บทที่ 6
ผลการทดลอง

ในบทนี้จะกล่าวถึง Benchmark ที่ใช้ในการทดลอง และผลการวัดสมรรถนะ ได้แก่ ค่า load miss ratio ที่โหมดการเขียนแบบ write back และ write through, ผลการเพิ่มสมรรถนะในการประมวลผลเมื่อใช้ non-blocking load ในหน่วย LMC (Load Miss Control), ผลการวัดสมรรถนะของหน่วย SHC (Store Hit Control) หลังจากการเพิ่มขนาดบัฟเฟอร์ และผลการวัดสมรรถนะของหน่วย SMC (Store Miss Control) หลังการเปลี่ยนแปลงขนาด FIFO และผลการสังเคราะห์วงจร LSU

6.1 Benchmark ที่ใช้ในการทดลอง

Benchmark คือโปรแกรมที่ใช้ในการสร้าง Trace เพื่อใช้ในการทดสอบสมรรถนะการทำงานของวงจรที่ออกแบบ ภายในโปรแกรม Trace จะประกอบไปด้วยใช้ชุดคำสั่งของไมโครโพรเซสเซอร์ในตระกูล MIPS ของบริษัท LSI Logic โดยวิธีการสร้าง Trace ที่ใช้ในงานวิจัยนี้ได้กล่าวไว้แล้วในบทที่ 5

Benchmark ที่ใช้ แสดงข้อมูลไว้ในตารางที่ 6.1 แสดงจำนวนคำสั่งที่สามารถประมวลใน LSU เท่านั้น

Benchmark	gzip	gunzip	diff	compress	uncompress	cc
LB	52	1061	1057	16661	13702	6042
LBU	12686	3262	5133	55100	154951	26993
LH	0	0	1	0	0	0
LHU	9904	775	138	34744	51643	8396
LW	32471	16452	20569	629296	523535	244704
LWL	0	0	43	0	0	4
LWR	10	9	46	2	3	630
SB	1576	185	67	24936	129955	880
SH	4573	33	0	6648	8847	0
SW	19004	5360	9817	237949	125180	93653
SWL	0	0	0	0	0	2
SWR	0	0	0	0	0	2
ADD	4	91	34	2	1111	472
All Load	55123	21559	26987	735803	743834	286769
% of load Instruction	*21.53%*	*25.91%*	*27.53%*	*31.77%*	*31.12%*	*29.43%*
All Store	25153	5578	9884	269533	263982	94537
% of store Instruction	*9.83%*	*6.70%*	*10.08%*	*11.64%*	*11.04%*	*9.70%*
% of LSU Instruction	*31.36%*	*32.61%*	*37.62%*	*43.41%*	*42.16%*	*39.13%*
All Instructions	255978	83222	98014	2315784	2390199	974567

ตารางที่ 6.1 แสดงจำนวนคำสั่งของแต่ละ Benchmark ที่ใช้ในงานวิจัยนี้

6.2 ผลการวัด load miss ratio

ปัญหาที่สำคัญประการหนึ่งในการจำลองการทำงานของแคช คือเรื่องของ Cache Initialization ซึ่งทำให้ผลการทดสอบสมรรถนะการทำงานของแคชเพี้ยนไปจากความเป็นจริง กล่าวคือปัญหาดังกล่าวเกิดขึ้นเนื่องจากในตอนเริ่มต้นของการทดสอบการทำงานของแคชทุกครั้งจะเกิด Cache Miss เพื่อเขียนข้อมูลที่ได้จากหน่วยความจำลงในแคชก่อน เพราะข้อมูลในแคชทุกบรรทัดยังคงไม่มีอยู่จริงในตอนเริ่มต้น ซึ่งในสถานการณ์จริงที่ใช้งาน แคชทำหน้าที่เก็บข้อมูลส่วนที่ถูกใช้บ่อย ดังนั้นข้อมูลที่อ้างถึงควรจะให้ผลการอ่านแคชเป็น Hit ดังนั้นในช่วงเริ่มของการทดสอบแคชที่เกิด Miss ขึ้นนั้นควรจะนับเป็น Hit

และเนื่องจากงานวิจัยนี้ใช้ขนาด Trace ที่ได้จากการรันโปรแกรม Benchmark มีขนาดไม่ใหญ่มากนัก ดังนั้นในช่วงเริ่มต้นของการทดสอบจะได้ Miss Ratio ที่ได้สูงมาก ผู้วิจัยจึงได้ทำการตัดค่า Miss ส่วนนี้ออกโดยการพิจารณาที่ Valid bit ของ D-Cache Tag ใน Set ที่มีการใช้งานว่าขณะที่เกิดการ Miss ว่าถูกตั้งค่าเป็น '1' ไว้แล้วหรือยัง ถ้าค่า Valid bit ยังคงมีค่าเป็น '0' อยู่ก็แสดงว่าขณะนั้นอยู่ในช่วง Cache Initialization ยังไม่ต้องนับว่าเป็น miss

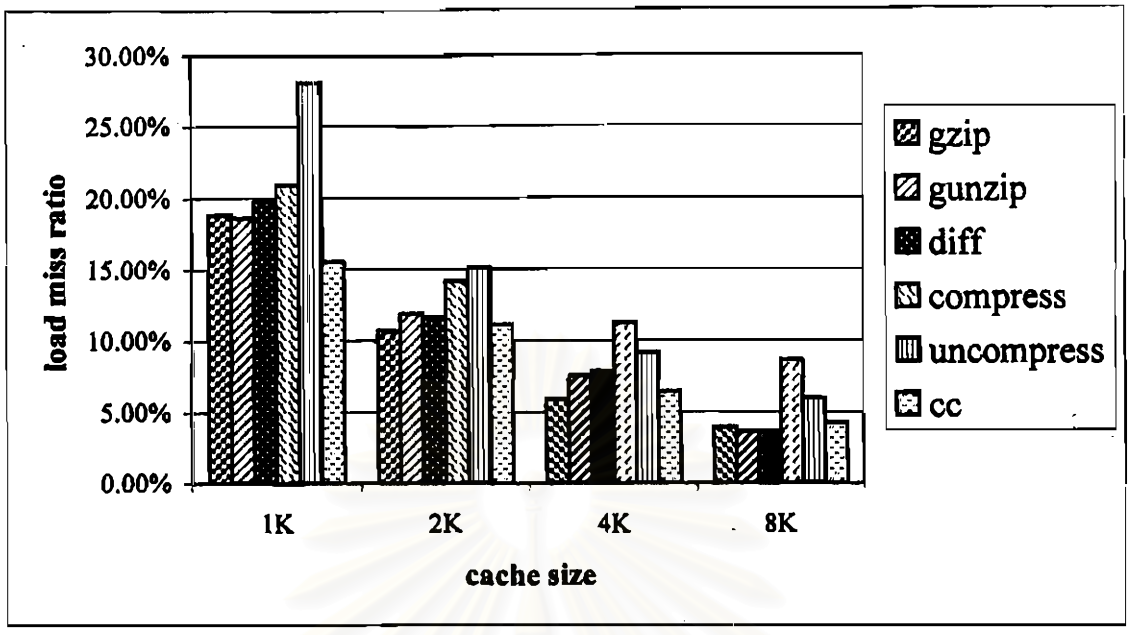
ผู้วิจัยทำการวัดค่า Load Miss Ratio ที่โหมดการเขียนแบบ Write Back และ Write Through สำหรับการจัดแคชแบบ Direct-Map และ 2-way Set Associative โดยสูตรการคำนวณค่า Load Miss Ratio คำนวณได้จากสมการที่ 6.1

$$\text{load miss ratio (\%)} = \frac{\text{จำนวนครั้งที่เกิด load miss}}{\text{จำนวนคำสั่ง load}} * 100 \quad \text{.....(6.1)}$$

