the performance of the CAM-type wakeup logic used in the baseline superscalar model processors such as the MIPS R10000 and Alpha 21264 processors. Up to now, there is still no published study evaluating the performance of the RAM-type wakeup logic used in the reservation station-based model processors such as the Intel P6 family processors. Therefore, we couldn't compare the advantages and drawbacks for these two types of wakeup logic. With this motivation, we implemented both types of wakeup logic and evaluated their performances. We believed we could get a better picture about the pros and cons of the CAM-type and RAM-type wakeup logic from this study. The select logic is not in the scope of our study.

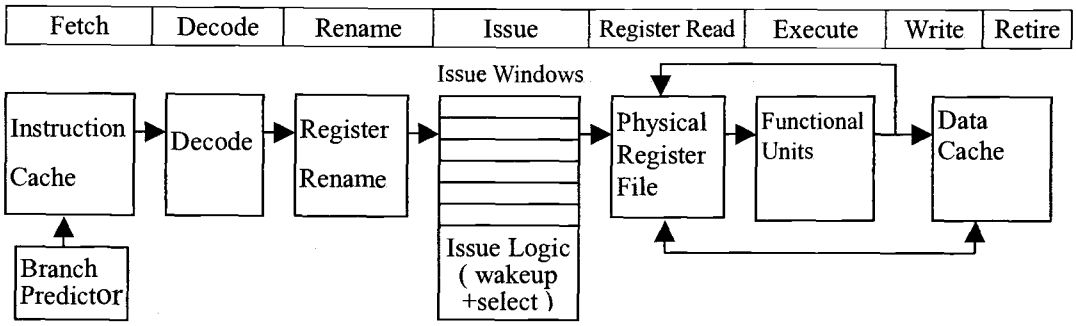


Figure 1.1 The baseline superscalar model

1.2 WAKEUP LOGIC IN OUT-OF-ORDER SUPERSCALAR MICROPROCESSORS

As mentioned in the previous section, there are two types of wakeup logic employed by modern microprocessors. The first one is CAM-type wakeup logic used in the baseline superscalar model processors and the second one is RAM-type wakeup logic used in the reservation station-based model processors. We will discuss these two types of wakeup logic separately in the following subsections.

1.2.1 The Baseline Superscalar Model and CAM-Type Wakeup Logic

Figure 1.1 illustrates the baseline superscalar model and its simplified pipeline stages for the MIPS R10000/R12000 and Alpha 21264 microprocessors [4,8,9,10]. The pipeline begins with the fetch stage that fetches up to 4 instructions from the instruction cache each cycle with the help of the branch predictor. These fetched instructions are then decoded and their operands (or architecture registers) are renamed to appropriate physical registers for the purpose of resolving register hazards, i.e. WAW and WAR hazards [10]. After the rename stage, the instructions are sent to the instruction queue in the issue window where

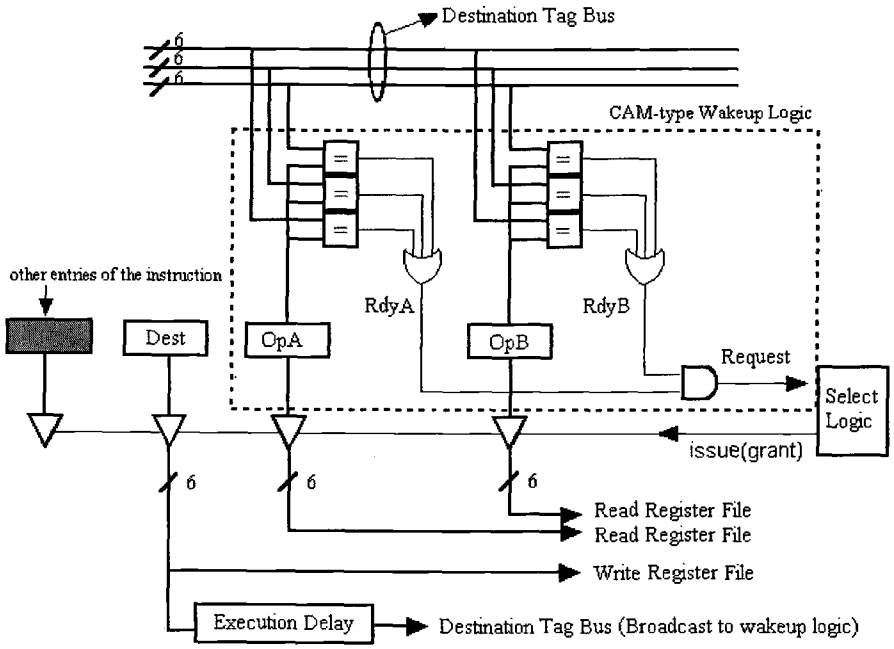


Figure 1.2 CAM-type wakeup logic for one instruction in the issue window

the instructions are waiting for their operands and functional units to become available. When a queued instruction in the issue window is dispatched for execution, its operand values are fetched either from the physical register file or are bypassed from the execution results of the previous instructions still in the pipeline. The next pipeline stage is write for load or store operation. The final pipeline stage is retire, which manages the retirement order of the executed instructions.

The issue logic in the issue window is responsible for checking the availability of the instruction operands and functional units as well as issuing ready instructions for execution. The issue logic has two components- wakeup logic and select logic. Figure 1.2 illustrates the relationship between wakeup logic, select logic and one queued instruction in the issue window [9]. In this example, the number of physical registers is 64 so the register (or

operand) tag width is 6 bits. As the issue width is three, up to three queued instructions could be assigned to functional units and up to three destination register tags could be updated each cycle. The wakeup logic keeps monitoring if both the required operand values, OpA and OpB, are available by comparing the operand tag with the destination register tags each cycle. If one operand tag matches any destination tag, then this operand value is available for execution. When both the required operands are available, i.e. RdyA and RdyB signals are high, the signal Request is asserted and this instruction is ready for execution. After the wakeup logic has marked all the instructions either ready or not ready to execute, the select logic dispatches up to three ready instructions to functional units according to the availability of functional units. When a queued instruction is issued, the destination tag of this instruction will update the destination tag bus for broadcasting to all the instructions waiting for their source operands to become available.

This type of wakeup logic we discussed above is named the CAM-type wakeup logic as its circuitry is mainly composed of CAMs (Content-Addressable Memory [11]). We will discuss the circuit implementation of the CAM-type wakeup logic in chapter 3.

1.2.2 The Reservation Station-based Superscalar Model and RAM-type Wakeup Logic

Figure 1.3 illustrates the reservation station-based superscalar model and its simplified pipeline stages for the Intel P6 family or PowerPC604 [4,12,13] microprocessors. After the fetch and decode stages, the instructions pass through the ROB (Reorder Buffer [14]). The ROB is actually a physical register file with some functional extension. It can log each

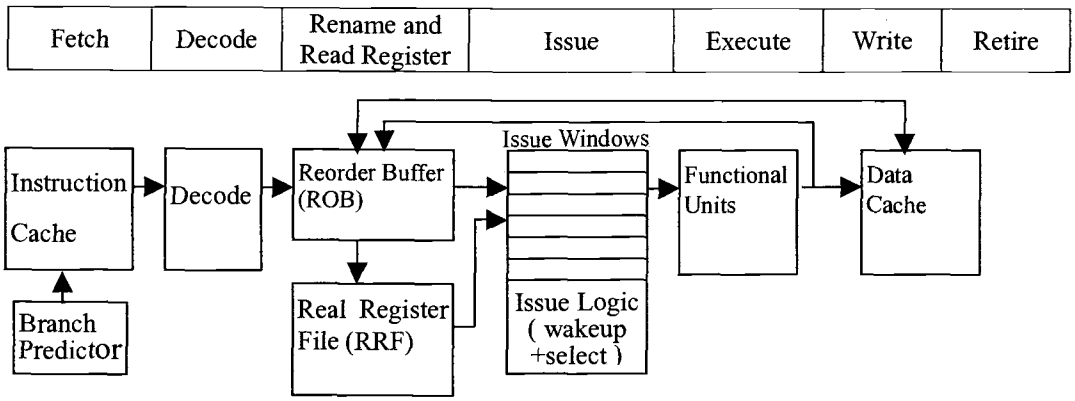


Figure 1.3 The reservation station–based superscalar model

instruction so each instruction can later retire in program order. It has the space to store the result of a load operation or the value of the destination operand after an instruction was executed. The destination value in one ROB entry is then written to the RRF (Real Register File) only when this instruction is retired. Up to three instructions (called uops in the Intel P6 family) can be renamed and logged into the ROB each cycle, then flow into the reservation station. The instruction waiting in the reservation station can hold the values of source operands that come either from the ROB or RRF. The former case happens only if the destination register of a previous instruction in the ROB is identical to any source register of this instruction, i.e. the true data dependence occurred. As soon as the value of the destination operand for an instruction is calculated, it's written into the ROB. When this instruction is retired, the destination operand value in the ROB is then written into the RRF and the related ROB entry is cleaned. The tasks of the wakeup and select logic in this model are exactly the same as those described in the baseline superscalar model.

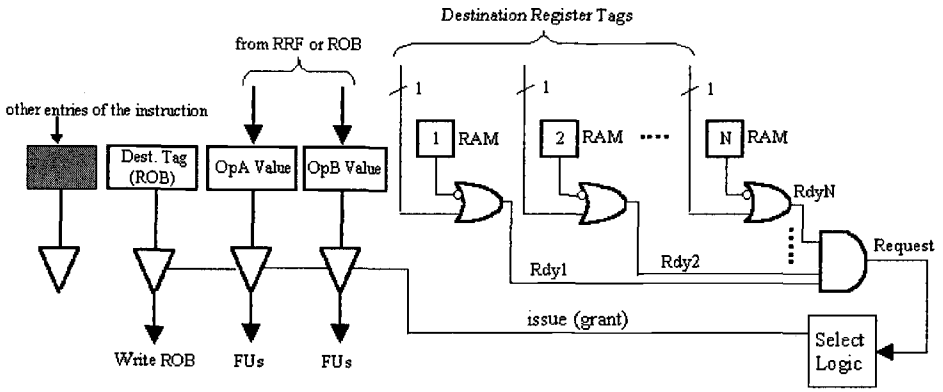


Figure 1.4 RAM-type wakeup logic for one instruction in the reservation station

Figure 1.4 illustrates the relationship between the RAM-type wakeup logic, select logic and one queued instruction in the reservation station [6,7]. In this example, there are N entries in the ROB as well as N 1-bit RAMs in one entry of the wakeup logic. If an instruction doesn't need any source operand from the ROB, then all the N 1-bit RAMs will be set to 0 and all the source operand values will come from the RRF. If an instruction needs any source operand from the ROB, i.e. a data dependency occurred, then the corresponding 1-bit RAM(s) will be set to 1 and the wakeup logic will keep watching if the destination value in this particular ROB entry has become available. Because the maximum number of source operands is two, at most two 1-bit RAMs will be set to 1. Therefore, at most two signals from Rdy1 to RdyN are likely low. If all the required operands are available, then all the signals from Rdy1 to RdyN will be high. Thus, the signal Request is asserted and this instruction is marked ready for execution.

This type of wakeup logic in the reservation station-based model is named the RAM-type wakeup logic as its circuitry is mainly composed of several 1-bit RAM arrays. We will discuss the circuit implementation of the RAM-type wakeup logic in chapter 3.

1.3 METHODOLOGY

This thesis is focused on the circuit and layout implementations of the CAM-type and RAM-type wakeup logic. The basic circuit design concept of the CAM-type wakeup logic was taken from some published studies showing the most promising implementation [4]. For the RAM-type wakeup logic, there is still no study showing the circuit implementation. Therefore, we utilized the design concept of high-speed CMOS circuits to implement the circuit for the RAM-type wakeup logic. In our study, TSMC CMOS 0.18μm and 0.25μm technologies [15] were chosen to design all the circuits. We set all the transistor channel lengths (L) to the minimum channel length for minimizing the silicon area. The main concern of wakeup logic is performance so all the circuits are implemented and optimized for speed. We used both static logic and dynamic logic in our design. Dynamic logic was used when the circuit performance was critical to the wakeup logic delay. Static logic was used for realizing some simple logic gates such as 2-input AND and 2-input NOR gates.

We did an accurate layout for each component in our circuit design so we could estimate the more reliable parasitic capacitance/resistance and transistor dimensions for the netlists used by the HSPICE simulator. MOSIS SCMOS design rules [16] for TSMC CMOS 0.18μm and 0.25μm processes were used for layout DRC (Design Rule Check).

HSPICE is the simulator we used for performance simulation. The BSIM3 HSPICE models for TSMC CMOS 0.18μm and 0.25μm technologies were also obtained from MOSIS organization. The RC values used in the appropriate nestlist nodes were calculated from the layouts with the help of TSMC parametric test results. The technology files mentioned above are listed in the Appendix for reference.

After the simulations were done, we analyzed and compared the performance results for both the CAM-type and RAM-type wakeup logic to find out their advantages and drawbacks.

1.4 THESIS OVERVIEW

In chapter 2, we briefly discussed the design concepts of high-speed CMOS digital circuits. Chapter 3 introduced the circuit implementation and layout for the CAM-type and RAM-type wakeup logic. Chapter 4 showed the simulation results and performance comparisons of these two types of wakeup circuits. Finally, in chapter 5, we draw conclusions regarding the advantages and drawbacks for these two types of wakeup logic.

2 CONCEPTS OF DIGITAL CIRCUIT DESIGN

In this chapter, we'll introduce some design concepts of CMOS digital integrated circuits used in the wakeup logic design. These concepts that include CMOS static/dynamic logic circuits, interconnect and layout will be discussed in the following sections.

2.1 STATIC CMOS LOGIC CIRCUITS

The static CMOS logic family is still the dominant circuit approach to realizing most logical functions for the following major reasons: noise immunity, low power consumption, ease of synthesis and uniform delay time [17,18]. However, its drawback compared with the dynamic logic circuit is the performance and the silicon area it occupies. The static CMOS logic circuit needs more PMOS transistors than the dynamic logic circuit does as the complexity of logical function increases. PMOS transistor consumes a lot of silicon area and introduces more capacitance to the loading points. Therefore, the switch speed of the static CMOS logic circuit is slower than that of the dynamic logic circuit for realizing the same logical function. For this reason, dynamic logic circuits are often used in the critical paths of microprocessor circuits. Static logic circuits are usually used in designs in which performance is not critical or the power consumption is important.

Consider the static 4-input NOR gate shown in figure 2.1 which realizes the Boolean function Z= not (A+B+C+D), the 4 PMOS transistors above node Z represent the PMOS pull-up network and the other 4 NOMS transistors represent the NMOS pull-down network.

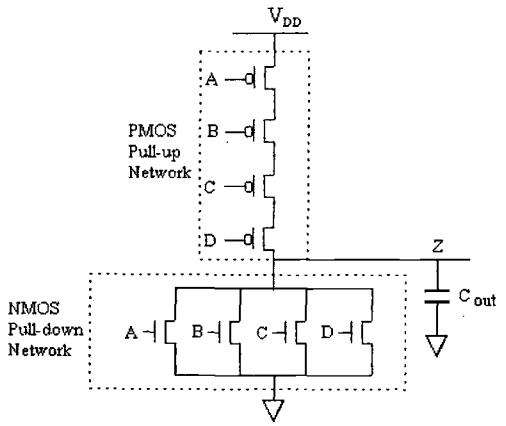


Figure 2.1 Static CMOS 4-input NOR gate

If any of the 4 inputs is high, then the PMOS pull-down network will block the current flowing from V_{DD} to the output node Z and the level of Z node will be pulled down to zero because the channel of the NMOS pull-down network is connected. Thus, the Boolean function Z= not (A+B+C+D) is realized. Figure 2.2 illustrates a complex logic gate that realizes the Boolean function Z = not [(A+B+C) (D+E)]. We can find out that more PMOS transistors have to be added to the circuit as the complexity of logical function is increased. Consequently, not only the silicon area, but also the delay will be significantly increased. In the next section, we'll discuss the dynamic logic circuit that could improve the performance and save the silicon area.

