การพัฒนาหน่วยประมวลผลแบบไปป์ไลน์ ชนิดอสมวาร

A Development of Asynchronous Pipeline Processor

กิตติมา พันสินทวีสุข, ตะวัน ภูรัต

ห้องปฏิบัติการวิศวกรรมระบบดิจิตอล ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสยาม

235 มหาวิทยาลัยสยาม บางหว้า ภาษีเจริญ กรุงเทพฯ 10160 โทรศัพท์ 66-2-457-0068 ต่อ 210, แฟกซ์ 66-2-457-3982

E-mail: hlin_22@hotmail.com, tawan@siamu.ac.th

บทคัดย่อ

ในบทความนี้ได้นำเสนอการพัฒนาหน่วยประมวลผลไปป์ไลน์แบบอสมวาร ขนาด 16 บิต โดยมีวัตถุประสงค์หลักเพื่อใช้เป็นหน่วยประมวลผลบนอุปกรณ์เอฟพีจีเอโดยได้นำเอาข้อดีของวงจรอสมวารคือการใช้พลังต่ำ ทั้งยังได้นำวิธีการเข้ารหัสข้อมูลแบบหนึ่งในสี่แทนรหัสรางคู่สามารถช่วยทำให้สามารถประหยัดพลังงานลงได้ หน่วยประมวลผลที่พัฒนาขึ้นนี้สร้างด้วยภาษาเวอริลอกซึ่งเป็นภาษาบรรยายพฤติกรรมของวงจรโดยแบ่งการทำงานออกเป็นสามขั้นคือ ขั้นการอ่านคำสั่งและการถอดรหัส ขั้นการทำงาน และขั้นการเขียนผลลัพธ์ และเพื่อตรวจสอบการทำงานของชุดคำสั่งต่างๆจึงได้ใช้โปรแกรมโมเดลซิม เอ็กซ์อี 6.1 ในการจำลองการทำงานของหน่วยประมวลผลนี้ซึ่งผลทดสอบแสดงให้เห็นว่าหน่วยประมวลผลนี้สามารถดำเนินการตามชุดคำสั่งต่างๆได้เป็นอย่างดี

Abstract

A Development of Pipeline Asynchronous Processor using hardware description language has been proposed in this article. The main objective is to provide a SOFT-CPU in order to apply with FPGA, and to include the advantage of asynchronous circuit which is the low power consumption. Additional, replacing dual-rail with 1 of 4 encoding, the circuits outperform power consumption. Using the Verilog, hardware description language, this processor has been divided into 3 stage of pipeline: fetch& decode, execution, and write result. To verify the operation of its instruction sets, the ModelSIM XE 6.1 software was used to simulate the operations of this processor. As a result, this processor can operate its instruction sets well.

1บทนำ

การพัฒนาส่วนควบคุมสำหรับงานอัตโนมัตินั้นสามารถกระทำได้ในหลายลักษณะเช่น ผู้ที่มีความรู้ทางด้านการออกแบบวงจรดิจิตอลเป็นอย่างดี ก็มักจะออกแบบส่วนควบคุมทั้งหมดโดยใช้วงจรดิจิตอลซึ่งจะมีข้อดีคือได้วงจรที่มีความเร็วสูง และยิ่งนำไปใช้กับอุปกรณ์ที่สามารถโปรแกรมได้อย่างเช่น เอฟพีจีเอ (Filed Programmable Gate Array:FPGA) ก็จะทำให้สามารถรวมเอาวงจรต่างๆที่เกี่ยวข้องบรรจุไว้ภายในชิพเดียว อย่างไรก็ตามถึงแม้จะมีข้อดีหลายประการแต่ปัญหาที่สำคัญคือจะต้องมีความรู้และทักษะในการออกแบบวงจรดิจิตอลเป็นอย่างดีจึงจะสามารถใช้วิธีนี้อย่างได้ประโยชน์สูงสุด แต่อีกวิธีการหนึ่งซึ่งเหมาะสำหรับผู้ที่ถนัดในด้านการเขียนโปรแกรมเพื่อควบคุมการทำงานก็อาจทำการออกแบบส่วนควบคุมดังกล่าวโดยการใช้ไมโครโพรเซสเซอร์ร่วมกับวงจรขับสัญญาณเข้าออก ซึ่งจะเขียนโปรแกรมเพื่อควบคุมการทำงานให้ได้ตามที่ต้องการ แต่การใช้โมโครโพรเซสเซอร์นั้นผู้ใช้จะต้องออกแบบวงจรไฟฟ้าซึ่งในกรณีที่ต้องการวงจรอื่นๆเพิ่มเติมก็จะต้องต่อวงจรเพิ่มเติมทำให้มีการเชื่อมสายสัญญาณต่างๆ เพิ่มเติมและอาจก่อให้เกิดปัญหาที่ไม่พึงประสงค์ได้ อย่างไรก็ตามถ้าหากสามารถนำเอาไมโครโพรเซสเซอร์ไปใช้งานบนเอฟพีจีเอได้ก็จะทำให้เกิดข้อดีเพิ่มเติมขึ้นคือผู้ใช้ที่ถนัดการเขียนโปรแกรมก็สามารถใช้งานได้และสามารถรวมเอาวงจรต่างๆที่จำเป็นบรรจุรวมไว้ภายในชิพเอฟพีจีเอเพียงชิพเดียวก็จะทำให้ได้รับความสะดวกมากยิ่งขึ้น และจากแนวโน้มความต้องการที่จะลดการใช้พลังงานลงทำให้หน่วยประมวลผลที่กินพลังงานต่ำจึงเป็นที่สนใจของนักพัฒนา และเมื่อพิจารณาการทำงานของวงจรอสมวาร(Asynchronous circuit) ซึ่งปราศจากสัญญาณนาฬิกาในการกำกับจังหวะของ

