การออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์โดยหลักการใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้นบน FPGAs

Design of a Digital Filter Bank Based on Hierarchical Data-path Resource-Sharing on FPGAs

วิวัฒน์ บุญสูง

Wiwat Bunsung

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา

วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering

Prince of Songkla University

2551

ชื่อวิทยานิพนธ์ การออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์โดยหลักการใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้นบน FPGAs

ผู้เขียน นายวิวัฒน์ บุญสูง

สาขาวิชา วิศวกรรมไฟฟ้า

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	**คณะกรรมการสอบ**
.....	..... ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ณัฏฐา จินดาเพ็ชร์)	(ดร.วรรณรัช สันติอมรทัต)
	..... กรรมการ
อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม	(ผู้ช่วยศาสตราจารย์ ดร.ณัฏฐา จินดาเพ็ชร์)
.....	..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.พรชัย พฤกษ์ภัทรานนต์)	(ผู้ช่วยศาสตราจารย์ ดร.พรชัย พฤกษ์ภัทรานนต์)
	..... กรรมการ
	(รองศาสตราจารย์ ดร.วัฒนพงศ์ เกิดทองมี)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

.....

(รองศาสตราจารย์ ดร.เกริกชัย ทองหนู)

คณบดีบัณฑิตวิทยาลัย

ชื่อวิทยานิพนธ์	การออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์โดยหลักการใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้นบน FPGAs
ผู้เขียน	นายวิวัฒน์ บุญสูง
สาขาวิชา	วิศวกรรมไฟฟ้า
ปีการศึกษา	2550

บทคัดย่อ

งานวิจัยนี้นำเสนอระเบียบวิธีการออกแบบวงจรดิจิตอลขนาดใหญ่สำหรับวงจรดิจิตอลฟิลเตอร์แบงก์ภายใต้พื้นที่บน FPGA (Field Programmable Gate Array) ที่มีอยู่อย่างจำกัด การออกแบบแบ่งเป็นสองส่วนคือ ส่วนวงจรข้อมูล (Data-path part) สำหรับการประมวลผลสัญญาณข้อมูล และส่วนวงจรควบคุม (Control part) สำหรับควบคุมจังหวะการทำงานวงจรข้อมูล โดยส่วนวงจรข้อมูลถูกออกแบบโดยอาศัยหลักการใช้ทรัพยากรร่วมกันแบบลำดับชั้นเพื่อลดขนาดของวงจร เริ่มจากเซทของฟังก์ชันที่อธิบายพฤติกรรมของวงจรถูกแปลงให้อยู่ในรูปกราฟกระแสข้อมูล (Data Flow Graph; DFG) ที่มีโครงสร้างเป็นลำดับชั้น จากนั้น DFG ที่เหมือนกันถูกจัดให้ใช้ทรัพยากรร่วมกันเป็นลำดับชั้นจากกลุ่มใหญ่ไปเล็ก โดยมีการคำนึงถึงเวลาที่ช้าลงหลังจากมีการใช้ทรัพยากรร่วมกัน ในส่วนวงจรควบคุม FSM (Finite State Machine) แบบ Moore machine ถูกใช้สำหรับควบคุมจังหวะการใช้ทรัพยากรร่วมกันของส่วนวงจรข้อมูลให้เป็นไปอย่างถูกต้อง วงจรที่ได้ออกแบบถูกทดสอบบนชิพ FPGA ของบริษัท Xilinx ตระกูล SPARTAN-3 เบอร์ XCS4000-5FG676 จากการจำลองแบบการทำงานวงจรสามารถทำงานได้ถูกต้องเมื่อเทียบกับผลการคำนวณจากโปรแกรม MATLAB และจากผลการสังเคราะห์วงจรที่ออกแบบใช้พื้นที่ 24285 LUTs ซึ่งมีขนาดลดลง 44% และมีรอบการทำงาน 1300 นาโนวินาที ซึ่งมีความเร็วลดลง 30% เมื่อเทียบกับวงจรที่ไม่มีการใช้ทรัพยากรร่วมกัน

Thesis Title	Design of Digital filter Bank Based on Hierarchical Data-path Resource-Sharing on FPGAs
Author	Mr.Wiwat Bunsung
Major Program	Electrical Engineering
Academic Year	2007

ABSTRACT

This research presents a large digital circuits design methodology for a digital filter bank circuit on an area limited FPGA (Field Programmable Gate Array). The design is divided into two parts: a data-path part for data processing and a control part for controlling data-path operations. The data-path part is designed by using a hierarchical resource-sharing technique to reduce the circuit size. The set of the functions describing the circuit behavior is translated into hierarchical data flow graphs (DFGs). Then the DFGs are hierarchically grouped the same structure DFGs to share the same resources from large to small groups. The slower operation of the circuit after resource-sharing is concerned. In the control part, the FSM (Finite State Machine) in Moore machine format is used for controlling the correct sequence of the resource-sharing in the data-path part. The designed circuit was tested on a Xilinx FPGA SPARTAN-3 XCS4000-5FG676. The circuit work correctly compared to the calculation results from MATLAB. From the synthesis results, the circuit size is 24285 LUTs, i.e. 44% better, and the cycle time is 1300 nanoseconds, i.e. 30% worse, compared to the original circuit without resource-sharing.

กิตติกรรมประกาศ

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.ณัฏฐา จินดาเพ็ชร์ ประธานกรรมการที่ปรึกษางานวิจัย ที่ได้เสียสละเวลาในการให้คำปรึกษา แนวคิดในการทำวิจัย รวมถึงการช่วยเหลือแก้ไขปัญหาที่เกี่ยวกับการวิจัย ตลอดจนตรวจสอบและแก้ไขวิทยานิพนธ์ให้ดำเนินไปอย่างลุล่วงสมบูรณ์

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.พรชัย พฤกษ์ภัทรานนต์ ที่ได้กรุณาให้คำปรึกษา คำแนะนำ และให้ความช่วยเหลือในงานวิจัย ตลอดจนช่วยตรวจทานแก้ไขวิทยานิพนธ์ให้ดำเนินไปด้วยดี

ขอขอบพระคุณ ดร.วรรณรัช สันติอมรทัต ที่ได้กรุณาเสียสละเวลาเป็นประธานกรรมการสอบวิทยานิพนธ์และตรวจทานแก้ไขวิทยานิพนธ์ให้มีความสมบูรณ์

ขอขอบพระคุณ รองศาสตราจารย์ ดร.วัฒนพงศ์ เกิดทองมี ที่กรุณาเสียสละเวลาเป็นกรรมการสอบวิทยานิพนธ์ อีกทั้งตรวจทานและแก้ไขวิทยานิพนธ์ให้มีความสมบูรณ์ยิ่งขึ้น

ขอขอบพระคุณ บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตหาดใหญ่ที่ให้การสนับสนุนทุนในการทำวิจัยและให้ความช่วยเหลือด้านการประสานงานต่างๆ

ขอขอบพระคุณ มูลนิธิเพื่อการศึกษาคอมพิวเตอร์และการสื่อสาร ที่กรุณาให้ทุนแก่ข้าพเจ้า

ขอขอบพระคุณ คณาจารย์ บุคลากร และนักศึกษาปริญญาโทภาควิชาวิศวกรรมไฟฟ้าทุกคนที่ได้ให้คำปรึกษา และกำลังใจในการทำงานเป็นอย่างดีเสมอมา

และสุดท้าย ข้าพเจ้าขอน้อมรำลึกถึงพระคุณของ บิดามารดา และครอบครัว ที่ส่งเสริมและสนับสนุนข้าพเจ้าในทุกๆเรื่องตลอดมาจนสำเร็จการศึกษา

วิวัฒน์ บุญสูง

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

รายการตาราง

รายการภาพประกอบ

รายการภาพประกอบ (ต่อ)

รายการภาพประกอบ (ต่อ)

รายการภาพประกอบ (ต่อ)

บทที่ 1

บทนำ

1.1 ความสำคัญและที่มาของหัวข้อวิจัย

ในอดีตการศึกษาวงจรดิจิตอลทางภาคปฏิบัติ ได้นำลอจิกเกต (logic gate) ของวงจรรวม หรือที่เรียกกันว่า "ไอซี (IntegratedCircuit: IC)" ซึ่งมีการกำหนดประเภทของเกตมาแล้วโดยผู้ผลิต เช่นเป็น AND, OR, NAND หรือ NOR มาเชื่อมต่อกันลงบนแผ่นทดลอง เช่น Proto-Board หรือ Logic trainer แล้วทำการเชื่อมต่อกับสวิตช์หรือตัวแสดงผลอื่นๆ เข้ากับลอจิกเกตบนแผ่นวงจร สุดท้ายก็ต้องวิเคราะห์วงจรที่ได้จากการออกแบบว่าถูกต้องตามทฤษฎีที่ได้ศึกษามาด้วยการทดลอง การศึกษาวิชาออกแบบวงจรดิจิตอลด้วยวิธีดังกล่าวเป็นไปได้อย่างช้าๆ เพราะต้องเสียเวลาในการเชื่อมต่อวงจรจริง อีกทั้งต้องใช้อุปกรณ์หรือลอจิกเกตอื่นๆอีกเป็นจำนวนมาก และผลที่ได้ก็อาจจะเป็นวงจรที่มีขนาดใหญ่เกินไป หรือมีราคาแพงเกินไป นอกจากนี้ผลการทำงานของวงจรที่ได้ออกแบบมา เรายังไม่สามารถคาดการณ์ได้ว่าถูกหรือผิด จนกว่าจะต่อวงจรจนแล้วเสร็จ

การศึกษาและการออกแบบวงจรดิจิตอลในปัจจุบัน ได้มีนำวงจรรวมดิจิตอลชนิดหนึ่งที่เรียกว่าเอฟพีจีเอ (Field Programmable Gate Arrays ; FPGA) มาเป็นองค์ประกอบในการเรียนรู้ ทั้งนี้ เนื่องจากวงจรทั้งหมดที่ต้องการออกแบบจะถูกออกแบบบนโปรแกรมคอมพิวเตอร์แล้วสามารถจำลองการทำงาน เพื่อวิเคราะห์ผลที่ได้ออกแบบ เมื่อถูกต้องแล้วก็ทำการโปรแกรมวงจรที่ได้ออกแบบไว้ลงบนชิพลงในชิพเพียงตัวเดียว ที่สามารถใช้ทำงานจริงได้ทันที จึงเหมาะกับการออกแบบวงจรดิจิตอลที่มีขนาดใหญ่

จากคุณสมบัติของชิพวงจรรวม FPGA พบว่าเหมาะสมที่จะนำมาใช้ในการศึกษาและออกแบบวงจรดิจิตอลฟิลเตอร์ที่เป็นวงจรรวมที่มีขนาดใหญ่ แต่ในการออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์ซึ่งเป็นจงจรดิจิตอลที่มีขนาดใหญ่นั้น หากนำหลักการใช้ทรัพยากรร่วมกัน (Resource-sharing) ระหว่างโอเปอเรชันต่าง ๆ ซึ่งเป็นขั้นตอนหนึ่งในสังเคราะห์วงจรที่ระดับสูงที่ทำให้วงจรขนาดใหญ่สามารถถูกสร้างได้บนชิพที่มีพื้นที่จำกัดมาใช้ในการออกแบบ จะมีประโยชน์

มากต่อการออกแบบ แต่หลังจากการใช้ทรัพยากรร่วมกันวงจรจะทำงานช้าลงและหากยิ่งมีการใช้ทรัพยากรร่วมกันมากขึ้นเท่าไหร่ วงจรก็จะทำงานช้าลงมากยิ่งขึ้นเช่นกัน นอกจากนี้ยังส่งผลในการต่อสายสัญญาณ (Interconnection) ภายในชิพที่จะซับซ้อนยิ่งขึ้นด้วย มีงานวิจัยหลายงานที่นำเสนอระเบียบวิธี Resource-sharing ที่มีประสิทธิภาพ แต่วิธีการเหล่านี้ไม่ได้คำนึงถึงความซับซ้อนการต่อสายสัญญาณภายในชิพไปพร้อม ๆ กับการทำ Resource-sharing แบบลำดับชั้น

1.2 การทบทวนเอกสารที่เกี่ยวข้อง

1.2.1 A digital filterbank hearing aid-design, implementation and evaluation (Thomas Lunner, Johan Hellgren, 1991) งานวิจัยนี้กล่าวถึงการนำดิจิตอลฟิลเตอร์แบงค์ไปใช้ในการออกแบบเครื่องช่วยฟังโดยที่งานวิจัยนี้จะเลือกออกแบบวงจรฟิลเตอร์แบงค์เป็นแบบ IFIR ซึ่งจะช่วยในการลดจำนวนตัวคูณในวงจรโดยฟิลเตอร์แบงค์ที่นำมาออกแบบจะมีจำนวน 8 แถบช่องสัญญาณ งานวิจัยจะแสดงให้เห็นโครงสร้างของวงจรฟิลเตอร์ตั้งแต่การบรรยายพฤติกรรมของวงจร การหาสัมประสิทธิ์ตัวคูณ ลักษณะสัญญาณที่ได้จากฟิลเตอร์แต่ละตัว ซึ่งงานวิจัยนี้จะตรวจสอบการทำงานด้วย DSP ชิพรุ่น TEXAS INSTRUMENTS TMS320E25

1.2.2 A scheduling algorithm for conditional resource sharing (Taewhan Kim, Noritake Yonezawa and Jane W.S. Liu,1994) บทความนี้จะนำเสนอขั้นตอนใหม่ในการแก้ปัญหาสำหรับการออกแบบ dataflow graphs ที่มีส่วนของการคำนวณผลโดยมีเงื่อนไข ในการออกแบบจะพิจารณาลำดับการทำงานของวงจรรวมทั้งหมดเพื่อพิจารณาและเปลี่ยนแปลง dataflow graphs ให้มีลำดับการคำนวณใหม่โดยนำส่วนการคำนวณผลที่ไม่มีเงื่อนไขมาคำนวณพร้อมกับส่วนการคำนวณผลที่มีเงื่อนไข ซึ่งจะได้ประโยชน์อย่างมากเมื่อมีส่วนการคำนวณผลโดยมีเงื่อนไขอยู่เป็นจำนวนมากใน dataflow graphs รวมและในการทดสอบพบว่าจะต้องมีการคำนึงถึงการทำงานแบบ interconnection ท่ามกลางการทำงานที่หลากหลายในส่วนการคำนวณผลอื่นด้วย

1.2.3 An Optimal Clock Period Selection Method Based on Slack Minimization Criteria (EN-Shou Chang, Daniel D. Gajski and Sanjiv Narayan, 1996) งานวิจัยนี้จะกล่าวถึงส่วนสำคัญในการสังเคราะห์ฮาร์ดแวร์เพื่อใช้อธิบายพฤติกรรมของวงจรนั้นคือการเลือกคาบสัญญาณนาฬิกาให้แก่วงจร ซึ่งการตัดสินใจเพื่อเลือกคาบสัญนาฬิกานี้มีแนวทางอย่างใดอย่างหนึ่งระหว่างเลือกคาบสัญญาณนาฬิกาที่แน่นอนหรือการพิจารณาเวลาหน่วงแล้วใช้จำนวนคาบสัญญาณนาฬิกาที่มากขึ้นซึ่งหากเลือกแนวทางที่ไม่เหมาะสมในการกำหนดคาบสัญญาณนาฬิกาแล้วนั้นจะก่อให้เกิดการเสียเวลาโดยเปล่าประโยชน์เนื่องจากกำหนดคาบสัญญาณที่ใหญ่เกินไป ดังนั้นใน

งานวิจัยนี้จะนำเสนอวิธีที่ใช้วิเคราะห์เพื่อกำหนดคาบสัญญาณนาฬิกาที่จะทำให้สูญเสียเวลาที่เหลือจากการทำงานให้น้อยที่สุดและสามารถนำแนวทางนี้ไปใช้ในการออกแบบแก่วงจรอื่นทั่วไปซึ่งจะก่อให้เกิดประสิทธิภาพที่ดีต่อการออกแบบ

1.2.4 การออกแบบหน่วยประมวลผลคณิตศาสตร์ความเร็วสูงสำหรับวงจรกรองปรับตัวบน FPGA (วุฒิ วิริยะสม, 2550) งานวิจัยนี้นำเสนอเทคนิคสำหรับการออกแบบวงจรการประมวลผลจำนวนทศนิยมให้มีความเร็วสูง รวมไปถึงการปรับปรุงวงจรอัลกอริทึมของวงจรหรือเทคนิคในการเขียนภาษา VHDL เพื่อให้วงจรที่ออกแบบมีการใช้ทรัพยากรบน FPGA ที่น้อยที่สุดเท่าที่จะสามารถทำได้ โดยจะมีการทดสอบวงจรประมวลผลทศนิยมที่ออกแบบด้วยการนำไปประยุกต์ใช้ในการออกแบบและสร้างวงจรกรองปรับตัวแบบโครงข่ายประสาท ADALINE Adaptive Filter ชนิดที่ไม่ใช้สัญญาณอ้างอิงจากภายนอกเพื่อแสดงประสิทธิภาพของวงจรประมวลจำนวนทศนิยมที่นำเสนอ พบว่าผลจากการทดสอบเป็นที่น่าพอใจเพราะวงจรสามารถประมวลผลสัญญาณตัวอย่างได้ถึง 1.25 ล้านตัวอย่างต่อวินาที (Data-samples per second) ซึ่งมีค่าสูงกว่ามากเมื่อนำไปเปรียบเทียบกับผลการทำงานที่ได้จากไมโครคอนโทรลเลอร์จากงานวิจัยที่ผ่านมา

1.3 วัตถุประสงค์

1.3.1 ศึกษาและออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์โดยระเบียบวิธี Resource-sharing แบบลำดับชั้นและออกแบบวงจรควบคุมการทำงานของวงจรดิจิตอลฟิลเตอร์แบงค์

1.3.2 ศึกษาและจำลองการทำงานวงจรดิจิตอลฟิลเตอร์แบงค์บน MATLAB

1.3.3 ศึกษาและจำลองการทำงานวงจรดิจิตอลฟิลเตอร์แบงค์บน FPGA

1.3.4 วิเคราะห์และเปรียบเทียบผลการทำงานวงจรดิจิตอลฟิลเตอร์แบงค์ที่ออกแบบด้วยโปรแกรม MATLAB กับวงจรดิจิตอลฟิลเตอร์แบงค์ที่ออกแบบด้วยโปรแกรม Xilinx

1.4 ขอบเขตของการวิจัย

ศึกษาและออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์พร้อมทั้งปรับปรุงวงจรที่ออกแบบให้มีขนาดเล็กลงโดยอาศัยเทคนิคการใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้น (Hierarchical Data-path Resource-sharing) เพื่อเป็นการใช้เนื้อที่บน FPGA ให้น้อยที่สุดเท่าที่จะสามารถทำได้ และวิเคราะห์ผลการจำลองการทำงานของวงจรที่ออกแบบบน FPGA กับการทำงานของวงจรที่ออกแบบด้วยโปรแกรม MATLAB

1.5 ขั้นตอนและวิธีการดำเนินการวิจัย

1.5.1 ศึกษาการทำงานของวงจรดิจิตอลฟิลเตอร์แบงค์

1.5.2 ออกแบบและสังเคราะห์วงจรดิจิตอลฟิลเตอร์แบงค์ จากหลักการที่ได้ศึกษาและค้นคว้าผ่านมา โดยใช้โปรแกรม Xilinx ISE 8.1i

1.5.3 จำลองผลการทำงานของวงจรดิจิตอลฟิลเตอร์แบงค์เพื่อให้ได้การทำงานของวงจรถูกต้อง

1.5.4 ศึกษาและออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์ด้วยเทคนิคการใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้น (Hierarchical Data-path Resource-sharing)

1.5.5 ศึกษาและออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์โดยใช้โปรแกรม MATLAB

1.5.6 เปรียบเทียบผลของวงจรที่จำลองการทำงานบน FPGA กับผลที่ได้รับจากการออกแบบและคำนวณโดยใช้โปรแกรม MATLAB

1.5.7 สรุปผลและเขียนรายงาน

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1 เข้าใจการทำงานและสามารถออกแบบวงจรดิจิตอลฟิลเตอร์แบงค์โดยใช้ระเบียบวิธี การใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้น (Hierarchical Data-path Resource-sharing)

1.6.2 วงจรที่ออกแบบด้วยการใช้ทรัพยากรร่วมกันสามารถทำงานและได้ผลการทำงานที่เหมือนกับวงจรที่ไม่มีการใช้ทรัพยากรร่วมกันในการออกแบบ

1.6.3 ได้เรียนรู้เทคนิคในการออกแบบวงจรที่เหมาะสมที่สุด เมื่อมีข้อจำกัดด้านเนื้อที่ที่จำกัดในการออกแบบ (Design Constraints)

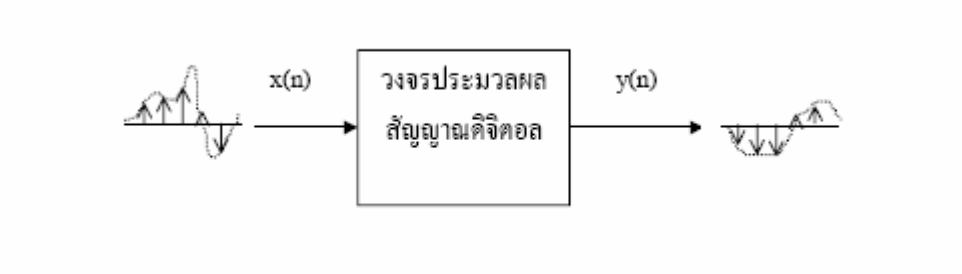
บทที่ 2

ทฤษฎีและหลักการ

2.1 ทฤษฎีเกี่ยวกับประมวลผลสัญญาณดิจิตอลและวงจรกรองความถี่ดิจิตอล (Digital Filter)

2.1.1 การประมวลผลสัญญาณดิจิตอล (Digital Signal Processing)

ทฤษฎีของการประมวลผลสัญญาณดิจิตอล (Digital Signal Processing) หากศึกษาและพิจารณาความหมายของทฤษฎี อาจกล่าวได้ว่าหมายถึงการประมวลผลของสัญญาณไม่ต่อเนื่อง (Discrete-time Signal Processing) ทั้งนี้เพราะว่า ทฤษฎีของการประมวลผลสัญญาณเป็นการกระทำโดยมองสัญญาณขาเข้าเป็นลักษณะของลำดับของข้อมูล (ซึ่งคือสัญญาณไม่ต่อเนื่อง) โดยนำข้อมูลเหล่านี้มาประมวลผล เช่น บวก ลบ คูณ หาร เพื่อหาสัญญาณขาออกในลักษณะเป็นลำดับข้อมูลเช่นเดียวกัน



ภาพประกอบ 2-1 ระบบประมวลผลสัญญาณดิจิตอล

ข้อกำหนดที่สำคัญในการประมวลผลสัญญาณคือ การเลือกใช้ตัวประมวลผลสัญญาณ นั่นคือ การที่ต้องมีตัวประมวลผลที่เร็วพอที่จะประมวลผลสัญญาณให้ทันได้โดยเฉพาะอย่างยิ่ง ถ้าสัญญาณที่ต้องการประมวลผลมีอัตราการสุ่มที่สูง หรืออัลกอริธึมที่ใช้มีความซับซ้อนในการคำนวณมาก ก็จำเป็นที่จะต้องใช้ตัวประมวลผลที่มีความเร็วสูงมากยิ่งขึ้น มีทางเลือกใหญ่ ๆ อยู่ 3 ทางในการทำตัวประมวลผล คือ

1) การเขียนซอฟท์แวร์เพื่อใช้กับคอมพิวเตอร์ หรือใช้กับชิพไมโครโปรเซสเซอร์ทั่ว ๆ ไป ซึ่งถึงแม้ว่าคอมพิวเตอร์หรือไมโครโปรเซสเซอร์จะไม่ได้ออกแบบมาเฉพาะสำหรับการ

ประมวลผลสัญญาณ แต่ก็สามารถนำมันมาใช้ได้ในงานที่ต้องการอัตราการประมวลผลไม่มากนัก หรือในการประมวลผลแบบไม่เป็นเวลาจริง อย่างไรก็ตาม ปัจจุบันคอมพิวเตอร์ส่วนบุคคลมีความเร็วสูงมากจนสามารถนำมาใช้ทำการประมวลผลแบบเวลาจริงหลาย ๆ อย่างได้ ตัวอย่างที่เห็นได้ชัด เช่น การถอดรหัสของสัญญาณเสียง หรือวีดีโอที่ถูกบีบอัดข้อมูลมาด้วยมาตรฐาน MPEG ซึ่งแต่ก่อนต้องใช้ฮาร์ดแวร์พิเศษในการถอดรหัส แต่ปัจจุบันใช้เพียงซอฟท์แวร์ก็สามารถทำได้แล้วโดยอาศัย CPU ที่มีความเร็วสูงขึ้น

2) การใช้ซอฟท์แวร์ร่วมกับชิพ DSP ชิพ DSP เป็นชื่อย่อของชิพประมวลผลสัญญาณ(Digital Signal Processor) ซึ่งคือ ไมโครโปรเซสเซอร์ที่ถูกออกแบบมาสำหรับงานประมวลผลสัญญาณแบบเวลาจริงโดยเฉพาะ โดยไมโครโปรเซสเซอร์ประเภทนี้จะมีสถาปัตยกรรมที่เอื้ออำนวยต่อการคำนวณ และการโอนถ่ายข้อมูลที่มีประสิทธิภาพ และความเร็วสูง เช่น การมีคำสั่งพิเศษในการคูณ, การบวกสะสม, หรือการอ้างข้อมูลแบบ circular buffer เป็นต้น บางชนิดยังสามารถทำการประมวลผลหลาย ๆ ส่วนได้พร้อมกันในตัวเดียว (multi-processing) อีกด้วย บริษัทที่เป็นผู้นำด้านการผลิตชิพ DSP ได้แก่ Texas Instruments, Motorola, Analog Devices, และ AT&T เป็นต้น ซึ่งชิพ DSPนี้มีทั้งประเภทที่เป็นการประมวลผลข้อมูลแบบจำนวนเต็ม (fixed-point) และประเภทที่ประมวลผลข้อมูลแบบเลขอิงดรรชนี (floating-point)

3) การใช้ฮาร์ดแวร์หรือไอซีที่ออกแบบเฉพาะงาน ฮาร์ดแวร์ที่นี้ก็หมายถึง วงจรดิจิตอล ซึ่งสามารถออกแบบให้ทำการประมวลผลข้อมูลได้เช่นเดียวกัน อัลกอริธึมที่เป็นที่นิยม เช่น FFT (Fast Fourier Transform) หรือ ตัวกรองดิจิตอลนั้น สามารถหาซื้อได้ทั่วไปเป็นไอซีสำเร็จรูปที่ทำเฉพาะฟังก์ชันนั้น ๆ แต่ถ้าต้องการอัลกอริธึมที่เฉพาะมากขึ้น ก็อาจต้องทำการออกแบบเป็นไอซีเฉพาะงานเอง (Application Specific Integrated Circuits หรือ ASIC) ซึ่งแน่นอนว่าต้นทุนในการออกแบบสำหรับทางเลือกนี้ค่อนข้างสูง

ทางเลือกอีกทางหนึ่ง คือ การใช้ไอซีดิจิตอลประเภทโปรแกรมได้ หรือ FPGA (Field Programmable Gate Array) ซึ่งปัจจุบันมีความจุมากพอที่จะนำมาใช้ทำการประมวลผลสัญญาณได้ การใช้ FPGA จะมีต้นทุนในการออกแบบที่ถูกกว่า ASIC ทั้งนี้การเลือกใช้ตัวประมวลผลแต่ละแบบก็ขึ้นอยู่กับลักษณะของงาน ความเร็วที่ต้องการ และต้นทุน

ปัจจุบันมีงานหลายอย่างที่ได้นำเอาการประมวลผลสัญญาณไปใช้งาน คงจะยกตัวอย่างได้เพียงแค่ส่วนหนึ่งของเท่านั้นซึ่งได้แก่

- การประมวลผลเสียง เช่น การบีบอัดเสียง หรือเข้ารหัสเสียง (speech coding) การรู้จำเสียง(speech recognition) การเติมเอฟเฟคเสียง (sound

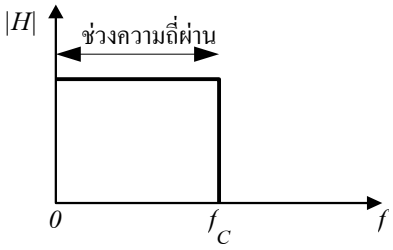
effect) การผสมเสียง การกรองเสียงรบกวน การสังเคราะห์เสียงดนตรี (music synthesizer) เป็นต้น

- วงการแพทย์ ได้แก่ การวิเคราะห์สัญญาณคลื่นสมอง (EEG) และสัญญาณคลื่นหัวใจ(ECG) เครื่องช่วยได้ยิน (hearing aid) เป็นต้น
- ในระบบการจ่ายกำลังไฟฟ้าซึ่งใช้ในการลดปริมาณของฮาร์มอนิกส์ที่เกิดขึ้น
- การประมวลผลสัญญาณแบบหลายมิติ ได้แก่ การประมวลผลภาพนิ่ง (2 มิติ) วีดีโอ (3มิติ) holography (ภาพ 3 มิติ) ตัวอย่างของการประยุกต์ใช้งาน ได้แก่ การบีบอัดสัญญาณวีดีโอ การทำภาพให้ชัดขึ้น เช่นใช้กับภาพถ่ายดาวเทียม ภาพทางโบราณคดีและภาพที่ถ่ายแล้วไม่ชัด ระบบรู้จำภาพ การมองเห็นของหุ่นยนต์ และการเคลื่อนไหวของภาพสามมิติ เป็นต้น
- ระบบสื่อสาร ได้แก่ modulation/demodulation การชดเชยผลของช่องสัญญาณ(channel equalizer) ในอุปกรณ์โมเดม และโทรศัพท์มือถือ การกรองเสียงสะท้อนในระบบโทรศัพท์ทางไกล และระบบการประชุมทางไกล (video conferencing) ระบบเรดาร์และโซนาร์ ระบบนำทาง (navigation system) GPS เป็นต้น
- ในอุปกรณ์ และเครื่องมือทางไฟฟ้า เช่น เครื่องวิเคราะห์ความถี่ (spectrum analyzer) เครื่องสร้างสัญญาณ (function generator) และเครื่องตรวจตัวสัญญาณ (pattern matching) เป็นต้น

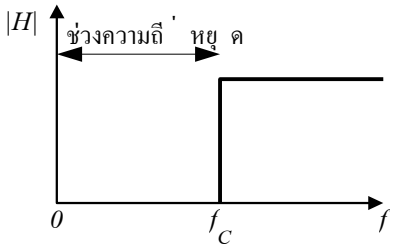
กล่าวได้ว่าในปัจจุบันการประมวลผลสัญญาณดิจิตอล ได้ปฏิวัติเทคโนโลยี ต่าง ๆ ให้ก้าวหน้า และมี ประสิทธิภาพขึ้นอย่างมาก

2.1.2 วงจรกรองความถี่ดิจิตอล (Digital Filter)

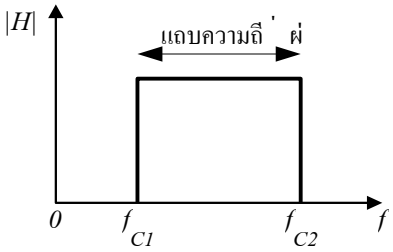
วงจรกรองความถี่ทำหน้าที่จำแนกความถี่ตามความต้องการของผู้ใช้ แบ่งตามคุณลักษณะของผลตอบสนองความถี่(frequency response) ดังภาพประกอบที่ 2-2 ได้ 4 ชนิด ด้วยกันคือ วงจรกรองความถี่ต่ำผ่าน(low-pass filter ; LPF) วงจรกรองความถี่สูงผ่าน(high-pass filter ; HPF) วงจรกรองแถบความถี่ผ่าน(band-pass filter ; BPF) และวงจรกรองแถบความถี่หยุดผ่าน(band-stop filter ; BSF)



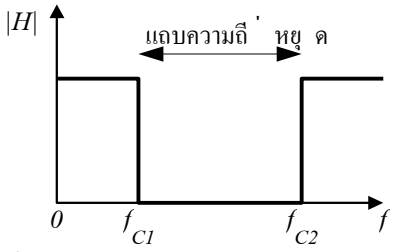
(ก) ผลตอบสนองของวงจรกรองความถี่ต่ำ



(ข) ผลตอบสนองของวงจรกรองความถี่สูง



(ค) ผลตอบสนองของวงจรกรองแถบความถี่ผ่าน



(ง) ผลตอบสนองของวงจรกรองแถบความถี่หยุด

ภาพประกอบ 2-2 ผลตอบสนองความถี่ของวงจรกรองอุดมคติ

จากผลตอบสนองความถี่ในภาพประกอบ 2-2 เมื่อให้ |H| คือขนาดของแรงดันทางด้านเอาต์พุต วงจรกรองความถี่ต่ำผ่าน จะยอมให้ความถี่ตั้งแต่ 0 Hz ถึงความถี่ f_c ผ่านไปยังขั้วเอาต์พุตของวงจรได้ ส่วนความถี่ที่สูงกว่า f_c ความถี่จะไม่ผ่านไปยังขั้วเอาต์พุตของวงจร สำหรับวงจรกรองความถี่สูงผ่านจะยอมให้ความถี่สูงกว่าความถี่ f_c ผ่านไปยังขั้วเอาต์พุตของวงจรได้ ส่วนความถี่ตั้งแต่ 0Hz ถึงความถี่ f_c จะไม่ผ่านไปยังขั้วเอาต์พุตของวงจร สำหรับวงจรกรองแถบความถี่ผ่าน จะยอมให้ความถี่ตั้งแต่ f_{c1} ถึงความถี่ f_{c2} ผ่านไปยังขั้วเอาต์พุตของวงจร ส่วนความถี่ตั้งแต่ 0Hz ถึงความถี่ f_{c1} กับความถี่ที่สูงกว่า f_{c2} จะไม่ผ่านไปยังขั้วเอาต์พุตของวงจร และวงจรกรองแถบความถี่หยุดผ่าน จะไม่ยอมให้ช่วงความถี่ f_{c1} ถึงความถี่ f_{c2} ผ่านไปยังขั้วเอาต์พุตของวงจร ส่วนความถี่อื่นๆ วงจรยอมให้ผ่านไปยังขั้วเอาต์พุตได้

