บทที่ 7
สรุปและข้อเสนอแนะ

งานวิจัยนี้มีวัตถุประสงค์เพื่อออกแบบ Load / Store Unit (LSU) ซึ่งเป็นหน่วยประมวลผลในส่วนของคำสั่ง Load และคำสั่ง Store โดยมี Data Cache เป็นหน่วยความจำลำดับแรกที่ LSU ร้องขอการอ่านและเขียนข้อมูล ผู้วิจัยได้ทำการพัฒนา LSU ในส่วนของการทำงานเป็น Non-Blocking Load เมื่อเกิดเหตุการณ์ Load Miss ขึ้น และสำหรับคำสั่ง Store ที่ต้องการเขียนข้อมูลลงหน่วยความจำนั้น ผู้วิจัยได้ออกแบบบัฟเฟอร์และ FIFO สำหรับพักข้อมูลก่อนการเขียนลงใน Data Cache และหน่วยความจำหลัก โดยวิธีการต่างๆ ที่นำเสนอนี้จุดประสงค์เพื่อลดการเกิด "STALL" Pipeline อันเนื่องมาจากในขณะที่ประมวลผลนั้น มีมากกว่า 1 งานที่ต้องการติดต่อกับ Data Cache หรือหน่วยความจำ

7.1 สรุปผลการวิจัย

วงจร LSU ที่ออกแบบทำงานเป็น 4-Stage Pipeline สามารถประมวลผลชุดคำสั่ง load, ชุดคำสั่ง store และคำสั่ง ADD ได้ตามที่แสดงไว้ในภาคผนวก ก. โดยผู้ใช้สามารถเลือกขนาดของบัฟเฟอร์ในหน่วย SHC ได้ว่าเป็น 1, 2, 4 หรือ 8 บัฟเฟอร์ และเลือกขนาดของ FIFO ในหน่วย SMC ได้ว่าเป็น 2, 4 หรือ 8 word / doubleword นอกจากนี้ยังสามารถกำหนดขนาดของ Data Cache ได้ว่าเป็น 1K, 2K, 4K หรือ 8Kbytes ต่อเซต และสามารถเลือกการจัดแคชเป็นแบบ Direct-Map หรือ 2-Way Set Associative และโหมดการเขียนเป็นแบบ Write Back หรือ Write Through

ความถูกต้องในการทำงานของวงจรยังไม่สมบูรณ์ 100% โดยทดสอบกับ Trace ขนาด 6 ล้านคำสั่ง (รวมทุก Trace) ผลปรากฏว่าการประมวลผลคำสั่ง Load สูญหายไป 0.01% แต่การทำงานในคำสั่ง Store ครบถ้วนสมบูรณ์

ผลการทดลองวัดค่า Load Miss Ratio เมื่อทำการเปลี่ยนแปลงขนาดแคชและการการจัดแคช ผลปรากฏว่า แคชยิ่งขนาดใหญ่ขึ้น จะทำให้ค่า Miss Ratio ลดต่ำลง และการจัดแคชเป็นแบบ 2-Way Set Associative ให้ผลดีกว่าการจัดแคชแบบ direct-map ที่ขนาดแคชเท่ากัน

ผลการวัดจำนวนที่ลดลงของการเกิด "STALL" Pipeline ของการทำงานที่เป็น Non-Blocking Load เมื่อเทียบกับการทำงานที่เป็น Blocking Load สำหรับผลการประมวลผลคำสั่ง Load ที่ให้ผลเป็น Miss ผลปรากฏว่า จำนวน "STALL" Pipeline ที่ลดลงของแต่ละ Benchmark ไม่เท่ากัน โดยมีค่าเฉลี่ยอยู่ที่ 46% (ค่าลดลงต่ำสุดที่ได้ประมาณ 30% และค่าสูงสุดที่ได้ประมาณ 55%)