วงจรเมื่อเทียบกับวงจรสมวาร (synchronous circuit) ที่ใช้สัญญาณนาฬิกาในการกำกับจังหวะของวงจรแล้วพบว่า มีข้อดีหลายประการได้แก่ มีความเร็วในการทำงานสูงกว่า ใช้พลังงานน้อยกว่าและแพร่กระจายสัญญาณรบกวนต่ำกว่า และเมื่อนำไปรวมกับการทำงานแบบไปป์ไลน์ ทำให้การทำงานของหน่วยประมวลผลเร็วขึ้นและกินพลังงานน้อยลง นอกจากนี้การใช้โปรโตคอลการเข้ารหัสข้อมูลแบบหนึ่งในสี่ (1 of 4 Encoding) ซึ่งสนใจการเปลี่ยนแปลงของสายสัญญาณที่เกิดขึ้นเพียงเส้นเดียว เมื่อสายสัญญาณทั้งสี่เส้นนั้นมีเส้นใดเส้นหนึ่งมีการเปลี่ยนแปลงส่งผลให้รู้ว่ามีข้อมูลใหม่เข้า จากการใช้สายสัญญาณเพียงเส้นเดียวในการเปลี่ยนแปลงทำให้ประหยัดพลังงานมากขึ้น ดังนั้นในงานวิจัยนี้จึงได้พยายามออกแบบหน่วยประมวลผลที่กินพลังงานต่ำและมีความเร็วการทำงานสูงและสามารถนำไปใช้ได้กับเอฟพีจีเอขึ้น

2งานวิจัยที่เกี่ยวข้อง

2.1 การประมวลผลแบบไปป์ไลน์ [1]

การประมวลผลแบบไปป์ไลน์ส่วนใหญ่แบ่งขั้นตอนการทำงานออกเป็น 5ขั้นตอน คือ

- *ขั้นการอ่านคำสั่ง* (Instruction Fetch)

ส่วนนี้จะทำหน้าที่อ่านคำสั่งใหม่ทั้งจากหน่วยความจำหลัก หรือจากใน Instruction Cache เข้ามา เพื่อรอส่งต่อให้ภาคถอดรหัสดำเนินการต่อไป

- *ขั้นการถอดรหัสคำสั่ง* (Instruction Decode)

ส่วนนี้จะทำหน้าที่แยกคำสั่งต่างๆ ถ้าเป็นสถาปัตยกรรมแบบซีไอเอสซี (Complex Instruction Set CPU: CISC) แต่ละคำสั่งจะมีขนาดที่ไม่แน่นอน ตรงส่วนนี้ก็จะทำการแบ่งคำสั่งนั้นออกเป็นคำสั่งย่อยๆให้มีความยาวเท่าๆกัน แล้วจึงดำเนินการเหมือนในสถาปัตยกรรมแบบอาร์ไอเอสซี (Reduced Instruction Set CPU: RISC) ซึ่งเรียกคำสั่งย่อยๆนั้นว่าไมโครโอเปอเรชัน (Micro Operation)

- *ขั้นการอ่านตัวกระทำ* (Operands Fetch)

ส่วนนี้จะทำหน้าที่รับข้อมูลที่จะใช้ในภาคประมวลผลเข้ามาเก็บไว้ เช่น จากภาคถอดรหัสคำสั่งเมื่อทราบว่าเป็นคำสั่งคูณก็ต้องทำการอ่านค่าที่จะใช้ในการคูณมาด้วยอีก 2 ค่า ซึ่งในการออกแบบบางครั้งภาคอ่านตัวกระทำนี้อาจถูกรวมไว้กับภาคถอดรหัสคำสั่ง