2.2 DYNAMIC LOGIC CIRCUITS

Dynamic logic is widely used in the critical circuits of high-performance microprocessors. Dynamic logic circuits have a better performance than static logic circuits because they have fewer PMOS transistors (or smaller n-well area) so the parasitic

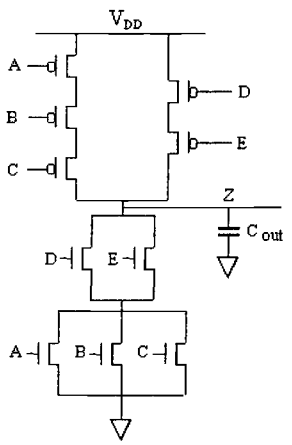


Figure 2.2 A complex static CMOS logic gate

capacitance in the output node has been significantly reduced. However, there are two main drawbacks that limit the design of dynamic logic circuits. The first one is large power consumption. Dynamic logic is a clocked logic so that its output node has to be precharged high each cycle during the precharge phase. If the evaluation operation (during the evaluate phase) for a circuit is prone to pull down the output level, then this circuit will consume a lot of power as the capacitance of the output node will be precharged from low to high frequently. The second drawback is the susceptibility to noise. Two main problems that dynamic logic circuits are susceptible to noise are charge sharing and capacitive coupling [18,19]. We'll discuss these two noise problems later in this chapter. Because noise problems in dynamic circuits could cause functional failures, careful analysis and verification are necessary to ensure the correct functionality. This means we have to spend more design efforts and resources on the dynamic circuit than the static circuit for implementing a same logical function. Thus, it may raise the cost or may sacrifice the time to market. Therefore, dynamic logic circuits are usually used when the performance is really

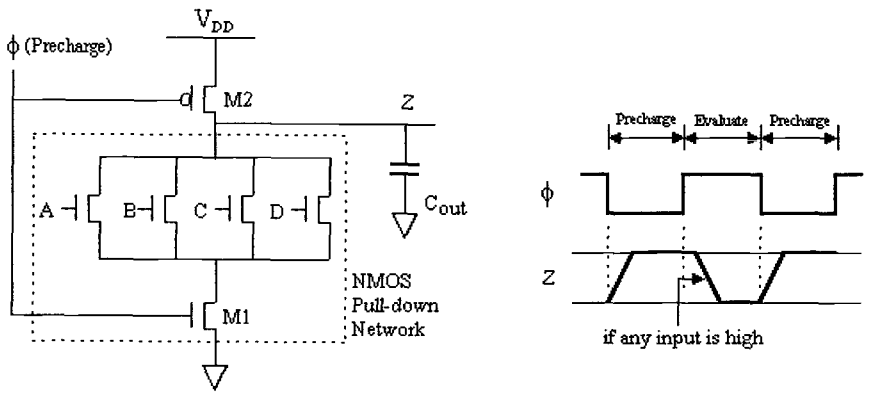


Figure 2.3 Dynamic 4-input NOR gate

critical or they can save a lot of silicon area.

Figure 2.3 shows the dynamic 4-input NOR gate that realizes the Boolean function Z= not (A+B+C+D). The operation of this dynamic circuit includes two phases: precharge phase and evaluate phase. During the precharge phase, the clocked signal ϕ is low and the parasitic capacitance of the output node Z is charged up because NMOS M1 is off and PMOS M2 is conducted. Subsequently, during the evaluate phase, the clocked signal ϕ is high and the output level is evaluated according to the applied inputs. If any of the four inputs is high during the evaluate phase, the output capacitance C_{out} will be discharged to zero because PMOS M2 is off and the NMOS pull-down network is conducted. Thus, the Boolean function Z= not (A+B+C+D) is realized. From this example we can find out that the dynamic 4-input NOR gate significantly reduces the number of transistors compared with the static circuit shown in figure 2.1 so the switch speed is faster because of less output capacitance.

Domino logic is one of the high-performance dynamic CMOS circuits. It cascades two or more simple dynamic circuits together to realize a more complex logic with better

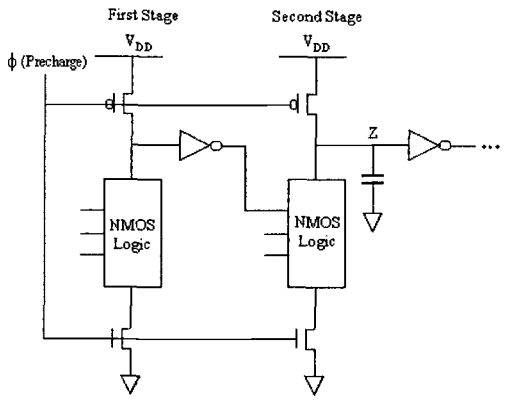


Figure 2.4 Domino circuit

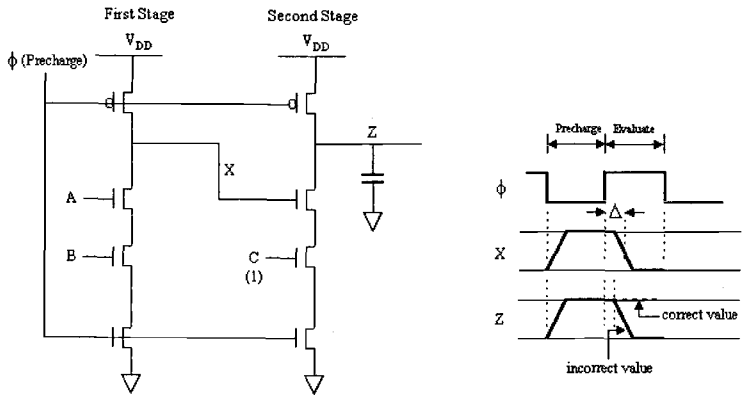


Figure 2.5 Function failure for a Domino circuit without inverter

performance. Figure 2.4 illustrates the generalized circuit for a Domino logic gate [18]. Its operation is similar to a single dynamic circuit except that the output of the first stage circuit is one input of the second stage circuit. One point has to be emphasized that the inverter between the first stage circuit and second stage circuit cannot be removed because that might cause a functional failure. Consider the example in figure 2.5, all the inputs (A, B and C) are high and the expected level for Z node is high after the evaluation is done. During the precharge phase, X and Z nodes are precharged to high. Subsequently, in the evaluate phase,

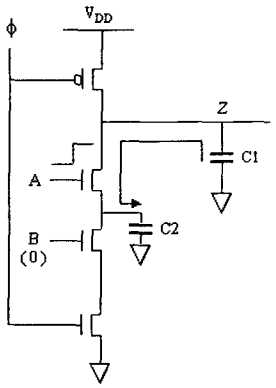
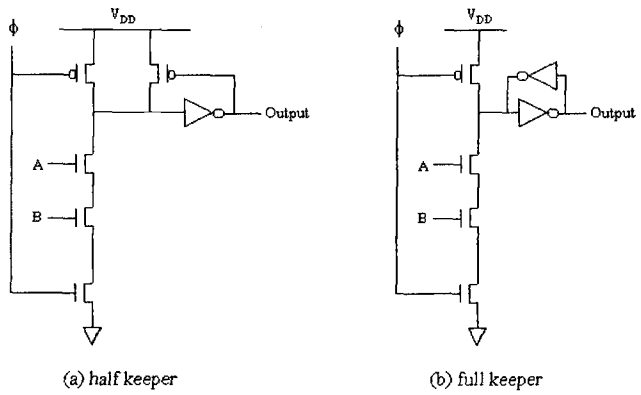


Figure 2.6 Charge sharing of a dynamic circuit

output node Z could be pulled down to ground first before X node is actually transited from 1 to 0. Thus, the output level is not what we expected. This situation will occur if the pull-down delay for the first stage is larger than that for the second stage. Therefore, the inverter between any two consecutive dynamic circuits is needed for realizing the Domino logic with the correct functionality.

One of the drawbacks for dynamic logic is its susceptibility to noise. Figure 2.6 illustrates the charge sharing of a dynamic circuit. During the precharge phase, node Z is precharged high and input A is low so there is no charge stored in C2 capacitor. During the evaluate phase, if input B is low and input A is asserted, then the charges stored in C1 capacitor will be shared with C2 capacitor. If C2 capacitance is large enough compared to C1 capacitance, the output level may be incorrect as a lot of charges flowed from C1 node to C2 node. To overcome the charge-sharing problem, we may add a half keeper or full keeper at the output node as shown in figure 2.7 [18,19]. An alternative solution is to precharge appropriate sharing points as well during the precharge phase. Interconnect capacitive



(a) half keeper (b) full keeper

Figure 2.7 Half keeper and full keeper for the dynamic circuit

coupling is also a noise problem that may cause the functional failure of a dynamic circuit. When other unrelated signal wires running closely adjacent to the output node of a dynamic circuit, the mutual capacitance among these signals could generate the noise problem. Thus, the charges in the output node of the dynamic circuit may be added or removed depending on the magnitudes of the output capacitance and the mutual capacitance. Such noise could be reduced by increasing the wire space or by shielding the unrelated wires near the output nodes of dynamic circuits.

2.3 INTERCONNECT

The interconnect delay has started to dominate the gate delay since the deep sub-micron technology was introduced to the market [17,18]. This trend has altered the nature of the microprocessor design that conventionally focused on the gate or logic delay. The estimation of the interconnect parasitics from the fub level to full-chip level for a chip must be reliably made in order to get the more realistic delay.

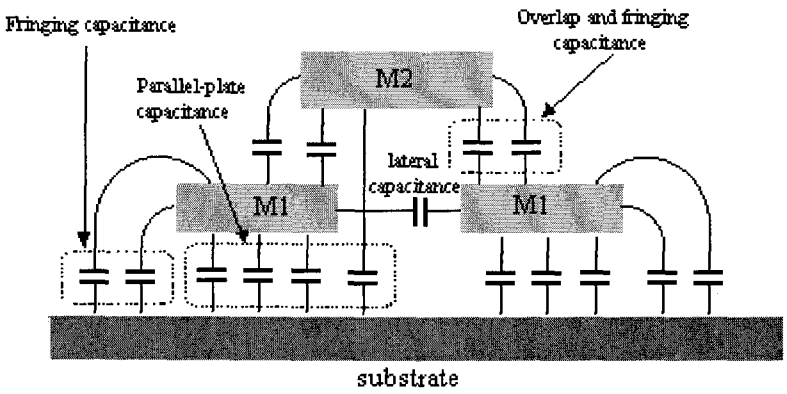


Figure 2.8 Capacitive coupling components

2.3.1 Parasitic Capacitance and Resistance of the Interconnection Wires

Figure 2.8 illustrates the capacitive coupling components between all interconnection lines running in parallel [18]. The first component is parallel-plate capacitance between the interconnection lines and the silicon substrate. The second one is fringing capacitance between the edges of interconnection lines and the substrate. The third one is lateral capacitance between any two parallel wires in the same layer. The last one is overlap and fringing capacitance between any two wires in the different layer. Though the capacitance models shown in figure 2.8 are more realistic, the estimation of interconnect capacitance in such a 3-D structure is difficult because the exact geometry for each wire must be taken into account.

Chip manufactures usually provide the parametric test result for a certain technology that shows the area capacitance and fringing capacitance values for all metal layers and poly. (The parametric test results for TSMC CMOS 0.18μm and 0.25μm are listed in Appendix.)

With these test results, we can estimate the parasitic capacitance and resistance from an accurate layout. Generally, the parasitic resistance of a metal will not have much influence on the propagation delay if the wire is not long enough. Take TSMC 0.18μm technology as example, the sheet resistance is 0.08 ohm/square for metal 1 to metal 4 and is 4.3ohm/square for poly. The resistance values for metal-poly and metal-diffusion contacts are around 5ohms. From the data shown above, the parasitic resistance could be ignored for the short-distance interconnection wires (or ploys) such as local wires. However, for the long-distance wiring, such as global wiring or inter-chip wiring, the parasitic resistance cannot be ignored because its total resistance is pretty large.

2.3.2 Delay Model

The delay model of an interconnection line can be modeled as a lumped or distributed RC network in most on-chip interconnects. This theory is base on the assumption that the time of flight across the interconnection line is significantly shorter than the signal rise/fall time [18]. On the other hand, if two wires are sufficiently long and they are closely running in parallel, then the inductance is not negligible and they have to be modeled as transmission line models (LRC model) instead of RC models.

2.4 LAYOUT

Layout is the physical presentation of a circuit. A good layout is essential as it can reduce the parasitic capacitance/resistance for transistors or interconnection lines for obtaining a better performance. Moreover, a compact layout can save a lot of silicon area

and the well-arranged metal geometry could lower and the possibility of functional failure caused by noise. Some techniques are used for reducing the parasitic capacitance and silicon area like sharing the common drain/source to two transistors or using the minimum metal width and device size.

3 CIRCUIT IMPLEMENTATIONS OF THE CAM-TYPE AND RAM-TYPE WAKEUP LOGIC

In this chapter we will start to introduce the circuit implementations of the CAM-type and RAM-type wakeup logic. Some partial layouts for these two types of wakeup circuits will also be discussed. We will show the performance results simulated by HSPICE in the next chapter. The following table defines the superscalar design parameters used in the wakeup logic design.

