บทที่ 7
วิเคราะห์และสรุปผลการวิจัย

ในบทนี้จะกล่าวถึงขั้นตอนการวิเคราะห์ประสิทธิภาพของวงจรหน่วยประมวลผล ซึ่งการวัดประสิทธิภาพนี้ได้ทำในหลายๆ ด้าน ได้แก่ ขนาดของโปรแกรม (Code size) ประสิทธิภาพการทำงานโดยวัดจากจำนวนรอบนาฬิกา (Cycle) ที่ใช้ในการทำงานโปรแกรมทดสอบ ความถี่ของสัญญาณนาฬิกาสูงสุดที่ทำงานได้ และทรัพยากรที่ใช้ในการสร้างวงจรหน่วยประมวลผลวัดจากจำนวนเกตสมมูล (Equivalent gate) โดยค่าเหล่านี้ได้จากผลการสังเคราะห์วงจรด้วยโปรแกรม Xilinx WebPack รุ่น 8.1i บนอุปกรณ์เอฟพีจีเอรุ่น Spartan 3 XC3S200

นอกจากนี้ยังทำการวิเคราะห์ประสิทธิภาพการอัดคำสั่งที่ใช้ในงานวิจัยนี้ ว่ามีส่วนช่วยลดขนาดของโปรแกรม และจำนวนรอบนาฬิกาที่ใช้ในการทำงานโปรแกรมทดสอบมากน้อยเพียงใด

ในการวิเคราะห์ประสิทธิภาพนี้ได้มีการนำหน่วยประมวลผลอื่นๆ มาใช้ในการเปรียบเทียบเพื่อให้การวิเคราะห์ประสิทธิภาพนี้ทำได้ง่ายขึ้น โดยหน่วยประมวลผลที่นำมาใช้ในการเปรียบเทียบในงานวิจัยนี้ประกอบด้วย

1. หน่วยประมวลผลแบบแสตก
2. หน่วยประมวลผลไมโครแบลซ (Micro Blaze) [18] ของบริษัท Xilinx

หน่วยประมวลผลแบบแสตกนั้นใช้ในการเปรียบเทียบขนาดของโปรแกรม เนื่องจากหน่วยประมวลผลแบบแสตกนี้ ใช้ชุดคำสั่งแบบแสตกซึ่งเป็นที่ยอมรับกันว่า ชุดคำสั่งแบบแสตกนี้ช่วยทำให้โปรแกรมมีขนาดเล็ก ในด้านประสิทธิภาพการทำงานนั้นจะไม่นำมาเปรียบเทียบ เนื่องจากหน่วยประมวลผลแบบแสตกนี้เป็นหน่วยประมวลผลขนาด 16 บิต

สำหรับหน่วยประมวลผลไมโครแบลซ (Micro Blaze) [18] ของบริษัท Xilinx หน่วยประมวลผลนี้สามารถทำการสังเคราะห์วงจรด้วยโปรแกรม Xilinx WebPack ได้ ทำให้เราสามารถเปรียบเทียบค่าต่างๆ ที่ได้จากผลการสังเคราะห์วงจร เช่น ค่าความถี่นาฬิกาสูงสุด จำนวนเกตสมมูล เป็นต้น นอกจากนี้จากการที่หน่วยประมวลผลนี้สามารถสังเคราะห์วงจรลงบนอุปกรณ์เอฟพีจีเอได้ ซึ่งทำให้สภาพแวดล้อมในการทำงานของหน่วยประมวลผลที่ใช้ในการเปรียบเทียบใกล้เคียงกันมาก

หน่วยประมวลผลไมโครแบลซ ใช้เป็นเกณฑ์ในการวัดประสิทธิภาพ เนื่องจากเป็นหน่วยประมวลผลที่มีขายตามท้องตลาด มีประสิทธิภาพเป็นที่ยอมรับของผู้คน

ในบทนี้จะขอกล่าวถึงโปรแกรมทดสอบที่ใช้เสียก่อน จากนั้นกล่าวถึงการวิเคราะห์ประสิทธิภาพจากการอัดคำสั่ง แล้วจึงเป็นการวิเคราะห์ประสิทธิภาพของหน่วยประมวลผลต่อไป