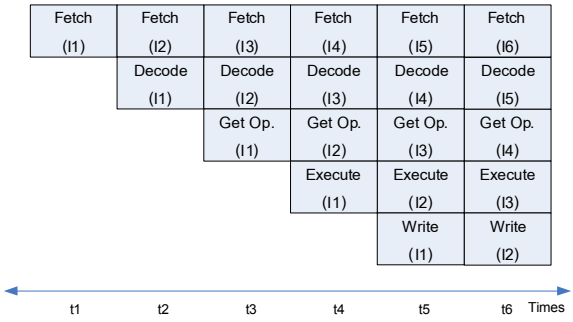
- *ขั้นการประมวลผล* (Execute)

ส่วนนี้เป็นขั้นตอนที่ทำการประมวลผลตามคำสั่งและการกระทำที่ได้รับมาจากภาคถอดรหัสคำสั่งและภาครับข้อมูล

- *ขั้นการเขียนผลลัพธ์* (Write Result)

เมื่อมีการประมวลผลเสร็จเรียบร้อยแล้ว ผลลัพธ์ที่ได้ก็จะนำไปเก็บไว้ในรีจิสเตอร์ หรือในแคช ซึ่งบางครั้งขั้นตอนนี้ก็ถูกจัดรวมไว้กับขั้นการประมวลผล

การประมวลผลแบบไปป์ไลน์นี้ จะช่วยให้หน่วยประมวลผลมีความสามารถในการประมวลผลได้รวดเร็วยิ่งขึ้น จะเห็นได้จากรูปที่ 1



รูปที่ 1 การทำงานของหน่วยประมวลผลแบบไปป์ไลน์

จากรูปที่ 1 เมื่อนำหน่วยประมวลผลมีการทำงานเป็นแบบไปป์ไลน์จะสามารถช่วยลดเวลาในการทำงานได้เนื่องจากมีการทำงานในลักษณะต่อเนื่องกันดังจะเห็นได้จากที่จังหวะเวลา t_1 ชุดคำสั่งI_1 จะเริ่มขั้นตอนอ่านคำสั่ง และส่งต่อไปทำการถอดรหัสคำสั่งที่เวลา t2 พร้อมๆกับที่หน่วยอ่านคำสั่งเองก็จะทำการอ่านคำสั่ง I_2 เข้ามา ซึ่งคำสั่ง I_1 จะประมวลผลเสร็จที่เวลา t_5 และคำสั่งI_2 จะประมวลผลเสร็จที่เวลา t_6 แต่ถ้าหน่วยประมวลผลไม่ได้ทำงานแบบไปป์ไลน์เมื่อจะต้องกระทำทีละคำสั่งโดยคำสั่ง I_1 จะทำงานเสร็จที่เวลา t_5 แล้วจึงเริ่มต้นอ่านคำสั่ง I_2 ที่เวลา t_6 และจะทำคำสั่ง I_2 เสร็จที่เวลา t_{10} ดังนั้นจึงเห็นได้ชัดว่าหากมีการทำงานแบบไปป์ไลน์แล้วหน่วย

ประมวลผลจะทำงานเสร็จโดยใช้เวลาเพียง t_6 เท่านั้นซึ่งทำให้เวลาในการทำงานเร็วขึ้น

2.2 วงจรสมวารและวงจรอสมวาร [2,3]

การออกแบบวงจรอสมวารได้มีการวิจัยและคิดค้นขึ้นในราวปี 1940 โดยนายอลัน ทัวร์ลิ่ง[2,3] เนื่องจากพบความแตกต่างของวงจรสมวารและวงจรอสมวารดังนี้

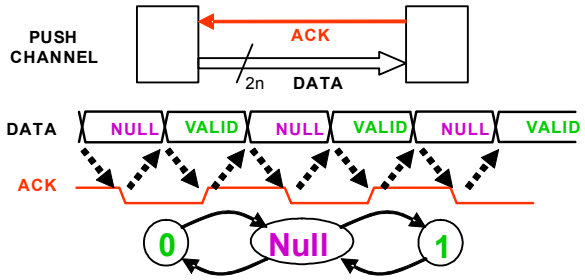
- *สัญญาณนาฬิกา* ในวงจรสมวารการกำกับจังหวะของการทำงานจะใช้สัญญาณนาฬิกาเป็นตัวควบคุมจังหวะในการรับส่งข้อมูลในระบบให้มีการทำงานที่พร้อมเพรียงกัน ซึ่งในทางปฏิบัติระยะทางจะส่งผลต่อความคลาดเคลื่อนของสัญญาณนาฬิกา กล่าวคือ ส่วนที่อยู่ไกลกว่าจะได้รับสัญญาณจากสัญญาณนาฬิกาได้ช้ากว่าส่วนที่อยู่ใกล้ทำให้เกิดความคลาดเคลื่อนขึ้น ดังนั้นหากวงจรมีขนาดใหญ่ปัญหาก็จะส่งผลมากขึ้นด้วย การแก้ปัญหานี้ของวงจรสมวารทำได้โดยการลดความถี่ของสัญญาณนาฬิกาลงซึ่งจะทำให้คาบเวลามีความยาวเพิ่มขึ้นเพื่อการประกันให้สัญญาณต่างๆทำงานเสร็จสิ้นแน่นอน แต่เนื่องจากแนวโน้มของการออกแบบวงจรสมวารจะมีขนาดใหญ่ขึ้นและต้องการความเร็วที่สูงขึ้น จึงทำให้ไม่อาจลดความถี่ของสัญญาณนาฬิกาลงได้ แต่ในวงจรอสมวารไม่ต้องอาศัยสัญญาณนาฬิกาเป็นตัวควบคุมจังหวะของการรับส่งข้อมูลในระบบ ทำให้ปัญหาที่เกิดจากสัญญาณนาฬิกาหมดไป