กราฟในรูปที่ 6.1 แสดงผลการวัดค่า Load Miss Ratio ที่มีการจัดแคชเป็นแบบ Direct-Map โดยขนาดของแคชเปลี่ยนแปลงจาก 1K, 2K, 4K และ 8Kbytes ซึ่งไม่ว่าจะใช้โหมดการเขียนเป็นแบบ Write Back หรือว่า Write Through ก็ตาม กราฟที่ได้จะออกมามีลักษณะเหมือนกัน เนื่องจากการจัดแคชเป็นแบบ Direct-Map ดังนั้นการตั้งค่า WB bit ของโหมดการเขียนแบบ Write Back ไม่ส่งผลต่อการเลือกเซตในกรณีแคชถูกแทนเมื่อเกิดเหตุการณ์ Load Miss ขึ้น การอ้างอิงแคชที่ใช้โหมดการเขียนแบบ Write Back จึงเหมือนกับการอ้างอิงแคชที่ใช้โหมดการเขียนเป็นแบบ Write Through

กราฟในรูปที่ 6.2 และ 6.3 แสดงผลการวัดค่า Load Miss Ratio ที่มีการจัดแคชเป็นแบบ 2-way Set Associative โดยขนาดของแคชเปลี่ยนแปลงจาก 2K, 4K, 8K และ 16 Kbytes โดยโหมดการเขียนเป็น Write Back และ Write Through ตามลำดับ ซึ่งกราฟทั้งสองที่ได้ยังคงมีลักษณะคล้ายกัน แต่มีค่าต่างกันเล็กน้อย เพราะโหมดการเขียนแบบ Write Through ไม่มีการตั้งค่า WB bit ใน D-Cache Tag ซึ่งมีผลต่อการเลือกการถูกแทนที่ของแคช

และจะเห็นได้ว่าเมื่อทำการเปรียบเทียบค่า Load Miss Ratio ของการจัดแคชแบบ Direct-Map และ 2-way Set Associative ที่ขนาดแคชเท่ากัน ผลจากการจัดแคชแบบ 2-way Set Associative จะให้ค่า Load Miss Ratio ที่ต่ำกว่า



รูปที่ 6.1 ผลการวัดค่า load miss ratio เมื่อมีการจัดแคชเป็นแบบ direct-map โดยขนาดของแคชเปลี่ยนแปลงจาก 1K , 2K 4K และ 8Kbytes (Write Back และ Write Through)

6.3 ผลการวัดสมรรถนะของ non-blocking load ในหน่วย LMC

หน่วย LMC (Load Miss Control) ช่วยเพิ่มสมรรถนะในส่วนของการประมวลผลคำสั่ง Load ที่เกิดการ Miss โดยภายในมีการทำงานเป็น State Machine ตามที่ได้กล่าวไว้แล้วในบทที่ 4 การทำงานในหน่วย LMC เป็นการทำงานที่เรียกว่า Non-Blocking Load โดยขณะที่กำลังประมวลคำสั่ง Load ที่เกิดการ Miss อยู่ การทำงานของ Pipeline ก็ยังคงดำเนินต่อไปได้ ตราบใดที่การทำงานของ Pipeline ไม่เกิด Data Hazard ขึ้น หรือตราบใดที่ยังไม่มีการประมวลผลคำสั่ง Load ซ้อนเข้ามาอีก ซึ่งถ้าหากเป็นการทำงานแบบ Blocking Load การทำงานของ Pipeline จะต้อง "STALL" ไปจนกว่าการทำงานในคำสั่ง Load ที่เกิดการ Miss นั้นทำงานเสร็จ ดังนั้นการทำงานแบบ Non-Blocking Load นี้จะช่วยลดการเกิด "STALL" Pipeline อันเนื่องมาจากการรอข้อมูลที่ต้องการ Load

ผู้วิจัยได้ทำการวัดสมรรถนะของ Non-Blocking Load ในหน่วย LMC โดยหาเปอร์เซนต์ความแตกต่างของจำนวนครั้งที่เกิดการ "STALL" Pipeline อันเนื่องมาจาก Blocking Load กับ Non-Blocking Load ซึ่งงานวิจัยนี้ทำแต่ในส่วนของ Non-Blocking Load แต่สามารถหาจำนวนครั้งของการเกิด "STALL" Pipeline ของ Blocking Load ได้จากการนับจำนวนครั้งที่ State การทำงานในหน่วย LMC ไม่เท่ากับ "iDLE" State เพราะการที่ LMC ทำงานอยู่ใน State อื่นแสดงว่า LMC ยังมีการประมวลผล Load Miss ค้างอยู่

ดังนั้นการทำ Non-Blocking Load สามารถลดจำนวนการเกิด "STALL" Pipeline ลงไปได้ โดยจะวัดเป็นเปอร์เซนต์ของค่า reduce ตามสมการที่ 6.2

$$\text{reduce (\%)} = \frac{\text{stall (blocking load)} - \text{stall (non-blocking load)}}{\text{stall (blocking load)}} * 100 \quad \text{.....(6.2)}$$

โดยค่า stall (blocking load) ได้จากการนับจำนวน LMCstate ไม่เท่ากับ "iDLE" State

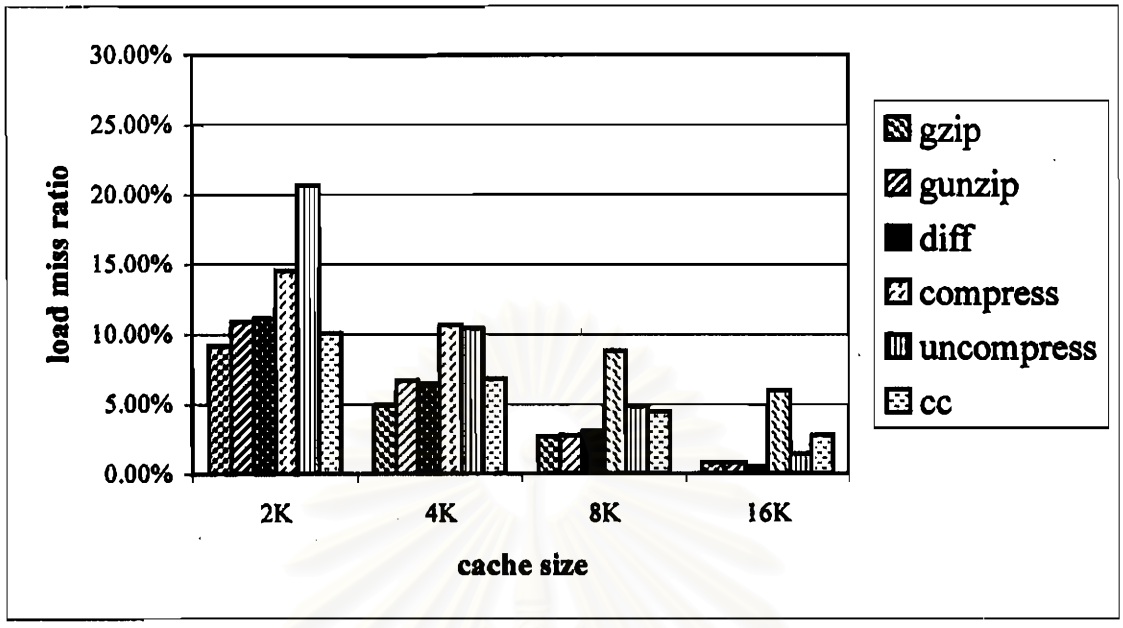
stall (non-blocking load) ได้จากการนับเหตุการณ์ที่เกิด "STALL" Pipeline จากการทำงานในส่วน LMC ที่เกิดขึ้นจริง ขณะทดลอง