วงจรกรองความถี่ที่ใช้งานกันอยู่ทั่วไป มักนิยมใช้วงจรกรองความถี่แบบแอนาลอก ซึ่งประกอบไปด้วย ตัวความต้านทาน ตัวเก็บประจุ ตัวเหนี่ยวนำ และอุปกรณ์กึ่งตัวนำเช่น ออปแอมป์ ข้อดีคือออกแบบได้ง่าย ราคาถูก แต่มีข้อเสียที่วงจรขาดเสถียรภาพ (stability) ความถี่ที่ต้องการมีความคลาดเคลื่อนสูง แต่ในปัจจุบันได้หันมานิยมใช้วงจรกรองความถี่แบบดิจิตอลกันมาก เพราะมีเสถียรภาพที่ดีกว่า ความถี่ที่ต้องการมีความคลาดเคลื่อนน้อยกว่า แต่มีข้อเสียคือ

การออกแบบทำได้ยากกว่า และมีราคาสูงกว่า สำหรับงานวิจัยนี้ขอกล่าวถึงการออกแบบวงจรกรองความถี่แบบดิจิตอล เพียงอย่างเดียวเท่านั้น

วงจรกรองดิจิตอลเป็นวงจรที่ทำหน้าที่ประมวลผลสัญญาณในโดเมนเวลาเพื่อดัดแปลงผลตอบสนองทางความถี่ในเชิงขนาด (Magnitude response) หรือผลตอบสนองเชิงเฟส (Phase response) โดยสามารถยกตัวอย่างประโยชน์ที่ได้รับจากวงจรกรองดิจิตอล ได้แก่ เพื่อกรองสัญญาณรบกวน ออกจากสัญญาณที่ต้องการ เช่นสัญญาณเสียงที่มีการรบกวนจากสภาวะรอบข้าง หรือ แยกสัญญาณที่ปะปนกัน เช่นในการวัดสัญญาณคลื่นไฟฟ้าหัวใจ (Electrocardiogram ; ECG) ของแม่และทารกในครรภ์ ที่จะมีการปะปนของสองสัญญาณ หรือ ปรับปรุงคุณภาพของเสียงเช่นในเครื่องช่วยฟัง เป็นต้น

โดยที่วงจรกรองอาจถูกแสดงอยู่ในรูปของสมการผลต่าง (Difference Equation) ซึ่งสมการผลต่างในที่นี้ คือ สมการที่บ่งบอกความสัมพันธ์ในเชิงเวลาระหว่างสัญญาณขาเข้าของวงจรกรอง คือ x(n) และสัญญาณขาออกของวงจรกรอง คือ y(n) สมการผลต่าง คือ สิ่งที่จะใช้สำหรับสร้างตัวประมวลผล เพราะสมการจะบอกว่าจะหา y(n) ได้ด้วยการคำนวณอย่างไร แสดงด้วยสมการ2-1 ดังนี้

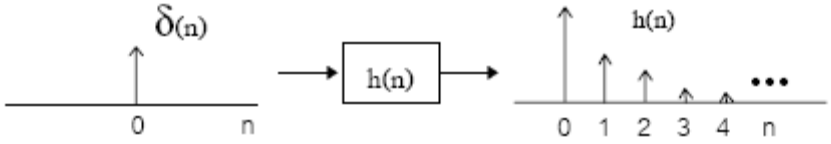
$$y(n) = a0x(n) + a1x(n-1) + a2x(n-2) + ... - b1y(n-1) - b2y(n-2) - ... \qquad (2\text{-}1)$$

จัดรูปแบบใหม่ได้สมการ (2-2)

$$y(n) = \sum_{i=0}^{\infty} a_i x(n-i) - \sum_{i=0}^{\infty} b_i y(n-i) \qquad (2\text{-}2)$$

โดยทั่วไป สำหรับวงจรกรอง FIR จะมีค่าสัมประสิทธิ์ bi เป็นศูนย์ทั้งหมด หรือไม่มีการใช้ค่าผลตอบในอดีตนั่นเอง แต่สำหรับวงจรกรอง IIR จะมีค่า bi อย่างน้อย 1 ตัวไม่เท่ากับศูนย์

จากสมการผลต่างข้างต้นทำให้สามารถระบุคุณลักษณะของวงจรกรองได้โดยสมบูรณ์ด้วยผลตอบสนองต่อสัญญาณอิมพัลส์ (Impulse Response) ซึ่งจะใช้สัญลักษณ์แทนว่า h(n) โดยที่ผลตอบสนองต่อสัญญาณอิมพัลส์ของวงจรกรองเป็นตัวกำหนดคุณสมบัติของวงจรกรอง การหาคุณสมบัติของวงจรกรองนั้นสามารถหาได้โดยการป้อนสัญญาณเข้าเป็นแบบอิมพัลส์ ($\delta(n)$) และเมื่อนำสัญญาณนี้ป้อนที่จุดสัญญาณเข้าของวงจรจะได้ออกมาเป็นผลตอบสนองต่อสัญญาณอิมพัลส์ ดังแสดงด้วยภาพประกอบ 2-3

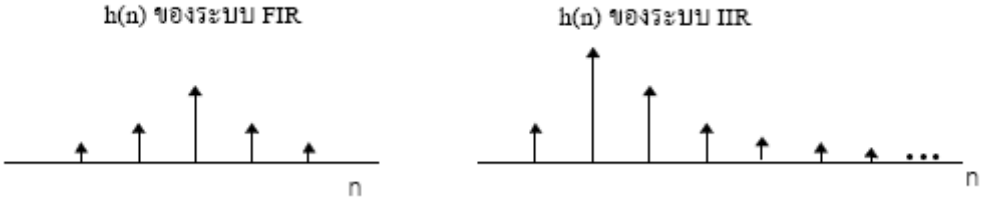


ภาพประกอบ 2-3 ผลตอบสนองต่อสัญญาณอิมพัลส์

ซึ่งลำดับสัญญาณขาออกจากการป้อนสัญญาณอิมพัลส์เข้าสู่ระบบ สามารถเกิดขึ้นได้ 2 กรณี ดังนั้นจึงสามารถแบ่งวงจรกรองตามผลตอบสนองอิมพัลส์ ออกเป็น 2 แบบ คือ

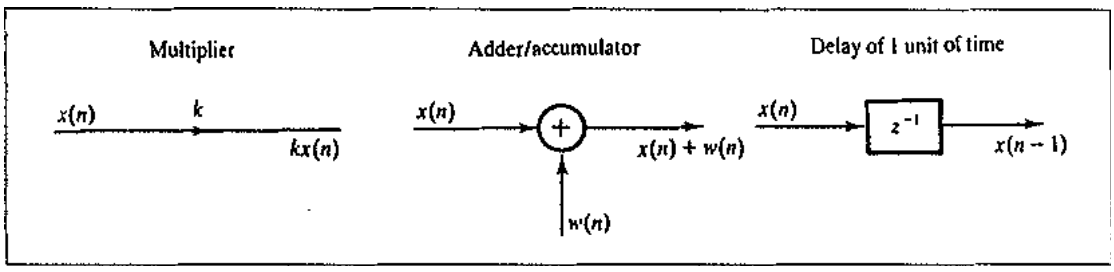
วงจรกรองที่มีผลตอบสนองอิมพัลส์จำกัด (Finite Impulse Response : FIR)

วงจรกรองที่มีผลตอบสนองอิมพัลส์อนันต์ (Infinite Impulse Response : IIR)



ภาพประกอบ 2-4 ตัวอย่างผลตอบสนองต่อสัญญาณอิมพัลส์ของวงจรแบบ FIR และ IIR

วงจรกรองดิจิตอลทั้งสองชนิดจะประกอบด้วยองค์ประกอบพื้นฐาน(Basic elements) คือ ตัวบวก (adder) ตัวคูณ (multiplier) และตัวหน่วง (Delay)



ภาพประกอบ 2-5 องค์ประกอบพื้นฐานทั้ง 3 ตัวที่ใช้เป็นส่วนประกอบวงจรกรองดิจิตอล

วงจรกรองอาจถูกแสดงอยู่ในรูปของฟังก์ชันถ่ายโอน (Transfer function) โดยหากว่าวงจรใดวงจรหนึ่งมีผลตอบสนองต่อสัญญาณอิมพัลส์เป็น h(n) และการแปลง z ของ h(n) ได้

ค่าเป็น H(z) เรากล่าวว่า H(z) เป็นฟังก์ชันถ่ายโอนของระบบ ซึ่งมีความสัมพันธ์กับการแปลง z ของสัญญาณขาเข้า และขาออกดังสมการ (2-3)

$$H(z) = \frac{Y(z)}{X(z)} \tag{2-3}$$

2.1.2.1 วงจรกรองแบบ IIR (Infinite Impulse Response)

วงจรกรองดิจิตอลแบบ IIR จะมีฟังก์ชันถ่ายโอนของระบบดังสมการ (2-4)

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{N} b_k z^{-k}}{1 + \sum_{k=1}^{M} a_k z^{-k}} \tag{2-4}$$

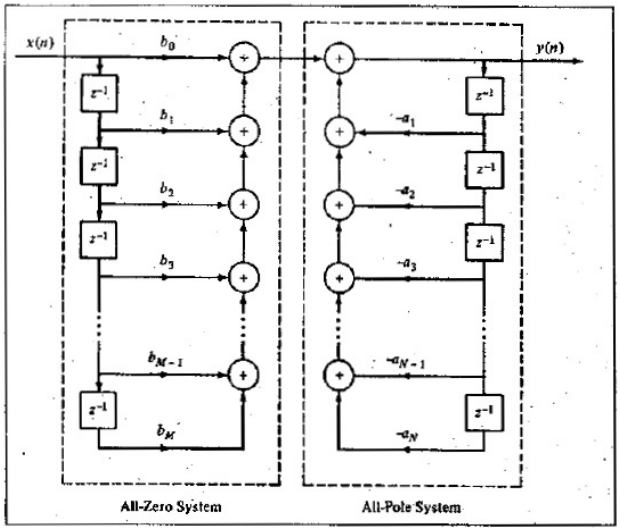
จากสมการ (2-3) และ สมการ (2-4) นำไปจัดเทอมต่างๆใหม่และเขียนให้อยู่ในรูปของสมการผลต่าง (Difference Equation) ได้ดังสมการ (2-5)

$$Y(z) = \sum_{k=0}^{N} b_k X(z) Z^{-k} - \sum_{k=1}^{M} a_k Y(z) Z^{-k}$$

$$Y(z) = b_0 X(z) + b_1 X(z) Z^{-1} + \ldots + b_k X(z) Z^{-k} + a_1 Y(z) Z^{-1} + \ldots + a_k Y(z) Z^{-k}$$

$$Y(n) = \sum_{k=0}^{N} b_k X(n-k) - \sum_{k=1}^{M} a_k Y(n-k) \tag{2-5}$$

จากสมการ (2-5) สามารถนำมาเขียนเป็นโครงสร้างได้ดังภาพประกอบ 2-5 เป็นโครงสร้างแบบตรง I (Direct form I)



ภาพประกอบ 2-6 โครงสร้างวงจรกรองIIRแบบตรง I

ทั้งนี้สำหรับงานวิจัยนี้จะเลือกใช้วงจรกรองแบบ FIR ในการออกแบบดังนั้นจะกล่าวถึงหลักการการออกแบบและคุณสมบัติที่สำคัญของวงจรกรองแบบ FIR ที่เป็นเหตุผลหลักในการตัดสินใจเลือกวงจรกรองแบบ FIR มาใช้สำหรับงานวิจัย

2.1.2.2 วงจรกรองแบบ FIR (Finite Impulse Response)

วงจรกรองดิจิตอลแบบ FIR จะมีฟังก์ชันถ่ายโอนของระบบดังสมการ (2-6)

$$H(z) = \sum_{k=0}^{n-1} h(k) z^{-k} \tag{2-6}$$

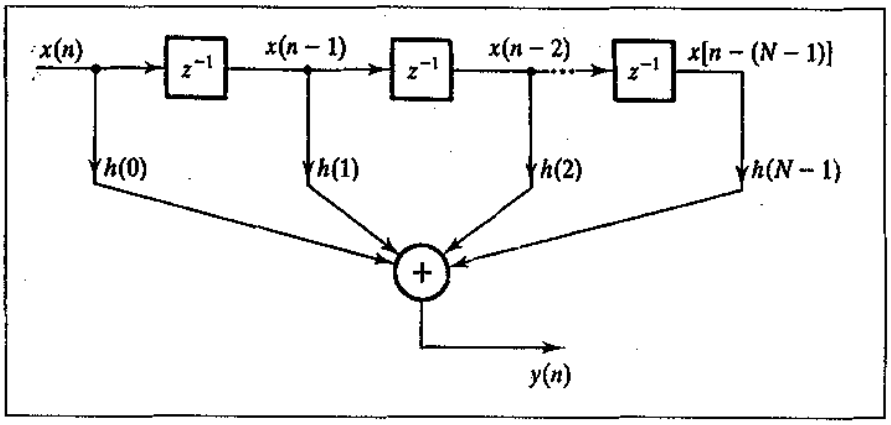
จากสมการ (2-3) และ สมการ (2-6) นำไปจัดเทอมต่างๆใหม่และเขียนให้อยู่ในรูปของสมการผลต่าง (Difference Equation) ได้ดังสมการ (2-7)

$$Y(z) = \sum_{k=0}^{n-1} h(k) X(z) Z^{-k}$$

$$Y(z) = h(0)X(z) + h(1)X(z)Z^{-1} + h(2)X(z)Z^{-2} + \ldots + h(k)X(z)Z^{-k}$$

$$Y(n) = \sum_{k=0}^{n-1} h(k) X(n-k) \tag{2-7}$$

จากสมการ (2-7) สามารถนำมาเขียนเป็นโครงสร้างได้ดังภาพประกอบ 2-7



ภาพประกอบ 2-7 โครงสร้างวงจรกรองFIR

โดยที่วงจรกรองแบบ FIR นั้นจัดเป็นวงจรกรองแบบนอนรีเคอร์ซีฟ (Non recursive) เนื่องจากจะไม่มีการป้อนกลับจากทางด้านเอาท์พุตและนอกจากนี้วงจรกรองแบบ FIR จะมีคุณสมบัติอื่นๆอีกคือ

สามารถสร้างได้ง่ายเมื่อเทียบกับวงจรกรองแบบ IIR

มีคุณสมบัติการตอบสนองทางเฟสเป็นเชิงเส้น(Linear phase)

สัมประสิทธิ์ที่เกิดจากการคำนวณจะมีค่าไม่เกินหนึ่ง

วงจรกรองแบบ FIR มีเสถียรภาพที่แน่นอน

วงจรกรองความถี่เฟสเชิงเส้น (Linear Phase Filter)

มีวงจรกรองความถี่ส่วนมากที่ยังคงต้องการรักษารูปร่างของสัญญาณเอาไว้ในขณะที่สัญญาณผ่านวงจรไป วงจรเช่นนั้นเป็นวงจรกรองความถี่เฟสเชิงเส้น ซึ่งการมีการตอบสนองความถี่ที่สามารถเขียนได้โดย

$$H(e^{j\theta}) = \left|H(e^{j\theta})\right| e^{-j\theta k} \tag{2-8}$$

เมื่อ $-\pi < \theta < \pi$ และ $H(e^{j\theta})$ เป็นการแปลงฟูริเยร์ของ h(n)

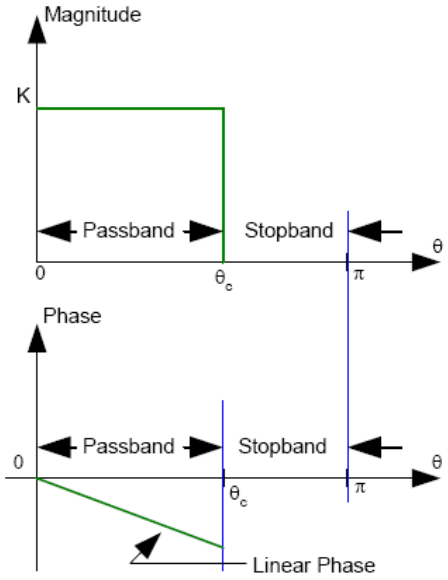
หากให้ $X(e^{j\theta})$ เป็นการแปลงฟูริเยร์ของ x(n) ให้การตอบสนองมีค่า $\left|H(e^{j\theta})\right|$ = 1 ในช่วงผ่านความถี่ สมการของเอาท์พุตเขียนได้เป็น

$$Y(e^{j\theta}) = X(e^{j\theta})H(e^{j\theta}) = X(e^{j\theta})e^{-j\theta k} \tag{2-9}$$

หรือเมื่อแปลงกลับฟูริเยร์

$$y(n) = x(n-k) \tag{2-10}$$

จะเห็นว่าวงจรกรองความถี่เฟสเชิงเส้นไม่ได้เปลี่ยนรูปร่างของอินพุทไป หากวงจรกรองความถี่ที่ $\left|H(e^{j\theta})\right| = 1$ แต่ไม่ได้เป็นเฟสเชิงเส้นก็จะไม่สามารถรักษารูปร่างของอินพุทไว้ได้



ภาพประกอบ 2-8 การตอบสนองความถี่ทั้งขนาดและเฟส (เชิงเส้น) ของวงจรกรองผ่านความถี่ต่ำในอุดมคติ

หาก h(n) เป็นการตอบสนองอิมพัลส์ของระบบสัญญาณเต็มหน่วย สำหรับวงจรกรองความถี่เฟสเชิงเส้นมีความสำคัญและเป็นเงื่อนไขพอเพียงที่ว่า h(n) ซึ่งจำกัดจำนวน N จะต้องมีความสมมาตรประมาณกึ่งกลาง h(n) = h(N-n-1) ค่าสัมประสิทธิ์ทางด้านซ้ายและด้านขวาจะเท่ากัน การคูณจึงสามารถที่จะลดลงมาได้ครึ่งหนึ่ง

พิจารณาสมการ (2-10) เมื่อทำการแปลงฟูเรียร์ในสมการที่ (2-10) จะได้

$$Y(j\omega = e^{-j\omega kT} X(j\omega) \tag{2-11}$$

ทำการย้ายข้างสมการ (2-11) จะได้ว่า

$$\frac{Y(j\omega)}{X(j\omega)} = H(j\omega) = e^{-j\omega kT} \qquad (2\text{-}12)$$

จากสมการจะมีค่าขนาด (magnitude)เท่ากับ 1 และมีเฟสดังสมการที่ (2-13)

$$\theta(\omega) = -\omega kT \qquad (2\text{-}13)$$

การที่วงจรกรองแบบ FIR มีผลตอบสนองทางเฟสเป็นแบบเชิงเส้นดังนั้นวงจรกรองแบบ FIR จึงถูกนำไปใช้งานอย่างกว้างขวาง เช่น ด้านการรับส่งข้อมูล วิดีโอ เครื่องมือแพทย์ เป็นต้น

ผลตอบสนองต่ออิมพัลส์ของระบบ (*h(n)*)

สำหรับตัวกรองแบบ FIR นั้น ในการออกแบบสิ่งที่เราต้องการหา คือ ค่าของผลตอบสนองต่ออิมพัลส์ หรือ h(n) ของระบบ สำหรับตัวกรอง FIR ที่มี h(n) ยาว N จุด เรากล่าวว่าตัวกรองนี้มีอันดับเท่ากับ N-1 เหตุผลก็คือ มีการใช้สัญญาณขาเข้าในอดีตย้อนหลังไป N-1 ตำแหน่ง หรือ ตัวกำลังสูงสุดที่อยู่ในฟังก์ชัน H(z) ก็คือ $z^{-(N-1)}$ และจากสมการที่ (2-13) ซึ่งเป็นสมการของเฟสถ้ากำหนดสมการผลตอบสนองทางเฟสใหม่โดยกำหนดให้

$$kT = \alpha \qquad (2\text{-}14)$$

จะได้

$$\theta(\omega) = -\alpha\omega \qquad (2\text{-}15)$$

หรือถ้าค่าของผลตอบสนองทางเฟสเท่ากับ

$$\theta(\omega) = \beta - \alpha\omega \qquad (2\text{-}16)$$

β เป็นค่าคงที่

ถ้าวงจรกรองมีผลการตอบสนองทางเฟสเป็นแบบเชิงเส้นตามสมการที่ (2-15) จะให้ค่าผลตอบสนองอิมพัลส์ของตัวกรอง h(n) เป็นแบบสมมาตรบวก (Positive symmetry) ดังสมการที่ (2-17)และ(2-18)

$$h(n) = h(N-n-1) \quad \text{ที่ n = 0,1,…..,(N-1)/2 และ n เป็นคี่} \qquad (2\text{-}17)$$

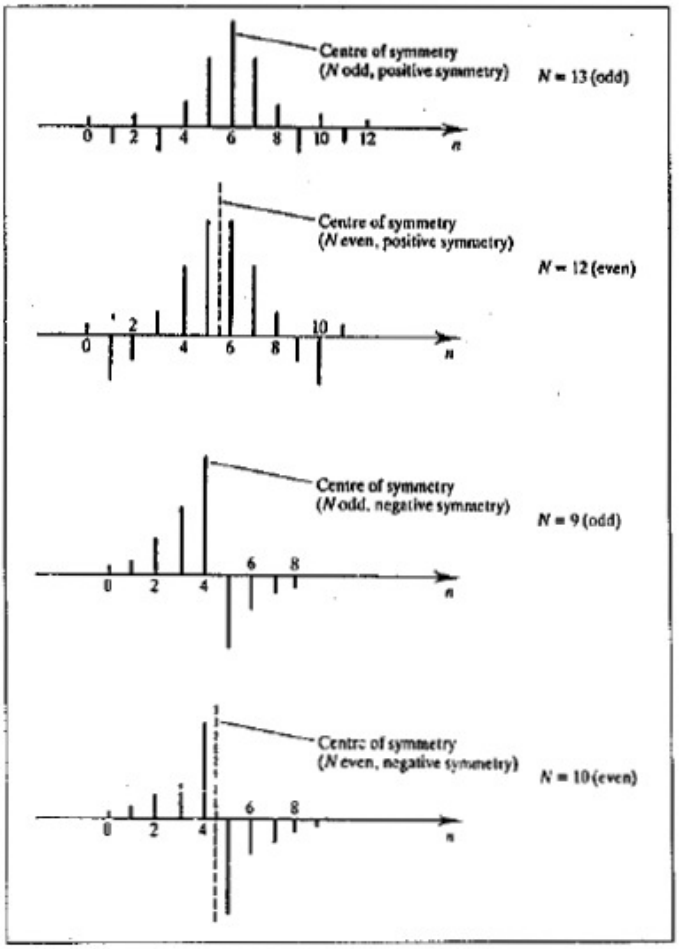
$$\alpha = \frac{(N-1)}{2} \quad \text{ที่ n = 0,1,…..,(N-1)/2 และ n เป็นคู่} \qquad (2\text{-}18)$$

และหากวงจรกรองมีผลการตอบสนองทางเฟสเป็นแบบเชิงเส้นตามสมการที่ (2-16) จะให้ค่าผลตอบสนองอิมพัลส์ของตัวกรอง h(n) เป็นแบบสมมาตรลบ (Negative symmetry) ดังสมการที่ (2-19) และ (2-20)

$$h(n) = -h(N-n-1) \tag{2-19}$$

$$\alpha = \frac{(N-1)}{2} \tag{2-20}$$

จากสมการ (2-17) และสมการ (2-19) สามารถนำมาเขียนเป็นกราฟได้ดังภาพประกอบ 2-9

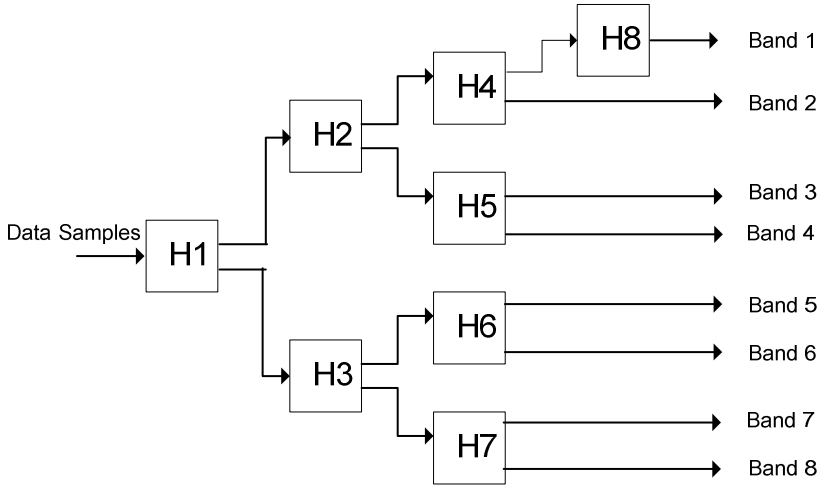


ภาพประกอบ 2-9 ผลตอบสนองอิมพัลส์ของเฟสเชิงเส้นของวงจรกรองทั้ง 4 ชนิด

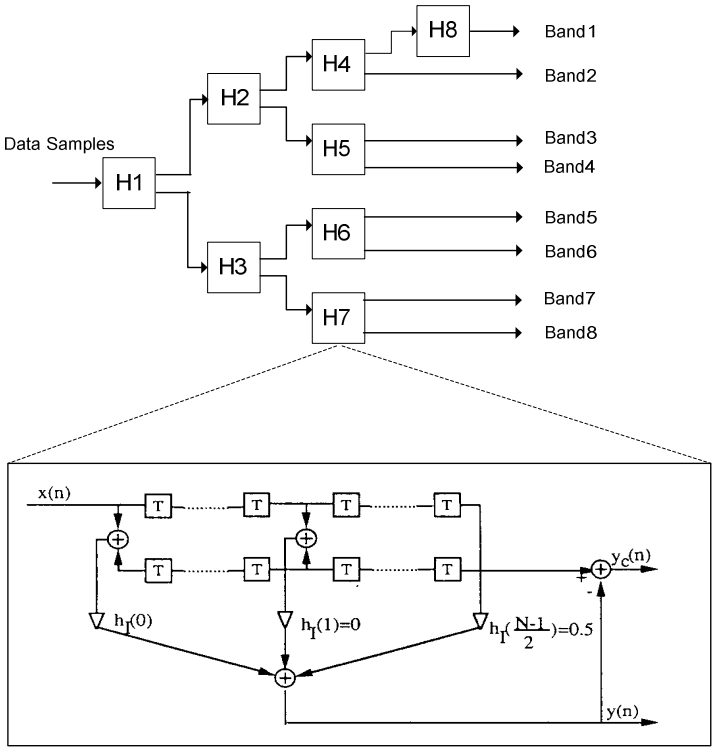
2.2 วงจรดิจิตอลฟิลเตอร์แบงค์ (Digital Filter Bank Circuit)

ฟิลเตอร์ แบงค์ คือวงจรกรองสั ญญาณเชิงดิ (Digital Filter) ประเภทหนึ่ ง มี การ ทำ งานคือทำ หน้าที่ แยกสั ญญาณด้านเข้าที่ ถู กป้ อนเข้ามาออกเป็ นช่วงความถี่ ต่างๆ เพื่ อปรั บปรุ งคุ ณภาพของสั ญญาณ โดยสั ญญาณที่ ถู กป้ อนเข้ามาจะถู กแยกหรือกรองเป็ นสั ญญาณความถี่ ต่างๆกั นไปซึ่ งเป็ นผลมาจากการทำ งานของฟิลเตอร์ ย่อยที่ อยู่ภายในฟิลเตอร์ แบงค์ ดังนั้ นสั ญญาณที่ ได้จากฟิลเตอร์ ย่อยตั วแรกจะถู กป้ อนเป็ นอินพุ ตให้แก่ฟิลเตอร์ ย่อยอื่ นๆที่ อยู่ต่อไปในแบงค์ ไปตามลำ ดั บจนครบซึ่ ง ฟิลเตอร์ แต่ละตั วเหล่านั้ นก็ จะทำ หน้าที่ แยกหรือกรองสั ญญาณที่ ได้รั บเพื่ อปรั บปรุ งคุ ณภาพของสั ญญาณจากการประมวลผลของฟิลเตอร์ แต่ละตั วเช่นกั น สุ ดท้ายจะได้สั ญญาณด้านออกของฟิลเตอร์ แบงค์

วงจรดิจิตอลฟิลเตอร์แบงค์ที่ใช้ในงานวิจัยนี้เป็นวงจรกรองดิจิตอลที่มีสัญญาณด้านเข้า 1 ช่อง ภายในประกอบไปด้วยบล็อกประมวลผล 8 บล็อก เพื่อทำการแยกสัญญาณที่เข้ามาออกเป็นสัญญาณด้านออก 8 ช่องสัญญาณ โดยแสดงได้ดังภาพประกอบ 2-10



ภาพประกอบ 2-10 วงจรดิจิตอลฟิลเตอร์แบงค์แบบเอาท์พุท 8 ช่องสัญญาณ



ภาพประกอบ 2-11 การประมวลผลของฟิลเตอร์แต่ละตัวภายในฟิลเตอร์แบงค์

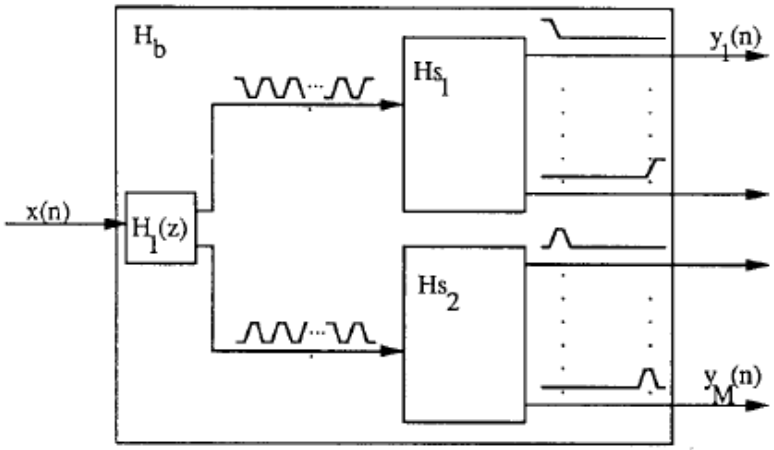
จากภาพประกอบที่ 2-11 แสดงให้เห็นถึงการทำงานขององค์ประกอบพื้นฐานของวงจรกรองดิจิตอล โดยหากสังเกตจะพบว่าโครงสร้างภายในของฟิลเตอร์ย่อยนั้นมีโครงสร้างแบบเดียวกันกับโครงสร้างของวงจรกรองแบบ FIR ซึ่งโครงสร้างการทำงานนี้สามารถเขียนให้อยู่ในของฟังก์ชันทางคณิตศาสตร์ในรูปสมการผลต่างของระบบ (Difference Equation) ได้ดังสมการ (2-7) ผลต่างของวงจรกรองแบบ FIR ที่กล่าวผ่านมาในหัวข้อวงจรกรองแบบ FIR

โดยการทำงานของวงจรฟิลเตอร์แบงค์นี้คือ ฟิลเตอร์ย่อยที่อยู่ภายในแต่ละตัวมีการทำงานที่เหมือนกันคือ รับสัญญาณอินพุตที่ด้านเข้าจากนั้นประมวลผลสัญญาณภายใต้สมการผลต่างโดยที่ฟิลเตอร์ย่อยแต่ละตัวจะมีสมการเฉพาะสำหรับฟิลเตอร์แต่ละตัว โดยการที่จะสร้างสมการผลต่างของฟิลเตอร์ได้นั้น จะต้องทราบค่าผลตอบสนองต่ออิมพัลส์ ($h(n)$) ซึ่งภาพประกอบ 2-12 แสดงให้เห็นความสัมพันธ์ของค่าตอบสนองต่ออิมพัลส์ของฟิลเตอร์แต่ละตัวเพื่อช่วยในการสร้างสมการผลต่างของฟิลเตอร์แต่ละตัวที่ใช้ประมวลผลสัญญาณภายในวงจรฟิลเตอร์แบงค์นั่นเอง