Superscalar design parameters

Symbol	Represent	Description
WSIZE	Instruction Window Size	The maximum number of instructions that issue window or reservation station can accommodate
IW	Issue Width	The maximum number of instructions that issue logic can issue to execution units per cycle
NREG	Number of Physical Registers	The number of entries in the ROB or the number of physical registers in the register file
TAG_{width}	Width of Physical Register Tags	The number of bits for the physical register tag or operand tag. (e.g. TAG_{width}=5 if NREG=32)

3.1 CAM-TYPE WAKEUP CIRCUIT

We've discussed the operation of the CAM-type wakeup logic from chapter 1. Now we'll introduce the circuit implementation of the CAM-type wakeup logic. Figure 3.1 illustrates one entry of the CAM-type wakeup logic with the configurations of 4-issue width and 32-NREG. The width of physical register tags or operand tags (TAG_{width}) is 5 bits

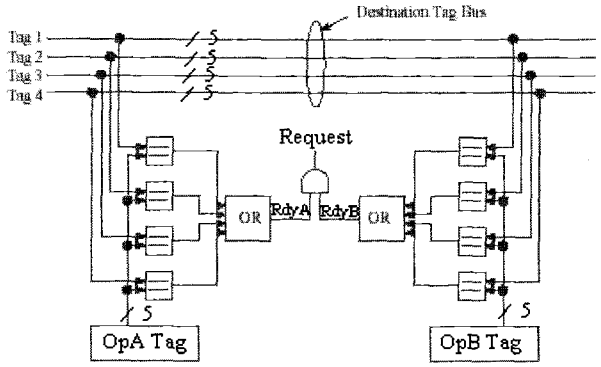


Figure 3.1 CAM-type wakeup logic for one instruction

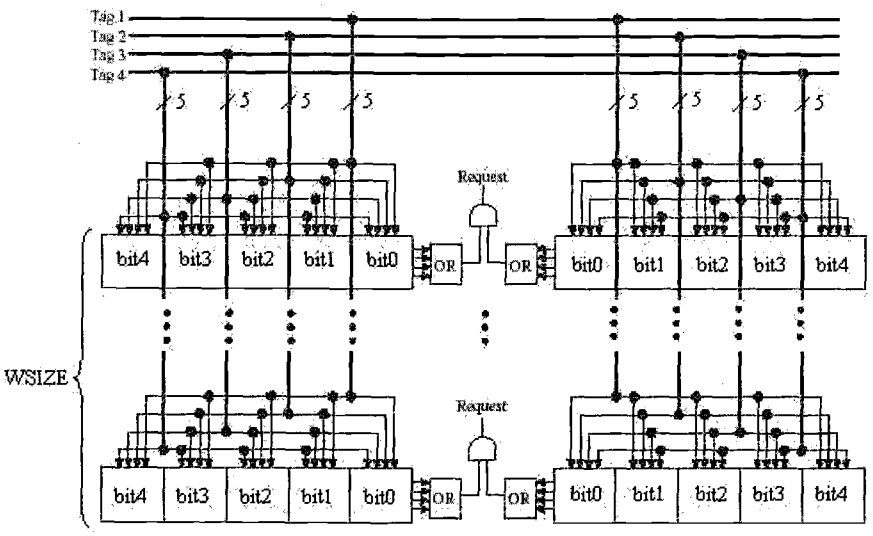


Figure 3.2 CAM-type wakeup logic for all instructions in the issue window

because the number of physical registers is 32. Each operand tag (OpA or OpB Tag) has four chances to compare with the destination tags each cycle because the issue width is 4. If the OpA tag (or OpB tag) matches any destination tag from Tag1 and Tag4, then RdyA (or RdyB) will be asserted. As soon as both RdyA and RdyB signals are high, the Request

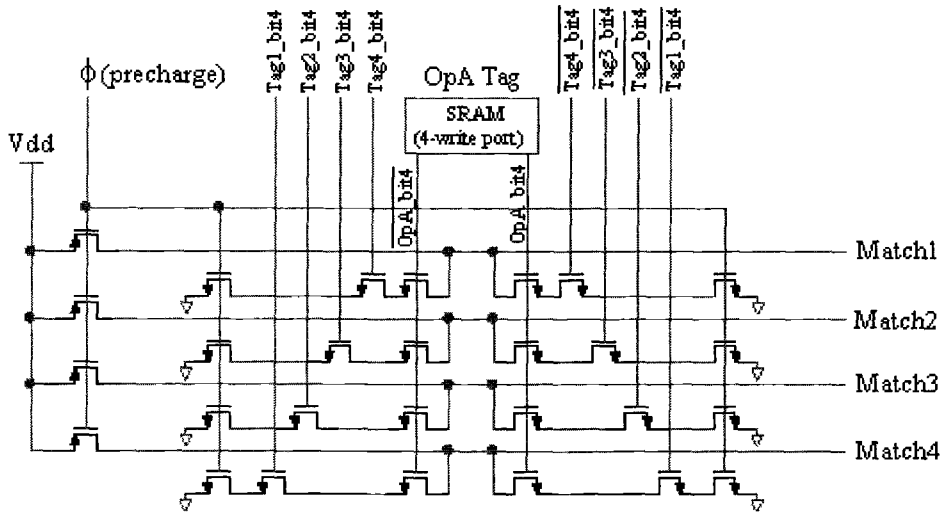


Figure 3.3 4-issue width CAM cell in the wakeup logic

signal will be asserted and this instruction is ready for execution. That means both the required operand values have become available and the data dependence hazard is resolved.

Figure 3.2 illustrates the complete CAM-type wakeup circuit for all the instructions waiting in the issue window [4]. The window size defines the maximum number of instructions the issue window can accommodate. If all the 5-bit data (from bit0 to bit4) of a source operand tag match those of any destination tag, then this source operand value is available for execution. Figure 3.3 shows the circuit implementation of the very left matching block (bit 4) for any entry shown in figure 3.2. It's basically a CAM (content-addressable memory [11]) structure which compares the 1-bit data stored in the SRAM with the 1-bit data driven from outside.

Let's consider the example shown in figure 3.3 for determining if the driven signal Tag1_bit4 matches the data OpA_bit4 stored in the SRAM. During the precharge phase (ϕ is low), the signal Match4 is precharged high as there is no path to ground for Match4 wire.

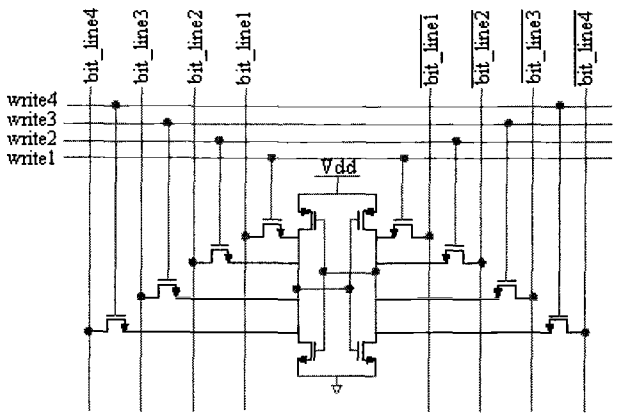
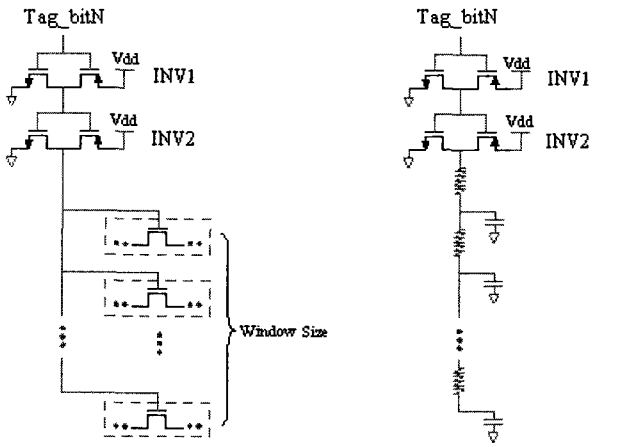


Figure 3.4 4-write port SRAM

Subsequently, during the evaluate phase (ϕ is high), if the data of Tag1_bit4 and OpA_bit4 are both 1 or 0, then the level of Match4 line will stay high as there is still no path for the charges stored in the parasitic capacitance of Match4 wire going ground. On the other hand, different data in Tag1_bit4 and OpA_bit4, the Match4 line will be pulled down to ground through one conducted channel. Such one-bit matching mechanism can be extended to N bits by employing N 1-bit CAMs and concatenating all their match lines together. Thus, any unmatched data for a bit position will cause the match line being pulled down so the N-bit tag comparison is not successful. Figure 3.4 illustrated the multiple-write port SRAM used for storing each individual bit of an operand tag. The number of write ports is identical to issue width.

In the following subsections we will introduce the partitions of the CAM-type wakeup circuits that determine the total wakeup logic delay.

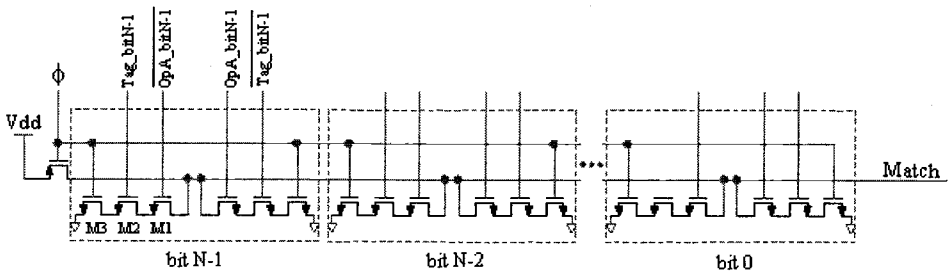


(a) Tag drive circuit (b) Interconnect RC model

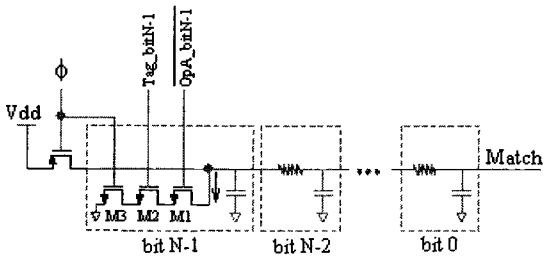
Figure 3.5 Tag drive circuit and the interconnect RC model

3.1.1 Tag Drive Circuit

The tag drive circuit is responsible for driving each individual bit of the destination tag into all the entries in the wakeup circuit for matching the operand tags. Figure 3.5 (a) illustrates the circuit driving a 1-bit signal into the wakeup circuit and figure 3.5 (b) shows its interconnect RC model. The tag drive circuit, which is composed of two inverters, is optimized for speed and silicon area. The first inverter, INV1, was sized reasonably small otherwise the gate capacitance of this inverter will be a heavy loading for the previous stage circuit outside the wakeup logic. The window size could significantly affect the tag drive time because the gate capacitance and the length of the output interconnection wire will increase as the window size is increased.



(a) N-bit tag match circuit

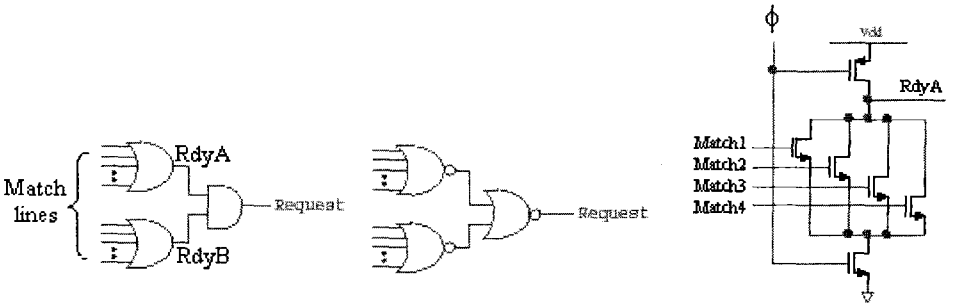


(b) Worst-case RC model

Figure 3.6 Tag match circuit and the worst-case RC model

3.1.2 Tag Match Circuit

As mentioned earlier, the CAM circuit is responsible for comparing two 1-bit data. The tag match circuit of the CAM-type wakeup circuit is basically composed of multiple CAMs with a common match line. Figure 3.6 (a) shows the N-bit tag match circuit that compares the operand tag with one destination register tag. If all bits in the operand tag and the destination tag match each other, then the level of Match line will stay high. Otherwise, the level of Match line will be pulled down. Figure 3.6 (b) illustrates the worst-case match delay and its RC model. The worst-case match delay occurred when only one bit of these two tags doesn't match each other. Thus, the capacitance for Match node has only one conducted channel to be discharged so the delay is the longest. The tag width, which is determined by the number of physical registers, has the most significant effect on the tag



(a) OR-AND circuit (b) Equivalent circuit (c) Dynamic 4-input NOR gate

Figure 3.7 Match OR-AND circuit

match time because it determines the magnitude of parasitic capacitance for the Match node. The sizes of the NMOS transistors-M1, M2 and M3-shown in figure 3.6 (b) were optimized for speed. Though the increase of their sizes has benefited the tag match performance, it will increase the tag drive time. Therefore, it's important to size these three transistors carefully to get the best trade-off between the tag match delay and tag drive delay.

3.1.3 Match OR-AND Circuit

The match OR-AND circuit, which is composed of two multiple-input OR gates and a 2-input AND, is responsible for asserting the Request signal depending on the availability of required operand values. If any match line for both operand tag comparisons is high, then the signal Request will be asserted. This means both the required operands are available and this instruction is marked ready for execution. Figure 3.7 (a) shows the gate-level implementation of the match OR-AND logic and figure 3.7 (b) shows the equivalent circuit used in our study. We used the dynamic circuit to realize a multiple-input NOR gate because its performance is better than the static circuit. The number of inputs for the NOR gate is 3,

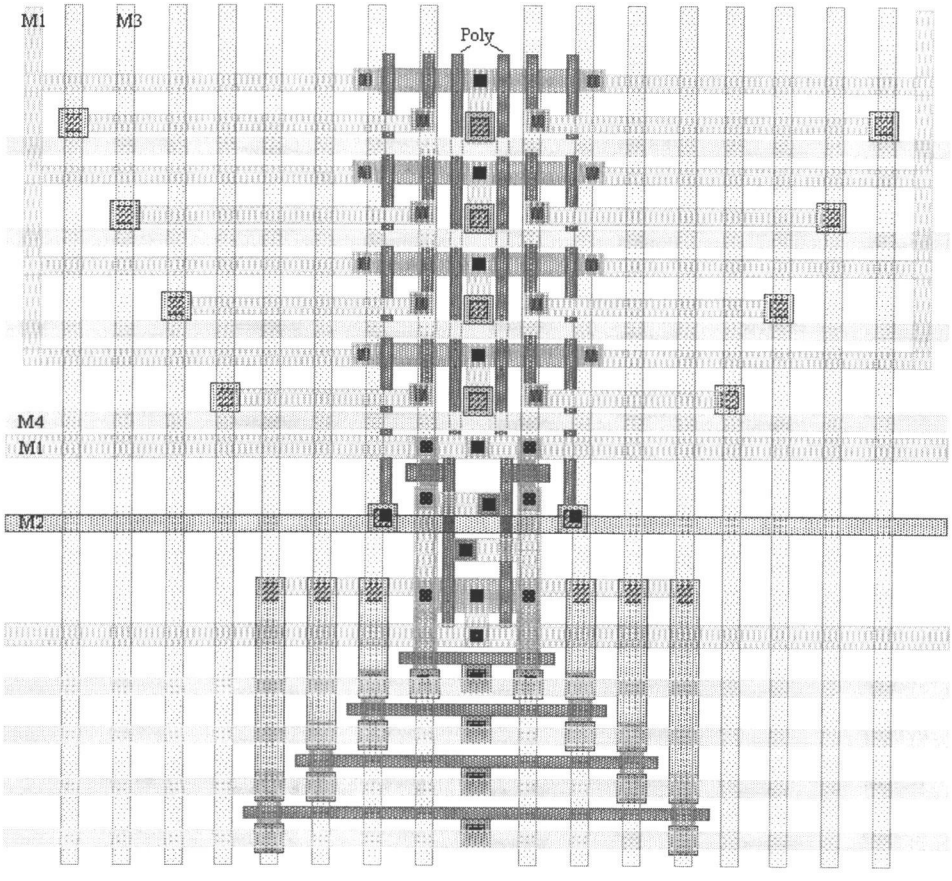


Figure 3.8 Layout of the CAM match circuit

4 or 5 depending on the issue width. Figure 2.7 (c) shows the circuit of a 4-input NOR gate used in our study.