กราฟในรูปที่ 6.4, 6.5, 6.6 และ 6.7 เป็นผลจากการวัดเปอร์เซนต์ที่ลดลงของจำนวนครั้งของการเกิด "STALL" Pipeline อันเนื่องมาจากการรอการประมวลผลคำสั่ง Load ที่เกิดการ Miss สำหรับการทำงานเป็น Non-Blocking Load โดยกราฟในรูปที่ 6.4 และ 6.5 แสดงเปอร์เซนต์ที่ลดลง (reduce) สำหรับการจัดแคชเป็นแบบ Direct-Map ที่ขนาดแคช 1K, 2K, 4K และ 8Kbytes โหมดการเขียนเป็น Write Back และ Write Through ตามลำดับ

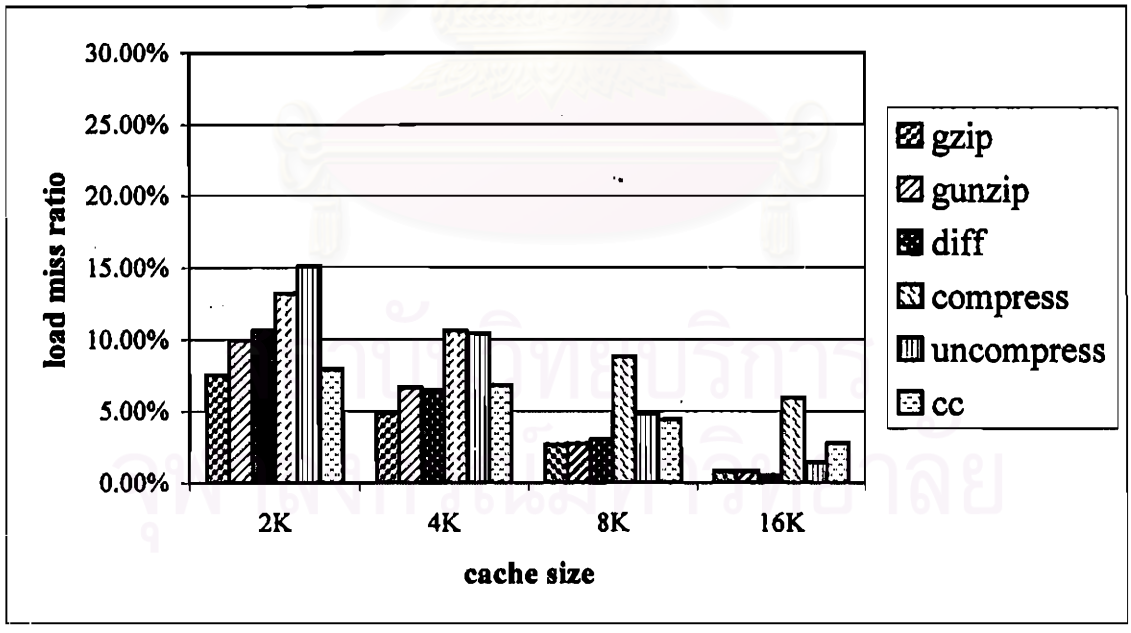
กราฟในรูปที่ 6.6 และ 6.7 แสดงเปอร์เซนต์ที่ลดลง สำหรับการจัดแคชเป็นแบบ 2-Way Set Associative ที่ขนาดแคช 2K, 4K, 8K และ 16Kbytes โหมดการเขียนเป็น Write Back และ Write Through ตามลำดับ

จากกราฟทั้งสี่สามารถสรุปได้เป็น 2 ประเด็นคือ

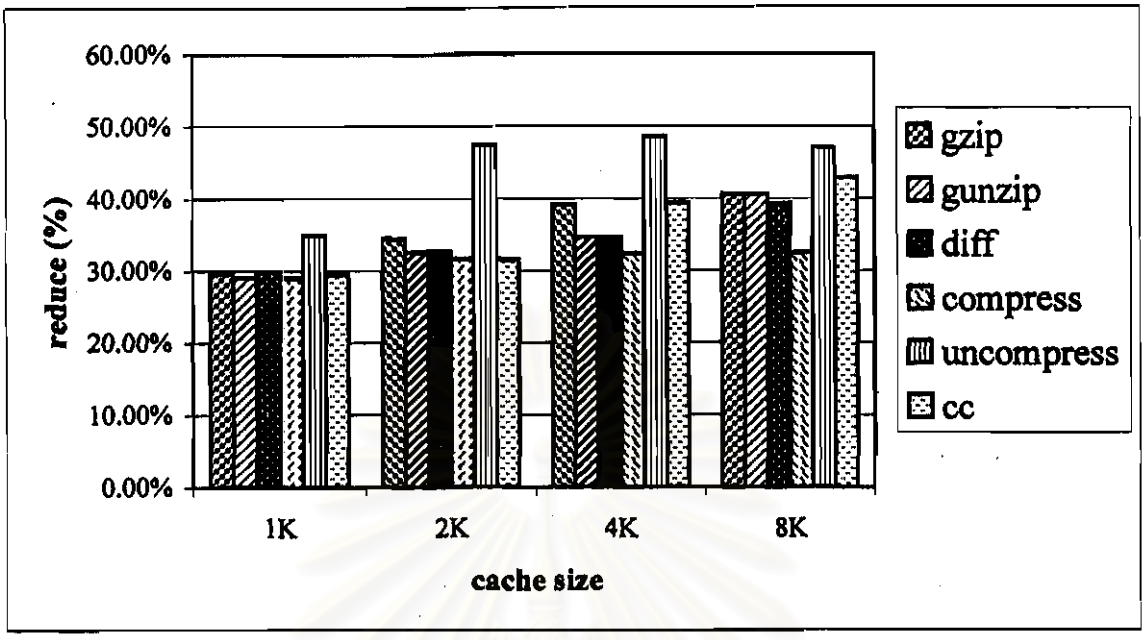
1) ถ้าขนาดแคชใหญ่ขึ้น ทำให้ค่า reduce ที่ได้เพิ่มขึ้น ทำให้สมรรถนะในการทำงานดีขึ้น และการจัดแคชแบบ 2-Way Set Associative ได้ค่า reduce ที่สูงกว่าการจัดแคชแบบ Direct-Map เหตุผลที่เป็นเช่นนี้เพราะว่า การเพิ่มขนาดของแคชและการเพิ่ม Degree of Set Associative ทำให้ค่า Load Miss Ratio ลดลง ส่งผลให้การประมวลผลในคำสั่ง Load กระทบต่อการทำงานใน LMC น้อยลง จึงเกิดเหตุการณ์ "STALL" Pipeline น้อยลงนั้นเอง
2) โหมดการเขียนแบบ Write Through ให้ค่า reduce สูงกว่าโหมดการเขียนแบบ Write Back เล็กน้อย (ประมาณ 1-2%) เหตุผลเนื่องมาจาก โหมดการเขียนแบบ Write Through ไม่มีการตั้งค่า WB bit จึงไม่มีการเขียนข้อมูลกลับไปยังหน่วยความจำหลักเมื่อมีการแทนที่แคช ดังนั้นค่า stall (blocking load) และค่า stall (non-blocking load) ของโหมดการเขียนแบบ Write Through มีค่าต่ำกว่าของโหมดการเขียนแบบ Write Back ทำให้ผลต่างของค่า stall ใกล้เคียงกัน แต่มีตัวหารที่น้อยกว่า เปอร์เซนต์ของค่า reduce ที่ออกมาของโหมดการเขียนแบบ Write Through จึงสูงกว่าเปอร์เซนต์ค่า reduce ที่โหมดการเขียนแบบ Write Back เล็กน้อย