- *พลังงาน* วงจรสมวารใช้พลังงานมากกว่าวงจรอสมวาร เนื่องจากวงจรสมวารนั้นใช้สัญญาณนาฬิกาในการควบคุมการทำงาน เมื่อวงจรต่างๆภายในชิบได้รับสัญญาณนาฬิกาจะทำให้วงจรนั้นๆเกิดการทำเปลี่ยนแปลงระดับสัญญาณขึ้นก็จะทำให้วงจรเหล่านั้นเกิดการใช้พลังงานขึ้นโดยไม่จำเป็น ซึ่งวงจรอสมวารจะไม่มีการใช้สัญญาณนาฬิกา และการทำงานของวงอสมวารจะเกิดขึ้นเฉพาะบริเวณที่เกี่ยวข้องเท่านั้นทำให้วงจรที่ไม่มีการทำงานก็จะไม่มีการใช้พลังงาน ทำให้วงจรอสมวารใช้พลังงานน้อยกว่าวงจรสมวาร

2.3 สัญญาณรหัสรางคู่ (Dual-Rail Signaling)

สายสัญญาณรหัสรางคู่เป็นโปรโตคอลที่ใช้ในการกำกับจังหวะของวงจรแทนการใช้สัญญาณนาฬิกาในวงจรสมวาร ซึ่งประกอบด้วยสายสัญญาณตอบรับ หรือเรียกว่า “ACK” (Acknowledge) 1 เส้น และสายสัญญาณส่งข้อมูล 2n เส้น กล่าวคือหากต้องการส่งข้อมูลจำนวน n บิตจะต้องใช้สายสัญญาณจำนวน 2n เส้น

เมื่อมีข้อมูลเข้ามาและค่าในสายสัญญาณ ACK ของฝั่งผู้รับมีค่าเป็น 0ก็จะเกิดการส่งข้อมูลโดยระหว่างการส่งผู้รับจะปรับค่าในสายสัญญาณ ACK ให้มีค่าเท่ากับ 1เพื่อป้องกันมิให้ผู้ส่งทำการส่งข้อมูลอื่นทับเข้ามาและเมื่อได้รับข้อมูลเรียบร้อยแล้วจะปรับค่าในสายสัญญาณ ACK ให้มีค่าเท่ากับ 0 อีกครั้งเพื่อพร้อมสำหรับการรับข้อมูลในครั้งถัดไป การทำงานดังกล่าวแสดงดังรูป 2



รูปที่ 2กรทำงานของการเข้ารหัสข้อมูลแบบหนึ่งในสี่

*หมายเหตุ : ค่า **VALID** หมายถึง มีข้อมูล ,*
*ค่า **NULL** หมายถึงขั้นว่างหรือไม่มีข้อมูล*

จากรูปที่ 2 ในสายสัญญาณข้อมูลมีการเปลี่ยนแปลงระหว่างขั้นว่างและขั้นการมีข้อมูลสลับกันไปมาตามลำดับซึ่งจะสอดคล้องกับจังหวะการเปลี่ยนแปลงของการตอบรับในสายสัญญาณตอบรับ และการผังการเปลี่ยนสถานะแสดงให้เห็นว่าข้อมูลจะมีการเปลี่ยนแปลงจากขั้นว่างไปสู่การมีข้อมูลซึ่งอาจเป็น 0 หรือ 1 ก็ได้และจะมีการเปลี่ยนกลับไปสู่ขั้นว่างก่อนจะมีการส่งข้อมูลถัดไปเข้ามา

เมื่อต้องการส่งข้อมูลขนาด 2 บิตจะต้องใช้สายส่งข้อมูลขนาด 4 เส้นดังตารางที่ 2 ซึ่งปกติแล้วเมื่อยังไม่มีการรับส่งข้อมูลสายทั้งสี่เส้นจะมีค่าเป็น “0000” และหากมีการส่งข้อมูลเช่น 01 ก็จะทำให้สายทั้งสี่เส้นเปลี่ยนแปลงจากขั้นว่างไปเป็น “0110” และกลับมาสู่ขั้นว่างคือ”0000” ตามลำดับ