3.1.4 Layout of the CAM-type Wakeup Circuit

The technology used for our wakeup circuit design is TSMC CMOS 0.18μm and 0.25μm processes. MOSIS SCMOS deep sub-micron design rule [16] for TSMC 0.18/0.25μm is used for DRC (Design Rule Check). Our layout methodology is area

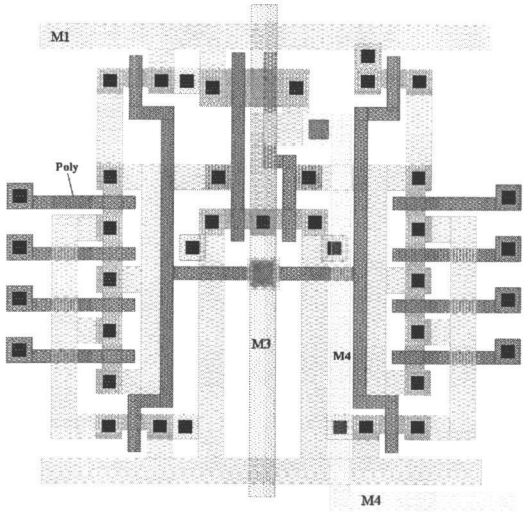
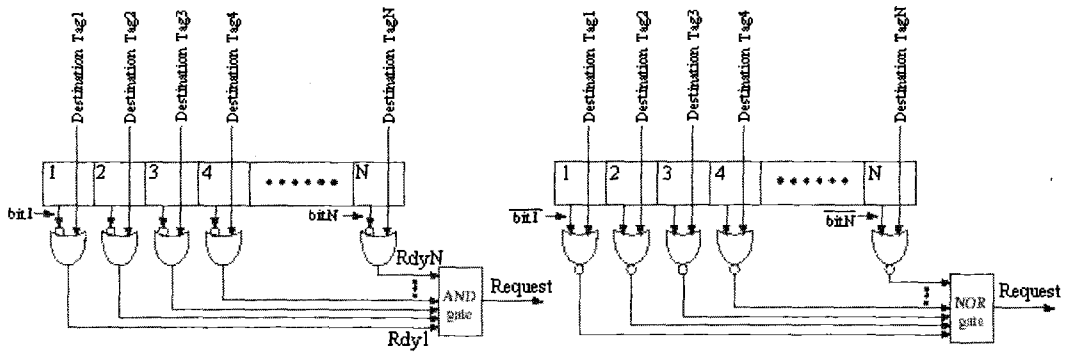


Figure 3.9 Layout of the match OR-AND circuit

optimization so the interconnection lines could be as short as possible. Shared drain/source is also a common technique to reduce the parasitic capacitance and silicon area [20], which is frequently used in our design. There are 6 metal layers in the TSMC 0.18/0.25μm technology, but only 4 of them (Metal1 to Metal4) are used in our design. Generally, the very 2 top metal layers are used for global routing within a chip.

Figure 3.8 shows the layout for the CAM match circuit shown in figure 3.3. The lower portion of the layout represents a 4-write port SRAM and the upper portion represents the NMOS transistors shown in figure 3.3. Figure 3.9 shows the layout for the 4-issue width match OR-AND gate that consists of two dynamic 4-input NOR circuits (in two sides) and a 2-input static AND circuit (in central).

The total layout area of the CAM-type wakeup circuit for 0.18μm technology is around 0.05 mm^2 (188μm X 283μm) with the design configurations of 4-IW, 20-WSIZE and 64-NREG.



(a) RAM-type wakeup logic for one instruction (b) Equivalent circuit

Figure 3.10 RAM-type wakeup logic for one instruction

3.2 RAM-TYPE WAKEUP CIRCUIT

We've described the operation of the RAM-type wakeup logic in chapter 1 (figure 1.4). Now we'll introduce the circuit implementation of the RAM-type wakeup logic. Consider one entry of the RAM-type wakeup circuit redrawn in figure 3.10 (a) [6]. There are N 1-bit data stored in the SRAMs representing the need or not for a certain physical register value (or an operand value). For example, if an instruction is waiting for two operand values that are represented by the No.1 and No.2 entries of the ROB to become available, the data in the No.1 and No.2 SRAMs will be set to 1 while the others remain 0. As soon as the values of these two physical registers are available, both destination tags, Tag1 and Tag2, will be set to 1. Subsequently, the signal Request is asserted and this instruction is ready for execution. The number N represents the number of entries in the ROB (or the number of physical registers). Figure 3.10 (b) shows the equivalent circuit we used for realizing the RAM-type wakeup logic. The dynamic wide-NOR gate is often used in high-performance circuit

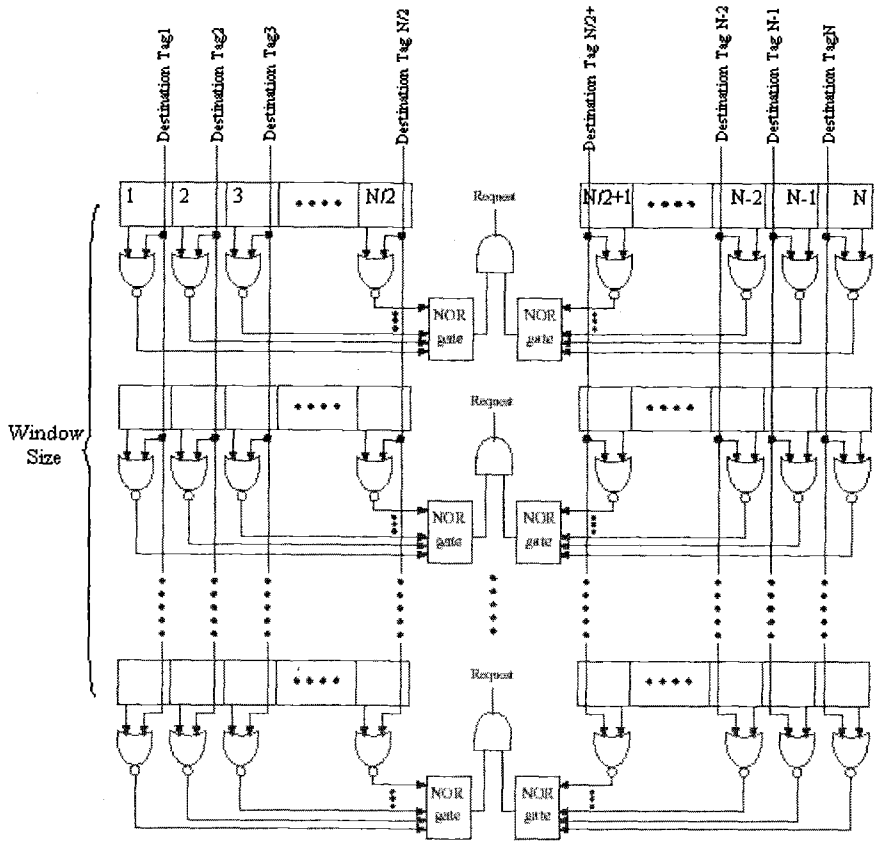


Figure 3.11 RAM-type wakeup logic for all instructions in the reservation station

design [7,17]. Notice that the data coming out from the 1-bit SRAM shown in figure 3.10 (b) has been inverted.

Figure 3.11 illustrates the complete RAM-type wakeup circuit for all the instructions waiting in the reservation station. The original circuit in figure 3.10 (b) has been divided into two symmetric parts, which are shown on left and right sides of figure 3.11, for minimizing the wide-NOR gate delay. An additional 2-input AND gate is needed for restoring the Request signal. Each entry (or row) in the wakeup circuit represents an instruction waiting in the reservation station. Figure 3.12 shows the circuit implementation

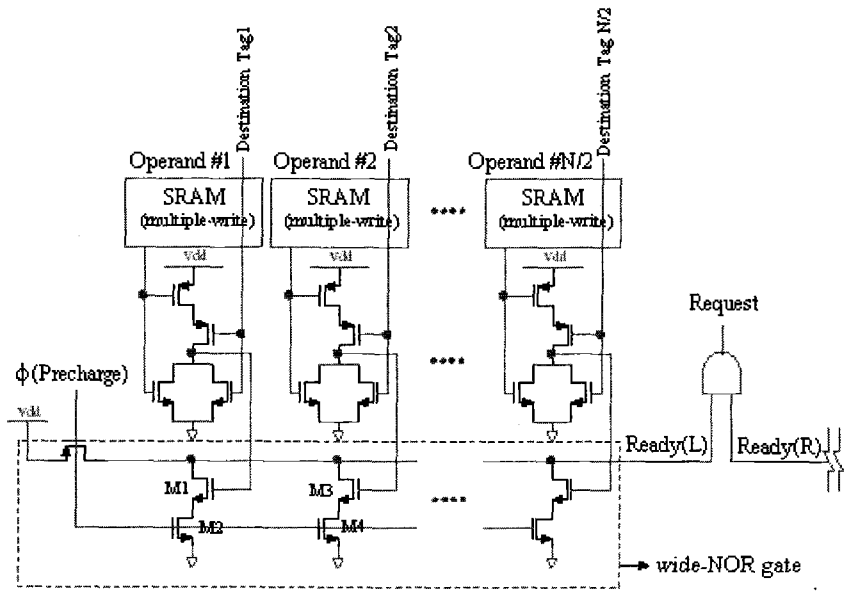
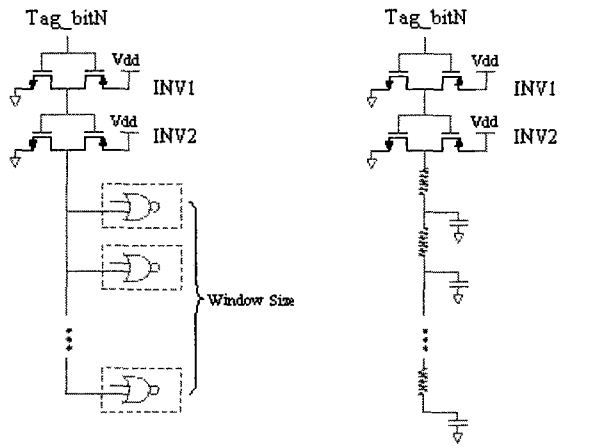


Figure 3.12 One entry of the RAM-type wakeup circuit

for the left portion of any wakeup circuit entry shown in figure 3.11. The structure of the multiple-write port SRAM is identical to that for the CAM-type wakeup circuit. The dynamic logic circuit shown in the lower portion of figure 3.11 is used for realizing the wide-NOR gate because of its high performance. Ready wire of the wide-NOR circuit is charged up during the precharge phase. Subsequently, during the evaluate phase, the level of Ready line will be pulled down if any signal generated from the 2-input NOR gate is high. This means that any of the required operands is still not available. On the other hand, if all the required operands are available, both Ready (L) and Ready(R) signals will remain high so the Request signal is asserted high. Thus, this instruction is ready for execution. In the following subsections we'll introduce the partitions of the RAM-type wakeup circuit that determine the total wakeup logic delay.



(a) Tag drive circuit (b) Interconnect RC model

Figure 3.13 Tag drive circuit of the RAM-type wakeup logic

3.2.1 Tag Drive Circuit

The structure and design concept of the tag drive circuit is very similar to the one we discussed in section 3.1.1 for the CAM-type wakeup logic. Figure 3.13 (a) illustrates the tag drive circuit for the RAM-type wakeup logic and figure 3.13 (b) shows its interconnect RC model.

3.2.2 Tag NOR2 Circuit

The tag NOR2 gate takes the destination register tag and the inverted SRAM data as it inputs, then generates the output for the use of the dynamic wide-NOR logic. We used a normal static CMOS circuit to realize the NOR2 gate because the tag NOR2 gate is not critical to the performance of the RAM-type wakeup logic. In our design, we minimized the size of tag NOR2 gate to increase the performance of the tag drive circuit and to save the layout area.

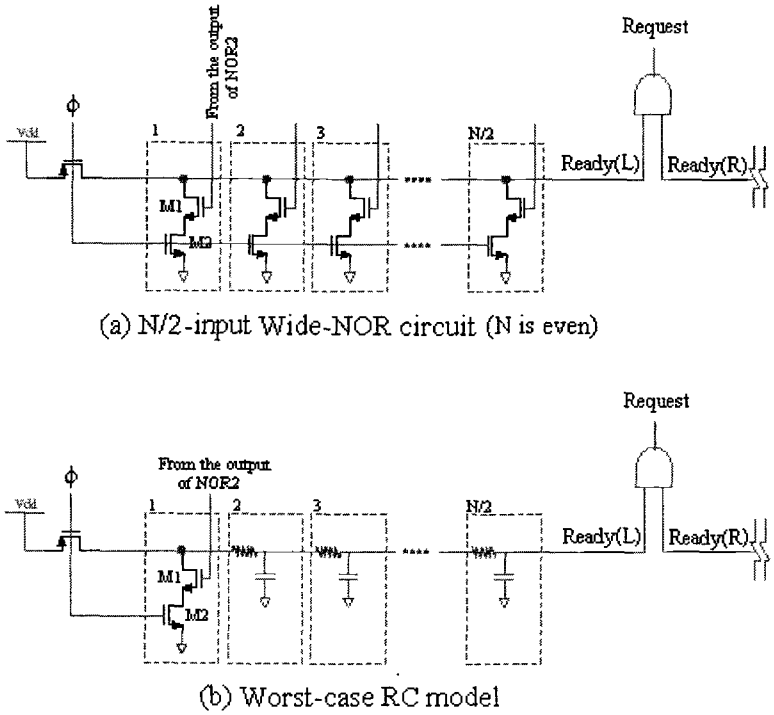


Figure 3.14 Wide-NOR circuit and the worst RC model

3.2.3 Wide-NOR Circuit

Figure 3.14 (a) depicted the dynamic wide-NOR gate that determines if the required operand values (for the left side in figure 3.11) are available. If any input, which came out from NOR2 output, of the wide-NOR gate is 1, then the Ready (L) signal will be pulled down to ground in the evaluate phase. That means the corresponding operand value is still not available. Otherwise, the level of Ready (L) wire will remain high. Figure 3.14 (b) illustrates the worst-case RC delay model for the wide-NOR gate. The worst-case delay occurred when only one operand value is not available. The sizes of M1 and M2 transistors

are optimized for speed. Though increasing the widths of these two transistors could enlarge the channel for Ready wire to discharge sooner, it also increases the parasitic capacitance for Ready wire. Therefore, there is a trade-off at scaling M1 and M2 transistors to obtain the best performance.

3.2.4 AND2 Circuit

As mentioned in section 3.2, we divided the original wide-NOR circuit into two symmetric parts for obtaining a better performance. Thus, we need an additional AND2 gate to restore the Request signal. The two inputs of the AND2 gate are generated from the left-side and right-side wide-NOR gates. If they are both 1, then Request signal will be asserted. We used the static CMOS logic to realize the AND2 gate because its fan-in is few.

3.2.5 Layout of the RAM-type Wakeup Circuit

The layout concept for the RAM-type wakeup circuit is basically the same as we discussed in section 3.1.4 for the CAM-type wakeup circuit. Figure 3-15 shows the layout for two adjacent cells of the circuit shown in figure 3.12 (except the precharge PMOS). The upper-central portion of the layout represents the four NMOS transistors, M1 to M4, shown in figure 3.12. The very-left and very-right portions of the layout represent two NOR gates and the other portions represent 2 four-write port SRAMs.

The total layout area of the RAM-type wakeup circuit for 0.18μm technology is around 0.09 mm^2 (532μm X 176μm) with the design configurations of 4-IW, 20-WSIZE and 64-NREG.