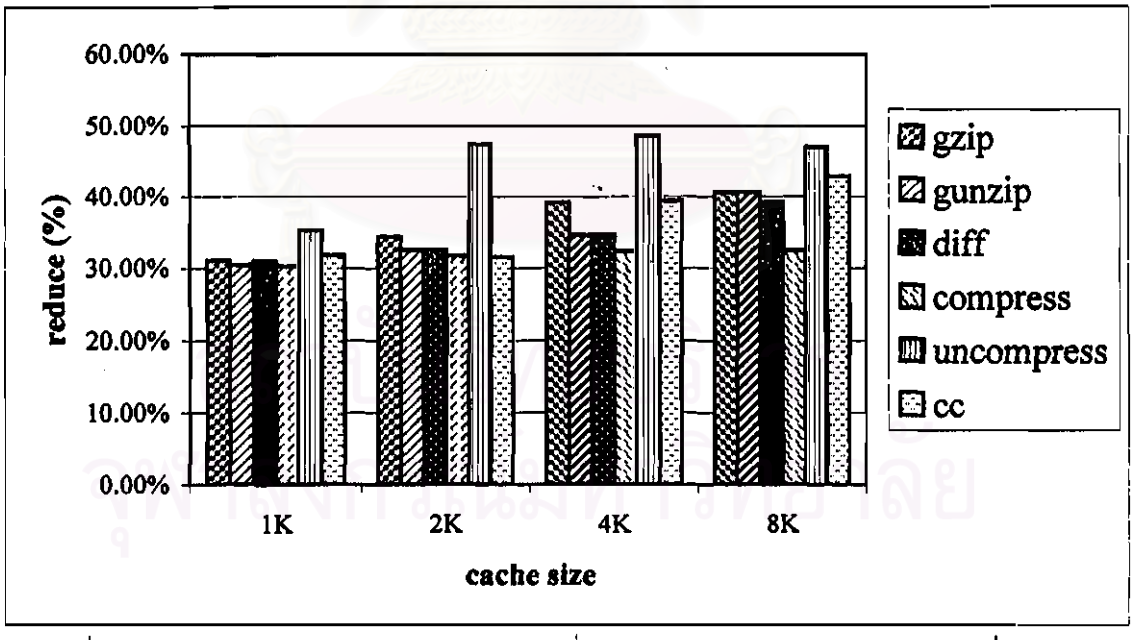
รูปที่ 6.2 ผลการวัดค่า load miss ratio เมื่อใช้โหมดการเขียนเป็น Write Back และมีการจัดแคชเป็นแบบ 2-way set associative โดยขนาดของแคชเปลี่ยนแปลงจาก 2K , 4K , 8K และ 16Kbytes



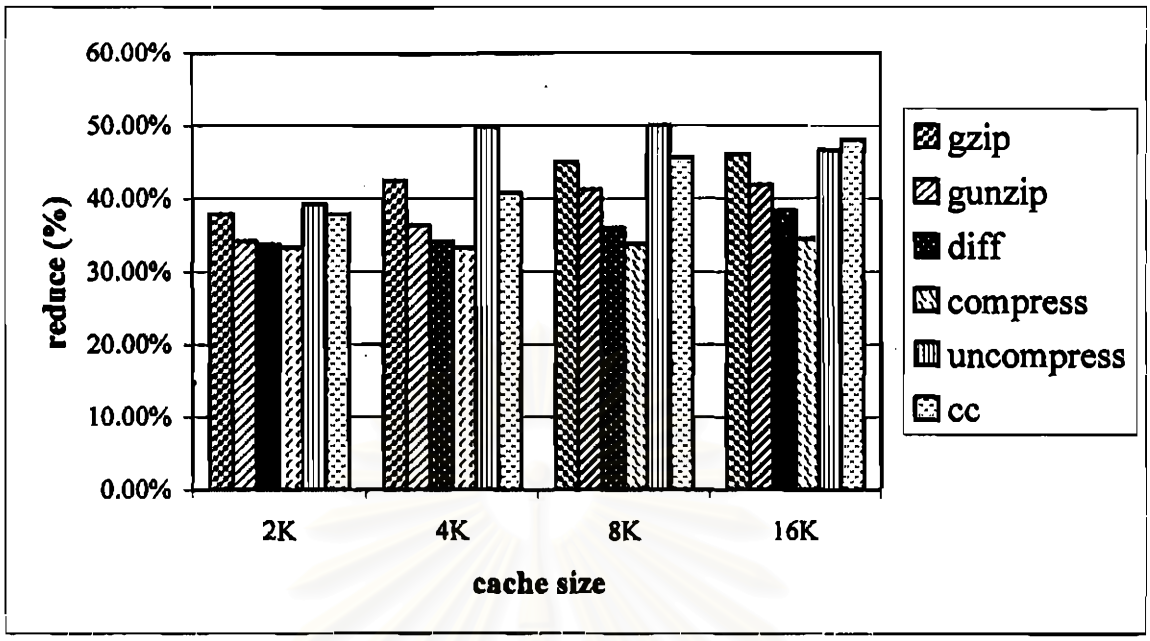
รูปที่ 6.3 ผลการวัดค่า load miss ratio เมื่อใช้โหมดการเขียนเป็น Write Through และมีการจัดแคชเป็นแบบ 2-way set associative โดยขนาดของแคชเปลี่ยนแปลงจาก 2K , 4K , 8K และ 16Kbytes



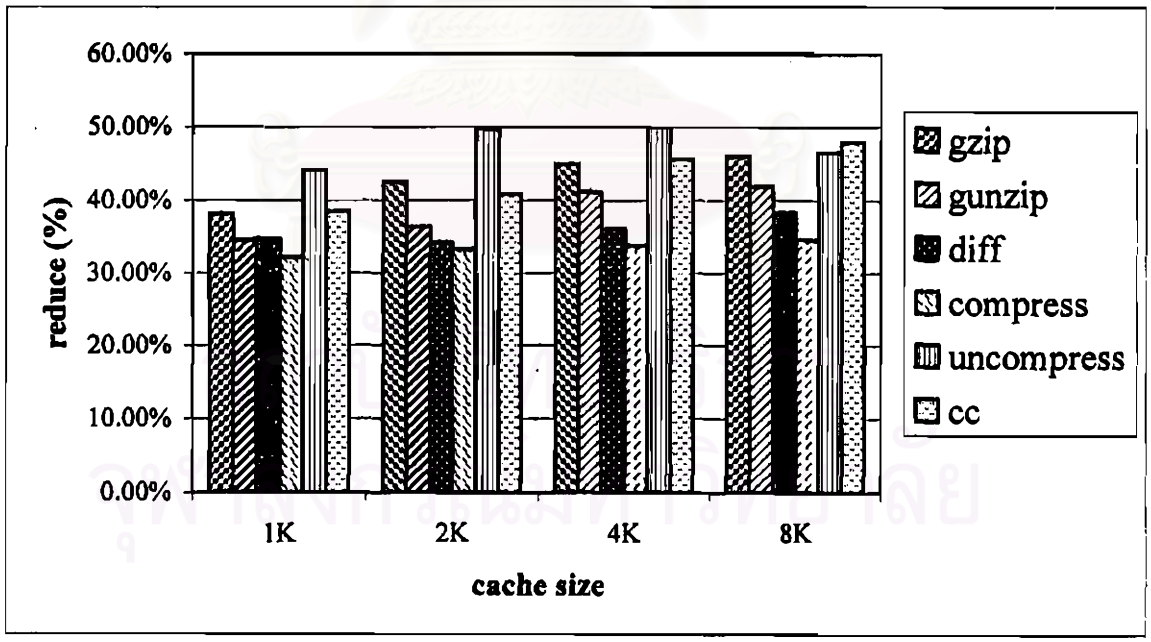
รูปที่ 6.4 ผลของการลดการเกิด "STALL" pipeline เมื่อใช้ non-blocking load (ในส่วน LMC) ที่โหมดการเขียนเป็นแบบ Write Back และการจัดแคชเป็นแบบ Direct-map ตามขนาดของแคชที่เปลี่ยนไป