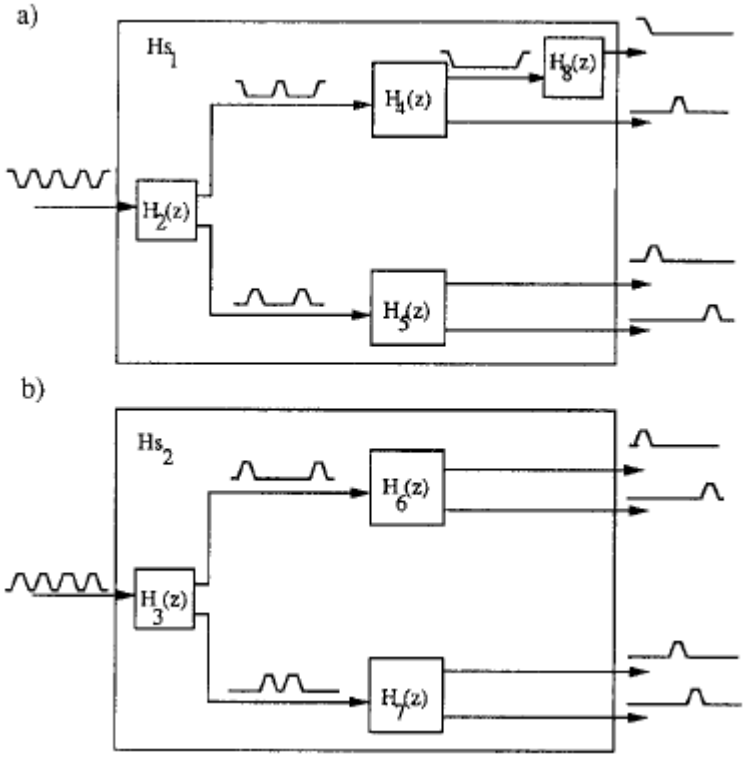
$H_1(z)$	$h_1(0)=h_1(48),\ h_1(16)=h_1(32),\ h_1(24)=0.5$
$H_2(z)$	$h_2(0)=h_2(24),\ h_2(8)=h_2(16),\ h_2(12)=0.5$
$H_3(z)$	$h_3(0)=h_3(28),\ h_3(4)=h_3(24),\ h_3(8)=h_3(20),\ h_3(12)=h_3(16),$ $h_3(14)=0.5$
$H_4(z)$	$h_4(0)=h_4(12),\ h_4(4)=h_4(8),\ h_4(6)=0.5$
$H_5(z)$	$h_5(0)=h_5(10),\ h_5(2)=h_5(8),\ h_5(4)=h_5(6),\ h_5(5)=0.5$
$H_6(z)$	$h_6(0)=h_6(6),\ h_6(2)=h_6(4),\ h_6(5)=0.5$
$H_7(z)$	$h_7(0)=h_7(30),\ h_7(6)=h_7(24),\ h_7(12)=h_7(18),\ h_7(15)=0.5$
$H_8(z)$	$h_8(0)=h_8(2),\ h_8(1)=0.5$

ภาพประกอบ 2-12 ความสัมพันธ์ของค่าผลตอบสนองต่ออิมพัลส์ของฟิลเตอร์แต่ละตัว

การทำงานของฟิลเตอร์แบงค์จะเริ่มจากสัญญาณอินพุต x(n) ผ่านเข้ามาที่ฟิลเตอร์ $H_1(z)$ จากนั้นสัญญาณจะถูกประมวลผลแล้วส่งค่าเอาท์พุตออกมาเป็นสองค่าคือ $y_1(n)$ กับ $y_{1c}(n)$ โดยที่เอาท์พุตสองค่านี้หากสังเกตจากภาพประกอบ 2-13 จะเห็นว่าเป็นค่าที่ตรงข้ามกันซึ่งกันและกันนั่นเอง จากภาพประกอบเมื่อได้ค่าเอาท์พุตสองค่าดังที่กล่าวมาแล้ว ค่าเอาท์พุตนี้จะถูกส่งไปประมวลผลต่อด้วยฟิลเตอร์ในลำดับถัดไป (Hs_1 และ Hs_2) จนได้สัญญาณเอาท์พุตของฟิลเตอร์แบงค์ ตั้งแต่ $y_1(n)$ จนถึง $y_M(n)$



ภาพประกอบ 2-13 โครงสร้างของฟิลเตอร์แบงค์และฟิลเตอร์ย่อยภายในฟิลเตอร์แบงค์



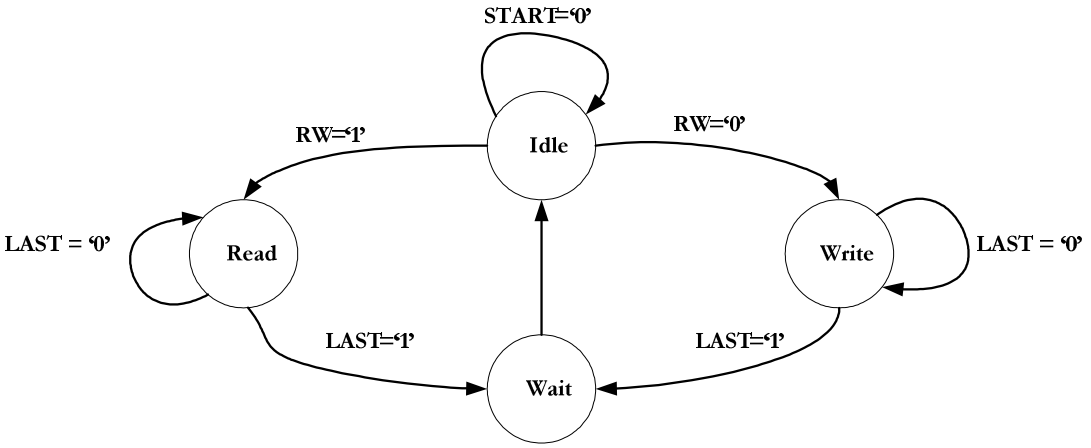
ภาพประกอบ 2-14 ลำดับการประมวลผลของฟิลเตอร์ (a) ฟิลเตอร์ย่อย Hs_1 (b) ฟิลเตอร์ย่อย Hs_2

จากภาพประกอบ 2-14 แสดงลำดับในการประมวลผลของฟิลเตอร์ย่อย Hs_1 และ Hs_2 เพื่อให้ได้ค่าเอาท์พุตทั้งหมดครบ โดยที่ ฟิลเตอร์ย่อย Hs_1 จะรับค่าอินพุตซึ่งคือค่าเอาท์พุต $y_1(n)$ ของฟิลเตอร์ $H_1(z)$ มาป้อนให้ฟิลเตอร์ $H_2(z)$ ซึ่งทำการแยกสัญญาณออกมาได้ค่าเอาท์พุตเป็น $y_2(n)$ และ $y_{2c}(n)$ จากนั้นค่าสัญญาณ $y_2(n)$ จะถูกป้อนเป็นอินพุตให้กับฟิลเตอร์ $H_4(z)$ โดยจากฟิลเตอร์ $H_4(z)$ นั้นสัญญาณอินพุตจะถูกประมวลผลได้ค่าสัญญาณเอาท์พุตเป็น $y_4(n)$ กับ $y_{4c}(n)$ ค่าเอาท์พุต $y_4(n)$ จะเป็นค่าอินพุตให้แก่ $H_8(z)$ จนสุดท้ายจะได้ค่า $y_8(n)$ ส่วนค่า $y_{8c}(n)$ จะไม่นำมาใช้ และสำหรับสัญญาณ $y_{2c}(n)$ จะป้อนไปยังฟิลเตอร์ $H_5(z)$ ได้ค่าเอาท์พุตเป็น $y_5(n)$ กับ $y_{5c}(n)$ ดังนั้นสำหรับฟิลเตอร์ย่อย Hs_1 เมื่อทำการประมวลผลเสร็จแล้วนั้นจะได้ค่าสัญญาณทั้งหมด 4 ค่า

ฟิลเตอร์ย่อย Hs_2 มีการทำงานที่เป็นไปในลักษณะเดียวกันกับ ฟิลเตอร์ย่อย Hs_1 โดยที่ค่าอินพุตเริ่มต้นคือค่า $y_{1c}(n)$ และเมื่อฟิลเตอร์ภายในฟิลเตอร์ย่อย Hs_2 ทำงานเสร็จแล้วนั่นจะได้ค่าเอาท์พุตอีก 4 ค่า ซึ่งเมื่อไปรวมกับเอาท์พุตจากฟิลเตอร์ย่อย Hs_1 ก็จะได้ค่าสัญญาณเอาท์พุตทั้งหมด 8 ค่านั่นเอง

2.3 หลักการของ Finite State Machine [ชำนาญ ปัญญาใส และ วัชรากร หนูทอง, 2547]

Finite State Machine ซึ่งเป็นหัวใจสำคัญในการออกแบบหน่วยควบคุม (Control Unit) โดยปกติจะบรรยายด้วยแผนภาพไดอะแกรม ตัวอย่างเช่น ภาพประกอบ 2-15 จะแสดงการเปลี่ยน สเตต (State) ของการอ่านเขียนของหน่วยควบคุม โดยเริ่มการทำงานจากสเตต Idle ถ้าสัญญาณ START มีค่าไปเป็น '1' สเตตจะเปลี่ยนไปเป็นสเตต Write หรือสเตต Read ขึ้นกับสัญญาณ RW ถ้าเป็น '1' จะเข้าสู่สเตต Read และถ้าเป็น '0' จะเข้าสู่สเตต Write และจากสเตต Read และสเตต Write จะนำไปสู่สเตต Wait ถ้าสัญญาณ LAST เป็น '1' และเข้าสู่สเตต Idle ต่อไป



ภาพประกอบ 2-15 ตัวอย่างไดอะแกรมของหน่วยควบคุม (Control Unit)

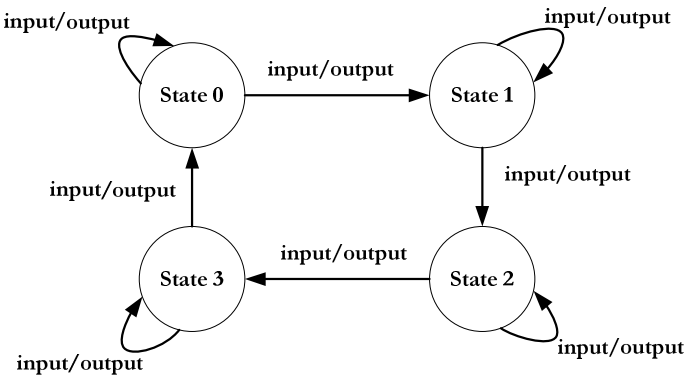
โดยพื้นฐานของการออกแบบ FSM จะมี 2 รูปแบบด้วยกันคือ Mealy และ Moore Machine

- Mealy Machine จะเป็นการเขียนสเตตไดอะแกรมรูปแบบ ที่ค่าของเอาท์พุตในแต่ละสเตตจะขึ้นอยู่กับค่าของสเตตปัจจุบัน (Current State) และค่าของอินพุตในแต่ละสเตต

- Moore Machine จะเป็นการเขียนสเตตไดอะแกรมรูปแบบ ที่ค่าของเอาท์พุตในแต่ละสเตตจะขึ้นอยู่กับค่าของสเตตปัจจุบัน (Current State) เท่านั้นโดยในรายละเอียดของวงจรทั้งสองนี้จะกล่าวในหัวข้อ 2.3.1 และ 2.3.2 ตามลำดับ

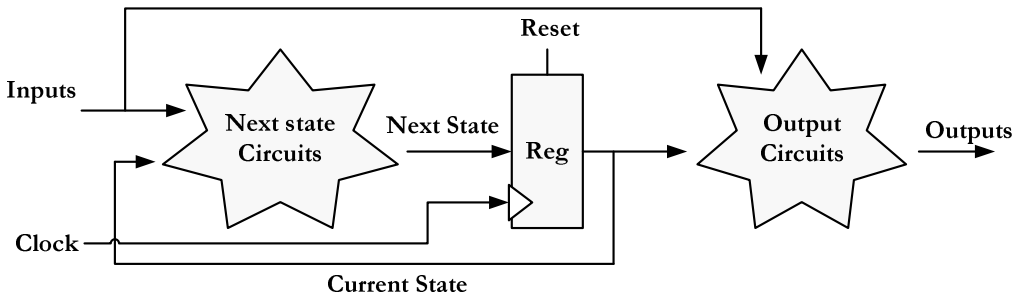
2.3.1 Mealy FSM

Mealy Machine เป็นรูปแบบการเขียนสเตตไดอะแกรมแบบที่ค่าของเอาท์พุตในแต่ละสเตตจะขึ้นอยู่กับค่าของสเตตปัจจุบัน (Current state) และค่าของอินพุตในแต่ละสเตตด้วย ซึ่งจะแตกต่างจาก Moore Machine ที่ค่าเอาท์พุตจะเปลี่ยนแปลงขึ้นอยู่กับค่าของสเตตปัจจุบันเท่านั้น เพื่อความเข้าใจให้พิจารณาจากภาพประกอบ 2-16



ภาพประกอบ 2-16 แผนภาพไดอะแกรมแบบ Mealy Machine

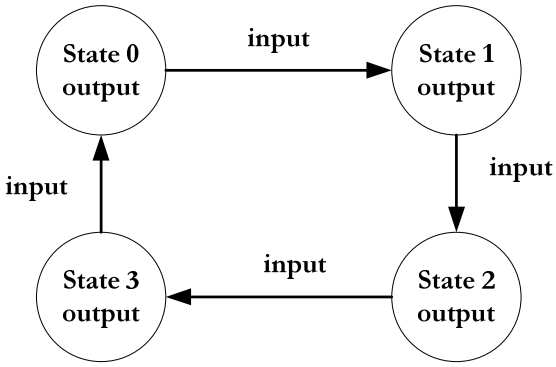
ในแผนภาพไดอะแกรมมีทั้งหมด 4 สเตต โดยค่าของเอาท์พุตในแต่ละสเตตจะขึ้นอยู่กับค่าของอินพุตและค่าของสเตต ปัจจุบันนั้นๆ ซึ่งถ้ามองในรูปของฮาร์ดแวร์หรือวงจรสามารถเขียนให้เป็นบล็อกไดอะแกรม ได้ดังภาพประกอบ 2-17 ซึ่งจะประกอบไปด้วยวงจรคอมบิเนชั่นในส่วนของ Next State และส่วนของรีจิสเตอร์ที่ใช้เก็บค่าสถานะของสเตตและวงจรคอมบิเนชันในส่วนการสร้างสัญญาณเอาท์พุต ที่นำค่าของอินพุตมาพิจารณาในการเปลี่ยนค่าเอาท์พุตในแต่ละสเตตด้วย



ภาพประกอบ 2-17 บล็อกไดอะแกรมแบบ Mealy Machine ในมุมมองทางฮาร์ดแวร์

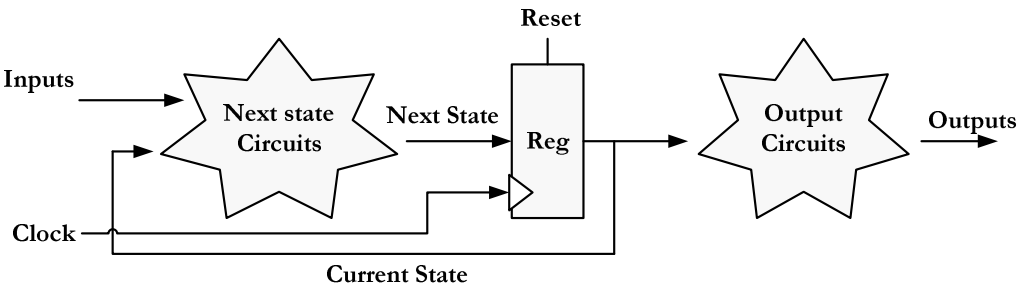
2.3.2 Moore FSM

Moore Machine เป็นรูปแบบการเขียนสเตตไดอะแกรมแบบค่าของเอาท์พุตในแต่ละสเตตจะขึ้นอยู่กับค่าของสเตตปัจจุบัน (Current State) เท่านั้น เพื่อความเข้าใจพิจารณาจากแผนภาพสเตตไดอะแกรม ดังภาพประกอบ 2-18



ภาพประกอบ 2-18 แผนภาพไดอะแกรมรูปแบบ Moore Machine

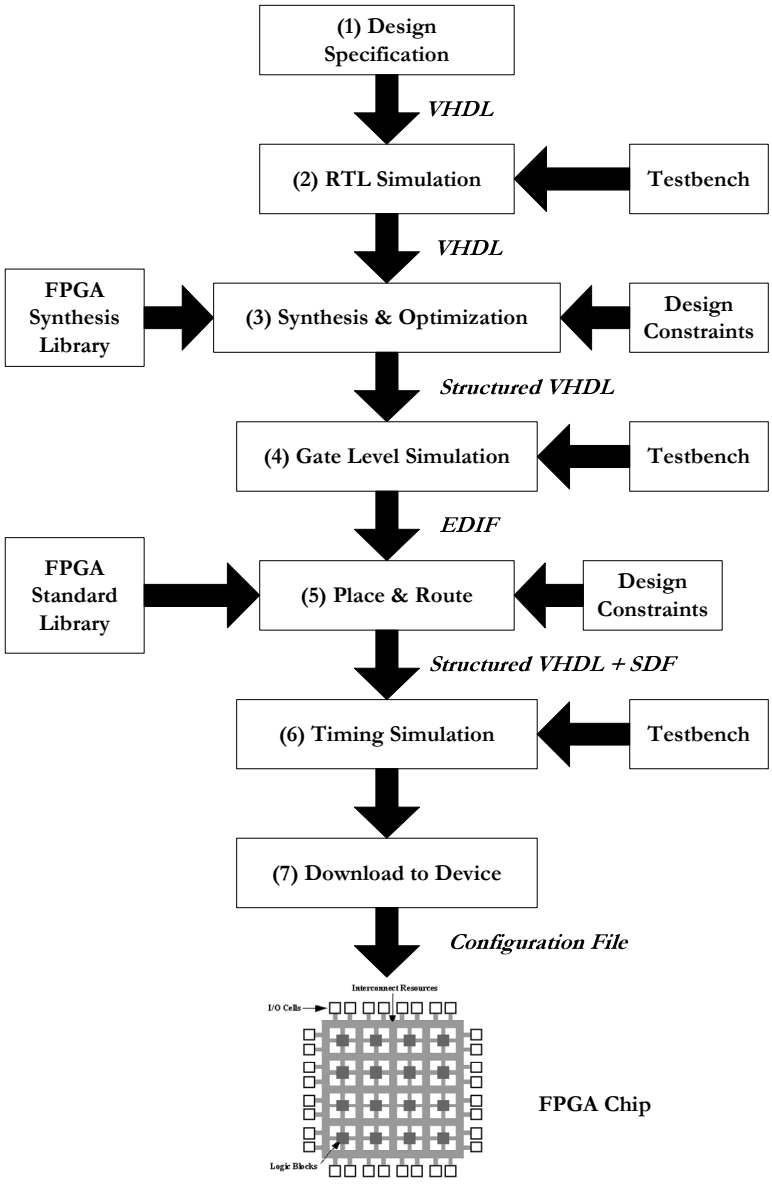
ในรูปไดอะแกรมมีทั้งหมด 4 สเตต โดยค่าของเอาท์พุตในแต่ละสเตตจะถูกเขียนไว้ในวงกลมของสเตตนั้นๆ และการเปลี่ยนแปลงสเตตจะขึ้นอยู่กับค่าอินพุตที่เข้ามาในแต่ละสเตต ซึ่งถ้ามองในรูปของฮาร์ดแวร์หรือวงจรสามารถที่จะเขียนเป็นบล็อกไดอะแกรมได้ดังภาพประกอบ 2-19 ซึ่งจะประกอบไปด้วยวงจรคอมบิเนชันในส่วนของ Next State และส่วนของวงจรรีจิสเตอร์ที่ใช้ในการเก็บค่าสถานะของสเตตและวงจรคอมบิเนชันในส่วนการสร้างสัญญาณเอาท์พุต



ภาพประกอบ 2-19 บล็อกไดอะแกรมแบบ Moore Machine ในมุมมองทางฮาร์ดแวร์

2.4 ขั้นตอนการออกแบบวงจรบนเทคโนโลยี FPGA

ในปัจจุบันได้มีบริษัทผู้ผลิตชิพที่สามารถนำวงจรที่ออกแบบขึ้นจากภาษา HDL ไปใช้งานได้จริง ชิพดังกล่าวคือชิพจำพวก FPGA (Field Programmable Gate Array) เป็นชิพที่จำลองการทำงานตามฟังก์ชันที่ออกแบบขึ้นจากภาษา HDL โดยในปัจจุบันมีบริษัทผู้ผลิตชิปรายใหญ่อยู่ 2 บริษัท ได้แก่ บริษัท Xilinx และ บริษัท Altera ซึ่งทั้งสองบริษัทจะมีซอฟท์แวร์ช่วยในการออกแบบ โดยทั่วไปการออกแบบวงจรด้วย FPGA จะมีขั้นตอนต่างๆดังภาพประกอบ 2-18



ภาพประกอบ 2-20 ขั้นตอนการออกแบบวงจรด้วย FPGA

2.4.1 การสร้างข้อกำหนดของการออกแบบ (Design Specification)

เป็นขั้นตอนการสร้างข้อกำหนดต่างๆของวงจร เช่น วงจรทำงานที่ความถี่เท่าไหร่ ฟังก์ชันการทำงานมีอะไรบ้าง ซึ่งเป็นรายละเอียดของวงจรที่ต้องการออกแบบและเขียนฟังก์ชั่นของการทำงานวงจรตามที่ผู้ออกแบบกำหนดด้วยภาษา VHDL ในระดับ RTL

2.4.2 จำลองการทำงานโมเดลวงจรระดับ RTL (RTL Simulation)

เป็นขั้นตอนตรวจสอบการทำงานของโมเดลวงจรในระดับ RTL หรือฟังก์ชัน โดยการจำลองการทำงาน จะถูกทดสอบด้วย Testbench ซึ่งการจำลองการทำงานในขั้นตอนนี้ จะจำลองการทำงานเพียงฟังก์ชัน ไม่คำนึงถึงค่าดีเลย์ (Delay) ของวงจรแต่อย่างใด

2.4.3 สังเคราะห์และออปติไมซ์วงจร (Synthesis & Optimization)

เป็นขั้นตอนการสร้างแผนภาพวงจร (Schematic) จากโมเดลวงจรระดับ RTL ให้อยู่ในรูปของลอจิกเกต โดยอาศัยซอฟต์แวร์ช่วยในการสังเคราะห์วงจร โดยในขั้นตอนนี้จะต้องมีการเลือกใช้เทคโนโลยี FPGA ที่ผู้ออกแบบต้องการเลือกใช้ ซึ่งบริษัทผู้ผลิต FPGA จะมีเทคโนโลยีไลบรารี (Technology Library) เตรียมไว้ให้ผู้ออกแบบในซอฟต์แวร์ที่ใช้พัฒนาชิพ FPGA ของแต่ละบริษัทผู้ผลิตไว้อยู่แล้ว โดยเมื่อทำการสังเคราะห์ผังวงจร จากนั้นซอฟต์แวร์จะทำการออปติไมซ์ (Optimize) วงจรตามข้อกำหนด หรือเงื่อนไขของการสังเคราะห์ (Design constraint) ที่ผู้ออกแบบกำหนดขึ้นตามขั้นตอนแรกของการออกแบบ ซึ่งผลลัพธ์ที่ได้จากการสังเคราะห์วงจรจะอยู่ในรูปแบบของไฟล์ VHDL แบบโครงสร้างลอจิก (Structured VHDL) และไฟล์เน็ตลิสต์มาตรฐาน (Netlist) ประเภท EDIF (Electronic Design Interchange Format) ที่จะนำไปใช้ในขั้นตอน Place & Route ต่อไป

2.4.4 การจำลองการทำงานของวงจรระดับลอจิกเกต (Gate level Simulation)

เป็นขั้นตอนที่ผู้ออกแบบจะต้องทดสอบไฟล์เน็ตลิสต์ที่เป็นโมเดลของวงจรระดับลอจิก โดยใช้ Testbench ตัวเดิมที่ใช้จำลองการทำงานระดับ RTL มาแล้ว ซึ่งในการจำลองการทำงานในระดับนี้ จะมีเรื่องของเกตดีเลย์ (Gate Delay) เข้ามาเกี่ยวข้องในผลการจำลองการทำงาน ซึ่งจะแตกต่างจากการจำลองการทำงานในระดับ RTL เนื่องจากโมเดลของวงจรในระดับเกตนี้ จะมีข้อมูลเรื่องดีเลย์ของเกตภายในเทคโนโลยีที่ผู้ออกแบบเลือกมาใช้ ดังนั้นผู้ออกแบบจะต้องทำการทดลองการทำงานเพื่อตรวจสอบไทม์มิ่งอีกครั้งหนึ่ง ว่ายังถูกต้องตามข้อกำหนดของวงจรหรือไม่ ถ้าไม่ตรงตามข้อกำหนดจะต้องกลับไปขั้นตอนที่3 เพื่อทำการสังเคราะห์และออปติไมซ์วงจรใหม่ เพื่อให้ผลการจำลองการทำงานถูกต้องตาม Design specification

2.4.5 การวางและเชื่อมต่อเซลล์ภายในของ FPGA (Place & Route)

เมื่อตรวจสอบการทำงานในระดับลอจิกเกตเป็นที่เรียบร้อยแล้ว เราจะนำไฟล์เน็ตลิสต์ที่อยู่ในรูปแบบ EDIF มาทำการแปลงลงสู่เทคโนโลยีเซลล์ภายในของ FPGA และทำการเชื่อมต่อเซลล์ภายในเข้าด้วยกัน ตามรูปแบบการเชื่อมต่ออุปกรณ์ต่างๆภายในเน็ตลิสต์ โดยขั้นตอนนี้ จะมีการเรียกใช้เทคโนโลยีเซลล์ของ FPGA เนื่องจากเทคโนโลยีเซลล์ของ FPGA อาจมีมาโครเซลล์ (Macro cells) สำหรับสร้างฟังก์ชันต่างๆให้กับผู้ออกแบบใช้งาน เพราะมาโครเซลล์ต่างๆพวกนี้ถูกทางผู้ผลิตออกแบบไว้สำหรับชิพ FPGA แต่ละตัว หรือกล่าวได้ว่า เป็นวงจรที่ออป ติไมซ์มาแล้วสำหรับชิพแต่ละเทคโนโลยี ซึ่งในการวางและเชื่อมต่อเซลล์หรือมาโครเซลล์ต่างๆจะถูกควบคุมด้วยข้อกำหนดในการ Place & Route เพื่อให้ซอฟต์แวร์ทำการและเชื่อมต่อกันตามความต้องการของผู้ออกแบบ ที่เราเรียกว่า Design constraints โดยผลลัพธ์จากขั้นตอนนี้ จะอยู่ในรูปของไฟล์ VHDL โดยโครงสร้างของเซลล์ภายในของ FPGA และไฟล์ประเภท SDF(Standard Delay Format) ซึ่งเป็นไฟล์รูปแบบมาตรฐานที่มีข้อมูลเกี่ยวกับค่าดีเลย์ของเส้นทางการเชื่อมต่อภายใน (Routing delay) เซลล์ FPGA และไฟล์ที่ใช้สำหรับไปโปรแกรมชิพ FPGA ซึ่งส่วนใหญ่จะอยู่ในรูปแบบของ Configuration File

2.4.6 การจำลองการทำงานระดับฐานเวลาจริง (Timing Simulation)

เป็นขั้นตอนสุดท้ายของการตรวจสอบความถูกต้อง ก่อนจะนำวงจรที่ออกแบบไปโปรแกรม หรือบางครั้งเรียกว่า Download ลงสู่ชิพจริงต่อไป โดยขั้นตอนการจำลองการทำงานในระดับไทม์มิ่ง (Timing model) โดยผลลัพธ์จากการจำลองการทำงานที่ได้จะมีความใกล้เคียงกับไทม์มิ่งการทำงานจริงบนชิพ FPGA เนื่องจากในการจำลองการทำงานในขั้นตอนนี้มีข้อมูลเกี่ยวกับดีเลย์ของเซลล์ภายใน FPGA และดีเลย์ของการเชื่อมต่อเซลล์เข้ามาเกี่ยวข้อง ทำให้ผลการจำลองการทำงานในระดับนี้ใกล้เคียงกันกับไทม์มิ่งการทำงานบนฮาร์ดแวร์ (Hardware) จริง

2.4.7 โปรแกรมลงสู่ชิพจริง (Download to device)

เป็นขั้นตอนสุดท้ายสำหรับการออกแบบวงจร เพื่อการใช้งานชิพ FPGA คือขั้นตอนโปรแกรม Configuration File ลงสู่ชิพ FPGA บนบอร์ด เพื่อทดสอบการทำงานจริงต่อไป

บทที่ 3

การออกแบบวงจร

3.1 การออกแบบฟิลเตอร์เพื่อหาค่าผลตอบสนองต่ออิมพัลส์ หรือ h(n)

จากบทก่อนหน้านี้ได้กล่าวนำไว้ว่างานวิจัยนี้จะเลือกใช้วงจรกรองแบบ FIR ทั้งนี้ด้วยเหตุผลหลักคือวงจรกรองแบบ FIR นี้ให้ผลตอบสนองเฟสที่เป็นเชิงเส้นและมีเงื่อนไขของการสมมาตรสำหรับ h(n) เป็นแบบชนิดที่ 1 คือ h(n) เป็นแบบสมมาตรบวกและ N เป็นเลขคี่ และเนื่องจากในการที่จะออกแบบวงจรกรองได้นั้นจะต้องเริ่มจากการหาค่าผลตอบสนองต่ออิมพัลส์ h(n) ก่อนเพื่อใช้ในการสร้างสมการผลต่างโดยคุณสมบัติของวงจรกรองแบบ FIR ที่กล่าวไปข้างต้นแล้วนั้นจะถูกนำมาใช้ประโยชน์ในการที่จะใช้หาค่า h(n) ของระบบที่ต้องการ โดยเริ่มต้นที่พิจารณาสมการของความสัมพันธ์ของอินพุตและเอาท์พุตของฟิลเตอร์ดังสมการ (3-1)และ(3-2) รวมไปถึงสมการแสดงความสมมาตรของค่าผลตอบสนองต่ออิมพัลส์ดังสมการ (3-3)

$$y(n) = \sum_{i=0}^{N} x(n-i)h(i) \tag{3-1}$$

$$y_c(n) = x(n-(N-1)/2) - y(n) \tag{3-2}$$

โดยที่ $y(n)$ คือ เอาต์พุทของระบบ

$y_c(n)$ คือ เอาต์พุทที่เป็น complementary $y(n)$

$h(i)$ คือ Impulse Response

n คือ Samples

N -1 คือ System Order

$h(n) = h(N-n-1)$ ที่ n = 0,1,…..,(N-1)/2 และ n เป็นคี่ (3-3)

ตารางที่ 3-1 ความสัมพันธ์ของค่าผลตอบสนองต่ออิมพัลส์ของฟิลเตอร์ย่อย

$H_1(z)$	$h_1(0)=h_1(48),\ h_1(16)=h_1(32),\ h_1(24)=0.5$
$H_2(z)$	$h_2(0)=h_2(24),\ h_2(8)=h_2(16),\ h_2(12)=0.5$
$H_3(z)$	$h_3(0)=h_3(28),\ h_3(4)=h_3(24),\ h_3(8)=h_3(20),\ h_3(12)=h_3(16),$ $h_3(14)=0.5$
$H_4(z)$	$h_4(0)=h_4(12),\ h_4(4)=h_4(8),\ h_4(6)=0.5$
$H_5(z)$	$h_5(0)=h_5(10),\ h_5(2)=h_5(8),\ h_5(4)=h_5(6),\ h_5(5)=0.5$
$H_6(z)$	$h_6(0)=h_6(6),\ h_6(2)=h_6(4),\ h_6(5)=0.5$
$H_7(z)$	$h_7(0)=h_7(30),\ h_7(6)=h_7(24),\ h_7(12)=h_7(18),\ h_7(15)=0.5$
$H_8(z)$	$h_8(0)=h_8(2),\ h_8(1)=0.5$

จากตารางแสดงความความสัมพันธ์ของค่าผลตอบสนองต่ออิมพัลส์ข้างต้นจะทำการกำหนดการเรียกค่า h(n) ใหม่เพื่อจะเป็นประโยชน์ในการลดรูปสมการผลต่างที่จะต้องสร้างขึ้นมาเพื่อใช้ในการคำนวณ ในที่นี้จะขอยกตัวอย่างการออกแบบหาค่า h(n) สำหรับฟิลเตอร์ $H_1(z)$ โดยเริ่มต้นพบว่าจะมีค่าผลตอบสนองที่จะต้องทำการหาค่าหรือนำมาใช้สร้างสมการผลต่างของฟิลเตอร์ $H_1(z)$ ทั้งหมด 5 ค่า แต่เนื่องจาก $h_1(0)$ กับ $h_1(48)$ มีค่าเท่ากัน เช่นเดียวกันกับ $h_1(16)$ กับ $h_1(32)$ จึงทำการกำหนดตัวแปรขึ้นมาแทนค่า $h_1(0)$ กับ $h_1(48)$ เพื่อการลดรูปการเขียนสมการผลต่าง ส่วน h(n) ค่าที่เหลือคือค่าของ $h_1(24)$ ซึ่งเป็นค่าตรงกลางความสมมาตรก็กำหนดตัวแปรใหม่มาแทนด้วยเช่นกัน โดยจากวิธีการข้างต้นจะทำให้สามารถกำหนดค่าตัวแปรที่ใช้แทนค่าผลตอบสนองตัวเดิมสำหรับฟิลเตอร์ $H_1(z)$ ได้คือ $h_1(0)$ กับ $h_1(48)$ จะเขียนแทนด้วย $h_1(0)$ ต่อจากนั้น $h_1(16)$ กับ $h_1(32)$ แทนด้วย $h_1(1)$ และสุดท้าย $h_1(24)$ จะแทนด้วย $h_1(3)$ สำหรับฟิลเตอร์ H(z) ตัวถัดไปที่เหลือทั้งหมดก็จะใช้วิธีการกำหนดเช่นเดียวกันนี้ ดังนั้นสุดท้ายจะสามารถเขียนตารางใหม่ได้ดังตารางประกอบ 3-2 ก่อนจะนำตัวแปรใหม่เหล่านี้ไปใช้ประโยชน์ในการสร้างสมการผลต่างเพื่อใช้หาค่า h(n) ในลำดับถัดไปนั่นเอง