7.1 โปรแกรมทดสอบที่ใช้ในงานวิจัยนี้

โปรแกรมทดสอบที่ใช้ในงานวิจัยนี้ ประกอบด้วยโปรแกรม 7 โปรแกรมดังแสดงในตารางที่ 7.1

ตารางที่ 7.1 โปรแกรมที่ใช้ในการทดสอบ

ชื่อโปรแกรม	คำอธิบาย
Bubble	Bubble sort ข้อมูลจำนวน 100 ตัว ข้อมูลเริ่มต้นเรียงจากมากไปน้อย
Merge	Merge sort ข้อมูลจำนวน 100 ตัว ข้อมูลเริ่มต้นเรียงจากมากไปน้อย
Quick	Quick sort ข้อมูลจำนวน 20, 60 และ 100 ตัวตามเลขที่ระบุ ข้อมูลเริ่มต้นเรียงจากมากไปน้อย
Fibo	หาค่าของลำดับที่ 10 ของลำดับ Fibonacci
Hanoi	แก้ปัญหาฮานอย 6 จาน (6 disks Hanoi problem)
Sieve	หาจำนวนเฉพาะทุกตัวที่มีค่าน้อยกว่า 100
AES	เข้ารหัสเออีเอส (Advance Encryption Standard : AES) ข้อมูลขนาด 128 บิต ด้วยคีย์ขนาด 128 บิต

7.2 ประสิทธิภาพที่เพิ่มขึ้นจากการอัดคำสั่ง

ประสิทธิภาพที่เพิ่มขึ้นจากการอัดคำสั่งนี้มี 2 ประการคือ ขนาดของโปรแกรมที่ลดลง และจำนวนรอบนาฬิกาใช้ในการทำงานน้อยลง ซึ่งทั้งสองค่านี้มีความสัมพันธ์กัน เนื่องจากขนาดโปรแกรมที่ลดลง ทำให้จำนวนรอบนาฬิกาที่ใช้ในการอ่านคำสั่งลดลง

ขนาดของโปรแกรม (Code size) ที่ลดลงจากการอัดคำสั่งนี้ วัดโดยการเขียนโปรแกรมทดสอบออกมา 2 ชุด ชุดหนึ่งเขียนโดยไม่ทำการอัดคำสั่ง อีกชุดทำการอัดคำสั่งตามที่ได้นำเสนอไปในบทที่ 4 แล้วจึงนำขนาดของโปรแกรมทั้งสองชุดมาเปรียบเทียบกัน ได้ผลดังแสดงในตารางที่ 7.2

จากการเปรียบเทียบในตารางที่ 7.2 พบว่าการอัดคำสั่งที่นำเสนอในบทที่ 4 ช่วยลดขนาดของโปรแกรมได้ โดยค่าเฉลี่ยที่วัดได้จากโปรแกรมทดสอบทั้ง 7 โปรแกรม การอัดคำสั่งนี้สามารถลดขนาดของโปรแกรมได้ร้อยละ 37.9 ของขนาดโปรแกรมปกติ

ในตารางที่ 7.3 เป็นการเปรียบเทียบจำนวนรอบนาฬิกาที่ใช้ในการทำงานโปรแกรมทดสอบทั้ง 2 แบบ พบว่าโปรแกรมที่ได้รับการอัดคำสั่งทำงานเร็วกว่า กล่าวคือ ใช้จำนวนรอบนาฬิกาน้อยกว่า โดยค่าเฉลี่ยที่วัดได้จากโปรแกรมทดสอบทั้ง 7 โปรแกรม โปรแกรมที่ได้รับการอัดคำสั่งทำงานได้เร็วกว่าโปรแกรมธรรมดาประมาณ 1.22 เท่า

ตารางที่ 7.2 การเปรียบเทียบขนาดของโปรแกรมที่ได้รับการอัดคำสั่งกับโปรแกรมปกติ

โปรแกรมทดสอบ	ขนาดของโปรแกรม (ไบต์)		ขนาดที่ลดลง (%)
	ไม่ได้อัดคำสั่ง	อัดคำสั่ง	
Bubble	100	68	32.0
Merge	380	212	44.2
Quick	252	168	33.3
Fibo	56	36	35.7
Hanoi	180	108	40.0
Sieve	152	84	44.7
AES	1,336	860	35.6
		เฉลี่ย	**37.9**

ตารางที่ 7.3 การเปรียบเทียบประสิทธิภาพการทำงานของโปรแกรมที่ได้รับการอัดคำสั่งกับโปรแกรมปกติ

โปรแกรมทดสอบ	จำนวนรอบนาฬิกาที่ใช้ในการทำงาน		เร็วขึ้น
	ไม่ได้รับการอัดคำสั่ง	อัดคำสั่ง	
Bubble	327,704	282,953	1.16
Merge	35,594	28,500	1.25
Quick			**1.17**
20items	5,388	4,575	1.18
60 items	32,948	28,116	1.17
100 items	82,908	70,856	1.17
Fibo	3,359	2,740	1.23
Hanoi	4,398	3,582	1.23
Sieve	2,341	1,854	1.26
AES	59,306	49,098	1.21
		เฉลี่ย	**1.22**