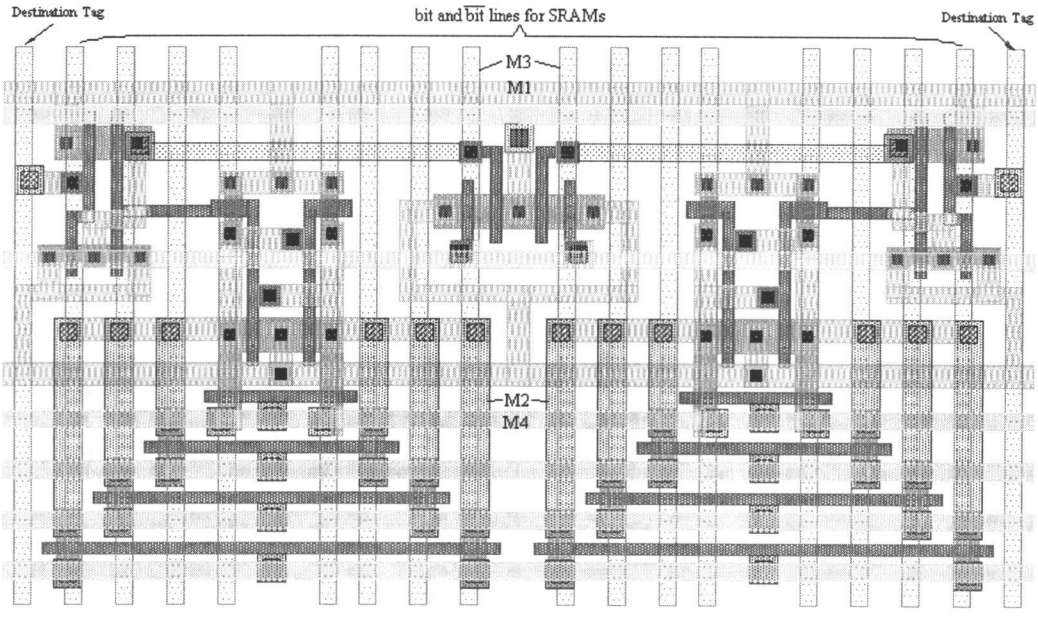


Figure 3.15 Layout for two adjacent cells in one entry of the RAM-type wakeup circuit shown in figure 3.12

4 SIMULATION RESULTS AND COMPARISONS

We've introduced the circuit implementations of the CAM-type and RAM-type wakeup logic in the previous chapter. In this chapter, we will show the their circuit performance (speed) done by the HSPICE simulator. TSMC CMOS 0.18μm and 0.25μm technologies are used for our circuit and layout designs. The technology files, including BSIM3 HSPICE models and parametric test reports, supporting the TSMC CMOS 0.18/0.25μm technology are obtained from MOSIS organization [16]. In our study, each circuit is optimized for achieving the best performance of wakeup logic. Besides, careful layout is done for minimizing the parasitic capacitance and silicon area. The RC parasitics is extracted form the real layout so it can be more correctly inserted to the HSPICE netlists for simulation. We decomposed the critical path of the wakeup logic into 3 to 4 parts for clearly observing the individual impact of these circuits on the total wakeup logic delay. We will show the delays of these partitioned circuits and the breakdown of the total wakeup logic delay. The simulation results for the CAM-type and RAM-type logic are separately shown in the following two subsections. Finally, we made a performance comparison between these two types of wakeup logic.

4.1 HSPICE RESULTS FOR THE CAM-TYPE WAKEUP LOGIC

The CAM-type wakeup logic delay is composed of three components: tag drive time, tag match time and match OR-AND time. The delay equation is shown as following:

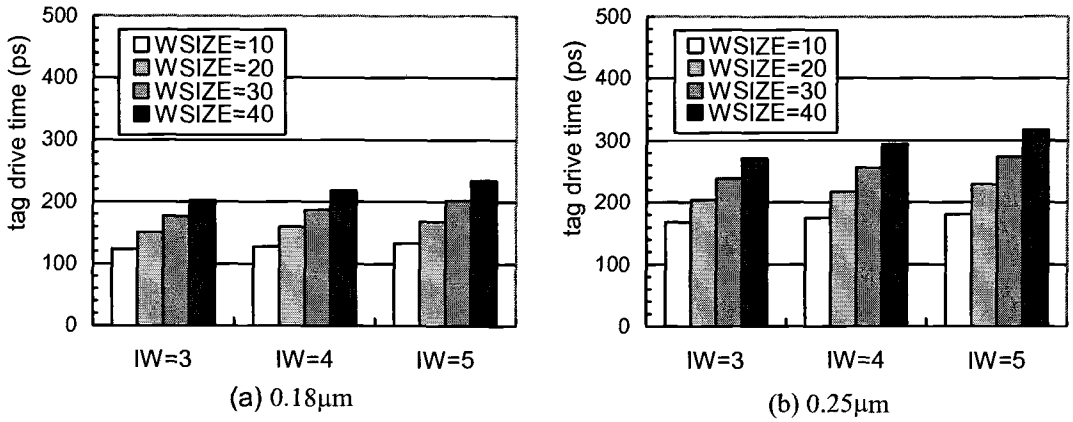


Figure 4.1 Tag drive time of the CAM-type wakeup logic

$Delay\_CAM = T_{tag\_drive} + T_{tag\_match} + T_{match\_OR\text{-}AND}$

T_{tag_drive} is the time taken by the tag drive circuit (figure 3.5) to drive the tag bit into each entry of the wakeup logic. T_{tag_match} is the time taken by the tag match circuit (figure 3.6) to pull down the level of the match line if a mismatch between the operand tag and the destination tag occurred. $T_{match\_OR\text{-}AND}$ is the time taken by the match OR-AND circuit (figure 3.7 (a)) to OR the match signals and to AND RdyA and RdyB signals for asserting the Request signal. Delay_CAM represents the total delay for the CAM-type wakeup logic. The following subsections show the delays for these three delay components and the breakdown of the total CAM-type wakeup logic delay.

4.1.1 Tag Drive Time

Figure 4.1 shows the tag drive time that varies with issue width (IW) and window size (WSIZE) for 0.18μ m and 0.25μ m technologies. As expected, the tag drive time increases as issue width or window size is increased. When the issue width is increased, the

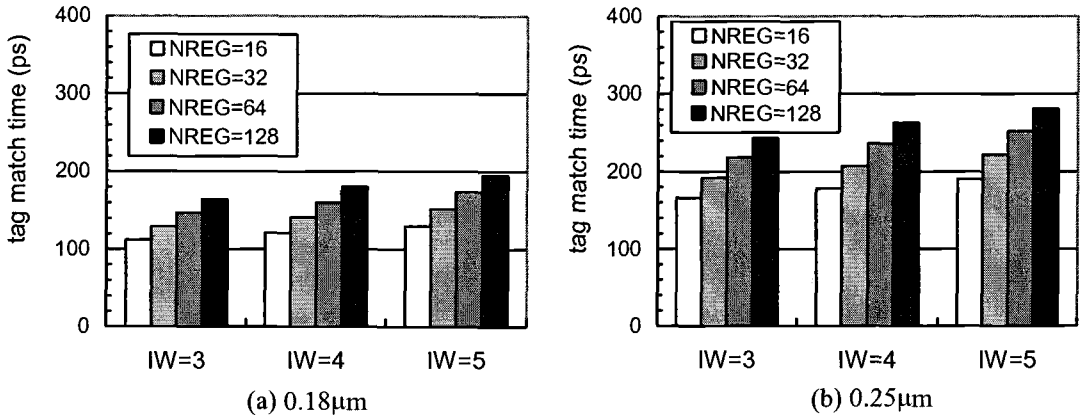


Figure 4.2 Tag match time of the CAM-type wakeup logic

interconnect length of the tag drive circuit (figure 3.5) will increase because the heights of the multiple-write port SRAM and the CAM match circuit (figure 3.3) are increased. When the window size is increased, the interconnect length of the tag drive circuit also increases and there will be more fan-out for the output node. From the two reasons above, the tag drive time will increase as issue width or window size is increased. However, the effect of window size is more significant than that of issue width because of the dominant gate capacitance. The tag drive time for 0.18μm technology is around 75% compared with the time for 0.25μm technology for the design configurations of window size and issue width.

4.1.2 Tag Match Time

Figure 4.2 shows the tag match time that varies with issue width and number of physical registers (NREG) for 0.18μ m and 0.25μm technologies. When the issue width is increased, the width of the 1-bit CAM match circuit (figure 3.3) will increase because more bit lines of the multiple-write port SRAM and destination tag will be employed. When the

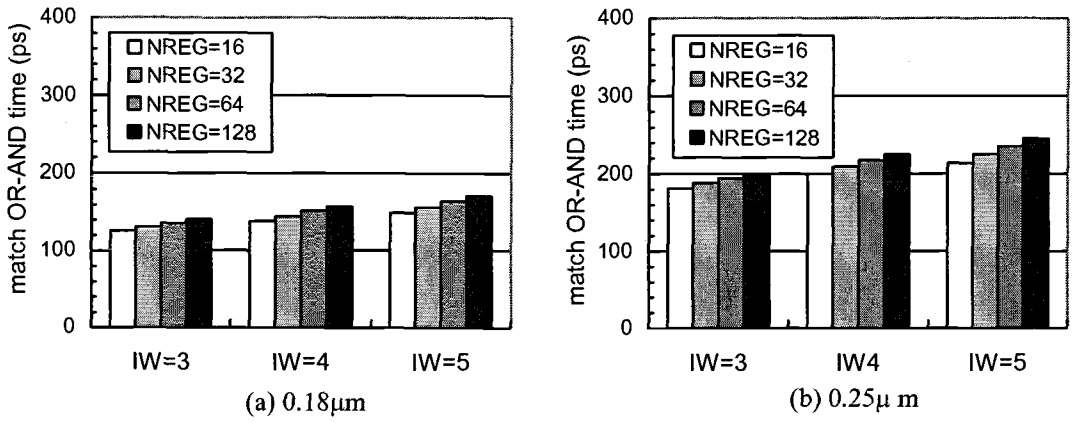


Figure 4.3 Match OR-AND time of the CAM-type wakeup logic

number of physical registers is increased, not only the match line of the tag match circuit will be connected to more NMOS drains, but the length of the match line will also be prolonged as the operand tag (or destination tag) width is expanded. Therefore, the tag match time is increased as issue width or number of physical registers increases. However, the delay impact from the increase of NREG is more significant because more drain capacitance is introduced to the match lines. The tag match time for 0.18μm technology is about 70% compared with the time for 0.25μm technology with the same issue width and number of physical registers.

4.1.3 Match OR-AND Time

Figure 4.3 shows the match OR-AND time that varies with issue width and number of physical registers (NREG) for 0.18μm and 0.25μm technologies. The reason that issue width has an impact on the delay time is that the fan-in for the dynamic NOR gate (figure 3.7) is equal to the issue width. As we assumed the Request wire has to be extended to the

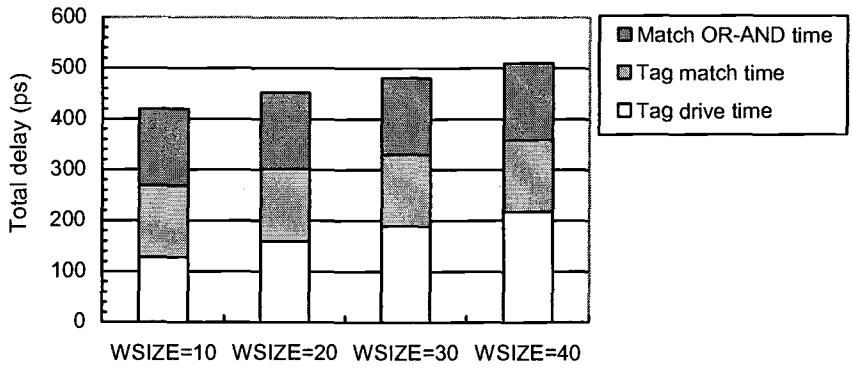


Figure 4.4 CAM-type wakeup logic delay versus window size. This result is based on 4-IW, 32-NREG and 0.18μm technology.

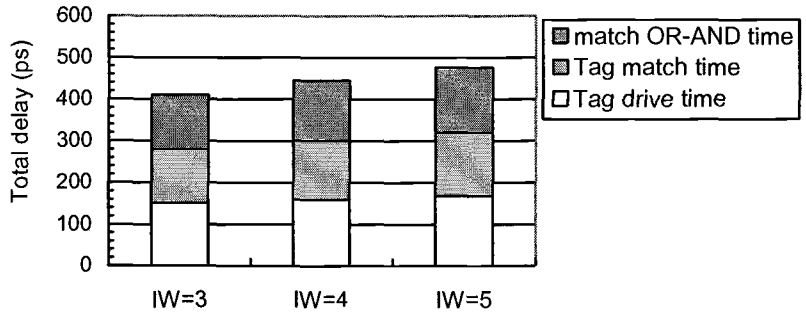


Figure 4.5 CAM-type wakeup logic delay versus issue width. This result is based on 20-WSIZE, 32-NREG and 0.18mm technology.

boundary of the wakeup logic, the length of Request line will vary with the dimension of wakeup logic. Therefore, number of physical registers also has a minor effect on the delay time because it will affect the dimension of wakeup logic.

4.1.4 Total Delay of the CAM-type Wakeup Logic

The total delay of the CAM-type wakeup logic is composed of three components: tag drive time, tag match time and match OR-AND time. Figure 4.4 to figure 4.7 show the breakdown of the total wakeup delay for various window sizes, issue widths, numbers of

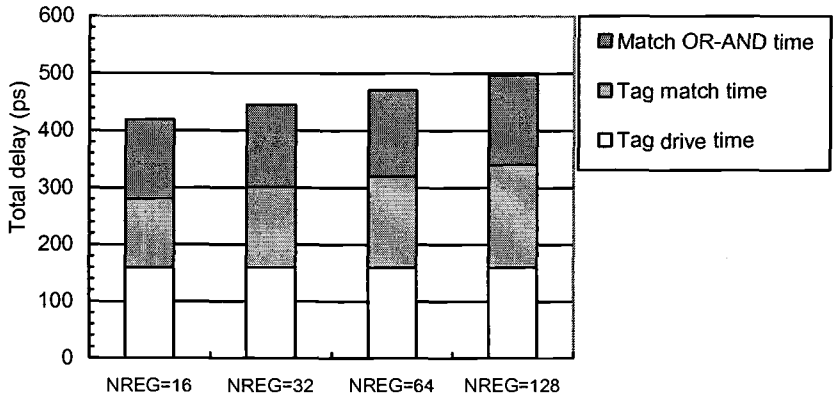


Figure 4.6 CAM-type wakeup logic delay versus number of physical registers. This result is based on 20-WSIZE, 4-IW and 0.18μm technology.

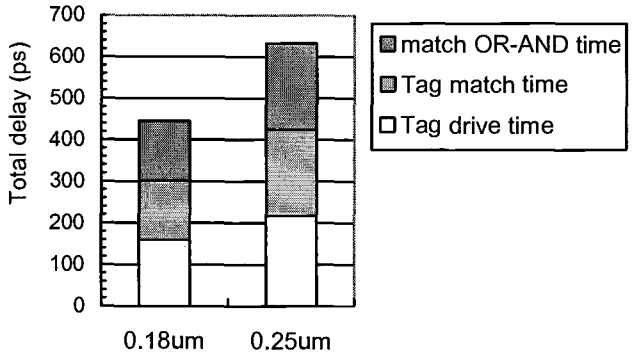


Figure 4.7 CAM-type wakeup logic delay versus feature size This result is based on 20-WSIZE, 4-IW and 32-NREG configurations.

physical registers and feature sizes. From figure 4.5 to figure 4.7 we know that each delay component contributes almost equal percentage to the total delay time. Consider figure 4.4, the window size is a factor that greatly impacts the total delay. Process is also an important factor that affects the total CAM-type wakeup logic delay. When the feature size is changed from 0.25μm to 0.18μm, there will be more than 30% performance improvement for the wakeup logic. We'll discuss more about the factors that affect the total delay with the presence of the RAM-type wakeup logic in section 4.3.