ตารางที่ 1 ค่าความจริงในวงจรรหัสรางคู่

Data		wire[3]	wire[2]	wire[1]	wire[0]
data[1]	data[0]				
0	0	0	1	0	1
0	1	0	1	1	0
1	0	1	0	0	1
1	1	1	0	1	0
Null		0	0	0	0

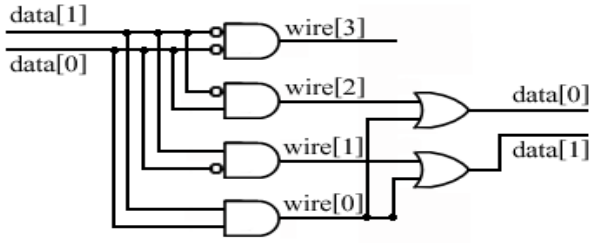
2.4 **สัญญาณรหัสแบบหนึ่งในสี่** (1-of 4 Signaling) [4]

รหัสหนึ่งในสี่นั้นเป็นการเข้ารหัสข้อมูลครั้งละ 2 บิต พร้อมกันโดยใช้สายจำนวน 4 เส้นเพื่อแทนข้อมูลขนาด 2 บิต ดังกล่าวซึ่งจะมีคุณสมบัติการไม่ไวต่อความหน่วงเช่นเดียวกันกับรหัสรางคู่ สัญญาณในสายจะเป็นดังตารางที่ 2 กล่าวคือเมื่อสัญญาณอยู่ในขั้นว่างสายทั้งสี่เส้นจะมีค่าเป็น"0000" เช่นเดียวกันกับในรหัสรางคู่ แต่เมื่อมีการส่งข้อมูล 00 จะทำให้เกิดการเปลี่ยนในสายทั้งสี่เส้นเป็น "1000" และกลับเป็นขั้นว่าง หรือ "0000" หรือถ้ามีการส่งข้อมูลเป็น 01 ก็จะมีการเปลี่ยนแปลงในสายสัญญาณเป็น "0100" และกลับสู่ขั้นว่างต่อไป

ตารางที่ 2 ค่าความจริงในวงจรสายสัญญาณแบบหนึ่งในสี่

Data		wire[3]	wire[2]	wire[1]	wire[0]
data[1]	data[0]				
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1
Null		0	0	0	0

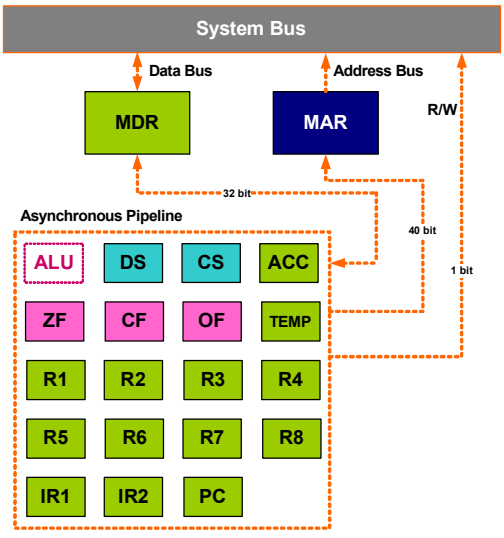
ข้อดีของสายสัญญาณแบบหนึ่งในสี่คือใช้พลังงานในวงจรน้อย เนื่องจากการเปลี่ยนแปลงของข้อมูลจะเปลี่ยนค่าในสายสัญญาณเพียงครั้งละเส้น ดังสังเกตได้เมื่อเทียบจากตารางที่ 1 กับตารางที่ 2 จะพบว่า สายสัญญาณแบบหนึ่งในสี่จะเปลี่ยนค่าในสายสัญญาณทีละเส้นต่อข้อมูลจำนวน 2 บิต แต่สายสัญญาณรหัสรางคู่จะเปลี่ยนค่าในสายสัญญาณทีละ 2 เส้นต่อข้อมูลจำนวน 2 บิต อย่างไรก็ตามการเข้ารหัสแบบหนึ่งในสี่ต้องการวงจรเพื่อช่วยในการเข้ารหัสและถอดรหัสจากฐานสองไปเป็นรหัสหนึ่งในสี่ ซึ่งกลุ่มของวงจรแอนด์เกต ในรูปที่ 3 จะทำหน้าที่แปลงสัญญาณจากฐานสองไปเป็นรหัสหนึ่งในสี่ และในทำนองเดียวกัน วงจรกลุ่มของออร์เกตด้านขวามือจะทำการแปลงรหัสจากหนึ่งในสี่กลับไปเป็นฐานสอง