ค่าที่นิยมใช้ในการเปรียบเทียบประสิทธิภาพการทำงานของหน่วยประมวลผลอีกอย่างหนึ่ง คือ รอบนาฬิกาที่ใช้ในการทำงานหนึ่งคำสั่ง (Cycles per instruction: CPI) ซึ่งในตารางที่ 7.4 แสดงการเปรียบเทียบค่ารอบนาฬิกาที่ใช้ในการทำงานหนึ่งคำสั่ง ของหน่วยประมวลผลที่มีการอัดคำสั่งกับหน่วยประมวลผลที่ไม่มีการอัดคำสั่ง เมื่อเฉลี่ยจากโปรแกรมทดสอบทั้ง 7 โปรแกรม พบว่าการอัดคำสั่งช่วยลดรอบนาฬิกาที่ใช้ในการทำงานหนึ่งคำสั่งลงได้ประมาณ 0.39 รอบนาฬิกาต่อคำสั่ง

ตารางที่ 7.4 การเปรียบเทียบรอบนาฬิกาที่ใช้ในการทำงานหนึ่งคำสั่งของโปรแกรมที่ได้รับการอัดคำสั่งกับโปรแกรมปกติ

โปรแกรมทดสอบ	ค่า CPI	
	ไม่อัดคำสั่ง	อัดคำสั่ง
Bubble	2.28	1.96
Merge	2.19	1.76
Quick (100 items)	2.30	1.96
Fibo	2.00	1.50
Hanoi	2.13	1.73
Sieve	2.21	1.84
AES	2.22	1.84
เฉลี่ย	**2.19**	**1.80**

7.3 วิเคราะห์ขนาดของโปรแกรมของหน่วยประมวลผล

ดังที่ได้กล่าวไปแล้วว่าการอัดคำสั่งช่วยในโปรแกรมมีขนาดเล็กลงถึงประมาณร้อยละ 30 ของขนาดโปรแกรมปกติ แต่ยังขาดข้อมูลที่ใช้เป็นเกณฑ์ในการบอกว่าโปรแกรมนั้นเล็กพอหรือยัง ในส่วนนี้จะทำการเปรียบเทียบขนาดของโปรแกรมของหน่วยประมวลผลนี้ กับโปรแกรมของหน่วยประมวลผลอื่น ซึ่งประกอบด้วยหน่วยประมวลผลแบบแสตก [15] และหน่วยประมวลผลไมโครแบลซ [18]

หน่วยประมวลผลแบบแสตกที่นำมาเปรียบเทียบนี้ มีชุดคำสั่งแบบแสตกซึ่งเป็นที่ยอมรับกันว่าช่วยให้ได้โปรแกรมที่มีขนาดเล็ก ส่วนหน่วยประมวลผลไมโครแบลซเป็นหน่วยประมวลผลขนาด 32 บิตที่มีขายในท้องตลาดทั่วไป

ในตารางที่ 7.5 แสดงการเปรียบเทียบขนาดโปรแกรมของหน่วยประมวลผลแบบแสตกกับโปรแกรมที่ได้รับการอัดคำสั่ง และในตารางที่ 7.6 แสดงการเปรียบเทียบขนาดโปรแกรมของหน่วยประมวลผลแบบไมโครแบลซกับโปรแกรมที่ได้รับการอัดคำสั่ง

ตารางที่ 7.5 การเปรียบเทียบขนาดโปรแกรมระหว่างโปรแกรมที่ได้รับการอัดคำสั่งกับโปรแกรมของหน่วยประมวลผลแบบแสตก

โปรแกรมทดสอบ	ขนาดของโปรแกรมของหน่วยประมวลผลแบบแสตก (ไบต์)	ขนาดของโปรแกรมที่ได้รับการอัดคำสั่ง (ไบต์)	ค่าเปรียบเทียบ (%)
Bubble	124	68	54.8
Quick	187	168	89.8
Fibo	50	36	72.0
Hanoi	153	108	70.6
Sieve	137	84	61.3

เมื่อเปรียบเทียบขนาดของโปรแกรมกับหน่วยประมวลผลแบบแสตก พบว่าขนาดของโปรแกรมของหน่วยประมวลผลที่ได้นำเสนอนี้ มีขนาดเล็กกว่าโปรแกรมของหน่วยประมวลผลแบบแสตก

ตารางที่ 7.6 การเปรียบเทียบขนาดโปรแกรมระหว่างโปรแกรมที่ได้รับการอัดคำสั่งกับโปรแกรมของหน่วยประมวลผลไมโครแบลซ