ผลจากการวัดจำนวนการเกิด "STALL" Pipeline อันเนื่องมาจากคำสั่ง Store ที่เกิดผลเป็น Hit เมื่อทำการเปลี่ยนแปลงจำนวนบัฟเฟอร์ในหน่วย SHC (Store Hit Control) (1, 2, 4 และ 8 SHB) และกำหนดขนาดของแคชที่ใช้ในการทดสอบเป็น 1K และมีการจัดแคชเป็นแบบ Direct-Map ผลปรากฏว่า จำนวนบัฟเฟอร์เป็น 2 จะให้ผลการเกิด "STALL" Pipeline น้อยที่สุด เนื่องจาก การที่ SHC มีจำนวนบัฟเฟอร์น้อยเกินไปจะทำให้เกิดเหตุการณ์ที่ต้องการเขียนข้อมูลลงบัฟเฟอร์แต่บัฟเฟอร์เต็ม แต่การที่มีจำนวนบัฟเฟอร์มากจนเกินไปก็จะทำให้เกิดโอกาสที่ Pipeline X Stage ต้องการติดต่อกับแคชในตำแหน่งเดียวกับที่ SHB เก็บไว้สูง แต่วงจรของ SHC ที่ 1

บัฟเฟอร์มีขนาดเล็กกว่าครึ่งหนึ่งของวงจร SHC ที่ 2 บัฟเฟอร์ โดยที่จำนวน "STALL" Pipeline มากกว่าเพียงเล็กน้อย

ผลการวัดจำนวนการเกิด "STALL" Pipeline อันเนื่องมาจากคำสั่ง Store ที่เกิดผลเป็น Miss เมื่อทำการเปลี่ยนแปลงขนาดของ FIFO (2, 4 และ 8) และรูปแบบการเก็บข้อมูลของ FIFO (word และ doubleword) และใช้แคช Direct-Map ขนาด 1K (เกิด Store Miss ประมาณ 20%) ผลปรากฏว่า FIFO ที่ขนาด 4 doublewords ให้จำนวนการเกิด "STALL" Pipeline เกือบเท่ากับศูนย์

7.2 จุดบกพร่อง

ข้อมูลจากงานวิจัยงานวิจัยของ P. Milligan, K. Kuchcinski, W. Grunewald และ T. Ungerer ได้ระบุไว้ว่าความเร็วในการติดต่อกับหน่วยความจำเป็นดังตารางที่ 7.1 โดยค่าที่ระบุเป็นจำนวนคาบสัญญาณนาฬิกาที่วัดเทียบกับความถี่ของไมโครโพรเซสเซอร์ (ความถี่ 300 MHz)

	access time	cycle time
on-chip cache	2	1
off-chip cache	9	9
DRAM memory	27	14-27

ตารางที่ 7.1 Access and cycle times

โดยค่า access time คือ จำนวนเวลาที่ใช้ในติดต่อกับหน่วยความจำจนกระทั่งเสร็จสมบรณ์
cycle time คือ ช่วงเวลาที่น้อยที่สุดระหว่างการติดต่อกับหน่วยความจำสองครั้ง

นอกจากนี้แล้ว เรายังสามารถติดต่อขออ่านข้อมูลเป็นชุดอย่างเร็ว (burst read) กับ DRAM 60 ns ได้ที่ความเร็ว 8-3-3-3 โดยความเร็วของบัสที่ใช้เป็น 100 MHz ซึ่งหมายถึง ข้อมูลชุดแรกที่ต้องการอ่านใช้เวลา 8 ช่วงสัญญาณนาฬิกาของความเร็วบัส และสามารถอ่านข้อมูลในชุดต่อไปที่แอดเดรสที่ติดกันได้ โดยใช้เวลาเพียง 3 ช่วงสัญญาณนาฬิกา

แต่ในงานวิจัยนี้ กำหนดเวลาที่ใช้ในการติดต่อกับ on-chip cache ไว้ว่าค่า access time เท่ากับ 1 และค่า cycle time เท่ากับ 0 อีกทั้งเวลาที่ใช้ในการติดต่อกับ DRAM memory ก็มีค่า access time เท่ากับ 1 และค่า cycle time เท่ากับ 0 อีกทั้งเวลาที่ใช้ในการติดต่อขออ่านข้อมูลเป็นชุดอย่างเร็ว เป็น 1-1-1-1 ซึ่งคาบสัญญาณนาฬิกาที่ใช้ดังกล่าวทั้งหมดนี้เป็นความเร็วของสัญญาณนาฬิกาที่ใช้กับไมโครโพรเซสเซอร์