ตารางที่ 3-2 การกำหนดค่าผลตอบสนองต่ออิมพัลส์

$H_1(z)$	$[\ h_1(0) = h_1(48) \boxplus h_1(0)\ ,\ [\ h_1(16) = h_1(32) \boxplus h_1(1)\ ,\ h_1(24) = h_1(2)$
$H_2(z)$	$[\ h_2(0) = h_2(24) \boxplus h_2(0)\ ,\ [\ h_2(8) = h_2(16) \boxplus h_2(1)\ ,\ h_2(12) = h_2(2)$
$H_3(z)$	$[\ h_3(0) = h_3(28) \boxplus h_3(0)\ ,\ [\ h_3(4) = h_3(24) \boxplus h_3(1)\ ,\ [\ h_3(8) = h_3(20) \boxplus h_3(2)\ ,$ $[\ h_3(12) = h_3(16) \boxplus h_3(3)\ ,\ h_3(14) = h_3(4)$
$H_4(z)$	$[\ h_4(0) = h_4(12) \boxplus h_4(0)\ ,\ [\ h_4(4) = h_4(8) \boxplus h_4(1)\ ,\ h_4(6) = h_4(2)$
$H_5(z)$	$[\ h_5(0) = h_5(10) \boxplus h_5(0)\ ,\ [\ h_5(2) = h_5(8) \boxplus h_5(1)\ ,\ [\ h_5(4) = h_5(6) \boxplus h_5(2)\ ,\ h_5(5) = h_3(3)$
$H_6(z)$	$[\ h_6(0) = h_6(6) \boxplus h_6(0)\ ,\ [\ h_6(2) = h_6(4) \boxplus h_6(1)\ ,\ h_6(5) = h_6(2)$
$H_7(z)$	$[\ h_7(0) = h_7(30) \boxplus h_7(0)\ ,\ [\ h_7(6) = h_7(24) \boxplus h_7(1)\ ,\ [\ h_7(12) = h_7(18) \boxplus h_7(2)\ ,\ h_7(15) = h_7(3)$
$H_8(z)$	$[\ h_8(0) = h_8(2) \boxplus h_8(0)\ ,\ h_8(1) = h_8(1)$

เมื่อนำสมการทั้ง (3-1), (3-2) และ (3-3) รวมทั้งตารางความสัมพันธ์ของค่า h(n) ข้างต้น มาสร้างสมการผลต่างสำหรับฟิลเตอร์ทั้ง 8 ตัวได้สมการผลต่างแสดงด้วยภาพประกอบ 3-1

$H_1(z)$: $y_1(n) = [x(n-0) + x(n-48) \boxtimes h_1(0) + [x(n-16) + x(n-32) \boxtimes h_1(1) + x(n-24) * h_1(2)$

$y_{1c}(n) = x(n-24) - y_1(n)$

$H_2(z)$: $y_2(n) = [x(n-0) + x(n-24) \boxtimes h_2(0) + [x(n-8) + x(n-16) \boxtimes h_2(1) + x(n-12) * h_2(2)$

$y_{2c}(n) = x(n-12) - y_2(n)$

$H_3(z)$: $y_3(n) = [x(n-0) + x(n-28) \boxtimes h_3(0) + [x(n-4) + x(n-24) \boxtimes h_3(1) +$

$[x(n-8) + x(n-20) \boxtimes h_3(2) + [x(n-12) + x(n-16) \boxtimes h_3(3) + x(n-14) * h_3(4)$

$y_{3c}(n) = x(n-14) - y_3(n)$

$H_4(z)$: $y_4(n) = [x(n-0) + x(n-12) \boxtimes h_4(0) + [x(n-4) + x(n-8) \boxtimes h_4(1) + x(n-6) * h_4(2)$

$y_{4c}(n) = x(n-6) - y_4(n)$

$H_5(z)$: $y_5(n) = [x(n-0) + x(n-10) \boxtimes h_5(0) + [x(n-2) + x(n-8) \boxtimes h_5(1) + [x(n-4) + x(n-6) \boxtimes h_5(2) + x(n-5) * h_5(3)$

$y_{5c}(n) = x(n-5) - y_5(n)$

$H_6(z)$: $y_6(n) = [x(n-0) + x(n-6) \boxtimes h_6(0) + [x(n-2) + x(n-4) \boxtimes h_6(1) + x(n-3) * h_6(2)$

$y_{6c}(n) = x(n-3) - y_6(n)$

$H_7(z)$: $y_7(n) = [x(n-0) + x(n-30) \boxtimes h_7(0) + [x(n-8) + x(n-24) \boxtimes h_7(1) + [x(n-12) + x(n-18) \boxtimes h_7(2) + x(n-15) * h_7(3)$

$y_{7c}(n) = x(n-15) - y_7(n)$

$H_8(z)$: $y_8(n) = [x(n-0) + x(n-2) \boxtimes h_8(0) + x(n-1) * h_8(1)$

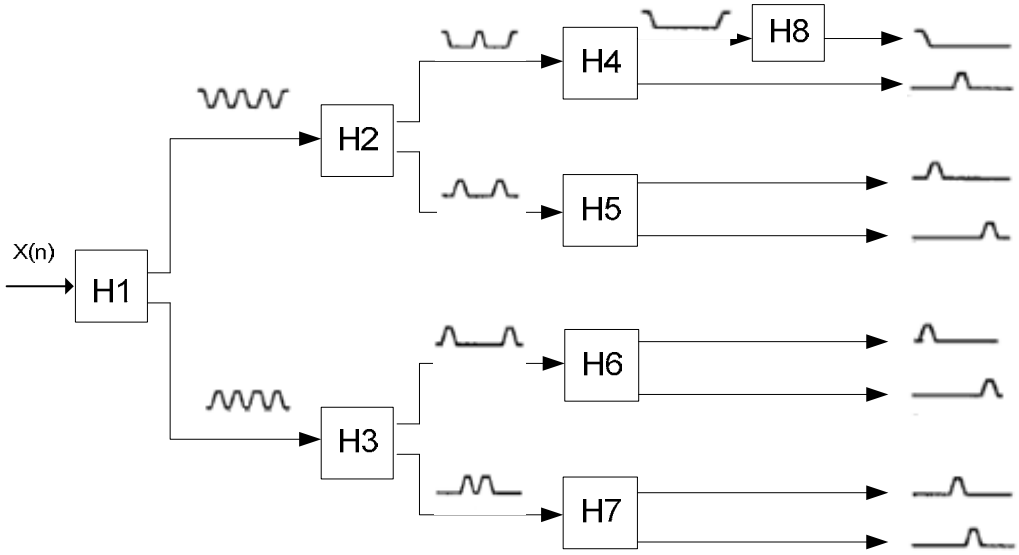
ภาพประกอบ 3-1 สมการความสัมพันธ์ระหว่างอินพุตและเอาท์พุตของฟิลเตอร์

ขั้นตอนต่อไปของการออกแบบคือการออกแบบฟิลเตอร์ด้วย MATLAB เพื่อหาค่า h(n) ทั้งหมดของฟิลเตอร์ย่อยแต่ละตัว โดยการออกแบบวงจรฟิลเตอร์สำหรับงานวิจัยนี้จะออกแบบวงจรฟิลเตอร์แบบ Interpolated FIR filters ซึ่งเป็นโครงสร้างของฟิลเตอร์ที่มีลำดับเท่ากับ N-1 และมีการเติมจุดที่มีค่าเป็นศูนย์เข้าไปในสัญญาณ x(n) ซึ่งจะส่งผลให้สเปกตรัมของสัญญาณมีจำนวนจุดมากขึ้น ซึ่งการเติมศูนย์ช่วยให้มองเห็นรูปร่างของสัญญาณได้ละเอียด และชัดเจนขึ้น แต่ในทางทฤษฎีแล้ว ไม่ได้เป็นการเพิ่มข้อมูลใด ๆ ให้แก่สัญญาณเลย

การแทรกค่าศูนย์ที่กล่าวมานั้นจะทำการแทรกค่าศูนย์ด้วยพารามิเตอร์ L นั่นคือวงจรจะทำการแทรกค่าศูนย์เป็นจำนวน L-1 ค่าเข้าไประหว่างค่าอิมพัลส์ของวงจรจะทำให้ได้ค่าอิมพัลส์ของวงจรชุดใหม่นั่นเอง ซึ่งสมการผลต่างที่ใช้ในการออกแบบสามารถแสดงได้ดังสมการ (3-4)

$$H(z) = \sum_{k=0}^{n-1} h(k) z^{-k} = \sum_{m=0}^{n-1} h(m) z^{-mL} = H(z^{L}) \qquad (3\text{-}4)$$

วงจรฟิลเตอร์แบงค์ที่ใช้ในงานวิจัยนี้จะมีรูปร่างของสัญญาณทั้งด้านเข้าและด้านออกของฟิลเตอร์แต่ละตัวไว้ โดยสังเกตจากภาพประกอบ 3-2



ภาพประกอบ 3-2 สัญญาณด้านเข้าและด้านออกของฟิลเตอร์ภายในฟิลเตอร์แบงค์

ข้อมูลข้างต้นที่กล่าวมาทั้งหมดนั้นจะนำไปใช้ประโยชน์ในการหาค่าอิมพัลส์ h(n) ของวงจรฟิลเตอร์โดยที่จะทำการออกแบบฟิลเตอร์ด้วยโปรแกรม MATLAB ซึ่งจะเป็นการเขียนชุดคำสั่งเพื่อให้โปแกรมทำการสร้างฟิลเตอร์ตามที่กล่าวไว้ เมื่อโปรแกรมทำการสร้างฟิลเตอร์ตามที่กำหนดแล้วนั้น จะสามารถทราบค่า h(n) ที่ต้องการได้ เมื่อได้ค่า h(n) มาแล้วนั้นจะนำค่าที่ได้ไปเขียนสมการการทำงานของวงจรฟิลเตอร์แบงค์ตามภาพประกอบที่ 3-1 เพื่อจะได้นำสมการนั้นไปเขียนวงจรรวมเพื่อใช้ในการออกแบบฟิลเตอร์แบงค์ในขั้นตอนต่อไป

3.2 การออกแบบวงจรฟิลเตอร์แบงค์

หลังจากที่ทราบค่า h(n) และได้สมการในการประมวลผลของวงจรฟิลเตอร์มาแล้วนั้นในขั้นตอนต่อไปจะเป็นการออกแบบวงจรฟิลเตอร์เพื่อวิเคราะห์และนำเสนอการออกแบบวงจรด้วยระเบียบวิธีที่ต้องการนำเสนอในงานวิจัยนี้นั่นเอง

โดยที่จะทำการออกแบบวงจรฟิลเตอร์ด้วยโปรแกรม 2 โปรแกรม คือ ออกแบบด้วยโปรแกรม MATLAB และออกแบบด้วยโปรแกรม Xilinx วัตถุประสงค์ที่ทำการออกแบบดังที่กล่าวไปนั้นก็เพื่อจะนำผลการทดสอบวงจรที่ได้มาเปรียบเทียบกัน โดยการออกแบบวงจรทั้งสองวิธีที่กล่าวนั้นจะออกแบบวงจรโดยให้วงจรมีลำดับการประมวลผลแบบเดียวกันทั้งนี้จะยึดลำดับการประมวลผลตามวงจรฟิลเตอร์แบงค์ในงานวิจัยที่นำมาอ้างอิงดังที่กล่าวไว้ในหัวข้อ 2.2 นั่นเอง

3.2.1 การออกแบบวงจรฟิลเตอร์แบงค์ด้วยโปรแกรม MATLAB

การออกแบบวงจรฟิลเตอร์ด้วยโปรแกรม MATLAB นั้นจะเป็นการเขียนชุดคำสั่งซึ่งในที่นี้ก็คือการเขียนสมการความสัมพันธ์ระหว่างอินพุตและเอาท์พุตของฟิลเตอร์ตามลำดับการทำงานของวงจรตั้งแต่ ฟิลเตอร์ H1 ไปจนถึง ฟิลเตอร์ H8 ดังที่เคยแสดงด้วยภาพประกอบ 3-1 ในหัวข้อการหาค่า h(n) โดยเมื่อเขียนสมการความสัมพันธ์ของทั้งวงจรเรียบร้อยแล้วนั้นจะทำการป้อนค่าอินพุตหรือสัญญาณเสียงด้านเข้า เพื่อที่จะรับผลเอาท์พุตจากวงจรจากนั้นจะนำค่าผลลัพธ์ที่ได้ไปเปรียบเทียบกับค่าอินพุตและนำไปเปรียบเทียบกับเอาท์พุตที่ได้จากวงจรฟิลเตอร์แบงค์ซึ่งออกแบบด้วยโปรแกรม Xilinx ต่อไป

3.2.2 การออกแบบวงจรฟิลเตอร์แบงค์ด้วยโปรแกรม Xilinx

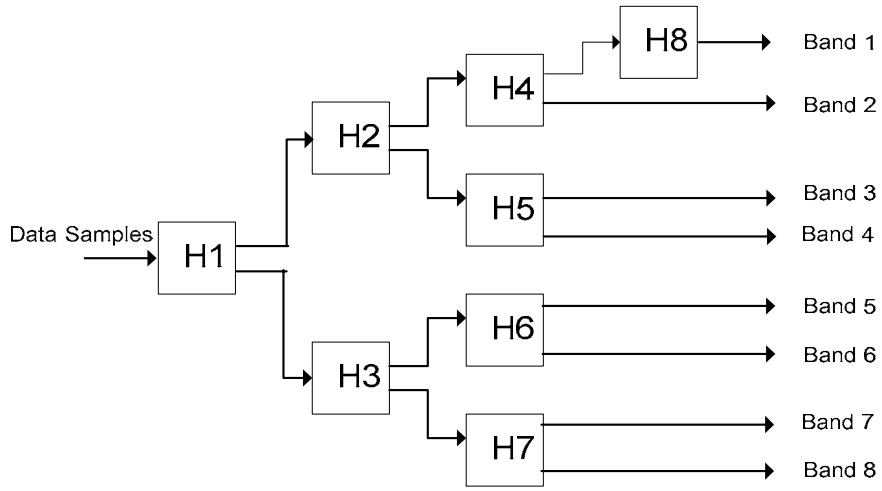
เมื่อทำการออกแบบวงจรฟิลเตอร์แบงค์ด้วยโปรแกรม MATLAB เรียบร้อยแล้วนั้น ลำดับถัดไปจะเป็นการออกแบบวงจรฟิลเตอร์แบงค์ด้วยโปรแกรม Xilinx ทั้งนี้ในการออกแบบ

ในขั้นตอนนี้จะมีรายละเอียดในการออกแบบมากกว่าการออกแบบด้วยโปรแกรม MATLAB เนื่องจากการออกแบบด้วย Xilinx ซึ่งคือการสร้างลอจิกเกตแทนสมการความสัมพันธ์ระหว่างค่าอินพุตและค่าเอาท์พุตของวงจรลงบน FPGA จะต้องคำนึงถึงขนาดของเนื้อที่หรือทรัพยากรบน FPGA ที่ต้องใช้ในการออกแบบด้วย เริ่มแรกนั้นทำการออกแบบวงจรฟิลเตอร์แบงค์ให้มีขั้นตอนการทำงานตรงกับการทำงานของวงจรฟิลเตอร์ที่ได้ศึกษาจากงานวิจัยที่ใช้อ้างอิง จากนั้นนำผลการออกแบบมาพิจารณาถึงการใช้ทรัพยากร(ตัวบวก,ตัวคูณและตัวหน่วง) ภายใน FPGA ที่จะถูกนำไปใช้เพื่อสร้างวงจรฟิลเตอร์แบงค์ โดยผลจากการออกแบบตรงนี้พบว่าเมื่อทำการออกแบบวงจรฟิลเตอร์แบงค์บน FPGA ขนาดหนึ่งที่มีอยู่โดยระเบียบวิธีทั่วไปนั้นทรัพยากรภายใน FPGA ที่มีอยู่ไม่เพียงพอต่อการนำมาใช้ในการออกแบบและการที่จะแก้ไขปัญหาด้วยการหาซื้อ FPGA ที่มีขนาดใหญ่ขึ้นเพื่อให้เพียงพอต่อการใช้ในการออกแบบก็จะเป็นการสิ้นเปลืองค่าใช้จ่ายมากขึ้น น่าจะมีระเบียบวิธีในการออกแบบที่สามารถออกแบบวงจรบน FPGA ตัวเดิมที่มีได้ จากเหตุผลที่กล่าวมาข้างต้นนำมาซึ่งการเสนอระเบียบวิธีที่ใช้ในการออกแบบวงจรฟิลเตอร์แบงค์สำหรับงานวิจัยนี้

การออกแบบวงจรขนาดใหญ่บนชิพที่มีพื้นที่จำกัดดังเช่น FPGA มีความจำเป็นที่จะต้องจัดการกับทรัพยากรที่ต้องใช้ในการออกแบบวงจรให้เหมาะสมกับขนาดของ FPGA งานวิจัยนี้จึงนำเสนอระเบียบวิธีการใช้ทรัพยากรร่วมกันแบบลำดับชั้นในการออกแบบวงจรฟิลเตอร์แบงค์ซึ่งเป็นวงจรที่มีขนาดใหญ่ทรัพยากรที่ต้องใช้ก็คือตัวบวก,ตัวคูณและดีเลย์ซึ่งเป็นองค์ประกอบพื้นฐานของวงจรฟิลเตอร์นั่นเอง โดยที่ระเบียบวิธีที่นำเสนอในการออกแบบวงจรดังที่กล่าวไปข้างต้นนั้นสามารถแบ่งย่อยออกเป็น 2 วิธี คือ

1. การใช้ทรัพยากรร่วมระหว่างบล็อก (Inter-Block Sharing)
2. การใช้ทรัพยากรร่วมภายในบล็อก (Intra-Block Sharing)

ก่อนที่จะกล่าวถึงความหมายและลักษณะการออกแบบด้วยระเบียบวิธี 2 วิธีดังกล่าวนั้นจะย้อนกลับไปที่วงจรฟิลเตอร์แบงค์ที่นำมาใช้ในงานวิจัยนี้ โดยจะนำโครงสร้างของฟิลเตอร์แบงค์ และสมการในการทำงานของฟิลเตอร์ทั้งหมดภายในวงจรรวมมาแสดงให้เห็นอีกครั้งหนึ่งซึ่งแสดงไว้ด้วยภาพประกอบ 3-3 และภาพประกอบ 3-4



ภาพประกอบ 3-3 วงจรดิจิตอลฟิลเตอร์แบงค์

$H_1(z)$: $y_1(n) = [x(n-0) + x(n-48)] * h_1(0) + [x(n-16) + x(n-32)] * h_1(1) + x(n-24) * h_1(2)$

$y_{1c}(n) = x(n-24) - y_1(n)$

$H_2(z)$: $y_2(n) = [x(n-0) + x(n-24)] * h_2(0) + [x(n-8) + x(n-16)] * h_2(1) + x(n-12) * h_2(2)$

$y_{2c}(n) = x(n-12) - y_2(n)$

$H_3(z)$: $y_3(n) = [x(n-0) + x(n-28)] * h_3(0) + [x(n-4) + x(n-24)] * h_3(1) +$

$[x(n-8) + x(n-20)] * h_3(2) + [x(n-12) + x(n-16)] * h_3(3) + x(n-14) * h_3(4)$

$y_{3c}(n) = x(n-14) - y_3(n)$

$H_4(z)$: $y_4(n) = [x(n-0) + x(n-12)] * h_4(0) + [x(n-4) + x(n-8)] * h_4(1) + x(n-6) * h_4(2)$

$y_{4c}(n) = x(n-6) - y_4(n)$

$H_5(z)$: $y_5(n) = [x(n-0) + x(n-10)] * h_5(0) + [x(n-2) + x(n-8)] * h_5(1) + [x(n-4) + x(n-6)] * h_5(2) + x(n-5) * h_5(3)$

$y_{5c}(n) = x(n-5) - y_5(n)$

$H_6(z)$: $y_6(n) = [x(n-0) + x(n-6)] * h_6(0) + [x(n-2) + x(n-4)] * h_6(1) + x(n-3) * h_6(2)$

$y_{6c}(n) = x(n-3) - y_6(n)$

$H_7(z)$: $y_7(n) = [x(n-0) + x(n-30)] * h_7(0) + [x(n-8) + x(n-24)] * h_7(1) + [x(n-12) + x(n-18)] * h_7(2) + x(n-15) * h_7(3)$

$y_{7c}(n) = x(n-15) - y_7(n)$

$H_8(z)$: $y_8(n) = [x(n-0) + x(n-2)] * h_8(0) + x(n-1) * h_8(1)$

ภาพประกอบ 3-4 โครงสร้างของฟิลเตอร์ภายในฟิลเตอร์แบงค์

จากภาพประกอบข้างต้นเมื่อนำมาพิจารณาโครงสร้างของฟิลเตอร์ย่อยแต่ละตัวภายในฟิลเตอร์แบงค์แล้วนั้น พบว่าสามารถทำการจัดกลุ่มโครงสร้างของฟิลเตอร์ภายในฟิลเตอร์แบงค์ตามลักษณะฟังก์ชันภายในที่เหมือนกันได้เป็น 4 แบบ ดังนี้

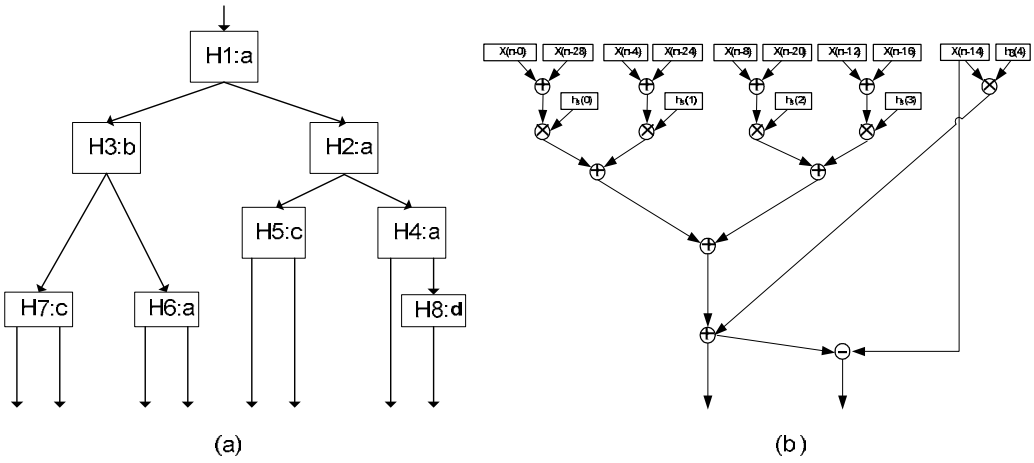
โครงสร้างแบบ a จะสอดคล้องกับ Block H1 ,H2 ,H4 และ H6

โครงสร้างแบบ b จะสอดคล้องกับ Block H3

โครงสร้างแบบ c จะสอดคล้องกับ Block H5 และ H7

โครงสร้างแบบ d จะสอดคล้องกับ Block H8

เมื่อนำโครงสร้างที่แบ่งกลุ่มแล้วและสมการการประมวลผลของฟิลเตอร์แต่ละตัวมาเขียนกราฟกระแสข้อมูล (Data Flo Graph; DFG) ซึ่งแสดงโครงสร้างลำดับการประมวลผลโดยเริ่มตั้งแต่รับค่าอินพุตจนกระทั่งได้ค่าเอาท์พุต นั้นสามารถแสดงได้ดังภาพประกอบ 3-5



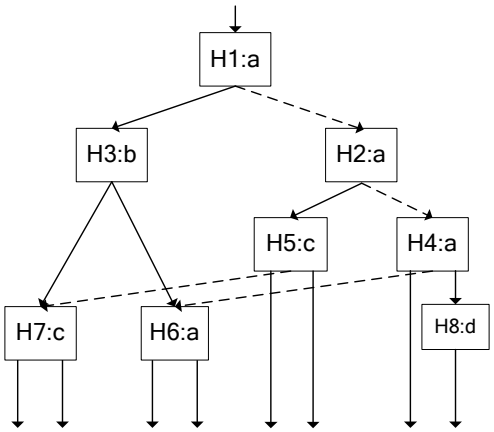
ภาพประกอบ 3-5 DFG ของวงจรฟิลเตอร์แบงค์ (a) DFG ลำดับบน และ (b) DFG ลำดับล่าง (block H3)

3.2.2.1 การใช้ทรัพยากรร่วมระหว่างบล็อก (Inter-Block Sharing)

การใช้ทรัพยากรร่วมระหว่างบล็อก (Inter-Block Sharing) คือการใช้ทรัพยากรร่วมกันระหว่างบล็อกในระดับบนสุดของ DFG โดยที่การใช้ทรัพยากรร่วมกันระหว่างบล็อกนี้จะถูกนำมาพิจารณาเมื่อฟังก์ชันภายในของทั้ง 2 บล็อกนั้นเหมือนกันและนอกจากนี้หากบล็อก 2 บล็อกหรือหลายๆ บล็อกมีโครงสร้างของ DFG ที่เหมือนกันนั้นก็สามารถทำการออกแบบโดยวิธีนี้ได้เช่นเดียวกัน จากนิยามหรือความหมายดังกล่าวเมื่อนำมาพิจารณาใช้กับวงจรฟิลเตอร์แบงค์นั้น

พบว่ากลุ่มโครงสร้างฟิลเตอร์ที่สามารถออกแบบโดยเลือกใช้ระเบียบวิธีการใช้ทรัพยากรร่วมกันก็คือ โครงสร้างแบบ a และโครงสร้างแบบ c เพราะมีฟังก์ชันการทำงานภายในที่เหมือนกัน

ดังนั้นเมื่อทำการแสดงลำดับการประมวลผลของวงจรฟิลเตอร์แบงค์ที่ทำการออกแบบโดยการใช้ระเบียบวิธีการใช้ทรัพยากรร่วมระหว่างบล็อกเรียบร้อยแล้วนั้น สามารถเขียนวงจรดังแสดงด้วยภาพประกอบ 3-6 โดยเส้นประที่แสดงในภาพประกอบนั้นเป็นการแสดงว่าบล็อกที่อยู่ระหว่างเส้นประนั้นถูกออกแบบด้วยการใช้ทรัพยากรร่วมกันระหว่างบล็อก โดยการเริ่มทำงานของบล็อกที่ตามมา (ด้านหัวลูกศร) จะเริ่มได้หลังจากบล็อกที่มาก่อน (หางลูกศร) ทำงานเสร็จสมบูรณ์

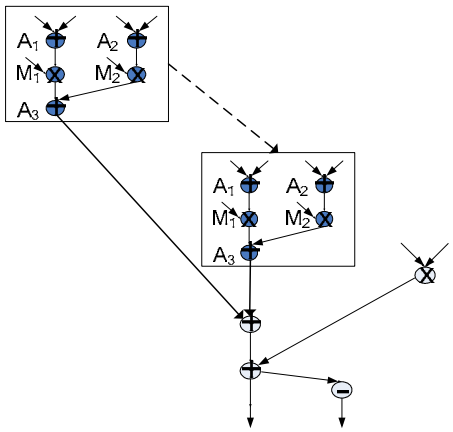


ภาพประกอบ 3-6 ฟิลเตอร์แบงค์ที่ออกแบบด้วยการใช้ทรัพยากรร่วมกันระหว่างบล็อก

3.2.2.2 การใช้ทรัพยากรร่วมภายในบล็อก (Intra-Block Sharing)

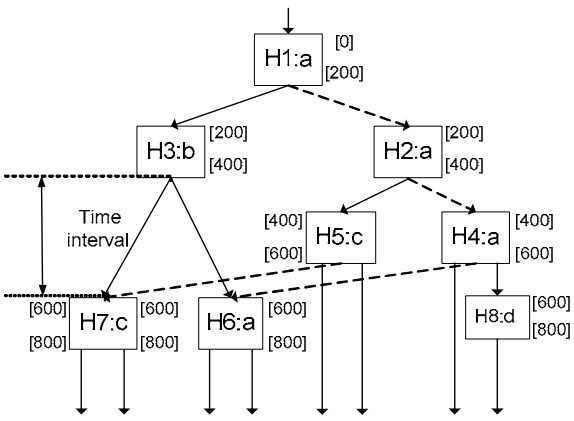
การใช้ทรัพยากรร่วมภายในบล็อก (Intra-Block Sharing) คือการใช้ทรัพยากรร่วมกันภายในบล็อคทั้งนี้เพื่อหลีกเลี่ยงความซับซ้อนของการเชื่อมต่อสัญญาณภายในชิพและบล็อกที่มีการใช้ทรัพยากรแบบ Inter-Block Sharing ไปแล้วจะไม่นำมาพิจารณาการออกแบบด้วยระเบียบวิธี Intra-Block Sharing อีก ดังนั้นสำหรับวงจรฟิลเตอร์แบงค์ภายในงานวิจัยนี้จะมีเพียงบล็อกโครงสร้างแบบ b และแบบ d เท่านั้นที่สามารถจะนำมาพิจารณาการทำ Intra-Block Sharing ได้

พิจารณาภาพประกอบ 3-7 ซึ่งคล้ายคลึงกับวิธีของ Inter-Block Sharing และเมื่อพิจารณาโครงสร้างของ H3 พบว่าจะมีวงจรบวก 3 ตัว คือ A1, A2 และ A3, วงจรคูณ 2 ตัวคือ M1 และM2 ที่จะถูกนำมาออกแบบโดยการใช้ทรัพยากรร่วมภายในบล็อก



ภาพประกอบ 3-7 โครงสร้างภายในของฟิลเตอร์ H3

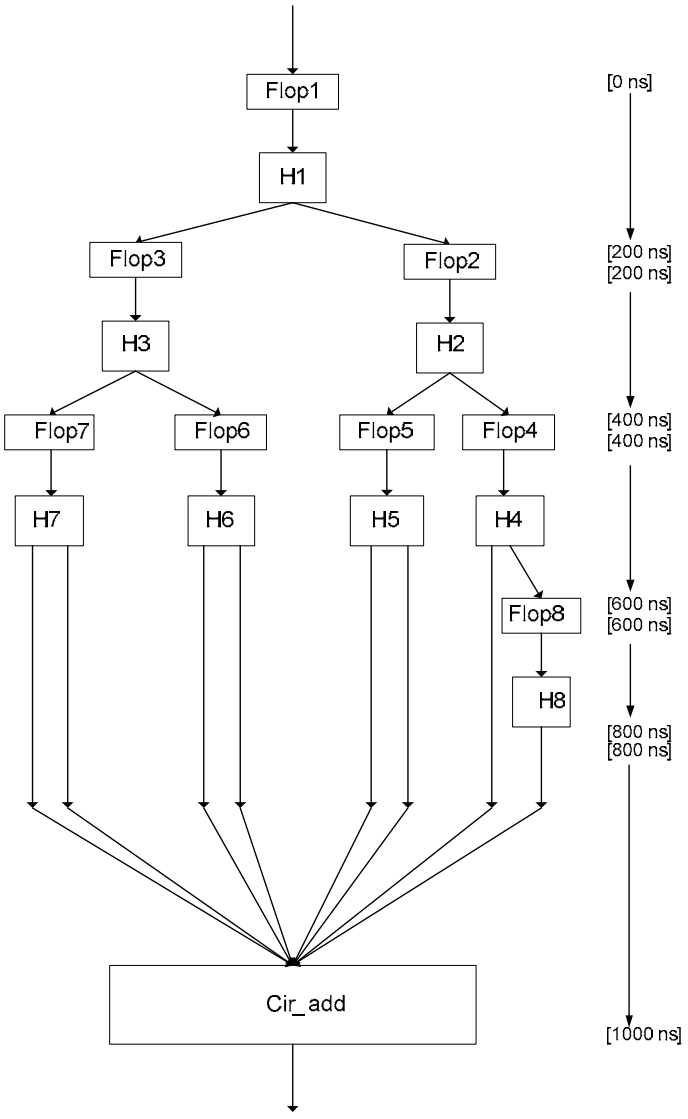
วิธีการออกแบบด้วยระเบียบวิธีดังกล่าวนี้นอกจากการพิจารณาโครงสร้างภายในแล้วนั้นจะต้องพิจารณาเวลาที่จะต้องสอดคล้องกับการเริ่มทำงานของวงจรรวมส่วนถัดไปด้วย ดังแสดงในภาพประกอบ 3-8 พบว่าเมื่อฟิลเตอร์ H3 ทำงานเสร็จแล้วนั้นเอาท์พุตที่ได้จะถูกนำไปประมวลผลในขั้นตอนต่อไปจะต้องรอให้ฟิลเตอร์ H4 และH5 ทำงานเสร็จก่อนเพราะฟิลเตอร์ H6 และ H7 นั้นได้ถูกออกแบบด้วยระเบียบวิธีการใช้ทรัพยากรร่วมกันระหว่างบล็อก ดังนั้นจะมีเวลาว่างเกิดขึ้น ซึ่งคาบสัญญาณนาฬิกาที่แสดงในภาพประกอบ 3-8 คือคาบสัญญาณนาฬิกาที่ได้จากการสังเคราะห์วงจรแบบที่ไม่การใช้ทรัพยากรร่วมกันซึ่งมีค่า 200 ns ดังนั้นพบว่าสามารถพิจารณาการใช้ทรัพยากรร่วมกันสำหรับโครงสร้างแบบ b ได้และสำหรับโครงสร้างแบบ d นั้นจะไม่มีการพิจารณาการใช้ทรัพยากรร่วมกัน เพราะโครงสร้างภายในที่ไม่ซับซ้อน