โปรแกรมทดสอบ	ขนาดของโปรแกรมของหน่วยประมวลผลไมโครแบลซ (ไบต์)	ขนาดของโปรแกรมที่ได้รับการอัดคำสั่ง (ไบต์)	ค่าเปรียบเทียบ (%)
Bubble	172	68	39.5
Merge	396	212	53.5
Quick	264	168	63.6
Fibo	112	36	32.1
Hanoi	168	108	64.3
Sieve	132	84	63.6
AES	1,524	860	56.4

เมื่อเปรียบเทียบขนาดของโปรแกรมกับหน่วยประมวลผลไมโครแบลซ พบว่าโปรแกรมที่ได้รับการอัดคำสั่ง มีขนาดเล็กกว่าโปรแกรมของหน่วยประมวลผลไมโครแบลซอย่างเห็นได้ชัด

จากข้อมูลในส่วนนี้ จึงพอสรุปได้ว่า โปรแกรมของหน่วยประมวลผลที่ได้ออกแบบในงานวิจัยนี้ เมื่อทำการอัดคำสั่งร่วมด้วย จะทำให้ได้โปรแกรมที่ถือได้ว่ามีขนาดเล็กกว่าโปรแกรมของหน่วยประมวลผลขนาด 32 บิตทั่วไป

7.4 วิเคราะห์ประสิทธิภาพของหน่วยประมวลผล

ในส่วนนี้ จะทำการวิเคราะห์ประสิทธิภาพการทำงานของหน่วยประมวลผลที่ได้ออกแบบในงานวิจัยนี้ โดยจะทำการเปรียบเทียบกับหน่วยประมวลผลไมโครแบลซ ค่าที่ใช้ในการเปรียบเทียบประกอบด้วย จำนวนรอบนาฬิกาที่ใช้ในการทำงานโปแกรมทดสอบ และค่าความถี่สูงสุดของสัญญาณนาฬิกา

ในตารางที่ 7.7 แสดงการเปรียบเทียบจำนวนรอบนาฬิกาที่ใช้ในการทำงานโปรแกรมทดสอบของหน่วยประมวลผลทั้งสองตัว พบว่าในโปรแกรมทดสอบส่วนใหญ่หน่วยประมวลผลที่ออกแบบในงานวิจัยนี้ ใช้รอบนาฬิกามากกว่าหน่วยประมวลผลไมโครแบลซประมาณร้อยละ 15

ตารางที่ 7.7 การเปรียบเทียบประสิทธิภาพการทำงานระหว่างหน่วยประมวลผลไมโครแบลซกับหน่วยประมวลผลที่มีการอัดคำสั่ง

โปรแกรมทดสอบ	จำนวนรอบนาฬิกาที่ใช้ในการทำงานโปรแกรมทดสอบ		เร็วขึ้น
	หน่วยประมวลผลไมโครแบลซ	หน่วยประมวลผลที่มีการอัดคำสั่ง	
Bubble	248,354	282,953	0.88
Merge	23,389	28,500	0.82
Quick			
20items	4,131	4,575	0.90
60 items	24,439	28,116	0.87
100 items	73,442	70,856	1.04
Fibo	4,810	2,740	1.76
Hanoi	4,102	3,582	1.15
Sieve	1,195	1,854	0.64
AES	43,500	49,098	0.89

ยกเว้นในโปรแกรมทดสอบ Sieve ที่หน่วยประมวลผลของงานวิจัยนี้ ใช้รอบนาฬิกามากกว่าหน่วยประมวลผลไมโครแบลซถึงร้อยละ 55 เนื่องจากโปรแกรมทดสอบดังกล่าวมีการ

คำนวณบ่อยครั้ง หน่วยประมวลผลไมโครแบลซซึ่งมีการทำงานแบบสายท่อ (Pipeline) จึงได้เปรียบจากการคำนวณที่เร็วกว่า

โปรแกรมทดสอบ Fibo และ Hanoi เป็นโปรแกรมที่มีการเรียกเวียนเกิด (Recursive) บ่อยครั้ง การเรียกเวียนเกิดนี้ทำให้ประสิทธิภาพของหน่วยประมวลผลไมโครแบลซต่ำลง เนื่องจากต้องเสียเวลาในการเก็บค่ารีจิสเตอร์ต่างๆ ส่วนหน่วยประมวลผลที่ออกแบบในงานวิจัยนี้มีรีจิสเตอร์น้อย ทำให้การเรียกเวียนเกิดสามารถทำได้อย่างรวดเร็ว จึงทำให้สามารถทำงานโปรแกรมทดสอบทั้งสองโปรแกรมนี้ได้เร็วกว่าหน่วยประมวลผลไมโครแบลซ