รูปที่ 6.5 ผลของการลดการเกิด "STALL" pipeline เมื่อใช้ non-blocking load (ในส่วน LMC) ที่โหมดการเขียนเป็นแบบ WriteThrough และการจัดแคชเป็นแบบ Direct-map ตามขนาดของแคชที่เปลี่ยนไป



รูปที่ 6.6 ผลของการลดการเกิด "STALL" pipeline เมื่อใช้ non-blocking load (ในส่วน LMC) ที่โหมดการเขียนเป็นแบบ Write Back และการจัดแคชเป็นแบบ 2-way set associative



รูปที่ 6.7 ผลของการลดการเกิด "STALL" pipeline เมื่อใช้ non-blocking load (ในส่วน LMC) ที่โหมดการเขียนเป็นแบบ Write Through และการจัดแคชเป็นแบบ 2-way set associative

6.4 ผลการวัดสมรรถนะของหน่วย SHC หลังการเพิ่มบัฟเฟอร์

หน่วย SHC (Store Hit Control) ช่วยเพิ่มสมรรถนะการทำงานในส่วนของการประมวลผลคำสั่ง Store ที่เกิด Hit โดยภายในหน่วยนี้มีบัฟเฟอร์จำนวนหนึ่งไว้สำหรับเก็บข้อมูลที่ต้องการเขียนลงแคช ข้อมูลในบัฟเฟอร์นั้นจะถูกเขียนลงแคชเมื่อไม่มีหน่วยประมวลผลอื่นต้องการติดต่อกับแคช การทำงานในหน่วย SHC นี้ช่วยลดการเกิด "STALL" Pipeline อันเนื่องมาจากการขอเขียนข้อมูลลงแคชในขณะเดียวกับที่หน่วยอื่นมีการขอติดต่อกับแคช

ผู้วิจัยได้ทำการวัดสมรรถนะของหน่วย SHC หลังการเพิ่มบัฟเฟอร์ ด้วยค่า stallSHC ซึ่งคำนวณได้จากสมการที่ 6.3 เพื่อใช้เป็นตัวกลางในการเปรียบเทียบสมรรถนะของหน่วย SHC ที่ Benchmark ที่ต่างกัน

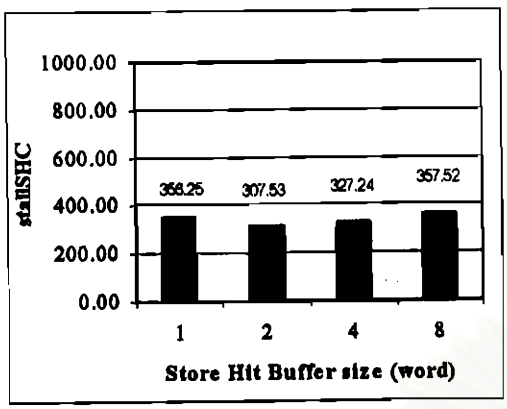
$$stallSHC = \frac{stall7p * 1000}{store\ hit} \quad \text{.....(6.3)}$$

โดยค่า stall7p เกิดขึ้นเนื่องจากการ "STALL" Pipeline อันเนื่องมาจากความต้องการเขียนข้อมูลลง SHB (Store Hit Buffer) แต่บัฟเฟอร์เต็ม หรือในกรณีที่ Pipeline X Stage ต้องการติดต่อข้อมูลในแคชตำแหน่งเดียวกับค่าตำแหน่งของแคชที่ SHC ต้องการติดต่อ

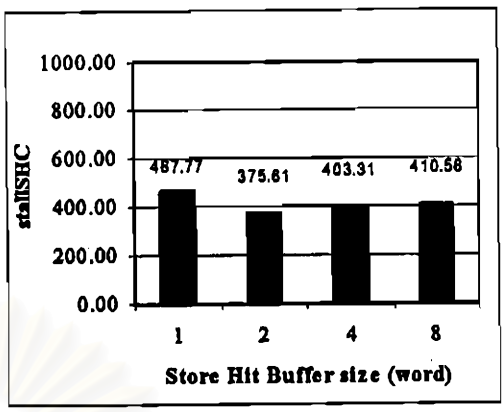
store hit คือจำนวนครั้งที่ประมวลผลคำสั่ง store แล้วเกิดผลเป็น hit

ผลจากการวัดค่า stallSHC ที่ทุกๆ Benchmark ที่ใช้เป็นดังรูปที่ 6.8 โดยเป็นแคชแบบ Direct-Map ขนาด 1Kbyte แกน Y แสดงค่า stallSHC และแกน X แสดงจำนวนบัฟเฟอร์ที่ใช้ในหน่วย SHC ซึ่งขนาดของบัฟเฟอร์ที่กำหนดใช้ในงานวิจัยนี้มี 1, 2, 4 และ 8 บัฟเฟอร์ จะเห็นว่าที่ขนาดบัฟเฟอร์เป็น 2 ต่ำที่สุดที่ทุกๆ Benchmark ที่ใช้ และค่า stallSHC จะมากขึ้นเมื่อเพิ่มขนาดบัฟเฟอร์ขึ้นไปอีก ทั้งนี้เนื่องจาก การที่ SHC มีจำนวนบัฟเฟอร์น้อยเกินไปจะทำให้เกิดเหตุการณ์ที่ต้องการเขียนข้อมูลลงบัฟเฟอร์แต่บัฟเฟอร์เต็ม แต่การที่มีจำนวนบัฟเฟอร์มากจนเกินไปก็จะทำให้เกิดโอกาสที่ Pipeline X Stage ต้องการติดต่อกับแคชในตำแหน่งเดียวกับที่ SHB เก็บไว้สูง

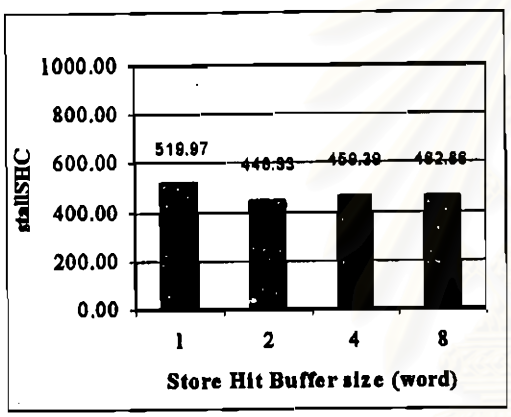
<u>หมายเหตุ</u> ผู้วิจัยไม่สามารถบอกได้ว่าถ้าหากหน่วย SHC ไม่มีบัฟเฟอร์ จะเกิดการ "STALL" pipeline อันเนื่องมาจากการเกิด Store Hit มีค่าเป็นเท่าไร เพราะการเกิด store hit ขึ้นในแต่ละครั้งนั้น Pipeline อาจจะ "STALL" หรือไม่ก็ได้ ขึ้นอยู่กับว่าขณะนั้นแคชมีหน่วยประมวลผลอื่นติดต่ออยู่หรือไม่ กล่าวคือถ้าหากแคชว่าง หน่วย SHC ก็สามารถเขียนข้อมูลลงในแคชได้ทันที แต่ถ้าหากแคชไม่ว่างก็จำเป็นต้องเกิดการ "STALL" pipeline เพื่อขอเขียนข้อมูลลงในแคช แต่ถ้าหากขณะนั้นหน่วย LMC หรือ WBB กำลังติดต่อกับแคชอยู่ เราก็ไม่สามารถตอบได้ว่านานเท่าไรที่หน่วย SHC จะสามารถเขียนข้อมูลลงแคชได้