ภาพประกอบ 3-8 เวลาที่นำมาพิจารณาสำหรับการออกแบบการใช้ทรัพยากรร่วมกันภายในบล็อกสำหรับวงจรดิจิตอลฟิลเตอร์แบงค์

ระเบียบวิธีในการออกแบบที่กล่าวไปข้างต้นทั้งสองแบบจะนำไปใช้เมื่อทำการออกแบบวงจรฟิลเตอร์แบงค์ด้วยโปรแกรม Xilinx เพื่อยืนยันว่าด้วยวิธีดังกล่าวนี้สามารถลดจำนวนทรัพยากรที่ต้องใช้ภายใน FPGA ได้ โดยเริ่มต้นนั้นจะทำการออกแบบวงจรฟิลเตอร์แบงค์ด้วยวิธีทั่วไปแต่ทั้งนี้ในการออกแบบจะคำนึงถึงการใช้ทรัพยากรให้ประหยัดที่สุดและวงจรสามารถทำงานได้ตรงกับวงจรที่ออกแบบด้วยโปรแกรม MATLAB ด้วย

(หมายเหตุ : เวลาที่แสดงเป็นเวลาการทำงานของวงจรทีละส่วนตามลำดับโดยแสดงเวลาเมื่อเริ่มป้อนอินพุตและเวลาที่ได้ค่าเอาท์พุตออกมา และกำหนดค่าดังกล่าวเป็นคาบสัญญาณนาฬิกาให้แก่วงจรในการทดสอบวงจร)



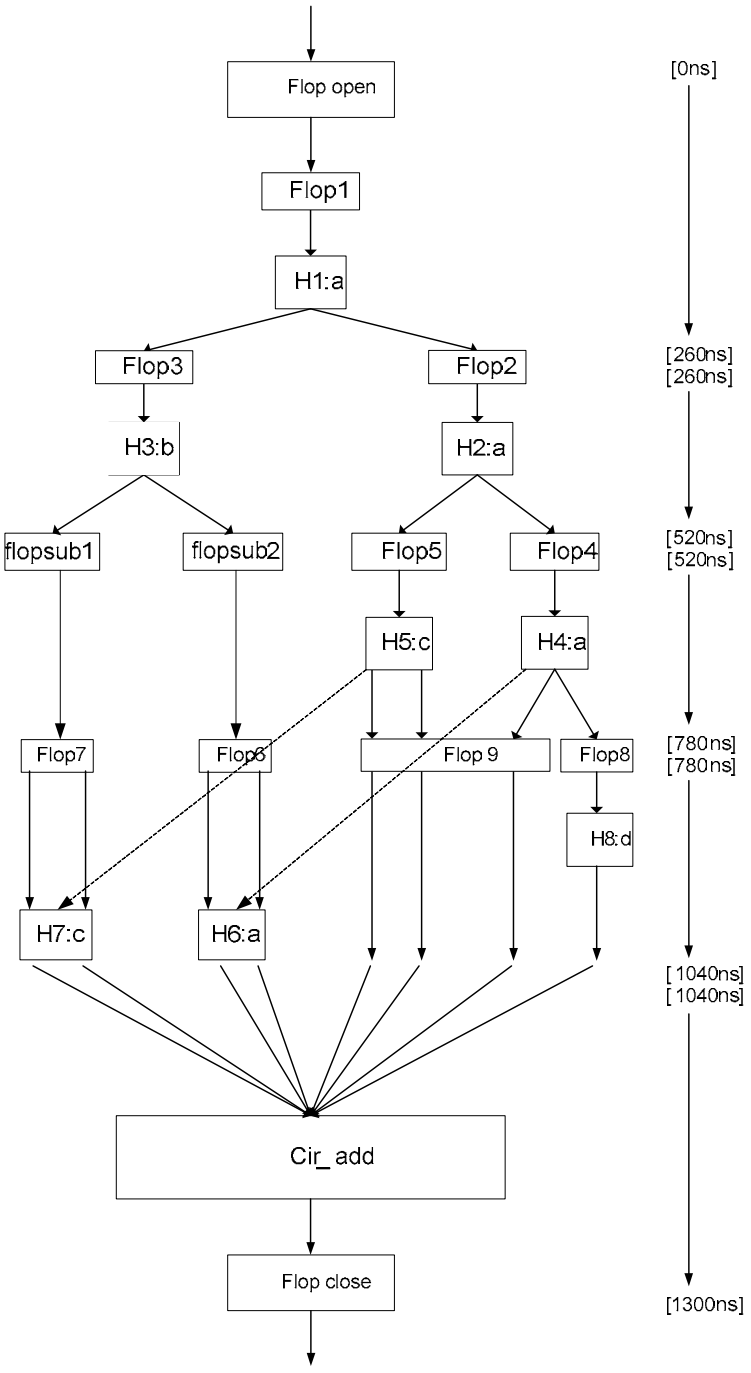
ภาพประกอบ 3-9 วงจรฟิลเตอร์แบงค์ที่ออกแบบโดยไม่มีการใช้ทรัพยากรร่วมกัน

จากภาพประกอบด้านบนสามารถอธิบายลำดับการประมวลผลได้ดังนี้คือเริ่มต้นด้วยการที่มีข้อมูลอินพุตป้อนไปยัง Flop1 ซึ่งเป็นรีจิสเตอร์พักข้อมูลที่มีสัญญาณควบคุมการทำงานคือ enable เมื่อสัญญาณ enable มีค่าเป็น0 ข้อมูลก็ยังคงถูกพักไว้จนกระทั่งสัญญาณ enable มีค่าเป็น 1 ข้อมูลจะถูกส่งออกไปยังส่วนถัดไปซึ่งในที่นี้คือบล็อกฟิลเตอร์ H1 สำหรับภายใน Flop1 นั้นเป็นวงจรรีจิสเตอร์เลื่อน (Shift register) ขนาด 32 บิต 49 สเตจ

$$H_1(z) : y_1(n) = [x(n-0) + x(n-48) * h_1(0) + [x(n-16) + x(n-32) * h_1(1) + x(n-24) * h_1(2) \quad (3\text{-}5)$$

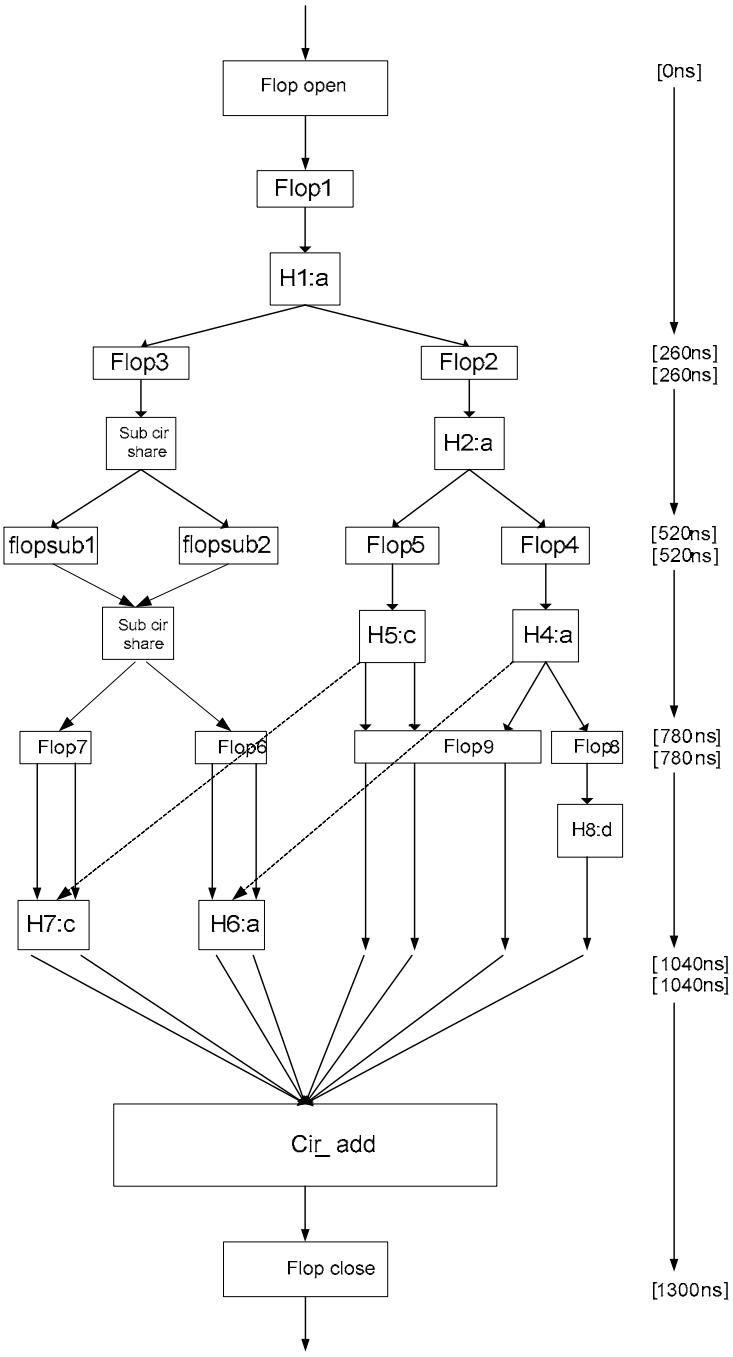
เมื่อพิจารณาสมการการประมวลผลของ H1 ดังสมการที่ (3-5) สมการการทำงานแสดงให้เห็นว่าจะมีการใช้ข้อมูลอินพุตที่ตำแหน่งเวลาก่อนหน้าไป 48 ตำแหน่งด้วย ในการออกแบบจึงออกแบบให้ภายใน flop1 นั้นบรรจุรีจิสเตอร์เลื่อนไว้จำนวน 49 ตัวและจะดึงข้อมูลจากรีจิสเตอร์ในสเตจที่ n-0, n-16, n-24 n-32 และ n-48 ไปใช้ในการคำนวณหาค่าเอาท์พุตของฟิลเตอร์ H1 ด้วยวิธีเดียวกันนี้การออกแบบ Flop ตัวอื่นๆในวงจรก็จะนำสมการของฟิลเตอร์ตำแหน่งที่อยู่หลัง Flop นั้นๆมาพิจารณาเช่น Flop2 ก็จะมี รีจิสเตอร์เลื่อนขนาด 25 สเตจ , Flop3 มีรีจิสเตอร์เลื่อนขนาด 29 สเตจ เป็นต้น จากนั้นในส่วนบล็อกที่เป็น H1 , H2 จนถึง H8 ก็คือฟิลเตอร์ตัวต่างๆนั่นเองซึ่งบล็อกเหล่านี้ภายในก็จะประกอบไปด้วยตัวบวก ตัวคูณและตัวลบ จำนวนขององค์ประกอบแต่ละตัวสามารถทราบได้จากสมการของฟิลเตอร์ตัวนั้นๆ เช่น H1 จะประกอบด้วยตัวบวก 4 ตัว ตัวคูณ 3 ตัว และตัวลบ 1 ตัว เป็นต้น ในลำดับสุดท้ายของวงจรก็จะเป็นการนำสัญญาณเอาท์พุตทั้ง 8 ค่าที่ได้คูณด้วยสัมประสิทธิ์ที่เป็น 1 ก่อนจะนำมาบวกกันทั้งหมดเพื่อเป็นค่าเอาท์พุตของวงจรฟิลเตอร์แบงก์เพียงค่าเดียว

การออกแบบครั้งแรกทำการออกแบบโดยไม่มีการใช้ทรัพยากรร่วมกันดังนั้นการออกแบบวงจรลำดับต่อมาจะออกแบบโดยมีการใช้การใช้ทรัพยากรร่วมกันโดยใช้ระเบียบวิธีการใช้ทรัพยากรร่วมกันระหว่างบล็อกพิจารณาในการออกแบบเพียงวิธีเดียวก่อน ซึ่งเมื่อออกแบบวงจรฟิลเตอร์ด้วยวิธีดังที่กล่าวไปนั้นวงจรที่ได้สามารถแสดงด้วยภาพประกอบ 3-10



ภาพประกอบ 3-10 วงจรที่ออกแบบด้วยระเบียบวิธีการใช้ทรัพยากรร่วมระหว่างบล็อก

ต่อจากนั้นในลำดับสุดท้ายก็จะเป็นการนำระเบียบวิธีการใช้ทรัพยากรร่วมกันภายในบล็อกและระหว่างบล็อกมาใช้ในการออกแบบทั้งคู่ซึ่งแสดงวงจรหลังจากออกแบบด้วยภาพประกอบ 3-11



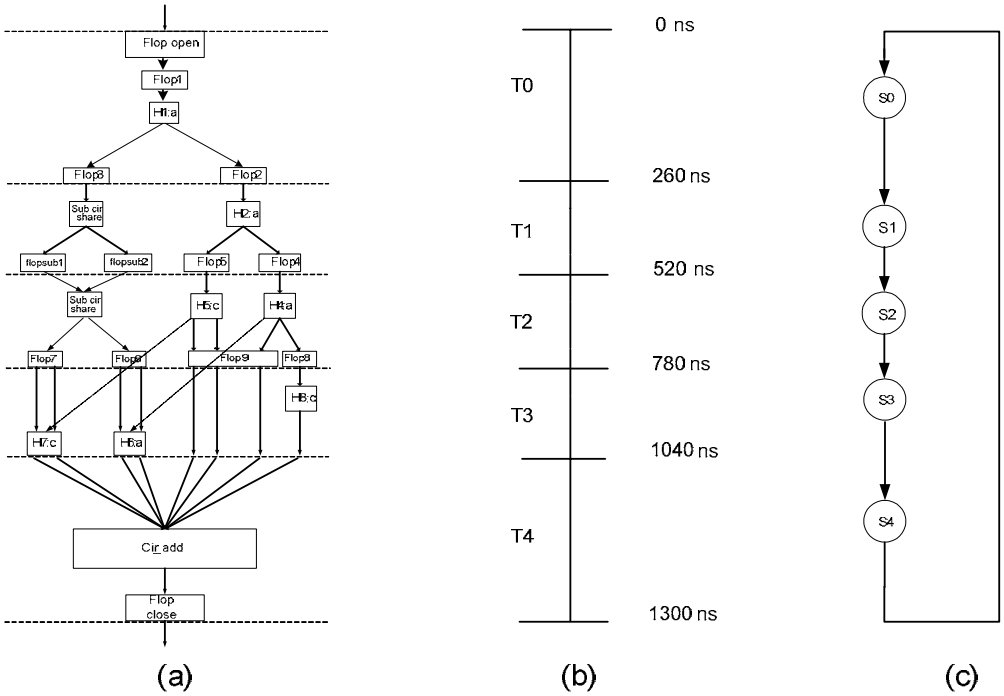
ภาพประกอบ 3-11 วงจรที่ออกแบบด้วยระเบียบวิธีการใช้ทรัพยากรร่วมระหว่างบล็อกและภายในบล็อก

เมื่อทำการออกแบบวงจรฟิลเตอร์แบงค์ครบทั้งสามแบบแล้วในขั้นตอนต่อไปจะกล่าวถึงการออกแบบวงจรควบคุมซึ่งเป็นวงจรที่ทำหน้าที่ส่งสัญญาณควบคุมการทำงานของวงจรฟิลเตอร์แบงค์นั่นเอง ซึ่งวงจรควบคุมที่กล่าวนี้มีความสำคัญเช่นกันเพราะหากทำการกำหนดหรือออกแบบวงจรควบคุมไม่ดีก็จะทำให้ค่าเอาท์พุตที่ได้ไม่ถูกต้องหรือวงจรฟิลเตอร์แบงค์อาจจะไม่สามารถทำงานได้

3.3 วงจรควบคุมแบบ FSM (Finite State Machine)

วงจรควบคุมแบบ FSM (Finite State Machine) ถูกนำมาใช้ในการควบคุมการทำงานของวงจรดาต้าพาทของวงจรดิจิตอลฟิลเตอร์แบงค์ โดยที่เลือกรูปแบบวงจรควบคุมแบบ Moore Machine ซึ่งเป็นรูปแบบการเขียนสเตตไดอะแกรมแบบค่าของเอาท์พุตในแต่ละสเตตจะขึ้นอยู่กับค่าของสเตตปัจจุบัน (Current State) เท่านั้นมาใช้ในการออกแบบควบคุมวงจรดาต้าพาทสำหรับงานวิจัยนี้ ทั้งนี้ในการออกแบบการทำงานของวงจรควบคุมจะต้องพิจารณาเวลาการทำงานของวงจรดาต้าพาทเป็นหลัก ดังนั้นค่าสัญญาณนาฬิกาที่เลือกใช้เพื่อออกแบบวงจรควบคุมจะต้องสัมพันธ์กับจังหวะเวลาการทำงานของวงจรภายในวงจรดาต้าพาทด้วย

ซึ่งสำหรับงานวิจัยนี้จะนำวงจรดาต้าพาทที่ได้ออกแบบไว้ข้างต้นมาใช้ในการพิจารณาหาค่าสัญญาณนาฬิกาที่เหมาะสมเพื่อการออกแบบวงจรควบคุม ตามวิธีที่แนะนำในบทก่อนหน้านี้ ซึ่งเริ่มต้นจากพิจารณาลำดับการทำงานของวงจรดาต้าพาทที่ต้องการออกแบบวงจรควบคุมแล้วทำการสร้างลำดับการทำงานของวงจรหรือเรียกว่าสเตปควบคุม (Control Step) และจากสเตปควบคุมจะสามารถเขียน State ของ FSM ได้ในขั้นตอนสุดท้ายของการออกแบบ โดยภาพประกอบที่ 3-12 เป็นการแสดงกระบวนการสร้าง State ของ FSM ที่จะควบคุมวงจรดาต้าพาท



ภาพประกอบ 3-12 วงจรควบคุมของวงจรดิจิตอลฟิลเตอร์แบงค์ ประกอบด้วย (a) คือ DFG ของวงจรดาต้าพาท (b) ขั้นเวลาในการออกแบบวงจรควบคุม และ (c) คือ วงจรควบคุมแบบ FSM

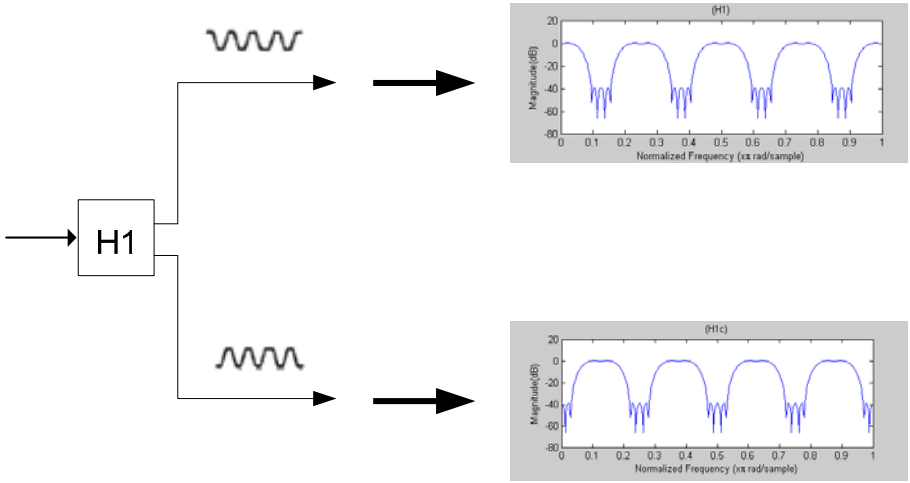
บทที่ 4

ผลการวิจัย

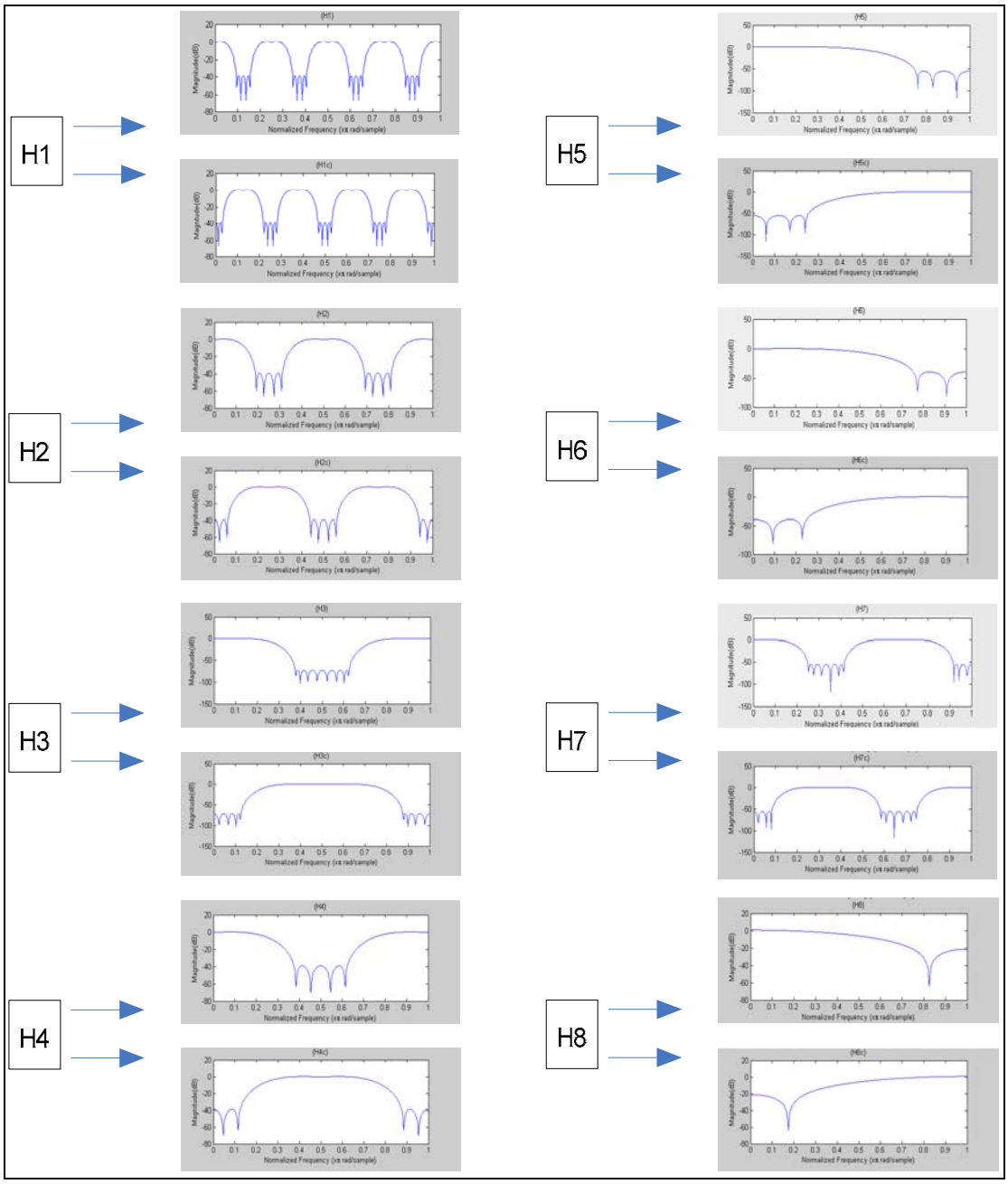
4.1 ผลการออกแบบฟิลเตอร์เพื่อหาค่าผลตอบสนองต่ออิมพัลส์ หรือ h(n)

จากการสร้างสมการผลต่างสำหรับใช้หาค่า h(n) ของวงจรฟิลเตอร์แบงค์ที่กล่าวไปในบทก่อนหน้านี้แล้ว ต่อจากนั้นจะนำสมการที่ได้ดังกล่าวไปออกแบบในโปรแกรม MATLAB เมื่อโปรแกรมทำการประมวลผลเสร็จแล้วจะสามารถทำการตรวจสอบดูค่า h(n) ของฟิลเตอร์แต่ละตัวได้ ทั้งนี้ในระหว่างกระบวนการหาค่า h(n) นั้นจะยึดหลักการตรวจสอบค่าของ h(n) โดยการให้โปรแกรมแสดงกราฟของลักษณะสัญญาณที่ได้จากการออกแบบสำหรับฟิลเตอร์แต่ละตัว รวมไปถึงจะทำการตรวจความถูกต้องของตำแหน่งค่า h(n) ตามตารางประกอบ 3-1 ไปด้วย โดยจะขอยกตัวอย่างการหาค่า h(n) ของ ฟิลเตอร์ $H_1(z)$ มาใช้ประกอบเพื่อความเข้าใจ

โดยเริ่มต้นจะทำการเปรียบเทียบรูปสัญญาณที่ควรได้จาก $H_1(z)$ กับกราฟแสดงรูปสัญญาณที่ได้จากการออกแบบบนโปรแกรม MATLAB โดยจะแสดงไว้ตามภาพประกอบ 4-1

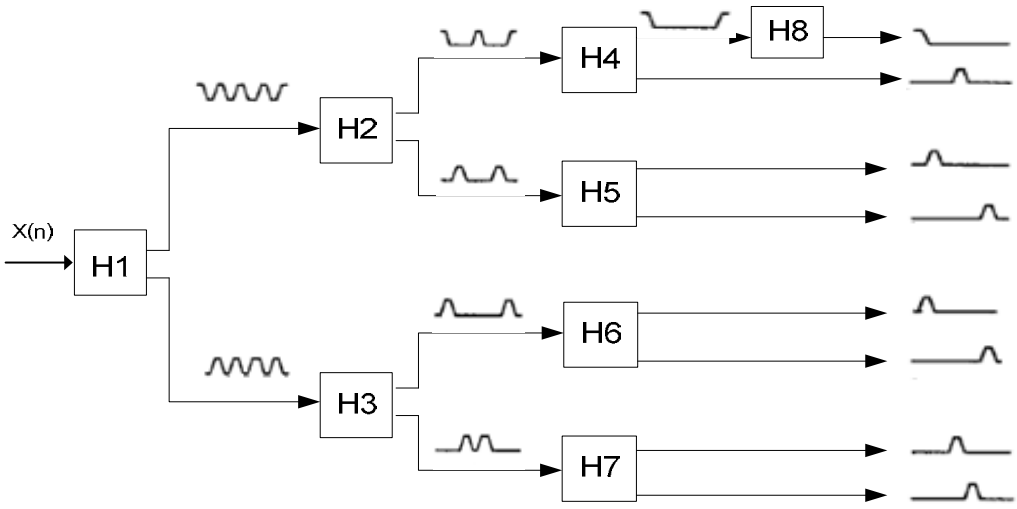


ภาพประกอบ 4-1 สัญญาณที่ออกจาก $H_1(z)$ ตามเอกสารอ้างอิงกับที่ได้จากการออกแบบ



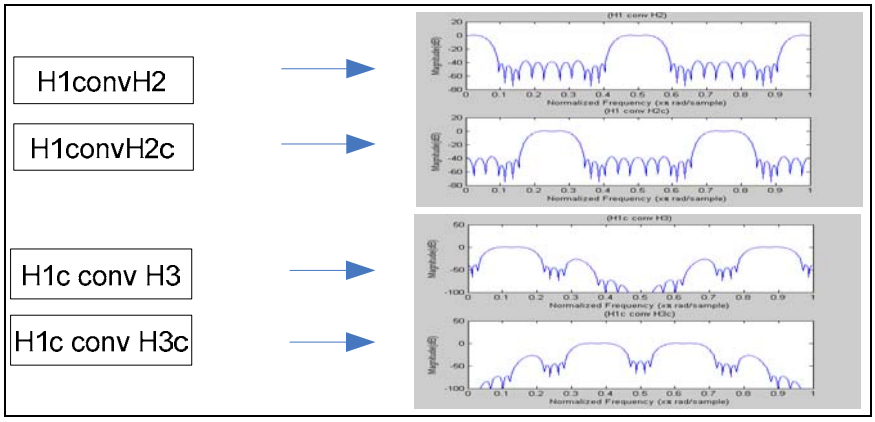
ภาพประกอบ 4-2 สัญญาณที่ออกจาก H1 กับสัญญาณภายในของฟิลเตอร์ทั้งหมดในฟิลเตอร์แบงค์

จากนั้นจะทำการตรวจสอบสัญญาณ output ทั้ง 8 ค่าของฟิลเตอร์แบงค์ โดยจะตรวจสอบความถูกต้องกับภาพประกอบสัญญาณ output ของฟิลเตอร์แบงค์ที่แสดงด้วยภาพประกอบ 4-3

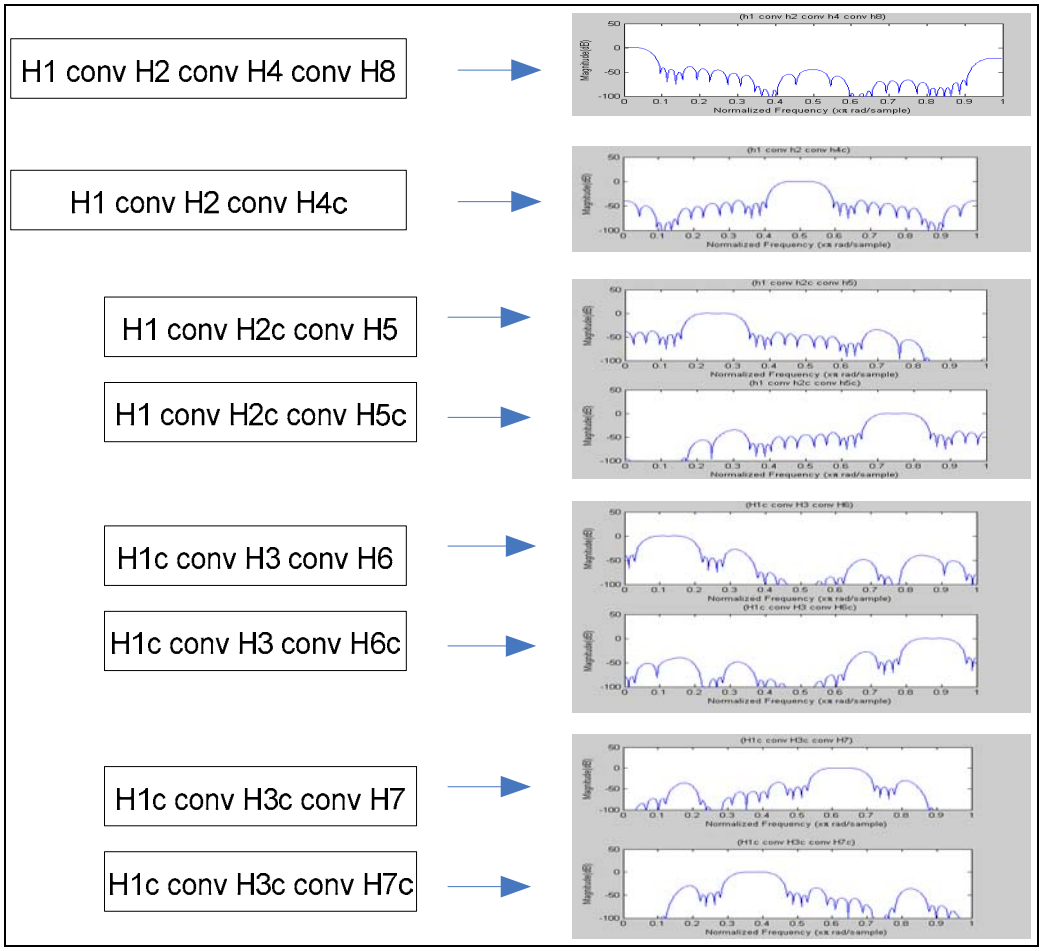


ภาพประกอบ 4-3 สัญญาณเอาท์พุตของฟิลเตอร์แบงค์

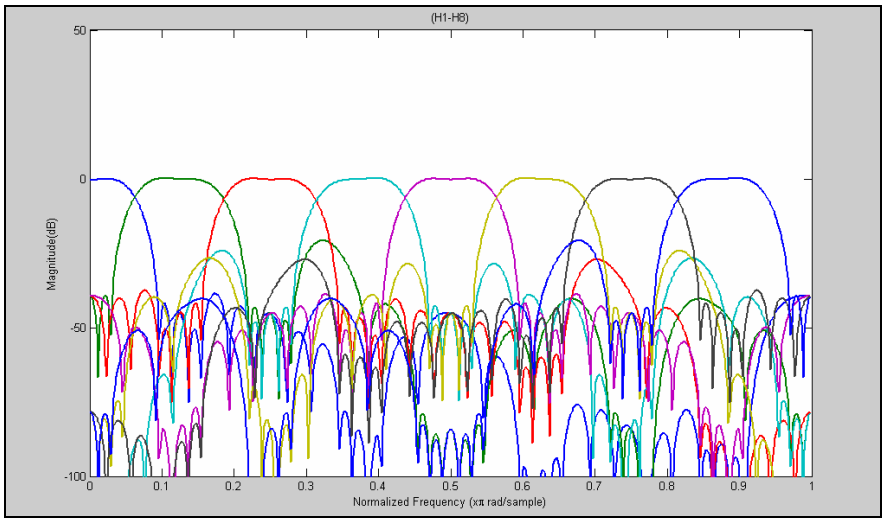
สังเกตสัญญาณด้านออกของฟิลเตอร์ H2 และ H3 จะเกิดจากการที่สัญญาณซึ่งออกจาก H1 ทั้งสองค่าผ่านเข้าไปประมวลผลในฟิลเตอร์ H2 และ H3 ตามลำดับ จากนั้นก็เขียนคำสั่งในโปรแกรม MATLAB เพื่อทำการตรวจสอบโดยจะเป็นการคอนโวลูชันสัญญาณเอาท์พุตจาก H1 กับสัญญาณภายใน H2 และ H3 จะได้สัญญาณของเอาท์พุตดังแสดงด้วยภาพประกอบ 4-4 จากหลักการเดียวกันนี้จะทำการตรวจสอบสัญญาณไปจนได้สัญญาณเอาท์พุตทั้งหมดของฟิลเตอร์แบงค์ ซึ่งหลังจากทำการเขียนคำสั่งและประมวลผลเรียบร้อยแล้ว