ในการทดสอบโปรแกรมทดสอบ Quick แบบข้อมูล 100 ตัว หน่วยประมวลผลไมโครแบลซซึ่งปกติใช้หน่วยความจำเป็นบล็อคแรม (Block RAM) ขนาด 8 กิโลไบต์ มีพื้นที่ไม่เพียงพอสำหรับทำงานโปรแกรมทดสอบดังกล่าว ในการทดสอบจึงต้องเปลี่ยนในหน่วยประมวลผลไมโครแบลซใช้หน่วยความจำเป็นหน่วยความจำสุ่มภายนอก ซึ่งมีการทำงานที่ช้ากว่าบล็อคแรมมาก ทำให้การทำงานของหน่วยประมวลผลช้าลง

หากพิจารณาค่าจำนวนรอบนาฬิกาที่ใช้ในการทำงานโปรแกรมทั่วไป หน่วยประมวลผลที่ออกแบบในงานวิจัยนี้ มิได้ด้อยไปกว่าหน่วยประมวลผลไมโครแบลซมากนัก โดยจากโปรแกรมที่ใช้ทดสอบพบว่า หน่วยประมวลผลนี้ใช้รอบนาฬิกาในการทำงานมากกว่าหน่วยประมวลผลไมโครแบลซเพียงร้อยละ 15

หากพิจารณาค่าความถี่สูงสุดของสัญญาณนาฬิกา แสดงในตารางที่ 7.8 ที่หน่วยประมวลผลทั้งสองสามารถทำงานได้ เนื่องจากหน่วยประมวลผลไมโครแบลซมีการทำงานแบบสายท่อ (Pipeline) หน่วยประมวลผลไมโครแบลซสามารถทำงานได้ที่ความถี่สูงกว่าหน่วยประมวลผลของงานวิจัยนี้

ตารางที่ 7.8 ความถี่สูงสุดของสัญญาณนาฬิกาที่หน่วยประมวลผลสามารถทำงานได้

ค่าความถี่สูงสุด (เมกะเฮิร์ทซ์)	
หน่วยประมวลผลไมโครแบลซ	**หน่วยประมวลผลที่มีการอัดคำสั่ง**
91	63

ในด้านประสิทธิภาพการทำงานนั้น หน่วยประมวลผลของงานวิจัยนี้มีประสิทธิภาพที่ต่ำกว่าหน่วยประมวลผลไมโครแบลซอยู่ แต่หากพิจารณารวมไปถึงทรัพยากรที่ใช้ในการสร้างวงจรหน่วยประมวลผล แสดงในตารางที่ 7.9 แล้ว หน่วยประมวลผลที่ได้นำเสนอไปนี้ ใช้ทรัพยากรน้อยมาก เมื่อเทียบกับหน่วยประมวลผลไมโครแบลซ โดยคิดเป็นประมาณร้อยละ 23.7 ของทรัพยากรที่ใช้ในการสร้างหน่วยประมวลผลไมโครแบลซ และร้อยละ 4.1 ของทรัพยากรที่ใช้ในการสร้างหน่วยประมวลผลไมโครแบลซที่มีบล็อคแรมขนาด 8 กิโลไบต์

ตารางที่ 7.9 ทรัพยากรที่ใช้ในการสร้างหน่วยประมวลผล

จำนวนเกตสมมูล (Equivalent gate)		
หน่วยประมวลผลไมโครแบลซ		**หน่วยประมวลผลที่มีการอัดคำสั่ง**
รวมบล็อคแรมขนาด 8 กิโลไบต์	**ไม่รวมบล็อคแรมขนาด 8 กิโลไบต์**	
315,528	55,000*	13,060

* ค่าที่ระบุด้วยเครื่องหมายนี้เป็นค่าโดยประมาณ

ในตารางที่ 7.9 ขนาดวงจรของหน่วยประมวลผลไมโครแบลซมี 2 ค่า เนื่องจากในการทำงานจริงนั้น หน่วยประมวลผลไมโครแบลซต้องทำงานร่วมกับบล็อคแรม (Block ram) ขนาด 8 กิโลไบต์ (kByte) ซึ่งทำหน้าที่เป็นหน่วยความจำ การสังเคราะห์วงจรจึงเป็นการสังเคราะห์ทั้งวงจรหน่วยประมวลผลและบล็อคแรม จำนวนเกตสมมูล (Equivalent gate) ที่โปรแกรมรายงานออกมาจึงเป็นจำนวนเกตสมมูลของวงจรทั้งสองรวมกัน จำนวนเกตสมมูลของวงจรหน่วยประมวลผลเพียงอย่างเดียว ประมาณโดยการนำจำนวนเกตสมมูลที่ใช้ในการสร้างบล็อคแรม มาลบออกจากจำนวนเกตสมมูลที่ใช้สร้างทั้งสองวงจร ค่าที่ได้จึงเป็นค่าโดยประมาณ ไม่ใช่ค่าที่รายงานออกมาจากโปรแกรมโดยตรง