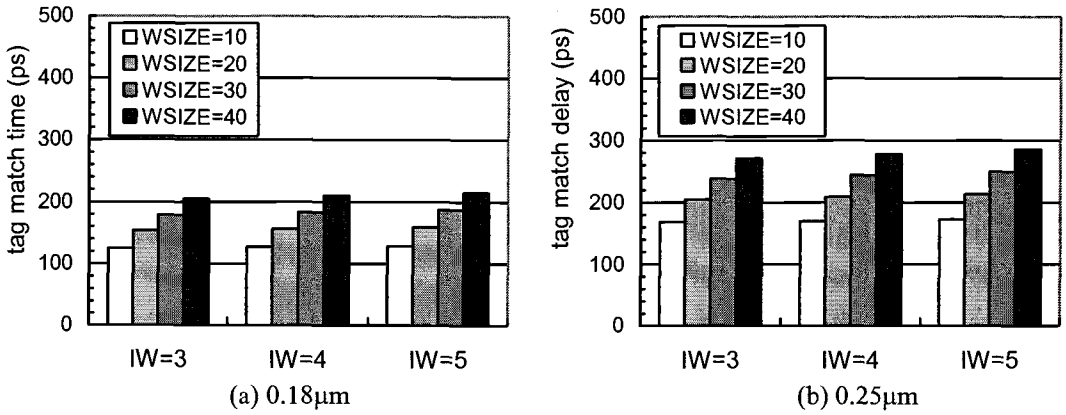


Figure 4.8 Tag drive time of the RAM-type wakeup logic

4.2 HSPICE RESULTS FOR THE RAM-TYPE WAKEUP LOGIC

The RAM-type wakeup logic delay is composed of four components: tag drive time, tag NOR2 time, wide-NOR time and AND2 time. The symbolic equation of the total delay is shown as following:

$$Delay\_RAM = T_{tag\_drive} + T_{tag\_NOR2} + T_{wide\_NOR} + T_{AND2}$$

T_{tag_drive} is the time taken by the tag drive circuit (figure 3.13) to drive the tag bit into each entry of the wakeup logic. T_{tag_NOR2} is the time taken by the NOR2 gate (shown in figure 3.11) to NOR the inverted 1-bit operand tag and one destination tag. T_{wide_NOR} is the time taken by the wide-NOR circuit (figure 3.14) to pull down the level of Ready signal if any required operand value is not available. T_{AND2} is the time taken by the AND2 gate to AND the Ready (L) and Ready(R) signals for asserting the Request signal. Delay_RAM represents the total delay time of the RAM-type wakeup logic.

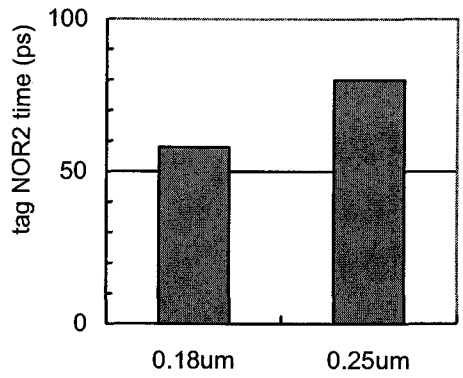


Figure 4.9 Tag NOR2 time of the RAM-type wakeup logic versus feature size

4.2.1 Tag Drive Time

Figure 4.8 shows the tag drive time that varies with issue width (IW) and window size (WSIZE) for 0.18μm and 0.25μm technologies. The reasons that the tag drive time is increased as issue width or window size increases is almost the same as those for the CAM-type wakeup logic discussed in section 4.1.1. The tag drive time will increase more rapidly as the window size is expanded.

4.2.2 Tag NOR2 Time

Figure 4.9 shows the tag NOR2 time that varies with different technologies. The tag NOR2 time is independent of issue width, window size and number of physical registers because none of them will affect the size and output interconnect of the tag NOR2 circuit.

4.2.3 Wide-NOR Time

The wide-NOR time is the most significant delay component for the RAM-type wakeup circuit because it increases sharply as the number of physical registers (NREG) is

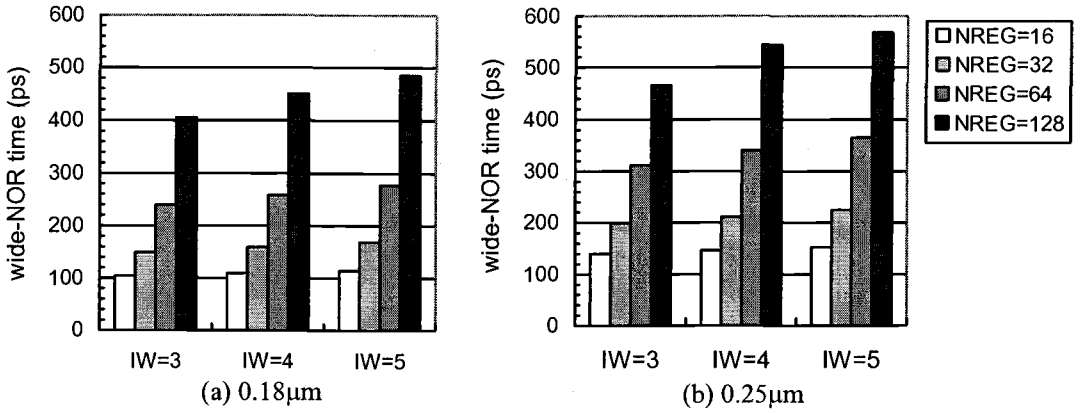


Figure 4.10 Wide-NOR time of the RAM-type wakeup logic

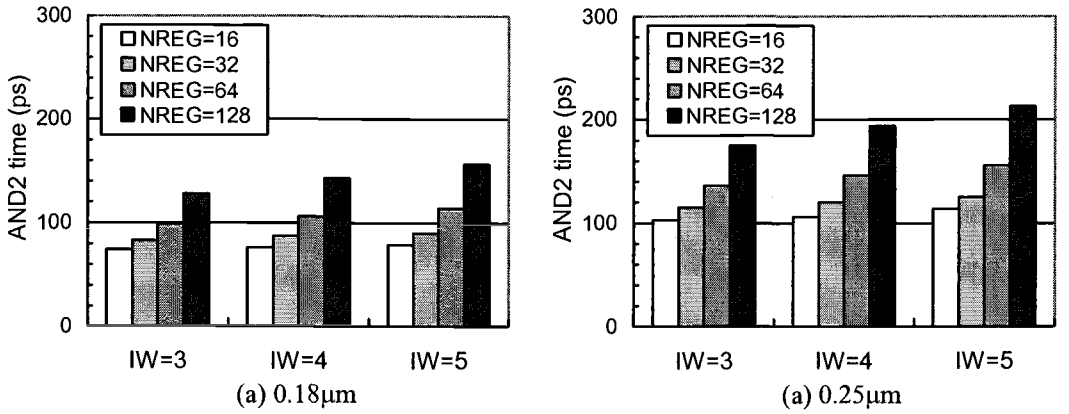


Figure 4.11 AND2 time of the RAM-type wakeup logic

increased. The relationship between the wide-NOR time, issue width and NREG is shown in figure 4.10. Issue width has a minor impact on the delay because the width of the multiple-write port SRAMs (figure 3.12) varies with the issue width. Thus, the length of Ready line in figure 3.12 is extended as the issue width increases. The reason why the wide-NOR delay is increased sharply as NREG increases is that more fan-in for the wide-NOR gate will be introduced. Therefore, more and more drain capacitance will be added to the output node of the wide-NOR circuit.

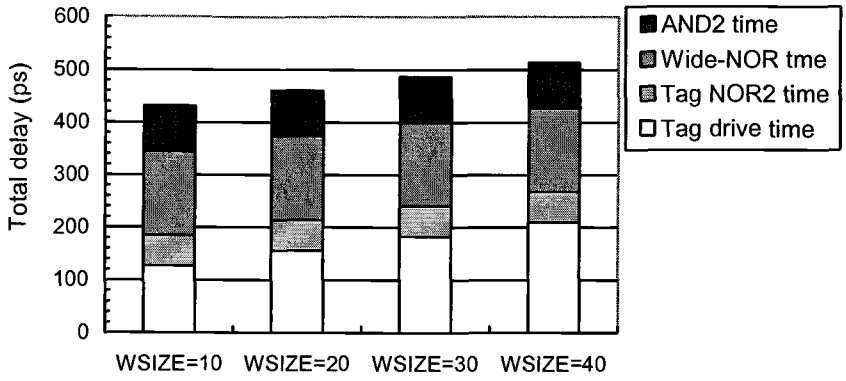


Figure 4.12 RAM-type wakeup logic delay versus window size. This result is based on 4-IW, 32-NREG and 0.18 μm technology.

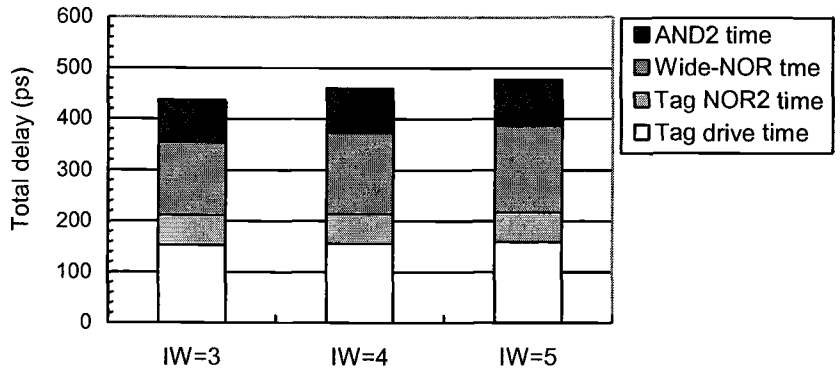


Figure 4.13 RAM-type wakeup logic delay versus issue width. This result is based on 20-WSIZE, 32-NREG and 0.18 μm technology.

4.2.4 AND2 Time

Figure 4.11 shows the AND2 delay varying with issue width and number of physical registers. We assumed the output wires of AND2 gates have to be extended to the fringe of the wakeup logic so the length of output wires will vary with the dimension of the wakeup circuit. Both Issue width and NREG could affect the dimension of the wakeup circuit. Therefore, the AND2 delay is increased as issue with or NREG increases.

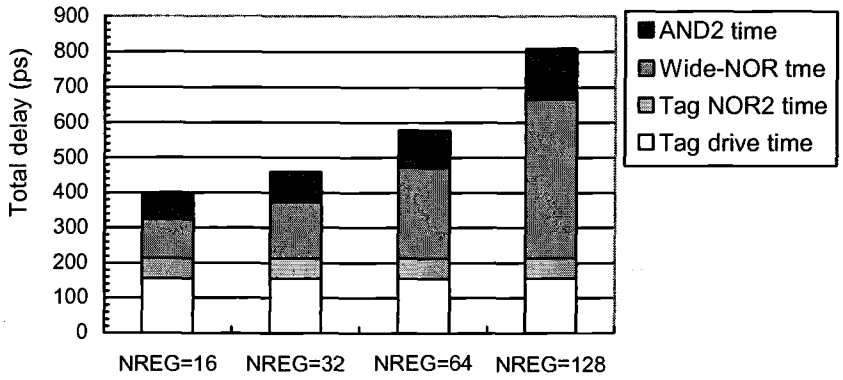


Figure 4.14 RAM-type wakeup logic delay versus number of physical registers. This result is based on 20-WSIZE, 4-IW, and 0.18μm technology.

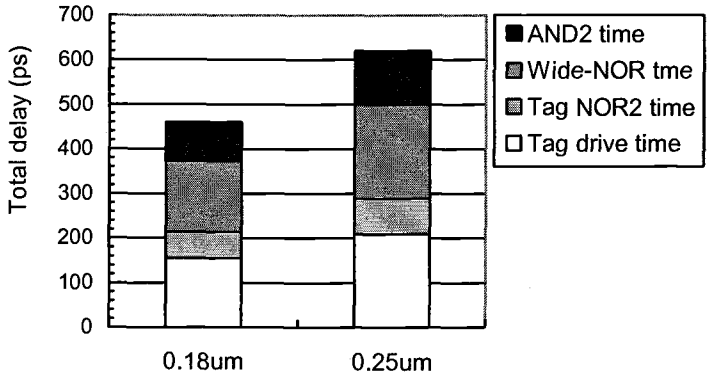


Figure 4.15 RAM-type wakeup logic delay versus feature size. This result is based on 20-WSIZE, 4-IW and 32-NREG configurations.

4.2.5 Total Delay of the RAM-type Wakeup Logic

The total delay of the RAM-type wakeup logic is composed of four components: tag drive time, tag NOR2 time, wide-NOR time and AND2 time. Figure 4.12 to figure 4.15 show the breakdown of the total wakeup delay versus various window sizes, issue widths, numbers of physical registers and feature sizes respectively. Consider the result shown in figure 4.12 with the design configuration of 4-IW and 32-NREG, the tag drive time or wide-NOR time contributes a major portion of the total delay. The tag drive time dominates

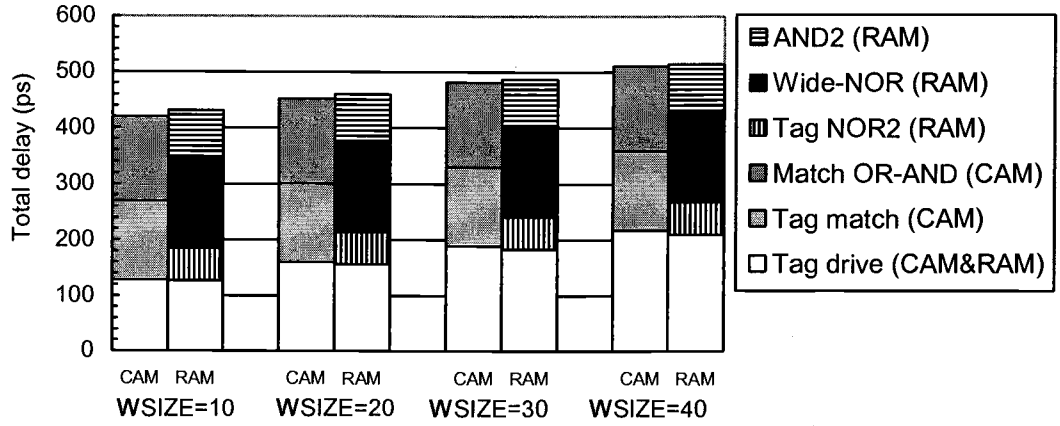


Figure 4.16 The total delay comparison of the CAM-type and RAM-type wakeup logic with various window sizes. This result is based on the configuration of 4-IW and 32-NREG for 0.18μm technology.

the wide-NOR time when the window size is large. Consider the result shown in figure 4.14 with the design configuration of 20-WSIZE and 4-IW, the total delay increases rapidly as the number of physical registers increases. It can be mostly attributed to the wide-NOR delay that is very sensitive to the number of physical registers. We know from figure 4.15 that feature size is also a significant factor impacting the delay of the RAM-type wakeup logic. We'll discuss more about the delay components that affect the total delay with the presence of the CAM-type wakeup logic in the next section.

4.3 PERFORMANCE COMPARISONS BETWEEN THE CAM-TYPE AND RAM-TYPE WAKEUP LOGIC

In this section we showed the performance comparison between the CAM-type and RAM-type wakeup logic based on varying only one of the three design parameters- issue width (IW), window size (WSIZE) and number of physical registers (NREG)- at a time.

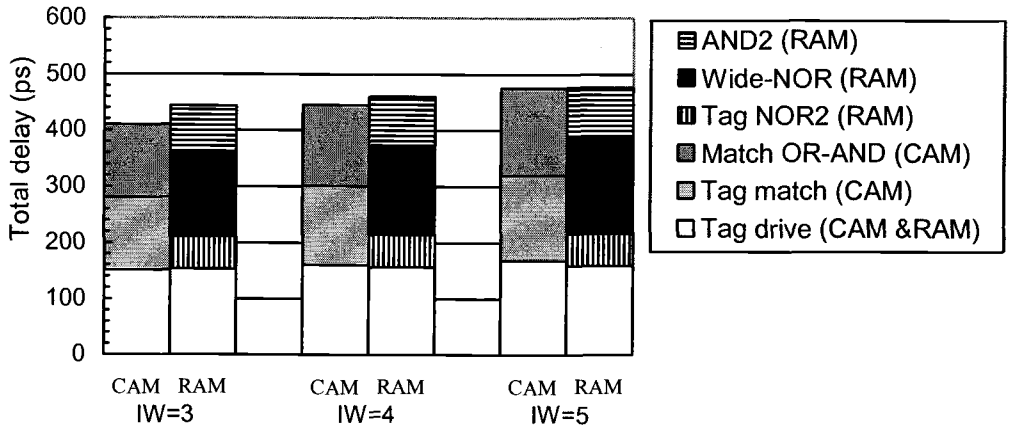


Figure 4.17 The total delay comparison of the CAM-type and RAM-type wakeup logic with various issue widths. This result is based on the configuration of 20-WSIZE and 32-NREG for 0.18μm technology.