รูปที่ 3 วงจรแปลงรหัสระหว่างรหัสฐานสองและรหัสหนึ่งในสี่

3. การออกแบบ

การออกแบบหน่วยประมวลผลแบบไปป์ไลน์ ชนิดอสมวาร ได้แบ่งออกเป็น 3 ส่วนหลักๆ คือส่วนควบคุมการทำงาน ส่วนดำเนินการทางคณิตศาสตร์และตรรก และหน่วยความจำหรือรีจิสเตอร์ ซึ่งผู้ออกแบบได้ทำการออกแบบโครงสร้างของหน่วยประมวลผลได้ดังรูปที่ 4



รูปที่ 4 โครงสร้างของหน่วยประมวลผล DSEL 1

จากรูปที่ 4 จะเห็นได้ว่าเมื่อใช้รหัสหนึ่งในสี่ทำให้สายสัญญาณต่างๆ เพิ่มขึ้นเป็นสองเท่าเช่นในกลุ่มของสัญญาณบัสข้อมูล (Data Bus) ขนาด 16 บิตก็ทำให้ต้องใช้สายสัญญาณ 32 เส้น และในกลุ่มของสัญญาณชี้ตำแหน่ง (Address Bus) มีสัญญาณขนาด 20 บิต ทำให้หน่วยประมวลนี้เข้าถึงหน่วยความจำได้สูงสุดถึง 1 MB

3.1 หน่วยควบคุม

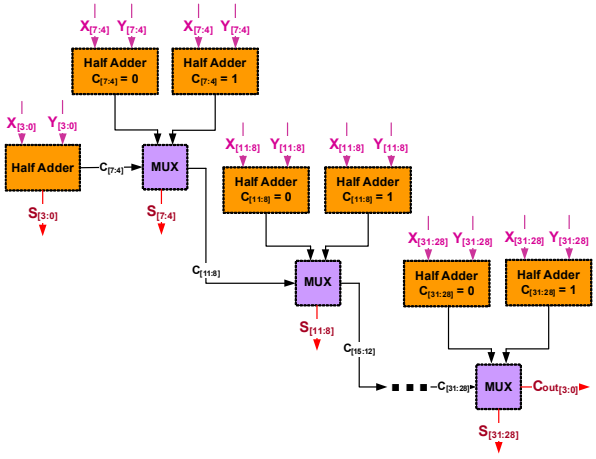
หน่วยควบคุม (Control Unit หรือ CU) ทำหน้าที่ติดต่อสื่อสารกับหน่วยคณิตศาสตร์และตรรก (Arithmetic and Logic Unit: ALU) และหน่วยความจำหลัก โดยกำหนดเส้นทางการส่งข้อมูลรวมทั้งถอดรหัสและควบคุมการถอดรหัสให้เป็นไปตามขั้นตอนการทำงานเพื่อให้ได้ผลลัพธ์ออกมา ซึ่งชุดคำสั่งจะเป็นตัวกำหนดการทำงานของหน่วยควบคุม

3.2 ส่วนดำเนินการทางคณิตศาสตร์และตรรก

ส่วนดำเนินการทางคณิตศาสตร์และตรรก เป็นส่วนที่เกี่ยวข้องกับการดำเนินการทางคณิตศาสตร์และตรรก ซึ่งวงจรคณิตศาสตร์ มีวงจรย่อย คือ วงจรบวกเลขแบบเลือกตัวทดล่วงหน้า วงจรบวกเลข วงจรเลื่อนบิตทางซ้าย/ขวาทั้งแบบคิดเครื่องหมายและไม่คิดเครื่องหมาย วงจรเพิ่มค่า/ลดค่า และในส่วนของวงจรตรรก มีวงจรย่อย คือ วงจรผกผัน วงจรแอนด์ วงจรออร์ วงจรเอ็กซ์คลูซีฟออร์ ซึ่งทั้ง 4 วงจรนี้ได้แบ่งออกเป็นแบบการกระทำระหว่างบิต (Bitwise Operation) และแบบลอจิก (Logical Operation)

- *วงจรบวกเลข แบบเลือกตัวทดล่วงหน้า* [5]

วงจรบวกเลข แบบเลือกตัวทดล่วงหน้า สามารถลดเวลาหน่วงจากการทด เนื่องจากวิธีการคำนวณของวงจรบวกเลขแบบเลือกตัวทด (Carry-Select Adder) คือ จะทำการบวกทีละบิตโดยเริ่มจากบิตต่ำสุดก่อน หลังจากที่บวกบิตแรกแล้วจะได้ผลลัพธ์ของบิตแรก และค่าตัวทด นำค่าตัวทดที่ได้ทำการเลือกคำตอบที่ได้คำนวณไว้แล้วล่วงหน้า โดยในส่วนของการคำนวณคำตอบล่วงหน้านี้ได้ทำการคำนวณโดยจากทุกคำตอบที่เป็นไปได้ทั้งกรณีที่มีตัวทดและไม่มีตัวทด เมื่อได้ค่าตัวทดจากบิตก่อนหน้าก็สามารถทำการเลือกคำตอบได้ โดยไม่เสียเวลาในการคำนวณอีก ทำให้ประหยัดเวลามาก แต่เนื่องจากการออกแบบเป็นการเข้ารหัสข้อมูลแบบหนึ่งในสี่ ทำให้ต้องทำการออกแบบใหม่ คือต้องทำการบวกทีละสองบิตสำหรับเลขฐานสอง เพื่อให้สอดคล้องกับหลักการการเข้ารหัสแบบหนึ่งในสี่ และเมื่อดัดแปลงวงจรเป็นการบวกเลขแบบเลือกตัวทดล่วงหน้าของการเข้ารหัสข้อมูลแบบหนึ่งในสี่ ได้ดังรูปที่ 5