หากกล่าวอย่างยุติธรรม หน่วยประมวลผลไมโครแบลซและหน่วยประมวลผลของงานวิจัยนี้ออกแบบมาด้วยจุดประสงค์ที่ต่างกัน หน่วยประมวลผลไมโครแบลซนั้นออกแบบมาเพื่องานที่ต้องการการประมวลผลปริมาณมากอย่างรวดเร็ว เช่นการทำงานในเครื่องเล่นเกมส์และมือถือในปัจจุบัน หน่วยประมวลผลไมโครแบลซนี้จะทำงานได้เต็มขีดความสามารถ จำเป็นต้องมีสภาพแวดล้อมการทำงานที่เหมาะสม

แต่สำหรับหน่วยประมวลผลที่ได้ออกแบบในงานวิจัยนี้ เป็นหน่วยประมวลผลที่มีประสิทธิภาพอยู่ในระดับกลาง ไม่เหมาะสำหรับงานที่ต้องการกำลังการประมวลผลที่สูง แต่สำหรับงานทั่วไปนั้น หน่วยประมวลผลนี้สามารถทำงานได้อย่างมีประสิทธิภาพ โดยที่ใช้ทรัพยากรต่ำ จึงมีความเหมาะสมมากกว่าการที่จะใช้หน่วยประมวลผลที่มีขีดความสามารถสูง

7.5 แนวทางการวิจัยในอนาคต

วิธีการอัดคำสั่งที่ใช้ในงานวิจัยนี้ ยังทำการอัดคำสั่งได้ไม่เต็มที่ กล่าวคือ มีคำสั่งบางคำสั่งในโปรแกรมที่ไม่สามารถทำการอัดคำสั่งได้ ดังที่ได้กล่าวไว้ในบทที่ 4 ทำให้ยังมีช่องว่างเหลืออยู่ในโค้ดที่ทำการอัดคำสั่งแล้ว

การปรับปรุงการอัดคำสั่งวิธีใหม่ให้สามารถทำการอัดคำสั่งได้อย่างเต็มที่นั้นสามารถทำได้โดยวิธีการอัดคำสั่งแบบปรับปรุงซึ่งได้แสดงไว้ในบทที่ 6 การอัดคำสั่งแบบปรับปรุงมีโครงสร้างที่รองรับการกระโดดไปยังคำสั่งที่อยู่ในตำแหน่งบิตสูง หรือคำสั่งเติมเต็ม (Complement instruction) ของคำสั่งอัด (Packed instruction) ช่วยให้ข้อจำกัดที่ไม่สามารถอัดคำสั่งที่เป็นปลายทางของการกระโดดและคำสั่งที่เป็นปลายทางของการเรียกโปรแกรมย่อยนั้นถูกกำจัดออกไป

ขนาดของโค้ดที่ได้รับการอัดคำสั่งด้วยวิธีที่ปรับปรุงนี้ จะมีขนาดเล็กกว่าขนาดของโค้ดที่ได้รับการอัดคำสั่งด้วยวิธีที่ใช้ในงานวิจัย เนื่องจากวิธีที่ปรับปรุงไม่มีข้อจำกัดในการอัดคำสั่งเลย จึงทำให้ไม่มีช่องว่างที่ไม่ได้ใช้ประโยชน์อยู่ในโค้ด

เหตุผลที่งานวิจัยนี้เลือกนำเสนอวิธีการอัดคำสั่งที่ได้นำเสนอไปในบทที่ 4 ทั้งที่วิธีการอัดคำสั่งแบบปรับปรุงที่นำเสนอในบทที่ 6 สามารถสร้างโค้ดได้มีขนาดเล็กกกว่า เพราะประสิทธิภาพการทำงานโค้ดที่ได้รับการอัดคำสั่งแบบปรับปรุงนี้ ไม่ได้ดีกว่าประสิทธิภาพการทำงานของโค้ดที่ได้รับการอัดคำสั่งแบบที่ใช้ในงานวิจัยนี้เสมอไป การอัดคำสั่งด้วยวิธีที่ปรับปรุงจำเป็นต้องพึ่งพาตัวสร้างโค้ด (Code generator) เพื่อให้ได้โค้ดที่มีประสิทธิภาพการทำงานที่ดีกว่าโค้ดที่ได้จากวิธีการอัดคำสั่งที่ได้นำเสนอไป หากการจัดเรียงคำสั่งลงในหน่วยความจำทำได้ไม่ดี จะส่งผลให้ประสิทธิภาพการทำงานต่ำลง