ภาพประกอบ 4-4 สัญญาณจากเอาท์พุตของ H1 คอนโวลูชันกับสัญญาณภายใน H2 และ H3



ภาพประกอบ 4-5 สัญญาณเอาท์พุตจากการทำการคอนโวลูชัน



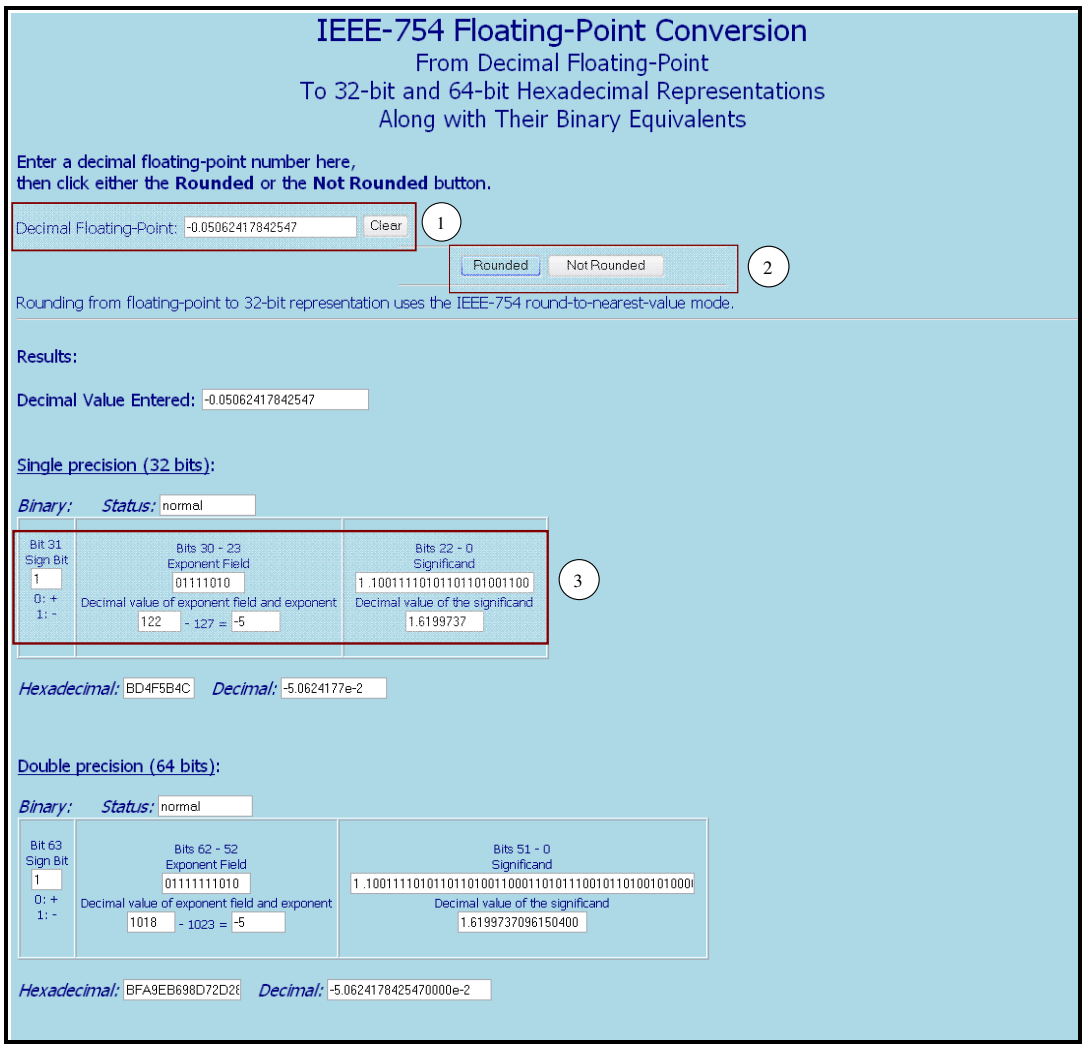
ภาพประกอบ 4-6 กราฟของเอาท์พุตเมื่อรวมกันทั้ง 8 แถบของฟิลเตอร์แบงค์

จากนั้นเมื่อตรวจสอบความถูกต้องของแถบสัญญาณทั้งหมดแล้ว ขั้นตอนต่อไปก็คือการตรวจสอบค่าของสัมประสิทธิ์ h(n) ของฟิลเตอร์ทั้งหมดซึ่งจะถูกเก็บไว้ในอาร์เรย์ของโปรแกรม MATLAB ซึ่งในการตรวจสอบนั้นจะต้องตรวจสอบตำแหน่งที่ค่า h(n) ถูกเก็บไว้ในอาร์เรย์เปรียบเทียบกับตำแหน่งที่เขียนไว้ในงานวิจัยที่นำมาอ้างอิง พบว่าเมื่อทำการออกแบบฟิลเตอร์ครบทั้งหมด 8 ตัว และทำการประมวลผลเพื่อหาค่าสัมประสิทธิ์ จากนั้นอ่านค่า h(n) ของฟิลเตอร์จากโปรแกรม MATLAB เมื่อนำค่าที่ได้มาเขียนตารางแสดงผลจะได้ตารางแสดงผลค่า h(n) ของฟิลเตอร์ทั้งหมดดังตารางประกอบ 4-1

ทั้งนี้ค่า h(n) ที่ได้นั้นจะแสดงให้เห็นเป็นเลขทศนิยมฐานสิบแต่การออกแบบวงจรฟิลเตอร์บน FPGA หรือบนโปรแกรม Xilinx สำหรับงานวิจัยนี้จะเป็นการประมวลผลแบบเลขฐานสองหรือไบนารีขนาด 32 บิต จึงต้องทำการแปลงค่า h(n) ที่ได้นี้ก่อนจะนำไปใช้ในวงจรบนโปรแกรม Xilinx ซึ่งจะใช้เวบไซต์ http://babbage.cs.qc.edu/IEEE-754/Decimal.html ในการแปลงค่าตัวเลขดังที่กล่าว

โดยภาพประกอบ 4-7 จะแสดงหน้าเวบไซต์ ที่ใช้ในการแปลงค่าอินพุตจากเลขทศนิยมฐานสิบให้อยู่ในรูปของเลขทศนิยมฐานสองซึ่งจะมีขั้นตอนในการแปลงค่าคือตำแหน่งที่1 ในหน้าเวบไซต์ตัวอย่างดังแสดงจะเป็นช่องว่างที่ใช้สำหรับการใส่ค่าตัวเลขทศนิยมซึ่งเป็นเลขฐานสิบ จากนั้นถัดมาที่ตำแหน่งที่ 2 เป็นการสอบถามถึงความต้องการความละเอียดในการแปลงค่าโดยจะให้เลือกว่าจะเลือกคิดค่าผลลัพธ์แบบใดระหว่างแบบปัดเศษหรือไม่ปัดเศษ ซึ่งหลังจากกดเลือกแล้วค่าเลขที่ถูกทำการแปลงแล้วจะแสดงให้เห็นบนหน้าเวบไซต์ในตำแหน่งที่ 3 ซึ่งค่าที่ผ่านการแปลงจะถูกแสดงในรูปแบบเลขฐานสองขนาด 32 บิตและ64 บิต รวมไปถึงแสดงเลขฐาน16 จากกระบวนที่กล่าวมานั้นค่าผลลัพธ์ที่จะนำไปใช้คือค่าที่เป็นเลขฐานสองขนาด 32 บิต ที่แสดงในตำแหน่งที่ 3 บนหน้าเวบไซต์นั่นเอง

จากวิธีเดียวกันนี้จะนำไปใช้เพื่อคิดค่า h(n) ทุกค่าที่ใช้ในการออกแบบสำหรับงานวิจัยนี้ โดยจะแสดงค่าเลขฐานสองของ h(n) สำหรับฟิลเตอร์ภายในฟิลเตอร์แบงค์ด้วยตารางที่ 4-1 และเวบไซต์แปลงค่านี้จะถูกนำไปใช้ในการแปลงค่าอินพุตที่จะป้อนให้แก่วงจรที่ออกแบบบน FPGA ด้วย



ภาพประกอบ 4-7 เวบไซต์ในการแปลงค่า h(n) จากเลขทศนิยมฐานสิบเป็นเลขทศนิยมฐานสอง

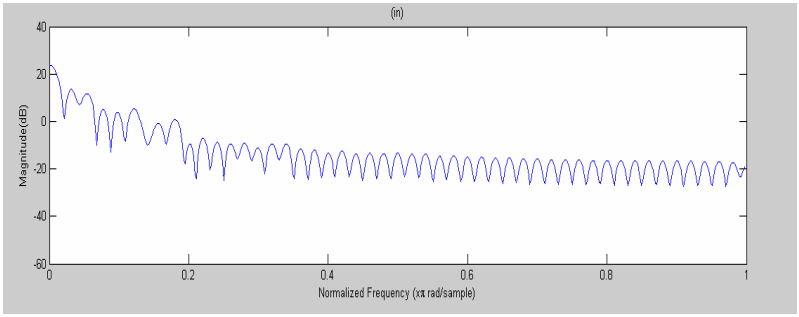
ตารางที่ 4-1 FIR Filter Bank Coefficient

Filter	H(n)	Decimal Floating Point	Binary Floating Point
$H_1(z)$	$h_1(0) = h_1(48)$	-0.05062417842547	10111101010011110101101101001100
	$h_1(16) = h_1(32)$	0.29505933470299	00111110100101110001001000000100
	$h_1(24)$	0.5	00111111000000000000000000000000
$H_2(z)$	$h_2(0) = h_2(24)$	-0.05062417842547	10111101010011110101101101001100
	$h_2(8) = h_2(16)$	0.29505933470299	00111110100101110001001000000100
	$h_2(12)$	0.5	00111111000000000000000000000000
$H_3(z)$	$h_3(0) = h_3(28)$	-0.00373765573262	10111011011101001111001101110101
	$h_3(4) = h_3(24)$	0.02056989487010	00111100101010001000001000110010
	$h_3(8) = h_3(20)$	-0.07232190470689	10111101100101000001110110000010
	$h_3(12) = h_3(16)$	0.30537047362544	00111110100111000101100110000101
	$h_3(14)$	0.5	00111111000000000000000000000000
$H_4(z)$	$h_4(0) = h_4(12)$	-0.05062417842547	10111101010011110101101101001100
	$h_4(4) = h_4(8)$	0.29505933470299	00111110100101110001001000000100
	$H_4(6)$	0.5	00111111000000000000000000000000
$H_5(z)$	$h_5(0) = h_5(10)$	0.01304920555205	00111100010101011100110001010110
	$h_5(2) = h_5(8)$	-0.06387151210405	10111101100000101100111100010001
	$h_5(4) = h_5(6)$	0.30161294807561	00111110100110100110110100000011
	$h_5(5)$	0.5	00111111000000000000000000000000
$H_6(z)$	$h_6(0) = h_6(6)$	-0.05062417842547	10111101010011110101101101001100
	$h_1(2) = h_1(4)$	0.29505933470299	00111110100101110001001000000100
	$h_6(3)$	0.5	00111111000000000000000000000000
$H_7(z)$	$h_7(0) = h_7(30)$	0.01304920555205	00111100010101011100110001010110
	$h_7(6) = h_7(24)$	-0.06387151210405	10111101100000101100111100010001
	$h_7(12) = h_7(18)$	0.30161294807561	00111110100110100110110100000011
	$h_7(15)$	0.5	00111111000000000000000000000000
$H_8(z)$	$h_1(0) = h_1(2)$	0.29289321881345	00111110100101011111011000011010
	$h_8(1)$	0.5	00111111000000000000000000000000

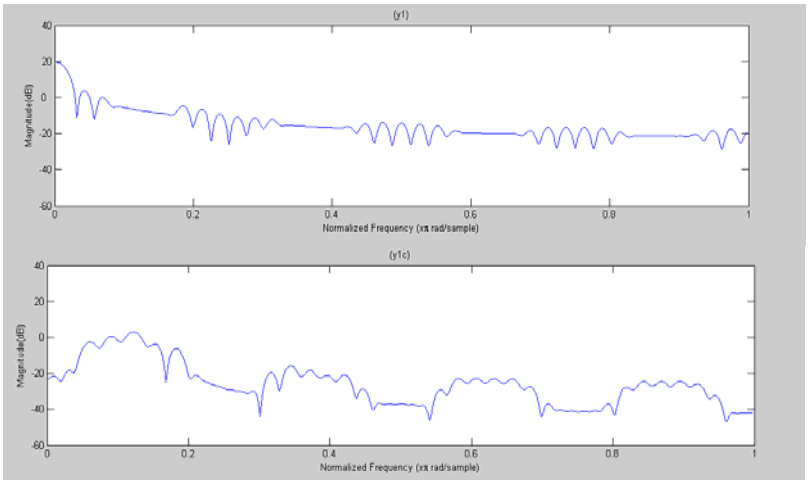
4.2 ผลการวิจัยของวงจรดิจิตอลฟิลเตอร์แบงค์ด้วยโปรแกรม MATLAB

หลังจากที่ได้ค่า h(n) และทราบสมการประมวลผลของวงจรฟิลเตอร์แบงค์แล้วนั้น ขั้นตอนต่อมาก็จะเป็นการทดสอบวงจรด้วยโปรแกรม MATLAB โดยจะเป็นการป้อนค่าอินพุตให้แก่วงจรฟิลเตอร์แบงค์และค่าเอาท์พุตที่ได้จากการทำงานของวงจรจะถูกนำไว้ใช้ตรวจสอบเปรียบเทียบกับเอาท์พุตที่ได้จากวงจรฟิลเตอร์แบงค์ที่ออกแบบบนโปรแกรม Xilinx ต่อไป

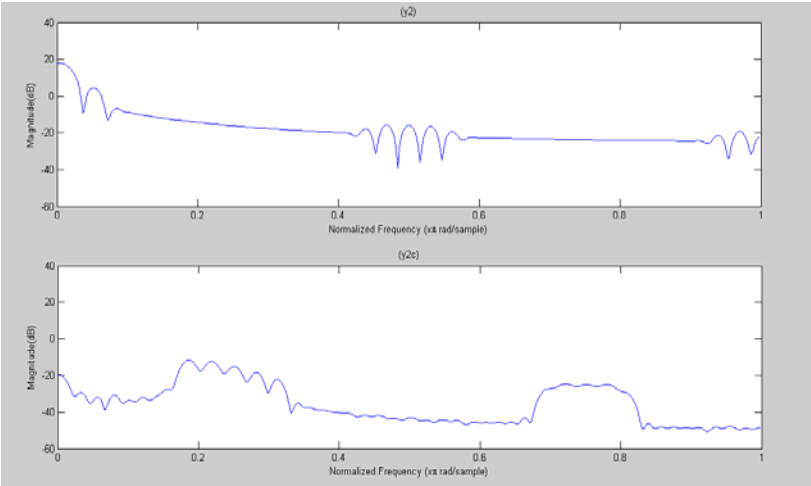
โดยในงานวิจัยนี้จะใช้อินพุตเป็นสัญญาณเสียงเพลงซึ่งจะถูกเก็บค่าไว้ในอาร์เรย์เป็นลำดับใน MATLAB ทั้งนี้สัญญาณเสียงเพลงดังกล่าวจะมีจำนวนอินพุตที่มากลำดับแต่ในการทดสอบจะขอยกอินพุต 100 ลำดับแรกมาใช้ทดสอบเพื่อเปรียบเทียบผลในงานวิจัยนี้โดยในภาพประกอบ 4-8 จะแสดงลักษณะของสัญญาณเสียงเพลงที่นำมาใช้จำนวน 100 ลำดับแรกเพื่อป้อนเป็นอินพุตให้วงจรฟิลเตอร์แบงค์เพื่อประมวลผลหาค่าเอาท์พุตและแสดงสัญญาณเสียงเอาท์พุตที่ได้



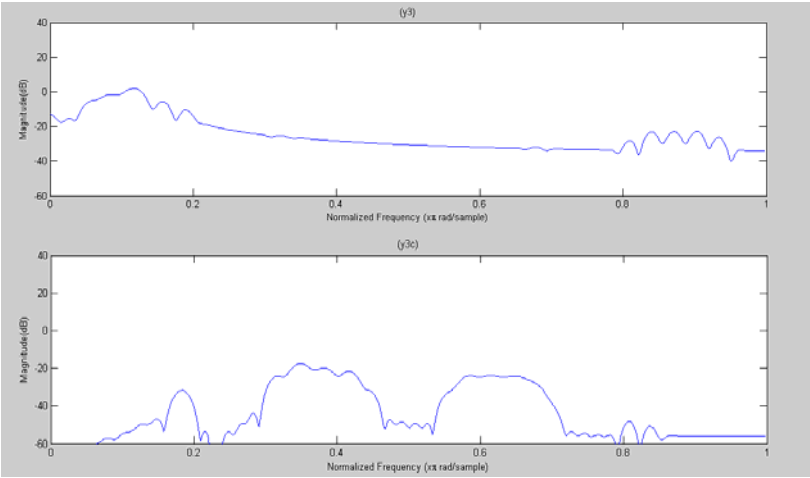
ภาพประกอบ 4-8 กราฟสัญญาณของอินพุตที่ใช้ทำการประมวลผลบน MATLAB



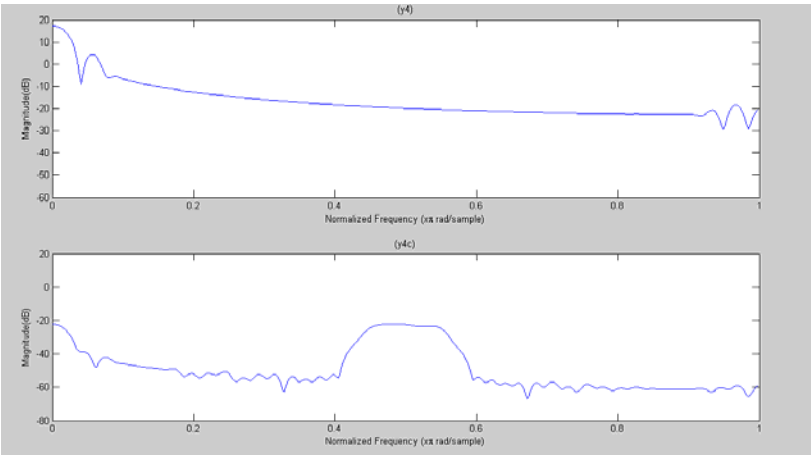
ภาพประกอบ 4-9 กราฟสัญญาณเอาท์พุต y1 และ y1c ที่ได้จากอินพุตผ่านฟิลเตอร์ H1



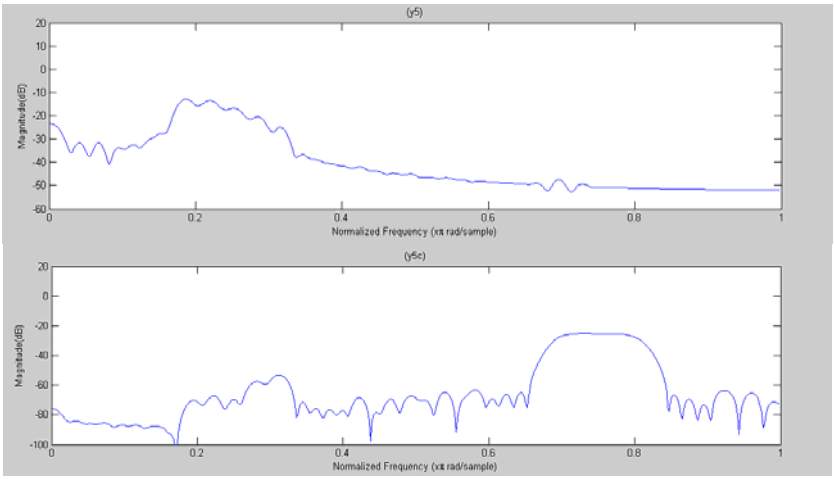
ภาพประกอบ 4-10 กราฟสัญญาณ y2 และ y2c ที่ได้จากเอาท์พุต y1 ผ่านฟิลเตอร์ H2



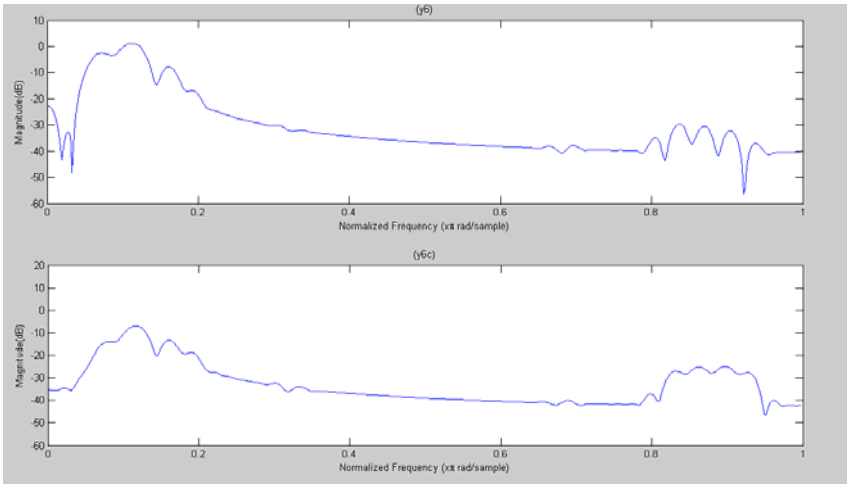
ภาพประกอบ 4-11 กราฟสัญญาณ y3 และ y3c ที่ได้จากเอาท์พุต y1c ผ่านฟิลเตอร์ H3



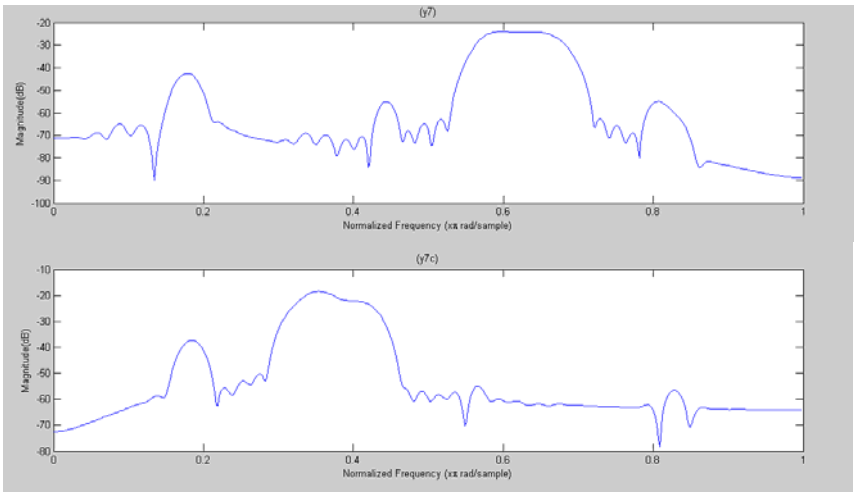
ภาพประกอบ 4-12 กราฟสัญญาณ y4 และ y4c ที่ได้จากเอาท์พุต y2 ผ่านฟิลเตอร์ H4



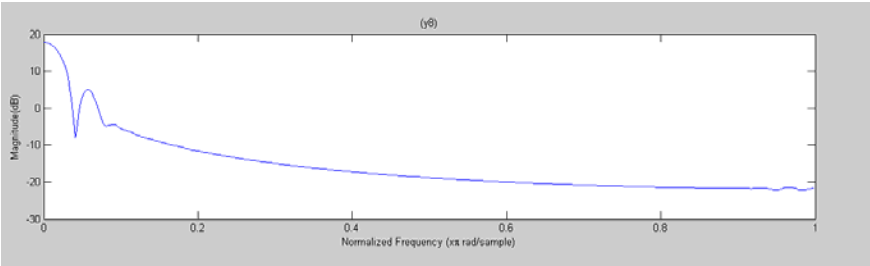
ภาพประกอบ 4-13 กราฟสัญญาณ y5 และ y5c ที่ได้จากเอาท์พุต y2c ผ่านฟิลเตอร์ H5



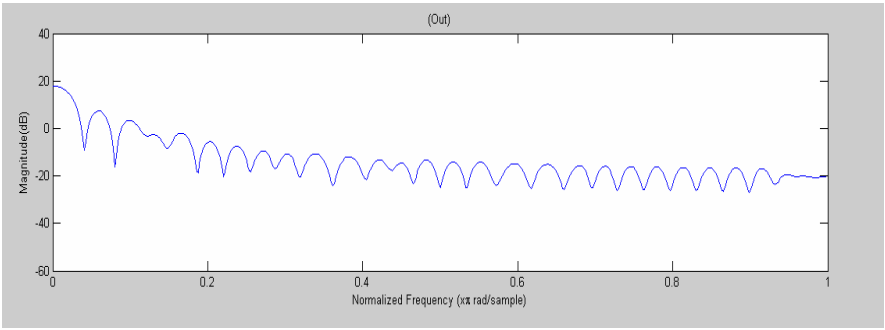
ภาพประกอบ 4-14 กราฟสัญญาณ y6 และ y6c ที่ได้จากเอาท์พุต y3 ผ่านฟิลเตอร์ H6



ภาพประกอบ 4-15 กราฟสัญญาณ y7 และ y7c ที่ได้จากเอาท์พุต y3c ผ่านฟิลเตอร์ H7



ภาพประกอบ 4-16 กราฟสัญญาณ y8 ที่ได้จากเอาท์พุต y4 ผ่านฟิลเตอร์ H8



ภาพประกอบ 4-17 กราฟสัญญาณของเอาท์พุตที่ได้จากการประมวลผลบน MATLAB

4.3 ผลการวิจัยของวงจรดิจิตอลฟิลเตอร์แบงค์ด้วยโปรแกรม Xilinx

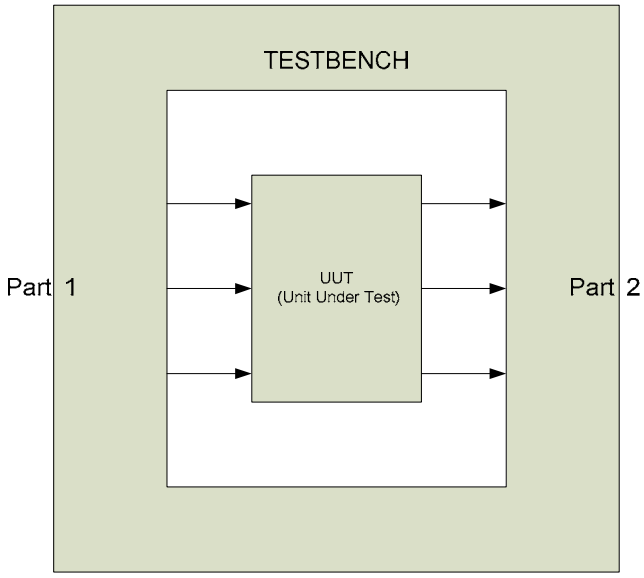
เมื่อผ่านการทดสอบการทำงานของวงจรฟิลเตอร์แบงค์บนโปรแกรม MATLAB มาแล้วนั้นในลำดับถัดไปก็คือการออกแบบและทดสอบการทำงานของฟิลเตอร์แบงค์บน FPGAs หรือหมายถึงการออกแบบและทดสอบบนโปรแกรม Xilinx นั่นเอง โดยที่ลำดับการออกแบบจะเป็นลำดับดังนี้คือ เริ่มต้นจากการออกแบบวงจรในรูปแบบธรรมดาคือการออกแบบวงจรแบบเดียวกันกับที่ออกแบบบน MATLAB จากนั้นจะออกแบบวงจรโดยจะใช้ระเบียบวิธีที่นำเสนอในงานวิจัยนี้คือการออกแบบบโดยใช้ทรัพยากรดาต้าพาทร่วมกันแบบลำดับชั้น ซึ่งจะออกแบบโดยใช้ระเบียบวิธีการใช้ทรัพยากรร่วมกันระหว่างบล๊อกเท่านั้นก่อน ในลำดับสุดท้ายจึงจะใช้ระเบียบวิธีการใช้ทรัพยากรร่วมกันระหว่างบล็อกและการใช้ทรัพยากรร่วมภายในบล็อกมาทำการออกแบบ

งานวิจัยนี้จะออกแบบวงจรและทดสอบบนชิพ FPGA ของบริษัท Xilinx ตระกูล SPARTAN-3 เบอร์ XC3S4000-5FG676 ในการออกแบบจะเป็นการเขียนคำสั่งเพื่อบรรยายวงจรด้วยภาษา VHDL และใช้โปรแกรม Xilinx ISE 8.1i ในการสังเคราะห์ (Synthesis) เพื่อดูผลจากการสังเคราะห์วงจรทั้งด้านการใช้ทรัพยากรบน FPGA และความเร็วในการทำงานของวงจรจากนั้นนำ

วงจรที่ได้ทำการจำลองการทำงาน (Simulation) เพื่อตรวจสอบผลการทำงานของวงจรแบบต่างๆทั้ง 3 แบบที่กล่าวไปข้างต้น

โดยจะทำการออกแบบวงจรมาจนถึงขั้นตอนการจำลองการทำงานระดับฐานเวลาจริง (Timing Simulation) แล้วทำการทดสอบวงจรด้วยการเขียน Testbench ซึ่งเป็นการสร้างระบบการทดสอบ (Test environment) ของวงจรที่ต้องการทดสอบการทำงาน

Testbench สามารถเขียนขึ้นจากภาษา VHDL โดยหน้าที่ของ Testbench คือการทำหน้าที่สร้างสัญญาณกระตุ้นให้กับวงจรที่ต้องการทดสอบการทำงานและดูผลจากการทดสอบได้



ภาพประกอบ 4-18 ระบบการทดสอบ (Testbench)

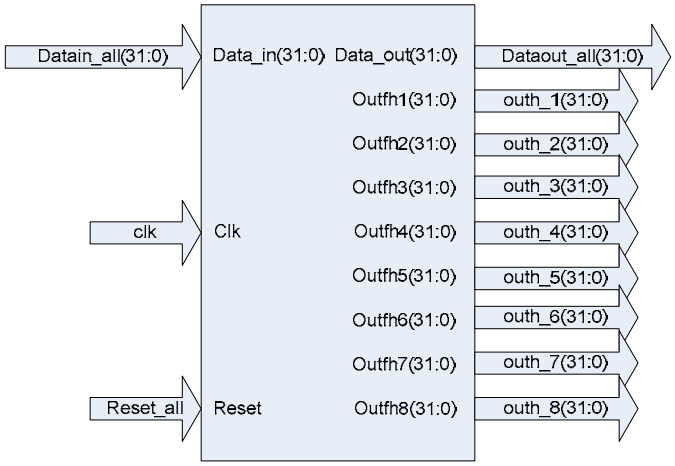
จากภาพประกอบ 3-13 จะเห็นได้ว่าในระบบทดสอบ (Testbench) จะถูกแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ 1 ทำหน้าที่สร้างสัญญาณกระตุ้นให้กับสัญญาณอินพุตของวงจรที่อยู่ในระบบหรือที่เรียกว่า UUT: Unit Under Test และส่วนที่ 2 จะเป็นส่วนของการตรวจสอบสัญญาณเอาท์พุตหลังจากการกระตุ้นสัญญาณให้กับสัญญาณอินพุตของวงจรเรียบร้อยแล้ว

4.3.1 วงจรฟิลเตอร์แบงค์ที่ไม่มีการใช้ทรัพยากรร่วมกัน

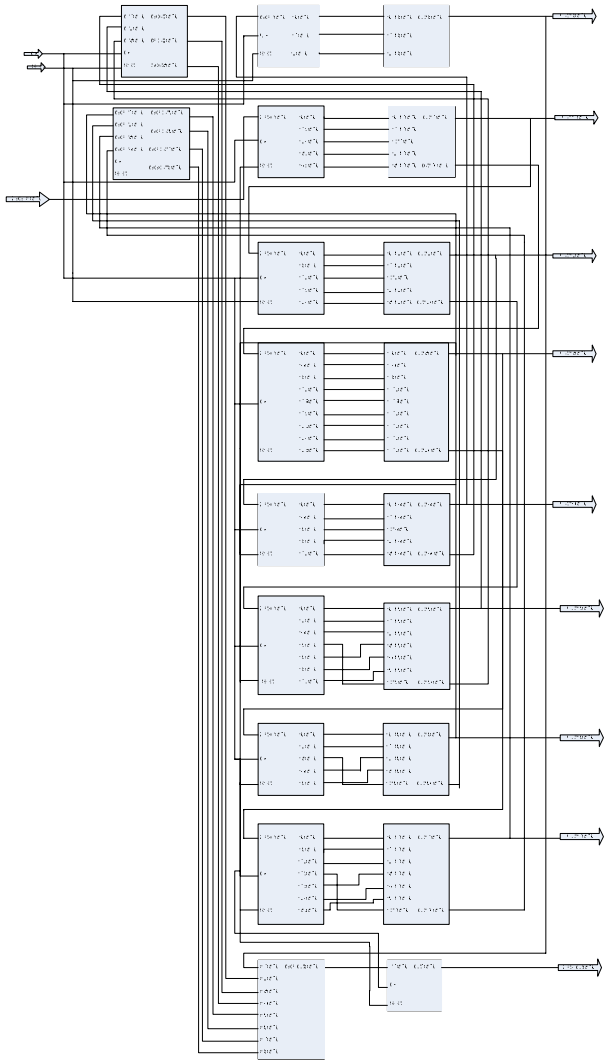
ขั้นตอนการออกแบบและการทดสอบวงจรฟิลเตอร์บนโปรแกรม Xilinx นั้นจะเริ่มต้นด้วยการออกแบบวงจรที่ไม่มีการใช้ทรัพยากรร่วมกันโดยวงจรจะมีพฤติกรรมในการทำงานแบบเดียวกับวงจรที่ได้ออกแบบและประมวลผลบน MATLAB ซึ่งในการออกแบบนั้นจะทำการออกแบบให้มีการใช้ทรัพยากรการออกแบบที่น้อยที่สุดเท่าที่สามารถจะทำได้และเอาท์พุตที่ได้จากการประมวลผลต้องตรงกันกับที่ประมวลผลด้วย MATLAB

ผลจากการออกแบบและสังเคราะห์วงจร(Synthesis) แล้วนั้นจะแสดงให้เห็นรูป Component ของวงจร ตารางเกี่ยวกับ Timing-Summary และ Device Utilization Summary ด้วยภาพประกอบ 4-19 ถึง ภาพประกอบ 4-22

จากนั้นเมื่อทำการสังเคราะห์วงจรแล้วขั้นตอนต่อไปก็คือการทดสอบการทำงานของวงจรเพื่อนำค่าเอาท์พุตที่ได้ไปเปรียบเทียบกับเอาท์พุตบน MATLAB โดยจะทำการทดสอบการทำงานของวงจรด้วยโปรแกรม ModelSim SE Plus 6.2b ผลการจำลองการทำงานของวงจรและกราฟเปรียบเทียบเอาท์พุตที่ได้จากวงจรบน MATLAB กับเอาท์พุตของวงจรที่ออกแบบบน Xilinx จะแสดงด้วยภาพประกอบ 4-23 และภาพประกอบ 4-24 ตามลำดับ



ภาพประกอบ 4-19 Component ของวงจรฟิลเตอร์แบงค์ที่ไม่มีการใช้ทรัพยากรร่วมกันที่สังเคราะห์บนโปรแกรม Xilinx ISE8.1i



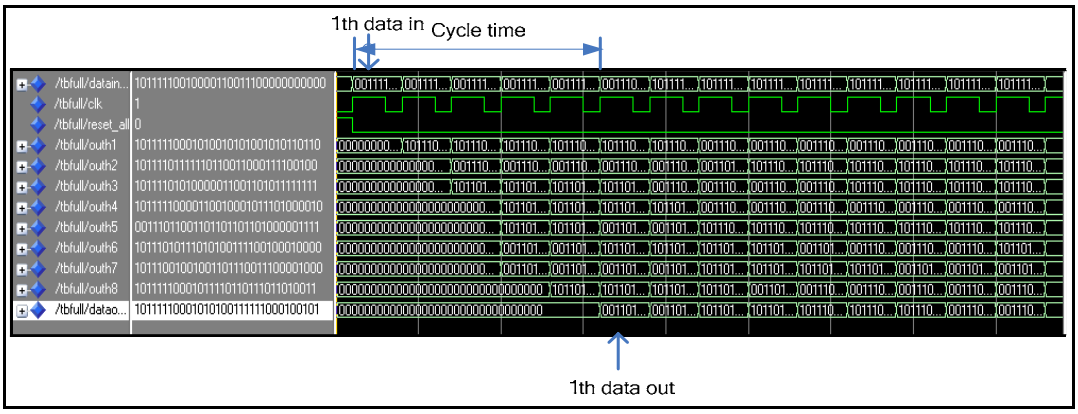
ภาพประกอบ 4-20 วงจรฟิลเตอร์แบงค์ที่ไม่มีการใช้ทรัพยากรร่วมกัน

```
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 3s4000fg676-5

 Number of Slices:                    22887  out of  27648    82%
 Number of Slice Flip Flops:           5641  out of  55296    10%
 Number of 4 input LUTs:              43476  out of  55296    78%
 Number of bonded IOBs:                 322  out of    489    65%
 Number of MULT18X18s:                   13  out of     96    13%
 Number of GCLKs:                         1  out of      8    12%


=========================================================================
```

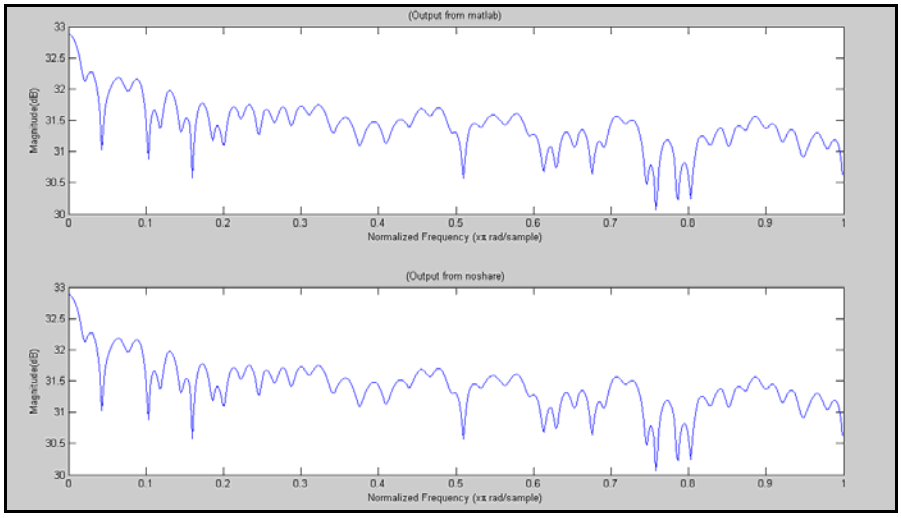
ภาพประกอบ 4-21 Device Utilization Summary ของวงจรฟิลเตอร์แบงค์ที่ไม่มีการใช้ทรัพยากรร่วมกัน

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 189.160ns (Maximum Frequency: 5.287MHz)
   Minimum input arrival time before clock: 1.572ns
   Maximum output required time after clock: 160.963ns
   Maximum combinational path delay: No path found
```

ภาพประกอบ 4-22 Timing Summary ของวงจรฟิลเตอร์แบงค์ที่ไม่มีการใช้ทรัพยากรร่วมกัน



ภาพประกอบ 4-23 ผลการจำลองการทำงานของวงจรฟิลเตอร์แบงค์ที่ไม่มีการใช้ทรัพยากรร่วมกันด้วยโปรแกรม ModelSim SE Plus 6.2b

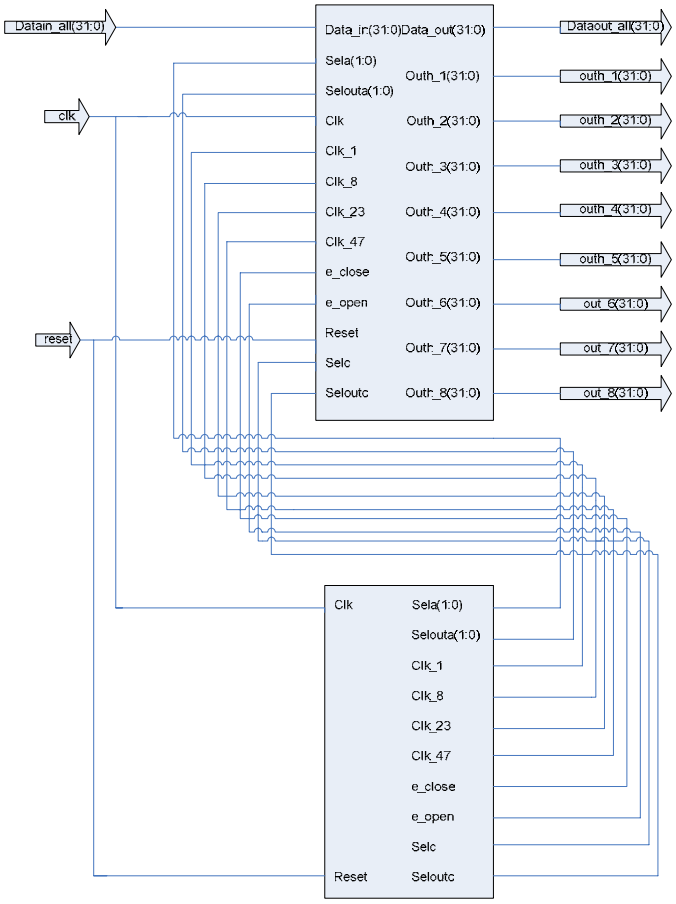


ภาพประกอบ 4-24 กราฟเปรียบเทียบเอาท์พุตที่ได้จากวงจรฟิลเตอร์แบงค์บน MATLAB กับเอาท์พุตของวงจรฟิลเตอร์แบงค์ที่ออกแบบบน Xilinx

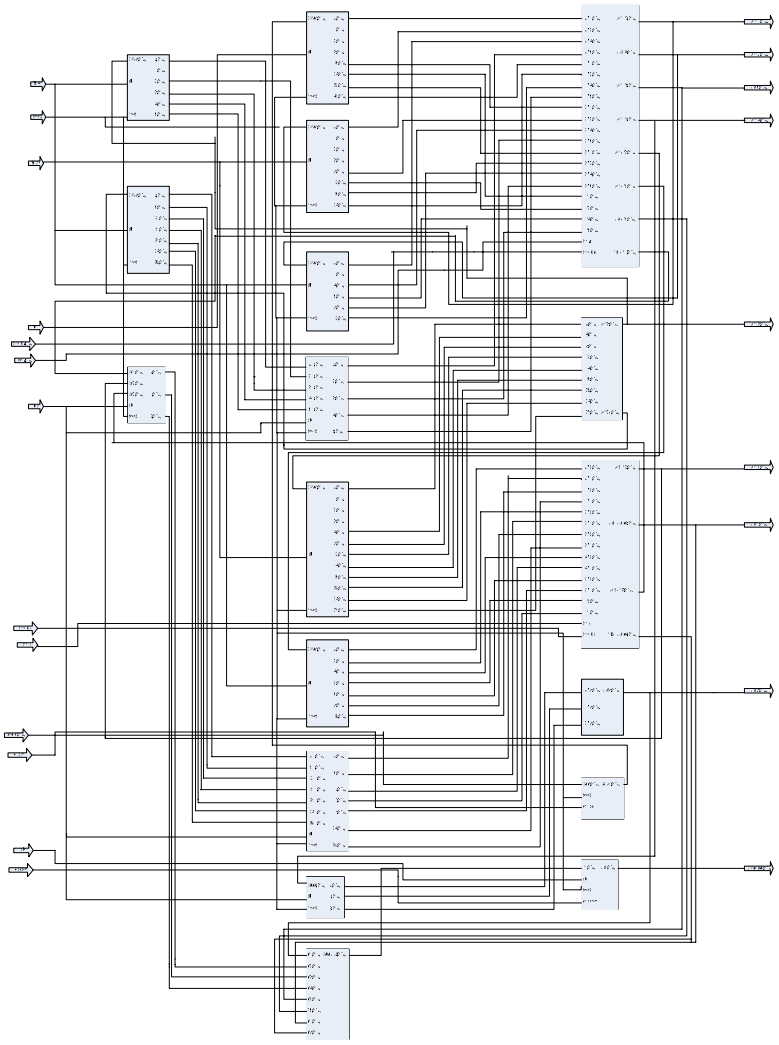
4.3.2 วงจรฟิลเตอร์แบงค์ที่ใช้ทรัพยากรร่วมกันระหว่างบล็อก (Inter-Block Sharing)

จากนั้นการออกแบบวงจรในขั้นตอนต่อไปคือจะนำระเบียบวิธีที่นำเสนอคือการใช้ทรัพยากรร่วมกันแบบลำดับชั้นมาใช้ในการออกแบบแต่จะใช้ระเบียบวิธีการใช้ทรัพยากรร่วมกันระหว่างบล็อกเพียงอย่างเดียว เพื่อดูพฤติกรรมการทำงานของวงจรว่าทำงานได้ผลเอาท์พุตเหมือนกับวงจรที่ไม่มีการใช้ทรัพยากรร่วมกันหรือไม่

ผลจากการออกแบบและสังเคราะห์วงจร(Synthesis) รูป Component ของวงจร ตารางเกี่ยวกับ Timing-Summary และ Device Utilization Summary ผลการจำลองการทำงานของวงจรและกราฟเปรียบเทียบเอาท์พุตของวงจรที่ออกแบบบน MATLAB กับเอาท์พุตของวงจรที่ออกแบบด้วยระเบียบวิธีการใช้ทรัพยากรร่วมกันระหว่างบล็อกบน Xilinx จะแสดงด้วย ภาพประกอบ 4-25 ถึง ภาพประกอบ 4-30 ตามลำดับ



ภาพประกอบ 4-25 Component ของวงจรฟิลเตอร์แบงค์ที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อก (Inter-block Sharing) ที่สังเคราะห์บนโปรแกรม Xilinx ISE8.1i



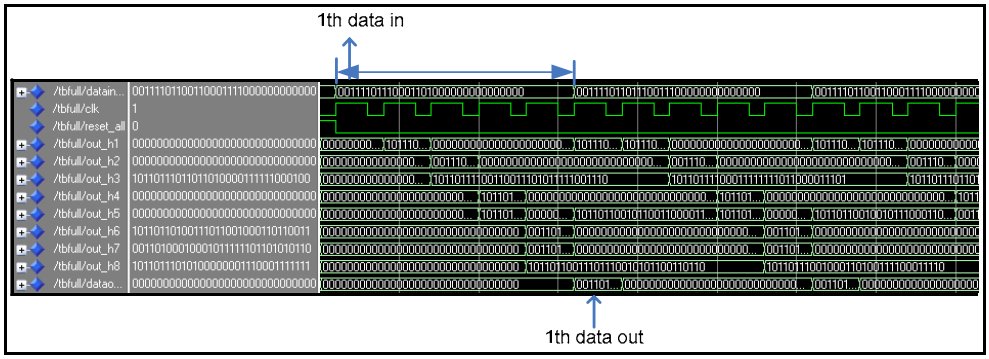
ภาพประกอบ 4-26 วงจรฟิลเตอร์แบงค์ที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อก(Inter-block Sharing)

```
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 3s4000fg676-5

 Number of Slices:                    14281  out of  27648    51%
 Number of Slice Flip Flops:           5929  out of  55296    10%
 Number of 4 input LUTs:              26591  out of  55296    48%
 Number of bonded IOBs:                 334  out of    489    68%
 Number of MULT18X18s:                    7  out of     96     7%
 Number of GCLKs:                         5  out of      8    62%


=========================================================================
```

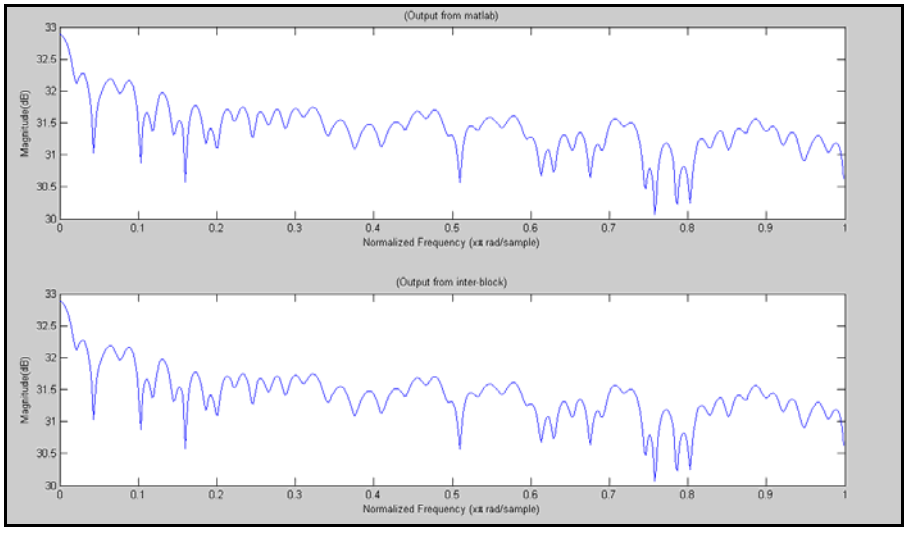
ภาพประกอบ 4-27 แสดง Device Utilization Summary ของวงจร Inter-block Sharing

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 255.620ns (Maximum Frequency: 3.912MHz)
   Minimum input arrival time before clock: 2.110ns
   Maximum output required time after clock: 250.151ns
   Maximum combinational path delay: No path found
```

ภาพประกอบ 4-28 Timing Summary ของวงจร Inter-block Sharing



ภาพประกอบ 4-29 ผลการจำลองการทำงานของวงจร Inter-block Sharing ด้วยโปรแกรม ModelSim SE Plus 6.2b

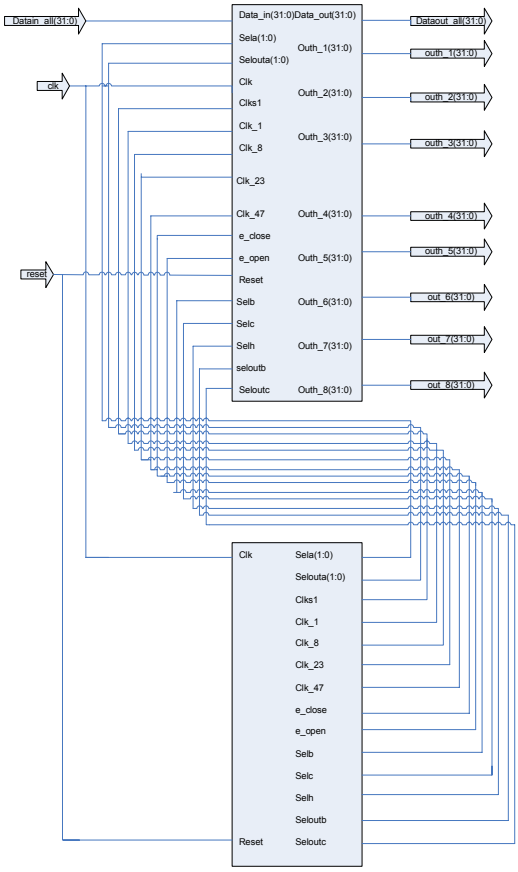


ภาพประกอบ 4-30 กราฟเปรียบเทียบเอาท์พุตที่ได้จากวงจรฟิลเตอร์แบงค์บน MATLAB กับเอาท์พุตของวงจรฟิลเตอร์แบงค์แบบ Inter-block Sharing

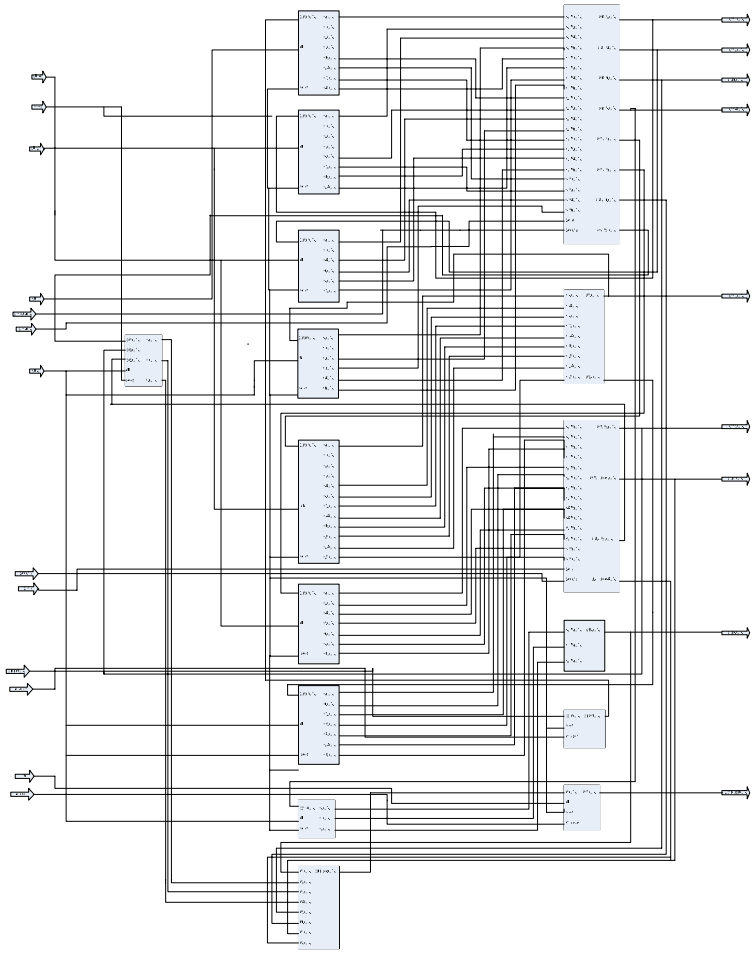
4.3.3 วงจรฟิลเตอร์แบงค์ที่ใช้ทรัพยากรร่วมกันระหว่างบล็อกและภายในบล็อก (Inter-Block and Intra-Block Sharing)

ในการออกแบบวงจรรูปแบบสุดท้ายสำหรับงานวิจัยนี้เป็นการออกแบบวงจรส่วนของดาต้าพาทด้วยการใช้ทรัพยากรร่วมกันทั้งระหว่างบล็อกและภายในบล็อกซึ่งในขั้นตอนนี้จะต้องใช้ความละเอียดในการออกแบบทั้งนี้เนื่องจากจะมีการใช้ทรัพยากรร่วมกันในหลายๆ ตำแหน่งภายในวงจรรวมดังนั้นสายสัญญาณเชื่อมต่อกันระหว่างหน่วยประมวลผลก็จะมีจำนวนมากเช่นกัน เมื่อทำการออกแบบเรียบร้อยแล้วก็ทำการสังเคราะห์วงจร

ผลจากการสังเคราะห์วงจรนั้นจะนำมาแสดงให้เห็นคือรูป Component ของวงจร ตารางแสดง Timing-Summary และ Device Utilization Summary โดยจะแสดงด้วยภาพประกอบ 4-31 ไปจนถึงภาพประกอบ 4-34 ผลการจำลองการทำงานของวงจร และกราฟเปรียบเทียบเอาท์พุตจะแสดงด้วยภาพประกอบ 4-35 และภาพประกอบ 4-36 ตามลำดับ



ภาพประกอบ 4-31 Component ของวงจรฟิลเตอร์แบงค์ที่ใช้ทรัพยากรร่วมกันระหว่างบล็อกและภายในบล็อกที่สังเคราะห์บน Xilinx



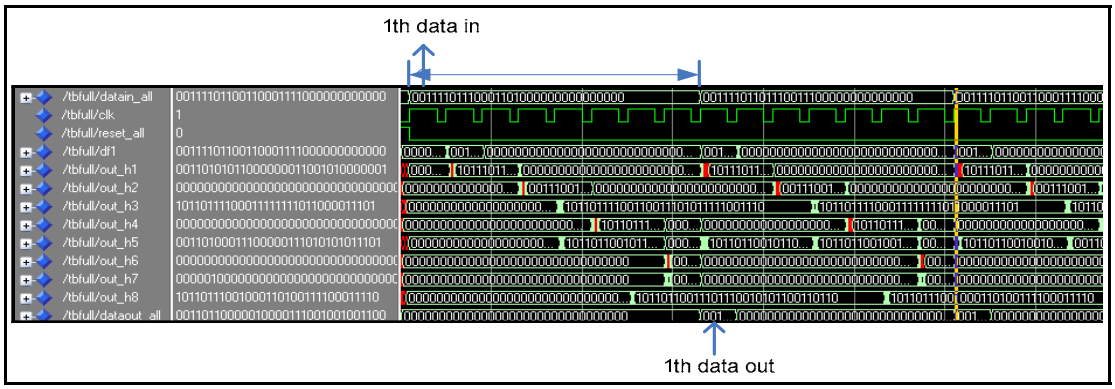
ภาพประกอบ 4-32 วงจรฟิลเตอร์แบงค์ที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อก และภายในบล๊อก(Inter and Intra-block sharing)

```
=========================================================================

Device utilization summary:
---------------------------

Selected Device : 3s4000fg676-5

 Number of Slices:                    12927  out of  27648    46%
 Number of Slice Flip Flops:           5745  out of  55296    10%
 Number of 4 input LUTs:              24285  out of  55296    43%
 Number of bonded IOBs:                 369  out of    489    75%
 Number of MULT18X18s:                   10  out of     96    10%
 Number of GCLKs:                         5  out of      8    62%


=========================================================================
```

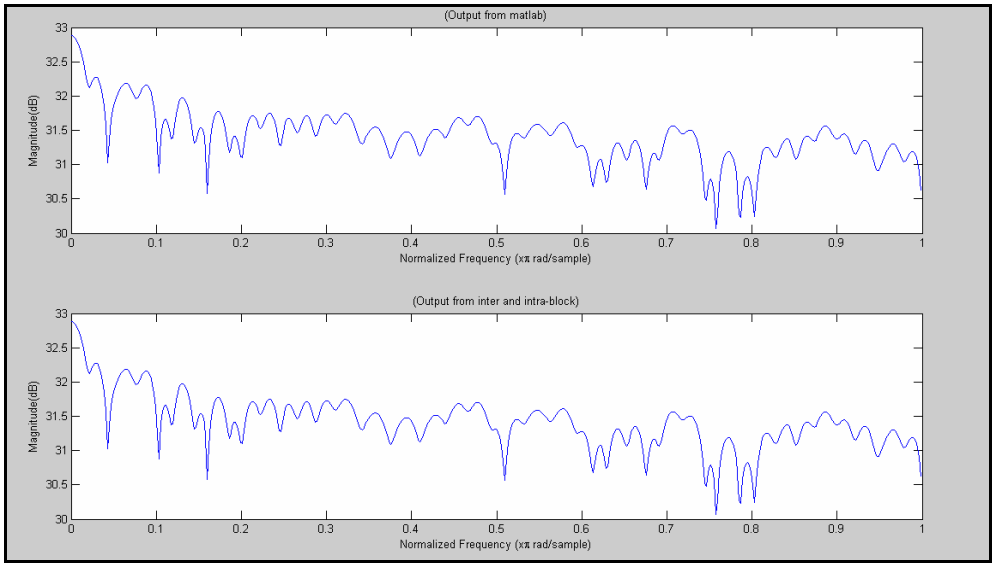
ภาพประกอบ 4-33 Device Utilization Summary ของวงจร Inter and Intra-block sharing

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 257.928ns (Maximum Frequency: 3.877MHz)
   Minimum input arrival time before clock: 11.551ns
   Maximum output required time after clock: 134.554ns
   Maximum combinational path delay: 17.029ns
```

ภาพประกอบ 4-34 Timing Summary ของวงจร Inter and Intra-block sharing



ภาพประกอบ 4-35 ผลการจำลองการทำงานของวงจร Inter and Intra-block sharing ด้วยโปรแกรม ModelSim SE Plus 6.2b



ภาพประกอบ 4-36 กราฟเปรียบเทียบเอาท์พุตที่ได้จากวงจรฟิลเตอร์แบงค์บน MATLAB กับเอาท์พุตของวงจรฟิลเตอร์แบงค์แบบ Inter and Intra-block Sharing

จากผลการสังเคราะห์และการจำลองการทำงานของวงจรที่ได้แสดงไปข้างต้นนั้น จะนำมาทำเป็นตารางเปรียบเทียบกันเพื่อวิเคราะห์ผลการใช้ทรัพยากรของวงจรในรูปแบบต่างๆที่ได้ทำการออกแบบโดยจะแสดงด้วยตารางที่ 4-2

ตารางที่ 4-2 เปรียบเทียบการใช้ทรัพยากรในการออกแบบวงจรฟิลเตอร์แบงค์ทั้ง 3 แบบ บน FPGA SPARTAN-3 เบอร์ XC3S4000FG676-5

วงจรฟิลเตอร์แบงค์	**ปริมาณการใช้ LUTs**	**เทียบกับวงจรแบบไม่มีการใช้ทรัพยากรร่วมกัน**
แบบที่ไม่มีการใช้ทรัพยากรร่วมกัน	43476 out of 55296	(เป็นวงจรต้นแบบ)
แบบที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อก (Inter-block Sharing)	26591 out of 55296	ลดลง 38 %
แบบที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อกและภายในบล็อก(Inter and Intra-block Sharing)	24285 out of 55296	ลดลง 44 %

ตารางที่ 4-3 เปรียบเทียบการใช้เวลาในการคำนวณผลของวงจรฟิลเตอร์แบงค์ทั้ง 3 แบบ บน FPGA SPARTAN-3 เบอร์ XC3S4000FG676-5

วงจรฟิลเตอร์แบงค์	**เวลาที่ใช้ใน 1 รอบของการคำนวณ 1 เอาท์พุต**	**เทียบกับวงจรแบบไม่มีการใช้ทรัพยากรร่วมกัน**
แบบที่ไม่มีการใช้ทรัพยากรร่วมกัน	1000 ns	(เป็นวงจรต้นแบบ)
แบบที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อก (Inter-block Sharing)	1300 ns	ช้าลง 30%
แบบที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อกและภายในบล็อก(Inter and Intra-block Sharing)	1300 ns	ช้าลง 30%

จากตารางเปรียบเทียบข้างต้นสามารถสรุปได้ว่าในการออกแบบวงจรฟิลเตอร์แบงค์บน FPGA ด้วยระเบียบวิธีการใช้ทรัพยากรในการประมวลผลร่วมกันนั้นสามารถลดปริมาณการใช้ทรัพยากรในการออกแบบได้เป็นปริมาณมากแม้ว่าจะต้องชดเชยด้วยเวลาการทำงานที่มากขึ้นแต่ผลจากการประมวลผลของวงจรก็ไม่ได้ผิดเพี้ยนไปจากวงจรที่ประมวลผลด้วยโปรแกรม MATLAB เลย ดังนั้นระเบียบวิธีการใช้ทรัพยากรร่วมกันในการออกแบบวงจรฟิลเตอร์แบงค์ในงานวิจัยนี้จึงเป็นอีกแนวทางที่น่าสนใจเพื่อใช้ในการออกแบบวงจรดิจิตอลขนาดใหญ่

บทที่ 5

บทสรุปและข้อเสนอแนะ

5.1 บทสรุป

งานวิจัยนี้เป็นการนำเสนอระเบียบวิธีในการออกแบบวงจรดิจิตอลขนาดใหญ่โดย งานวิจัยนี้จะใช้แนวทางการออกแบบดังกล่าวกับวงจรดิจิตอลฟิลเตอร์แบงก์ชนิด 8 แถบซึ่งเป็นวงจรที่ช่วยในการปรับปรุงเกี่ยวกับคุณภาพเสียงที่ใช้ในงานเกี่ยวกับเครื่องช่วยฟัง ทั้งนี้ที่เลือกใช้วงจรชนิดนี้เนื่องจากเมื่อศึกษาและหาข้อมูลอัลกอริทึมของวงจรพบว่าวงจรดิจิตอลฟิลเตอร์ที่กล่าวไปเป็นวงจรที่มีขนาดใหญ่และหากแนวทางการออกแบบดังกล่าวได้ผลตรงกับที่ต้องการแล้วนั้นในอนาคตอาจพัฒนาการออกแบบวงจรที่มีขนาดใหญ่ด้วยแนวทางที่นำเสนอนี้เพื่อลดปริมาณค่าใช้จ่ายด้านการออกแบบวงจรได้ด้วย ดังนั้นวงจรดิจิตอลฟิลเตอร์แบงค์จึงถูกนำมาใช้เพื่อศึกษาและออกแบบในงานวิจัยนี้ที่ต้องการนำเสนอแนวทางในการออกแบบวงจรขนาดใหญ่ภายใต้พื้นที่บน FPGA (Field Programmable Gate Array) ที่มีอยู่อย่างจำกัด โดยงานวิจัยจะแบ่งการออกแบบเป็นสองส่วนคือ ส่วนวงจรข้อมูล (Data-path part) สำหรับการประมวลผลสัญญาณข้อมูล และส่วนวงจรควบคุม (Control part) สำหรับควบคุมจังหวะการทำงานวงจรข้อมูล ซึ่งในส่วนวงจรข้อมูลก็คืออัลกอริทึมการทำงานของวงจรฟิลเตอร์แบงค์ซึ่งภายในวงจรรวมจะมีวงจรฟิลเตอร์ย่อยอีก 8 ตัว โดยที่ภายในฟิลเตอร์แต่ละตัวจะเป็นสมการของการบวก การคูณและการลบค่าของสัญญาณที่ผ่านเข้ามาในวงจรนั่นเองซึ่งมีสมการทั้งหมด 15 สมการประกอบด้วยตัวบวกจำนวน 38 ตัว ตัวคูณ 27 ตัวและตัวลบ 7 ตัว ซึ่งตรงนี้พบว่าหากสามารถลดปริมาณหน่วยของตัวคำนวณที่กล่าวไปลงได้จะเป็นการลดขนาดของวงจรที่ออกแบบด้วย ดังนั้นจึงนำวงจรรวมมาออกแบบโดยอาศัยหลักการใช้ทรัพยากรร่วมกันแบบลำดับชั้นเพื่อลดขนาดของวงจรโดยมีการคำนึงถึงเวลาการทำงานที่ช้าลงของวงจรด้วย

ส่วนวงจรควบคุม FSM (Finite State Machine) แบบ Moore machine ถูกใช้สำหรับควบคุมจังหวะการใช้ทรัพยากรร่วมกันของส่วนวงจรข้อมูลให้เป็นไปอย่างถูกต้อง วงจรที่ได้ออกแบบถูกทดสอบบนชิพ FPGA ของบริษัท Xilinx ตระกูล SPARTAN-3 เบอร์ XC3S4000-5FG676

ดังนั้นเริ่มจากเซทของฟังก์ชันหรือสมการที่อธิบายพฤติกรรมของวงจรถูกแปลงให้อยู่ในรูปกราฟกระแสข้อมูล (Data Flow Graph; DFG) ที่มีโครงสร้างเป็นลำดับชั้น จากนั้นพิจารณา DFG ที่เหมือนกันจากลำดับชั้นนอกสุดก่อนซึ่งก็คือการพิจารณาสมการทั้งหมดว่ามีสมการที่มีความเหมือนกันหรือไม่หากมีสมการใดเหมือนกันก็จะจัดไว้เป็นกลุ่มเดียวกัน จากแนวคิดนี้เมื่อนำไปใช้กับวงจรดิจิตอลฟิลเตอร์ในงานวิจัยนี้พบว่าสามารถจัดกลุ่มได้เป็น 4 กลุ่มที่เหมือนกันในลำดับวงจรชั้นนอกสุดหรือในงานวิจัยนี้เรียกว่าการใช้ทรัพยากรร่วมกันระหว่างบล็อก (Inter-block sharing) หากออกแบบวงจรใหม่ด้วยการใช้ทรัพยากรร่วมกันนี้พบว่าจะมีการปริมาณตัวประมวลผลทางคณิตศาสตร์ในวงจรจะลดลงและเมื่อทำการสังเคราะห์วงจรบนโปรแกรม Xilinx เพื่อยืนยันแนวคิดพบว่ามีการใช้เนื้อที่บน FPGA ลดลงถึง 38% หลังจากมีการใช้ทรัพยากรร่วมกันแบบดังกล่าวแต่ในงานวิจัยนี้จะพิจารณาการใช้ทรัพยากรร่วมกันในอีกหนึ่งระดับ(Intra-block sharing) กล่าวคือจะพิจารณาเข้าไปที่สมการว่าสามารถจัดกลุ่มการคำนวณได้อีกหรือไม่ แต่ทั้งนี้จะพิจารณาการใช้ทรัพยากรร่วมกันในระดับนี้จะต้องไม่มีผลกระทบต่อการทำงานของวงจรรวมและไม่นำกลุ่มฟังก์ชันที่ใช้การออกแบบแบบ Inter-block ไปแล้วมาพิจาณาอีก ซึ่งในงานวิจัยนี้พบว่าจะใช้สามารถแนวคิดดังกล่าวในการออกแบบได้ด้วยและเมื่อทำการสังเคราะห์วงจรพบว่าปริมาณการใช้เนื้อที่บน FPGA จะลดลงจากวงจรที่ไม่ใช้ทรัพยากรร่วมกันถึง 44 %

การตรวจสอบว่าการลดการใช้ทรัพยากรในการออกแบบก็ทำการป้อนค่าสัญญาณซึ่งถูกแปลงเป็นชุดของตัวเลขที่เป็นเลขฐานสองขนาด 32 บิตเป็นอินพุตให้แก่วงจรทุกรูปแบบที่ทำการทดลองบน FPGA โดยจะเทียบความถูกต้องกับค่าเอาท์พุตที่ได้จากการออกแบบการทำงานของวงจรฟิลเตอร์แบงค์ด้วยโปรแกรม MATLAB ซึ่งพบว่าวงจรที่ออกแบบบน FPGA ทั้งแบบที่ไม่มีการใช้ทรัพยากรร่วมกัน แบบใช้ทรัพยากรร่วมกันระหว่างบล๊อกและแบบที่มีการใช้ทรัพยากรร่วมกันระหว่างบล็อกและภายในบล็อก เมื่อประมวลผลเรียบร้อยแล้วจะให้ค่าเอาท์พุตที่เหมือนกันกับวงจรที่ประมวลผลด้วย MATLAB

5.2 ข้อเสนอแนะ

1. เนื่องจากงานวิจัยนี้เป็นการทดสอบวงจรที่ออกแบบด้วยวิธีที่ต้องการนำเสนอในการทำการทดลองจึงต้องแสดงให้เห็นถึงการเปลี่ยนแปลงผลจากการสังเคราะห์วงจรทั้ง 3 รูปแบบในการทดสอบวงจรบน FPGA และงานวิจัยนี้ไม่ได้ทำการทดสอบด้วยการ Implement ลงบนบอร์ดจริง จึงจะทำการออกแบบด้วยชิพเบอร์เดียวกัน ในงานวิจัยนี้จึงเลือกใช้รุ่น SPARTAN-3 เบอร์ XC3S4000-5FG676 ทั้งนี้เพราะหากใช้เบอร์ต่ำจากนี้วงจรแบบที่ไม่มีการใช้ทรัพยากรร่วมกันจะใช้เนื้อที่มากกว่าที่มีบน FPGA ดังนั้นหากจะพิจารณาเฉพาะวงจรที่มีการใช้ทรัพยากรร่วมกันหรือต้องการทำการทดสอบลงบน FPGA จริงก็สามารถทำการเลือกเบอร์ FPGA ที่ต่ำกว่านี้ได้ เช่น SPARTAN-3 เบอร์ XC3s1500-5FG676 หรือ SPARTAN-3 เบอร์ XC3S2000-5FG676 เป็นต้น

2. วงจรคูณและวงจรบวกที่ใช้ในการทดสอบจะใช้ในการประมวลผลจำนวนทศนิยมแบบ 32 บิตเท่านั้น หากต้องการประมวลผลจำนวนทศนิยมในรูปแบบอื่นๆ เช่น 24 บิต หรือ 64 บิต นั้นก็จะจำเป็นต้องทำการออกแบบในส่วนของวงจรคูณและวงจรบวกนี้ก่อน เพื่อความถูกต้องของค่าที่ได้จากการคำนวณ

3. เนื่องจากการทำงานของโปรแกรม Xilinx ISE 8.1i นั้น เมื่อมีการใช้งานพร้อมกันกับโปรแกรมอื่น อาจเกิดกรณีที่ตัวโปรแกรมใช้ทรัพยากรของหน่วยความจำในเครื่อง PC มากเกินไปจนทำให้เครื่องเกิดอาการแฮงก์ (Hang) หรือ เกิดรีสตาร์ทเครื่องขึ้นมาใหม่ ส่งผลทำให้ไฟล์ข้อมูลอาจเสียหาย หรือ สูญหายขึ้นได้ ดังนั้นเพื่อความปลอดภัยและความสะดวกในการทำงานของโปรแกรมนี้ ควรตรวจสอบในส่วนของหน่วยความจำ Ram (Random Access Memory) ของเครื่องคอมพิวเตอร์นั้นก่อนโดยควรให้มีจำนวนสูงไว้ จะได้ไม่เกิดปัญหาดังกล่าวข้างต้น

บรรณานุกรม

ชำนาญ ปัญญาใส และ วัชรากร หนูทอง. 2547. ภาษา VHDL สำหรับการออกแบบวงจรดิจิตอล. กรุงเทพฯ: ซีเอ็ดยูเคชั่น.

วุฒิ วิริยะสม. 2550.การออกแบบหน่วยประมวลผลคณิตศาสตร์ความเร็วสูงสำหรับวงจรกรองปรับตัวบน FPGAs . วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์.

พรชัย ภววงษ์ศักดิ์.2543.การประมวลผลสัญญาณดิจิตอลเบื้องต้น.(หนังสือนี้แจกฟรีผู้อ่านสามารถหาหนังสือนี้อ่านได้ทางอินเตอร์เนตที่ http://www.ee.mut.ac.th/home/pornchai).

Taewhan Kim, Noritake Yonezawa and Jane W.S. Liu. " A scheduling algorithm for conditional resource sharing ". 1994: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume 13, Issue 4, 425-438.

Chang E., Gajski D.D., Narayan S.,. "An optimal clock period selection method based on slack minimization criteria". ACM Transactions on Design Automation of Electronic Systems, Volume 1 , Issue 3 (July 1996), 352 – 370.

S.O. Memik, G. Memik, R. Jafari, E. Kursun,. "Global resource sharing for synthesis of control data flow graphs on FPGAs". Proceedingsof Design Automation Conference 2003, pp. 604-609, 2-6 June 2003.

C. Jaschke, R. Laur,. "Resource constrained modulo scheduling with global resource sharing". Proceedings of 11th International Symposium on System Synthesis 1998, pp. 60-65, 2-4 Dec. 1998.

C. Jaschke, F. Beckmann, R. Laur, "Time constrained module scheduling with global resource sharing". Proceedings of Design, Automation and Test in Europe Conference and Exhibition 1999, pp. 210-216, 9-12 March 1999.

L.S. Nielsen and J. Sparso,. Designing asynchronous circuits for low power : an IFIR filter bank for digital hearing aid". Proceedings of the IEEE vol. 87(2), February 1999, pp: 268-281.

T. Lunner, J. Hellgren,. "A digital filterbank hearing-aid design, implementation and evaluation". International Conference on Acoustics, Speech, and Signal Processing, ICASSP-91, 14-17 April 1991, vol.5, 3661-3664.

Y. Lin, P.P. Vaidyanathan,. “Application of DFT filter banks and cosine modulated filter banks in filtering”. IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS94), 5-8 Dec, 1994, 254 - 259.

R. Bernardini, R. Rinaldo,. “Oversampled filter banks from extended perfect reconstruction filter banks”. IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 54(7), July 2006, 2625 - 2635.

B. Dautrich, L. Rabiner,. T. Martin, “On the use of filter bank features for isolated word recognition”. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP ’83. Vol.8, April 1983, 1061 - 1064.

R. Bregovic, T. Saramaki,. “An efficient approach for designing nearly perfect-reconstruction low-delay cosine-modulated filter banks”. IEEE International Symposium on Circuits and Systems, ISCAS 2002. Vol.1, 26-29 May 2002, I-825 - I-828.

T. Uto, M. Okuda, M. Ikehara, S.Takahashi,. “Image coding using wavelets based on two-channel linear phase orthogonal IIR filter banks”. International Conference on Image Processing,ICIP 99. Vol. 2, 24-28 Oct. 1999, 265 - 268.

G. De Micheli, “Synthesis and Optimization of Digital Circuits”, McGraw-Hill, 1994.

http://en.wikipedia.org/wiki/Filterbank

Xilinx Corp. http://www.xilinx.com.

ภาคผนวก

Design of a filter bank based on hierarchical datapath resource-sharing

Wiwat Bunsung[1] Nattha Jindapetch[2] Pornchai Phukpattranont[3] Kanadit Chetpattananondh[4]

[1,2,3,4]Department of Electrical Engineering, Faculty of Engineering, Prince of Songkla University, Hat Yai, Songkhla 90112

E-mail: zenki_2911@hotmail.com[1] nattha.s@psu.ac.th[2] pornchai.p@psu.ac.th[3] kanadit.c@psu.ac.th[4]

ABSTRACT

This article describes a methodology for design a digital filter bank on an FPGA with limited area. From the set describing circuit behavior, the method translates it into the data flow graph (DFG) representation. Then, the method analyze to search and to assign the groups of same functions hierarchically share the same resources from big to small groups while considering the slower speed and the wiring complexity that come from resource-sharing effects. The last step is determining the control circuit and the optimal clock period. The designed circuit is about 50% area reduced, while only 1.1 time speed reduced.

Keywords

resource-sharing, digital filter bank, FPGA (Field Programmable Gate Array)

1. INTRODUCTION

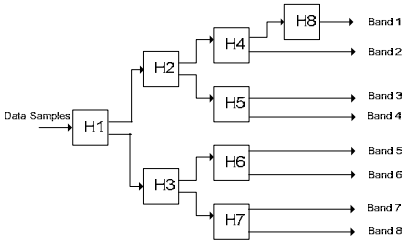
Currently, Field Programmable Gate Array (FPGA) is used as an important component in the design of digital circuit.this is due to the fact that all of circuits can be designed and simulated on PCs. Then, we can program the designed circuit on the chip, which can work immediately. So, FPGA is appropriate for the design of large digital circuit .Filter banks have the advantages of separating a signal into different frequency ranges in many modern signal processing applications [1] - [7]. The separation into sub-band components is intended to make further processing more convenient**.**

This article describes a methodology for designing a digital filter bank with hierarchical resource sharing on an FPGA with limited area while considering the slower speed and the wiring complexity that come from resource-sharing effects. The set proposed into the data flow graph (DFG) representation. Then, the analysis is performed to search and assign the groups of same functions hierarchically in order to share the same resources from big to small groups. Finally, the digital filter bank circuits proposed designed by hierarchical resource sharing method are implemented and correctly tested on FPGA.

2. A DIGITAL FILTER BANK

A filter bank is an array of band-pass filters that separates the input signal into several components, each one carrying a single frequency sub-band of the original signal. It also is desirable to design the filter bank in such a way that subbands can be recombined to recover original signal. The filter bank serves to isolate different frequency components in a signal. This is useful because for most application some frequencies are more important than others. For example these important frequencies can be coded with a fine resolution. Small differences at these frequencies are significant and a coding scheme that preserves these differences must be used. On the other hand, less important frequencies do not have to be exact. A coarser coding scheme can be used, even though some of the finer details will be lost in the coding [9].

Input specifications of the method proposed in this paper is a set of functions representing a DSP system. They may be described in VHDL or Verilog HDL. Here, it is manually translated to be a corresponding DFG [8]. For example, Fig. 1 shows a 8-band filter bank block diagram and functions [2]. Digital filter banks are widely used in digital hearing aid applications. There are nine processing blocks with the corresponding functions. In each block, yi and yic are outputs, xi is a data sample x(n–i), and hi is system a coefficient. Fig. 2(a) shows the corresponding top level DFG, and Fig. 2(b) shows a corresponding DFG of block H3.



$H1(z)$: $y_1(n) = (x_0 + x_{48})*h_0 + (x_{16} + x_{32})*h_1 + x_{24}*h_2$
$y_{1c}(n) = x_{24} - y_1(n)$

$H2(z)$: $y_2(n) = (x_0 + x_{24})*h_0 + (x_8 + x_{16})*h_1 + x_{12}*h_2$
$y_{2c}(n) = x_{12} - y_2(n)$

$H3(z)$: $y_3(n) = (x_0 + x_{28})*h_0 + (x_4 + x_{24})*h_1 + (x_8 + x_{20})*h_2 + (x_{12} + x_{16})*h_3 + x_{14}*h_4$
$y_{3c}(n) = x_{14} - y_3(n)$

$H4(z)$: $y_4(n) = (x_0 + x_{12})*h_0 + (x_4 + x_8)*h_1 + x_6*h_2$
$y_{4c}(n) = x_6 - y_4(n)$

$H5(z)$: $y_5(n) = (x_0 + x_{10})*h_0 + (x_2 + x_8)*h_1 + (x_4 + x_6)*h_2 + x_5*h_3$
$y_{5c}(n) = x_5 - y_5(n)$

$H6(z)$: $y_6(n) = (x_0 + x_6)*h_0 + (x_2 + x_4)*h_1 + x_3*h_2$
$y_{6c}(n) = x_3 - y_6(n)$

$H7(z)$: $y_7(n) = (x_0 + x_{30})*h_0 + (x_6 + x_{24})*h_1 + (x_{12} + x_{18})*h_2 + x_{15}*h_3$
$y_{7c}(n) = x_{15} - y_7(n)$

$H8(z)$: $y_8(n) = (x_0 + x_2)*h_0 + x_1*h_1$

Figure 1: A 8-band filter bank [2].

3. HIERARCHICAL RESOURCE-SHARING

Because of the limited area, the available resources such as functional units (such as multipliers, adder) and registers must be shared by many operations. The following subsections explain a method for resource-sharing decision, inter-block sharing and intra-block sharing. Here an asynchronous style ASAP (As Soon As Possible) scheduling [10] is applied regardless of clock period (control step) to obtain the best speed under the resource-constraints.

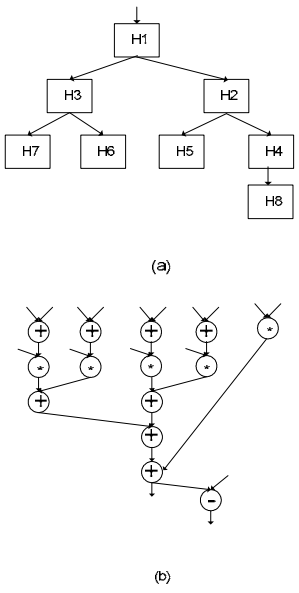


Figure 2: A DFG of 7-band filter bank of Fig 1: (a)The top-level DFG, and (b) a DFG of block H3.

3.1 Inter-block Sharing

Inter-block sharing means the resource-sharing among the blocks at the top level DFGs. The resource-sharing between any two blocks is allowed when the functions in both blocks are the same. In the other word, if any two or more blocks have the same DFG structure, they can share the same DFG block. For example, the functions of Fig. 1 can be classified into 4 DFG structures. Structure a corresponds to blocks H1, H2, H4, and H6. Structure b corresponds to block H3. Structure c corresponds to blocks H5 and H7. Structure d corresponds to blocks H8. Therefore, there are 4 DFGs to be used.Under the resource-constraints the asynchronous style ASAP scheduling is applied to obtained the best speed result.The ASAP scheduling under the resource-constraint of 4 DFGs is shown in Fig. 3(a). Here there are doted arcs inserted between the same DFG structure. These doted arcs represent the resource-constrained relations between two DFGs, i.e. the successor DFG can be started only after the predecessor DFG has been completed.

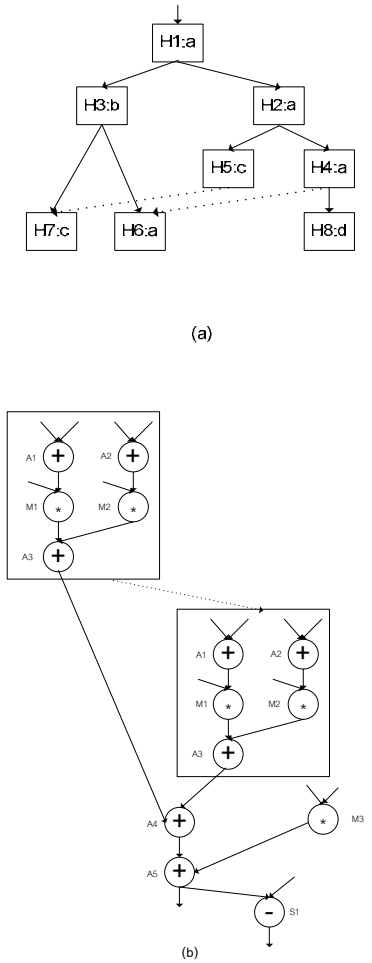


Figure 3: A DFG of Fig.1 and the corresponding resource sharing: (a) Inter-block sharing and (b)Intra-block sharing of H3.

3.2 Intra-block Sharing

Intra-block sharing means the resource-sharing among the same operations inside the same DFG. To avoid the complicated overheads of interconnect and multiplexors, the intrablock sharing is allowed when a DFG block is not shared by other blocks. For example, from Fig. 2, the intra-block sharing is allowed for only H3.The intra-block sharing solutions depend on the time interval calculated from the following method. Each DFG of Fig. 3(a) is synthesized into a logic circuit. Then the static timing analysis is performed for the logic circuit to obtain the maximum execution delay of each DFG.From the top-level DFG of Fig. 3(a), the earliest start time and the earliest end time of each DFG block is calculated as shown in Fig. 4. Now, we can observe that the time interval the allow the resource-sharing feature of H3 is 78 ns. It is the time that H3 is idle before H7 can start. As a result, the resource-sharing solution inside block H3 is shown in Fig. 3(b). Similar to inter-block sharing, the ASAP scheduling is applied. The maximum path delay of this solution should be less than the allowed time interval.

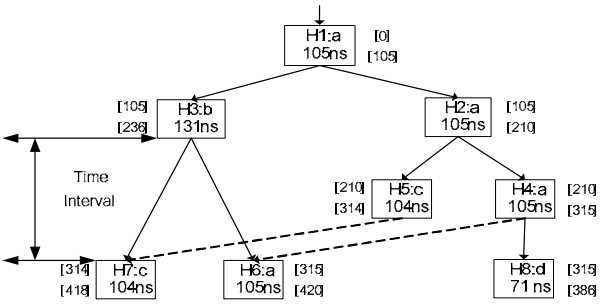


Figure 4: Resource-Sharing time interval calculation.

4. EXPERIMENTAL RESULTS

An experiment has been performed to show the effect of the proposed resource-sharing method. VHDL is used to explain the circuit behaviors. The circuits are synthesized and time-analyzed from Xilinx 8.1i [12]. Xilinx Virtex 2 : xc2v1000fg256 is selected as a target device. The synthesis results are shown in Table 1.

Table 1: Synthesis results.

	Without Sharing	Only inter-	Inter- and intra-
# of Slices	6223	4572	4516
# of FFs	3180	3178	3234
# of LUTs	11067	6958	6637
# of MULs	27	14	12
DFG time	386	420	422

In Table 1, the second column shows the results of the circuit without sharing. The third column shows the results of the circuit applied only inter-block sharing. The fourth column shows the results of the circuit applied both inter-block and intra-block sharing. We can observe that the number of resources needed for the circuit applied both inter-block and intra-block sharing was reduced about 50% while the speed is reduced only 1.1 time slower compared to the circuit without sharing.

5. CONCLUSIONS

A methodology for design a digital filter bank on an FPGA with limited area has been proposed. From the set describing circuit behavior, the method translates it into the data flow graph (DFG) representation. Then, the method analyze to search and to assign the groups of same functions hierarchically share the same resources from big to small groups while considering the slower speed and the wiring complexity that come from resource-sharing effects. The designed circuit is about 50% area reduced, while only 1.1 time speed reduced.

6. REFERENCES

[1] L.S. Nielsen, J. Sparso, “Designing asynchronous circuits for low power: an IFIR filter bank for a digital hearing aid”, Proceedings of the IEEE Volume 87, Issue2, Feb. 1999 Page(s):268 - 281.

[2] T. Lunner, J. Hellgren, “A digital filterbank hearing aid-design, implementation and evaluation”, International Conference on Acoustics, Speech, and Signal Processing,ICASSP-91, 14-17 April 1991 Page(s):3661- 3664 vol.5.

[3] Y. Lin, P.P. Vaidyanathan, “Application of DFT filter banks and cosine modulated filter banks in filtering”, IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS94), 5-8 Dec, 1994, Page(s):254 - 259.

[4] R. Bernardini, R. Rinaldo, “Oversampled filter banks from extended perfect reconstruction filter banks”, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 54(7), July 2006, Page(s):2625 - 2635.

[5] B. Dautrich, L. Rabiner, T. Martin, “On the use of filter bank features for isolated word recognition”, IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP ’83. Vol.8, April 1983, Page(s):1061 - 1064.

[6] R. Bregovic, T. Saramaki, “An efficient approach for designing nearly perfect-reconstruction low-delay cosine-modulated filter banks”, IEEE International Symposium on Circuits and Systems, ISCAS 2002. Vol.1, 26-29 May 2002, Page(s):I-825 - I-828.

[7] T. Uto, M. Okuda, M. Ikehara, S.Takahashi, “Image coding using wavelets based on two-channel linear phase orthogonal IIR filter banks”, International Conference on Image Processing,ICIP 99. Vol. 2, 24-28 Oct. 1999, Page(s):265 - 268.

[8] G. De Micheli, “Synthesis and Optimization of Digital Circuits”, McGraw-Hill, 1994.

[9] http://en.wikipedia.org/wiki/Filterbank

[10] B. Bachman, H. Zheng, and C. Myers, “Architectural Synthesis of Timed Asynchronous Systems”, In IEEE International Conference on Computer Design (ICCD), October, 1999.
[11] N. Jindapetch, H. Saito, K. Thongnoo, and T. Nanya, “ Determination of Worst-Case Independent Clock Periods for Resource-Constrainted Systems”, Proc. Of the 2006 Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI) International Conference (ECTI-CON 2006), vol. 1, Ubon Ratchathani, Thailand, May 10-13, 2006,pp.344-347.
[12] http://www.xilinx.com

Design of a filter bank based on hierarchical datapath resource-sharing

Wiwat Bunsung[1] Nattha Jindapetch[2] Pornchai Phukpattranont[3] Kanadit Chetpattananondh[4]

[1,2,3,4]Department of Electrical Engineering, Faculty of Engineering, Prince of Songkla University, Hat Yai, Songkhla 90112

E-mail: zenki_2911@hotmail.com[1] nattha.s@psu.ac.th[2] pornchai.p@psu.ac.th[3] kanadit.c@psu.ac.th[4]

1. INTRODUCTION

Filter banks have the advantages of separating a signal into different frequency ranges in many modern signal processing applications [1] - [6]. The separation into sub-band components is intended to make further processing more convenient. However, this is intended to required larger circuits. This article describes a methodology for design a digital filter bank on an FPGA with limited area. From the set describing circuit behavior, the method translates it into the data flow graph (DFG) representation. Then, the method analyze to search and to assign the groups of same functions hierarchically share the same resources from big to small groups while considering the slower speed and the wiring complexity that come from resource-sharing effects. The designed circuit is about 50% area reduced, while only 1.1 time speed reduced.

2. A DIGITAL FILTER BANK

Input specifications of the method proposed in this paper is a set of functions representing a DSP system. They may be described in VHDL or Verilog HDL. Here, it is manually translated to be a corresponding DFG. Fig.1(a) shows the corresponding top level DFG, and Fig.1(b) shows a corresponding DFG of block H3.

3. HIERARCHICAL RESOURCE-SHARING

Because of the limited area, the available resources such as functional units (such as multipliers, adder) and registers must be shared by many operations. We explain a method for resource-sharing decision, inter-block sharing and intra-block sharing. Here an asynchronous style ASAP (As Soon As Possible) scheduling [10] is applied regardless of clock period (control step) to obtain the best speed under the resource-constraints.

3.1 Inter-block Sharing

Inter-block sharing means the resource-sharing among the blocks at the top level DFGs. The resource-sharing between any two blocks is allowed when the functions in both blocks are the same. In the other word, if any two or more blocks have the same DFG structure, they can share the same DFG block. For example, the functions of 8-band filter bank can be classified into 4 DFG structures. Structure a corresponds to blocks H1, H2, H4, and H6. Structure b corresponds to block H3. Structure c corresponds to blocks H5 and H7. Structure d corresponds to block H8. Therefore, there are 4 DFGs to be used. The ASAP scheduling under the resource-constraint of 4 DFGs is shown in Fig.1(a), Here there are doted arcs inserted between the same DFG structure. These doted arcs represent the resource-constrained relations between two DFGs, i.e. the successor DFG can be started only after the predecessor DFG has been completed.

3.2 Intra-block Sharing

Intra-block sharing means the resource-sharing among the same operations inside the same DFG. To avoid the complicated overheads of interconnect and multiplexors, the intra-block sharing is allowed when a DFG block is not shared by other blocks. For example, from Fig.1, the intra-block sharing is allowed for only H3. The intra-block sharing solutions depend on the time interval calculated from the following method. Each DFG of Fig.1(a) is synthesized into a logic circuit. Then the static timing analysis is performed for the logic circuit to obtain the maximum execution delay of each DFG. From the top-level DFG of Fig.1(a), the earliest start time and the earliest end time of each DFG block is calculated as shown in Fig.1(a). Now, we can observe that the time interval the allow the resource-sharing feature of H3 is 78 ns. It is the time that H3 is idle before H7 can start. As a result, the resource-sharing solution inside block H3 is shown in Fig.1(b). Similar to inter-block sharing, the ASAP scheduling is applied. The maximum path delay of this solution should be less than the allowed time interval.

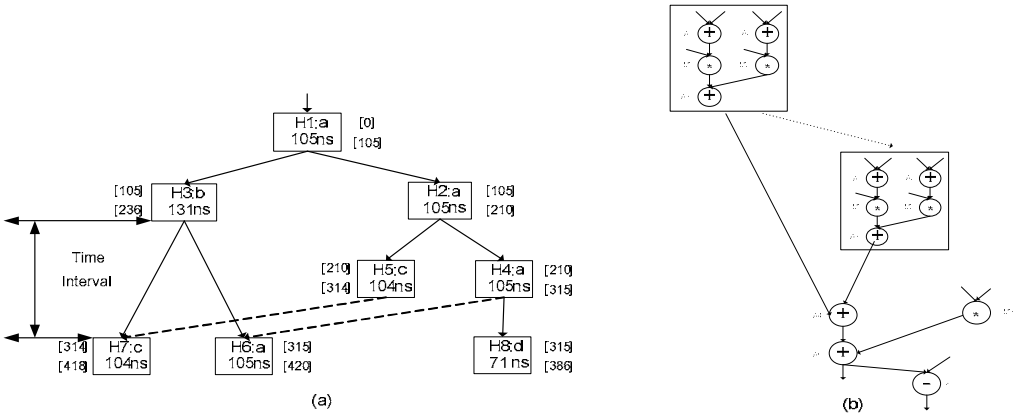


Figure 1. A DFG of Filter bank and the corresponding resource sharing (a) Inter-block sharing and Resource-Sharing time interval calculation. and (b) Intra-block sharing of H3.

1. EXPERIMENTAL RESULTS

An experiment has been performed to show the effect of the proposed resource-sharing method. VHDL is used to explain the circuit behaviors. The circuits are synthesized and time-analyzed from Xilinx 8.1i [12]. Xilinx Virtex 2 : xc2v1000fg256 is selected as a target device. The synthesis results are shown in Table 1.

Table 1: Synthesis results.

	Without Sharing	Only inter-	Inter- and intra-
# of Slices	6223	4572	4516
# of FFs	3180	3178	3234
# of LUTs	11067	6958	6637
# of MULs	27	14	12
DFG time	386	420	422

In Table 1, We can observe that the number of resources needed for the circuit applied both inter-block and intra-block sharing was reduced about 50% while the speed is reduced only 1.1 time slower compared to the circuit without sharing.

2. CONCLUSIONS

A methodology for design a digital filter bank on an FPGA with limited area has been proposed. From the set describing circuit behavior, the method translates it into the data flow graph (DFG) representation. Then, the method analyze to search and to assign the groups of same functions hierarchically share the same resources from big to small groups while considering the slower speed and the wiring complexity that come from resource-sharing effects. The designed circuit is about 50% area reduced, while only 1.1 time speed reduced.

3. REFERENCES

[1] L.S. Nielsen, J. Sparso, “Designing asynchronous circuits for low power: an IFIR filter bank for a digital hearing aid”, Proceedings of the IEEE Volume 87, Issue2, Feb. 1999 Page(s):268 - 281.

[2] T. Lunner, J. Hellgren, “A digital filterbank hearing aid-design, implementation and evaluation”, International Conference on Acoustics, Speech, and Signal Processing,ICASSP-91, 14-17 April 1991 Page(s):3661- 3664 vol.5.

[3] Y. Lin, P.P. Vaidyanathan, “Application of DFT filter banks and cosine modulated filter banks in filtering”, IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS94), 5-8 Dec, 1994, Page(s):254 - 259.

[4] B. Dautrich, L. Rabiner, T. Martin, “On the use of filter bank features for isolated word recognition”, IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP ’83. Vol.8, April 1983, Page(s):1061 - 1064.

[5] R. Bregovic, T. Saramaki, “An efficient approach for designing nearly perfect-reconstruction low-delay cosine-modulated filter banks”, IEEE International Symposium on Circuits and Systems, ISCAS 2002.Vol.1, 26-29 May 2002,Page(s):I-825 - I828.

[6] T. Uto, M. Okuda, M. Ikehara, S.Takahashi, “Image coding using wavelets based on two-channel linear phase orthogonal IIR filter banks”, International Conference on Image Processing,ICIP 99. Vol. 2, 24-28 Oct. 1999, Page(s):265 - 268.

[7] G. De Micheli, “Synthesis and Optimization of Digital Circuits”, McGraw-Hill, 1994.

[8] B. Bachman, H. Zheng, and C. Myers, “Architectural Synthesis of Timed Asynchronous Systems”, In IEEE International Conference on Computer Design (ICCD), October, 1999.

[9] http://www.xilinx.com

ประวัติผู้เขียน

ชื่อ สกุล นายวิวัฒน์ บุญสูง

รหัสประจำตัวนักศึกษา 4712069

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วิศวกรรมศาสตรบัณฑิต (วิศวกรรมไฟฟ้า-ไฟฟ้าสื่อสาร)	มหาวิทยาลัยสงขลานครินทร์	2547

ทุนการศึกษา (ที่ได้รับในระหว่างการศึกษา)

ทุนมูลนิธิเพื่อการศึกษาคอมพิวเตอร์และการสื่อสาร ประจำปีการศึกษา 2547

การตีพิมพ์เผยแพร่ผลงาน

Wiwat Bunsung, Nattha Jindapetch, Pornchai Phukpattranont, and Kanadit Chetpattananondh, “Design of a filter bank based on hierarchical datapath resource-sharing”, ANSCSE11,Phuket,Thailand, March 28-30,2007, pp.175-176