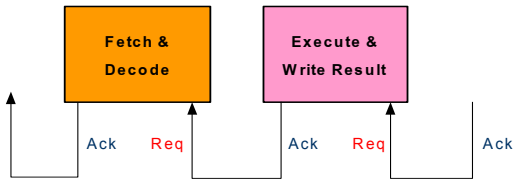


รูปที่ 5วงจรบวกแบบเลือกตัวทด ที่ใช้รหัสข้อมูลแบบหนึ่งในสี่

3.3 รีจิสเตอร์

ส่วนของรีจิสเตอร์ (Register) ใช้สำหรับเก็บข้อมูลชั่วคราวเพื่อนำไปประมวลผล โดยมีขนาด 32 บิต เพื่อรองรับข้อมูลฐาน 2ขนาด 16 บิตที่เข้ารหัสแบบหนึ่งในสี่

โครงสร้างของไปป์ไลน์ผู้ออกแบบได้ออกแบบได้ลดความซับซ้อนของวงจรต่างๆลงเพื่อให้เกิดความรวดเร็วในการทำงานมากขึ้น จึงลดขั้นตอนการทำงานลงเหลือเพียง 2 ขั้นตอนดังรูปที่ 6

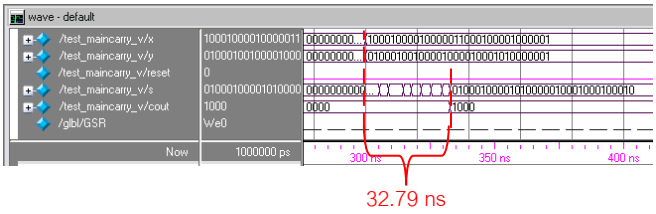


รูปที่ 6 โครงสร้างโดยรวมของไปป์ไลน์

4. ผลการดำเนินการ

จากการออกแบบส่วนดำเนินการทางคณิตศาสตร์และตรรกนั้น จะขอยกตัวอย่างผลการทดลองวงจรบวกเลขแบบเลือกตัวทดล่วงหน้าที่มีการเข้ารหัสข้อมูลแบบหนึ่งในสี่ โดยจะ

ขออธิบายเป็นเลขฐานสิบเพื่อให้ง่ายต่อความเข้าใจ โดยกำหนดให้ ตัวโดยตัวตั้งคือ 39,321 ตัวบวกคือ 13,107 จะได้ผลลัพธ์คือ 52,428 ภายในเวลา 32.79 นาโนวินาที ดังรูปที่ 7



รูปที่ 7 ผลการทดลองวงจรบวกเลขแบบเลือกตัวทดล่วงหน้าของการเข้ารหัสข้อมูลแบบหนึ่งในสี่

จากการออกแบบและผลการทดลองที่ผู้ออกแบบได้ทำไว้ ทำให้สามารถวิเคราะห์ถึงประสิทธิภาพ เรื่องการประหยัดพลังงาน เนื่องจากการออกแบบหน่วยประมวลผลได้ใช้โปรโตคอลการเข้ารหัสข้อมูลแบบหนึ่งในสี่ ซึ่งเมื่อทำการเปรียบเทียบกับ แบบฐานสอง แบบรหัสรางคู่และแบบการเข้ารหัสข้อมูลแบบหนึ่งในสี่ จะเห็นว่าสายสัญญาณแบบหนึ่งในสี่จะเปลี่ยนค่าในสายสัญญาณทีละเส้นต่อข้อมูลจำนวน 2 บิต แต่รหัสรางคู่จะเปลี่ยนค่าในสายสัญญาณทีละ 2 เส้นต่อข้อมูลจำนวน 2บิต ส่งผลให้การโปรโตคอลแบบการเข้ารหัสแบบหนึ่งในสี่ประหยัดพลังงานมากกว่าแบบรหัสรางคู่ และหากรีจิสเตอร์ขนาด 16 บิตต้องมีการเพิ่มค่าที่ละ1 จาก 0 ไปจนครบ 65535 ซึ่งการเปลี่ยนแปลงระดับสัญญาณภายในรีจิสเตอร์จะเป็นสามารถบอกถึงอัตราการใช้พลังงานได้ จากตารางที่ 3แสดงให้เห็นว่าการใช้รหัสหนึ่งในสี่จะมีการเปลี่ยนแปลงระดับสัญญาณน้อยกว่ารหัสรางคู่ถึงสองเท่า ทำให้สามารถเชื่อได้ว่าการใช้รหัสหนึ่งในสี่สามารถช่วยประหยัดการใช้พลังงานในวงจรได้มากกว่าการใช้รหัสรางคู่นั้นเอง