หากพัฒนาตัวสร้างโค้ดสำหรับการอัดคำสั่งแบบปรับปรุงให้มีคุณภาพ สามารถจัดเรียงคำสั่งลงในหน่วยความจำได้อย่างมีประสิทธิภาพ อย่างไรก็ตามโค้ดที่ได้จากวิธีการอัดคำสั่งแบบปรับปรุงนี้ มีศักยภาพในด้านประสิทธิภาพการทำงานที่สูงกว่าโค้ดที่ได้จากวิธีการอัดคำสั่งที่ได้นำเสนอไปเพียงเล็กน้อยเท่านั้น เนื่องจากโค้ดที่ได้รับการอัดคำสั่งแบบปรับปรุงนี้ เมื่อถูกปรับให้มีประสิทธิภาพการทำงานที่สูง จะมีโครงสร้างเหมือนโค้ดที่ได้จากการอัดคำสั่งที่ใช้ในงานวิจัยนี้ โดยมีขนาดเล็กกว่าเล็กน้อย ซึ่งจุดนั้นเป็นข้อได้เปรียบจุดเดียวของโค้ดที่ได้รับการอัดคำสั่งแบบปรับปรุง

7.6 สรุปผลการวิจัย

จากหน่วยประมวลผลแบบแสตกที่ได้พัฒนาโดย นายอลงกต บุรุษอาชาไนยและคณะ [15] ได้นำการดำเนินการบนแสตก (Stack) มาประยุกต์สร้างเป็นชุดคำสั่งของวงจรหน่วยประมวลผล ทำให้ได้เป็นชุดคำสั่งที่ช่วยให้โปรแกรมมีขนาดเล็ก อีกทั้งวงจรหน่วยประมวลผลที่มีขนาดเล็ก ทำให้หน่วยประมวลผลนี้มีความเหมาะสมกับการใช้งานในอุตสาหกรรมระบบฝังตัว

ในงานวิจัยนี้มุ่งเน้นปรับปรุงข้อด้อยของหน่วยประมวลผลแบบแสตกคือ ความเร็วในการทำงาน โดยการปรับเปลี่ยนชุดคำสั่งกำจัดการทำงานแบบแสตกที่ช้าออกไป ทำให้ได้ชุดคำสั่งซึ่งทำงานได้เร็วขึ้น แต่ก็ทำให้โปรแกรมที่เขียนด้วยชุดคำสั่งใหม่นี้ มีขนาดใหญ่ขึ้นเช่นกัน เพื่อลด

ผลกระทบนี้จึงได้พัฒนาวิธีการอัดคำสั่งมาใช้กับชุดคำสั่งใหม่นี้ หน่วยประมวลผลที่พัฒนาขึ้นมาใหม่จึงมีประสิทธิภาพการทำงานที่ดีขึ้น ในขณะที่ยังรักษาขนาดของโปรแกรมที่เล็กไว้ได้

งานวิจัยนี้ผลงานคือได้หน่วยประมวลผลขนาด 32 บิตตัวหนึ่ง ซึ่งมีการทำงานคล้ายเครื่องตัวสะสม (Accumulator machine) ได้รับการออกแบบให้วงจรมีขนาดเล็ก โดยที่ยังเพียงพอที่จะทำงานต่างๆ ได้อย่างมีประสิทธิภาพ หน่วยประมวลผลที่ได้ออกแบบประกอบด้วยรีจิสเตอร์ 6 ตัว และทางเดินข้อมูลที่ไม่ซับซ้อน

การอัดคำสั่ง ทำโดยการใส่คำสั่งรูปแบบสั้น 2 คำสั่งลงในหน่วยความจำตำแหน่งเดียว โดยในกรณีที่เป็นคำสั่งรูปแบบยาวให้แบ่งคำสั่งนั้นออกเป็น 2 ส่วน ส่วนละ 16 บิตแล้วปฏิบัติเช่นเดียวกับคำสั่งรูปแบบสั้น ด้วยวิธีการนี้เมื่อหน่วยประมวลผลอ่านหน่วยความจำตำแหน่งดังกล่าว จะทำให้ได้คำสั่งรูปแบบสั้น 2 คำสั่ง จากปกติที่การอ่านหน่วยความจำหนึ่งครั้งจะได้คำสั่งคำสั่งรูปแบบสั้นคำสั่งเดียว

วิธีการอัดคำสั่งที่ใช้ในงานวิจัยนี้ เป็นวิธีที่ไม่ซับซ้อน การปรับปรุงหน่วยประมวลผลในรองรับการทำงานแบบอัดคำสั่งนั้น สามารถทำได้โดยใช้ทรัพยากรไม่มาก ซึ่งเป็นข้อดีของวิธีการอัดคำสั่งนี้ อีกทั้งการอัดคำสั่งมีกฎเกณฑ์ที่ชัดเจน ไม่จำเป็นต้องอาศัยตัวสร้างโค้ด (Code generator) ที่มีประสิทธิภาพ