เนื่องจากผู้วิจัยไม่มีเวลาเพียงพอในการแก้ไขปัญหาในกรณีดังกล่าว แต่ได้พยายามแก้ไขด้วยวิธีการเพิ่มเวลาที่ใช้ในการติดต่อหน่วยความจำโดยการเพิ่มจำนวน cycle time และทดลองกับ Trace gunzip3.asm เพราะเป็น Trace ตัวที่เล็กที่สุดที่ใช้ วิธีการแก้ไขเฉพาะช่วง cycle time ดังกล่าวไม่ตรงกับความเป็นจริงมากนัก แต่ถ้าหากผู้วิจัยทำการแก้ไขค่าเวลาที่ใช้ในการติดต่อขออ่านข้อมูลเป็นชุดอย่างเร็ว จะส่งผลกระทบต่อการทำงานของวงจรเป็นอย่างมาก อีกทั้งต้องตรวจสอบความถูกต้องของการทำงานของวงจรใหม่ทั้งหมด ซึ่งขั้นตอนในการ

ตรวจสอบนี้ผู้วิจัยใช้เวลาไม่ต่ำกว่าครึ่งปี อีกทั้งการเพิ่มเวลาที่ใช้ในการติดต่อกับหน่วยความจำต้องใช้เวลาในการวัดสมรรถนะนานขึ้นกว่าเดิมอีก 1 เท่าตัว ซึ่งแต่เดิมผู้วิจัยใช้เวลาประมาณ 2 เดือนในการทดสอบ

ดังนั้นผู้วิจัยจึงทำการเพิ่มค่า cycle time ขึ้นเพื่อให้ผลใกล้เคียงกับความเป็นจริงมากที่สุด แต่การเพิ่มเพียง cycle time ทำให้การทำงานของการใช้วิธี Non-Blocking Load แย่ลง เพราะข้อมูลใน Doubleword แรกซึ่งเป็นข้อมูลที่ต้องการโหลดนั้นได้ช้าลง ทำให้จำนวนครั้งในที่เกิดการ "STALL" Pipeline มีมากขึ้น

เมื่อเพิ่มค่า cycle time มีผลต่อการวัดสมรรถนะดังนี้

1. ค่า Load Miss Ratio และค่า Store Miss Ratio

   ค่า Miss Ratio ที่วัดได้ยังคงเป็นค่าที่ถูกต้อง เนื่องจากความเร็วที่ใช้ในการติดต่อกับหน่วยความจำไม่มีผลต่อการตรวจสอบว่า Cache ว่าให้ผลเป็น Hit หรือ Miss

   <u>สรุป</u> ค่า Load Miss Ratio ที่วัดได้มีค่าเท่าเดิม

2. ค่า Reduce ที่ได้จากการเปรียบเทียบจำนวนครั้งที่เกิดการ "STALL" Plipeline อันเนื่องมาจากการทำงานแบบ Non-Blocking Load และ Blocking Load เมื่อการประมวลผลคำสั่ง Load เกิด Miss ขึ้น

   จำนวนครั้งที่เกิด "STALL" Pipeline อันเนื่องมาจากการทำงานแบบ Non-Blocking Load และ Blocking Load เมื่อเกิดเหตุการณ์ Load Miss เพิ่มขึ้นเมื่อกำหนดเวลาที่ใช้ในการติดต่อกับหน่วยความจำมากขึ้น แต่เปอร์เซนต์ของค่า Reduce ที่ได้ใกล้เคียงค่าเดิม คือที่ cycle time เท่ากับ 0 ค่าเปอร์เซนต์ Reduce เท่ากับ 29.02% และที่ cycle time เท่ากับ 17 เป็น 29.33%[*]

   <u>สรุป</u> ได้ว่าค่า Reduce มีค่าใกล้เคียงกับค่าเดิม

3. จำนวนครั้งที่เกิดการ "STALL" Pipeline อันเนื่องมากจากเกิดเหตุการณ์ Store Hit ขึ้น