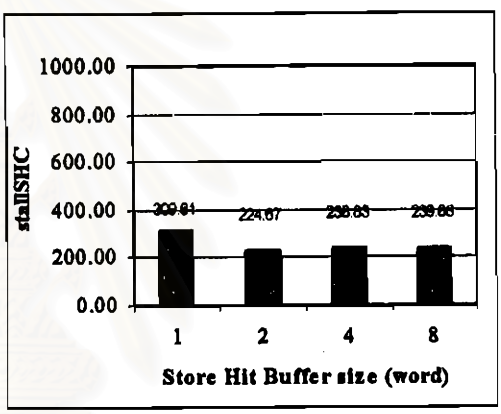
(a) gzip



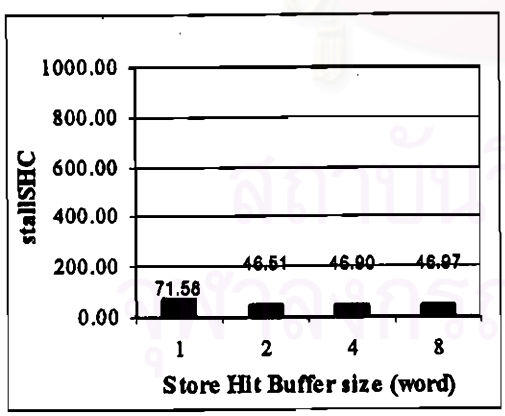
(b) gunzip



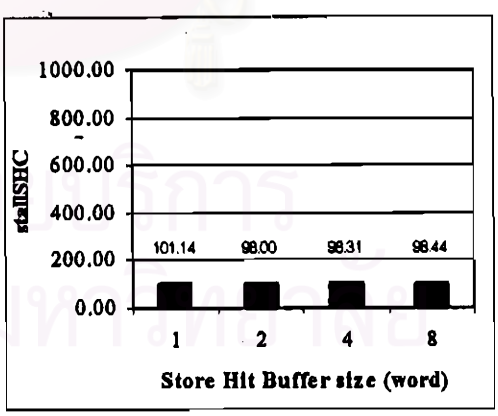
(c) diff



(d) cc



(e) compress



(f) uncompress

รูปที่ 6.8 ผลการวัดสมรรถนะของการเพิ่มบัฟเฟอร์ในหน่วย SHC (Store Hit Control) โดยแกน Y แสดงค่า stallSHC และแกน X เป็นขนาดบัฟเฟอร์ที่ใช้ในหน่วย SHC

(a) gzip , (b) gunzip , (c) diff , (d) cc , (e) compress , (f) uncompress

6.5 ผลการวัดสมรรถนะของหน่วย SMC หลังการเพิ่มขนาด FIFO

หน่วย SMC (Store Miss Control) ช่วยเพิ่มสมรรถนะการทำงานในส่วนของการประมวลผลคำสั่ง Store ที่เกิด Miss โดยภายในหน่วย SMC มี FIFO จำนวนหนึ่งไว้สำหรับเก็บข้อมูลที่ต้องการเขียนลงหน่วยความจำ ในขณะที่ยังมีหน่วยอื่นติดต่อกับหน่วยความจำอยู่

ผู้วิจัยได้ทำการวัดสมรรถนะของหน่วย SMC หลังการเพิ่มขนาด FIFO ด้วยค่า stallSMC ซึ่งคำนวณได้จากสมการที่ 6.4 เพื่อใช้เป็นตัวกลางในการเปรียบเทียบจำนวนการเกิด "STALL" Pipeline อันเป็นผลมาจากการทำงานของหน่วย SMC ที่ Benchmark ที่ต่างกัน

$$\text{StallSMC} = \frac{\text{stall8p} * 1000}{\text{store miss}} \quad \text{.....(6.4)}$$

โดยค่า stall8p เกิดขึ้นจาก "STALL" pipeline ในกรณีที่ต้องการเขียนข้อมูลลง FIFO อันเนื่องมากจากการทำงานในคำสั่ง store แล้วเกิดการ miss (ถ้าเป็นโหมดการเขียนแบบ write through จะเพิ่มกรณีที่เป็น hit ด้วย) แต่ FIFO เต็ม

store miss คือจำนวนครั้งที่ประมวลผลคำสั่ง store แล้วเกิดผลเป็น miss

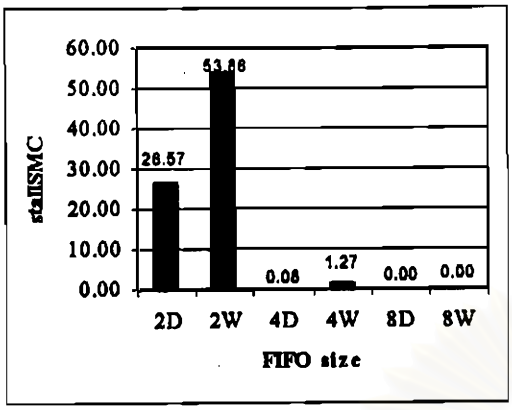
ผลจากการวัดค่า stallSMC ที่ทุกๆ benchmark ที่ใช้เป็นดังรูปที่ 6.9 โดยเป็นแคชแบบ direct-map ขนาด 1Kbyte แกน Y แสดงค่า stallSMC และแกน X แสดงขนาดของ FIFO ที่ใช้ในหน่วย SMC ซึ่งขนาดของ FIFO ที่กำหนดใช้ในงานวิจัยนี้มี 2 doublewords (2D), 2 words (2W), 4 doublewords (4D), 4 words (4W), 8 doublewords(8D) และ 8 words (8W)

จะเห็นว่าค่า stallSMC ต่ำมากแม้ว่าจะมีบัฟเฟอร์แค่เพียง 2 doublewords ก็ตาม เช่นข้อมูลจาก benchmark gzip ได้ค่า stallSMC ที่ 2D เป็น 26.57 หมายถึงเกิดการ "STALL" pipeline อันเนื่องมาจากหน่วย SMC ประมาณ 27 ครั้งต่อการเกิด store miss 1000 ครั้ง และขนาดของ FIFO ที่เหมาะสมกับการใช้งานมากที่สุดคือ 4 doublewords เพราะจะได้ค่า stallSMC เป็นศูนย์หรือใกล้เคียง หรือกล่าวได้อีกแบบก็คือ การเกิด store miss จะไม่ทำให้การประมวลผลต้อง "STALL" pipeline

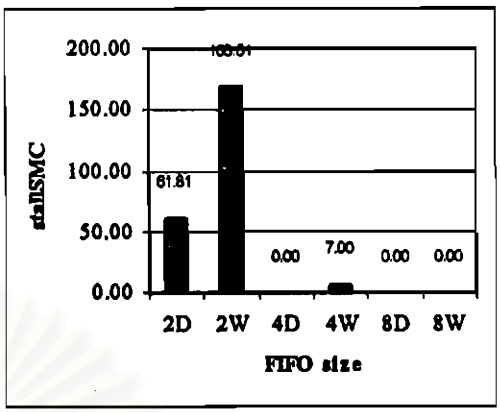
ในรูปที่ 6.10 และ 6.11 แสดงผลของ store miss ratio โดยมีโหมดการเขียนเป็นแบบ write back และการจัดแคชเป็นแบบ direct-map และ 2-way set associative ตามลำดับ ซึ่งเปอร์เซนต์การเกิด store miss ดังที่เห็นในกราฟ เคยเป็นปัญหาเรื่องการ "STALL" pipeline เพื่อขอเขียนข้อมูลลงหน่วยความจำ ก็สามารถแก้ได้ด้วยการเพิ่ม FIFO ลงในหน่วย SMC ปัญหาดังกล่าวก็จะหมดไป