หน่วยประมวลผลที่รองรับการทำงานกับโปรแกรมที่ได้รับการอัดคำสั่งด้วยวิธีที่ใช้ในงานวิจัยนี้ได้ทำการเปรียบเทียบแบ่งเป็น 2 ด้านคือ ด้านขนาดของโปรแกรม และด้านประสิทธิภาพการทำงานและการใช้ทรัพยากรของหน่วยประมวลผล

ในด้านขนาดของโปรแกรมนั้น เมื่อเทียบกับขนาดโปรแกรมของหน่วยประมวลผลแบบแสตก [15] ซึ่งเป็นที่ยอมรับว่า ชุดคำสั่งแบบแสตกนั้นช่วยให้โปรแกรมมีขนาดเล็ก พบว่ามีขนาดที่ใกล้เคียงกัน และเมื่อเปรียบเทียบกับขนาดโปรแกรมของหน่วยประมวลผลไมโครแบลซซึ่งเป็นหน่วยประมวลผล 32 บิตด้วยกันแล้ว พบว่าขนาดโปรแกรมของหน่วยประมวลผลที่พัฒนาในงานวิจัยนี้มีขนาดเล็กกว่าอย่างเห็นได้ชัด โดยมีขนาดเล็กกว่าร้อยละ 50 ถึง 60 ของขนาดโปรแกรมของหน่วยประมวลผลไมโครแบลซ

ในด้านประสิทธิภาพการทำงาน เนื่องจากหน่วยประมวลผลไมโครแบลซ (Micro Blaze) เป็นหน่วยประมวลผลความเร็วสูง ได้ออกแบบมาเพื่อใช้งานในสภาพแวดล้อมที่ต้องการการคำนวณที่รวดเร็ว มีการทำงานแบบสายท่อ (Pipeline) คำสั่งส่วนใหญ่ทำงานเสร็จได้ภายในหนึ่งรอบนาฬิกา ในขณะที่หน่วยประมวลผลในงานวิจัยนี้ออกแบบมา มีจุดประสงค์ให้วงจรมีขนาดเล็กและใช้ทรัพยากรต่างๆอย่างคุ้มค่า หน่วยประมวลผลที่ออกแบบในงานวิจัยนี้จึงมีประสิทธิภาพการทำงานที่ต่ำกว่าหน่วยประมวลผลไมโครแบลซ

ในด้านรอบนาฬิกาที่ใช้ในการทำงานโปรแกรมทดสอบนั้น หน่วยประมวลผลที่มีการอัดคำสั่ง ใช้มากกว่าหน่วยประมวลผลไมโครแบลซเพียงประมาณร้อยละ 15 เท่านั้น ซึ่งถือว่าต่างกัน

ไม่มาก และหากพิจารณาโปรแกรมที่มีการเรียกโปรแกรมย่อยแบบเวียนเกิด (Recursive) จะเห็นว่าหน่วยประมวลผลนี้ทำงานได้ดีกว่าหน่วยประมวลผลแบบไมโครแบลซ เนื่องจากหน่วยประมวลผลนี้โครงสร้างการเรียกโปรแกรมย่อยที่รวดเร็ว

อย่างไรก็ตามการพิจารณาประสิทธิภาพในการทำงานนั้น ต้องคำนึงถึงความถี่นาฬิกาที่ใช้ด้วย ความถี่ของสัญญาณนาฬิกาสูงสุดที่วงจรหน่วยประมวลผลทำงานได้นั้น เป็นค่าที่ได้จากผลการสังเคราะห์วงจรหน่วยประมวลผลด้วยโปรแกรม Xilinx WebPack รุ่น 8.1i ค่าความถี่ที่ได้คือ 91 เมกะเฮิร์ทซ์ (MHz) และ 63 เมกะเฮิร์ทซ์สำหรับหน่วยประมวลผลไมโครแบลซและหน่วยประมวลผลที่ได้ออกบบในงานวิจัยนี้ตามลำดับ

แม้ว่าประสิทธิภาพการทำงานของหน่วยประมวลผลที่มีการอัดคำสั่ง จะต่ำกว่าหน่วยประมวลผลไมโครแบลซ แต่ผลที่ได้นั้นเป็นที่น่าพอใจ เนื่องจากประสิทธิภาพนั้นต่ำกว่าไม่มาก ในขณะที่วงจรหน่วยประมวลผลที่ได้ออกแบบนั้น มีขนาดเล็กกว่าหน่วยประมวลผลไมโครแบลซหลายเท่าตัว ซึ่งแสดงให้เห็นถึงประสิทธิภาพในการใช้ทรัพยากรของหน่วยประมวลผลที่ได้ออกแบบนี้