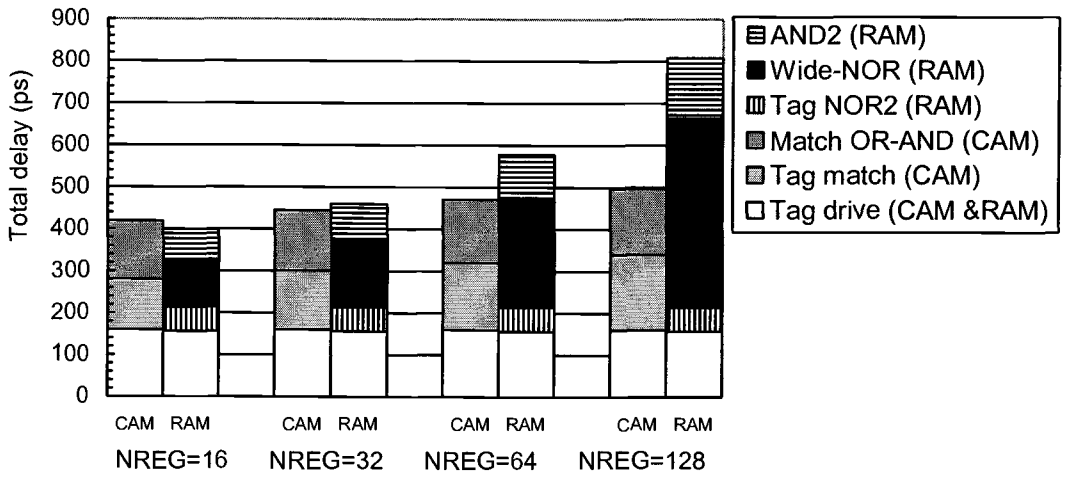


Figure 4.18 The total delay comparison of the CAM-type and RAM-type wakeup logic with various numbers of physical registers. This result is based on 20-WSIZE, 4-IW and 0.18μm technology.

Figure 4.16 shows the total delay comparison of the CAM-type and RAM-type wakeup logic with the design configuration of 4-IW and 32-NREG for 0.18μm technology. The total delays for both types of wakeup logic are almost equally increased as the window size increases because the tag drive times are almost equally increased as well. This means the

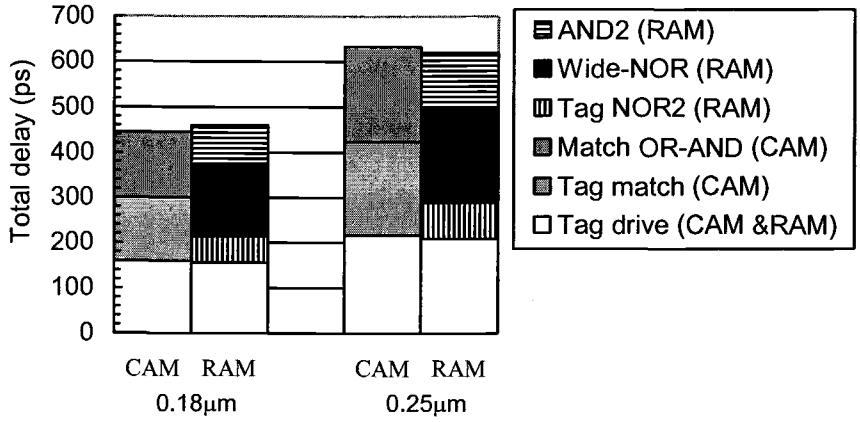


Figure 4.19 The total delay comparison of the CAM-type and RAM-type wakeup logic with various feature sizes. This result is based on the configuration of 20-WSIZE, 4-IW and 32-NREG.

performances of the tag drive circuit for both types of wakeup logic are equally sensitive to window size. Consider the delay comparison in figure 4.17 with the design configurations of 20-WSIZE and 32-NREG for 0.18μm technology, the total delay of the CAM-type wakeup logic is more sensitive to issue width than the RAM-type wakeup logic is. All the delay components for the CAM-type wakeup logic are increased faster compared with those for the RAM-type wakeup logic as the issue width increases. Figure 4.18 shows the delay comparison for the design configuration of 20-WSIZE and 4-IW for 0.18μm technology. This is the most interesting part of the comparison. When NREG is 16, the performance of the RAM-type wakeup logic is better than the CAM-type wakeup logic. As NREG increases, the wide-NOR delay for the RAM-type wakeup logic increases steeply. Finally, the performance of the CAM-type wakeup logic surpasses that of the RAM-type wakeup logic when NREG is greater than 32. It's a drawback for the RAM-type wakeup circuit to employ a large number of physical registers because the large fan-in for the wide-NOR gate will

introduce a huge parasitic capacitance to the output node of the wide-NOR gate. Figure 4.19 shows how the feature size (or technology) affects the total delays of the CAM-type and RAM-type wakeup logic. The CAM-type wakeup circuit has a greater benefit when the feature size is shrunk from 0.25μm to 0.18μm.

5 CONCLUSION

From the HSPICE simulation results shown in chapter 4, the performance of the CAM-type wakeup circuit is better than the RAM-type wakeup circuit if the number of physical registers employed by the processor is large. When the number of physical registers is increased from 32 to 128, the ratio of the RAM-type wakeup delay to the CAM-type wakeup delay is rapidly increased from 1.03 to 1.63 with the design configurations of 20-window size, 4-issue width and 0.18μm technology. Such performance degradation for the RAM-type wakeup logic is caused by the wide-NOR circuit delay that increases rapidly as the number of physical registers increases. Therefore, it's not appropriate to utilize the RAM-type wakeup circuit if the number of physical registers employed by the processor is large. The CAM-type wakeup logic delay is more sensitive to issue width because it increases more quickly than the RAM-type wakeup logic does as the issue width is increased. The CAM-type wakeup circuit can take a better advantage of the technology change because its delay decreases more than the RAM-type wakeup circuit does when the feature size shrinks from 0.25μm to 0.18μm. The delay impacts caused by altering the instruction window size are not much different for both types of wakeup logic implementations.

RAM-type wakeup circuit consumes much more silicon area than CAM-type wakeup circuit does when a large number of physical registers is employed. The layout areas for both types of wakeup logic implementations are very close with the configuration of

32-NREG. However, when the number of physical registers is increased to 64, the layout area of the RAM-type wakeup logic is almost twice as large as that for the CAM-type wakeup logic.

BIBLIOGRAPHY

[1] Intel Corporation, "IA-32 Intel Architecture Software Developer's Manual Volume 1: Basic Architecture," 2001.

[2] G. Hinton, D. Sager, M. Upton, D. Boggs, D. Carmean, A. Kyker, and P. Roussel, "The Microarchitecture of the Pentium 4 Processor," Intel Technical Journal, Feb. 2001. Q1 2001 Issue.

[3] S. Palacharla, N.P. Jouppi and J.E. Smith, "Complexity-Effective Superscalar Processors," Proc. of the Intl. Symp. on Computer Architecture, pp. 206-218, 1997.

[4] S. Palacharla, "Complexity-Effective Superscalar Processors," University of Wisconsin, Doctoral Dissertation, 1998.

[5] J. Stark, M.D. Brown and Y.N. Patt, "On pipelining Dynamic Instruction Scheduling Logic," Proc. of the international Symp. on Microarchitecture, pp. 57-66, 2000.

[6] M.D. Brown, J. Stark, and Y.N. Patt, "Select-Free Instruction Scheduling Logic," Proceedings of the 34th ACM/IEEE International Symposium on Microarchitecture, Austin, Texas, December 2001.

[7] E.Morancho, J. M. Llaberia, and A. Olive, "Recovery mechanism for latency misprediction," Proceedings of the 2001 ACM/IEEE International Conference on Parallel Architectures and Compilation Techniques, 2001.

[8] K.C. Yeager, "The MIPS 10000 Superscalar Microprocessor," IEEE MICRO, Vol. 16, n 2, pp 28-41, April 1996.

[9] R.E. Kessler, "The Alpha 21264 Microprocessor," IEEE Micro, 19 (2): 24 36, March/April 1999.

[10] J. E. Smith and G. S. Sohi, "The Microarchitecture of Superscalar Processors," Proceedings of the IEEE, vol. 83, pp 1609--1624, Dec 1995.

[11] Neil H. E. Weste, K. Eshraghian, "Principles of Cmos Vlsi Design," Addison-Wesley Pub Co, October, 1994.

[12] L. Gwennap, "Intel's P6 Uses Decoupled Superscalar Design," Microprocessor Report, Vol. 9, no. 2, February 16, 1995.

[13] S. P. Song et al, "The PowerPC 604 RISC Microprocessor," IEEE Micro, pp. 8-17, Oct. 1994.

[14] M. Johnson, " Superscalar Microprocessor Design," Prentice Hall, January 1991.

[15] TSMC (Taiwan Semiconductor Manufacture Company), www.tsmc.com.tw

[16] MOSIS organization, www.mosis.org

[17] A. Chandrakasan, W. J. Bowhill, and F. Fox, editors, "Design of High-Performance Microprocessor Circuits," IEEE Press, 2001.

[18] S-M Kang and Y. Leblebigi, "CMOS Digital Integrated Circuits-Analysis and Design," second edition, McGraw-Hill, 1999.

[19] Jan M. Rabaey, "Digital Integrated Circuits: A Design Perspective," Prentice Hall, December 1995.

[20] J. Martin, " Analog Integrated Circuit Design," John Wiley & Sons, Inc., 1997.

APPENDIX

APPENDIX TECHNOLOGY FILES

The technology files used for TSMC CMOS 0.18μm and 0.25μm technologies are listed in this appendix. Table 1 and Table 2 list the BSIM3 models of TSMC CMOS 0.18μm and 0.25μm processes used by the HSPICE simulator. Table 3 and Table 4 list the parametric test results used for calculating the parasitic capacitance and resistance of the interconnection lines.

Table 1. BSIM3 SPICE model for TSMC CMOS 0.18μm technology

```
* SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8
* DATE: Oct 17/01
* LOT: T18H                    WAF: 7006
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                    LEVEL    = 49
+VERSION = 3.1          TNOM    = 27                   TOX      = 4.2E-9
+XJ      = 1E-7         NCH     = 2.3549E17            VTH0     = 0.3710619
+K1      = 0.5940793    K2      = 2.070131E-3          K3       = 1E-3
+K3B     = 2.7158495    W0      = 1E-7                 NLX      = 2.005089E-7
+DVT0W   = 0            DVT1W   = 0                    DVT2W    = 0
+DVT0    = 1.4615376    DVT1    = 0.3798134            DVT2     = 0.0692378
+U0      = 293.522312   UA      = -6.73646E-10         UB       = 1.164182E-18
+UC      = -2.84532E-11 VSAT    = 9.286324E4           A0       = 1.7591856
+AGS     = 0.3162202    B0      = -5.950938E-8         B1       = -1E-7
+KETA    = 0.0111532    A1      = 3.896574E-4          A2       = 1
+RDSW    = 139.0465393  PRWG    = 0.5                  PRWB     = -0.2
+WR      = 1            WINT    = 0                    LINT     = 9.265899E-9
+XL      = -2E-8        XW      = -1E-8                DWG      = -1.343579E-9
+DWB     = -1.391607E-8 VOFF    = -0.0765575           NFACTOR  = 2.4791597
+CIT     = 0            CDSC    = 2.4E-4               CDSCD    = 0
+CDSCB   = 0            ETA0    = 0                    ETAB     = -0.0608407
+DSUB    = 1            PCLM    = 0.8853499            PDIBLC1  = 0.116863
+PDIBLC2 = 0.01         PDIBLCB = -0.0475298           DROUT    = 0.5922434
+PSCBE1  = 8E10         PSCBE2  = 5.248199E-10         PVAG     = 0.089248
+DELTA   = 0.01         RSH     = 6.8                  MOBMOD   = 1
+PRT     = 0            UTE     = -1.5                 KT1      = -0.11
+KT1L    = 0            KT2     = 0.022                UA1      = 4.31E-9
+UB1     = -7.61E-18    UC1     = -5.6E-11             AT       = 3.3E4
+WL      = 0            WLN     = 1                    WW       = 0
+WWN     = 1            WWL     = 0                    LL       = 0
+LLN     = 1            LW      = 0                    LWN      = 1
+LWL     = 0            CAPMOD  = 2                    XPART    = 0.5
+CGD0    = 7.75E-10     CGS0    = 7.75E-10             CGB0     = 1E-12
+CJ      = 9.955315E-4  PB      = 0.7345743            MJ       = 0.3629904
+CJSW    = 2.586055E-10 PBSW    = 0.6451808            MJSW     = 0.1296914
+CJSWG   = 3.3E-10      PBSWG   = 0.6451808            MJSWG    = 0.1296914
+CF      = 0            PVTH0   = 1.33957E-3           PRDSW    = -5
+PK2     = -1.7189E-4   WKETA   = 0.010864             LKETA    = -0.0102793
+PU0     = 37.4749547   PUA     = 1.762367E-10         PUB      = 9.411793E-25
```

```
+PVSAT    = 2E3              PETA0    = -1E-4          PKETA    = -1.356792E-3
+ACM=3 )
*
.MODEL CMOSP PMOS (                                   LEVEL    = 49
+VERSION = 3.1               TNOM     = 27             TOX      = 4.2E-9
+XJ       = 1E-7             NCH      = 4.1589E17      VTH0     = -0.4220357
+K1       = 0.5813738        K2       = 0.0303955      K3       = 0
+K3B      = 11.3426872       W0       = 1E-6           NLX      = 9.876034E-8
+DVTOW    = 0                DVT1W    = 0              DVT2W    = 0
+DVT0     = 0.5131166        DVT1     = 0.2665264      DVT2     = 0.1
+U0       = 120.5316596      UA       = 1.645481E-9    UB       = 1E-21
+UC       = -1E-10           VSAT     = 2E5            A0       = 1.671928
+AGS      = 0.3934127        B0       = 1.830733E-6    B1       = 4.739218E-6
+KETA     = 0.0202801        A1       = 0.1976849      A2       = 0.5787213
+RDSW     = 265.2609374      PRWG     = 0.5            PRWB     = -0.2145086
+WR       = 1                WINT     = 0              LINT     = 2.176517E-8
+XL       = -2E-8            XW       = -1E-8          DWG      = -4.223522E-8
+DWB      = 7.670464E-9      VOFF     = -0.096172      NFACTOR  = 2
+CIT      = 0                CDSC     = 2.4E-4         CDSCD    = 0
+CDSCB    = 0                ETA0     = 0.023671       ETAB     = -0.3005133
+DSUB     = 1.2320494        PCLM     = 2.2844319      PDIBLC1  = 4.836921E-3
+PDIBLC2  = 0.0442167        PDIBLCB  = -1E-3          DROUT    = 9.991187E-4
+PSCBE1   = 1.732893E9       PSCBE2   = 5E-10          PVAG     = 14.9616148
+DELTA    = 0.01             RSH      = 7.6            MOBMOD   = 1
+PRT      = 0                UTE      = -1.5           KT1      = -0.11
+KT1L     = 0                KT2      = 0.022          UA1      = 4.31E-9
+UB1      = -7.61E-18        UC1      = -5.6E-11       AT       = 3.3E4
+WL       = 0                WLN      = 1              WW       = 0
+WWN      = 1                WWL      = 0              LL       = 0
+LLN      = 1                LW       = 0              LWN      = 1
+LWL      = 0                CAPMOD   = 2              XPART    = 0.5
+CGDO     = 6.6E-10          CGSO     = 6.6E-10        CGBO     = 1E-12
+CJ       = 1.183858E-3      PB       = 0.8534482      MJ       = 0.4124158
+CJSW     = 2.066263E-10     PBSW     = 0.6189346      MJSW     = 0.2893774
+CJSWG    = 4.22E-10         PBSWG    = 0.6189346      MJSWG    = 0.2893774
+CF       = 0                PVTH0    = 2.308546E-3    PRDSW    = 13.6874174
+PK2      = 2.657069E-3      WKETA    = 2.467864E-3    LKETA    = -2.56649E-3
+PU0      = -1.846164        PUA      = -8.06063E-11   PUB      = 1E-21
+PVSAT    = -50              PETA0    = 1E-4           PKETA    = 2.794471E-3
+ACM=3 )
*
```