6.6 ขนาดของวงจร

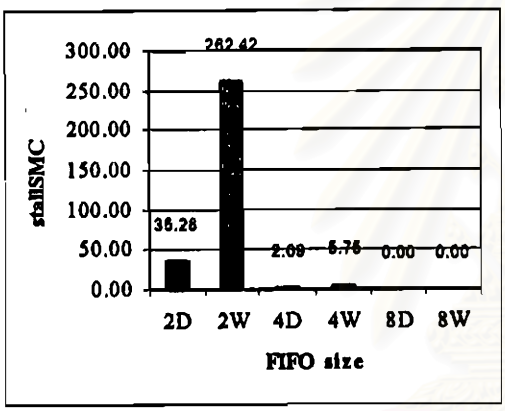
ผู้วิจัยได้สังเคราะห์วงจร ด้วย software "Synergy HDL Synthesizer and Optimizer" ของบริษัท CADENCE โดยทำการสังเคราะห์วงจรเฉพาะหน่วยประมวลผลภายใน LSU เท่านั้น ได้ผลออกมาดังตารางที่ 6.2 ,ตารางที่ 6.3 แสดงขนาดของวงจรหน่วย SHC ที่ขนาดบัฟเฟอร์เป็น 1, 2, 4 และ 8 บัฟเฟอร์ และตารางที่ 6.4 แสดงขนาดของวงจรหน่วย SMC ที่ขนาดของ FIFO เป็น 2, 4 และ 8 words / doublewords