ตารางที่3จนวนการเปลี่ยนแปลงในรีจิสเตอร์ขนาด 16 บิต

รหัส	จำนวนการเปลี่ยนแปลง(ครั้ง)
ฐานสอง Binary	131054
รหัสรางคู่	1048576
รหัสหนึ่งในสี่	524288

5 สรุปและข้อเสนอแนะ

ในการวิจัยนี้ผู้วิจัยได้ทำการออกแบบวงจรประมวลผลที่วงจรคณิตศาสตร์และตรรกและรวมถึงรีจิสเตอร์ต่างๆในหน่วยประมวลผลทั้งหมดเป็นรหัสหนึ่งในสี่แทนรหัสรางคู่เพื่อให้มีการทำงานเป็นวงจรอสมวาร และได้แก้ไขวงจรบวกชนิดคำนวณตัวทดล่วงหน้าโดยปรับไปใช้วงจรประเภทเลือกตัวทด ทำให้วงจรที่ได้มีขนาดเล็กลงในขณะที่ยังคงรักษาประสิทธิภาพของวงจรไว้ได้ และถึงแม้ว่าในทางทฤษฎีการออกแบบวงจรไปป์ไลน์จะสามารถแยกการทำงานของไปป์ไลน์ได้ถึง 5 ขั้น แต่ในทางปฏิบัติพบว่าการแบ่งขั้นตอนการทำงานให้เหลือเพียงสามขั้นก็ยังคงทำให้วงจรมีประสิทธิภาพเท่าเทียบกันเนื่องจากในขั้นตอนการอ่านคำสั่งและการถอดรหัสคำสั่งจะใช้เวลาเพียงเล็กน้อยเนื่องจากหน่วยประมวลผลมีลักษณะเฉพาะ กล่าวคือมีการสร้างหน่วยความจำสังเคราะห์ขึ้นภายใน และไม่ได้ติดต่อกับหน่วยความจำภายนอก จึงแตกต่างจากหน่วยประมวลทั่วไปที่ใช้หน่วยความจำภายนอกที่มีความเร็วในการอ่านข้อมูลต่ำกว่าความเร็วของหน่วยประมวลผล และขั้นตอนที่ใช้เวลามากที่สุดจะอยู่ที่การคำนวณ จึงทำให้สามารถยุบรวมวงจรต่างๆเป็น 3 ขั้นได้ ซึ่งจะทำให้สามารถลดทรัพยากรที่ต้องใช้ลงได้ และเมื่อมีการประมวลผลเป็นแบบไปป์ไลน์แล้วจะทำให้พบปัญหาในเรื่องของการเกิดความผิดพลาดจาก data hazard ทั้งนี้เพราะหน่วยประมวลผลอนุญาตให้ทำงานในลักษณะที่ต่อเนื่องกันถึงสามคำสั่ง ซึ่งจะต้องระมัดระวังการเขียนข้อมูลลงในรีจิสเตอร์ในขณะที่ข้อมูลของรีจิสเตอร์นั้นจะต้องใช้งานในอีกขั้นหนึ่ง แนวทางป้องกันคือโปรแกรมแปลภาษาจะต้องตรวจสอบชุดคำสั่งที่ในลักษณะดังกล่าวและจะต้องแทรกคำสั่งเว้นว่างเข้าไปคั่นกลางระหว่างคำสั่งที่ต่อเนื่องกันซึ่งจะช่วยป้องกันไม่ให้เกิดปัญหา data hazard ขึ้นได้

6.เอกสารอ้างอิง

1. Tony Werner , “Asynchronous Processor Survey” , Research Feature University of California Davis , 1997 IEEE

2. Scott Hauck , “Asynchronous Design Methodologics : An overview ” , Processing of the IEEE , Vol.83 No.1 , pp.69-93 , January 1995
3. Takashi Nanya and other computer developers consider , “A 32 bit Scalable – Delay – Insensitive Microprocessor”, TITAC-2
4. Ran Ginosar, “VLSI Architectures 048878”, 4 phase dual rail protocol , www.ee.technion.ac.il/courses/048878
5. Nève and D. Flandre, “BRANCH-BASED CARRY-SELECT ADDERS”, Microelectronics Laboratory – Université catholique de Louvain – Louvain-la-Neuve - Belgium