Table 2. BSIM3 SPICE model for TSMC CMOS 0.25μm technology

```
* SPICE 3f5 Level 8, Star-HSPICE Level 49, UTMOST Level 8
* DATE: Oct 17/01
* LOT: T18I                    WAF: 1006
* Temperature_parameters=Default
.MODEL CMOSN NMOS (                                   LEVEL    = 49
+VERSION = 3.1               TNOM     = 27             TOX      = 5.6E-9
+XJ       = 1E-7             NCH      = 2.3549E17      VTH0     = 0.3630413
+K1       = 0.4547995        K2       = 3.68668E-3     K3       = 1E-3
+K3B      = 3.0944722        W0       = 1E-7           NLX      = 2.399261E-7
+DVTOW    = 0                DVT1W    = 0              DVT2W    = 0
+DVT0     = 0.3716693        DVT1     = 0.4697927      DVT2     = -0.5
+U0       = 291.4385527      UA       = -1.376383E-9   UB       = 2.628885E-18
+UC       = 3.988729E-11     VSAT     = 1.512228E5     A0       = 1.6710899
+AGS      = 0.314112         B0       = -1.304764E-7   B1       = 8.598609E-7
```

```
+KETA     = -5.981033E-3   A1       = 0              A2       = 0.4374119
+RDSW     = 144.0476408    PRWG     = 0.5            PRWB     = -0.2
+WR       = 1              WINT     = 0              LINT     = 1.567069E-8
+XL       = 3E-8           XW       = -4E-8          DWG      = -2.093584E-8
+DWB      = 1.756252E-9    VOFF     = -0.0987789     NFACTOR  = 1.5643085
+CIT      = 0              CDSC     = 2.4E-4         CDSCD    = 0
+CDSCB    = 0              ETA0     = 4.367542E-3    ETAB     = 7.052956E-4
+DSUB     = 0.029053       PCLM     = 1.8429314      PDIBLC1  = 0.9990675
+PDIBLC2  = 5.063115E-3    PDIBLCB  = -0.0999223     DROUT    = 0.9540249
+PSCBE1   = 7.973869E10    PSCBE2   = 5.194639E-10   PVAG     = 0.014452
+DELTA    = 0.01           RSH      = 4.9            MOBMOD   = 1
+PRT      = 0              UTE      = -1.5           KT1      = -0.11
+KT1L     = 0              KT2      = 0.022          UA1      = 4.31E-9
+UB1      = -7.61E-18      UC1      = -5.6E-11       AT       = 3.3E4
+WL       = 0              WLN      = 1              WW       = 0
+WWN      = 1              WWL      = 0              LL       = 0
+LLN      = 1              LW       = 0              LWN      = 1
+LWL      = 0              CAPMOD   = 2              XPART    = 0.5
+CGDO     = 5.99E-10       CGSO     = 5.99E-10       CGBO     = 1E-12
+CJ       = 1.747074E-3    PB       = 0.9862514      MJ       = 0.4566299
+CJSW     = 4.077326E-10   PBSW     = 0.99           MJSW     = 0.3371634
+CJSWG    = 3.29E-10       PBSWG    = 0.99           MJSWG    = 0.3371634
+CF       = 0              PVTH0    = -0.01          PRDSW    = -10
+PK2      = 2.665323E-3    WKETA    = 6.864037E-3    LKETA    = -7.793877E-3    )
*
.MODEL CMOSP PMOS (                                  LEVEL    = 49
+VERSION  = 3.1            TNOM     = 27             TOX      = 5.6E-9
+XJ       = 1E-7           NCH      = 4.1589E17      VTH0     = -0.5467919
+K1       = 0.6050815      K2       = 5.207745E-3    K3       = 0
+K3B      = 12.6429662     W0       = 1E-6           NLX      = 1E-9
+DVTOW    = 0              DVT1W    = 0              DVT2W    = 0
+DVT0     = 4.5811661      DVT1     = 0.8799706      DVT2     = -0.092277
+U0       = 105.1637464    UA       = 1.126619E-9    UB       = 1E-21
+UC       = -1E-10         VSAT     = 2E5            A0       = 0.8989949
+AGS      = 0.1313337      B0       = 1.10465E-6     B1       = 5E-6
+KETA     = 0.0143617      A1       = 6.14246E-3     A2       = 0.3821788
+RDSW     = 849.5361721    PRWG     = 0.2872539      PRWB     = -0.2476987
+WR       = 1              WINT     = 0              LINT     = 3.879217E-8
+XL       = 3E-8           XW       = -4E-8          DWG      = -4.532087E-8
+DWB      = 9.361442E-10   VOFF     = -0.1109244     NFACTOR  = 1.0687326
+CIT      = 0              CDSC     = 2.4E-4         CDSCD    = 0
+CDSCB    = 0              ETA0     = 0.6184712      ETAB     = -0.4263845
+DSUB     = 1.1618544      PCLM     = 1.1509108      PDIBLC1  = 5.480515E-3
+PDIBLC2  = 5.321941E-5    PDIBLCB  = -9.475839E-4    DROUT    = 0.0589831
+PSCBE1   = 3.665084E10    PSCBE2   = 2.974201E-9    PVAG     = 0.0145235
+DELTA    = 0.01           RSH      = 3.6            MOBMOD   = 1
+PRT      = 0              UTE      = -1.5           KT1      = -0.11
+KT1L     = 0              KT2      = 0.022          UA1      = 4.31E-9
+UB1      = -7.61E-18      UC1      = -5.6E-11       AT       = 3.3E4
+WL       = 0              WLN      = 1              WW       = 0
+WWN      = 1              WWL      = 0              LL       = 0
+LLN      = 1              LW       = 0              LWN      = 1
+LWL      = 0              CAPMOD   = 2              XPART    = 0.5
+CGDO     = 6.84E-10       CGSO     = 6.84E-10       CGBO     = 1E-12
+CJ       = 1.902469E-3    PB       = 0.99           MJ       = 0.4661629
+CJSW     = 3.191262E-10   PBSW     = 0.5821134      MJSW     = 0.283689
+CJSWG    = 2.5E-10        PBSWG    = 0.5821134      MJSWG    = 0.283689
+CF       = 0              PVTH0    = 6.249528E-3    PRDSW    = -4.0136946
+PK2      = 3.252073E-3    WKETA    = 0.0308356      LKETA    = -6.54006E-3     )
```

Table 3. Parametric test result for TSMC CMOS 0.18μm technology

MOSIS PARAMETRIC TEST RESULTS
RUN: T18H (LO_EPI) VENDOR: TSMC
TECHNOLOGY: SCN018 FEATURE SIZE: 0.18 microns

INTRODUCTION: This report contains the lot average results obtained by MOSIS from measurements of MOSIS test structures on each wafer of this fabrication lot. SPICE parameters obtained from similar measurements on a selected wafer are also attached.

COMMENTS: DSCN6M018_TSMC

TRANSISTOR PARAMETERS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	0.27/0.18			
Vth		0.54	-0.54	volts
SHORT	20.0/0.18			
Idss		530	-268	uA/um
Vth		0.55	-0.54	volts
Vpt		4.7	-5.4	volts
WIDE	20.0/0.18			
Ids0		7.5	-5.4	pA/um
LARGE	50/50			
Vth		0.45	-0.44	volts
Vjbkd		3.8	-5.0	volts
Ijlk		<50.0	<50.0	pA
Gamma		0.55	0.63	V^0.5
K' (Uo*Cox/2)		167.5	-35.5	uA/V^2
Low-field Mobility		407.46	86.36	cm^2/V*s

COMMENTS: Poly bias varies with design technology. To account for mask and etch bias use the appropriate value for the parameters XL and XW in your SPICE model card.

Design Technology	XL	XW
SCN6M_DEEP (lambda=0.09)	-0.02	-0.01
thick oxide	-0.03	-0.01
TSMC18	-0.02	0.00
thick oxide	-0.02	0.00
SCN6M_SUBM (lambda=0.10)	-0.04	0.00
thick oxide	-0.07	0.00

FOX TRANSISTORS	GATE	N+ACTIVE	P+ACTIVE	UNITS
Vth	Poly	>6.6	<-6.6	volts

PROCESS PARAMETERS	N+ACTV	P+ACTV	POLY	N+BLK	PLY+BLK	MTL1	MTL2	UNITS
Sheet Resistance	6.8	7.6	7.9	60.5	339.4	0.08	0.08	ohms/sq
Contact Resistance	10.8	11.3	10.0				6.58	ohms
Gate Oxide Thickness	42							angstrom

PROCESS PARAMETERS	MTL3	MTL4	MTL5	MTL6	N_WELL	UNITS
Sheet Resistance	0.08	0.08	0.08	0.03	981	ohms/sq
Contact Resistance	13.75	19.26	25.24	27.09		ohms

COMMENTS: BLK is silicide block.

CAPACITANCE PARAMETERS	N+ACTV	P+ACTV	POLY	M1	M2	M3	M4	M5	M6	N_WELL	UNITS
Area (substrate)	1000	1182	95	37	18	13	8	8	3	71	aF/um^2
Area (N+active)			8275	50	19	13	10	9	8		aF/um^2
Area (P+active)			8046								aF/um^2
Area (poly)				61	16	10	7	5	4		aF/um^2
Area (metal1)					36	14	9	6	5		aF/um^2
Area (metal2)						38	14	9	6		aF/um^2
Area (metal3)							36	14	8		aF/um^2
Area (metal4)								37	13		aF/um^2
Area (metal5)									33		aF/um^2
Area (no well)	146										aF/um^2
Fringe (substrate)	262	218		16	58	53	41	23	--		aF/um
Fringe (poly)				66	38	28	23	20	17		aF/um
Fringe (metal1)					50	33		22	18		aF/um
Fringe (metal2)						47	35	27	22		aF/um
Fringe (metal3)							53	34	27		aF/um
Fringe (metal4)								58	35		aF/um
Fringe (metal5)									52		aF/um
Overlap (P+active)			660								aF/um

CIRCUIT PARAMETERS			UNITS
Inverters	K		
Vinv	1.0	0.77	volts
Vinv	1.5	0.81	volts
Vol (100 uA)	2.0	0.08	volts
Voh (100 uA)	2.0	1.64	volts
Vinv	2.0	0.83	volts
Gain	2.0	-23.55	
Ring Oscillator Freq.			
D1024_THK (31-stg,3.3V)		331.03	MHz
DIV1024 (31-stg,1.8V)		384.10	MHz
Ring Oscillator Power			
D1024_THK (31-stg,3.3V)		0.07	uW/MHz/gate
DIV1024 (31-stg,1.8V)		0.02	uW/MHz/gate

COMMENTS: DEEP_SUBMICRON

Table 4. Parametric test result for TSMC CMOS 0.25μm technology

MOSIS PARAMETRIC TEST RESULTS

RUN: T18I (MM_NON-EPI) VENDOR: TSMC
TECHNOLOGY: SCN025 FEATURE SIZE: 0.25 microns

INTRODUCTION: This report contains the lot average results obtained by MOSIS from measurements of MOSIS test structures on each wafer of this fabrication lot. SPICE parameters obtained from similar measurements on a selected wafer are also attached.

TRANSISTOR PARAMETERS	W/L	N-CHANNEL	P-CHANNEL	UNITS
MINIMUM	0.36/0.24			
Vth		0.48	-0.46	volts

SHORT	20.0/0.24			
Idss		579	-282	uA/um
Vth		0.50	-0.50	volts
Vpt		7.5	-7.2	volts
WIDE	20.0/0.24			
Ids0		11.1	-3.6	pA/um
LARGE	50/50			
Vth		0.42	-0.56	volts
Vjbkd		5.8	-7.0	volts
Ijlk		<50.0	<50.0	pA
Gamma		0.44	0.60	V^0.5
K' (Uo*Cox/2)		124.7	-25.8	uA/V^2
Low-field Mobility		404.46	83.68	cm^2/V*s

COMMENTS: Poly bias varies with design technology. To account for mask and etch bias use the appropriate value for the parameters XL and XW in your SPICE model card.

Design Technology	XL	XW
SCN5M_DEEP (lambda=0.12)	0.03	-0.04
thick oxide, NMOS	0.02	-0.04
thick oxide, PMOS	-0.03	-0.04
TSMC25	0.03	0.00
thick oxide, NMOS	0.03	0.00
thick oxide, PMOS	0.03	0.00
SCN5M_SUBM (lambda=0.15)	-0.03	0.00
thick oxide, NMOS	0.02	0.00
thick oxide, PMOS	-0.03	0.00

FOX TRANSISTORS	GATE	N+ACTIVE	P+ACTIVE	UNITS
Vth	Poly	>6.6	<-6.6	volts

PROCESS PARAMETERS	N+ACTV	P+ACTV	POLY	N+BLK	PLY+BLK	MTL1	MTL2	UNITS
Sheet Resistance	4.9	3.6	4.3	58.3	174.2	0.07	0.07	ohms/sq
Contact Resistance	5.7	4.7	4.9				2.58	ohms
Gate Oxide Thickness	56							angstrom

PROCESS PARAMETERS	MTL3	MTL4	MTL5	N_WELL	UNITS
Sheet Resistance	0.07	0.07	0.03	1054	ohms/sq
Contact Resistance	5.13	7.73	10.11		ohms

COMMENTS: BLK is silicide block.

CAPACITANCE PARAMETERS	N+ACTV	P+ACTV	POLY	M1	M2	M3	M4	M5	M4P	N_WELL	UNITS
Area (substrate)	1739	1897	102	38	18	13	8	8		59	aF/um^2
Area (N+active)			6199	54	21	15	12	10			aF/um^2
Area (P+active)			5919								aF/um^2
Area (poly)				64	18	11	8	6			aF/um^2
Area (metal1)					40	16	10	7			aF/um^2
Area (metal2)						45	16	10			aF/um^2
Area (metal3)							44	17			aF/um^2
Area (metal4)								45	1003		aF/um^2
Area (no well)	236										aF/um^2
Fringe (substrate)	422	344		23	58	53	39	28			aF/um
Fringe (poly)				75	42	32	26	22			aF/um
Fringe (metal1)					60	39		25			aF/um

Fringe (metal2)			56 39 31	aF/um
Fringe (metal3)			58 40	aF/um
Fringe (metal4)			67	aF/um
Overlap (N+active)		599		aF/um
Overlap (P+active)		684		aF/um

CIRCUIT PARAMETERS			UNITS
Inverters	K		
Vinv	1.0	1.00	volts
Vinv	1.5	1.09	volts
Vol (100 uA)	2.0	0.18	volts
Voh (100 uA)	2.0	2.10	volts
Vinv	2.0	1.15	volts
Gain	2.0	-16.68	
Ring Oscillator Freq.			
D1024_THK (31-stg,3.3V)		211.44	MHz
DIV1024 (31-stg,2.5V)		284.15	MHz
Ring Oscillator Power			
D1024_THK (31-stg,3.3V)		0.10	uW/MHz/gate
DIV1024 (31-stg,2.5V)		0.06	uW/MHz/gate

COMMENTS: DEEP_SUBMICRON