   จำนวน "STALL" Pipeline ลดลงเล็กน้อยเมื่อเพิ่ม cycle time ให้มากขึ้น เนื่องมาจากว่า ขณะที่เกิดการ "STALL" Pipeline อันเนื่องมาจากสาเหตุอื่น ซึ่งได้แก่ WBB หรือ SMC ขอเขียนข้อมูลลงในหน่วยความจำ การทำงานของ SHC สามารถนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์ไปเขียนลงในแคชได้ ทำให้โอกาสที่บัฟเฟอร์ใน SHC เต็มมีน้อยลง และโอกาสที่ Pipeline X Stage จะอ้างอิงข้อมูลตำแหน่งเดียวกับ SHC ก็มีน้อยลงด้วยเช่นกัน

   <u>สรุป</u> ได้ว่า SHB เท่ากับ 2 บัฟเฟอร์ยังคงเป็นขนาดบัฟเฟอร์ที่ทำให้จำนวน "STALL" Pipeline อันเนื่องมาจากหน่วย SHC ต่ำที่สุด

4. จำนวนครั้งที่เกิดการ "STALL" Pipeline อันเนื่องมากจากเกิดเหตุการณ์ Store Miss ขึ้น

   จำนวน "STALL" Pipeline เพิ่มขึ้นอย่างมาก เนื่องจากเวลาที่ FIFO นำข้อมูลที่เก็บไว้เขียนลงหน่วยความจำนานขึ้น ซึ่งเวลาที่หน่วยประมวลผลอื่นใช้ในการติดต่อกับหน่วยความจำก็นานขึ้นด้วยเช่นกันส่งผลทำให้โอกาสที่ FIFO ในหน่วย SMC เต็มมีสูงมาก

   <u>สรุป</u> ได้ว่า ขนาดของ FIFO ที่ดีที่สุดเปลี่ยนจาก 4 doublewords ไปเป็น 8 words

---

[*] ค่าที่ได้เป็นผลจากการทดสอบบน Trace gunzip3.asm เพียงตัวเดียว จึงไม่สามารถยืนยันได้

7.3 ข้อเสนอแนะ

1. การทำงานเป็น Non-Blocking Load ในหน่วย LMC สามารถพัฒนาต่อไปได้อีกโดยใช้วิธีที่ K. I. Farkas, N. P. Jouppi, P. Chow (1995) ได้เสนอ ไว้ในบทความ "How Useful Are Non-blocking Loads, Stream Buffers and Speculative Execution in Multiple Issue Processor?" ได้เสนอมีส่วนจัดการ Load Miss สำหรับทุกๆ รีจีสเตอร์ ซึ่งทำให้สามารถประมวลผลคำสั่ง Load ที่เกิดผลเป็น Miss ได้มากกว่า 1 คำสั่ง การทำเช่นนี้ทำให้ลดการเกิด "STALL" Pipeline ลงได้อย่างมาก
2. ในเรื่องความเร็วในการติดต่อกับหน่วยความจำ มีการเปลี่ยนแปลงตามเทคโนโลยีในแต่ละยุคที่เปลี่ยนไป ซึ่งในปัจจุบันนี้ความเร็วในการติดต่อกับหน่วยความจำเป็นไปดังตารางที่ 7.2 ที่สัญญาณนาฬิกาของโพรเซสเซอร์ 66 MHz

Memory Type	Latency (processor cycle per bit delevered)			
	Bit1	Bit 2	Bit 3	Bit 4
L2 (SRAM – Static Random Access Mem.)	2	1	1	1
SDRAM (Synchronous Dynamic Random Access Mem.)	5	1	1	1
BEDO RAM (Burst Extended Data Out DRAM)	5	1	1	1
EDO DRAM (Extended Data Out DRAM)	5	2	2	2
FPM DRAM (Fast Page Mode DRAM)	5	3	3	3

ตารางที่ 7.2 ความเร็วในการติดต่อกับหน่วยความจำ (ข้อมูลจาก WWW ของ IBM Microelectronics)