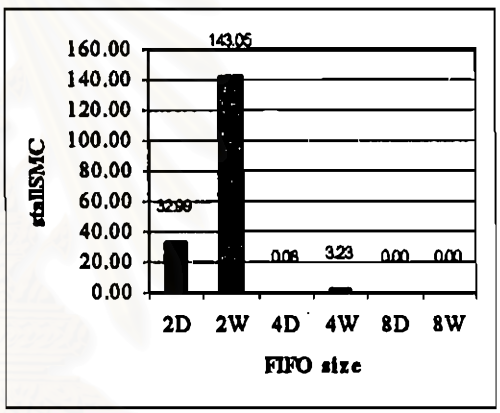
(a) gzip



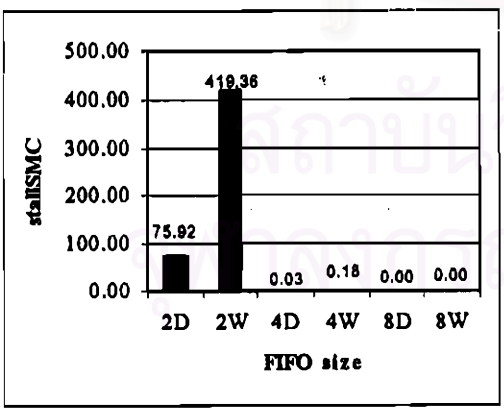
(b) gunzip



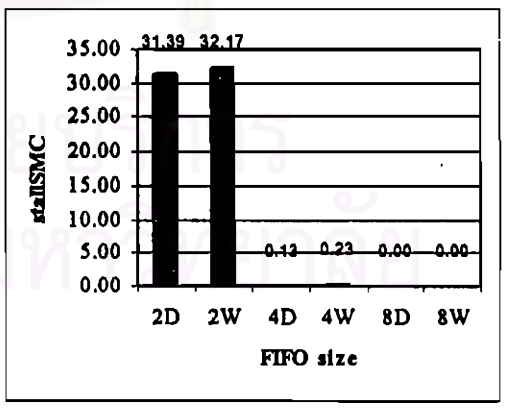
(c) diff



(d) cc



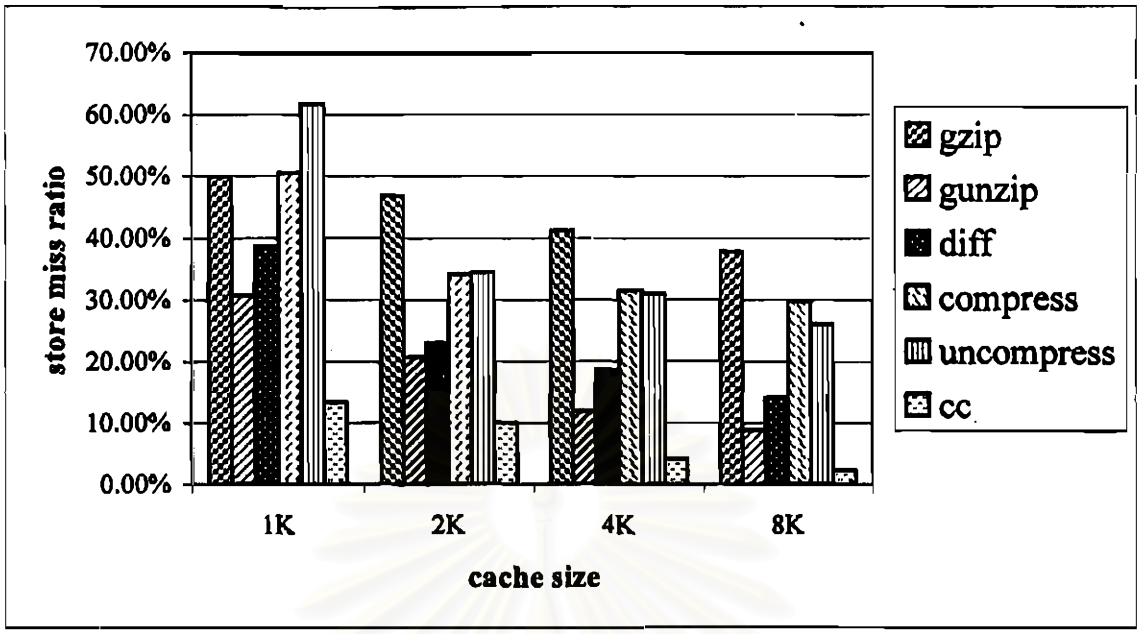
(e) compress



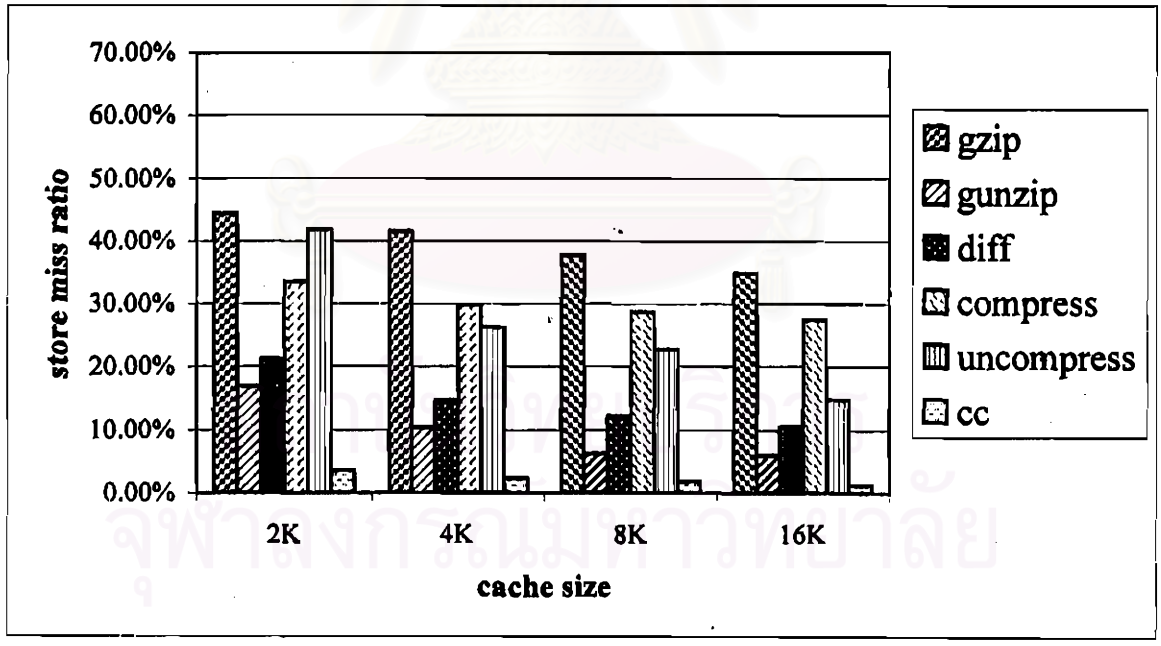
(f) uncomrpess

รูปที่ 6.9 ผลการวัดสมรถนะของการเพิ่มขนาด FIFO ในหน่วย SMC (Store Miss Control) โดยแกน Y แสดงค่า stallSMC และแกน X เป็นขนาด FIFO ที่ใช้ในหน่วย SMC .

(a) gzip , (b) gunzip , (c) diff , (d) cc , (e) compress , (f) uncomrpess



รูปที่ 6.10 ผลการวัดค่า store miss ratio เมื่อใช้โหมดการเขียนเป็น Write Back และมีการจัดแคชเป็นแบบ direct-map โดยขนาดของแคชเปลี่ยนแปลงจาก 1K , 2K , 4K และ 8Kbytes



รูปที่ 6.11 ผลการวัดค่า store miss ratio เมื่อใช้โหมดการเขียนเป็น Write Back และมีการจัดแคชเป็นแบบ 2-way set associative โดยขนาดของแคชเปลี่ยนแปลงจาก 2K , 4K , 8K และ 16Kbytes

Unit	PC	LMC	DCCtrl	AccessBIU	Delay	Stall
Number of ports	736	83	313	343	58	58
Number of nets	2015	156	525	523	114	94
Number of cells	1445	114	352	288	83	54
Number of reference	32	18	17	16	4	16
Combination area	2607	170	415	481	84	104
Noncombination area	1128	21	410	33	189	7
Net Interconnection area	0	0	0	0	0	0
Total area	3735	191	825	514	273	111

ตารางที่ 6.2 ขนาดวงจรของหน่วยประมวลผลใน LSU

Unit	SHC 1 buffer	SHC 2 buffers	SHC 4 buffers	SHC 8 buffers
Number of ports	125	125	125	125
Number of nets	216	409	777	1437
Number of cells	95	244	515	976
Number of reference	18	24	28	30
Combination area	78	298	561	1145
Noncombination area	522	1019	2016	3986
Net Interconnection area	0	0	0	0
Total area	600	1317	2577	5131

ตารางที่ 6.3 ขนาดของวงจรหน่วย SHC ที่ขนาดบัฟเฟอร์เป็น 1, 2, 4 และ 8

Unit	SMC 2D	SMC 2W	SMC 4D	SMC 4W	SMC 8D	SMC 8W
Number of ports	207	207	207	207	207	207
Number of nets	1720	1136	3268	2182	6135	4094
Number of cells	957	636	1910	1285	3588	2404
Number of reference	20	18	26	20	28	28
Combination area	921	590	1756	1217	3420	2539
Noncombination area	4634	2665	8604	5315	16534	10605
Net Interconnection area	0	0	0	0	0	0
Total area	5555	3255	10360	6532	19954	13144

ตารางที่ 6.4 ขนาดของวงจรหน่วย SMC ที่ขนาด FIFO เป็น 2, 4 และ 8 word และ doubleword

จากตารางที่ 6.3 จะเห็นได้ว่าจำนวนเซลล์ (Number of cells) และพื้นที่รวม (Total area) ของวงจรสำหรับหน่วย SHC เพิ่มขึ้นมากกว่า 1 เท่าตัวเมื่อเพิ่มจำนวนบัฟเฟอร์ ดังนั้นในการนำไปใช้งานจริง จำเป็นต้องเปรียบเทียบระหว่างสมรรถนะที่ได้กับจำนวนเซลล์และพื้นที่รวมที่เสียไป โดยที่ SHB เท่ากับ 1 บัฟเฟอร์มีสมรรถนะในการลดจำนวนค่า "STALL" Pipeline ของหน่วย SHC ด้อยกว่าที่ SHB เท่ากับ 2 บัฟเฟอร์เพียงเล็กน้อย แต่ใช้จำนวนเซลล์และพื้นที่รวมของวงจรน้อยกว่าเกินครึ่ง

จากตารางที่ 6.4 จะเห็นได้ว่าจำนวนเซลล์และพื้นที่รวมของวงจรสำหรับหน่วย SMC เพิ่มขึ้นตามขนาด FIFO ที่เปลี่ยนแปลงไป ดังนั้นจึงจำเป็นอย่างยิ่งที่ต้องหาขนาดของ FIFO ที่ให้สมรรถนะในการลดจำนวน "STALL" Pipeline ของหน่วย SMC ให้มากที่สุด

6.7 สรุป

ในบทนี้ได้กล่าวถึงจำนวนคำสั่งของ Benchmark ที่ใช้ในการทดสอบสมรรถนะการทำงานของ LSU ที่ถูกออกแบบ และได้นำ Benchmark ดังกล่าวไปทดสอบ ผลที่ได้คือ ขนาดแคชที่ใหญ่ขึ้นจะทำให้ Load Miss Ratio ลดลง และที่โหมดการเขียนแบบ Write Back และ Write Through ให้ผลของ Load Miss Ratio ใกล้เคียงกันและผลการวัดสมรรถนะของการทำงานเป็น Non-Blocking Load ในหน่วย LMC สามารถลดจำนวนการเกิด "STALL" Pipeline อันเนื่องมาจากการรอผลการทำงานของคำสั่ง Load ที่เกิดการ Miss ได้ประมาณ 30% และผลการวัดสมรรถนะของหน่วย SHC หลังจากที่มีการเพิ่มขนาดบัฟเฟอร์ ซึ่งที่ 2 SHB จะทำให้เกิดการ "STALL" Pipeline น้อยที่สุด ส่วนผลการวัดสมรรถนะของหน่วย SMC หลังการเพิ่มขนาด FIFC เราได้ FIFO ที่ขนาด 4 doubleword ให้ผลที่น่าพึงพอใจมากที่สุด และในตอนท้ายได้กล่าวถึงขนาดของวงจรของหน่วยประมวลผลภายใน LSU ที่ออกแบบ โดยที่ SHB เท่ากับ 1 บัฟเฟอร์ใช้จำนวนเซลล์และพื้นที่รวมของวงจรน้อยกว่าเกินครึ่งหนึ่งของที่ SHB เท่ากับ 2 บัฟเฟอร์ใช้ ทั้งที่สมรรถนะในการลดจำนวนการเกิด "STALL" Pipeline สูงกว่าเพียงเล็กน้อย และจำนวนเซลล์และพื้นที่รวมของวงจรในหน่วย SMC เพิ่มขึ้นสูงมากตามขนาดของ FIFO ที่ใช้

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย