การประเมินการสร้างจริงบน FPGA ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท SIGN-SIGN ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง

นายสมบูรณ์ กลิ่นจันทร์กลั่น

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2545

ISBN 974-17-9759-1

EVALUATION OF THE FPGA IMPLEMENTATION OF THE SIGN-SIGN LEAST-MEAN-SQUARE ALGORITHM EMPLOYING THE SQUARED ERROR STEP-SIZE ADAPTATION

Mr. Somboon Klinchanklan

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Electrical Engineering

Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University

Academic Year 2002

ISBN 974-17-9759-1

หัวข้อวิทยานิพนธ์	การประเมินการสร้างจริงบน FPGA ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท SIGN-SIGN ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง
โดย	นายสมบูรณ์ กลิ่นจันทร์กลั่น
สาขาวิชา	วิศวกรรมไฟฟ้า
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.เจษฎา ชินรุ่งเรือง

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

…….. คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.สมศักดิ์ ปัญญาแก้ว)

คณะกรรมการสอบวิทยานิพนธ์

…… ประธานกรรมการ
(รองศาสตราจารย์ ดร.เอกชัย ลีลารัศมี)

……... อาจารย์ที่ปรึกษา
(ผู้ช่วยศาสตราจารย์ ดร.เจษฎา ชินรุ่งเรือง)

……กรรมการ
(รองศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

……กรรมการ
(อาจารย์ ดร.สุภาวดี อร่ามวิทย์)

สมบูรณ์ กลิ่นจันทร์กลั่น : การประเมินการสร้างจริงบน FPGA ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท SIGN-SIGN ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง (EVALUATION OF THE FPGA IMPLEMENTATION OF THE SIGN-SIGN LEAST-MEAN-SQUARE ALGORITHM EMPLOYING THE SQUARED ERROR STEP-SIZE ADAPTATION) อ. ที่ปรึกษา : ผศ.ดร.เจษฎา ชินรุ่งเรือง, 95 หน้า. ISBN 974-17-9759-1.

เครื่องช่วยฟังเป็นอุปกรณ์ที่ใช้สำหรับช่วยเพิ่มประสิทธิภาพทางการได้ยินของผู้ที่มีปัญหาในการรับรู้ข่าวสารทางเสียง ปัญหาหลักอย่างหนึ่งที่เกิดขึ้นกับผู้ใช้เครื่องช่วยฟังคือ การเกิดเสียงหอนอันเนื่องมาจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง วิธีการแก้ปัญหาดังกล่าวที่มีประสิทธิภาพและใช้กันโดยทั่วไปคือ การใช้วงจรกรองปรับตัวสร้างสัญญาณเลียนแบบสัญญาณป้อนกลับเพื่อหักล้างกับสัญญาณป้อนกลับที่เกิดขึ้นจริง โดยการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวดังกล่าว โดยทั่วไปจะอาศัยขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดแบบดั้งเดิม

เนื่องจากความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดแบบดั้งเดิมค่อนข้างสูง วิทยานิพนธ์นี้ศึกษาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท SIGN-SIGN เพื่อใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวแทนขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดแบบดั้งเดิม เพื่อเพิ่มประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท SIGN-SIGN การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองได้นำมาประยุกต์ใช้ ขั้นตอนวิธีที่ได้ดัดแปลงใหม่นี้มีความซับซ้อนทางด้านการคำนวณต่ำแต่มีประสิทธิภาพดีพอสำหรับแก้ปัญหาการเกิดเสียงหอน เพื่อศึกษาความเป็นไปได้ของการใช้งาน ขั้นตอนวิธีที่เสนอได้ถูกนำมาสร้างจริงบน FPGA พร้อมทั้งประเมินความซับซ้อนทางฮาร์ดแวร์

ภาควิชา วิศวกรรมไฟฟ้า ลายมือชื่อนิสิต.....

สาขาวิชา วิศวกรรมไฟฟ้า ลายมือชื่ออาจารย์ที่ปรึกษา.....

ปีการศึกษา 2545

4370662321 : MAJOR ELECTRICAL ENGINEERING

KEY WORD: FPGA / VHDL / SIGN-SIGN LMS / STEP SIZE

SOMBOON KLINCHANKLAN : EVALUATION OF THE FPGA IMPLEMENTATION OF THE SIGN-SIGN LEAST-MEAN-SQUARE ALGORITHM EMPLOYING THE SQUARED ERROR STEP-SIZE ADAPTATION. THESIS ADVISOR : ASSIST. PROF. DR.CHEDSADA CHINRUNGRUENG, Ph.D. 95 pp. ISBN 974-17-9759-1.

An hearing aid is an equipment used for amplifying the audio signal in order to enhance the hearing efficiency of people with hearing impairment. One major problem that hearing aid users usually encounter is screeching sound, which results from acoustic feedback in hearing aid. One effective and widely used solution for solving this problem is to employ an adaptive filter to produce signal for canceling out the acoustic feedback signal. Usually, the traditional Least-Mean-Square (LMS) algorithm is employed to adapt the coefficients of the adaptive filter.

As the computation complexity of the traditional LMS algorithm is rather high, this thesis investigates the use of the sign-sign LMS algorithm to substitute the traditional LMS algorithm. In order to increase the adaptation capability of the sign-sign LMS algorithm, its step-size is adjusted using the squared error step-size adjustment. This modification results in the adaptive filter which is computational-wise simple, yet comparatively efficient for solving the acoustic feedback cancellation in hearing aids. To investigate its practicality, the sign-sign LMS algorithm with the squared error step-size adjustment is then implemented on FPGA chip and its hardware complexity is then evaluated.

Department Electrical Engineering Student's signature.....

Field of study Electrical Engineering Advisor's signature.....

Academic year 2002

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้เกิดขึ้นและสำเร็จลุล่วงไปได้ด้วยคำแนะนำและความช่วยเหลืออย่างดียิ่งของผู้ช่วยศาสตราจารย์ ดร.เจษฎา ชิ่นรุ่งเรือง อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งกรุณาให้คำแนะนำและความช่วยเหลือทางด้านความรู้ในการทำวิจัยมาโดยตลอด

ขอขอบคุณพี่ ๆ งานทุนการศึกษาของสำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ ที่คอยดูแลเรื่องทุนการศึกษาอย่างดีมาโดยตลอด

ขอขอบคุณ พี่ ๆ ทีมงานของศูนย์พัฒนาธุรกิจออกแบบวงจรรวมที่ให้ความรู้และความช่วยเหลือในการใช้งานอุปกรณ์ในการออกแบบและทดสอบวงจร

ขอขอบคุณห้องปฏิบัติการวิจัยกรรมวิธีสัญญาณดิจิทัล ซึ่งเป็นสถานที่ทำวิจัย รวมถึงเพื่อน รุ่นพี่และรุ่นน้องทุกท่านที่มีส่วนช่วยเหลือในการให้ข้อคิดเห็น และคำแนะนำต่างๆ

สุดท้ายนี้ ผู้วิจัยขอกราบขอบพระคุณบิดามารดา ที่ได้มอบความรัก ความอบอุ่น และกำลังใจตลอดมา ตลอดจนพี่ๆ ของข้าพเจ้าที่ให้การสนับสนุนเสมอมา

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

หน้า

สารบัญรูป

หน้า

สารบัญรูป (ต่อ)

หน้า

สารบัญรูป (ต่อ)

หน้า

สารบัญรูป (ต่อ)

หน้า

บทที่ 1

บทนำ

เครื่องช่วยฟัง (Acoustic Hearing Aids) เป็นอุปกรณ์ที่ใช้สำหรับช่วยเพิ่มประสิทธิภาพทางการได้ยินของผู้ที่ปัญหาในการรับรู้ข่าวสารทางเสียง เครื่องช่วยฟังถูกสร้างขึ้นมาให้มีขนาดเล็กเพื่อที่จะสามารถใส่เข้าไปในรูหูหรือติดไว้ข้าง ๆ ใบหูของคนเรา ทำให้ผู้ใช้เครื่องช่วยฟังรู้สึกมั่นใจกับเครื่องช่วยฟังที่มีขนาดเล็ก ซึ่งจะทำให้ไม่เป็นที่สังเกตของบุคคลอื่น การที่เครื่องช่วยฟังถูกสร้างให้มีขนาดเล็กทำให้เกิดข้อจำกัดที่ว่า การประมวลผลสัญญาณของเครื่องช่วยฟังทางเสียงต้องไม่ซับซ้อนเกินไป เพื่อให้วงจรประมวลผลมีขนาดเล็กและใช้พลังงานน้อย (ทำให้ไม่ต้องใช้แบตเตอรี่ขนาดใหญ่และไม่ต้องเปลี่ยนแบตเตอรี่บ่อย ๆ)

ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง เป็นปัญหาหลักที่เกิดขึ้นกับผู้ใช้เครื่องช่วยฟัง ดังนั้นจึงมีผู้ที่พยายามหาวิธีเพื่อแก้ปัญหาดังกล่าว ซึ่งวิธีการแก้ปัญหาการเกิดเสียงหอนในเครื่องช่วยฟังมีหลายวิธี เช่น การเพิ่มวงจรเข้าไปในเครื่องช่วยฟังทางเสียงเพื่อลดทอนอัตราขยาย หรือเปลี่ยนแปลงเฟสของสัญญาณเสียงเฉพาะช่วงความถี่ที่จะเกิดเสียงหอน และวิธีการเพิ่มวงจรกรองปรับตัวเข้าไปในเครื่องช่วยฟังเป็นต้น

วิธีการแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังที่นิยมใช้กันคือ การแก้ปัญหาด้วยการใช้วงจรกรองปรับตัว ซึ่งขั้นตอนวิธีที่นิยมใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวแบบทรานส์เวอร์ (Transversal Filter) คือ ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด (Least Mean Square Algorithm: LMS) [1,2] ซึ่งขั้นตอนวิธีดังกล่าวมีความซับซ้อนในการประมวลมากไม่เหมาะแก่การสร้างจริง (Implementation) ในรูปแบบของวงจรรวม (Very Large Scale Integrated Circuit: VLSI) วิทยานิพนธ์นี้จึงนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท sign-sign (sign-sign LMS) มาใช้เพื่อช่วยลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดโดยอาศัยการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง (Squared Error) มาช่วยปรับปรุงประสิทธิภาพเพื่อให้เหมาะกับการสร้างจริงบนฮาร์ดแวร์ โดยที่วิทยานิพนธ์นี้จะทำการออกแบบวงจรกรองปรับตัวโดยอาศัยขั้นตอนวิธีดังกล่าวด้วยภาษา VHDL (VHSIC Hardware Description Language, Very High Speed Integrated Circuit: VHSIC) [3] ให้สามารถนำไปใช้งานได้จริงบนFPGA (Field Programmable Gate Array) และสามารถนำไปสร้างในรูปแบบของวงจรรวม VLSI ได้จริง

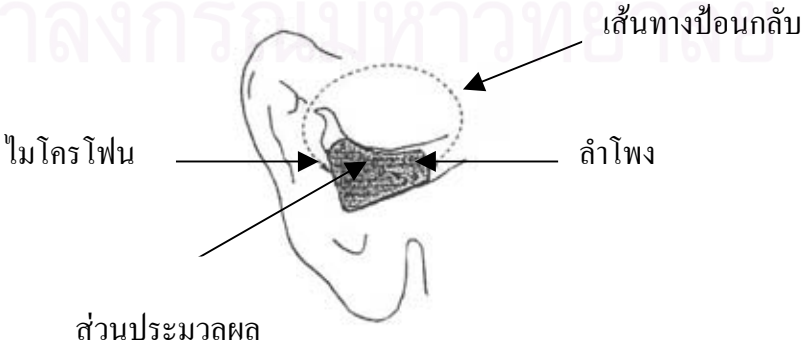
ในบทนี้จะกล่าวถึง เครื่องช่วยฟัง สาเหตุของปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง การแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง รวมถึงแนวทาง วัตถุประสงค์ ขอบเขตของวิทยานิพนธ์ ขั้นตอนการดำเนินงาน ภาพรวมของเนื้อหาในแต่ละบทและนิยามสัญลักษณ์

1.1 เครื่องช่วยฟัง (Acoustic Hearing Aids)

โดยทั่วไปเครื่องช่วยฟังจะประกอบไปด้วย ไมโครโฟน ทำหน้าที่แปลงสัญญาณเสียงหรือคลื่นเสียง (Acoustic Wave) เป็นสัญญาณไฟฟ้า เพื่อส่งต่อไปยังส่วนที่เรียกว่า โครงสร้างหลักที่ทำหน้าที่ขยายสัญญาณ จากนั้นสัญญาณที่ถูกขยายแล้วจะถูกส่งผ่านไปยังลำโพงและจะเปลี่ยนสัญญาณไฟฟ้ากลับไปเป็นสัญญาณเสียง โดยปกติผู้ใช้เครื่องช่วยฟังต้องปรับอัตราขยายเสียงให้เหมาะกับการสูญเสียการได้ยินด้วยตนเอง นอกจากปัญหาในการปรับอัตราขยายเสียงให้เหมาะสมแล้ว ผู้ใช้เครื่องช่วยฟังยังต้องพบกับปัญหาอื่น ๆ เช่น ปัญหาสัญญาณรบกวน (ซึ่งสัญญาณรบกวนจะถูกขยายไปพร้อมกับสัญญาณเสียงที่เราสนใจ) และปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง สำหรับในงานวิจัยนี้เลือกศึกษาเฉพาะปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

1.2 การป้อนกลับทางเสียงในเครื่องช่วยฟัง (Acoustic Feedback in Hearing Aids)

การป้อนกลับทางเสียงในเครื่องช่วยฟัง เกิดจากการที่สัญญาณเสียงย้อนกลับจากลำโพงของเครื่องช่วยฟังไปยังไมโครโฟนของเครื่องช่วยฟังอีกครั้ง โดยสัญญาณเสียงที่ย้อนกลับจากลำโพงไปยังไมโครโฟนจะเรียกว่า สัญญาณเสียงป้อนกลับ (Acoustic Feedback) และเส้นทางที่สัญญาณเสียงป้อนกลับเดินทางย้อนกลับจากลำโพงไปยังไมโครโฟนจะเรียกว่า เส้นทางป้อนกลับ (Feedback Path) ดังแสดงในรูปที่ 1.1



รูปที่ 1.1 การป้อนกลับทางเสียงในเครื่องช่วยฟัง

การป้อนกลับทางเสียงในเครื่องช่วยฟังเกิดจากหลายสาเหตุ เช่น เครื่องช่วยฟังโดยทั่วไปจะมีช่องระบายอากาศ และมีช่องว่างระหว่างเครื่องช่วยฟังกับช่องหู ซึ่งช่องทางทั้งสองนี้เป็นเส้นทางหลักที่ทำให้เสียงสามารถเดินทางย้อนกลับจากลำโพงไปยังไมโครโฟนได้ ถึงแม้ว่าเครื่องช่วยฟังจะไม่มีช่องระบายอากาศ และช่องว่างระหว่างเครื่องช่วยฟังกับช่องหูจะน้อยมาก เสียงก็ยังสามารถย้อนกลับจากลำโพงไปยังไมโครโฟนได้ด้วยเส้นทางอื่น ๆ อีก [4] เนื่องจากเส้นทางป้อนกลับซึ่งทำให้เกิดการลดทอนสัญญาณแต่ เครื่องช่วยฟังทำหน้าที่ขยายสัญญาณ ดังนั้นถ้าสัญญาณที่วนผ่านเส้นทางป้อนกลับถูกลดทอนน้อยกว่าอัตราขยายของเครื่องช่วยฟัง (โดยทั่วไปอัตราขยายของเครื่องช่วยฟังจะมีค่ามากกว่า 1) จะทำให้สัญญาณป้อนกลับถูกขยายทุก ๆ รอบที่มีการป้อนกลับจนในที่สุดทำให้สัญญาณดังกล่าวดังขึ้นที่เรียกว่า การเกิดเสียงหอน

1.3 การแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

วิธีการแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังมี 2 แนวทางคือ พยายามป้องกันไม่ให้สัญญาณป้อนกลับกลายเป็นเสียงหอนและพยายามกำจัดสัญญาณป้อนกลับ วิทยานิพนธ์นี้สนใจที่จะนำเอาวิธีการกำจัดสัญญาณเสียงป้อนกลับมาใช้ โดยการนำวงจรกรองปรับตัวมาใช้สร้างสัญญาณเลียนแบบสัญญาณป้อนกลับเพื่อหักล้างกับสัญญาณป้อนกลับที่เกิดขึ้นจริง (Feedback Cancellation) [4-9] ซึ่งวงจรกรองปรับตัวที่จะนำมาใช้พิจารณาในวิทยานิพนธ์นี้เป็นวงจรกรองปรับตัวอย่างไม่ต่อเนื่องซึ่งรายละเอียดจะกล่าวถึงในบทที่ 3

1.4 แนวทางของวิทยานิพนธ์

จากปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงของเครื่องช่วยฟัง และวิธีการแก้ปัญหาโดยการใช้วงจรกรองปรับตัวสร้างสัญญาณเลียนแบบสัญญาณป้อนกลับเพื่อหักล้างกับสัญญาณป้อนกลับที่เกิดขึ้นจริง วิทยานิพนธ์นี้จะทำการสร้างจริงวงจรกรองปรับตัวได้ที่สามารถสร้างสัญญาณเลียนแบบเพื่อหักล้างสัญญาณป้อนกลับที่เกิดขึ้นจริง โดยใช้ภาษา VHDL โดยที่วงจรกรองปรับตัวที่สร้างขึ้นนี้สามารถใช้งานได้จริงบน FPGA และสามารถนำไปสร้างจริงในรูปแบบของวงจรรวม VLSI ได้ วิทยานิพนธ์นี้จะนำเอาหลักการของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว เพื่อช่วยลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดที่ไม่เหมาะสมในการสร้างจริงในรูปแบบของวงจรรวม VLSI และจากผลของการลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ทำให้ประสิทธิภาพในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวด้อยลง เพื่อเพิ่มประสิทธิภาพให้กับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign จึงนำเอาหลักการปรับช่วงก้าวแบบค่าผิดพลาดกำลัง

สองที่มีความซับซ้อนในการคำนวณต่ำมาช่วยปรับปรุงประสิทธิภาพเพื่อให้เหมาะสมกับการสร้างจริงบนฮาร์ดแวร์

1.5 วัตถุประสงค์ของวิทยานิพนธ์

วัตถุประสงค์ของวิทยานิพนธ์นี้สามารถแบ่งเป็นข้อ ๆ ได้ดังนี้

1. ประเมินขั้นตอนวิธีของวงจรกรองปรับตัวแบบกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่อาศัยการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง สำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง
2. ออกแบบวงจรดิจิทัลต้นแบบของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ด้วยภาษา VHDL โดยใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธี ทั้งนี้วงจรที่สร้างขึ้นสามารถใช้งานได้จริงบน FPGA เพื่อศึกษาความซับซ้อนและประสิทธิภาพ

1.6 ขอบเขตของวิทยานิพนธ์

ขอบเขตของวิทยานิพนธ์นี้สามารถแบ่งเป็นข้อ ๆ ได้ดังนี้

1. ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง ปรับปรุงประสิทธิภาพขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign สำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง
2. ทำการออกแบบวงจรดิจิทัลต้นแบบของขั้นตอนวิธีกำลังสองเฉลี่ยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง ซึ่งวงจรดังกล่าวจะถูกบรรยายด้วยภาษา VHDL
3. สร้างจริงขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign โดยใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองตามที่ได้ออกแบบไว้ในข้อที่ 2 ให้สามารถใช้งานได้จริงบน FPGA เพื่อทำการศึกษาความซับซ้อนและประสิทธิภาพ

1.7 ประโยชน์ที่จะได้รับ

ประโยชน์ที่จะได้รับจากการทำวิทยานิพนธ์สามารถแบ่งเป็นข้อ ๆ ได้ดังนี้

1. ได้เรียนรู้ถึงขั้นตอนวิธีในการออกแบบวงจรดิจิทัลโดยใช้ภาษา VHDL

2. ได้เรียนรู้เทคนิคในการออกแบบวงจรกรองปรับตัวได้ แบบกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่มีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง

3. สามารถนำความรู้ที่ได้ในการออกแบบวงจรดิจิทัลด้วยภาษา VHDL และการใช้งานบอร์ด FPGA ไปใช้งานได้จริงในอนาคต

4. เสริมสร้างประสบการณ์ในการวิจัย โดยเฉพาะอย่างยิ่งในด้านการประมวลผลสัญญาณด้วยวงจรกรองแบบปรับตัว และการแก้ปัญหาการป้อนกลับทางเสียงในเครื่องช่วยฟัง

1.8 ขั้นตอนและวิธีการดำเนินการ

ขั้นตอนและวิธีการดำเนินการสามารถแบ่งเป็นข้อ ๆ ได้ดังนี้

1. ศึกษาการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง และนำเอาวิธีการปรับช่วงก้าวดังกล่าวมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign สำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง

2. ศึกษาการเขียนโปรแกรมภาษา VHDL

3. ศึกษาแบบจำลองการทำงานของวงจรกรองปรับตัวแบบกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง โดยการจำลองผลในรูปของ Floating Point Arithmetic และในรูปของ Fixed Point Arithmetic ทำการประเมินผลและประสิทธิภาพที่ได้ด้วย Matlab

4. ทำการออกแบบวงจรต้นแบบของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่อาศัยการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองด้วยภาษา VHDL โดยที่วงจรดังกล่าวจะใช้การคำนวณแบบ Fixed Point Arithmetic พร้อมทั้งวิเคราะห์ผลการทำงานด้วยการจำลองทางคอมพิวเตอร์

5. นำวงจรต้นแบบที่เขียนด้วยภาษา VHDL ในข้อ 4 มาทำการ Synthesize และ Implement ด้วยโปรแกรม Xilinx ISE เพื่อตรวจสอบความถูกต้องและให้สามารถทำงานได้จริงบน FPGA รวมถึงสามารถสร้างเป็นวงจรรวม VLSI ได้จริง

6. สรุปผลการศึกษาและเขียนวิทยานิพนธ์

1.9 ภาพรวมของวิทยานิพนธ์

เนื้อหาของวิทยานิพนธ์ฉบับนี้ แบ่งออกเป็น 6 บท บทที่ 2 จะกล่าวถึงวงจรกรองปรับตัว ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด จากนั้นจะกล่าวถึงขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และหลักการในการเลือกวิธีการปรับช่วงก้าวที่จะนำมาใช้ให้เหมาะสมกับการสร้างจริงในรูปแบบของวงจรรวม VLSI บทที่ 3 จะกล่าวถึงแบบจำลองของปัญหาการป้อนกลับทางเสียงในเครื่องช่วยฟัง การจำลองผลของ แบบจำลองของปัญหาการป้อนกลับทางเสียงในเครื่องช่วยฟังในรูปของ Floating Point Arithmetic การวิเคราะห์หาจำนวนบิตที่เหมาะสมของระบบ เมื่อจะนำเอาหลักการดังกล่าวไปใช้งานจริงในวงจรดิจิทัลในรูปของ Fixed Point Arithmetic และทำการจำลองผลที่ได้ของระบบเมื่อใช้การคำนวณแบบ Fixed Point Arithmetic จากนั้นทำการเปรียบเทียบและประเมินผลที่ได้ บทที่ 4 จะเป็นส่วนของการออกแบบวงจรที่จะใช้ในการสร้างจริงวงจรกรองปรับตัวของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง สำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งจะกล่าวถึงสถาปัตยกรรม (Architecture) ของส่วนประกอบต่าง ๆ ที่ใช้ในวงจร จากนั้นจะทำการสังเคราะห์ (Synthesize) และการสร้างจริง (Implementation) ของวงจรที่ได้ออกแบบไว้ด้วยภาษา VHDL ด้วยโปรแกรม Xilinx ISE เพื่อที่จะทำให้สามารถนำวงจรที่ออกแบบไว้มาใช้งานจริงบน FPGA และสามารถสร้างจริงในรูปแบบของวงจรรวม VLSI ได้ บทที่ 5 จะกล่าวถึงการทดสอบวงจรกรองปรับตัวทำการสร้างจริงบน FPGA และบทที่ 6 เนื้อหาในบทนี้จะเริ่มจากการสรุปผลการวิจัย จากนั้นจะกล่าวถึงข้อดี-ข้อเสีย และสุดท้ายเป็นข้อเสนอแนะสำหรับการวิจัยในอนาคต

1.10 นิยามสัญลักษณ์

สัญลักษณ์ตัวพิมพ์เล็ก หมายถึง สัญญาณในแต่ละเวลา หรือแทนสมาชิกแต่ละตัวของเมตริกซ์ หรือเวกเตอร์ สัญลักษณ์ตัวพิมพ์ที่มีลูกศรข้างบน หมายถึง เวกเตอร์ สัญลักษณ์ตัวพิมพ์ใหญ่ หมายถึง เมตริกซ์ นิยามสัญลักษณ์ที่กล่าวมาข้างต้นนี้ จะถูกใช้ไปตลอดทุกบทของวิทยานิพนธ์

บทที่ 2

ทฤษฎีวงจรกรองปรับตัวแบบขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด

เนื้อหาในบทนี้กล่าวถึงวงจรกรองปรับตัวที่จะนำมาใช้ในการออกแบบวงจรด้วยภาษา VHDL สำหรับแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ทฤษฎีของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด แบบจำลองของปัญหาการป้อนกลับทางเสียงในเครื่องช่วยฟัง และการนำเอาหลักการของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาประยุกต์ใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว รวมถึงหลักการในการเลือกวิธีการปรับช่วงก้าวแบบพลวัตมาใช้ให้เหมาะสมกับการสร้างจริงในรูปแบบของวงจรรวม VLSI

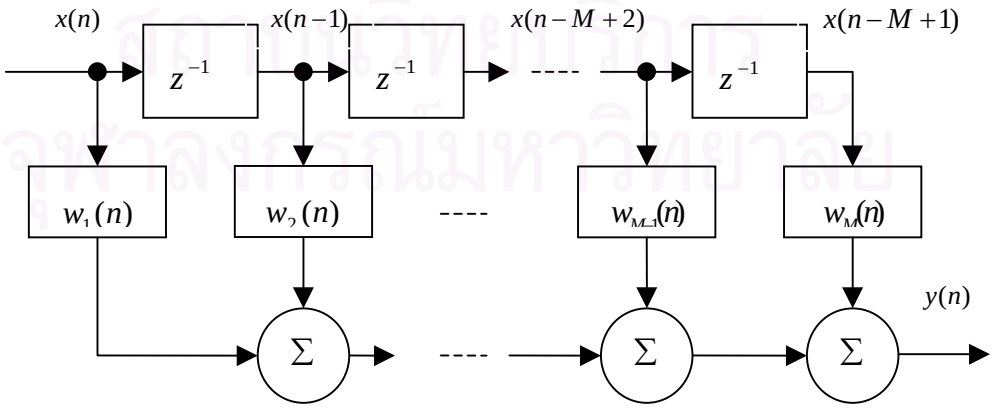
2.1 วงจรกรองปรับตัวแบบความจำจำกัด (Finite Impulse Response (FIR) Adaptive Filter)

ขั้นตอนวิธีวงจรกรองปรับตัวถูกนำมาใช้กันอย่างแพร่หลาย เห็นได้จากการนำมาประยุกต์ใช้ในงานกรรมวิธีสัญญาณดิจิทัลด้านโทรคมนาคมและคอมพิวเตอร์จำนวนมาก เช่น การปรับแต่งช่องสัญญาณสื่อสาร (Channel Equalization) การกำจัดสัญญาณเสียงก้องในช่องสัญญาณสื่อสาร (Echo Cancellation) และการเข้ารหัสสัญญาณเสียงพูด (Speech Encoding) เป็นต้น การสร้างจริงขั้นตอนวิธีวงจรกรองปรับตัวในการประยุกต์ใช้งานด้านกรรมวิธีสัญญาณดิจิทัลสามารถทำได้ 2 รูปแบบ ในรูปแบบแรกขั้นตอนวิธีวงจรกรองดังกล่าวจะถูกสร้างจริงในรูปแบบของซอฟต์แวร์ (Software) บนเครื่องไมโครโปรเซสเซอร์ทั่วไป (General Microprocessor) หรือบนเครื่องไมโครโปรเซสเซอร์สำหรับงานกรรมวิธีสัญญาณดิจิทัล (Digital Signal Microprocessor) การสร้างจริงในรูปแบบแรกนี้เหมาะสำหรับงานที่มีอัตราการประมวลผลสัญญาณไม่สูง เช่น การประยุกต์ในด้านเกี่ยวกับเสียงซึ่งมีอัตราการสุ่มตัวอย่าง (Sampling Rate) ต่ำกว่าหนึ่งส่วนร้อยของอัตราสัญญาณเวลา (Clock Rate) ที่ใช้ในการประมวลผลและไม่มีข้อจำกัดในเรื่องของพลังงาน สำหรับในรูปแบบที่สอง ขั้นตอนวิธีวงจรกรองจะถูกสร้างจริงในรูปของวงจรรวม VLSI การสร้างจริงในรูปแบบที่สองนี้จะซับซ้อนกว่ารูปแบบแรกอย่างมาก โดยส่วนมากจะถูกใช้ในการประยุกต์ที่ต้องการอัตราการประมวลผลสัญญาณสูง เช่น การประยุกต์ในด้านการส่งสัญญาณมัลติมีเดียผ่านสาย UTP (Unshielded Twisted Pair) [10]

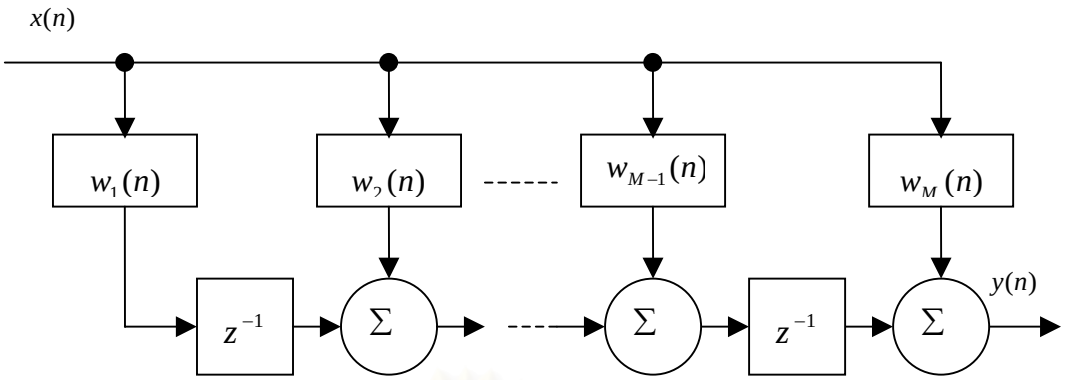
วงจรกรองปรับตัวที่ใช้ในวิทยานิพนธ์นี้คือ วงจรกรองปรับตัวแบบทรานส์เวอร์ (Adaptive Transversal Filter) ประกอบด้วยส่วนประกอบหลัก 2 ส่วนคือ วงจรกรองและส่วนปรับค่าสัมประสิทธิ์ของวงจรกรอง

วงจรกรองปรับตัว (Adaptive Filter) คือ วงจรกรองที่สามารถปรับค่าสัมประสิทธิ์ (Tap Weight) ของวงจรกรองได้ กระบวนการทำงานจะเหมือนกับวงจรกรองทั่ว ๆ ไป โดยทำการคูณค่าสัญญาณขาเข้า (Tap Input Vector) ที่เวลา n กับค่าสัมประสิทธิ์ของวงจรกรอง (Tap Weight Vector) ที่เวลา n เพื่อที่จะหาผลลัพธ์ที่ได้จากการประมาณ (Estimate) ที่เวลา n จากนั้นนำค่าผลลัพธ์ที่ได้ไปทำการลบออกจากค่าผลลัพธ์ที่ต้องการ (Desired Response) ที่เวลา n จะได้ค่าความผิดพลาด (Error) ที่เวลา n ซึ่งจะนำค่าความผิดพลาดดังกล่าวไปใช้ในขั้นตอนวิธีของการปรับค่าสัมประสิทธิ์ของวงจรกรองต่อไป สำหรับในส่วนของการปรับค่าสัมประสิทธิ์ของวงจรกรองนั้น จะนำเอาค่าความผิดพลาดที่เวลา n ที่คำนวณได้มาใช้ในการคำนวณ โดยที่วิทยานิพนธ์นี้จะนำเสนอหลักการของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว ซึ่งจะกล่าวถึงในหัวข้อต่อไป

วงจรกรองที่นำมาใช้เป็นส่วนประกอบของวงจรกรองปรับตัวคือ วงจรกรองแบบ FIR (Finite Impulse Response) หรือวงจรกรองที่มีผลตอบสนองต่ออิมพัลส์จำกัด โดยทั่วไปวิธีการที่นำมาใช้ในการสร้างจริง (General Implementation Techniques) [11] ของวงจรกรองแบบ FIR มี 2 รูปแบบคือ แบบ Direct Form (DF) และแบบ Transposed Direct Form (TDF) ซึ่งทั้งสองรูปแบบมีข้อดีและข้อเสียที่แตกต่างกัน แบบ Direct Form จะใช้เวลาในการคำนวณสูงทำให้มีความเร็วในการคำนวณต่ำ เนื่องจากต้องเสียเวลาในการหาผลรวมของผลบวกในแต่ละ Tap ของวงจรกรอง แต่จะใช้พื้นที่ในการสร้างจริงที่น้อยทำให้มีการสูญเสียพลังงานน้อย ส่วนแบบ Transposed Direct Form จะใช้เวลาในการคำนวณต่ำทำให้ความเร็วในการคำนวณสูงกว่า แต่ต้องการพื้นที่ในการสร้างจริงที่มากกว่า เนื่องจากขนาดของ Register ที่นำมาใช้ในแต่ละ Tap จะต้องสามารถรองรับขนาดของผลบวกของวงจรกรองได้ ซึ่งจะทำให้ต้องสูญเสียพลังงานมากกว่าด้วย

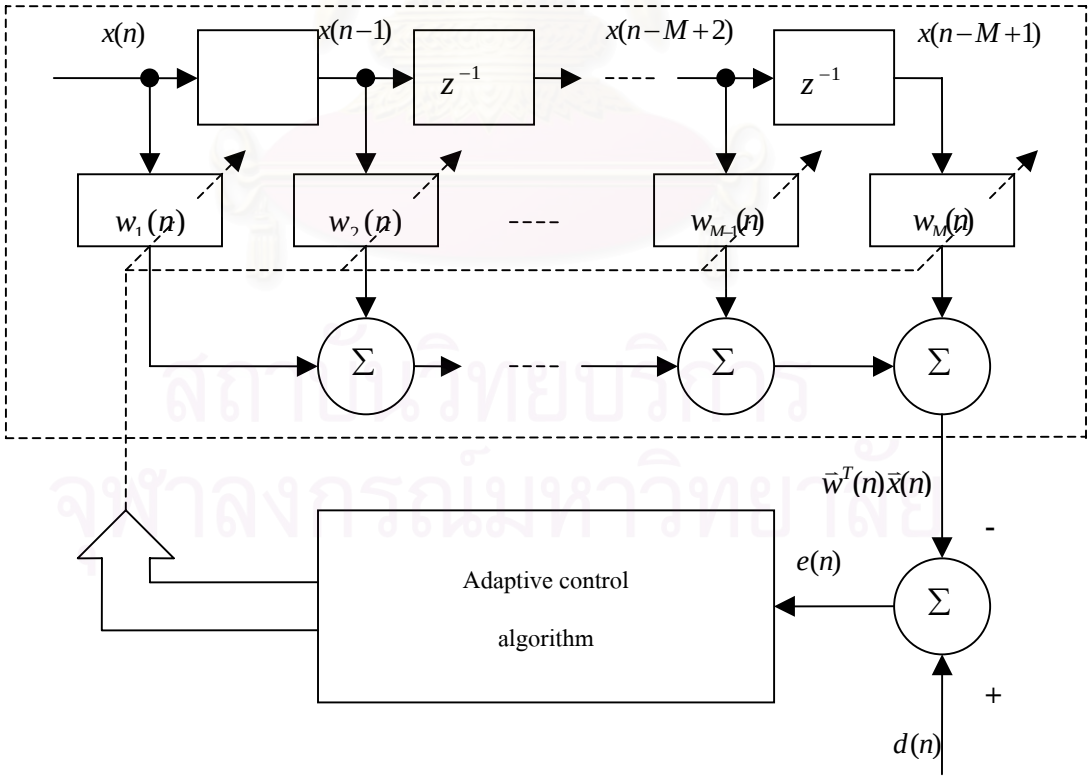


รูปที่ 2.1 วงจรกรองแบบ FIR (แบบ Direct Form)



รูปที่ 2.2 วงจรกรองแบบ FIR (แบบ Transposed Direct Form)

รูปที่ 2.1 และ 2.2 แสดงรูปโครงสร้างของวงจรกรองแบบ FIR ในรูปแบบของ Direct Form และ Transposed Direct Form ตามลำดับ ซึ่งการสร้างจริงของวงจรกรองที่จะนำเสนอในวิทยานิพนธ์นี้มีข้อจำกัดเกี่ยวกับทรัพยากรในด้านพื้นที่บนชิป FPGA รวมทั้งความเร็วที่ใช้ในการประมวลผลเกี่ยวกับสัญญาณเสียงที่จะทำการวิเคราะห์เกี่ยวกับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังไม่สูงมาก ดังนั้นการสร้างจริงวงจรกรองในวิทยานิพนธ์นี้จะใช้วงจรกรองแบบ FIR ในรูปแบบ Direct Form มาใช้ในการสร้างจริง รูปที่ 2.3 แสดงโครงสร้างของวงจรกรองปรับตัวที่จะนำมาใช้ในการสร้างจริง



รูปที่ 2.3 โครงสร้างของวงจรกรองปรับตัวแบบทรานส์เวอร์

กำหนดให้ $\vec{x}(n)$ คือ สัญญาณขาเข้า (Tap Input Vector) ที่เวลา n

$\vec{w}(n)$ คือ ค่าสัมประสิทธิ์ของวงจรกรอง (Tap Weight Vector) ที่เวลา n

$d(n)$ คือ ผลลัพธ์ที่ต้องการ (Desired Response) ที่เวลา n

$e(n)$ คือ ค่าความผิดพลาด (Error) ที่เวลา n

โดยที่

$$\vec{x}(n) = \left(x(n), \ldots, x(n-M+1)\right)^T \tag{2-1}$$

$$\vec{w}(n) = \left(w_1(n), \ldots, w_M(n)\right)^T \tag{2-2}$$

$$y(n) = \vec{w}^T(n)\vec{x}(n) \tag{2-3}$$

$$e(n) = d(n) - y(n) \tag{2-4}$$

โดยที่ M คือ อันดับของวงจรกรองปรับตัวและ $(.)^T$ คือ ตัวสลับเปลี่ยน (Transpose) ของเวกเตอร์หรือของเมตริกซ์ จากสมการที่ (2-1) แสดงให้เห็นถึงสัญญาณขาเข้าที่เวลา n สมการที่ (2-2) แสดงให้เห็นถึงสัมประสิทธิ์ของวงจรกรองปรับตัวที่เวลา n สมการที่ (2-3) แสดงผลลัพธ์ที่ได้จากการคำนวณของวงจรกรอง ที่เวลา n และสมการที่ (2-4) แสดงให้เห็นถึงการคำนวณของค่าความผิดพลาดที่เวลา n ที่จะนำไปใช้ในกระบวนการปรับค่าสัมประสิทธิ์ (Adaptive Control Algorithm) ของวงจรกรองปรับตัวต่อไป

2.2 ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด (Least Mean Square Algorithm: LMS)

ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดเป็นขั้นตอนวิธีที่นิยมใช้กันอย่างแพร่หลายในการปรับค่าสัมประสิทธิ์ของวงจรกรองแบบทรานส์เวอร์ (Transversal Filter) ซึ่งต่อไปจะกล่าวถึงหลักการในการคำนวณของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด โดยจะเริ่มพิจารณาที่ค่าความผิดพลาดกำลังสองเฉลี่ย หลักการของ Steepest Descent และขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดตามลำดับ

2.2.1 ค่าความผิดพลาดกำลังสองเฉลี่ย (Mean-Squared Error: MSE)

พิจารณารูปที่ 2.3 สามารถหาค่าความผิดพลาดได้จากสมการที่ (2-4) และเมื่อนำสมการที่ (2-3) แทนค่าในสมการที่ (2-4) จะได้

$$e(n) = d(n) - \vec{w}^T(n)\vec{x}(n) \tag{2-5}$$

ถ้าสัญญาณขาเข้า $\vec{x}(n)$ และผลลัพธ์ที่ต้องการ $d(n)$ อยู่ในสภาวะ Stationary ร่วมกัน สามารถหาค่าความผิดพลาดกำลังสองเฉลี่ย $J(n)$ ที่เวลา n โดยการหาค่ากำลังสองเฉลี่ยของค่าความผิดพลาดในสมการที่ (2-5) จะได้

$$J(n) = \sigma_d^2 - 2\vec{w}^T(n)\vec{p} + \vec{w}^T(n)R\vec{w}(n) \tag{2-6}$$

โดยที่ σ_d^2 คือ ค่าความแปรปรวน (Variance) ของผลลัพธ์ที่ต้องการ

$\vec{p}$ คือ เวกเตอร์สหสัมพันธ์ข้าม (Cross-Correlation Vector) ระหว่างสัญญาณขาเข้าและผลลัพธ์ที่ต้องการ

R คือ เมตริกซ์สหสัมพันธ์ (Correlation Matrix) ของสัญญาณขาเข้า

โดยปกติแล้วเมื่อทำการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวไปเรื่อย ๆ ค่าความผิดพลาดกำลังสองเฉลี่ย $J(n)$ จะลดต่ำลงจนกระทั่งค่าสัมประสิทธิ์ของวงจรกรองปรับตัวเข้าสู่ค่าออปติมัม (Optimum) ทำให้ได้ค่าความผิดพลาดกำลังสองเฉลี่ยน้อยสุด (Minimum Mean-Squared Error: MMSE) ซึ่งค่าสัมประสิทธิ์ของวงจรกรองที่สภาวะออปติมัมหาได้จาก

$$\vec{w}_{opt} = R^{-1}\vec{p} \tag{2-7}$$

เมื่อนำค่าสัมประสิทธิ์ของวงจรกรองที่สภาวะออปติมัมแทนค่าในสมการที่ (2-6) จะได้ค่าความผิดพลาดกำลังสองเฉลี่ยน้อยสุดดังสมการ

$$J_{\min} = \sigma_d^2 - \vec{p}^T\vec{w}_{opt} \tag{2-8}$$

จากสมการที่ (2-7) จะเห็นว่าค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่สภาวะออปติมัมมีส่วนประกอบของเมตริกซ์ผกผันอยู่ด้วย ซึ่งถ้าเมตริกซ์ดังกล่าวมีขนาดใหญ่จะทำให้ยากลำบากในการคำนวณ เพื่อหลีกเลี่ยงปัญหาดังกล่าว จึงได้นำหลักการของ Steepest Descent เข้ามาช่วยในการคำนวณ ซึ่งในหัวข้อต่อไปจะกล่าวถึงหลักการของ Steepest Descent

2.2.2 หลักการของ Steepest Descent (Method of Steepest Descent)

หลักการของ Steepest Descent นำมาประยุกต์ใช้เพื่อความสะดวกในการหาค่าสัมประสิทธิ์ของวงจรกรอง มีวัตถุประสงค์เพื่อหลีกเลี่ยงปัญหาในการคำนวณค่าเมตริกซ์ผกผันใน

การหาค่าสัมประสิทธิ์ของวงจรกรอง ซึ่งจะสร้างความยากลำบากในการคำนวณหากเมตริกซ์ดังกล่าวมีขนาดใหญ่ หลักการของ Steepest Descent สามารถแบ่งได้เป็น 4 ขั้นตอนดังนี้

1. กำหนดค่าเริ่มต้นของค่าสัมประสิทธิ์ที่ใช้ในวงจรกรองปรับตัว โดยปกติจะให้มีค่าเป็นศูนย์ (Null Vector)

2. ใช้ค่าสัมประสิทธิ์เริ่มต้น (หรือปัจจุบัน) คำนวณหาเกรเดียนต์เวกเตอร์ (Gradient Vector)

$$\nabla J(n) = \frac{\partial J(n)}{\partial \vec{w}(n)} = -2\vec{p} + 2R\vec{w}(n) \tag{2-9}$$

3. คำนวณหาค่าสัมประสิทธิ์ของวงจรกรองที่เวลา $n+1$

$$\vec{w}(n+1) = \vec{w}(n) + \frac{1}{2}\mu\left[-\nabla J(n)\right] \tag{2-10}$$

โดยที่ μ คือ ค่าช่วงก้าว (Step Size)

4. กลับไปที่ขั้นตอนที่ 2 แล้วทำซ้ำ

2.2.3 ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด (Least Mean Square Algorithm)

จากหลักการของ Steepest Descent เพื่อให้มีความสะดวกมากยิ่งขึ้นในการใช้งานจริงจึงได้พัฒนาหลักการของ Steepest Descent มาเป็นขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ซึ่งนำมาประยุกต์ให้ง่ายขึ้น คือ ไม่ต้องหาเมตริกซ์ผกผันของ R และไม่ต้องหาค่าเกรเดียนต์จริง (True Gradient) อีกด้วย โดยใช้การประมาณค่าเมตริกซ์สหสัมพันธ์ของสัญญาณขาเข้า R และเวกเตอร์สหสัมพันธ์ข้ามระหว่างสัญญาณขาเข้าและผลลัพธ์ที่ต้องการ $\vec{p}$ ด้วยค่าประมาณที่เวลานั้น ๆ

โดยที่
$$R(n) = \vec{x}(n)\vec{x}^T(n) \tag{2-11}$$

$$\vec{p}(n) = \vec{x}(n)d(n) \tag{2-12}$$

เมื่อนำสมการที่ (2-11) และ (2-12) แทนค่าในสมการที่ (2-9) จะได้

$$\nabla J(n) = -2\vec{x}(n)d(n) + 2\vec{x}(n)\vec{x}^T(n)\vec{w}(n) \tag{2-13}$$

นำสมการที่ (2-13) แทนค่าในสมการที่ (2-10) จะได้

$$\vec{w}(n+1) = \vec{w}(n) + \mu \vec{x}(n)[d(n) - \vec{x}^T(n)\vec{w}(n)] \quad (2\text{-}14)$$

ซึ่งสามารถเขียนได้ในรูป

$$e(n) = d(n) - \vec{w}^T(n)\vec{x}(n) \quad (2\text{-}15)$$

$$\vec{w}(n+1) = \vec{w}(n) + \mu \vec{x}(n)e(n) \quad (2\text{-}16)$$

ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดจะอยู่ในรูปแบบของสมการที่ (2-15) และสมการที่ (2-16) ค่าช่วงก้าว μ ในสมการที่ (2-16) เป็นค่าพารามิเตอร์ที่สำคัญอีกตัวหนึ่งใช้สำหรับกำหนดอัตราเร็วในการปรับตัวของขั้นตอนวิธี โดยที่ค่าช่วงก้าวของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดจะถูกกำหนดให้เป็นค่าคงที่ ในกรณีที่ค่าช่วงก้าวมีขนาดใหญ่ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดจะมีอัตราการปรับตัวสูงแต่ค่าความถูกต้องในการปรับตัวที่สภาวะคงตัวจะต่ำ ในทางตรงกันข้ามถ้าค่าช่วงก้าวมีขนาดเล็ก ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดจะมีอัตราการปรับตัวต่ำแต่ค่าความถูกต้องในการปรับตัวที่สภาวะคงตัวจะสูง จากสมการข้างต้นชี้ให้เห็นว่าในแต่ละรอบของการสุ่มตัวอย่าง การคำนวณของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดจะประกอบด้วยการคูณ $M+1$ ครั้ง และการบวก $M+1$ ครั้ง

2.3 ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign (Sign-Sign LMS)

เนื่องจากในวิทยานิพนธ์นี้จะทำการสร้างจริงวงจรกรองปรับตัว ซึ่งจะนำมาประยุกต์ใช้เพื่อแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยจะทำการสร้างสัญญาณเลียนแบบสัญญาณป้อนกลับที่เกิดขึ้นจริงดังที่กล่าวมาแล้วข้างต้น ดังนั้นการนำขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดซึ่งมีรูปแบบดังสมการที่ (2-15) และสมการที่ (2-16) มาใช้ในการสร้างจริงในรูปแบบฮาร์ดแวร์จึงไม่เหมาะสม เนื่องจากความซับซ้อนของขั้นตอนวิธี ทำให้ต้องใช้พื้นที่สูงในการสร้างจริง นอกจากนี้ทำให้ต้องสูญเสียพลังงานในการประมวลผลจากความซับซ้อนของขั้นตอนวิธีสูงอีกด้วย โครงการวิทยานิพนธ์นี้จึงนำขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาใช้เพื่อแก้ปัญหาดังกล่าว ซึ่งขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มีรูปแบบสมการดังต่อไปนี้

$$e(n) = d(n) - \vec{w}^T(n)\vec{x}(n) \quad (2\text{-}17)$$

$$\vec{w}(n+1) = \vec{w}(n) + \mu\,[\mathrm{sgn}(\vec{x}(n))\ \mathrm{sgn}(e(n))] \quad (2\text{-}18)$$

โดยที่

$$\mathrm{sgn}(x) = \begin{cases} 1, x \geq 0 \\ -1, x < 0 \end{cases} \quad (2\text{-}19)$$

สมการข้างต้นชี้ให้เห็นว่าการประมวลผลของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ประกอบด้วยการบวกเพียง $M+1$ ครั้งเท่านั้น ความซับซ้อนของการคำนวณสำหรับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign จึงน้อยกว่าขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดมาก อย่างไรก็ตามการลดความซับซ้อนในการคำนวณของขั้นตอนวิธี ทำให้สมรรถนะการปรับตัวของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ลดลง เพื่อแก้ปัญหาดังกล่าววิทยานิพนธ์นี้จึงนำหลักการของการปรับช่วงก้าวแบบพลวัต (Dynamic Adaptation of Step Size) เข้ามาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ให้มีประสิทธิภาพดีขึ้นและสามารถนำไปใช้งานจริงได้ ซึ่งในหัวข้อต่อไปจะกล่าวถึงหลักการปรับช่วงก้าวแบบพลวัตวิธีต่าง ๆ รวมถึงวิธีการเลือกการปรับช่วงก้าวแบบพลวัตมาใช้งานจริง

2.4 วิธีการปรับช่วงก้าวแบบพลวัต (Dynamic Adaptation of Step Size)

ค่าช่วงก้าว (Step Size) ใช้สำหรับกำหนดอัตราเร็วและความถูกต้องในการปรับตัวของขั้นตอนวิธี ซึ่งในขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดจะถูกกำหนดให้เป็นค่าคงที่ ในกรณีที่ช่วงก้าวมีขนาดใหญ่และเล็กจะมีประสิทธิภาพที่แตกต่างกันดังที่กล่าวมาแล้วข้างต้น ดังนั้นจึงมีผู้นำเสนอวิธีการปรับช่วงก้าวแบบพลวัตขึ้นมา เพื่อให้สามารถใช้งานได้อย่างมีประสิทธิภาพ ซึ่งขนาดของช่วงก้าวจะถูกกำหนดให้มีขนาดใหญ่ในขณะที่อยู่ห่างจากสภาวะคงตัว เพื่อให้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดลู่เข้าด้วยความรวดเร็ว เมื่อเข้าสู่สภาวะคงตัวแล้ว ขนาดของช่วงก้าวจะถูกลดลงเพื่อเพิ่มความถูกต้องในการประมาณค่าสัมประสิทธิ์ของวงจรกรอง ในที่นี้จะกล่าวถึงวิธีการปรับช่วงก้าวแบบพลวัต 4 วิธีด้วยกันคือ วิธีการปรับช่วงก้าวแบบ Squared Error วิธีการปรับช่วงก้าวแบบ Normalized Step-Size วิธีการปรับช่วงก้าวแบบ Cross Correlation และ วิธีการปรับช่วงก้าวแบบ Delta-Bar-Delta

2.4.1 วิธีการปรับช่วงก้าวแบบ Squared Error (Kwong and Johnston [12])

Kwong และ Johnston เสนอความคิดว่าควรปรับขนาดช่วงก้าวตามค่าความผิดพลาดกำลังสองเฉลี่ย เมื่อค่าความผิดพลาดกำลังสองเฉลี่ยมีค่ามาก ค่าสัมประสิทธิ์ของวงจรกรองจะต่างจากค่าสัมประสิทธิ์ที่สภาวะออปติมัมมาก จึงควรใช้ค่าช่วงก้าวขนาดใหญ่เพื่อให้ค่าสัมประสิทธิ์ของวงจรกรองลู่เข้าสู่ค่าสัมประสิทธิ์ที่สภาวะออปติมัมเร็วขึ้น ในทางตรงกันข้ามเมื่อค่าความผิดพลาด

กำลังสองเฉลี่ยมีค่าน้อยแสดงว่าค่าสัมประสิทธิ์ของวงจรกรองใกล้เคียงกับค่าสัมประสิทธิ์ที่สภาวะออปติมัมก็ควรให้ค่าช่วงก้าวมีขนาดเล็ก เพื่อให้การปรับค่าสัมประสิทธิ์ของวงจรกรองมีความถูกต้องมากขึ้น ซึ่งหลักการดังกล่าวมีรูปแบบสมการดังนี้

$$\mu'(n+1) = \alpha\,\mu(n) + \gamma e^2(n) \tag{2-20}$$

$$\mu(n+1) = \begin{cases} \mu_{\max}, & \mu'(n+1) > \mu_{\max} \\ \mu_{\min}, & \mu'(n+1) < \mu_{\min} \\ \mu'(n+1), & Others \end{cases} \tag{2-21}$$

เมื่อ $e(n)$ คือ ค่าความผิดพลาดของวงจรกรองที่เวลา n, $0<\alpha<1$, $\gamma>0$ และ $0<\mu_{\min}<\mu_{\max}$ โดยที่ $\mu_{\min},\mu_{\max}$ คือ ค่าคงตัวแทนขนาดช่วงก้าวที่เล็กที่สุดและขนาดช่วงก้าวที่ใหญ่ที่สุดตามลำดับ ซึ่งค่าดังกล่าวมีไว้เพื่อให้ขนาดของช่วงก้าวอยู่ในช่วงที่สามารถส่งผลให้มีการปรับค่าสัมประสิทธิ์ของวงจรกรองได้ และมีไว้เพื่อป้องกันไม่ให้ขนาดของช่วงก้าวมีขนาดใหญ่จนทำให้การปรับค่าสัมประสิทธิ์ของวงจรกรองขาดเสถียรภาพตามลำดับ

2.4.2 วิธีการปรับช่วงก้าวแบบ Normalized Step-Size (Honig and Messerschmitt [13])

Honig และ Messerschmitt เสนอวิธีการปรับขนาดช่วงก้าว โดยการหารขนาดช่วงก้าวด้วยกำลังของสัญญาณขาเข้า ซึ่งหลักการดังกล่าวมีรูปแบบสมการดังนี้

$$\mu(n) = \frac{a}{\sigma^2(n)+b} \tag{2-22}$$

เมื่อ $\mu(n)$ คือ ขนาดช่วงก้าวที่เวลา n โดยที่ a และ b คือ ค่าคงตัวที่เป็นค่าบวก (ค่าคงตัว b มีไว้เพื่อป้องกันไม่ให้ขนาดของช่วงก้าวมีขนาดใหญ่เกินไป เมื่อกำลังของสัญญาณขาเข้ามีค่าน้อยมาก ๆ) และ $\sigma^2(n)$ คือ ค่าประมาณของกำลังของสัญญาณขาเข้าที่เวลา n ซึ่งนิยมใช้ค่าประมาณกำหนดดังสมการ

$$\sigma^2(n) = \alpha\sigma^2(n-1) + (1-\alpha)x^2(n) \tag{2-23}$$

โดยที่ α เป็นค่าคงตัวที่เป็นค่าบวกและน้อยกว่าหนึ่งและ $x(n)$ คือ สัญญาณขาเข้าของวงจรกรองที่เวลา n

2.4.3 วิธีการปรับช่วงก้าวแบบ Cross Correlation (Karni and Zeng [14])

Karni และ Zeng ใช้ค่าสหสัมพันธ์ไขว้ (Cross Correlation) ระหว่างค่าความผิดพลาดกับสัญญาณขาเข้าของวงจรกรองเป็นตัวกำหนดขนาดช่วงก้าว ทั้งนี้เพราะมีทฤษฎีอยู่ว่าที่สภาวะคงตัวค่าสหสัมพันธ์ไขว้ระหว่างค่าความผิดพลาดกับสัญญาณขาเข้าของวงจรกรองจะมีค่าเป็นศูนย์ ดังนั้นถ้าค่าสหสัมพันธ์ไขว้ระหว่างค่าความผิดพลาดกับสัญญาณขาเข้ามีค่ามาก แสดงว่าค่าสัมประสิทธิ์ของวงจรกรองยังคงอยู่ห่างจากสภาวะคงตัวจึงควรใช้ค่าของขนาดช่วงก้าวที่มีขนาดใหญ่ และถ้าค่าสหสัมพันธ์ไขว้ระหว่างค่าความผิดพลาดและสัญญาณขาเข้ามีค่าน้อย ควรจะใช้ค่าของขนาดช่วงก้าวที่มีขนาดเล็ก ซึ่งหลักการดังกล่าวมีรูปแบบของสมการดังนี้

$$C = \left\| e(n)\vec{x}(n) \right\|^2 \tag{2-24}$$

$$\mu(n+1) = \mu_{\max}(1-\exp(-\alpha C)) \tag{2-25}$$

เมื่อ $e(n)$ คือ ค่าความผิดพลาดที่เวลา n และ $\vec{x}(n)$ คือ สัญญาณขาเข้าของวงจรกรองที่เวลา n โดยที่ α คือ ค่าคงตัวที่มากกว่าศูนย์

2.4.4 วิธีการปรับช่วงก้าวแบบ Delta-Bar-Delta (Jacobs [15])

Jacobs ได้เสนอความคิดในการปรับขนาดของช่วงก้าวไว้ว่า ขนาดของช่วงก้าวที่เหมาะสมกับค่าสัมประสิทธิ์ของวงจรกรองแต่ละตัวไม่จำเป็นต้องเท่ากัน ค่าสัมประสิทธิ์แต่ละตัวควรจะมีขนาดช่วงก้าวเป็นของตนเอง และขนาดช่วงก้าวแต่ละตัวควรจะปรับเปลี่ยนค่าได้ โดยสังเกตว่าถ้าค่าสัมประสิทธิ์เพิ่มขึ้นติดต่อกันหรือลดลงติดต่อกันก็ควรจะเพิ่มขนาดช่วงก้าว ในทางกลับกันถ้าค่าสัมประสิทธิ์ของวงจรกรองเพิ่มขึ้นและลดลงสลับกันก็ควรจะลดขนาดช่วงก้าว เนื่องจากค่าสัมประสิทธิ์แต่ละตัวมีขนาดช่วงก้าวเป็นของตนเอง ทำให้การปรับค่าสัมประสิทธิ์ไม่ใช้การปรับในทิศทางที่ตรงกันข้ามกับเกรเดียนต์ของค่าความผิดพลาดกำลังสองเฉลี่ย แต่เป็นการปรับค่าสัมประสิทธิ์ที่อาศัยการหาอนุพันธ์ย่อยของค่าความผิดพลาดกำลังสองเฉลี่ยเทียบกับค่าสัมประสิทธิ์แต่ละตัว ซึ่งหลักการดังกล่าวมีรูปแบบของสมการดังนี้

$$\mu_i(n+1) = \begin{cases} \mu_i(n)+k, & \bar{\delta}_i(n-1)\delta_i(n) > 0 \\ \mu_i(n)-\phi\mu_i(n), & \bar{\delta}_i(n-1)\delta_i(n) < 0 \\ \mu_i(n), & Others \end{cases} \tag{2-26}$$

$$\bar{\delta}_i(n) = \alpha\bar{\delta}_i(n-1) + (1-\alpha)\delta_i(n) \tag{2-27}$$

$$\delta_i(n) = e(n)x_i(n) \tag{2-28}$$

โดยที่ k, ϕ และ α คือ ค่าคงตัวที่เป็นค่าบวก $(\alpha < 1)$ และ $\delta_i(n)$ คือ ค่าประมาณของอนุพันธ์ย่อยของค่าความผิดพลาดกำลังสองเฉลี่ยเทียบกับค่าสัมประสิทธิ์ของวงจรกรองตัวที่ i ที่เวลา n ส่วน $\overline{\delta}_i(n)$ คือ ค่าเฉลี่ยของ $\delta_i(n)$

จากสมการที่ (2-26) หมายความว่าขนาดของช่วงก้าวตัวที่ i จะเพิ่มขึ้นครั้งละ k ถ้า $\overline{\delta}_i(n)$ มีเครื่องหมายเหมือนกับ $\delta_i(n)$ และขนาดช่วงก้าวตัวที่ i จะลดลง $\phi\mu_i(n)$ ทุก ๆ ครั้งที่ $\overline{\delta}_i(n)$ มีเครื่องหมายต่างกับ $\delta_i(n)$ ทั้งนี้การที่เพิ่มขนาดของช่วงก้าวขึ้นแบบเชิงเส้น แต่ลดขนาดของช่วงก้าวลงแบบเอกซ์โพเนนเชียล เพื่อเป็นการป้องกันไม่ให้ขนาดของช่วงก้าวมีขนาดใหญ่เกินไป หรือเพิ่มขึ้นเร็วเกินไป เพราะการลดขนาดของช่วงก้าวแบบนี้ทำให้ขนาดของช่วงก้าวสามารถลดลงได้อย่างรวดเร็ว อีกทั้งยังแน่ใจได้ว่าขนาดของช่วงก้าวจะเป็นบวกตลอดเวลา

ตารางที่ 2.1 เปรียบเทียบความซับซ้อนในการคำนวณของวิธีการปรับช่วงก้าวแบบต่าง ๆ

วิธีการปรับขนาดช่วงก้าว	Squared Error	Normalized Step-Size	Cross Correlation	Delta-Bar-Delta
การบวก	1 ครั้ง	2 ครั้ง	M -1 ครั้ง	M +(M /3) ครั้ง
การลบ	-	1 ครั้ง	1 ครั้ง	1+(M /3) ครั้ง
การคูณ	3 ครั้ง	3 ครั้ง	M +4 ครั้ง	4M +(M /3) ครั้ง
การหาร	-	1 ครั้ง	-	-
การเปรียบเทียบ	2 ครั้ง	-	-	M ครั้ง
ค่าคงตัว	4 ตัว	3 ตัว	2 ตัว	3 ตัว
ตัวแปร	-	1 ตัว	-	M ตัว
ฟังก์ชัน Exponential	-	-	1 ครั้ง	-

จากวิธีการปรับช่วงก้าวแบบพลวัตทั้ง 4 วิธีที่นำเสนอมาแล้วข้างต้นนี้ สามารถสรุปกระบวนการคำนวณของทั้ง 4 วิธีได้ดังตารางที่ 2.1

เนื่องจากการสร้างจริงวงจรกรองปรับตัว เพื่อใช้สำหรับแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ในวิทยานิพนธ์นี้จะนำขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาประยุกต์ใช้ เพื่อลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ดังนั้นการเลือกวิธีการปรับช่วงก้าวแบบพลวัตที่จะนำมาใช้ในการปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ควรจะเป็นวิธีการปรับช่วงก้าวที่ไม่ซับซ้อน วิทยานิพนธ์นี้จึงเลือกใช้วิธีการปรับช่วงก้าวแบบ Squared Error ที่นำเสนอโดย Kwong และ Johnston มาใช้ในการสร้างจริงวงจรกรองปรับตัวที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

บทที่ 3

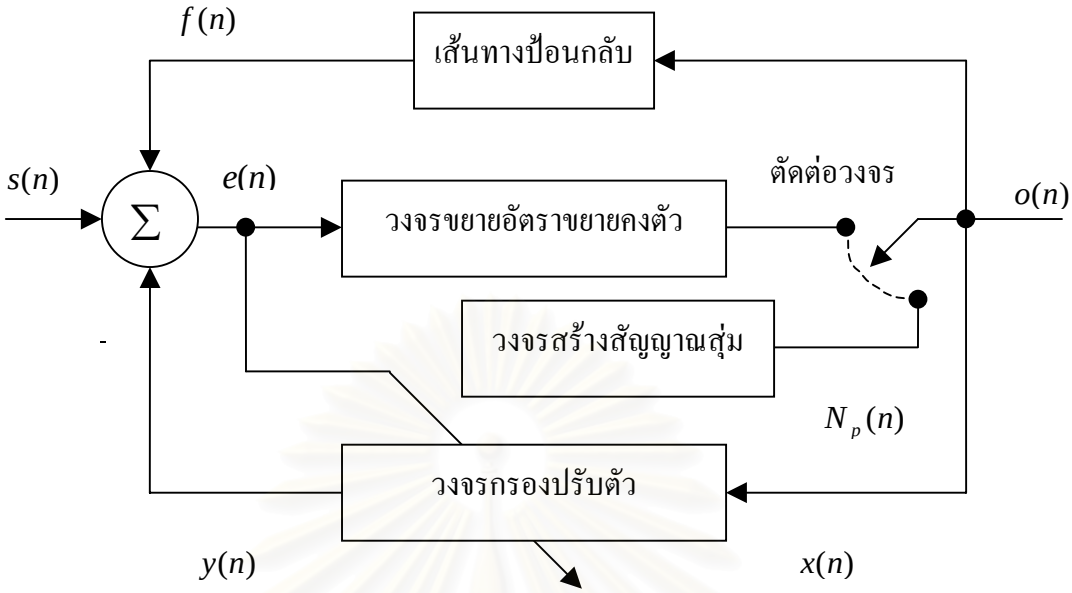
การจำลองผลของเครื่องลดทอนสัญญาณป้อนกลับทางเสียง

เนื้อหาในบทนี้จะกล่าวถึง แบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งการจำลองผลการทำงานจะกระทำใน 2 ลักษณะ คือ การจำลองผลการทำงานเชิงโปรแกรมและการจำลองผลการทำงานเชิงฮาร์ดแวร์ โดยที่การจำลองผลการทำงานเชิงโปรแกรมในแต่ละขั้นตอนของการประมวลผลจะทำการคำนวณแบบ Floating Point Arithmetic ส่วนการจำลองผลการทำงานเชิงฮาร์ดแวร์ในแต่ละขั้นตอนของการประมวลผลจะทำการคำนวณแบบ Fixed Point Arithmetic ในส่วนของการจำลองผลการทำงานเชิงฮาร์ดแวร์ มีวัตถุประสงค์เพื่อจำลองผลการทำงานที่จะเกิดขึ้นจริง เมื่อนำไปสร้างจริงในรูปแบบของฮาร์ดแวร์ โดยจะทำการวิเคราะห์รายละเอียด (Resolution) จำนวนบิตของข้อมูลในแต่ละขั้นตอนที่จะใช้ในการสร้างจริงของวงจรกรองปรับตัวบน FPGA ซึ่งการจำลองผลการทำงานของวงจรที่ออกแบบในบทนี้จะกระทำบนโปรแกรม Matlab

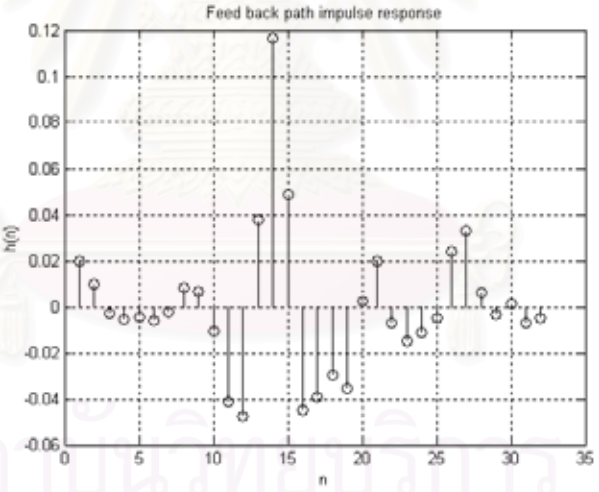
3.1 แบบจำลองของปัญหาการป้อนกลับทางเสียงในเครื่องช่วยฟัง

เนื่องจากโครงการวิทยานิพนธ์นี้จะทำการสร้างจริงวงจรกรองปรับตัว เพื่อใช้สำหรับสร้างสัญญาณเลียนแบบสัญญาณป้อนกลับที่เกิดขึ้นจริง ที่ทำให้เกิดปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง วิทยานิพนธ์นี้จะใช้แบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังเป็นแบบที่มีการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวอย่างไม่ต่อเนื่อง ดังแสดงในรูปที่ 3.1 โดยมีลักษณะการทำงานดังนี้ คือ วงจรกรองปรับตัวจะปรับค่าสัมประสิทธิ์เฉพาะช่วงที่เสียงเบามาก (สัญญาณขาเข้ามีกำลังต่ำ) หรืออาจจะทำการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวในช่วงอื่นด้วย เช่น ช่วงใกล้จะเกิดเสียงหอน (ต้องเพิ่มวงจรคอยตรวจสอบ) รวมทั้งอาจกำหนดให้วงจรกรองปรับตัวปรับค่าสัมประสิทธิ์แบบรายคาบ เช่น ปรับค่าทุก ๆ 3 นาที เป็นต้น

แบบจำลองของปัญหานี้จะใช้ผลตอบสนองต่ออิมพัลส์ (Impulse Response) ของเส้นทางป้อนกลับ [16] ที่หาได้จากผลตอบสนองทางความถี่ (Frequency Response) ของเส้นทางป้อนกลับที่วัดจริงจากเครื่องช่วยฟัง [5] โดยทำการสุ่มตัวอย่างจากผลตอบสนองทางความถี่ที่วัดได้ จากนั้นใช้การแปลงกลับดิสครีทไทม์ฟูริเยร์ (Inverse Discrete-Time Fourier Transform) [17] ซึ่งจะได้ผลตอบสนองต่ออิมพัลส์ดังแสดงในรูปที่ 3.2



รูปที่ 3.1 แบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบไม่ต่อเนื่อง



รูปที่ 3.2 ผลตอบสนองต่ออิมพัลส์ของเส้นทางป้อนกลับ

กำหนดให้ที่เวลา n สัญญาณต่าง ๆ มีคุณสมบัติดังนี้

$Np(n)$ คือ สัญญาณสุ่มเพื่อช่วยในการปรับตัวของวงจรกรองปรับตัว กำหนดให้มีคุณสมบัติเป็นสัญญาณสุ่มแบบ White Gaussian มีค่าเฉลี่ยเป็นศูนย์และมีค่าความแปรปรวนเท่ากับ 0.1

$s(n)$ คือ สัญญาณขาเข้าของเครื่องช่วยฟัง กำหนดให้มีคุณสมบัติเป็นสัญญาณสุ่ม แบบ White Gaussian มีค่าเฉลี่ยเป็นศูนย์และกำหนดให้สัญญาณ $s(n)$ มีความแปรปรวน 3 ระดับ คือ 10

1 และ 0.1 เท่าของสัญญาณ $Np(n)$ สัญญาณ $s(n)$ แทนระดับความดังของเสียง 3 ระดับคือ แทนเสียงที่มีความดังปกติ เสียงเบาและเสียงเบามาก

$\vec{x}(n)$ คือ เวกเตอร์ของสัญญาณขาเข้าของวงจรกรองปรับตัว

$$\vec{x}(n) = (x(n), x(n-1), ..., x(n-M+1))^T \quad (3\text{-}1)$$

โดยที่ M คือ อันดับของวงจรกรองปรับตัว

$\vec{h}$ คือ เวกเตอร์ผลตอบสนองต่ออิมพัลส์ของเส้นทางป้อนกลับทางเสียงในเครื่องช่วยฟัง

$$\vec{h} = (h(0), h(1), ..., h(M-1))^T \quad (3\text{-}2)$$

$f(n)$ คือ สัญญาณที่เกิดจากการป้อนกลับทางเสียงในเครื่องช่วยฟังมีค่าตามสมการ

$$f(n) = \vec{h}^T \vec{x}(n) \quad (3\text{-}3)$$

$\vec{w}(n)$ คือ เวกเตอร์ของสัมประสิทธิ์ของวงจรกรองปรับตัว

$$\vec{w}(n) = (w_1(n), w_2(n), ..., w_M(n))^T \quad (3\text{-}4)$$

$y(n)$ คือ สัญญาณขาออกของวงจรกรองปรับตัว มีค่ากำหนดตามสมการ

$$y(n) = \vec{w}^T(n) \vec{x}(n) \quad (3\text{-}5)$$

$e(n)$ คือ ความผิดพลาด มีค่ากำหนดตามสมการ

$$e(n) = s(n) + f(n) - y(n) \quad (3\text{-}6)$$

$o(n)$ คือ สัญญาณขาออกของเครื่องช่วยฟัง

วิทยานิพนธ์นี้จะใช้แบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบไม่ต่อเนื่อง ดังแสดงในรูปที่ 3.1 ซึ่งการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวจะทำการปรับค่าสัมประสิทธิ์เฉพาะช่วงที่เสียงเบามาก (สัญญาณขาเข้ามีกำลังต่ำ) เพราะจะทำให้ค่าสัมประสิทธิ์ของวงจรกรองลู่เข้าสู่สภาวะออปติมัมได้ง่ายกว่า การปรับค่าสัมประสิทธิ์ในช่วงที่มีเสียงดัง แบบจำลองในรูปที่ 3.1 นี้จะถูกนำไปใช้ในการวิเคราะห์และพิจารณาทั้ง 2 รูปแบบ คือ ทั้งการประมวลผลแบบ Floating Point Arithmetic และ Fixed Point Arithmetic ตามลำดับ ด้วยโปรแกรม Matlab หลังจากนั้นจึงนำผลที่ได้จากการคำนวณแบบ Fixed Point Arithmetic มา

วิเคราะห์และพิจารณาก่อนที่จะนำไปออกแบบและสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังบน FPGA ต่อไป

3.2 การจำลองผลโดยใช้การคำนวณแบบ Floating Point Arithmetic

ตามที่ได้กล่าวมาแล้วก่อนหน้านี้ที่ว่า วิทยานิพนธ์นี้จะทำการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งจะนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาประยุกต์ใช้ เพื่อลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ซึ่งในแต่ละรอบของการสุ่มตัวอย่างต้องใช้การคำนวณที่ประกอบด้วยการคูณ $M+1$ ครั้ง และการบวก $M+1$ ครั้ง ให้เหลือเพียงการบวก $M+1$ ครั้ง เท่านั้น ดังที่แสดงให้เห็นในสมการที่ (2-16) และสมการที่ (2-18) ในบทที่ 2 นอกจากนี้ในวิทยานิพนธ์นี้จะนำเอาการปรับช่วงก้าวแบบพลวัตมาใช้เพื่อช่วยเพิ่มประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ซึ่งจะใช้วิธีการปรับช่วงก้าวแบบ Squared Error ที่มีหลักการไม่ซับซ้อนมากนัก ซึ่งการจำลองผลในหัวข้อนี้จะแบ่งเป็น 2 ส่วน ดังนี้ คือ การจำลองผลเพื่อเปรียบเทียบประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดกับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และในส่วนที่ 2 การจำลองผลจะแสดงให้เห็นถึงการนำวิธีการปรับช่วงก้าวแบบพลวัตมาใช้เพื่อปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign

3.2.1 การจำลองผลของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดและขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign

จากแบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบไม่ต่อเนื่องดังแสดงในรูปที่ 3.1 ซึ่งจะทำการปรับค่าสัมประสิทธิ์เฉพาะช่วงที่เสียงเบามาก (สัญญาณขาเข้ามีกำลังต่ำ) โดยที่สัญญาณต่าง ๆ ที่จะนำมาใช้ในการวิเคราะห์และพิจารณาจะมีข้อกำหนดตามหัวข้อที่ 3.1 คือ สัญญาณสุ่ม $Np(n)$ ที่ช่วยในการปรับตัวของวงจรกรองเป็นสัญญาณสุ่มแบบ White Gaussian มีค่าเฉลี่ยเป็นศูนย์และมีค่าความแปรปรวนเท่ากับ 0.1 ในส่วนของสัญญาณขาเข้า $s(n)$ ที่นำมาพิจารณาจะทำการพิจารณาในช่วงที่สัญญาณขาเข้ามีกำลังต่ำ ซึ่งสัญญาณขาเข้าเป็นสัญญาณแบบ White Gaussian มีค่าเฉลี่ยเป็นศูนย์และมีค่าความแปรปรวนเท่ากับ 0.1 เท่าของสัญญาณสุ่ม $Np(n)$ (สัญญาณขาเข้า $s(n)$ มีค่าความแปรปรวนเท่ากับ 0.01)

คุณสมบัติของสัญญาณสุ่ม $Np(n)$ เป็นสัญญาณแบบ White Gaussian มีค่าเฉลี่ยเป็นศูนย์และมีความแปรปรวนเท่ากับ 0.1 ซึ่งหมายความว่า สัญญาณสุ่มเป็นสัญญาณที่มีขนาดเป็นเลขจำนวนจริง มีค่าเป็นไปได้ทั้งจำนวนจริงบวก จำนวนจริงลบและจำนวนเต็มศูนย์ โดยที่มีความ

แปรปรวน (Variance) เท่ากับ 0.1 และมีส่วนเบี่ยงเบนมาตรฐาน (Standard Deviation) ของสัญญาณสุ่มเท่ากับกรณ์ที่สองของค่าความแปรปรวน โดยที่ค่าความแปรปรวนจะเป็นตัวกำหนดขนาดของกำลังของสัญญาณที่ใช้ในการพิจารณา

ความแปรปรวนของสัญญาณสุ่ม $Np(n)$ มีค่าดังสมการ

$$\sigma^2_{N_{p(n)}} = 0.1 \tag{3-7}$$

จะได้ส่วนเบี่ยงเบนมาตรฐานของสัญญาณสุ่ม $Np(n)$ ดังนี้

$$\sqrt{\sigma^2_{N_{p(n)}}} = 0.3162278 \tag{3-8}$$

โดยทั่วไป 3 เท่า ของส่วนเบี่ยงเบนมาตรฐานทั้งทางด้านบวกและลบของค่าเฉลี่ยจะครอบคลุมขนาดของสัญญาณสุ่มได้ทั้งหมดร้อยละ 99.73 เพราะฉะนั้น 3 เท่าของส่วนเบี่ยงเบนมาตรฐานทั้งทางด้านบวกและลบเท่ากับ ± 3 คูณกับส่วนเบี่ยงเบนมาตรฐานจะได้เท่ากับ ± 0.94868 ซึ่งหมายความว่าขนาดของสัญญาณสุ่มสูงสุดและต่ำสุดมีขนาดอยู่ที่ประมาณ 0.94868

คุณสมบัติของสัญญาณขาเข้า $s(n)$ ของเครื่องช่วยฟังมีคุณสมบัติเหมือนกับสัญญาณสุ่ม $Np(n)$ ซึ่งในแบบจำลองนี้จะทำการพิจารณาเฉพาะช่วงที่เสียงเบามาก (สัญญาณขาเข้ามีกำลังต่ำ) คือ พิจารณาเฉพาะช่วงที่สัญญาณขาเข้า $s(n)$ มีความแปรปรวนเท่ากับ 0.01

ความแปรปรวนของสัญญาณสุ่ม $s(n)$ มีค่าดังสมการ

$$\sigma^2_{S(n)} = 0.01 \tag{3-9}$$

จะได้ส่วนเบี่ยงเบนมาตรฐานของสัญญาณขาเข้า $s(n)$ ดังนี้

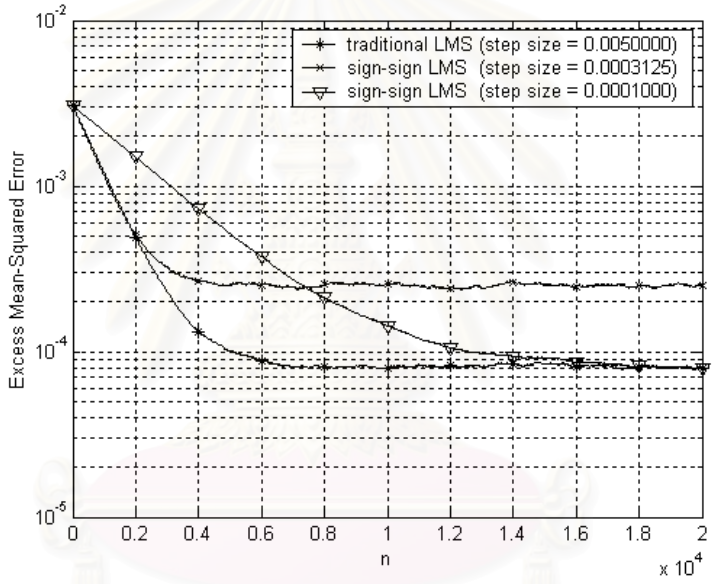
$$\sqrt{\sigma^2_{S(n)}} = 0.1 \tag{3-10}$$

เช่นเดียวกัน 3 เท่า ของส่วนเบี่ยงเบนมาตรฐานทั้งทางด้านบวกและลบของค่าเฉลี่ยจะครอบคลุมขนาดของสัญญาณสุ่มได้ทั้งหมดร้อยละ 99.73 เพราะฉะนั้น 3 เท่าของส่วนเบี่ยงเบนมาตรฐานทั้งทางด้านบวกและลบเท่ากับ ± 3 คูณกับส่วนเบี่ยงเบนมาตรฐานจะได้เท่ากับ ± 0.3 ซึ่งหมายความว่าขนาดของสัญญาณสุ่มสูงสุดและต่ำสุดมีขนาดอยู่ที่ประมาณ 0.3

คุณสมบัติของสัญญาณสุ่ม $Np(n)$ และสัญญาณขาเข้า $s(n)$ ที่กล่าวมาแล้วข้างต้นนี้จะถูกนำมาใช้ในการวิเคราะห์และพิจารณาในการจำลองผลการทำงานเชิงโปรแกรมที่ใช้การคำนวณ

แบบ Floating Point Arithmetic การจำลองผลการทำงานเชิงฮาร์ดแวร์ที่ใช้การคำนวณแบบ Fixed Point Arithmetic และการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังบน FPGA

การจำลองผลการทำงานเชิงโปรแกรมซึ่งจะใช้การคำนวณแบบ Floating Point Arithmetic ในหัวข้อนี้จะแสดงให้เห็นถึงผลที่ได้จากการใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดและขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign โดยที่การจำลองผลการทำงานในหัวข้อนี้จะใช้ผลตอบสนองต่ออิมพัลส์ของเส้นทางป้อนกลับที่วัดได้จริงตามรูปที่ 3.2 ซึ่งผลการจำลองการทำงานได้ผลลัพธ์ดังแสดงในรูปที่ 3.3



รูปที่ 3.3 ผลการจำลองของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดเปรียบเทียบกับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign สำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง

จากผลการจำลองทางคอมพิวเตอร์ในรูปที่ 3.3 ซึ่งเป็นเส้นกราฟแสดงผลที่ได้จากการจำลองทางคอมพิวเตอร์ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด เปรียบเทียบกับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign สำหรับวงจรกรองปรับตัวแบบทรานส์เวอร์อันดับ 32 สำหรับอันดับของวงจรกรองที่นำมาใช้ กำหนดให้มีขนาดเท่ากับผลตอบสนองต่ออิมพัลส์ของเส้นทางป้อนกลับในรูปที่ 3.2 โดยที่สัญญาณสุ่ม $Np(n)$ และสัญญาณขาเข้า $s(n)$ มีคุณสมบัติตามที่กล่าวมาแล้วข้างต้น รูปกราฟดังกล่าวแสดงค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินที่ตำแหน่งเวลา n (MSE = Minimum MSE + Excess MSE) สำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง กราฟแต่ละเส้นเป็นผลเฉลี่ยที่ได้จากการจำลองทั้งหมด 100 การทดลองที่เป็นอิสระจากกัน

สำหรับกรณีที่ค่าช่วงก้าวของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มีค่าเป็น 1 ใน 16 ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด อัตราการลดลงของค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินของทั้งสองขั้นตอนวิธีมีค่าใกล้เคียงกัน อยู่ที่ประมาณ $(n < 2000)$ แต่ที่สภาวะคงตัวของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign จะมีค่าสูงกว่าของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด เพื่อให้ได้ค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินของขั้นตอนวิธีทั้งสองมีค่าประมาณเท่ากัน ช่วงก้าวของขั้นตอนวิธีประเภท Sign-Sign ต้องลดลงมาเป็น 1 ใน 50 ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด

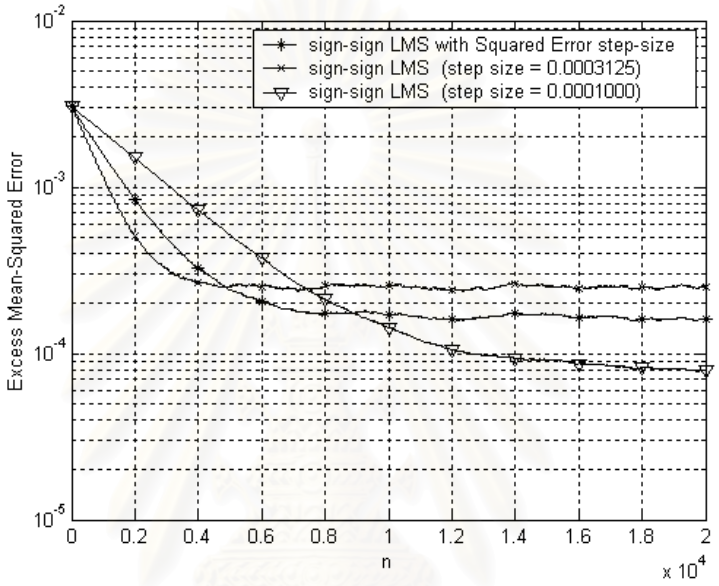
จากเส้นกราฟในรูปที่ 3.3 จะเห็นว่าค่าช่วงก้าวที่ใช้ในขั้นตอนวิธีกำสังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ทั้งสองกรณีมีอัตราการปรับตัวที่แตกต่างกัน ในกรณีแรกที่ค่าช่วงก้าวมีค่าเท่ากับ 1 ใน 16 ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด อัตราการปรับตัวจะสูงแต่ความถูกต้องที่สภาวะคงตัวจะต่ำ ซึ่งมีค่าสูงกว่าประมาณ 3 เท่าของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด สำหรับกรณีที่สองที่ค่าช่วงก้าวมีค่าเท่ากับ 1 ใน 50 ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด อัตราการปรับตัวจะต่ำแต่ความถูกต้องที่สภาวะคงตัวจะสูง เมื่อเปรียบเทียบกับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ซึ่งจะให้ค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินใกล้เคียงกันอยู่ที่ประมาณ $(n > 16000)$

จากรูปที่ 3.3 แสดงให้เห็นอย่างชัดแล้วว่า ผลของการลดความซับซ้อนในการคำนวณของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดมาเป็นขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ทำให้ประสิทธิภาพในการปรับตัวของขั้นตอนวิธีลดลง แต่เนื่องจากวิทยานิพนธ์นี้ต้องการที่จะลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ซึ่งมีความซับซ้อนในการคำนวณมากไม่เหมาะแก่การสร้างจริงในรูปแบบของวงจรรวม VLSI ซึ่งมีผลทำให้ต้องใช้พื้นที่ในการสร้างจริงที่สูงและต้องใช้พลังงานในการทำงานสูงอีกด้วย จึงได้นำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาประยุกต์ใช้เพื่อลดความซับซ้อนของขั้นตอนวิธี โดยจะนำเอาการปรับช่วงก้าวแบบพลวัต ซึ่งจะใช้วิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง ที่มีความซับซ้อนในการคำนวณไม่มากนักมาใช้เพื่อปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ซึ่งผลการจำลองของการนำเอาวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign จะแสดงให้เห็นในหัวข้อต่อไป

3.2.2 การจำลองผลของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign โดยใช้การปรับช่วงก้าวแบบพลวัต

จากผลการจำลองทางคอมพิวเตอร์ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดเปรียบเทียบกับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ในหัวข้อที่แล้ว เป็นที่ทราบกันดีแล้วว่าผลจาก

การลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ทำให้ประสิทธิภาพในการปรับตัวของขั้นตอนวิธีลดลง ในหัวข้อนี้จะทำการจำลองผลทางคอมพิวเตอร์ โดยการนำเอาวิธีการปรับช่วงก้าวแบบพลวัตแบบค่าผิดพลาดกำลังสอง มาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign เพื่อใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว เพื่อใช้สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง



รูปที่ 3.4 ผลการจำลองของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign เปรียบเทียบกับขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองสำหรับปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟัง

จากผลการจำลองทางคอมพิวเตอร์ในรูปที่ 3.4 จะเห็นว่าเส้นกราฟของค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินในช่วงแรกของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธี จะมีค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินสูงกว่าขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้ค่าช่วงก้าวเท่ากับ 1 ใน 16 ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดเล็กน้อยแต่ที่สภาวะคงตัวจะต่ำกว่าประมาณ 1.5 เท่า และเมื่อเปรียบเทียบกับเส้นกราฟของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่มีค่าช่วงก้าวเท่ากับ 1 ใน 50 ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกินของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองจะมีค่าสูงกว่าเล็กน้อยแต่จะเข้าสู่สภาวะคงตัวเร็วกว่ามาก (ประมาณ 10000 ครั้งของการสุ่มตัวอย่าง) สำหรับค่าคงตัวของวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองที่นำมาใช้มีค่าดังนี้คือ α มีค่าเท่ากับ 0.69, γ มีค่าเท่ากับ 0.005, $\mu_{\min}$ มีค่าเท่ากับ

0.0001 และ μ_{max} มีค่าเท่ากับ 0.0005 โดยที่กราฟแต่ละเส้นเป็นผลเฉลี่ยที่ได้จากการจำลองทั้งหมด 100 การทดลองที่เป็นอิสระจากกัน

จากรูปที่ 3.4 แสดงให้เห็นว่า การนำเอาวิธีการปรับช่วงก้าวแบบพลวัตมาใช้ เพื่อปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign สามารถช่วยทำให้ประสิทธิภาพของขั้นตอนวิธีดังกล่าวดีขึ้น ดังนั้นสามารถสรุปได้ว่าการนำเอาวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองมาใช้ เพื่อช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่นำมาใช้แทนขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด เพื่อลดความซับซ้อนของขั้นตอนวิธี ทำให้ได้ผลลัพธ์ที่ดีขึ้นจริง เมื่อเปรียบเทียบกับการใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign แต่เพียงอย่างเดียว

จากผลการจำลองการทำงานของแบบจำลองการแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบไม่ต่อเนื่อง ในส่วนของการจำลองที่ใช้การคำนวณแบบ Floating Point Arithmetic ถึงแม้ว่าประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign จะด้อยกว่าขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด แต่เมื่อนำเอาการปรับช่วงก้าวแบบพลวัตเข้ามาใช้เพื่อช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ทำให้ประสิทธิภาพที่ได้ดีขึ้น แต่ความซับซ้อนในการคำนวณยังคงน้อยกว่าขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด ซึ่งเป็นสิ่งที่แสดงให้เห็นถึงประสิทธิภาพในการนำเอาขั้นตอนวิธีดังกล่าวและวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองไปประยุกต์ใช้งานจริง

3.3 รายละเอียดการออกแบบบิตข้อมูล (Resolution of Data)

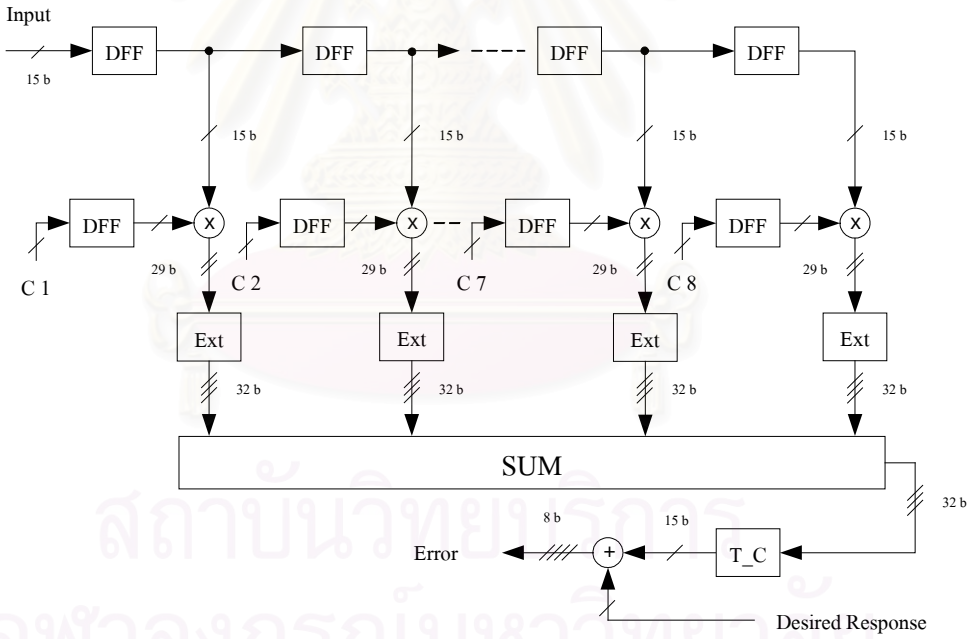
จากผลการจำลองทางคอมพิวเตอร์ในหัวข้อที่ 3.2 แสดงให้เห็นแล้วว่าสามารถนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบพลวัตด้วยค่าผิดพลาดกำลังสอง มาช่วยในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง มีประสิทธิภาพในการทำงานอยู่ในระดับที่ดี ถึงแม้ว่าจะด้อยกว่าขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดก็ตาม แต่ก็ทำให้ความซับซ้อนของขั้นตอนวิธีลดลงอย่างมาก ซึ่งจะเป็นประโยชน์ในการสร้างจริงที่จะช่วยลดความซับซ้อนในการสร้างจริง รวมถึงลดพื้นที่ในการสร้างจริงและยังส่งผลถึงการใช้พลังงานที่จะลดลงในการปรับลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดลง

อย่างไรก็ตามผลการจำลองทางคอมพิวเตอร์ที่ผ่านมาใช้การคำนวณแบบ Floating Point Arithmetic ในการประมวลผลของแต่ละขั้นตอน แต่เนื่องจากในระบบการทำงานจริงของฮาร์ดแวร์

การประมวลผลในขั้นตอนต่าง ๆ จะใช้การคำนวณในระดับบิตซึ่งจะใช้การคำนวณในรูปแบบของเลขฐานสองเพื่อประหยัดฮาร์ดแวร์ ค่าของสัญญาณที่ใช้ในการคำนวณจะแทนอยู่ในรูปแบบ Fixed Point ดังนั้นการจำลองผลการทำงานที่ถูกต้องควรจะทำการจำลองผลในรูปแบบเชิงฮาร์ดแวร์ โดยที่การประมวลผลในแต่ละขั้นตอนจะต้องทำการประมวลผลในรูปแบบของเลขฐานสองแบบ Fixed Point Arithmetic เพื่อจำลองผลการทำงานที่ได้ซึ่งจะเกิดขึ้นจริง เมื่อทำการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

สำหรับการวิเคราะห์ในการออกแบบการทำงานของวงจรกรองปรับตัวที่จะทำการสร้างจริงที่ใช้การคำนวณแบบ Fixed Point Arithmetic ซึ่งจะกล่าวถึงต่อไป จะแบ่งเป็น 2 ส่วน คือ ส่วนแรกจะทำการวิเคราะห์เกี่ยวกับวงจรกรอง และส่วนที่สองจะทำการวิเคราะห์ในส่วนของการปรับค่าสัมประสิทธิ์

3.3.1 รายละเอียดของค่าสัมประสิทธิ์ในส่วนต่าง ๆ ของวงจรกรองปรับตัว



รูปที่ 3.5 รายละเอียดของวงจรกรองปรับตัวที่ใช้ในการสร้างจริง

จากรูปที่ 3.5 เป็นรูปของวงจรกรองปรับตัวที่จะนำมาใช้ในการสร้างจริง เพื่อใช้แก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งทำการออกแบบจากแบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบไม่ต่อเนื่องในรูปที่ 3.1 โดยที่การสร้างจริงวงจรกรองปรับตัวจะทำการสร้างจริงบน FPGA ซึ่งมีข้อจำกัดทางด้านพื้นที่ จำนวน Gate และทรัพยากรต่าง ๆ ของชิป FPGA ดังนั้นการสร้างจริงวงจรกรองปรับตัวจะทำการสร้างจริง

วงจรกรองปรับตัวอันดับ 8 เพื่อให้สามารถนำวงจรที่ออกแบบด้วยภาษา VHDL มาทำการสร้างจริง และสามารถใช้งานได้จริงบน FPGA

ส่วนประกอบต่าง ๆ ในรูปที่ 3.5 ประกอบด้วย D ฟลิบฟลอบ สัญญาณขาเข้าหรือสัญญาณสุ่ม วงจรคูณ ตัวขยายจำนวนบิตข้อมูล (Ext) วงจรบวก (SUM) ส่วน Truncate ข้อมูลและ Complement (T_C) จากนั้นก็เป็นส่วนของสัญญาณที่ต้องการ (Desired Response) และค่าความผิดพลาด (Error) จากแบบจำลองในรูปที่ 3.1 Desired Response จะประกอบด้วยสัญญาณ 3 สัญญาณ คือ สัญญาณเสียงที่เกิดจากการป้อนกลับรวมกับสัญญาณขาเข้า $s(n)$ และหักล้างกับสัญญาณที่สร้างจากวงจรกรองปรับตัวทำให้ได้ค่าความผิดพลาดที่เวลา n ใด ๆ

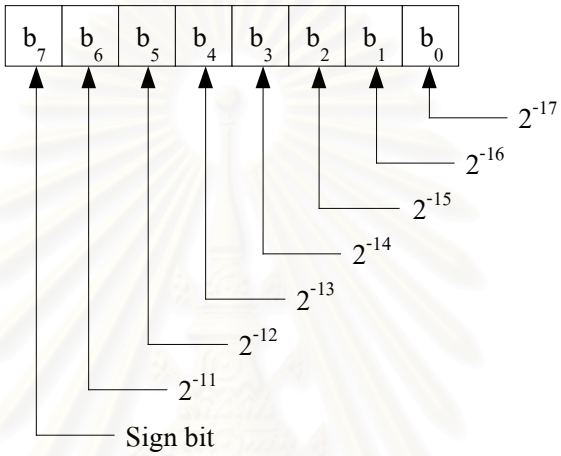
การออกแบบวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังที่จะใช้ในการสร้างจริงบน FPGA ในที่นี้จะใช้แบบจำลองของปัญหาการลดการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบไม่ต่อเนื่องเป็นต้นแบบในการสร้างจริง โดยจะเริ่มพิจารณาที่สัญญาณต่าง ๆ ที่ได้ทำการกำหนดคุณสมบัติเอาไว้แล้ว เริ่มต้นจากสัญญาณสุ่ม $Np(n)$ ที่ป้อนให้กับวงจรกรองปรับตัว สำหรับในรูปที่ 3.5 แทนสัญญาณสุ่มด้วยสัญญาณอินพุท (Input) ซึ่งมีคุณสมบัติดังที่กล่าวมาแล้วก่อนหน้านี้ คือ มีความแปรปรวนเท่ากับ 0.1 ซึ่งจะทำให้สามารถหาส่วนเบี่ยงเบนมาตรฐานของสัญญาณได้ สุดท้ายสามารถประมาณค่าขนาดของสัญญาณสูงสุดและต่ำสุดได้ ซึ่งข้อกำหนดดังกล่าวมีส่วนในการพิจารณาจำนวนบิตที่จะนำมาใช้ในการแทนขนาดของสัญญาณ สัญญาณที่ถูกป้อนเข้าวงจรกรองปรับตัวจะถูกนำไปคูณเข้ากับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว เนื่องจากค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่จะทำการสร้างจริงสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังจะใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง ซึ่งวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองมีสมการดังนี้

$$\mu'(n+1) = \alpha\,\mu(n) + \gamma\, e^2(n) \tag{3-11}$$

$$\mu(n+1) = \begin{cases} \mu_{\max}, & \mu'(n+1) > \mu_{\max} \\ \mu_{\min}, & \mu'(n+1) < \mu_{\min} \\ \mu'(n+1), & Others \end{cases} \tag{3-12}$$

เนื่องจากค่าคงตัวในสมการที่ (3-11) และสมการที่ (3-12) ที่จะนำมาใช้ในการสร้างจริงวงจรกรองปรับตัว จะใช้ค่าเดียวกับค่าคงตัวที่ใช้การจำลองผลเชิงโปรแกรมที่ใช้การคำนวณแบบ Floating Point Arithmetic ซึ่งมีค่าคงตัวดังนี้ คือ α มีค่าเท่ากับ 0.69, γ มีค่าเท่ากับ 0.005, $\mu_{\max}$ มี

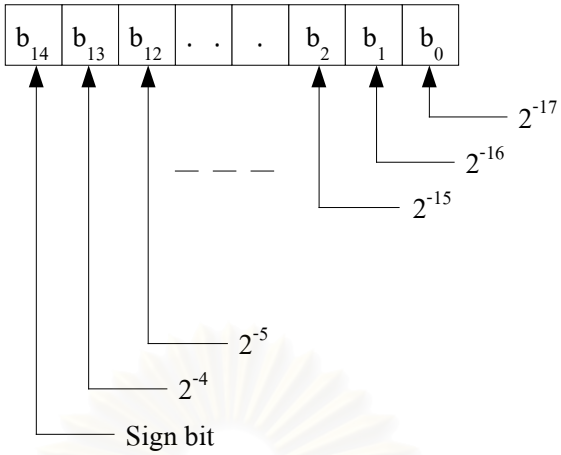
ค่าเท่ากับ 0.0005 และ μ_{min} มีค่าเท่ากับ 0.0001 จากการจำลองผลทางคอมพิวเตอร์ค่าช่วงก้าวของวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง สามารถแทนได้ด้วยเลขฐานสองจำนวน 8 บิต ซึ่งมีรายละเอียดดังนี้ ที่ตำแหน่งซ้ายสุดแทนตำแหน่งของบิตเครื่องหมาย (Sign bit) ตำแหน่งที่ 2 จากทางซ้ายแทนตำแหน่งของบิตที่มีค่าสูงสุด (Most Significant Bit: MSB) ซึ่งมีค่าประจำตำแหน่ง คือ 2^{-11} และบิตที่อยู่ที่ตำแหน่งทางขวาสุดเป็นบิตที่มีค่าต่ำสุด (Least Significant Bit: LSB) ซึ่งมีค่าประจำตำแหน่งคือ 2^{-17} สำหรับรายละเอียดในตำแหน่งอื่น ๆ สามารถดูได้จากรูปที่ 3.6



รูปที่ 3.6 รายละเอียดบิตข้อมูลของค่าช่วงก้าว (Step Size)

ในกรณีที่ค่าช่วงก้าวมีค่าสูงสุดสามารถแทนค่าด้วยเลขฐานสองได้เท่ากับ 01000001_b (มีค่าเป็นเลขจำนวนจริงเท่ากับ $2^{-11} + 2^{-17} = 0.00049591$) ส่วนกรณีที่ค่าช่วงก้าวมีค่าต่ำสุดสามารถแทนค่าด้วยเลขฐานสองได้เท่ากับ 00001101_b (มีค่าเป็นเลขจำนวนจริงเท่ากับ $2^{-14} + 2^{-15} + 2^{-17} = 0.000099182$) ซึ่งจากผลของการที่ค่าช่วงก้าวค่าอยู่ในช่วง 2^{-11} ถึง 2^{-17} รวมกับผลของค่าคงตัวของเส้นทางป้อนกลับที่นำมาใช้ในการสร้างจริงวงจรกรองปรับตัวอันดับ 8 ที่มีค่าอยู่ในช่วง –0.0475 ถึง 0.1167 (คัดเลือกจากผลตอบสนองต่ออิมพัลส์ของเส้นทางป้อนกลับที่ n เท่ากับ 11 ถึง 18 ตามรูปที่ 3.2) ทำให้จำเป็นต้องใช้ค่าสัมประสิทธิ์ของวงจรกรองปรับตัว (Tap Weight) แทนด้วยเลขฐานสองจำนวน15 บิต เพื่อให้สามารถครอบคลุมได้ถึงค่าสูงสุดและต่ำสุดของค่าช่วงก้าว ที่ใช้หลักการปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง

จากการที่ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวถูกออกแบบให้แทนค่าด้วยเลขฐานสองจำนวน 15 บิต ซึ่งรายละเอียดของแต่ละบิตแสดงให้เห็นดังรูปที่ 3.7 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-4} และที่ตำแหน่ง LSB เท่ากับ 2^{-17} จากผลที่ได้ทำให้ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวสามารถแทนค่าต่ำสุดได้เท่ากับ - 0.125 และสูงสุดเท่ากับ 0.12499



รูปที่ 3.7 รายละเอียดบิตข้อมูลของค่าสัมประสิทธิ์ของวงจรกรองปรับตัว

จากการที่ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวต้องใช้เลขฐานสองจำนวน 15 บิต เพื่อความสะดวกในการคำนวณ ดังนั้นสัญญาณขาเข้าของวงจรกรองปรับตัว (สัญญาณสุ่ม) ควรจะแทนค่าด้วยจำนวนบิตของข้อมูลที่เท่ากัน เพราะฉะนั้นจึงแทนสัญญาณขาเข้าของวงจรกรองปรับตัวด้วยจำนวนบิตเท่ากับ 15 บิต ซึ่งรายละเอียดของแต่ละบิตจะมีลักษณะใกล้เคียงกับรูปที่ 3.7 ต่างกันที่ค่าประจำตำแหน่งของแต่ละหลัก โดยที่ค่าสูงสุดของสัญญาณขาเข้ามีค่าประจำหลักเท่ากับ 2^{-1} (ตำแหน่งที่ 2 จากซ้าย) และค่าต่ำสุดมีค่าประจำหลักเท่ากับ 2^{-14} (ตำแหน่งขวาสุด) จากคุณสมบัติของสัญญาณสุ่มที่มีค่าความแปรปรวนเท่ากับ 0.1 และเนื่องจากการใช้เลขฐานสองจำนวน 15 บิต ในการแทนค่าสัญญาณสุ่ม ทำให้สามารถแทนค่าต่ำสุดได้เท่ากับ – 1 และสูงสุดเท่ากับ 0.99994

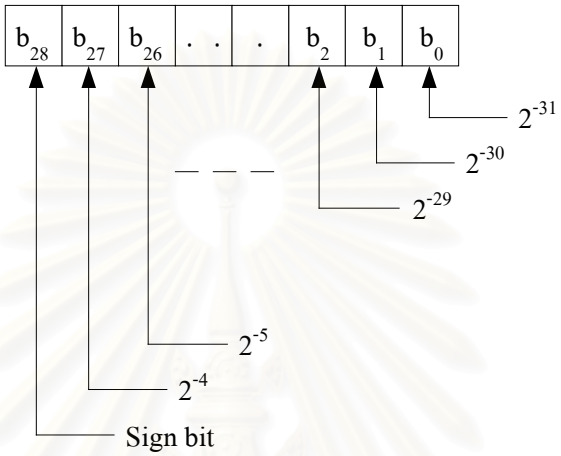
จากการที่ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวและสัญญาณขาเข้าของวงจรกรองปรับตัวใช้เลขฐานสองจำนวน 15 บิต แทนค่าเลขจำนวนจริง เพื่อที่จะทำการประมวลโดยใช้การคำนวณในรูปแบบ Fixed Point Arithmetic ซึ่งเป็นการจำลองการทำงานของระบบในรูปแบบของฮาร์ดแวร์ ซึ่งค่าสูงสุดของการคำนวณจะเกิดขึ้นในกรณีที่ทั้งค่าสัมประสิทธิ์ของวงจรกรองปรับตัวและสัญญาณขาเข้าของวงจรกรองปรับตัวมีค่าเท่ากับ 100000000000000_b ซึ่งจะได้ผลคูณดังนี้

$$(100000000000000_b)*(100000000000000_b) = 01000000000000000000000000000000_b$$

จากการคำนวณในกรณีที่ค่าทั้งสองมีค่าเป็นลบต่ำสุดทั้ง 2 ค่าจะให้ผลลัพธ์ที่เป็นค่าบวกสูงสุด ซึ่งต้องใช้จำนวนบิตของเลขฐานสองแทนค่าดังกล่าวเป็นจำนวน 30 บิต กรณีอื่น ๆ สามารถใช้จำนวนบิตข้อมูลเพียง 29 บิตก็เพียงพอ ซึ่งโอกาสที่จะเกิดขึ้นน้อยมาก ดังนั้นในการออกแบบจึงแทนผลลัพธ์ของผลคูณทั้งสองดังนี้

$$(100000000000000_b)*(100000000000000_b) = 011111111111111111111111111111_b$$

จากผลลัพธ์ที่ได้จากกระบวนการคูณ ซึ่งได้ผลลัพธ์เป็นเลขฐานสองจำนวน 29 บิต ซึ่งรายละเอียดของแต่ละบิตแสดงให้เห็นดังรูปที่ 3.8 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-4} และลดลงไปตำแหน่งละ 2^{-1} จนกระทั่งถึง LSB มีค่าประจำตำแหน่งเท่ากับ 2^{-31}

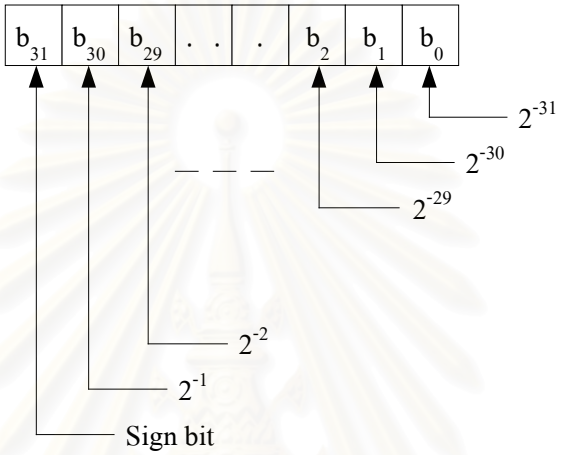


รูปที่ 3.8 รายละเอียดบิตข้อมูลที่ได้จากการคูณของสัญญาณสุ่มกับค่าสัมประสิทธิ์ของวงจรกรอง

จากรูปที่ 3.5 เมื่อผ่านกระบวนการคูณเรียบร้อยแล้ว จะได้ผลลัพธ์เป็นเลขฐานสองจำนวน 29 บิต เนื่องจากการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน สำหรับแก้ปัญหาการป้อนกลับทางเสียงในเครื่องช่วยฟัง จะทำการสร้างจริงโดยใช้วงจรกรองปรับตัวอันดับ 8 ดังนั้นค่าสูงสุดที่เป็นไปได้ คือ นำเอากรณีที่ผลคูณเป็นค่าสูงสุด คูณด้วยอันดับของวงจรกรองปรับตัว ในที่นี้ผลคูณค่าสูงสุด คือ $011111111111111111111111111111_b$ ซึ่งมีค่าประจำตำแหน่งต่าง ๆ ตามรูปที่ 3.8 (เมื่อทำการแปลงกลับเป็นเลขฐานสิบจะได้ค่าเท่ากับ 0.12499) จากนั้นนำไปคูณกับค่าอันดับของวงจรกรองปรับตัวซึ่งเท่ากับ 8 จะได้ค่าเท่ากับ 0.99999 เพราะฉะนั้นเพื่อให้ในส่วนของการบวกของข้อมูลที่ผ่านกระบวนการคูณมาเรียบร้อยแล้ว เป็นค่าผลรวมที่ถูกต้อง จะต้องใช้จำนวนบิตให้สามารถครอบคลุมค่าสูงสุดและต่ำสุดที่สามารถเกิดขึ้นได้ จากความเป็นไปได้เมื่อแต่ละ Tap ของวงจรกรองปรับตัวมีผลคูณเป็นค่าสูงสุดที่ทำให้ผลรวมเท่ากับ 0.99999 ดังนั้นในการออกแบบในส่วนนี้ได้ทำการขยายบิตข้อมูลเพิ่มขึ้น 3 บิต หลังจากที่ทำกระบวนคูณเสร็จเรียบร้อยแล้ว โดยทำการพิจารณาที่ Sign bit ของผลคูณ หากมีค่าเป็น 1 จะเพิ่มบิตด้านหน้าที่เป็น 1 เข้าไป 3 ตัว ส่วนในกรณีที่เป็น 0 ก็จะเพิ่มบิตด้านหน้าที่เป็น 0 เข้าไป 3 ตัว เช่นเดียวกัน ซึ่งขั้นตอนที่ว่านี้หากพิจารณาเทียบกับรูปที่ 3.5 ก็จะอยู่ที่กระบวนการเพิ่มจำนวนบิตข้อมูล (Ext) จากนั้นก็ทำการบวกตามปกติ ในกรณีที่ผลลัพธ์ที่ได้จากการคูณมีค่าทั้งบวกและลบ ก็จะพิจารณาในลักษณะเดียวกัน ในกระบวน

การบวกก็จะทำงานไปตามปกติ เนื่องจากสัญญาณที่เข้าสู่ระบบได้ทำการแทนค่าด้วยเลขฐานสองที่มีค่าเป็นไปได้ทั้งบวกและลบเรียบร้อยแล้ว โดยใช้หลักการ 2's Complement เพื่อให้ง่ายในการรวมสัญญาณ

เมื่อผ่านกระบวนการบวกเรียบร้อยแล้ว ข้อมูลที่ได้จะมีขนาด 32 บิต ซึ่งมีรายละเอียดของบิตแสดงให้เห็นดังรูปที่ 3.9 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-1} และที่ LSB เท่ากับ 2^{-31}

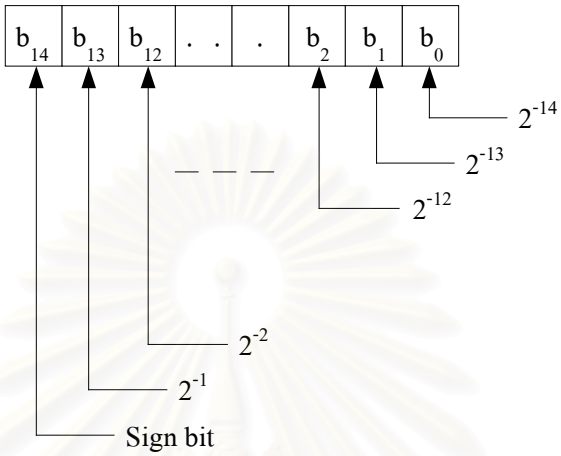


รูปที่ 3.9 รายละเอียดบิตข้อมูลที่สัญญาณขาออกของวงจรกรอง

เนื่องจากระบบที่ทำการออกแบบ ให้ขนาดของ Desired Response เป็นเลขฐานสองจำนวน 15 บิต ดังนั้นในขั้นตอนต่อไปจะนำเอาผลลัพธ์ที่ได้จากวงจรกรองปรับตัว ซึ่งมีขนาด 32 บิต โดยนำเอา 15 บิตแรกทางด้าน MSB มาใช้งาน ส่วนที่เหลือ 17 บิตทางด้าน LSB ตัดทิ้ง จากนั้นก็นำเอาข้อมูลทั้ง 15 บิต ไปผ่านกระบวนการ 2's Complement ก่อนที่จะนำไปรวมกับ Desired Response

การออกแบบวงจรกรองปรับตัวอันดับ 8 ที่จะทำการสร้างจริงนี้ จะทำการรวมสัญญาณขาเข้า $s(n)$ ซึ่งมีค่าความแปรปรวนเท่ากับ 0.1 เท่าของสัญญาณสุ่ม $Np(n)$ กับสัญญาณเสียงที่ผ่านเส้นทางป้อนกลับก่อน เพื่อความสะดวกในการพิจารณาหาค่าความผิดพลาด ซึ่งในที่นี้จะแทนผลรวมของสัญญาณทั้งสองด้วย Desired Response โดยที่ค่าสูงสุดและต่ำสุดที่ได้ของ Desired Response มีขนาดเท่ากับ 0.68507 ทั้งทางด้านบวกและลบ ดังนั้นในการออกแบบจะใช้เลขฐานสองจำนวน 15 บิตแทนค่าดังกล่าว ซึ่งคิดจากขนาดของสัญญาณสูงสุดที่ผ่านเส้นทางป้อนกลับรวมกับขนาดของสัญญาณขาเข้า $s(n)$ สูงสุดของแบบจำลองที่เป็นได้

ซึ่งรายละเอียดของบิตข้อมูลของ Desired Response และผลลัพธ์ของวงจรกรองปรับตัวที่ผ่านกระบวนการ 2's Complement แสดงให้เห็นดังรูปที่ 3.10 ซึ่งมีค่าประตำแหน่งที่ MSB เท่ากับ 2^{-1} และที่ LSB เท่ากับ 2^{-14}



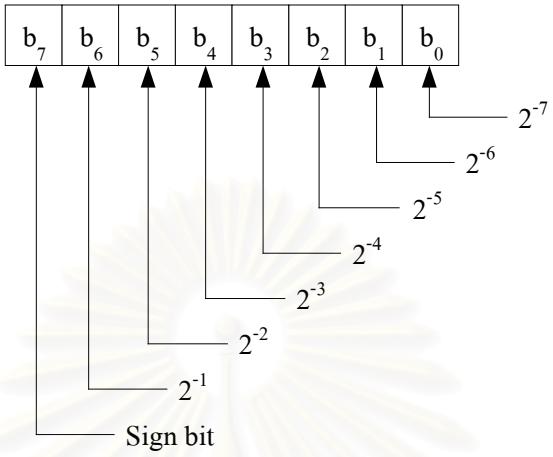
รูปที่ 3.10 รายละเอียดบิตข้อมูลของ Desired Response

เมื่อจำนวนบิตข้อมูลของทั้ง Desired Response และผลรวมของวงจรกรองปรับตัวที่ผ่านกระบวนการ 2's Complement เท่ากันแล้ว ต่อไปก็ทำการบวกข้อมูลทั้งสองเข้าด้วยกัน ก็จะได้ค่าความผิดพลาดที่มีขนาด 15 บิต สำหรับการออกแบบที่จะใช้ในการสร้างจริงจะใช้จำนวนบิตของค่าความผิดพลาดเท่ากับ 8 บิต โดยจะทำการเลือก 8 บิตทางด้าน MSB มาใช้ในการคำนวณในขั้นตอนของการปรับค่าช่วงก้าวที่จะนำไปใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวต่อไป นอกจากนี้ยังนำไปใช้ในการประเมินประสิทธิภาพที่ได้ของขั้นตอนวิธีอีกด้วย ซึ่งในการจำลองนี้จะแสดงให้เห็นในรูปของค่าความผิดพลาดกำลังสองเฉลี่ยส่วนเกิน รายละเอียดของบิตข้อมูลของค่าความผิดพลาดแสดงให้เห็นดังรูปที่ 3.11 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-1} และที่ LSB เท่ากับ 2^{-7} จากผลที่ได้ในแบบจำลองแสดงให้เห็นว่าค่าความผิดพลาดจะมีค่าสูงสุดที่ 0.99609 และมีค่าต่ำสุดที่ –1 เมื่อแทนด้วยเลขฐานสองจะได้ 01111111_b และ 10000000_b ตามลำดับ

ค่าความผิดพลาดที่คำนวณได้จะถูกนำไปใช้ในวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง และ Sign bit ของค่าความผิดพลาดจะถูกนำไปใช้ในการพิจารณาในการปรับค่าสัมประสิทธิ์ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign

ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่สำคัญ ๆ เพื่อใช้ในการออกแบบ ได้นำเสนอมาครบแล้ว ในส่วนต่อไปก็จะกล่าวถึงค่าสัมประสิทธิ์ในส่วนของขั้นตอนวิธีการปรับค่าสัมประสิทธิ์

ซึ่งประกอบด้วยขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และหลักการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง



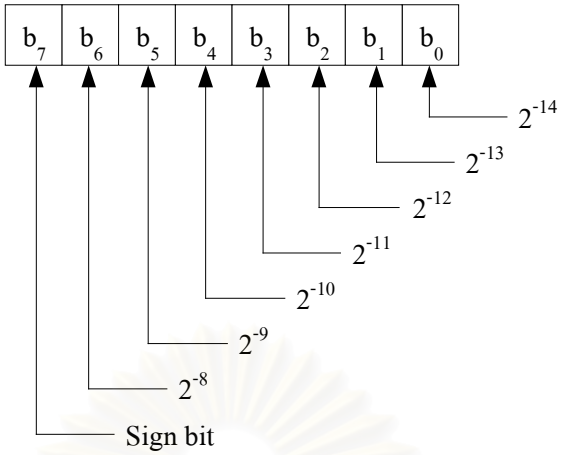
รูปที่ 3.11 รายละเอียดบิตข้อมูลของค่าความผิดพลาด $e(n)$

3.3.2 รายละเอียดของค่าสัมประสิทธิ์ในส่วนต่าง ๆ ของขั้นตอนวิธีการปรับค่าสัมประสิทธิ์

จากสมการของวิธีการปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสองในสมการที่ (3-11) และสมการที่ (3-12) ซึ่งมีค่าคงตัวดังนี้ คือ α มีค่าเท่ากับ 0.69, γ มีค่าเท่ากับ 0.005, μ_{max} มีค่าเท่ากับ 0.0005 และ μ_{min} มีค่าเท่ากับ 0.0001 ซึ่งค่าช่วงก้าวสูงสุดและต่ำสุดของวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง สามารถแทนได้ด้วยเลขฐานสองจำนวน 8 บิต ดังแสดงในรูปที่ 3.6

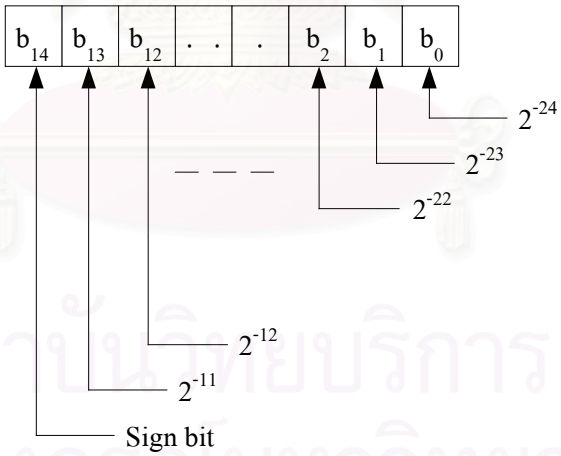
ค่าคงตัว α ซึ่งมีค่าเท่ากับ 0.69 สามารถแทนได้ด้วยเลขฐานสองจำนวน 8 บิต ซึ่งเท่ากับ 01011000_b ซึ่งมีค่าประจำตำแหน่งเช่นเดียวกับค่าความผิดพลาดในรูปที่ 3.11 ส่วนค่าช่วงก้าวที่ค่าสูงสุด เมื่อแทนเป็นเลขฐานสองจำนวน 8 บิต จะมีค่าเท่ากับ 01000001_b ในกรณีที่ค่าช่วงก้าวมีค่าต่ำสุดจะมีค่าเท่ากับ 00001101_b ซึ่งรายละเอียดของบิตข้อมูลแต่ละตำแหน่งสามารถดูได้จากรูปที่ 3.6 ส่วนค่า γ ซึ่งมีค่าเท่ากับ 0.005 สามารถแทนได้ด้วยเลขฐานสองจำนวน 8 บิต ซึ่งเท่ากับ 01010010_b โดยที่รายละเอียดและค่าประจำตำแหน่งของแต่ละบิตแสดงดังรูปที่ 3.12 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-8} และที่ LSB เท่ากับ 2^{-14}

การปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง ในสมการที่ (3-11) ค่าช่วงก้าวใหม่ที่เวลา $n+1$ จะเท่ากับผลรวมของ ผลคูณของ α กับค่าช่วงก้าวเก่าที่เวลา n และผลคูณของ γ กับค่าความผิดพลาดกำลังสอง



รูปที่ 3.12 รายละเอียดบิตข้อมูลของค่า γ

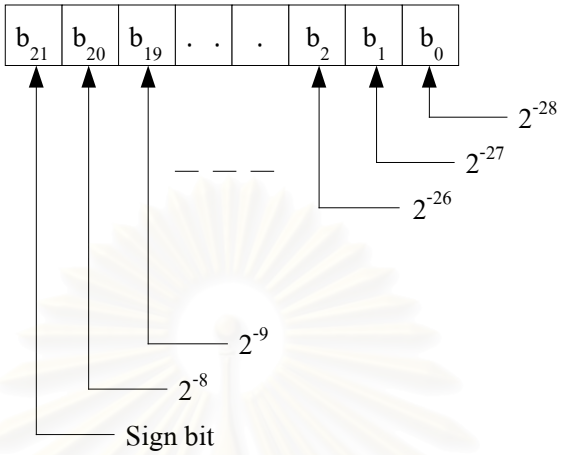
ผลคูณของ α กับค่าช่วงก้าวเก่าที่เวลา n เมื่อแทนด้วยเลขฐานสองจะมีขนาดเท่ากับ 15 บิต รายละเอียดและค่าประจำหลักในแต่ละบิตของค่าช่วงก้าวดูได้จากรูปที่ 3.6 ส่วนรายละเอียดและค่าประจำหลักในแต่ละบิตของ α เหมือนกับรูปที่ 3.11 เมื่อค่าทั้งสองคูณกันเสร็จเรียบร้อยแล้วจะได้เลขฐานสองจำนวน 15 บิต ซึ่งรายละเอียดและค่าประจำตำแหน่งในแต่ละบิตแสดงดังรูปที่ 3.13 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-11} และที่ LSB เท่ากับ 2^{-24}



รูปที่ 3.13 รายละเอียดบิตข้อมูลผลคูณของ α กับค่าช่วงก้าวเก่าที่เวลา n

ผลคูณของ γ กับค่าความผิดพลาดกำลังสอง จะต้องทำการคูณค่าความผิดพลาดกำลังสองก่อนแล้วจึงคูณด้วยค่า γ การคูณค่าความผิดพลาดกำลังสองจะได้ผลลัพธ์เป็นเลขฐานสองจำนวน 15 บิต ซึ่งจะได้ค่าประจำหลักต่าง ๆ เหมือนกับรูปที่ 3.10 จากนั้นจึงนำค่า γ ซึ่งเป็นค่าคงตัวคูณกับค่าความผิดพลาดกำลังสองจะได้ผลลัพธ์สูงสุดอยู่ในเลขฐานสองจำนวน 22 บิต ซึ่งมีราย

ละเอียดและค่าประจำตำแหน่งดังรูปที่ 3.14 ซึ่งมีค่าประจำตำแหน่งที่ MSB เท่ากับ 2^{-8} และที่ LSB เท่ากับ 2^{-28}



รูปที่ 3.14 รายละเอียดบิตข้อมูลผลคูณของ γ กับค่าความผิดพลาดกำลังสอง

ในขั้นตอนที่จะนำเอาผลคูณทั้งสองมาบวกกัน จะต้องทำการเลือกบิตข้อมูลของผลคูณระหว่าง γ และค่าความผิดพลาดกำลังสอง จากที่ผลคูณดังกล่าวมีค่าประจำตำแหน่งดังรูปที่ 3.14 จะทำการเลือกข้อมูลในตำแหน่งที่มีค่าประจำหลักเท่ากับ 2^{-11} ถึง 2^{-24} มาใช้ โดยที่จะกำหนดให้ Sign bit เป็น 0 เพื่อให้มีขนาดเท่ากับผลคูณของ α กับค่าช่วงก้าวเก่าที่เวลา n จากนั้นทำการบวกกันและทำการเลือกเอาเฉพาะ 8 บิต ทางด้าน MSB มาพิจารณากับค่าช่วงก้าวในสมการที่ (3-12) จากนั้นก็สามารถนำค่าช่วงก้าวดังกล่าวไปใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวได้

ในส่วนของการปรับค่าสัมประสิทธิ์ของวงจรกรองที่เวลา $n+1$ จะนำเอาค่าช่วงก้าวที่คำนวณได้จากวิธีค่าผิดพลาดกำลังสองมารวมกับค่าสัมประสิทธิ์ของวงจรกรองที่เวลา n ตามสมการของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ซึ่งมีรูปแบบสมการดังนี้

$$\vec{w}(n+1) = \vec{w}(n) + \mu\,[\mathrm{sgn}(\vec{x}(n))\ \mathrm{sgn}(e(n))] \qquad (3\text{-}13)$$

ในการคำนวณค่าสัมประสิทธิ์ของวงจรกรองที่ใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign จะทำการตรวจสอบ Sign bit ของค่าความผิดพลาดและสัญญาณขาเข้าของวงจรกรอง ถ้า Sign bit ของค่าทั้ง 2 มีค่าเหมือนกันก็จะทำการขยายค่าช่วงก้าวให้มีขนาด 15 บิต โดยทำการเพิ่มเลข 0 เข้าไปข้างหน้าค่าช่วงก้าวอีก 7 ตัว ส่วนในกรณีที่ค่า Sign bit มีค่าแตกต่างกัน ก็จะนำเอาค่าช่วงก้าวที่หามาได้ไปผ่านกระบวนการ 2's Complement จากนั้นก็จะเพิ่มเลข 1 เข้าไปข้างหน้าอีก 7 ตัว เมื่อทำการปรับขนาดค่าช่วงก้าวให้มีค่าประจำตำแหน่งให้เหมือนกับค่าสัมประสิทธิ์

ของวงจรกรองปรับตัวเรียบร้อยแล้ว จากนั้นทำการรวมค่าสัมประสิทธิ์ของวงจรกรองที่เวลา n กับค่าช่วงก้าวที่ขยายเป็น 15 บิต ก็จะได้ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่เวลา $n+1$

ในหัวข้อ 3.3 ที่ผ่านมาได้แสดงให้เห็นถึงรายละเอียดในการวิเคราะห์และพิจารณาการแทนค่าสัญญาณต่าง ๆ ด้วยเลขฐานสองเพื่อให้สามารถนำไปใช้ทำการคำนวณในรูปแบบ Fixed Point Arithmetic เพื่อจำลองการทำงานในรูปแบบของฮาร์ดแวร์ซึ่งจะนำไปใช้ในการสร้างจริง ในส่วนสุดท้ายของหัวข้อนี้จะแสดงให้เห็นถึงค่าต่าง ๆ ที่เป็นไปได้ที่มีขนาดสูงสุดทั้งทางด้านบวกและลบ

ตารางที่ 3.1 สรุปขนาดของสัญญาณต่าง ๆ ภายในวงจรกรองปรับตัว

สัญญาณ	ขนาดของสัญญาณ
สัญญาณขาเข้าวงจรกรองปรับตัว $x(n)$	ใช้เลขฐานสอง 15 บิต ค่าต่ำสุดเท่ากับ 100000000000000_b = -1 ค่าสูงสุดเท่ากับ 011111111111111_b = 0.99994
ค่าสัมประสิทธิ์ของวงจรกรองปรับตัว $w(n)$	ใช้เลขฐานสอง 15 บิต ค่าต่ำสุดเท่ากับ 100000000000000_b = -0.125 ค่าสูงสุดเท่ากับ 011111111111111_b = 0.12499
สัญญาณขาออกของวงจรกรองปรับตัว $y(n)$	ใช้เลขฐานสอง 15 บิต ค่าต่ำสุดเท่ากับ 100000000000000_b = -1 ค่าสูงสุดเท่ากับ 011111111111111_b = 0.99999
Desired Response $d(n)$	ใช้เลขฐานสอง 15 บิต ค่าต่ำสุดเท่ากับ 101010000101000_b = -0.68507 ค่าสูงสุดเท่ากับ 010101111011000_b = 0.68507

ตารางที่ 3.1 (ต่อ) สรุปขนาดของสัญญาณต่าง ๆ ภายในวงจรกรองปรับตัว

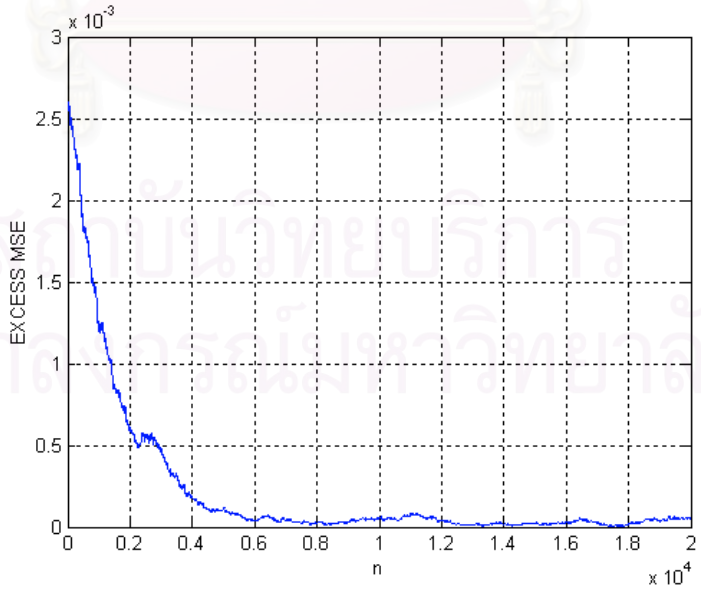
ค่าความผิดพลาด $e(n)$	ใช้เลขฐานสอง 8 บิต ค่าต่ำสุดเท่ากับ 10000000_b = -1 ค่าสูงสุดเท่ากับ 01111111_b = 0.99609
ค่าช่วงก้าว μ μ_{min} μ_{max}	ใช้เลขฐานสอง 8 บิต ค่าต่ำสุดเท่ากับ 00001101_b = 0.000099182 ค่าสูงสุดเท่ากับ 01000001_b = 0.00049591
ค่าคงตัว α	ใช้เลขฐานสอง 8 บิต มีค่าเท่ากับ 01011000_b = 0.6875
ค่าคงตัว γ	ใช้เลขฐานสอง 8 บิต มีค่าเท่ากับ 01010010_b = 0.005004882

ทั้งนี้ค่าประจำตำแหน่งของค่าสัมประสิทธิ์ต่าง ๆ ที่เป็นเลขฐานสองสามารถดูได้จากเนื้อหาที่กล่าวมาแล้วข้างต้น ค่าสัมประสิทธิ์ต่าง ๆ ในตารางที่ 3.1 จะใช้ค่าเดียวกับการคำนวณในรูปแบบ Floating Point Arithmetic มาเปลี่ยนเป็นเลขฐานสอง ซึ่งการกำหนดค่าต่าง ๆ จะพยายามให้มีค่าใกล้เคียงกับค่าสัมประสิทธิ์ที่ใช้ในการคำนวณในรูปแบบ Floating Point Arithmetic ให้มากที่สุด เพื่อให้ผลการจำลองที่ได้จากการคำนวณในรูปแบบ Fixed Point Arithmetic สามารถเปรียบเทียบกับผลการจำลองที่ใช้การคำนวณในรูปแบบ Floating Point Arithmetic ได้ หากการจำลองผลของแบบจำลองที่ใช้ค่าสัมประสิทธิ์แทนด้วยเลขฐานสองและใช้การคำนวณในรูปแบบ Fixed Point Arithmetic ได้ผลการจำลองที่มีประสิทธิภาพดีจะทำให้สามารถประเมินผลที่จะเกิดขึ้นจริง เมื่อนำเอาข้อกำหนดต่าง ๆ ที่ใช้การคำนวณในรูปแบบ Fixed Point Arithmetic ไปสร้างจริงเป็นวงจรด้วยภาษา VHDL

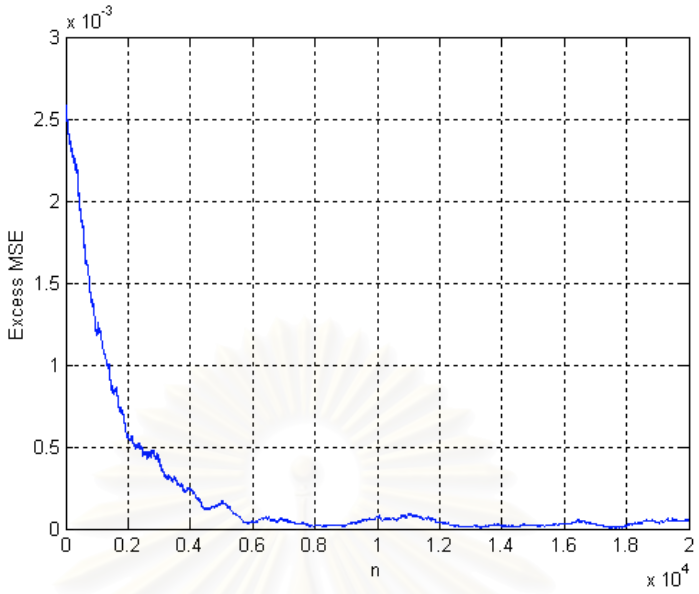
3.4 การจำลองผลโดยใช้การคำนวณแบบ Fixed Point Arithmetic

จากผลการจำลองของแบบจำลองการแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังแบบที่มีการปรับค่าสัมประสิทธิ์แบบไม่ต่อเนื่อง ที่ใช้การคำนวณในรูปแบบ Floating Point Arithmetic ให้ผลลัพธ์ของการจำลองดังที่แสดงให้เห็นแล้วในหัวข้อ 3.2 แต่เนื่องจากว่าการทำงานจริงในระบบต่าง ๆ ไม่สามารถประมวลผลด้วยตัวเลขซึ่งเป็นจำนวนจริงได้ ดังนั้นการจำลองการทำงานของระบบที่จะเกิดขึ้นจริง เมื่อนำไปสร้างเป็นฮาร์ดแวร์ที่ใช้ค่าสัมประสิทธิ์ต่าง ๆ แทนด้วยเลขฐานสอง โดยใช้การคำนวณในรูปแบบ Fixed Point Arithmetic จึงเป็นสิ่งจำเป็นที่จะแสดงให้เห็นถึงประสิทธิภาพที่จะเกิดขึ้น เมื่อนำเอาแบบจำลองต่าง ๆ ที่ได้ออกแบบไว้ไปทำการสร้างจริง

เนื่องจากการสร้างจริงจะทำการสร้างจริงวงจรกรองปรับตัวอันดับ 8 ที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง เพื่อที่จะแสดงให้เห็นถึงประสิทธิภาพที่ได้ของการจำลองผลการทำงานของแบบจำลองที่ใช้การคำนวณในรูปแบบ Fixed Point Arithmetic และสามารถเปรียบเทียบกับผลที่ได้ของแบบจำลองที่ใช้การคำนวณในรูปแบบ Floating Point Arithmetic ได้ ดังนั้นในหัวข้อนี้จะทำการจำลองผลที่ได้ของวงจรกรองปรับตัวอันดับ 8 ที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอนเนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยใช้การคำนวณทั้ง 2 รูปแบบ



รูปที่ 3.15 ผลการจำลองของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง ที่ใช้การคำนวณแบบ Floating Point Arithmetic



รูปที่ 3.16 ผลการจำลองของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และ การปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง ที่ใช้การคำนวณแบบ Fixed Point Arithmetic

จากผลที่ได้จากการจำลองในรูปที่ 3.15 และ 3.16 จะเห็นว่าผลที่ได้จากการจำลองที่ใช้การคำนวณในรูปแบบ Fixed Point Arithmetic ใกล้เคียงกับผลที่ได้จากการจำลองที่ใช้การคำนวณในรูปแบบ Floating Point Arithmetic ซึ่งผลการจำลองที่ได้ในรูปที่ 3.16 จะเป็นเครื่องยืนยันผลการทำงานที่จะเกิดขึ้นจริง เมื่อนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่ใช้การปรับช่วงก้าวแบบพลวัตแบบค่าผิดพลาดกำลังสองไปทำการสร้างจริงในรูปแบบฮาร์ดแวร์ ซึ่งการสร้างจริงจะนำเสนอต่อไปในบทที่ 4

บทที่ 4

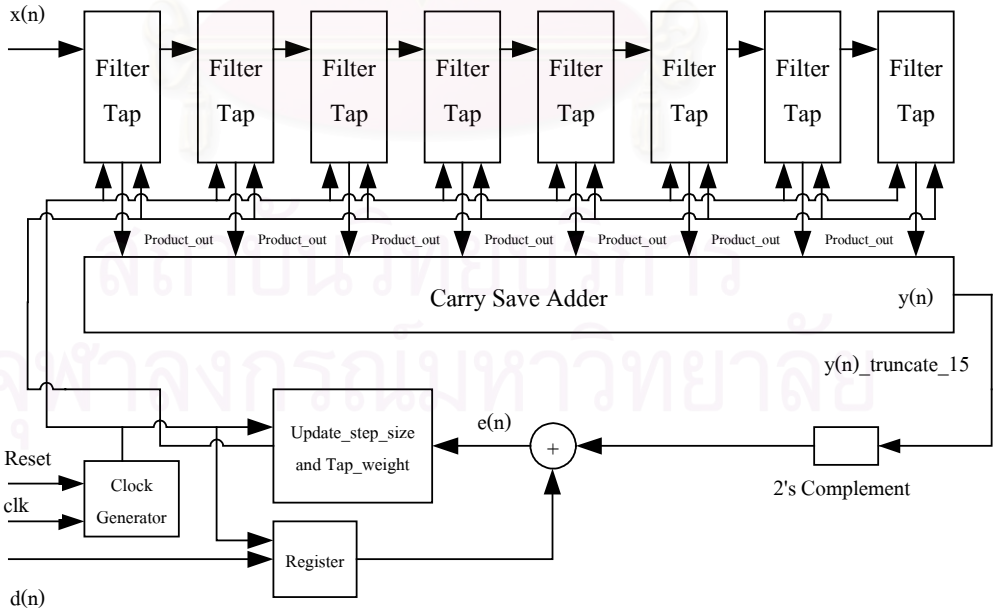
การสร้างจริงวงจรกรองปรับตัว

การสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง จะใช้ผลที่ได้จากการจำลองเชิงฮาร์ดแวร์ที่ใช้การคำนวณในรูปแบบ Fixed Point Arithmetic ในบทที่ 3 เป็นพื้นฐานในการออกแบบส่วนประกอบต่าง ๆ ที่จะใช้ในการสร้างจริงบน FPGA ของบริษัท Xilinx รุ่น Spartan2 xc2s200-5pq208

เนื้อหาในบทนี้จะกล่าวถึงโครงสร้างทางสถาปัตยกรรมของวงจรปรับตัว ขั้นตอนการทำงานของวงจรกรองปรับตัว จากนั้นจะกล่าวถึงส่วนประกอบต่าง ๆ ที่นำมาใช้ในการสร้างจริงวงจรกรองปรับตัวรวมถึงระบบการทำงานของส่วนต่าง ๆ ต่อจากนั้นจะแสดงผลที่ได้จากการทำงานของวงจรกรองปรับตัวที่ผ่านการสังเคราะห์ทั้งในระดับ Behavioral และระดับ Logic

4.1 โครงสร้างทางสถาปัตยกรรมของวงจรกรองปรับตัว

โครงสร้างทางสถาปัตยกรรมของวงจรกรองปรับตัวที่จะใช้ในการสร้างจริงมีโครงสร้างดังแสดงดังรูปที่ 4.1



รูปที่ 4.1 โครงสร้างทางสถาปัตยกรรมของวงจรกรองปรับตัวที่ใช้ในการสร้างจริง

จากรูปที่ 4.1 แสดงโครงสร้างทางสถาปัตยกรรมของวงจรกรองปรับตัวที่ใช้ในการสร้างจริง สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งประกอบด้วยส่วนต่าง ๆ ดังนี้ คือ Filter Tap วงจรบวก (Carry Save Adder, Carry Propagate Adder) วงจร 2's Complement วงจรกำเนิดสัญญาณนาฬิกา (Clock Generator) Register วงจรปรับค่าช่วงก้าวและสัมประสิทธิ์ของวงจรกรอง

4.2 ขั้นตอนการทำงานของระบบ

การสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียง จะทำการสร้างจริงวงจรกรองอันดับ 8 ทั้งนี้เพื่อให้สามารถทำงานได้จริงบนทรัพยากรต่าง ๆ ที่มีจำกัด การสร้างจริงวงจรกรองปรับตัวจะทำการสร้างจริงบน FPGA ของบริษัท Xilinx รุ่น Spartan2 xc2s200-5pq208 โดยที่การสร้างจริงจะทำตามแบบจำลองที่ใช้การประมวลผลในรูปแบบ Fixed Point Arithmetic

ขั้นตอนการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง เริ่มต้นจากการรับข้อมูลเข้ามาในวงจรกรองปรับตัว ทำการประมวลในส่วนของวงจรกรอง หาค่าความผิดพลาด ทำการปรับค่าช่วงก้าวและทำการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว ซึ่งการทำงานในขั้นตอนต่าง ๆ จะแสดงให้เห็นต่อไป

เริ่มต้นจากการคำนวณในแต่ละ Tap ของวงจรกรองปรับตัว เพื่อคำนวณหาผลลัพธ์ที่ได้จากกระบวนการกรองที่เวลา n

$$y(n) = w_1(n)x(n-1) + w_2x(n-2) + \ldots + w_8x(n-8) \qquad (4\text{-}1)$$

โดยที่

$y(n)$ คือ สัญญาณขาออกของวงจรกรองที่เวลา n

$d(n)$ คือ Desired Response ที่เวลา n (เป็นผลรวมจากสัญญาณเสียงขาเข้ากับสัญญาณเสียงที่ผ่านเส้นทางป้อนกลับ)

$e(n)$ คือ ค่าความผิดพลาดที่เวลา n

$w_i(n)$ คือ สัมประสิทธิ์ของวงจรกรอง Tap ที่ i ที่เวลา n $(1 \le i \le 8)$

$x(n-i)$ คือ สัญญาณขาเข้าของวงจรกรอง ณ Tap ที่ i ที่เวลา n $(1 \le i \le 8)$

$\mu(n)$ คือ ค่าช่วงก้าว (Step Size) ที่เวลา n

$y(n)\_truncate\_15$ คือ สัญญาณขาออกของวงจรกรองปรับตัว $y(n)$ ที่มีค่าเท่ากับ 15 บิต ทางด้าน MSB ของ $y(n)$

$y(n)\_truncate\_15\_com$ คือ สัญญาณ $y(n)\_truncate\_15$ ที่ผ่านกระบวนการ 2's Complement

จากนั้นนำเอาค่าสัญญาณขาออกของวงจรกรอง $y(n)$ ที่มีขนาด 32 บิต ทำการเลือกบิตข้อมูลทางด้าน MSB 15 บิต เก็บไว้ใน $y(n)\_truncate\_15$ จากนั้นนำผลลัพธ์ที่ได้ไปทำกระบวนการ 2's Complement และเก็บค่าไว้ใน $y(n)\_truncate\_15\_com$ หลังจากนั้นนำค่าที่ได้ไปรวมกับค่า Desired Response $d(n)$ เพื่อหาค่าความผิดพลาด $e(n)$ ต่อจากนั้นนำค่าความผิดพลาด $e(n)$ และค่าช่วงก้าว $\mu(n)$ เพื่อนำไปคำนวณหาค่าช่วงก้าว $\mu(n+1)$ ที่จะใช้ในการคำนวณที่เวลา $n+1$ โดยใช้สมการที่ (4-2) และสมการที่ (4-3) ขั้นตอนสุดท้ายนำค่าช่วงก้าว $\mu(n)$ ค่า Sign bit ของค่าความผิดพลาด $e(n)$ Sign bit ของสัญญาณขาเข้า $x(n-i)$ ของวงจรกรอง ณ Tap ที่ i ที่เวลา n และค่าสัมประสิทธิ์ของวงจรกรอง $w_i(n)$ เพื่อนำไปใช้คำนวณหาค่าสัมประสิทธิ์ของวงจรกรอง $w_i(n+1)$ ที่เวลา $n+1$ โดยใช้สมการที่ (4-4) และสมการที่ (4-5)

สมการของวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง (Squared Error)

$$\mu'(n+1) = \alpha\,\mu(n) + \gamma\, e^2(n) \tag{4-2}$$

โดยที่

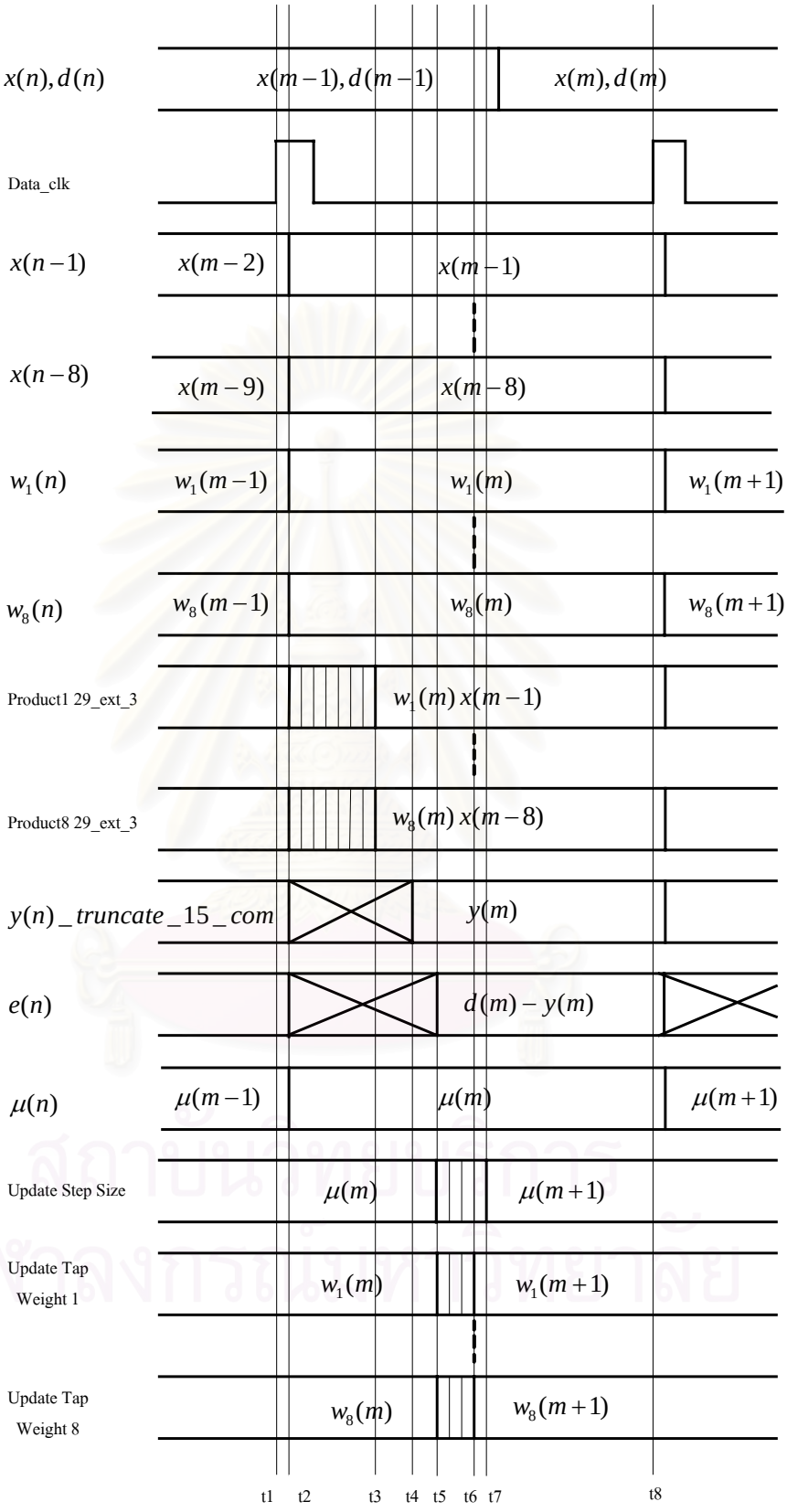
$$\mu(n+1) = \begin{cases} \mu_{\max}, & \mu'(n+1) > \mu_{\max} \\ \mu_{\min}, & \mu'(n+1) < \mu_{\min} \\ \mu'(n+1), & Others \end{cases} \tag{4-3}$$

สมการของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ณ Tap ที่ i ที่เวลา n

$$w_i(n+1) = w_i(n) + \mu(n)\,[\mathrm{sgn}(x(n-i))\ \mathrm{sgn}(e(n))] \tag{4-4}$$

โดยที่

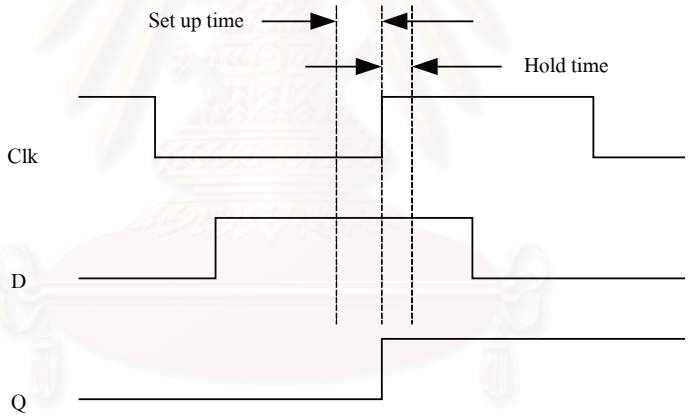
$$\mathrm{sgn}(x) = \begin{cases} 1, x \geq 0 \\ -1, x < 0 \end{cases} \tag{4-5}$$



รูปที่ 4.2 Timing Diagram การทำงานของวงจรกรองปรับตัว

จากขั้นตอนการทำงานของวงจรกรองปรับตัวที่กล่าวมาแล้วข้างต้นสามารถแสดงขั้นตอนในแต่ละรอบของการทำงานได้ด้วย Timing Diagram ดังรูปที่ 4.2 โดยจะพิจารณาการทำงานของระบบที่เวลา $n = m$

สมมติให้ระบบเริ่มต้นการทำงานที่เวลา t1 ซึ่งสัญญาณ data_clk มีค่าเป็น 1 ที่เวลาดังกล่าว ข้อมูลต่าง ๆ ที่ใช้ในระบบ เช่น $x(m-1)$, $d(m-1)$, $w_1(m)$ และ $\mu(n)$ เป็นต้น จะถูก Load เข้าไปเก็บไว้ใน Register ซึ่งจากรูปที่ 4.2 แสดงให้เห็นว่าจะเสร็จสิ้นกระบวนการ Load ข้อมูลที่เวลา t2 ซึ่งช่วงเวลาระหว่าง t1 ถึง t2 จะเรียกว่า Clock to Q time คือ ระยะเวลาที่เริ่มเกิดการเปลี่ยนแปลงของสัญญาณ Clock จาก 0 เป็น 1 (สมมติให้ Register ทำงานที่ขอบขาขึ้น) เพื่อทำการรับข้อมูลทางด้าน Input จนกระทั่งข้อมูลดังกล่าวปรากฏทางด้าน Output ของ Register ทั้งนี้ข้อมูลที่จะป้อนเข้าสู่ Register จะต้องคำนึงถึงค่า Set up time และ Hold time [18] ของ Register ด้วย ข้อมูลต่าง ๆ ที่จะป้อนเข้าสู่ Register จะต้องมีเสถียรภาพ (Stable) เป็นระยะเวลานานกว่าค่า Set up time และค่า Hold time รวมกัน เพื่อให้ข้อมูลที่รับเข้าไปใน Register ถูกต้องดังแสดงในรูปที่ 4.3



รูปที่ 4.3 Set up time และ Hold time

จากรูปที่ 4.2 เมื่อทำการ Load ข้อมูลต่าง ๆ เข้าสู่ระบบเรียบร้อยแล้ว ขั้นตอนต่อไปจะเป็นขั้นตอนการคูณค่าสัมประสิทธิ์ของวงจรกรองที่ Tap ต่าง ๆ กับสัญญาณขาเข้าของวงจรกรองที่เวลา m เท่ากับ $w_i(m)x(m-i)$ ซึ่งจะเป็นผลลัพธ์ที่ได้จากการคูณของวงจรกรองแต่ละ Tap จากนั้นนำผลคูณที่ได้ไปขยายจำนวนบิตเพิ่มทางด้าน MSB ของผลคูณ ให้มีค่าเหมือนกับ Sign bit ของผลคูณแต่ละ Tap กระบวนการทำงานของขั้นตอนเหล่านี้ใช้เวลาตั้งแต่ t2 ถึง t3 ต่อจากนั้นจะเป็นขั้นตอนของการรวมค่าผลคูณของวงจรกรองและทำการเลือกจำนวนบิตที่จะนำมาใช้และนำข้อมูลที่เลือกไปผ่านกระบวนการ 2's Complement เพื่อให้ได้ค่า $y(m)$ ซึ่งเป็นผลลัพธ์ของวงจรกรองที่ผ่านทั้งกระบวนการ Truncate 15 บิต และ 2's Complement เรียบร้อยแล้ว ซึ่งกระบวนการที่ผ่านมาใช้เวลา

ในการประมวลผลตั้งแต่เวลา t3 ถึงเวลา t4 หลังจากนั้นหาค่าความผิดพลาด ซึ่งจะได้ค่าความผิดพลาด $d(m)-y(m)$ ที่เวลา t5 จากนั้นนำค่าความผิดพลาดที่ได้ไปคำนวณหาค่าช่วงก้าวเพื่อที่ใช้ในรอบต่อไปของการประมวลผล ในขณะเดียวกันก็ทำการคำนวณหาค่าสัมประสิทธิ์ที่จะนำไปใช้ในการประมวลในรอบต่อไปเช่นเดียวกัน ในที่นี้กำหนดให้การปรับค่าสัมประสิทธิ์ของวงจรกรองปรับใช้เวลาในการคำนวณตั้งแต่เวลา t5 ถึง t6 ส่วนการปรับช่วงก้าวใช้เวลาตั้งแต่ t5 ถึง t7 หลังจากเวลา t6 และ t7 จะได้ค่าสัมประสิทธิ์และค่าช่วงก้าวของวงจรกรองใหม่ตามลำดับ พร้อมที่จะนำค่าดังกล่าวไปใช้ในการประมวลผลรอบต่อไป จากรูปที่ 4.2 แต่ละรอบของการประมวลผลใช้เวลาตั้งแต่ t1 ถึง t7 ซึ่งจะเรียกเส้นทางที่ใช้เวลาในการประมวลผลตั้งแต่เริ่มต้นจนสิ้นสุดกระบวนการว่า Critical Path ดังนั้นในแต่ละรอบของการสุ่มตัวอย่าง data_clk ของระบบต้องมีค่าเท่ากับหรือมากกว่า ค่าของ Critical Path เพื่อที่จะทำให้ระบบที่ออกแบบยังคงสามารถทำงานได้ถูกต้อง

4.3 การสร้างจริงส่วนประกอบต่าง ๆ ตามสถาปัตยกรรมของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

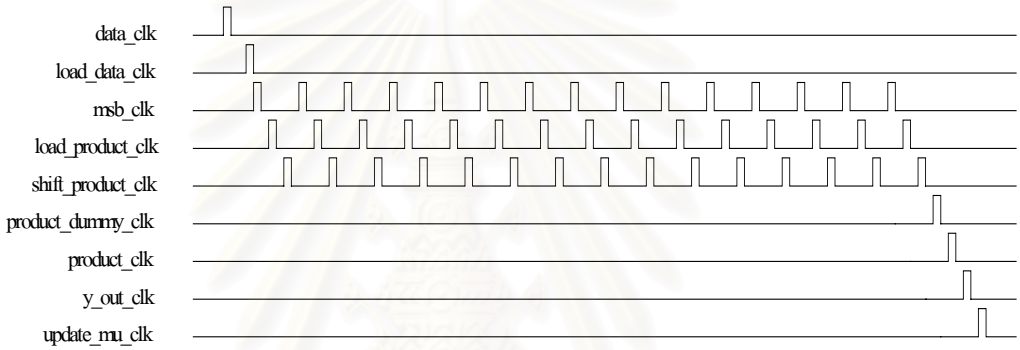
จากโครงสร้างทางสถาปัตยกรรมในรูปที่ 4.1 การสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง สามารถแบ่งได้เป็น 2 ส่วนหลัก ๆ คือ ส่วนของวงจรกรองและส่วนปรับค่าสัมประสิทธิ์ของวงจรกรอง

วงจรกรองประกอบด้วยส่วนประกอบที่สำคัญ คือ Filter Tap (ภายในประกอบด้วย Register และวงจรคูณแบบ Booth Encode) วงจรบวก (Carry Save Adder, Carry Propagate Adder) Register และวงจร 2's Complement ส่วนประกอบของวงจรในการปรับค่าช่วงก้าวและปรับค่าสัมประสิทธิ์ของวงจรกรองประกอบด้วย วงจรคูณแบบ Array วงจรบวกแบบ Carry Propagate Adder และวงจร 2's Complement นอกจากนี้ยังมีส่วนประกอบที่สำคัญที่ใช้ในการควบคุมการทำงานของระบบ คือ วงจรกำเนิดสัญญาณนาฬิกา (Clock Generator)

ในหัวข้อนี้จะกล่าวถึงส่วนประกอบต่าง ๆ ที่จะนำมาใช้ในการสร้างจริงวงจรกรองปรับตัว ซึ่งจะทำการออกแบบวงจรด้วยภาษา VHDL โดยจะอธิบายหลักการในการทำงานของส่วนประกอบต่าง ๆ หลังจากนั้นจะแสดงผลการทำงานของวงจรต่าง ๆ ที่ได้ทำการออกแบบด้วยภาษา VHDL ส่วนประกอบต่าง ๆ จะต้องผ่านการทดสอบว่าสามารถทำงานได้จริงบน FPGA ก่อนที่จะนำเอาส่วนประกอบต่าง ๆ มาประกอบรวมกันเพื่อที่จะทำการสร้างจริงเป็นวงจรกรองปรับตัว สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังให้สามารถทำงานบน FPGA ได้จริง

4.3.1 ระบบ Clock ที่ใช้ควบคุมการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

การทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ที่จะทำการสร้างจริงจะต้องมีการทำงานที่เป็นระบบและมีเวลาในการทำงานที่แน่นอนเที่ยงตรงในแต่ละขั้นตอน เนื่องจากผลการทำงานในแต่ละขั้นตอนจะต้องสอดคล้องกัน ดังนั้นระบบ Clock ที่จะใช้ควบคุมระบบจึงมีความสำคัญมาก ในหัวข้อนี้จะแสดงให้เห็นถึงระบบของสัญญาณ Clock ที่นำมาใช้งานจริงซึ่งถูกสร้างขึ้นจากวงจรกำเนิดสัญญาณนาฬิกา (Clock Generator) ที่ออกแบบด้วยภาษา VHDL



รูปที่ 4.4 ระบบของสัญญาณ Clock ที่ใช้ในการสร้างจริงวงจรกรองปรับตัว

จากรูปที่ 4.4 แสดงสัญญาณ Clock ต่าง ๆ ที่จะทำการสร้างขึ้นมาเพื่อใช้ควบคุมการทำงานของวงจรกรองปรับตัว โดยที่สามารถแบ่งสัญญาณ Clock ดังกล่าวได้เป็น 2 กลุ่ม คือ สัญญาณ Clock ที่ใช้งานในระบบของวงจรคูณ Booth Encode ซึ่งประกอบสัญญาณ Clock ต่าง ๆ ดังนี้ คือ load_data_clk, msb_clk, load_product_clk, shift_product_clk, product_dummy_ck และ product_clk

ส่วนสัญญาณ Clock ที่เหลือเป็นสัญญาณ Clock ที่ใช้ในการควบคุมการทำงานของระบบวงจรกรองปรับตัว ซึ่งใช้สำหรับควบคุมการรับส่งข้อมูล การปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสองและการปรับค่าสัมประสิทธิ์ของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ซึ่งประกอบด้วยสัญญาณ Clock ต่าง ๆ ดังนี้ data_clk, y_out_clk และ update_mu_clk

ในการทำงานของระบบที่ใช้งานจริง สัญญาณ Clock ที่ป้อนให้กับระบบควรจะมีเพียงสัญญาณเดียว เพื่อความมีเสถียรภาพของวงจรที่ออกแบบเมื่อนำไปใช้งานจริง เพราะถ้าหากระบบที่ออกแบบต้องใช้สัญญาณ Clock จากภายนอกหลายสัญญาณในการควบคุมการทำงาน เมื่อเวลาผ่าน

ไปหากความถี่ของสัญญาณใดสัญญาณหนึ่งเกิดการเปลี่ยนแปลง จะทำให้คาบเวลาในการทำงานของระบบบางส่วนเกิดการเปลี่ยนแปลงตามไปด้วย ซึ่งอาจส่งผลให้ระบบมีการทำงานที่ผิดพลาดได้ ดังนั้นในการออกแบบระบบของสัญญาณ Clock ที่จะนำไปใช้งานจริงด้วยภาษา VHDL จะใช้สัญญาณ Clock จากภายนอกเพียงสัญญาณเดียว โดยการนำวงจรนับ (Counter) มาใช้นับจำนวนของสัญญาณ Clock ที่ป้อนเข้าสู่ระบบ ซึ่งค่าที่นับได้จะเป็นตัวกำหนดการทำงานของขั้นตอนต่าง ๆ เช่น ค่าที่นับได้เท่ากับหนึ่งจะทำการรับข้อมูลจากภายนอกเข้าสู่ระบบ ค่าที่นับได้เท่ากับสองจะทำการประมวลผลตามขั้นตอนต่าง ๆในระบบ และค่าที่นับได้เท่ากับสามจะนำผลลัพธ์ที่ได้ออกสู่ส่วนแสดงผล เป็นต้น

การสร้างจริงสัญญาณ Clock ต่าง ๆ ที่กล่าวมาแล้วข้างต้น เพื่อใช้ในการควบคุมการทำงานของวงจรกรองปรับตัว อาศัยสัญญาณ Clock ที่ป้อนเข้าสู่ระบบซึ่งเป็นสัญญาณ Clock ที่มีความถี่เท่ากับ 10 MHz ซึ่งเป็นสัญญาณ Clock ที่มีคาบเวลาเท่ากับ 100 ns

วงจรกรองปรับตัวที่จะนำมาใช้ในการสร้างจริงออกแบบให้แต่ละรอบของการสุ่มตัวอย่างใช้เวลาเท่ากับ 12000 ns ในช่วงเวลาดังกล่าวจะทำการประมวลผลและส่งข้อมูลออกทางพอร์ตสัญญาณขาออก รวมถึงการปรับค่าสัมประสิทธิ์ของวงจรกรอง และการปรับค่าช่วงก้าวเพื่อที่จะนำไปใช้ในการประมวลผลในรอบต่อไป จะเห็นว่าในแต่ละรอบของการสุ่มตัวอย่างใช้เวลาค่อนข้างมาก ซึ่งความถี่ในการสุ่มตัวอย่างเท่ากับ 1/12000 ns = 83.33 KHz การสร้างจริงวงจรกรองปรับตัวจะนำมาใช้กับสัญญาณเสียงซึ่งมีความถี่อยู่ที่ประมาณ 20 KHz จะเห็นได้ว่าความถี่ของการสุ่มตัวอย่างมีค่าสูงกว่าความถี่ของสัญญาณเสียง ดังนั้นวงจรที่ทำการออกแบบสามารถที่จะนำไปใช้งานกับสัญญาณเสียงได้

วงจรนับจะทำการนับขอบขาขึ้นของสัญญาณ Clock ที่ป้อนเข้าสู่ระบบ ซึ่งระบบของวงจรกรองที่จะทำการสร้างจริงจะเริ่มต้นนับตั้งแต่ 1 ถึง 120 และจะทำการ Reset ค่าของวงจรนับ เมื่อถึงค่าสูงสุดที่ตั้งไว้ คือ 120 และจะเริ่มต้นนับค่าดังกล่าวใหม่ ซึ่งขั้นตอนการทำงานของวงจรนับจะทำงานดังกล่าวต่อไปเรื่อย ๆ เพื่อที่จะนำค่าที่นับได้ไปใช้ในการสร้างสัญญาณ Clock ต่าง ๆ ที่จะนำไปใช้ในระบบของวงจรกรองปรับตัว ค่าที่นับได้ของวงจรนับที่นำมาใช้ในการสร้างสัญญาณ Clock ต่าง ๆ มีรายละเอียดดังนี้ คือ ค่าที่นับได้เท่ากับ 1 นำไปใช้สร้างสัญญาณ data_clk ค่าที่นับได้เท่ากับ 5, 11, 17, 23, 29, 35, 41, 47, 53, 59, 65, 71, 77, 83 และ 89 นำไปใช้สร้างสัญญาณ msb_clk ค่าที่นับได้เท่ากับ 120 นำไปใช้ในการ Reset ระบบของวงจรนับ

สัญญาณ Clock ที่เหลือจะสร้างจากสัญญาณ data_clk และ msb_clk สัญญาณ Clock ที่สร้างจากสัญญาณ data_clk เป็นสัญญาณที่เกิดขึ้นเพียงครั้งเดียวในแต่ละรอบของการประมวลผล ซึ่งประกอบด้วยสัญญาณดังต่อไปนี้ คือ สัญญาณ load_data_clk สัญญาณ product_dummy_clk สัญญาณ product_clk สัญญาณ y_out_clk และสัญญาณ update_mu_clk

สัญญาณ Clock ทั้ง 5 สัญญาณ สร้างขึ้นโดยการนำเอาสัญญาณ data_clk ต่อเข้ากับ Register 3 ตัว, 94 ตัว, 96 ตัว, 98 ตัว และ 100 ตัว ตามลำดับ เพื่อใช้หน่วงเวลาในการเกิดขึ้นของสัญญาณดังกล่าว ส่วนที่เหลือเป็นสัญญาณ Clock ที่สร้างจากสัญญาณ msb_clk เป็นสัญญาณที่มีลัญษณะเหมือนกับสัญญาณ msb_clk ซึ่งประกอบด้วยสัญญาณดังต่อไปนี้ คือ สัญญาณ load_product_clk และสัญญาณ shift_product_clk

การสร้างสัญญาณทั้งสองทำโดยนำเอาสัญญาณ msb_clk มาต่อกับ Register 2 ตัว และ 4 ตัว ตามลำดับเพื่อใช้หน่วงเวลาในการเกิดของสัญญาณดังกล่าวเช่นเดียวกัน

ที่ผ่านมาได้กล่าวถึงวิธีการในการสร้างสัญญาณ Clock ต่าง ๆ ที่จะนำมาใช้ในการควบคุมระบบการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ที่จะทำการสร้างจริงบน FPGA ต่อไปจะอธิบายถึงหน้าที่ในการทำงานของสัญญาณ Clock ต่าง ๆ

สัญญาณ data_clk ใช้สำหรับ Load ข้อมูลต่าง ๆ เข้าสู่ระบบของวงจรกรองปรับตัว เช่น สัญญาณขาเข้าของวงจรกรอง สัญญาณที่ผ่านการป้อนกลับรวมกับสัญญาณเสียงที่มีกำลังต่ำ (เสียงเบามาก) ค่าช่วงก้าวและค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่ผ่านการ Update ในรอบที่ผ่านมาของการประมวลผล

สัญญาณ load_data_clk ใช้สำหรับ Load ค่าสัญญาณขาเข้าของวงจรกรองที่เวลา n และค่าสัมประสิทธิ์ของวงจรกรองที่เวลา n เข้าไปในวงจรคูณ Booth Encode

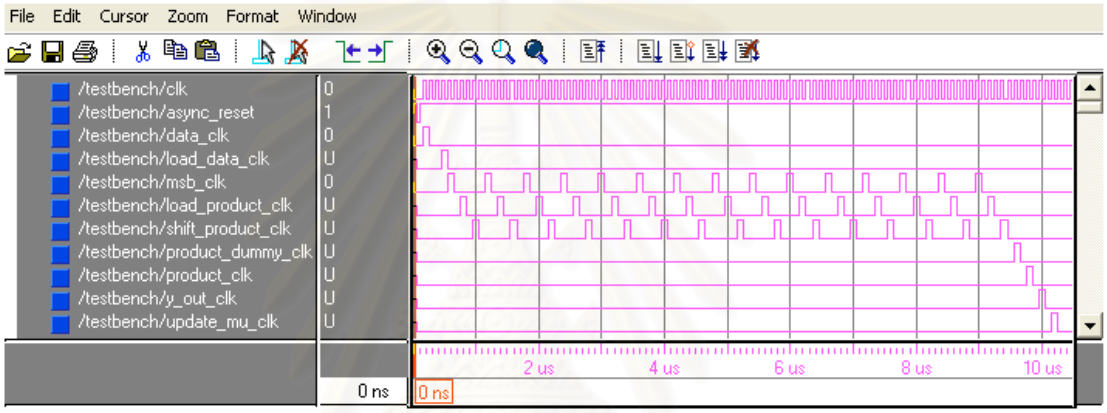
สัญญาณ msb_clk, load_product_clk และ shift_product_clk เป็นสัญญาณที่ใช้สำหรับกำหนดขั้นตอนในการทำงานของวงจร Booth Encode

สัญญาณ product_dummy_clk ใช้สำหรับพิจารณาจำนวนบิตของผลคูณ ที่เกิดจากวงจรคูณ Booth Encode

สัญญาณ product_clk ใช้สำหรับนำผลคูณจากวงจรคูณ Booth Encode ที่เสร็จเรียบร้อยแล้วออกสู่สัญญาณขาออกของวงจรคูณ Booth Encode

สัญญาณ y_out_clk ใช้สำหรับกำหนดช่วงเวลาในการนำข้อมูลที่ประมวลผลได้ของวงจรกรองปรับตัวออกสู่พอร์ตสัญญาณขาออก

สัญญาณ update_mu_clk ใช้สำหรับกำหนดช่วงเวลาในการปรับค่าสัมประสิทธิ์และค่าช่วงก้าวของวงจรกรอง โดยจะทำการปรับค่าดังกล่าวที่ขอบขาขึ้นและขอบขาลงของสัญญาณตามลำดับ

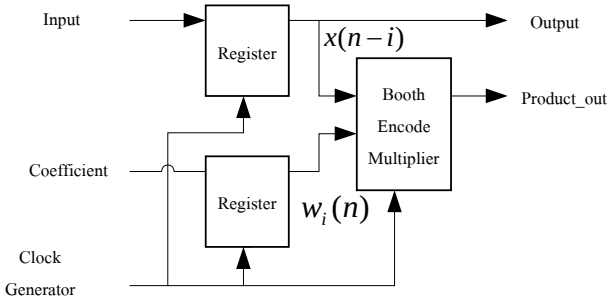


รูปที่ 4.5 สัญญาณ Clock ต่าง ๆ ที่ได้จากวงจรกำเนิดสัญญาณ Clock ที่ออกแบบด้วยภาษา VHDL

รูปที่ 4.5 แสดงสัญญาณ Clock ต่าง ๆ ที่สร้างขึ้นจากวงจรกำเนิดสัญญาณ Clock ซึ่งจะนำไปใช้ในการสร้างจริงบน FPGA โดยใช้ภาษา VHDL ในการออกแบบวงจร สัญญาณ Clock ต่างๆ ที่สร้างขึ้นมาจะนำไปใช้ในการควบคุมการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังดังที่กล่าวมาแล้วข้างต้น

4.3.2 การสร้างจริง Filter Tap

จากโครงสร้างของสถาปัตยกรรมของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังในรูปที่ 4.1 Filter Tap เป็นส่วนประกอบที่สำคัญของวงจรกรอง เพราะเป็นส่วนที่ใช้ในการประมวลผลของวงจรกรอง โดยจะทำการคูณค่าสัมประสิทธิ์ของวงจรกรองที่ Tap ต่าง ๆ กับค่าของสัญญาณขาเข้าที่ Tap ดังกล่าวที่เวลา n ใด ๆ Filter Tap แต่ละตัวประกอบด้วย Register 15 บิต แบบขนานและวงจรคูณ Booth Encode ดังแสดงในรูปที่ 4.6 ซึ่งต่อไปจะอธิบายขั้นตอนการออกแบบและหลักการทำงาน เพื่อที่จะทำการสร้างจริงส่วนประกอบทั้งสองด้วยภาษา VHDL



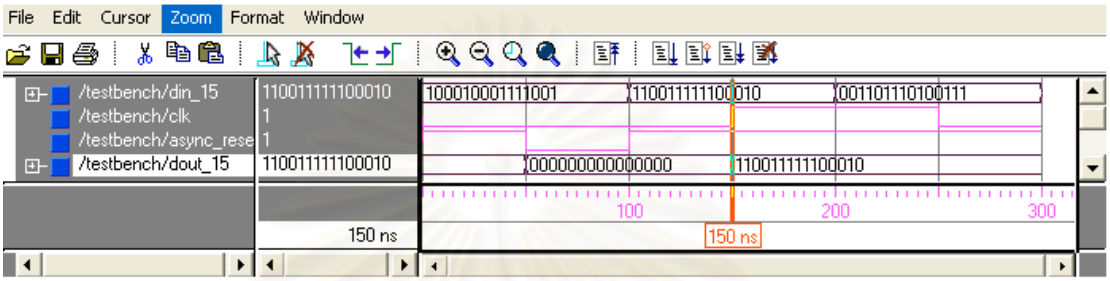
รูปที่ 4.6 รายละเอียดทางโครงสร้างของ Filter Tap

รูปที่ 4.6 แสดงรายละเอียดทางโครงสร้างของ Filter Tap ซึ่งมีส่วนประกอบที่สำคัญดังนี้ คือ Register ขนาด 15 บิต แบบขนาน 2 ตัว และวงจรคูณ Booth Encode การทำงานของ Filter Tap แต่ละตัว (ในที่นี้จะพิจารณา Tap ที่ i) จะทำการรับค่าสัญญาณขาเข้าและค่าสัมประสิทธิ์ ที่เวลา n ใด ๆ เข้ามาเก็บไว้ใน Tap ที่ i (สัญญาณ Input อาจจะเป็นสัญญาณที่เข้ามาสู่ระบบเป็นครั้งแรก หาก Filter Tap ดังกล่าวเป็น Tap ที่ 1 หรือเป็นสัญญาณที่ถูกส่งมาจาก Tap ก่อนหน้านี้) เมื่อ Register ได้รับค่าสัญญาณขาเข้าและค่าสัมประสิทธิ์ของวงจรกรองเรียบร้อยแล้ว ข้อมูลทั้งสองจะถูกส่งต่อเข้าไปในวงจรคูณ Booth Encode เพื่อคำนวณผลลัพธ์ของ Filter Tap กระบวนการทำงานทั้งหมดจะถูกกำหนดช่วงเวลาในการทำงานด้วยสัญญาณ Clock จากตัวกำเนิดสัญญาณ Clock

Register เป็นส่วนประกอบของวงจรกรองที่ใช้เก็บข้อมูล เพื่อให้คงค่าของสัญญาณต่าง ๆ ไว้จนกว่าจะมีการเปลี่ยนแปลงของสัญญาณ Clock เข้ามาเป็นตัวเปิด Gate ของ Register เพื่อรับข้อมูลใหม่ที่เข้ามาทางด้าน Input ของ Register ให้ไปปรากฏทางด้าน Output ของ Register ซึ่งช่วงเวลาในการทำงานดังกล่าวเรียกว่า Clock to Q time ส่วนการเปลี่ยนแปลงของสัญญาณ Clock ที่จะนำมาใช้ในการเปิด Gate ของ Register สามารถใช้ได้ทั้งขอบขาขึ้นและขอบขาลงของสัญญาณ Clock นอกจากสัญญาณ Clock ที่ใช้ควบคุมการรับข้อมูลของ Register แล้วในการใช้งานจะต้องมีการ Reset ระบบ ซึ่งจะต้องใช้สัญญาณที่ทำหน้าที่ Reset ระบบโดยเฉพาะ ในระบบวงจรกรองปรับตัวที่ทำการออกแบบจะเรียกสัญญาณดังกล่าวว่า ASYNC_RESET การสร้างจริง Register 15 บิต แบบขนานโดยการออกแบบวงจรด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.7 ซึ่ง Register ทั้งหมดที่จะทำการสร้างจริงด้วยภาษา VHDL ที่จะนำมาใช้งานในวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังจะมีคุณลักษณะเหมือนกับ D Flip Flop ทั่วไป

จากรูปที่ 4.7 สามารถอธิบายหลักการทำงานของ Register ที่ออกแบบวงจรด้วยภาษา VHDL ได้ดังนี้ เมื่อสัญญาณ ASYNC_RESET มีการเปลี่ยนแปลงจาก 1 เป็น 0 จะเกิดการ Reset

ระบบของ Register ทำให้ค่าที่ Output ของ Register ถูก Reset ให้เป็น 0 ทั้ง 15 บิต และหลังจากนั้นเมื่อสัญญาณ ASYNC_RESET เปลี่ยนค่าจาก 0 เป็น 1 เรียบร้อยแล้ว เมื่อมีสัญญาณ Clock ที่ขอบขาขึ้น(ให้วงจรทำงานที่ขอบขาขึ้นของสัญญาณ Clock) เข้ามา Register ก็จะรับค่าของสัญญาณทางด้าน Input ไปปรากฏออกทางด้าน Output ของ Register ซึ่งการทำงานทุกครั้งต้องเริ่มต้นด้วยการ Reset การทำงานของระบบ เพื่อให้สามารถทำงานได้ถูกต้อง



รูปที่ 4.7 ผลการทำงานของ Register ที่ออกแบบวงจรด้วยภาษา VHDL

วงจรคูณ Booth Encode เป็นวงจรคูณที่จะนำมาใช้ในการคูณค่าสัญญาณขาเข้าของ Filter Tap กับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว สาเหตุของการเลือกใช้วงจรคูณแบบ Booth Encode เนื่องจากการสร้างจริงวงจรกรองปรับตัวจะทำการสร้างจริงบน FPGA ซึ่งมีข้อจำกัดทางด้านทรัพยากรดังที่ได้กล่าวมาแล้ว แต่สำหรับวงจรคูณแบบ Booth Encode จะใช้จำนวน Gate ในการสร้างจริงที่น้อยกว่าวงจรคูณที่ใช้กันทั่วไป เช่น วงจรคูณแบบ Array เป็นต้น ดังนั้นการสร้างจริงวงจรคูณในส่วนของ Filter Tap ที่จำเป็นต้องใช้วงจรคูณหลายตัว จึงเลือกที่จะใช้วงจรคูณแบบ Booth Encode

ขั้นตอนวิธีการคูณของวงจรคูณ Booth Encode (Booth Algorithm) ที่ใช้ในการคูณเลขฐานสองซึ่งประกอบด้วย ตัวตั้ง (Multiplicand) และตัวคูณ (Multiplier) จำนวน n บิต มีขั้นตอนดังต่อไปนี้

ขั้นตอนที่ 1 กำหนดค่าเริ่มต้นให้ผลคูณ (Product) มีจำนวนบิตเท่ากับ 2n+1 โดยที่ n บิตแรกทางด้านซ้ายมือกำหนดให้มีค่าเป็น 0 ทุกบิต ต่อจากนั้นกำหนดค่าให้กับ n บิตถัดมาให้มีค่าเท่ากับตัวคูณ (Multiplier) และบิตสุดท้ายทางด้านขวามือสุด (LSB) ให้มีค่าเท่ากับ 0

ขั้นตอนที่ 2 พิจารณา 2 บิต สุดท้ายของผลคูณ ที่ได้กำหนดไว้ในข้อ 1 หากมีค่าเท่ากับ 00 หรือ 11 ไม่มีการคำนวณของขั้นตอนวิธี, หากมีค่าเป็น 01 ให้ทำการบวกค่าตัวตั้งเข้ากับผลคูณที่ตำแหน่ง n บิตแรก และหาก 2 บิต สุดท้ายมีค่าเท่ากับ 10 ให้นำค่าตัวตั้งลบออกจากผลคูณที่ตำแหน่ง n บิตแรก

ขั้นตอนที่ 3 ทำการเลื่อนบิตของผลคูณไปทางขวา 1 บิต และแทนที่บิตทางซ้ายสุดของผลคูณด้วยค่าเดิม

ขั้นตอนที่ 4 กลับไปที่ขั้นตอนที่ 2 ทำซ้ำขั้นตอนต่าง ๆ จนครบ n ครั้ง จากนั้นตัดบิตสุดท้ายของผลคูณที่ได้ทิ้ง จะได้ผลคูณที่ถูกต้อง

ตัวอย่างการใช้ขั้นตอนวิธีการคูณแบบ Booth Encode คูณเลขฐานสองขนาด 4 บิต แสดงดังตารางที่ 4.1 กำหนดให้ตัวตั้ง (Multiplicand) มีค่าเท่ากับ 0010_b และตัวคูณ (Multiplier) มีค่าเท่ากับ 0110_b ซึ่งจะได้ผลคูณ (Product) เท่ากับ 0001100_b

ตารางที่ 4.1 ตัวอย่างการคำนวณของขั้นตอนวิธีการคูณแบบ Booth Encode

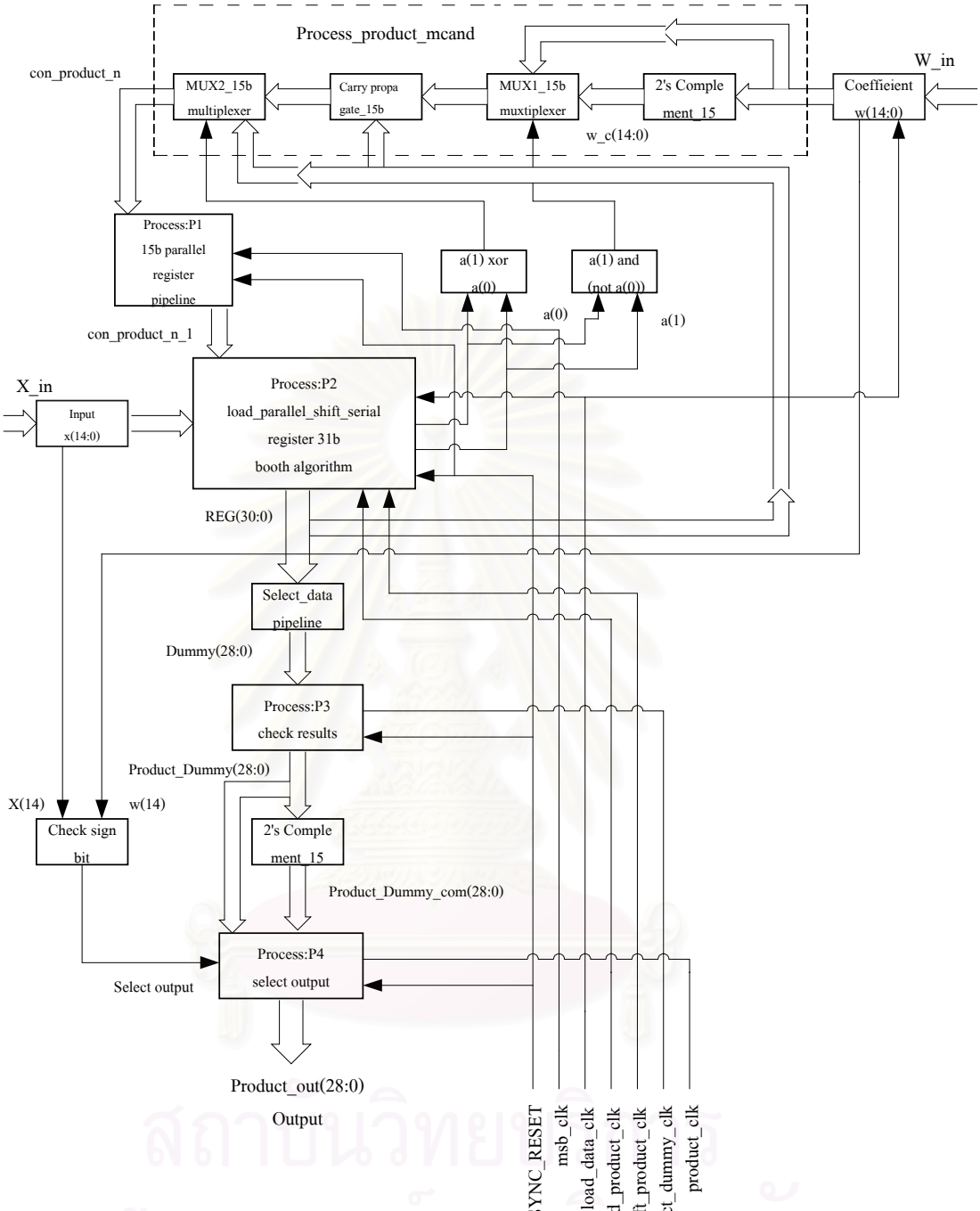
Iteration	Step	Multiplicand	Product
0	Initial Values	0010	0000 0110 0
1	00 => No Operation	0010	0000 0110 0
	Shift Right Product	0010	0000 0011 0
2	10 => Prod = Prod - Mcand	0010	1110 0011 0
	Shift Right Product	0010	1111 0001 1
3	11 => No Operation	0010	1111 0001 1
	Shift Right Product	0010	1111 1000 1
4	01 => Prod = Prod + Mcand	0010	0001 1000 1
	Shift Right Product	0010	0000 1100 0

จะเห็นว่าการคำนวณของขั้นตอนวิธีการคูณแบบ Booth Encode จะเป็นลำดับขั้นตอนที่ซ้ำ ๆ กัน แต่จะต้องมีการกำหนดช่วงเวลาในการทำงานให้ในแต่ละขั้นตอนให้แน่นอน ทำให้ต้องใช้สัญญาณ Clock ในการควบคุมการทำงานของวงจรให้ถูกต้องตามช่วงเวลาที่กำหนด

วงจรคูณ Booth Encode เมื่อทำการสร้างจริงจะช่วยประหยัดจำนวน Gate บนชิป FPGA ได้มาก แต่การคำนวณของขั้นตอนวิธีจะใช้เวลาในการคำนวณค่อนข้างสูง ซึ่งการคำนวณในแต่ละขั้นตอนจะใช้สัญญาณ Clock ต่าง ๆ ที่สร้างจากวงจรกำเนิดสัญญาณ Clock ที่ได้ทำการออกแบบวงจรด้วยภาษา VHDL ไว้ก่อนหน้านี้ ในส่วนของช่วงเวลาในการคำนวณ เมื่อขนาดของสัญญาณขาเข้าและค่าสัมประสิทธิ์ของ Filter Tap มีขนาดเท่ากับ 15 บิต ถึงแม้ว่าจะใช้เวลาในการคำนวณค่อนข้างมาก แต่เมื่อทำการคำนวณความถี่ของการสุ่มตัวอย่างในแต่รอบของการประมวลผลความถี่ของการสุ่มตัวอย่างยังคงสูงกว่าความถี่เสียง ดังนั้นสามารถใช้กับสัญญาณเสียงได้ ดังที่กล่าวมาแล้วในหัวข้อของวงจรกำเนิดสัญญาณ Clock

จากลำดับขั้นตอนการประมวลผลของขั้นตอนวิธีการคูณแบบ Booth Encode สามารถนำเอาหลักการดังกล่าวมาทำการสร้างจริงเป็นวงจรคูณ Booth Encode ได้ การสร้างจริงวงจรคูณ Booth Encode จะทำการสร้างจริงตามโครงสร้างที่ได้ออกแบบไว้ในรูปที่ 4.8

จากโครงสร้างของวงจรคูณ Booth Encode ในรูปที่ 4.8 สามารถอธิบายหลักการในการทำงานของวงจรได้ดังนี้ วงจรคูณ Booth Encode พร้อมที่จะทำงานเมื่อมีการ Reset ระบบ โดยใช้สัญญาณ ASYNC_RESET ซึ่งเป็นสัญญาณที่ใช้ Reset ระบบการทำงานของวงจรกรองปรับตัว เพื่อให้ระบบต่าง ๆ พร้อมที่จะทำงาน หลังจากมีการ Reset ระบบเรียบร้อยแล้ว การทำงานของวงจร Booth Encode จะทำงานในขั้นต่อไปเมื่อมีสัญญาณ load_data_clk สัญญาณดังกล่าวจะ Load ค่าสัญญาณขาเข้าและค่าสัมประสิทธิ์ของ Filter Tap เข้าไปในวงจรคูณ Booth Encode ซึ่งการทำงานในขั้นตอนนี้จะอยู่ที่กระบวนการ P2 ในรูปที่ 4.8 ซึ่งเป็นขั้นตอนที่ 1 ของขั้นตอนวิธีการคูณแบบ Booth Encode เนื่องจากค่าสัญญาณขาเข้าและค่าสัมประสิทธิ์ของวงจรกรองปรับตัวใช้เลขฐานสองที่มีขนาด 15 บิต ดังนั้นต้องใช้ตัวแปรที่เก็บค่าในการคำนวณเท่ากับ 31 บิต โดยที่ครั้งแรกของการทำงาน 15 บิตแรกทางด้าน MSB จะมีค่าเป็น 0 ทั้ง 15 บิต, 15 บิต ต่อจากนั้นจะมีค่าเท่ากับสัญญาณขาเข้าและบิตสุดท้าย LSB จะมีค่าเริ่มต้นเท่ากับ 0 นอกจากการ Set ค่าเริ่มต้นดังกล่าวแล้วจะมีการตรวจสอบ Sign bit ของตัวคูณทั้ง 2 จำนวน เพื่อใช้ตรวจเช็คในขั้นตอนสุดท้าย นอกจากนี้ค่าสัมประสิทธิ์ของ Filter Tap จะถูกนำเข้าไปที่กระบวนการ Process_product_mcand เพื่อนำไปผ่านกระบวนการ 2's Complement ต่อจากนั้นจะทำตามขั้นตอนที่ 2 โดยการตรวจเช็ค 2 บิตสุดท้าย ซึ่งการทำงานของวงจรจะนำ 2 บิตสุดท้ายมาพิจารณาสำหรับเลือกค่าของสัมประสิทธิ์ระหว่างค่าของสัมประสิทธิ์ที่รับเข้ามาจากภายนอก หรือนำเอาค่าสัมประสิทธิ์ที่ผ่านกระบวนการ 2's Complement ไปใช้ สำหรับทำการบวกหรือลบ หรือไม่ต้องทั้งสองอย่าง ขึ้นอยู่กับการพิจารณาค่า 2 บิตสุดท้ายของผลลัพธ์ที่ได้ในขั้นตอนที่ 2



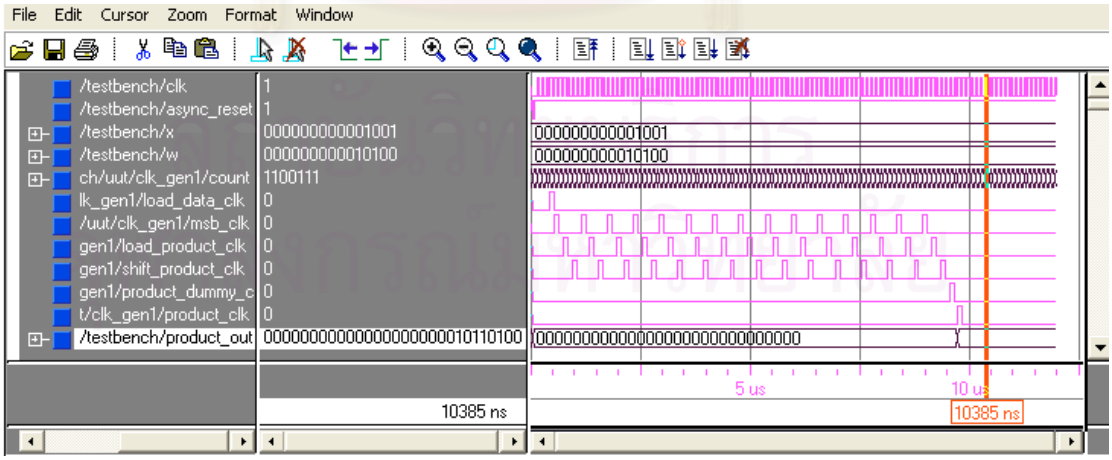
รูปที่ 4.8 โครงสร้างของวงจร Booth Encode

เมื่อทำตามขั้นตอนที่ 2 เสร็จเรียบร้อยแล้ว วงจรก็นำค่าที่คำนวณส่งต่อไปยังกระบวนการ P1 ซึ่งการทำงานมีลักษณะเหมือน Buffer เพื่อรอสัญญาณ msb_clk เมื่อมีสัญญาณ msb_clk เข้ามาก็จะทำการ Load ค่าที่คำนวณได้ไปเก็บไว้ในกระบวนการ P1 จากนั้นรอสัญญาณ load_product_clk เพื่อที่จะนำผลการคำนวณของ 15 บิต ทางด้าน MSB ที่ได้ไปเก็บไว้ที่ตัวแปรเริ่มต้นที่ตำแหน่ง 15

บิตแรกทางด้าน MSB ของตัวแปรที่มีขนาดเท่ากับ 31 บิต หลังจากนั้นรอจนกว่าจะมีสัญญาณ shift_product_clk เพื่อเลื่อนค่าของตัวแปรตั้งต้นไปทางขวา 1 บิต ในการเลื่อนบิตข้อมูลดังกล่าว บิตข้อมูลทางด้านซ้ายสุด MSB จะต้องมีค่าเท่ากับค่าเดิมก่อนการเลื่อนบิตข้อมูล เมื่อวงจรทำงานถึงขั้นตอนนี้จะทำทั้งหมด 15 รอบ (เนื่องจากใช้เลขฐานสองขนาด 15 บิต แทนค่าสัญญาณขาเข้าและสัมประสิทธิ์ของ Filter Tap) จึงจะจบการทำงานของขั้นตอนที่ 3

หลังจากการทำงานของวงจรผ่านขั้นตอนที่ 3 จะทำการเลือกบิตที่จะนำมาใช้เป็นผลลัพธ์ของผลคูณที่ได้เก็บไว้ในตัวแปร Dummy ที่มีขนาดเท่ากับ 29 บิต จากนั้นตรวจเช็คความถูกต้องของผลลัพธ์ที่ได้ จากรูปจะอยู่ที่กระบวนการ P3 หลังจากนั้นรอสัญญาณ product_dummy_clk เพื่อที่จะส่งผลที่ได้ตรวจเช็คเรียบร้อยแล้วไปเก็บได้ในตัวแปร Product_dummy เพื่อที่จะเช็คผลลัพธ์กับค่า Sign bit ของสัญญาณขาเข้าและสัมประสิทธิ์ให้มีความถูกต้อง หากค่า Sign bit ของสัญญาณขาเข้าและสัมประสิทธิ์เหมือนกันจะใช้ค่าในตัวแปร Product_dummy เป็นผลลัพธ์ที่ถูกต้อง ในทางตรงกันข้ามจะใช้ค่า Product_dummy ที่ผ่านกระบวนการ 2’s Complement เป็นผลลัพธ์ที่ถูกต้อง เมื่อทำการตรวจสอบความถูกต้องทุกอย่างเรียบร้อยแล้ว วงจรจะรอสัญญาณ product_clk เพื่อที่จะส่งผลลัพธ์ที่ถูกต้องออกสู่ทางด้าน Output ของวงจรคูณ Booth Encode เป็นการเสร็จสิ้นกระบวนการทำงานของวงจรคูณ Booth Encode

จากโครงสร้างของวงจรคูณ Booth Encode ดังแสดงในรูปที่ 4.8 เมื่อทำการออกแบบวงจรคูณดังกล่าวด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.9



รูปที่ 4.9 ผลการทำงานของวงจร Booth Encode ที่ออกแบบด้วยภาษา VHDL

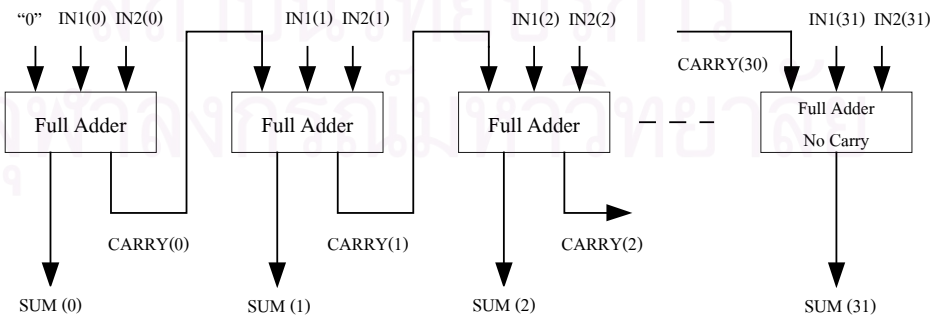
จากผลที่ได้จากการออกแบบวงจรด้วยภาษา VHDL ในส่วนของการสร้างจริง Filter Tap การทำงานของวงจรให้ผลลัพธ์ที่ถูกต้อง สามารถนำวงจรดังกล่าวไปใช้งานได้จริงในวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

4.3.3 การสร้างจริงวงจรบวก

วงจรบวกที่จะนำมาใช้ในการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง มี 2 แบบ คือ วงจรบวกแบบ Carry Propagate Adder และวงจรบวกแบบ Carry Save Adder

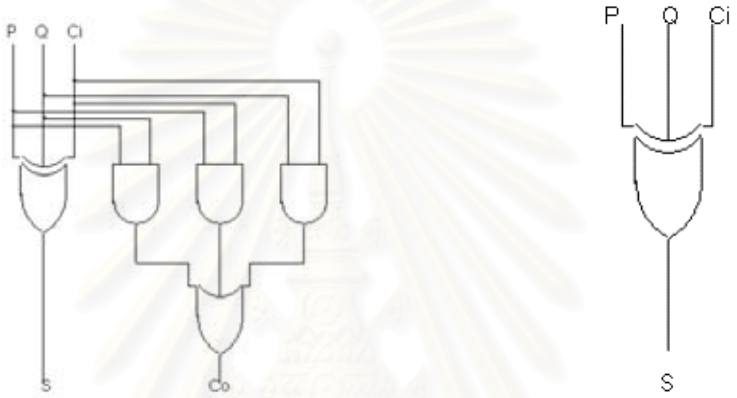
1. วงจรบวกแบบ Carry Propagate Adder

วงจรบวกแบบ Carry Propagate Adder เป็นวงจรบวกที่ใช้สำหรับรวมสัญญาณ 2 สัญญาณเข้าด้วยกัน ส่วนมากวงจรบวกแบบ Carry Propagate Adder ถูกนำมาใช้ในขั้นตอนสุดท้ายของการบวกในแต่ละกระบวนการ เนื่องจากต้องใช้เวลาในการคำนวณค่อนข้างสูง เพราะการบวกแบบ Carry Propagate Adder การบวกจะเริ่มที่บิตที่มีค่าต่ำสุด การบวกแต่ละครั้งจะสร้างตัวทด (Carry) ซึ่งตัวทดดังกล่าวจะถูกนำไปเป็น Input ของการบวกในบิตต่อไปด้วย ดังนั้นการบวกจะต้องใช้เวลาตั้งแต่การบวกบิตแรกที่มีค่าต่ำจนถึงบิตสุดท้ายที่มีค่าสูงสุด สำหรับวงจรกรองที่จะทำการสร้างจริงในการบวกค่าผลคูณที่ผ่านเพิ่มจำนวนบิตเรียบร้อยแล้วจะต้องใช้วงจรบวกแบบ Carry Propagate Adder ขนาด 32 บิต ซึ่งหลักการบวกของวงจรบวกแบบ Carry Propagate Adder จะเริ่มที่บิตที่มีค่าต่ำสุด โดยนำค่าของสัญญาณทั้งสองสัญญาณที่บิต 0 (IN1(0) และ IN2(0)) ป้อนเข้าวงจรบวกแบบ Full Adder ดังแสดงในรูปที่ 4.10 ที่ตำแหน่งดังกล่าว ตัวทดที่ถูกป้อนเข้าวงจรบวกแบบ Full Adder จะถูกกำหนดให้มีค่าเท่ากับศูนย์



รูปที่ 4.10 วงจรบวกแบบ Carry Propagate Adder

เมื่อวงจรบวกแบบ Full Adder ทำการรวมค่าสัญญาณที่บิต 0 ของสัญญาณทั้งสองเรียบร้อยแล้ว จะได้ผลลัพธ์ 2 ค่า คือ ผลรวมที่บิต 0 (SUM(0)) และตัวทดที่บิต 0 (Carry(0)) ผลรวมที่บิตดังกล่าวจะถูกส่งต่อไปที่ Output ของวงจรบวก ส่วนตัวทดจะถูกป้อนเป็นตัวทดขาเข้าให้กับการบวกในบิตต่อไป กระบวนการดังกล่าวจะเกิดขึ้นต่อไปจนถึงการบวกในบิตที่ 30 เมื่อถึงการบวกในบิตที่ 31 ซึ่งเป็นบิตสุดท้าย วงจรบวกที่นำมาใช้จะเป็นวงจรบวกแบบ Full Adder No Carry ซึ่งจะคำนวณหาเฉพาะค่าผลบวกในบิตที่ 31 เท่านั้น ซึ่งผลลัพธ์ที่ได้ของวงจรบวกแบบ Carry Propagate Adder ขนาด 32 บิต แสดงให้เห็นดังรูปที่ 4.10



S = P XOR Q XOR Ci

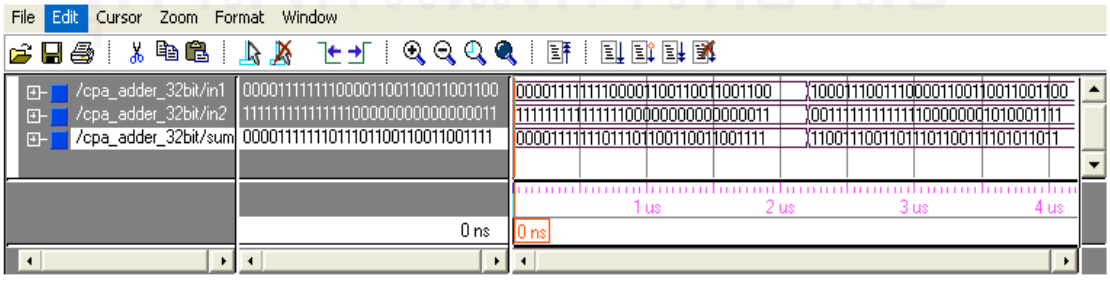
Co = (P AND Q) OR (Q AND Ci) OR (P AND Ci)

S = P XOR Q XOR Ci

(ก) (ข)

รูปที่ 4.11 วงจรบวกแบบ (ก) Full Adder (ข) Full Adder No Carry

รูปที่ 4.11 แสดงส่วนประกอบของวงจรบวกแบบ Carry Propagate Adder ซึ่งประกอบด้วยวงจรบวกแบบ Full Adder และ Full Adder No Carry สำหรับผลการทำงานของวงจรบวกแบบ Carry Propagate Adder เมื่อทำการออกแบบวงจรดังกล่าวด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างสร้างจริงได้ดังรูปที่ 4.12

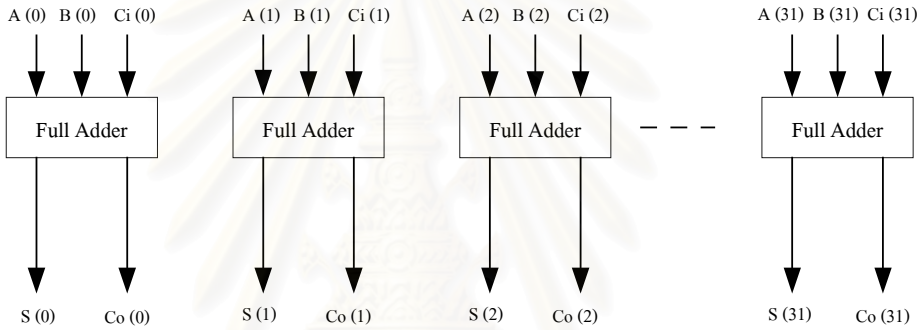


รูปที่ 4.12 ผลการทำงานของวงจรบวกแบบ Carry Propagate Adder ที่ออกแบบด้วยภาษา VHDL

2. วงจรบวกแบบ Carry Save Adder

วงจรบวกแบบ Carry Save Adder เป็นวงจรบวกที่นิยมนำมาใช้ในการรวมค่าสัญญาณที่มีจำนวนมาก เพื่อความรวดเร็วในการประมวลผล สำหรับการใช้งานจริงจะนำเอาวงจรบวกแบบ Carry Save Adder มาใช้ในการคำนวณจนกระทั่งถึงขั้นตอนในการบวกครั้งสุดท้ายจึงจะใช้การบวกแบบ Carry Propagate Adder

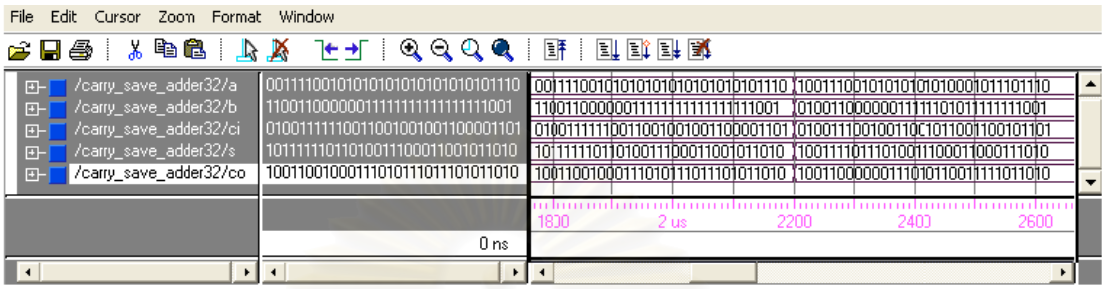
โครงสร้างของวงจรบวกแบบ Carry Save Adder ขนาด 32 บิต ที่จะทำการสร้างจริง เพื่อนำไปใช้ในการประมวลผลของวงจรกรองปรับตัว จะนำเอาวงจรบวกแบบ Full Adder มาใช้ในการประมวลผลดังแสดงในรูปที่ 4.13



รูปที่ 4.13 วงจรบวกแบบ Carry Save Adder

วงจรบวกแบบ Carry Save Adder ที่จะทำการสร้างจริงมีโครงสร้างตามรูปที่ 4.13 การทำงานของวงจรจะต้องป้อนสัญญาณ Input ให้กับวงจรบวกแบบ Carry Save Adder 3 สัญญาณ ส่วนประกอบภายในของวงจรจะใช้วงจรบวกแบบ Full Adder ซึ่งการประมวลผลภายในแต่ละบิตจะเป็นอิสระจากกัน จากรูปที่ 4.13 สัญญาณ Input ที่ป้อนให้กับวงจรบวกแบบ Carry Save Adder คือ สัญญาณ A, B และ Ci ซึ่งมีขนาด 32 บิต โดยที่การคำนวณภายในจะแยกการคำนวณของแต่ละบิตออกจากกัน การประมวลผลในบิตที่ 0 ผลบวก S(0) และตัวทด Co(0) ได้จาก A(0), B(0) และ Ci (0) จนกระทั่งถึงบิตสุดท้าย ผลบวก S(31) และตัวทด Co(31)ได้จากการรวมกันของ A(31), B(31) และ Ci(31) การประมวลในแต่ละบิตจะทำงานเหมือนกัน จากผลลัพธ์ที่ได้ของวงจรบวกแบบ Carry Save Adder จะได้ผลบวกและตัวทด ค่าของผลบวกสามารถนำไปใช้งานได้ทันที ส่วนค่าของตัวทดจะต้องทำการเลื่อนตำแหน่งไปทางด้าน MSB 1บิต จากนั้นใส่ 0 เข้าไปในบิตที่มีค่าต่ำสุด LSB จึงจะสามารถนำค่าดังกล่าวไปใช้งานในกระบวนการต่อไปได้

จากหลักการในการทำงานของวงจรบวกแบบ Carry Save Adder เมื่อทำการออกแบบวงจรดังกล่าวด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.14

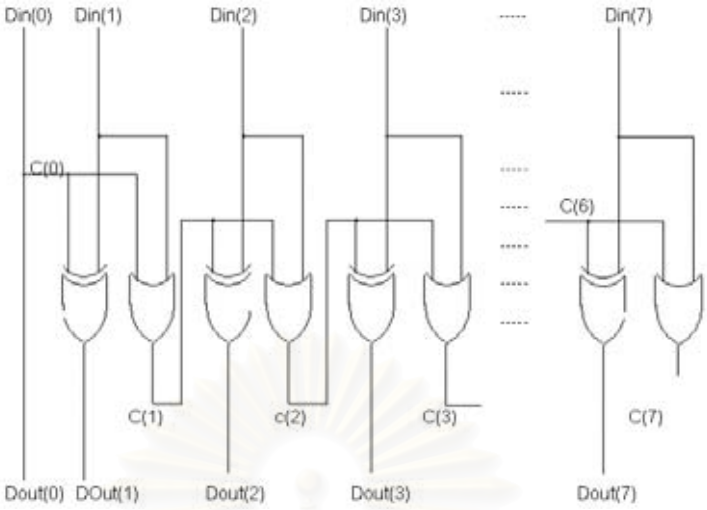


รูปที่ 4.14 ผลการทำงานของวงจรบวกแบบ Carry Save Adder ที่ออกแบบด้วยภาษา VHDL

4.3.4 การสร้างจริงวงจร 2's Complement

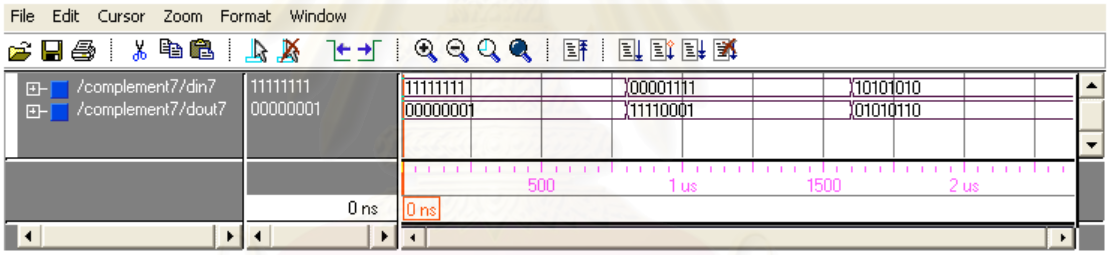
วงจร 2's Complement เป็นวงจรที่มีความสำคัญในการบวกและลบ ทำให้ผู้ออกแบบวงจรสามารถประยุกต์นำเอากระบวนการบวกมาใช้แทนการลบได้อย่างมีประสิทธิภาพ ซึ่งวงจรดังกล่าวจะเปลี่ยนค่าของสัญญาณขาออกให้มีค่าตรงกันข้ามกับสัญญาณขาเข้า โดยที่ขนาดของสัญญาณยังคงเดิม เช่น สัญญาณขาเข้าของวงจรเท่ากับ 01010100_b (มีค่าเท่ากับ 84 ในเลขฐานสิบ) จะได้สัญญาณขาออกมีค่าเท่ากับ 10101100_b (มีค่าเท่ากับ –84 ในเลขฐานสิบ) การออกแบบวงจรจะกำหนดตัวแปรขึ้นมา 3 ตัว คือ สัญญาณขาเข้า (Din) ตัวทด (C) และสัญญาณขาออก (Dout) ซึ่งตัวอย่างของวงจรที่จะนำเสนอนี้เป็นวงจร 2's Complement ขนาด 8 บิต ดังนั้นตัวแปรที่กำหนดขึ้นมาในวงจรทั้ง 3 ตัว จะมีขนาดเท่ากับ 8 บิต

การทำงานของวงจรจะเริ่มต้นที่บิต 0 (LSB) ที่บิตดังกล่าวค่าของสัญญาณขาออกจะมีค่าเท่ากับค่าของสัญญาณขาเข้า (Dout(0) = Din(0)) และค่าของตัวทด C(0) จะถูกกำหนดให้มีค่าเท่ากับ Din(0) สำหรับค่าของสัญญาณขาออกในบิตต่อไปสามารถหาได้ดังนี้ นำค่าของสัญญาณในบิตดังกล่าวทำกระบวนการ Exclusive Or กับตัวทดในบิตที่ผ่านมา ตัวอย่างเช่น Dout(1) = Din(1) XOR C(0) เป็นต้น สำหรับค่าของตัวทดสามารถหาได้จากการนำค่าของสัญญาณขาเข้าในบิตที่ต้องการหาค่าตัวทด มาทำกระบวนการ Or กับตัวทดในบิตที่ผ่านมา ตัวอย่างเช่น C(1) = Din(1) Or C(0) สิ่งที่สำคัญในการทำงานของวงจร คือ ต้องกำหนดค่าเริ่มต้นทั้ง 2 ค่า คือ สัญญาณขาออกที่บิต 0 (Dout(0)) และตัวทดที่บิต 0 (C(0)) ให้มีค่าถูกต้องตามข้อกำหนด หลังจากนั้นจึงทำการคำนวณค่าสัญญาณขาออกและตัวทดที่เหลือในบิตต่อไป จากหลักการในการทำงานของวงจร 2's Complement ที่ผ่านมาสามารถเขียนเป็นรูปวงจรได้ดังรูปที่ 4.15



รูปที่ 4.15 วงจร 2's Complement

สำหรับผลการทำงานของวงจร 2's Complement เมื่อทำการออกแบบวงจรดังกล่าวด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.16



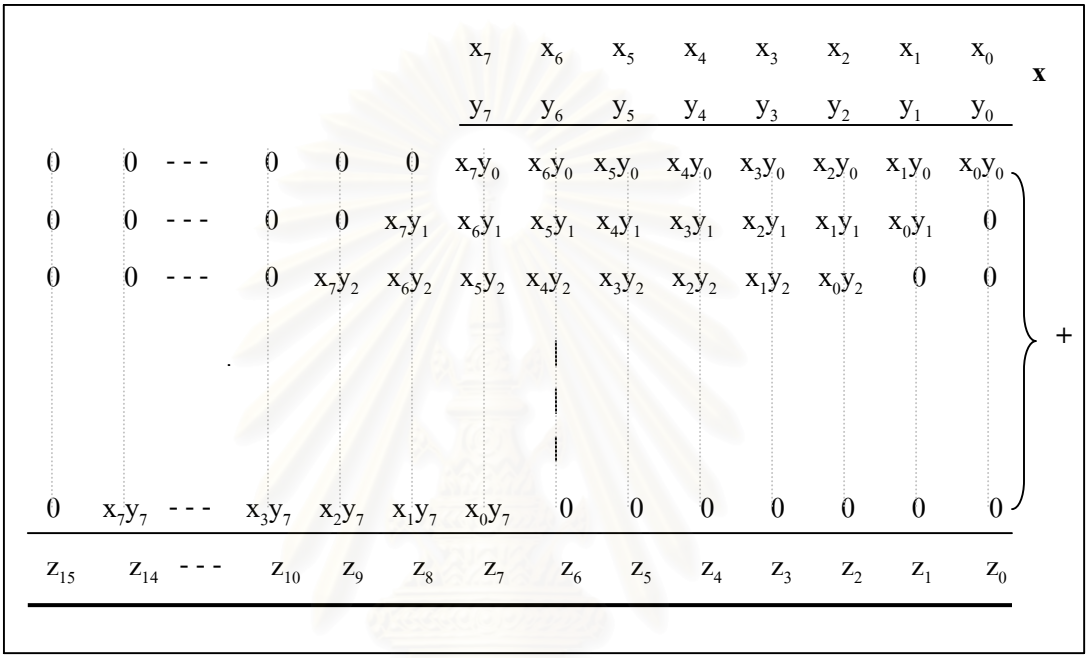
รูปที่ 4.16 ผลการทำงานของวงจร 2's Complement ที่ออกแบบด้วยภาษา VHDL

4.3.5 การสร้างจริงวงจรคูณแบบ Array

วงจรคูณแบบ Array เป็นวงจรคูณอีกแบบหนึ่งที่จะนำมาใช้ในการออกแบบวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง การสร้างจริงของวงจรกรองปรับตัวดังกล่าว จะใช้วงจรคูณแบบ Array ในส่วนของการปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง

การคูณแบบ Array จะมีลักษณะดังรูปที่ 4.17 ซึ่งแสดงตัวอย่างการคูณเลขฐานสองที่มีขนาด 8 บิต กำหนดให้ตัวตั้งเท่ากับ X = $x_7x_6x_5x_4x_3x_2x_1x_0$ ตัวคูณเท่ากับ Y = $y_7y_6y_5y_4y_3y_2y_1y_0$และผลลัพธ์เท่ากับ Z ในการคูณจะคูณทีละบิต เริ่มจากบิตที่มีค่าต่ำสุด นำบิตที่มีค่าต่ำสุดของตัวคูณ y_0 คูณเข้ากับตัวตั้ง X ทุกบิต จะได้ผลคูณทั้งหมด 8 บิต ส่วนบิตที่เหลือด้านหน้าจะแทนด้วย 0 ทั้ง 8 บิต (เนื่องจากการคูณเลขฐานสอง 8 บิต 2 ตัวจะได้ผลลัพธ์มีขนาด 16 บิต) ซึ่งจะได้ผลคูณเป็นแถว

ที่ 1 ต่อจากนั้นจะใช้ ตัวคูณบิตถัดมา คือ y_1 เป็นตัวคูณ ผลคูณของบิตแรกจะต้องใส่ไว้ในตำแหน่งเดียวกับตัวคูณ y_1 เมื่อทำการคูณครบทั้ง 8 บิตแล้วในบิตที่เหลือจะแทนค่าด้วยเลข 0 ซึ่งผลลัพธ์ที่ได้จะอยู่ในแถวที่ 2 ทำตามขั้นตอนดังกล่าวจนครบทั้ง 8 บิตของตัวคูณ หลังจากนั้นทำการรวมค่าในแต่ละคอลัมน์ของผลคูณที่ได้แต่ละแถว จะได้ผลลัพธ์ $Z = z_{15}z_{14}z_{13}z_{12}z_{11}z_{10}z_9z_8z_7z_6z_5z_4z_3z_2z_1z_0$ ที่มีขนาดเท่ากับ 16 บิต

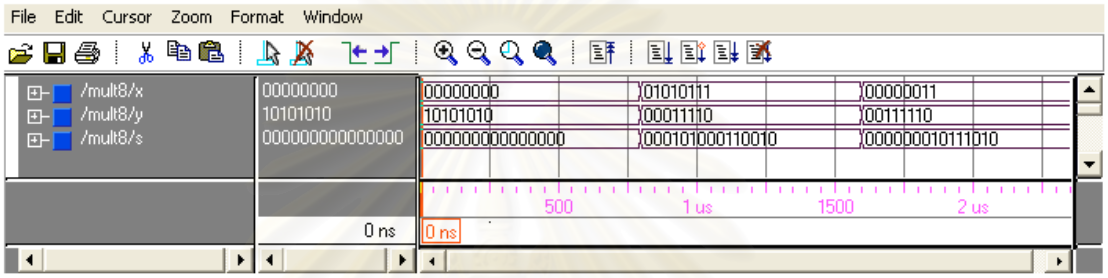


รูปที่ 4.17 วิธีการคูณแบบ Array

จากรูปแบบการคูณแบบ Array ดังแสดงในรูปที่ 4.17 ในการออกแบบวงจรคูณดังกล่าวด้วยภาษา VHDL จะอาศัยการเลื่อนบิตในการคูณ หากบิตของตัวคูณที่นำมาพิจารณามีค่าเป็น 1 ให้นำตัวตั้งมาใส่ไว้ที่ผลคูณของบิตดังกล่าว โดยที่บิตที่มีค่าต่ำสุดของตัวตั้งต้องอยู่ที่ตำแหน่งเดียวกับบิตของตัวคูณ ส่วนบิตอื่น ๆ ของตัวตั้งจะถูกใส่ไว้ในตำแหน่งที่มีค่าสูงขึ้นไปทางด้าน MSB และบิตที่เหลือให้แทนด้วย 0 ทั้งหมด ในกรณีที่บิตของตัวคูณที่นำมาพิจารณามีค่าเป็น 0 ผลคูณในแถวดังกล่าวให้แทนค่าด้วย 0 ทั้งแถว เมื่อทำการคูณโดยวิธีดังกล่าวซึ่งใช้วิธีพิจารณาจำนวนบิตที่เป็นเลข 1 และเลข 0 ครบทุกบิตแล้วต่อจากนั้นก็ใช้วงจรบวกที่ได้กล่าวมาแล้วก่อนหน้านี้คำนวณหาผลลัพธ์ของผลคูณที่ได้ เพื่อความสะดวกในการคูณอาจจะกำหนดให้ตัวตั้งและตัวคูณมีค่าเป็นบวกทั้งสองตัวก็ได้โดยใช้วงจร 2's Complement เปลี่ยนค่าดังกล่าวให้มีค่าเป็นบวก แต่จะต้องเก็บค่า Sign bit ครั้งแรกของทั้ง 2 ตัวไว้พิจารณาด้วยคือ หากค่า Sign bit ของทั้ง 2 ตัวเหมือนกัน สามารถที่จะนำผลคูณดังกล่าวเป็นผลลัพธ์ได้ทันที แต่ถ้าค่าของ Sign bit มีค่าต่างกัน ผลคูณที่ได้ต้องนำไปผ่าน

กระบวนการ 2's Complement ด้วยวงจร 2's Complement ที่ออกแบบไว้อีกครั้งจึงจะสามารถนำผลคูณดังกล่าวไปเป็นผลลัพธ์ได้

เนื่องจากวงจรกรองปรับตัวที่จะทำการสร้างจริง จะนำเอาวงจรคูณแบบ Array มาใช้งานจริง จึงนำเอาวิธีการคูณแบบ Array ที่นำเสนอข้างต้นมาสร้างเป็นวงจรคูณแบบ Array ด้วยภาษา VHDL และสำหรับผลการทำงานของวงจรคูณแบบ Array ที่ทำการออกแบบวงจรด้วยภาษาดังกล่าว สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.18



รูปที่ 4.18 ผลการทำงานของวงจรคูณแบบ Array ที่ออกแบบด้วยภาษา VHDL

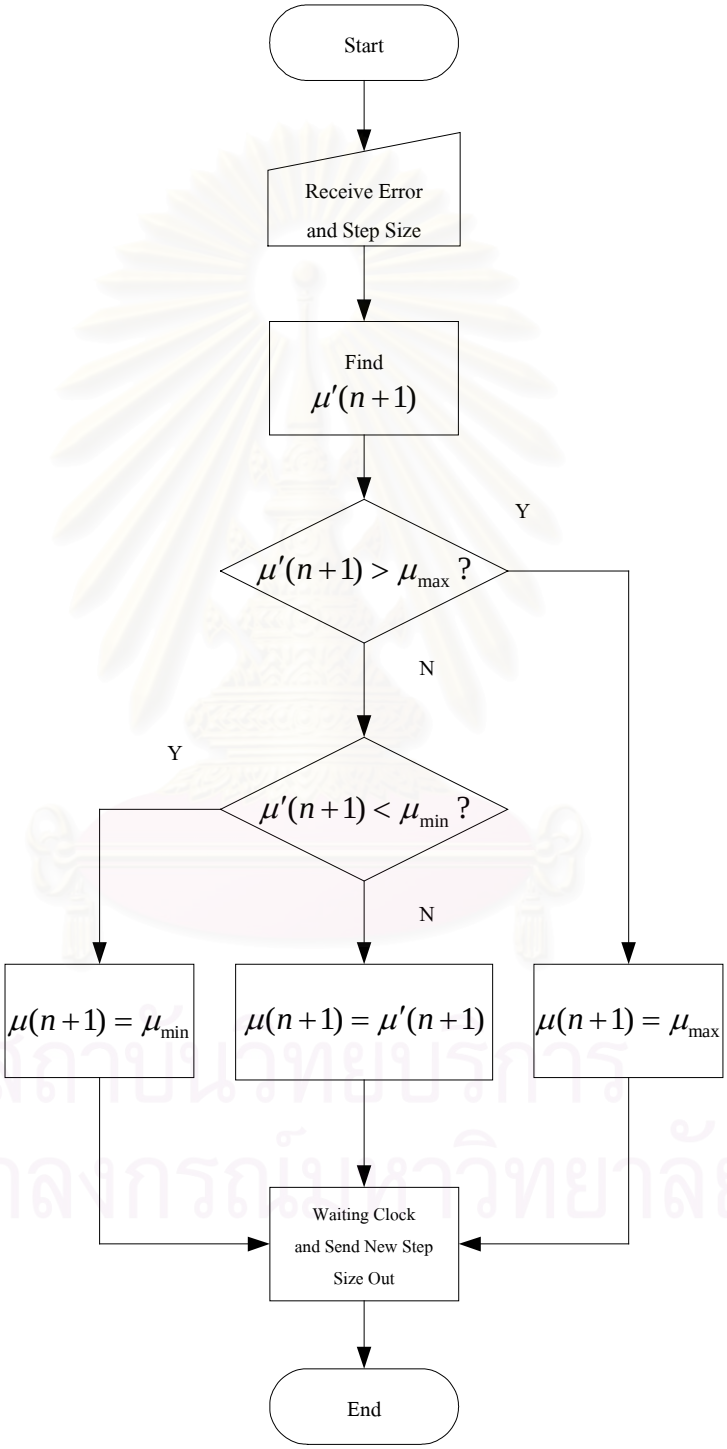
4.3.6 การสร้างจริงวงจรปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง

วิธีการปรับช่วงก้าวแบบพลวัตที่นำมาใช้ในวิทยานิพนธ์นี้เป็นวิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสอง (Squared Error) ซึ่งมีสมการสำหรับการปรับค่าช่วงก้าวดังสมการที่ (4-2) และสมการที่ (4-3)

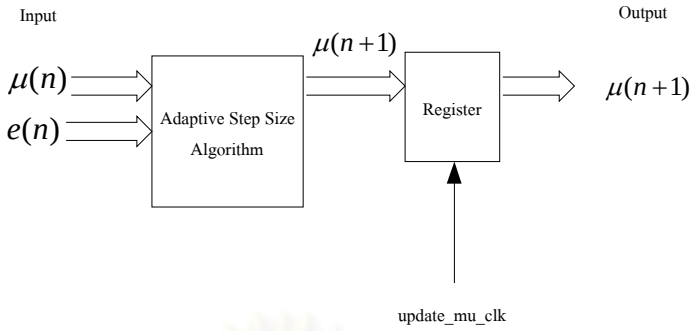
จากสมการที่ (4-2) $\mu'(n+1) = \alpha\,\mu(n) + \gamma e^2(n)$ การคำนวณค่าช่วงก้าวที่เวลา $n+1$ จะเริ่มจากการคำนวณหาผลคูณของค่าความผิดพลาดกำลังสอง $e^2(n)$ ในขณะเดียวกันก็จะหาค่าผลคูณของค่าช่วงก้าวที่เวลา n กับค่าคงตัว α จากนั้นจะหาค่าผลคูณของค่าคงตัว γ กับค่าความผิดพลาดกำลังสอง $e^2(n)$ โดยที่กระบวนการคูณที่ผ่านมาจะใช้วงจรคูณแบบ Array เมื่อทำการคูณเสร็จเรียบร้อยแล้ว ต่อจากนั้นจะทำกระบวนการบวกต่อไปตามสมการ โดยใช้วงจรบวกแบบ Carry Propagate Adder จากนั้นนำค่าช่วงก้าวที่คำนวณได้ไปเปรียบเทียบกับค่าช่วงก้าวที่มีค่าสูงสุดและต่ำสุด $\mu_{\max}$ และ $\mu_{\min}$ ตามลำดับ เพื่อหาค่าช่วงก้าวที่เหมาะสมตามสมการที่ (4-3) จากลำดับขั้นตอนในการคำนวณดังกล่าว สามารถเขียน Flowchart การทำงานของวงจรที่จะทำการออกแบบด้วยภาษา VHDL สำหรับใช้เป็นวงจรปรับค่าช่วงก้าวแบบค่าความผิดพลาดกำลังสอง ดังแสดงในรูปที่ 4.19

จาก Flowchart ในรูปที่ 4.19 สามารถอธิบายการทำงานของวงจรสำหรับการปรับค่าช่วงก้าวแบบค่าความผิดพลาดกำลังสอง ที่จะใช้ในการสร้างจริงได้ดังนี้ เมื่อวงจรกรองคำนวณค่าความ

ผิดพลาดเสร็จเรียบร้อยแล้ว วงจรปรับค่าช่วงก้าวแบบค่าความผิดพลาดกำลังสองจะนำค่าความผิดพลาดและค่าช่วงก้าวที่เวลา n เข้ามาในวงจรและเริ่มทำการคำนวณหาค่าช่วงก้าวที่เวลา $n+1$ โดยใช้สมการที่ (4-2)

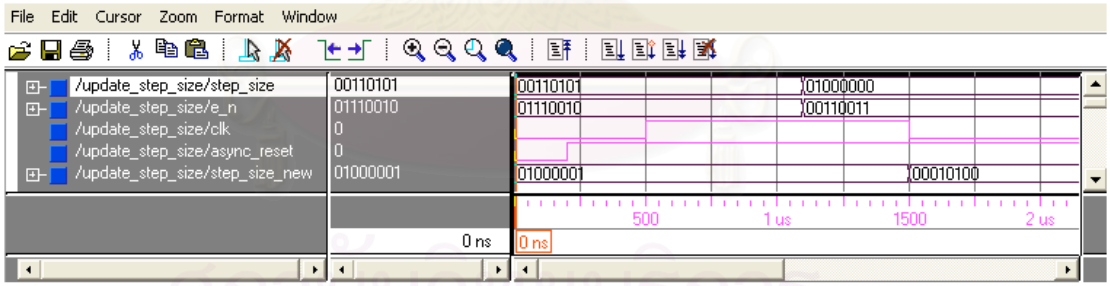


รูปที่ 4.19 Flowchart การทำงานของวงจรสำหรับปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง



รูปที่ 4.20 โครงสร้างของวงจรปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง

เมื่อได้ผลลัพธ์เรียบร้อยแล้ว นำไปเปรียบเทียบกับค่าช่วงก้าวสูงสุดและต่ำสุดเพื่อตรวจสอบขนาดของช่วงก้าวให้อยู่ในช่วงที่ได้กำหนดไว้ หลังจากนั้นนำค่าช่วงก้าวที่คำนวณได้ไปเก็บไว้ใน Regiter โดยอาศัยการทำงานของขอบขาลงของสัญญาณ update_mu_clk ดังแสดงในรูปที่ 4.20 หลังจากนั้นค่าช่วงก้าวที่คำนวณได้จะถูกส่งต่อไปยัง Register ตัวต่อไป เพื่อรอสัญญาณ data_clk เข้ามาจึงจะ Load ค่าช่วงก้าวที่คำนวณได้ไปใช้ในการประมวลผลในรอบต่อไป สำหรับผลการทำงานของวงจรปรับค่าช่วงก้าวแบบค่าความผิดพลาดกำลังสอง ที่ทำการออกแบบวงจรดังกล่าวด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรดังกล่าวที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.21

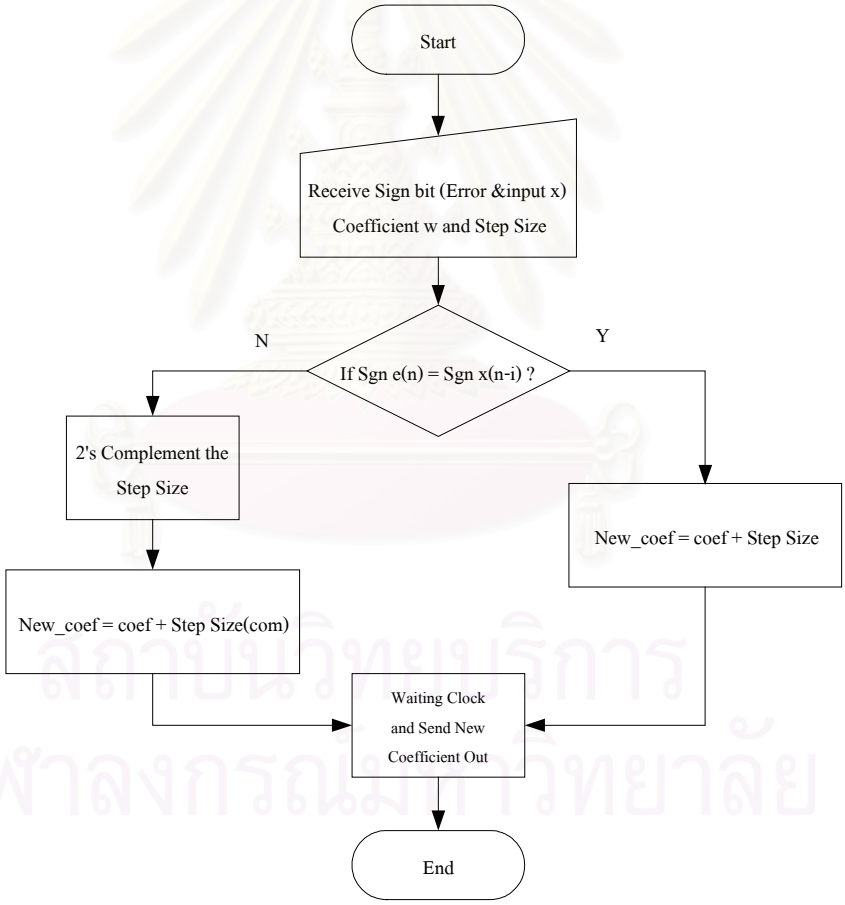


รูปที่ 4.21 ผลการทำงานของวงจรปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสอง ที่ออกแบบด้วยภาษา VHDL

4.3.7 การสร้างจริงวงจรปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวโดยใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign

การสร้างจริงวงจรสำหรับปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว ซึ่งใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ในการปรับค่าสัมประสิทธิ์ เริ่มต้นจากการพิจารณาสมการที่ (4-4) และสมการที่ (4-5)

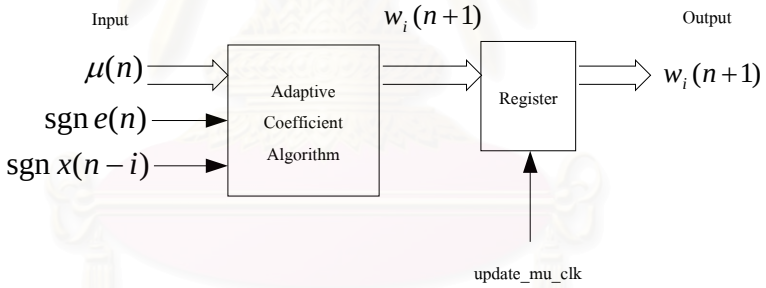
เริ่มพิจารณาจากสมการที่ (4-4) $w_i(n+1) = w_i(n) + \mu(n)\,[\mathrm{sgn}(x(n-i))\ \mathrm{sgn}(e(n))]$ ค่าสัมประสิทธิ์ของวงจรกรองปรับตัวใน Tap ที่ i ที่เวลา $n+1$ สามารถคำนวณหาได้จากค่าสัมประสิทธิ์ของวงจรกรองปรับตัวใน Tap ที่ i ที่เวลา n บวกหรือลบกับค่าช่วงก้าวที่เวลา n โดยจะพิจารณาค่า Sign bit ของสัญญาณขาเข้าของ Filter Tap ณ Tap i ที่เวลา n และค่า Sign bit ของค่าความผิดพลาดที่หาได้จากวงจรกรองปรับตัวที่เวลา n ถ้าค่า Sign bit ของทั้งสองค่าเหมือนกันจะใช้การบวกในการปรับค่าสัมประสิทธิ์ของ Tap ที่ i โดยใช้วงจรบวกแบบ Carry Propagate Adder แต่ถ้าหากว่าค่า Sign bit ของทั้งสองต่างกันก็จะนำค่าช่วงก้าว ไปผ่านกระบวนการ 2's Complement ด้วยวงจร 2's Complement แล้วจึงนำค่าที่ได้ไปทำการบวกกันด้วยวงจรบวกแบบ Carry Propagate Adder แทนการลบ จากขั้นตอนในการคำนวณดังกล่าวสามารถเขียน Flowchart การทำงานของวงจรซึ่งจะออกแบบวงจรดังกล่าวด้วยภาษา VHDL ได้ดังรูปที่ 4.22



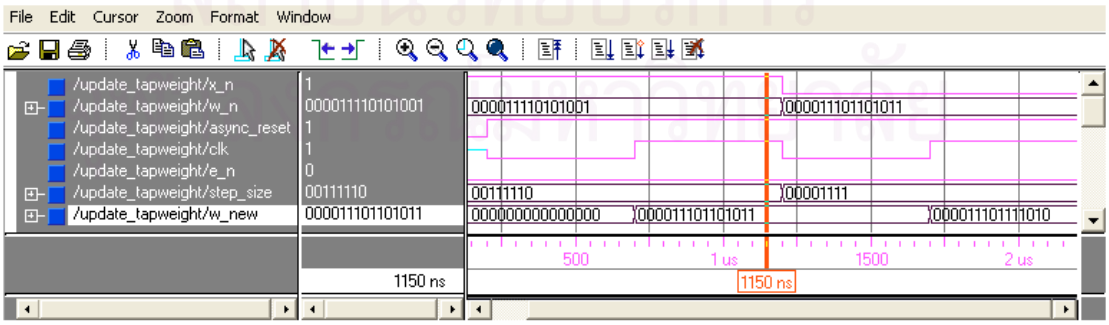
รูปที่ 4.22 Flowchart การทำงานของวงจรสำหรับปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว

จาก Flowchart การทำงานของวงจรสำหรับปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวในรูปที่ 4.22 วงจรจะเริ่มคำนวณค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่เวลา $n+1$ ในแต่ละ Tap เมื่อวงจรกรองปรับตัวได้ทำการคำนวณค่าความผิดพลาด $e(n)$ เสร็จเรียบร้อยแล้ว ค่า Sign bit ของค่า

ความผิดพลาด และค่า Sign bit ของค่าสัมประสิทธิ์ของวงจรกรองใน Tap ที่ i ที่เวลา n จะถูกนำมาพิจารณาในการปรับค่าสัมประสิทธิ์ของวงจรกรองใน Tap ที่ i ซึ่งการปรับค่าสัมประสิทธิ์ในแต่ละ Tap ของวงจรกรองปรับตัวจะทำไปพร้อม ๆ กัน และเป็นอิสระจากกันในการคำนวณแต่จะใช้ค่าช่วงก้าวที่เวลา n ค่าเดียวกัน ตามสมการที่ (4-4) ซึ่งในการคำนวณค่าสัมประสิทธิ์ในแต่ละ Tap ถ้าหากค่า Sign bit ของค่าความผิดพลาดและสัญญาณขาเข้าของ Filter Tap มีค่าเท่ากัน วงจรจะทำการปรับค่าสัมประสิทธิ์โดยการนำเอาช่วงก้าวที่รับเข้ามาทางด้าน Input โดยจะต้องนำค่าทั้งสองมาเรียงให้มีบิตตรงกันก่อน แล้วจึงทำการบวกกัน แต่ถ้ามีค่าต่างกันจะต้องนำค่าช่วงก้าวที่รับเข้ามาไปผ่านกระบวนของวงจร 2's Complement จากนั้นจึงนำค่าทั้ง 2 มาเรียงให้มีบิตตรงแล้วจึงทำการบวก เมื่อทำการคำนวณค่าสัมประสิทธิ์ของแต่ละ Tap เสร็จเรียบร้อยแล้ว จะส่งค่าสัมประสิทธิ์ที่ได้ในแต่ละ Tap ไปเก็บไว้ที่ Register ซึ่งจะใช้สัญญาณ update_mu_clk ที่ขอบขาขึ้นในการรับค่าดังกล่าวมาเก็บไว้ใน Register และเมื่อมีสัญญาณ data_clk ค่าสัมประสิทธิ์ดังกล่าวก็จะถูก Load เข้าไปเก็บไว้ใน Register ที่ใช้เก็บค่าใน Filter Tap อีกครั้งหนึ่ง เพื่อนำไปใช้ในการประมวลในผลรอบต่อไป จากขั้นตอนในการทำงานของวงจรปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่ผ่านมาสามารถแสดงโครงสร้างของวงจรได้ดังรูปที่ 4.23



รูปที่ 4.23 โครงสร้างของวงจรปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวสำหรับ Tap ที่ i



รูปที่ 4.24 ผลการทำงานของวงจรปรับค่าสัมประสิทธิ์ของวงจกรองปรับตัว ที่ออกแบบด้วยภาษา VHDL

สำหรับผลการทำงานของวงจรปรับค่าสัมประสิทธิ์ ที่ทำการออกแบบวงจรด้วยภาษา VHDL สามารถแสดงผลการทำงานของวงจรที่จะนำไปใช้ในการสร้างจริงได้ดังรูปที่ 4.24

ส่วนประกอบต่าง ๆ ที่ได้กล่าวมาแล้วทั้งหมด เป็นส่วนประกอบที่จะนำไปใช้ในการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ตามโครงสร้างทางสถาปัตยกรรมของวงจรกรองปรับตัวในรูปที่ 4.1 ซึ่งส่วนประกอบต่าง ๆ ทั้งหมดได้ทำการออกแบบวงจรด้วยภาษา VHDL สำหรับส่วนประกอบอื่น ๆ ที่ไม่ได้นำเสนอไว้ ณ ที่นี้จะเป็นส่วนประกอบที่ทำหน้าที่เช่นเดียวกับส่วนประกอบหลัก ๆ ที่ได้นำเสนอไปแล้ว เช่น Register ขนาด 8 บิต วงจรบวกแบบ Carry Propagate Adder ขนาด 8 บิต และวงจรบวกแบบ Carry Propagate Adder ขนาด 15 บิต เป็นต้น ซึ่งรายละเอียดและหลักการทำงานต่าง ๆ จะเหมือนกับ Register ขนาด 15 บิต และวงจรบวกแบบ Carry Propagate Adder ขนาด 32 บิต ที่นำเสนอไปแล้วก่อนหน้านี้

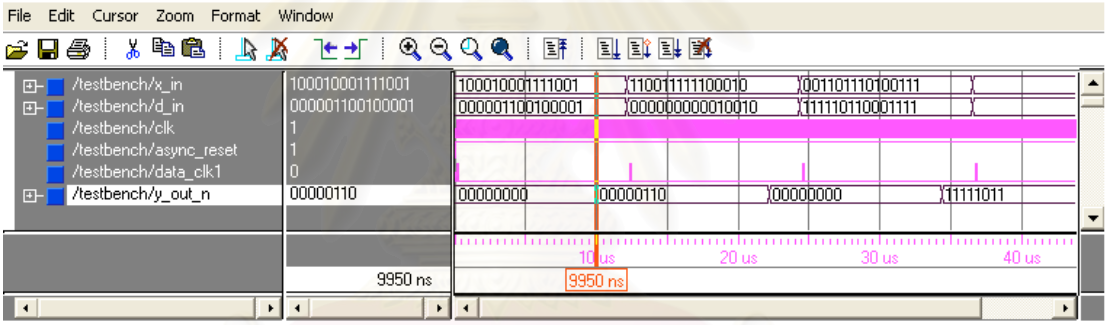
4.4 การสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง

การสร้างจริงส่วนประกอบต่าง ๆ ที่ได้ทำการออกแบบวงจรด้วยภาษา VHDL ซึ่งได้กล่าวมาแล้วก่อนหน้านี้ เป็นส่วนประกอบที่จะนำมาใช้ เพื่อประกอบรวมกันเป็นวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง หลังจากที่ส่วนประกอบต่าง ๆ เช่น Register วงจรกำเนิดสัญญาณ Clock วงจรบวกแบบ Carry Save Adder วงจรคูณแบบ Booth Encode และวงจร 2's Complement เป็นต้น สามารถทำงานได้จริงแล้ว (สามารถทำงานในระดับ Logic ได้) จึงนำเอาส่วนประกอบต่าง ๆ ที่ได้ทำการออกแบบวงจรดังกล่าวด้วยภาษา VHDL มาประกอบรวมกันเป็นวงจรกรองปรับตัวตามโครงสร้างของสถาปัตยกรรมของวงจรกรองปรับตัว ที่ได้ทำการออกแบบไว้

หลังจากนำเอาส่วนประกอบต่าง ๆ ที่ทำการออกแบบวงจรด้วยภาษา VHDL มาประกอบรวมกันเป็นวงจรกรองปรับตัวสำเร็จเรียบร้อยแล้ว จะต้องทำการทดสอบวงจรดังกล่าวที่ได้ออกแบบไว้ โดยการเขียนวงจรที่ใช้สำหรับตรวจสอบการทำงานของระบบ หรือที่เรียกว่า Testbench ครอบวงจรกรองปรับตัวที่ได้ทำการออกแบบไว้ เพื่อตรวจสอบความถูกต้องในการทำงานของวงจร ซึ่งสิ่งที่สำคัญในการจำลองผลการทำงานของวงจรที่ออกแบบด้วยภาษา VHDL คือ การจำลองผลการทำงานจะทำบนโปรแกรม Xilinx ISE ซึ่งจะต้องจำลองผลการทำงานในระดับ Function (Behavioral) เพื่อตรวจสอบความถูกต้องในการทำงานของวงจรที่ได้ทำการออกแบบไว้ และจำลองผลการทำงานในระดับ Gate (Logic) เพื่อตรวจสอบผลการทำงานของวงจรที่จะเกิดขึ้นจริง เมื่อทำ

การ Download วงจรดังกล่าวที่ได้ออกแบบไว้ลงบน FPGA ซึ่งผลการจำลองจะรวมผลของ Delay Time ภายในระหว่าง Gate แต่ละตัวที่จะเกิดขึ้นจริงด้วย ถ้าผลของการจำลองการทำงานในระดับ Gate ยังคงถูกต้อง สามารถสรุปได้ว่า วงจรดังกล่าวที่ได้ทำการออกแบบด้วยภาษา VHDL สามารถใช้งานได้จริงและสามารถทำการสร้างจริงในรูปแบบของวงจรรวม VLSI ได้

การจำลองผลการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง เพื่อตรวจสอบความถูกต้องในการทำงานของวงจรที่ออกแบบด้วยภาษา VHDL ว่าจะสามารถทำงานได้จริงก่อนที่จะนำเอาวงจรที่ออกแบบดังกล่าวไปใช้งานจริงบน FPGA จะต้องทำตามขั้นตอนดังนี้ เริ่มต้นจากการเปรียบเทียบผลการทำงานของวงจรกรองปรับตัวที่ทำการจำลองผลบนโปรแกรม Modelsim เปรียบเทียบกับการจำลองผลการทำงานในระดับ Function บนโปรแกรม Xilinx ISE เพื่อตรวจสอบความถูกต้องในการทำงานของวงจรที่ได้ออกแบบไว้

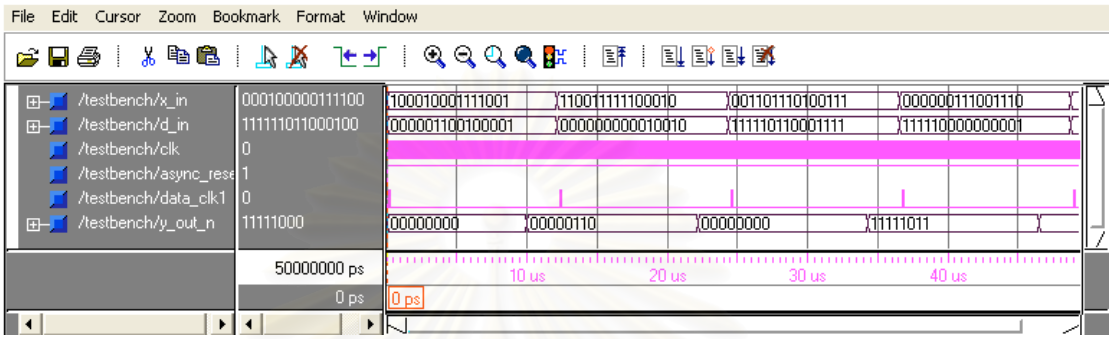


รูปที่ 4.25 ผลการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยใช้การจำลองผลการทำงานบนโปรแกรม Modelsim

จากรูปที่ 4.25 แสดงผลการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งวงจรต่าง ๆ ที่เป็นส่วนประกอบของวงจรกรองปรับตัว ทำการออกแบบวงจรด้วยภาษา VHDL รูปที่ 4.25 แสดงผลการทำงานที่ได้จากการจำลองผลบนโปรแกรม Modelsim เพื่อใช้สำหรับตรวจเช็คความถูกต้องในการทำงานของระบบในขั้นตอนต่าง ๆ ของการทำงาน นอกจากนี้ยังสามารถใช้เปรียบเทียบกับผลที่ได้จากการจำลองผลบนโปรแกรม Xilinx ISE ซึ่งในขั้นตอนแรกจะทำการเปรียบเทียบกับผลการทำงานของวงจรในระดับ Function (Simulate Behavioral VHDL Model)

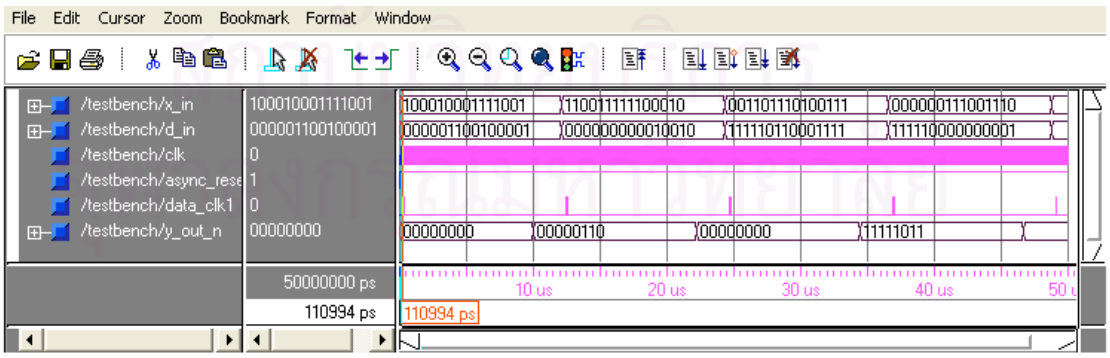
รูปที่ 4.26 แสดงผลการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยทำการจำลองผลบนโปรแกรม Xilinx ISE เพื่อตรวจสอบความถูกต้องของการทำงานในระดับ Function เทียบกับผลที่ได้จากการจำลองการทำงาน

ของวงจรกรองปรับตัวบนโปรแกรม Modelsim จากผลการจำลองการทำงานของวงจรกรองปรับตัวที่ได้ในรูปที่ 4.25 และ 4.26 เมื่อทำการตรวจสอบการทำงานของวงจรกรองปรับตัว โดยตรวจสอบค่าของสัญญาณต่าง ๆ ซึ่งจากผลที่ได้มีค่าเท่ากัน ดังนั้นสามารถสรุปได้ว่า การทำงานในระดับ Function ถูกต้อง



รูปที่ 4.26 ผลการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยใช้การจำลองผลการทำงานในระดับ Function (Simulate Behavioral VHDL Model) บนโปรแกรม Xilinx ISE

เพื่อตรวจสอบการทำงานของวงจรกรองปรับตัวที่ได้ทำการออกแบบด้วยภาษา VHDL ให้สามารถนำวงจรที่ได้ออกแบบไว้ไปใช้งานจริงบน FPGA และสามารถนำไปสร้างจริงในรูปแบบของวงจรรวม VLSI จะต้องทำการจำลองผลในระดับ Gate (Simulate Post-Place & Route VHDL Model) ซึ่งผลการจำลองการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังในระดับ Gate ได้ผลการจำลองการทำงานดังรูปที่ 4.27



รูปที่ 4.27 ผลการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยใช้การจำลองผลการทำงานในระดับ Gate (Simulate Post-Place & Route VHDL Model) บนโปรแกรม Xilinx ISE

จากผลการจำลองการทำงานในระดับ Gate ซึ่งรวมผลของ Delay Time ที่จะเกิดขึ้นในการทำงานของ Gate แต่ละตัวให้ผลลัพธ์ดังรูปที่ 4.27 เมื่อนำผลการจำลองที่ได้เทียบกับผลการจำลองในระดับ Function ปรากฏว่าผลการทำงานของวงจรถูกต้อง ดังนั้นทำให้สามารถสรุปได้ว่าวงจรกรองปรับตัวที่ทำการออกแบบวงจรด้วยภาษา VHDL เพื่อใช้แก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง สามารถนำไปสร้างและใช้งานได้จริงบน FPGA รวมถึงสามารถนำไปสร้างในรูปแบบของวงจรรวม VLSI ได้จริง

บทที่ 5

การทดสอบวงจรกรองปรับตัว

วงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังที่ได้ทำการออกแบบวงจรด้วยภาษา VHDL ในบทที่ 4 เมื่อนำเอาวงจรดังกล่าวไปทำการจำลองผลในระดับ Gate (Simulate Post-Place & Route VHDL Model) ผลการจำลองในการทำงานยังคงถูกต้องเหมือนกับการจำลองผลในระดับ Function (Simulate Behavioral VHDL Model) จากผลการจำลองดังกล่าวเป็นเครื่องยืนยันถึงผลการทำงานของวงจรกรองปรับตัวที่จะเกิดขึ้น (วงจรสามารถทำงานได้จริง) เมื่อนำเอาวงจรดังกล่าวไปทำการสร้างจริงบน FPGA หรือทำการสร้างจริงในรูปแบบของวงจรรวม VLSI

เนื่องจากการจำลองผลการทำงานของวงจรกรองปรับตัวที่ได้ทำการออกแบบวงจรด้วยภาษา VHDL ในบทที่ 4 เป็นการจำลองผลการทำงานบนคอมพิวเตอร์ ถึงแม้ว่าการจำลองผลการทำงานดังกล่าว จะเป็นเครื่องยืนยันการทำงานของวงจรว่าสามารถนำวงจรดังกล่าวไปใช้งานได้จริง รวมถึงนำวงจรดังกล่าวไปทำการสร้างจริงได้ แต่เพื่อแสดงให้เห็นถึงการทำงานของวงจรกรองปรับตัวที่จะเกิดขึ้นจริง เมื่อนำเอาวงจรดังกล่าวไปใช้งานจริง ดังนั้นเนื้อหาในบทนี้จะกล่าวถึงการทดสอบวงจรกรองปรับตัวที่ได้ทำการออกแบบไว้บน FPGA เพื่อทดสอบผลการทำงานของวงจร ซึ่งจะทำให้สามารถสรุปได้ว่า วงจรกรองปรับตัวที่ได้ทำการออกแบบด้วยภาษา VHDL สามารถทำงานได้จริง

5.1 การทดสอบวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังบน FPGA

การทดสอบการทำงานของวงจรกรองปรับตัว โดยจะทำการสร้างจริงบน FPGA เพื่อแสดงให้เห็นถึงผลการทำงานของวงจรที่จะเกิดขึ้นจริงบน FPGA ซึ่งสามารถแบ่งขั้นตอนในการทดสอบวงจรกรองปรับตัวบน FPGA ได้ดังนี้ เริ่มจากการทดสอบการทำงานของวงจรกรองปรับตัวที่ได้ทำการออกแบบด้วยภาษา VHDL บนโปรแกรม Xilinx ISE จากนั้นทำการกำหนด Input Port และ Output Port ของวงจรบน FPGA และทำการ Download วงจรที่ได้ออกแบบไว้ลงบน FPGA หลังจากนั้นทำการเลือกวิธีที่จะใช้ในการป้อนข้อมูลเข้าสู่วงจรกรองปรับตัวที่อยู่บน FPGA และทำการวัดสัญญาณ Output ต่าง ๆ บนขาของ FPGA ต่อไปจะกล่าวถึงรายละเอียดในการดำเนินการของแต่ละขั้นตอน

5.1.1 การทดสอบผลการทำงานของวงจรกรองปรับตัว โดยการจำลองผลการทำงานบนโปรแกรม Xilinx ISE

เมื่อทำการออกแบบวงจรกรองปรับตัวด้วยภาษา VHDL เสร็จเรียบร้อยแล้ว หลังจากที่ทำการจำลองผลการทำงานของวงจรบนโปรแกรม Modelsim จนได้ผลการทำงานของวงจรถูกต้อง จากนั้นนำวงจรที่ได้ออกแบบไว้ไปทำการจำลองผลการทำงานบนโปรแกรม Xilinx ISE โดยที่วงจรดังกล่าวจะต้องสามารถจำลองผลการทำงานในระดับ Gate ได้ถูกต้องจึงจะดำเนินการทดสอบในขั้นตอนต่อไปได้ อย่างไรก็ตามต้องระวังในเรื่องของจำนวนสัญญาณ Clock ที่ใช้ในการควบคุมการทำงานของระบบ เนื่องจากการจำลองผลการทำงานบนคอมพิวเตอร์จะทำการเขียนวงจรที่เรียกว่า Testbench ทดสอบการทำงานของวงจร ดังนั้นการทำงานของวงจรกรองปรับตัวที่ทำการจำลองผลทั้งในระดับ Gate และในระดับ Logic ถ้าสัญญาณ Clock ที่ป้อนให้กับวงจรกรองปรับตัวมากกว่า 1 สัญญาณ จะยังคงให้ผลการจำลองที่ถูกต้อง เนื่องจากเป็นสัญญาณที่ป้อนมาจากภายนอกและจะมีเสถียรภาพของสัญญาณสูง (เป็นไปตามค่าที่กำหนดไว้) แต่ในความเป็นจริงถ้าจำนวนของสัญญาณ Clock มากกว่า 1 สัญญาณ จะทำให้การทำงานของระบบผิดพลาดได้เมื่อนำไปใช้งานจริง เพื่อแก้ปัญหาดังกล่าว ในการออกแบบวงจรกรองปรับตัวจะรับสัญญาณ Clock จากภายนอกเพียงสัญญาณเดียวและใช้วงจรกำเนิดสัญญาณ Clock สร้างสัญญาณ Clock อื่น ๆ ที่จำเป็นต้องใช้ขึ้นมาเอง ซึ่งจะทำให้ระบบมีเสถียรภาพในการทำงาน ถ้าวงจรที่ออกแบบมีเสถียรภาพในการทำงานและผ่านการจำลองผลการทำงานในระดับ Gate เรียบร้อยแล้วสามารถดำเนินการทดสอบในขั้นต่อไปได้อย่างมีประสิทธิภาพ

5.1.2 การกำหนด Input Port และ Output Port ของวงจรกรองปรับตัวและการ Download วงจรลงบน FPGA

เมื่อระบบที่ออกแบบมีเสถียรภาพในการทำงานและผ่านการจำลองผลในระดับ Gate เรียบร้อยแล้ว ต่อจากนั้นจะทำการกำหนดขาของ Input และ Output ของวงจรกรองปรับตัวลงบน FPGA โดยจะกระทำกระบวนการดังกล่าวบนโปรแกรม Xilinx ISE ซึ่งเรียกขั้นตอนดังกล่าวว่า Back-annotate Pin Locations (ขั้นตอนในการใช้โปรแกรม Xilinx ISE ดูได้จากภาคผนวก ก) หลังจากผ่านกระบวนการดังกล่าว โปรแกรม Xilinx ISE จะกำหนดขาที่เป็น Input Port และ Output Port ลงบน FPGA เพื่อใช้สำหรับรับข้อมูลเข้าและส่งข้อมูลออกจากวงจรกรองปรับตัวที่ตามลำดับ โดยปกติแล้วขาต่าง ๆ ที่ถูกกำหนดโดยโปรแกรมจะสามารถใช้งานได้ ทั้งนี้ถ้าผู้ออกแบบต้องการเปลี่ยนตำแหน่งของขาบน FPGA ก็สามารถทำได้ โดยเข้าไปที่กระบวนการ Edit Implementation Constraints (Constraints Editor) เพื่อแก้ตำแหน่งขาที่จะใช้เป็น Input Port และ Output Port

เมื่อกำหนดตำแหน่งขาของ FPGA ที่เป็น Input Port และ Output Port ของวงจรกรองปรับตัวเรียบร้อยแล้ว จะต้องทำการ Implement Design ใหม่อีกครั้ง ในกรณีที่มีการเปลี่ยนแปลงตำแหน่งขาของ FPGA แต่ถ้าใช้ตำแหน่งขาของ FPGA ที่กำหนดโดยโปรแกรมสามารถทำขั้นตอนต่อไปได้เลย จากนั้นเตรียมบอร์ด FPGA ให้พร้อม โดยการต่อสาย Download เข้ากับคอมพิวเตอร์ เสียบไฟเข้าบอร์ด FPGA กด Switch On เนื่องจากว่าการ Download วงจรลงบน FPGA บอร์ดที่ใช้งานจะต้องมีไฟเลี้ยงตลอดเวลาวงจรที่ Download จึงจะสามารถทำงานได้ หากไฟดับก็จะต้องทำการ Download วงจรลงบนบอร์ด FPGA ใหม่วงจรดังกล่าวจึงจะสามารถทำงานได้ต่อไป

ในขั้นตอนการ Download วงจรกรองปรับตัวที่ได้ทำการออกแบบไว้ลงบอร์ด FPGA จะต้องเข้าไปที่กระบวนการ Generate Programming File จากนั้นเข้าไปที่ Properties เพื่อกำหนดคุณสมบัติของ Start-Up Clock ให้เป็นแบบ JTAG Clock เนื่องจากการ Download วงจรลงบน FPGA จะใช้สาย JTAG ในการ Download จากนั้นเข้าไปที่กระบวนการ Configure Device (iMPACT) โปรแกรม Xilinx ISE ก็จะเริ่มกระบวนการ Download วงจรลงบน FPGA ถ้าสามารถ Download วงจรลงบน FPGA ได้สำเร็จ โปรแกรมก็จะแสดงผลว่าประสบความสำเร็จในการ Download วงจรลงบน FPGA แต่ถ้าไม่สามารถ Download วงจรลงบน FPGA ได้ก็จะแสดงข้อความดังกล่าวออกมา เมื่อทำการ Download วงจรลงบน FPGA ได้สำเร็จแล้วก็จะดำเนินการในขั้นตอนต่อไป

5.1.3 การป้อนข้อมูลเข้าสู่วงจรกรองปรับตัวบน FPGA

การทดสอบการทำงานของวงจรกรองปรับตัวจะต้องป้อนข้อมูลเข้าสู่ Input Port ของวงจรกรองปรับตัวที่ผ่านการ Download วงจรดังกล่าวลงบน FPGA เรียบร้อยแล้ว ซึ่งการป้อนข้อมูลที่เป็นข้อมูลขาเข้าของวงจรกรองปรับตัวสามารถทำได้ 3 วิธี วิธีแรกโดยการป้อนแรงดันเข้าที่ขาของ FPGA ที่ทำหน้าที่เป็น Input Port ซึ่งระดับแรงดันที่ป้อนด้วยแรงดันสูงจะแทนด้วย Logic High ‘1’ ส่วนแรงดันต่ำแทนด้วย Logic Low ‘0’ แต่เนื่องจากการทดสอบด้วยวิธีการดังกล่าวไม่สามารถทำการปรับเปลี่ยนข้อมูลในแต่ละบิตที่จะป้อนเข้าสู่ Input Port ของวงจรกรองปรับตัวได้พร้อมกันจึงไม่นำวิธีการดังกล่าวมาใช้ในการทดสอบ วิธีที่สองโดยการใช้เครื่อง Pattern Generator เป็นตัวป้อนข้อมูลให้กับ Input Port ของวงจรกรองปรับตัวบน FPGA ซึ่งวิธีที่สองนี้จะต้องมีความรู้ในการใช้งานเครื่อง Pattern Generator และวิธีที่สามโดยการเขียนวงจรด้วยภาษา VHDL ให้ทำหน้าที่เป็น ROM สำหรับเก็บข้อมูลที่จะป้อนให้กับ Input Port ของวงจรกรองปรับตัวบน FPGA สำหรับการเขียนวงจรจำลองที่ทำหน้าที่เป็น ROM เมื่อจะใช้งานสามารถทำได้ 2 แบบ คือ แบบแรกทำการ Download วงจรจำลองที่ทำหน้าที่เป็น ROM ลงบน FPGA ตัวเดียวกับวงจรที่จะทำการทดสอบ

และแบบที่สองใช้บอร์ด FPGA อีกหนึ่งบอร์ดสำหรับ Download วงจรจำลองที่ทำหน้าที่เป็น ROM ลงบน FPGA ซึ่งการทำงานของทั้ง 2 แบบจะเหมือนกัน

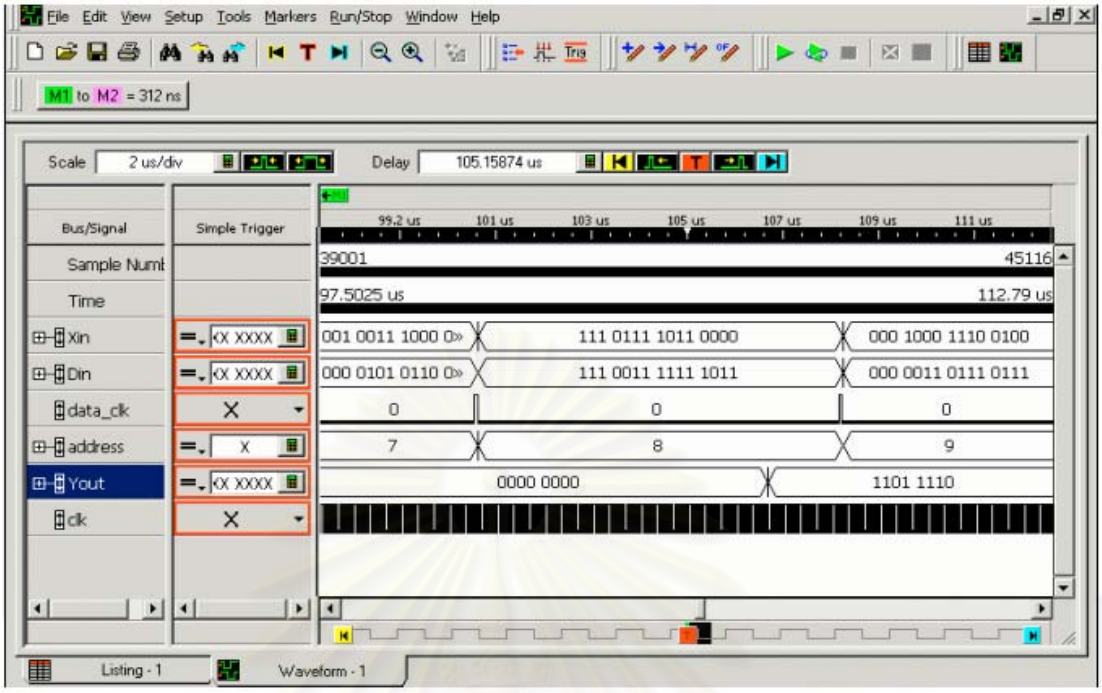
วิธีการที่นำมาใช้ในการป้อนข้อมูลเข้าที่ Input Port ของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA เพื่อทดสอบการทำงานของวงจรกรองปรับตัวสำหรับวิทยานิพนธ์นี้ จะใช้วิธีการเขียนวงจรจำลองที่ทำหน้าที่เป็น ROM ที่ใช้สำหรับป้อนข้อมูลให้กับวงจรกรองปรับตัว โดยจะใช้สัญญาณ Clock จากวงจรกำเนิดสัญญาณ Clock เป็นตัวควบคุมการเปลี่ยนแปลงข้อมูลภายในเพื่อที่จะส่งข้อมูลดังกล่าวออกมาที่ Input Port ของวงจรกรองปรับตัว ซึ่งจะรอสัญญาณ data_clk ของวงจรกำเนิดสัญญาณ Clock ในการ Load ข้อมูลต่าง ๆ เข้าสู่วงจรกรองปรับตัวต่อไป โดยจะทำการสร้างจริงวงจรกรองปรับตัวและวงจรที่ทำหน้าที่เป็น ROM ลงบน FPGA ตัวเดียวกัน

ส่วนสัญญาณ Clock ที่เป็นสัญญาณ Input ของวงจรกรองปรับตัวที่จะนำไปใช้ในการสร้างสัญญาณ Clock ต่าง ๆ เพื่อใช้ควบคุมการทำงานของวงจรกรองปรับตัว สามารถเลือกแหล่งกำเนิดสัญญาณ Clock ดังกล่าวได้ 3 แบบ คือ สร้างสัญญาณ Clock ดังกล่าวจากเครื่อง Pattern Generator หรือสร้างสัญญาณ Clock จากเครื่อง Signal Generator และใช้สัญญาณ Clock จาก Oscillator บนบอร์ด FPGA ซึ่งในการทดสอบการทำงานของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA สำหรับวิทยานิพนธ์นี้ใช้สัญญาณ Clock บนบอร์ด FPGA เป็นสัญญาณ Input ที่ป้อนให้กับวงจรกรองปรับตัว

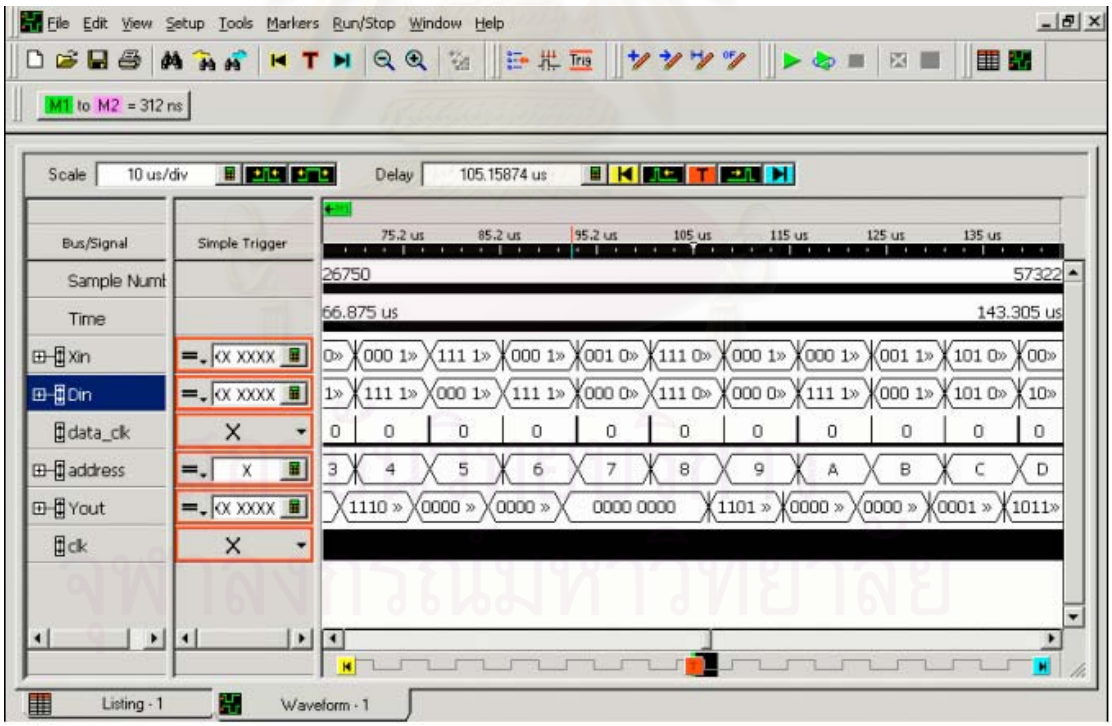
5.1.4 การตรวจสอบผลการทำงานของวงจรกรองปรับตัวบน FPGA

สำหรับการตรวจสอบผลการทำงานของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA สามารถวัดผลการทำงานของวงจรดังกล่าวได้ โดยใช้เครื่อง Logic Analyzer สำหรับตรวจวัดสัญญาณที่ต้องการตรวจสอบ ซึ่งตัวอย่างของสัญญาณที่วัดได้จริงในการทดสอบวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA แสดงให้เห็นดังรูปที่ 5.1 และรูปที่ 5.2

รูปที่ 5.1 แสดงผลการทำงานของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA โดยใช้เครื่อง Logic Analyzer วัดสัญญาณที่ขาของ FPGA ที่ทำหน้าที่เป็น Input Port และ Output Port ให้กับวงจรกรองปรับตัว โดยที่จะแสดงผลในรูปแบบของสัญญาณ ส่วนรูปที่ 5.2 แสดงผลการทำงานของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA โดยใช้เครื่อง Logic Analyzer วัดสัญญาณที่ขาของ FPGA ที่ทำหน้าที่เป็น Input Port และ Output Port ให้กับวงจรกรองปรับตัว โดยที่จะแสดงผลในรูปแบบของรายละเอียดของสัญญาณต่าง ๆ ที่ต้องการวัด

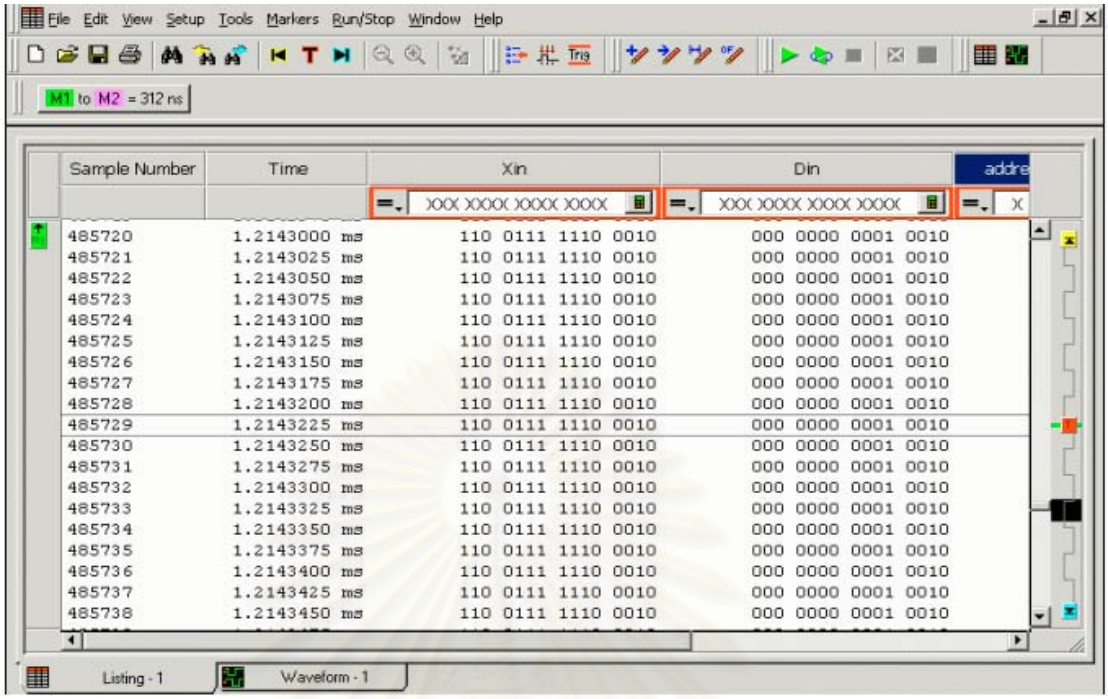


(ก)

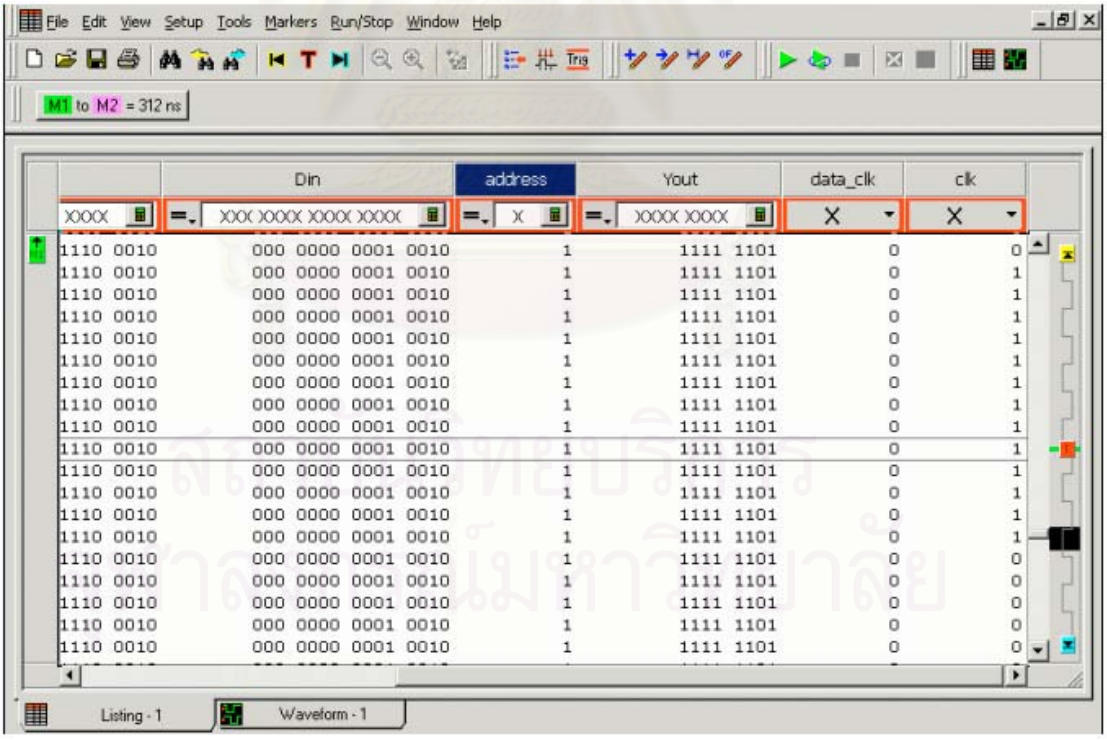


(ข)

รูปที่ 5.1 ผลการทำงานของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA โดยแสดงรูปสัญญาณที่วัดได้จากเครื่อง Logic Analyzer (ก) Zoom In (ข) Zoom Out



(ก)



(ข)

รูปที่ 5.2 ผลการทำงานของวงจรกรองปรับตัวที่ทำการสร้างจริงบน FPGA โดยแสดงรายละเอียดของข้อมูลที่วัดได้จากเครื่อง Logic Analyzer (ก) ส่วนที่ 1 (ข) ส่วนที่ 2

5.2 สรุปผลการทดสอบวงจรกรองปรับตัวบน FPGA

จากการทดสอบวงจรกรองปรับตัวที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟังที่ได้ทำการออกแบบด้วยภาษา VHDL และนำมาสร้างจริงบน FPGA จากผลการทดสอบซึ่งวัดผลการทำงานโดยใช้เครื่อง Logic Analyzer สามารถสรุปได้ว่า วงจรกรองปรับตัวที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง โดยนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign และอาศัยการปรับช่วงก้าวแบบพลวัตซึ่งใช้วิธีการปรับช่วงก้าวแบบค่าผิดพลาดกำลังสองมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัว สามารถทำงานได้จริงบน FPGA และสามารถทำการสร้างจริงในรูปแบบของวงจรรวม VLSI ได้

บทที่ 6

บทสรุป

6.1 สรุปผลการวิจัย

วิทยานิพนธ์นี้นำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาประยุกต์ใช้ในการปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอนในเครื่องช่วยฟัง เพื่อลดความซับซ้อนของขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด โดยใช้การปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสองมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธี ทำให้สามารถลดความซับซ้อนของฮาร์ดแวร์ ทั้งนี้จะส่งผลให้สามารถประหยัดพื้นที่ในการสร้างจริงซึ่งจะทำให้ขนาดของเครื่องช่วยฟังมีขนาดเล็กกว่าการนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดมาทำการสร้างจริง และยังส่งผลต่อการสูญเสียพลังงานที่จะลดลงจากการลดความซับซ้อนของขั้นตอนวิธี ซึ่งจะทำให้อายุการใช้งานของแบตเตอรี่ในเครื่องช่วยฟังมีอายุการทำงานที่ยาวนานขึ้น และจะทำให้ผู้ใช้เครื่องช่วยฟังไม่ต้องเปลี่ยนแบตเตอรี่บ่อย ๆ

การสร้างจริงวงจรต่าง ๆ ที่เป็นส่วนประกอบของวงจรกรองปรับตัว เมื่อนำมาประกอบรวมกันสามารถทำงานได้อย่างมีประสิทธิภาพ เนื่องจากในการออกแบบวงจรดังกล่าวด้วยภาษา VHDL ได้ทำการทดสอบวงจรต่าง ๆ ที่เป็นส่วนประกอบของวงจรกรองปรับตัวให้สามารถทำงานได้จริงก่อนนำมาใช้งาน นอกจากนี้ระบบในการทำงานของวงจรกรองปรับตัวยังถูกออกแบบมาให้สามารถใช้งานได้จริงและมีเสถียรภาพในการทำงาน เนื่องจากใช้สัญญาณ Clock จากภายนอกมาควบคุมการทำงานเพียงสัญญาณเดียว จากผลการทดสอบการทำงานของวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน สามารถสรุปได้ว่า วงจรกรองปรับตัวดังกล่าวที่ทำการสร้างจริงด้วยภาษา VHDL สามารถทำงานได้จริงบน FPGA และสามารถนำไปสร้างจริงในรูปแบบของวงจรรวม VLSI เพื่อนำไปใช้งานในรูปแบบของฮาร์ดแวร์ได้จริง

ข้อเสียสำหรับการนำเอาขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign มาประยุกต์ใช้ในการสร้างจริงวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง คือ จะทำให้ประสิทธิภาพในการปรับตัวของขั้นตอนวิธีลดลง เนื่องจากเป็นการประมวลผลที่ใช้ค่าน้ำหนักเป็นค่า Sign bit ของค่าความผิดพลาดและค่าของสัญญาณขาเข้าในแต่ละ Tap ไม่ใช่ค่าที่แท้จริง

อย่างไรก็ตามจากที่มาและเหตุผลที่ได้กล่าวมาแล้วข้างต้น และจากผลการจำลองการทำงานที่ผ่านมา วงจรกรองปรับตัวที่ใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ที่นำเอาวิธีการปรับค่าช่วงก้าวแบบค่าความผิดพลาดกำลังสองมาช่วยปรับปรุงประสิทธิภาพของขั้นตอนวิธีมีประสิทธิภาพในการทำงานที่ดี เหมาะที่จะนำไปใช้ในการสร้างจริงมากกว่าขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุด

6.2 ข้อเสนอแนะสำหรับการวิจัยในอนาคต

สำหรับงานที่ควรได้รับการศึกษาหรือพัฒนาต่อไปในอนาคต

1. ศึกษาการเขียนโปรแกรมภาษา VHDL เพิ่มเติม เพื่อที่จะได้สามารถนำเอาเทคนิคต่าง ๆ ที่ได้เรียนรู้ มาประยุกต์ใช้ในการออกแบบวงจรที่จะทำการสร้างจริงได้อย่างมีประสิทธิภาพ

2. ศึกษาการใช้โปรแกรม Chipscope Analyzer เพื่อใช้ตรวจสอบการทำงานภายในของวงจรที่ออกแบบ ซึ่งจะทำให้สามารถตรวจเช็คจุดบกพร่องในการทำงานของวงจรได้ง่ายขึ้น

3. พยายามหาวิธีการใช้งานMultilinx Cable เพื่อความสะดวกในการออกแบบวงจรที่จะทำการสร้างจริงบน FPGA ซึ่งจะรวมถึงการ Download วงจรและการตรวจสอบการทำงานภายในของวงจร โดยจะใช้งานร่วมกับโปรแกรม Chipscope Analyzer

4. ควรจะทำความเข้าใจเกี่ยวกับระบบการทำงานของฮาร์ดแวร์ให้มากขึ้น เนื่องจากการออกแบบวงจรด้วยภาษา VHDL เป็นการสร้างฮาร์ดแวร์จากโปรแกรม ถ้าเขียนในลักษณะซอฟต์แวร์มากเกินไปวงจรที่ออกแบบจะไม่สามารถทำงานได้จริง

รายการอ้างอิง

1. Widrow, B. and Stearns, S. D. Adaptive Signal Processing. Prentice Hall, 1985.
2. Haykin, S. Adaptive Filter Theory. Prentice Hall, 1996.
3. Yalamanchili, S. VHDL Starter's Guide. Prentice Hall, 1998.
4. Bustamante, T. L., Worrall, D. K. and Williamson, M. J. Measurement and adaptive suppression of acoustic feedback in hearing aids. IEEE ICASSP No. 3 (1989) : 2017-2020.
5. Kates, J. M. Feedback cancellation in hearing aids. IEEE Trans. on Signal Processing Vol. 39 No. 3 (March 1991) : 553-562.
6. Maxwell, J. A. and Zurek, P. M. Reducing acoustic feedback in hearing aids. IEEE Trans. on Speech and Audio Processing Vol. 3 No. 4 (July 1995) : 304-313.
7. Wang, R. and Harjani, R. Acoustic feedback cancellation in hearing aids. IEEE ICASSP Vol. 1 (1993) : 137- 140.
8. Engebretson, M. P., O' Connell, A. M. and Gong, F. An adaptive feedback Equalization algorithm for digital hearing aids. Proc. IEEE Ann. Int. Conf. Eng. Med. And Biol. Soc. Vol. 12 (1990) : 2286- 2289.
9. Estermann, P. and Kaelin, A. Feedback cancellation in hearing aids. IEEE ISCAS Vol. 2 (1994) : 257-260.
10. Nicol, C. J., Larsson, P., Azadet, K., and O'Neill, J. H. A Low-power 128-tap digital adaptive equalizer for broadband modems. IEEE Journal of Solid-State Circuits Vol. 32 No. 11 (November 1997): 1777-1789.
11. Muhammad, K., Staszewski, R. B., and Balsara, P. T. Speed, power, area, and latency tradeoffs in adaptive FIR filtering for PRML read channels. IEEE Transactions on very large scale integration(VLSI) systems Vol. 9 No. 1(February 2001): 42-51.
12. Hwang, R. H. and Johnston, E. W. A variable step size LMS adaptation. IEEE Trans. on Signal Processing Vol.40 No. 7 (July 1992) :1633-1642.
13. Honig, M.L. and Messerschmitt, D.G. Adaptive filters: Structures, Algorithms and Applications (n.p.). Kluwer Academic Publishers, 1988.
14. Karni, S. and Zeng, G. A new convergence factor for adaptive filters. IEEE Trans. on Circuits and Systems Vol. 36 No. 7 (July 1989) : 1011-1012.

15. Jacobs, R.A. Increased rates of convergence through learning rate adaptation. Neural networks 1 (1988) : 295-307.

16. Thippayathetthana, S., and Chinrungrueng, C. Variable step size of the lms algorithm for reducing acoustic feedback in hearing aids. IEEE APCCAS (2000) : 407-410.

17. Oppenheim, A. V., Schafer R. W., and Buck, J. R. Discrete-time signal processing. Prentice Hall, 1999.

18. Brown, S., and Vranesic, Z. Fundamentals of Digital Logic with VHDL Design. Mcgraw Hill, 2000.

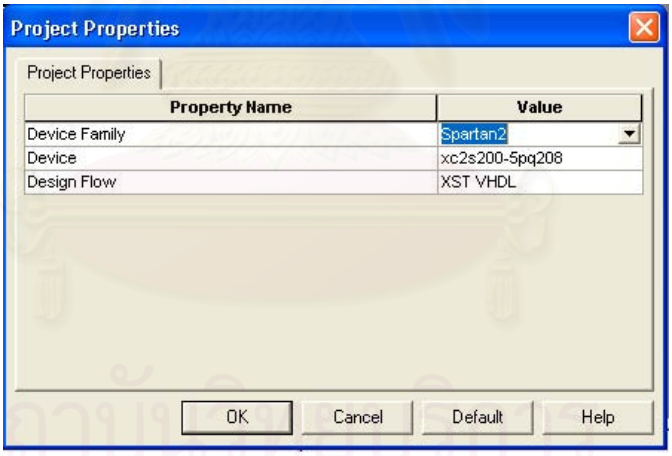
ภาคผนวก

ภาคผนวก ก

ขั้นตอนการใช้โปรแกรม Xilinx ISE ในการสร้างจริงวงจรกรองปรับตัว

ขั้นตอนในการออกแบบวงจรกรองปรับตัวที่ใช้สำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงในเครื่องช่วยฟัง ซึ่งจะทำการสร้างจริงวงจรดังกล่าวบน FPGA โดยใช้โปรแกรม Xilinx ISE ทำการ Synthesis และ Implement ดังนั้นเพื่อแสดงให้เห็นถึงขั้นตอนต่าง ๆ ของการออกแบบวงจรกรองปรับตัวที่ใช้โปรแกรม Xilinx ISE ในการสร้างจริงบน FPGA จะอธิบายลำดับขั้นตอนต่าง ๆ ที่ใช้งานจริงในการออกแบบวงจรกรองปรับตัวดังกล่าว

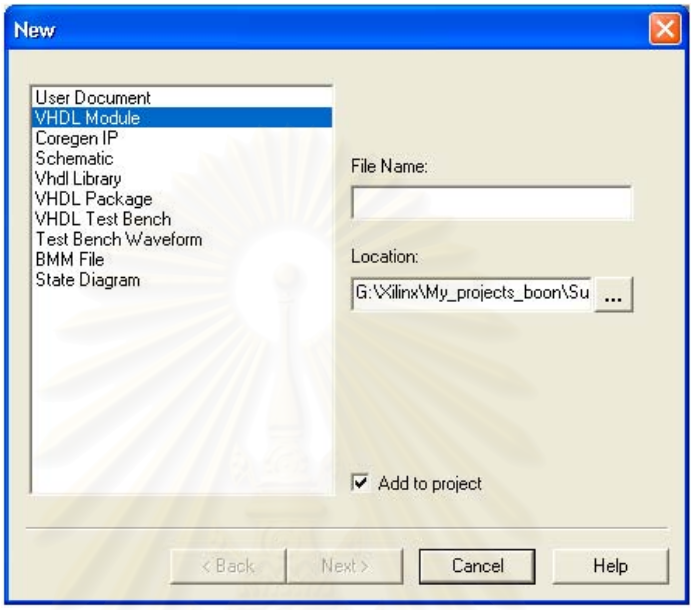
เริ่มต้นจากการเปิดโปรแกรม Xilinx ISE จากนั้นเข้าไปที่ File เลือก New Project และทำการกำหนดคุณสมบัติของตัว FPGA รวมถึงภาษาที่จะใช้ในการออกแบบ เนื่องจากการสร้างจริงวงจรกรองปรับตัวจะใช้บอร์ด FPGA ของบริษัท Xilinx รุ่น Spartan2 xc2s200-5pq208 และทำการออกแบบวงจรด้วยภาษา VHDL ดังนั้นจะใส่คุณสมบัติต่าง ๆ เหล่านี้ลงในหน้าต่างดังรูปที่ ก-1



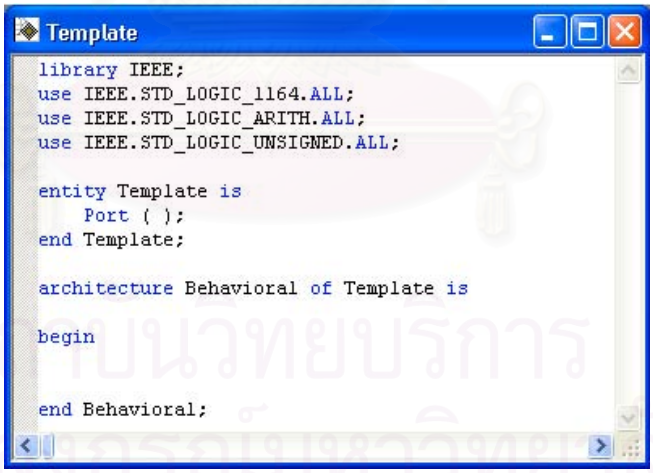
รูปที่ ก.1 หน้าต่างกำหนดคุณสมบัติของบอร์ด FPGA

จากนั้นเข้าไปที่ Project ที่อยู่บนหน้าต่างหลักของโปรแกรมและเลือก New Source จะปรากฏหน้าต่างดังรูปที่ ก-2 ซึ่งจะใช้สำหรับสร้าง Source Code ของวงจร โดยที่การสร้างจริงจะทำกระบวนดังกล่าว 2 ครั้ง ครั้งแรกสร้างส่วนที่เป็นวงจรจะต้องเลือกเข้าไปสร้างที่ VHDL Module จากนั้นใส่ชื่อในช่อง File Name (ชื่ออะไรก็ได้) สำหรับช่อง Location จะใช้ที่อยู่ที่จะเก็บวงจรที่ออกแบบด้วยภาษา VHDL เมื่อทำขั้นตอนดังกล่าวครบแล้วให้กด Next จนกระทั่งโปรแกรมสร้าง Template ที่จะให้ใช้สำหรับเขียนวงจรที่ต้องการออกแบบด้วยภาษา VHDL ดังแสดงในรูปที่ ก-3

ส่วนในครั้งที่สองจะทำการสร้าง Source Code ของตัว Testbench ซึ่งขั้นตอนนี้จะต่างจากครั้งแรก โดยที่จะต้องเปลี่ยนจาก VHDL Module ไปเป็น VHDL Test Bench ส่วนการตั้งชื่อ File แล้วแต่จะกำหนด ส่วน Location ใช้ที่เดียวกับวงจร จากนั้นทำกระบวนการต่าง ๆ เหมือนกับการสร้างวงจร

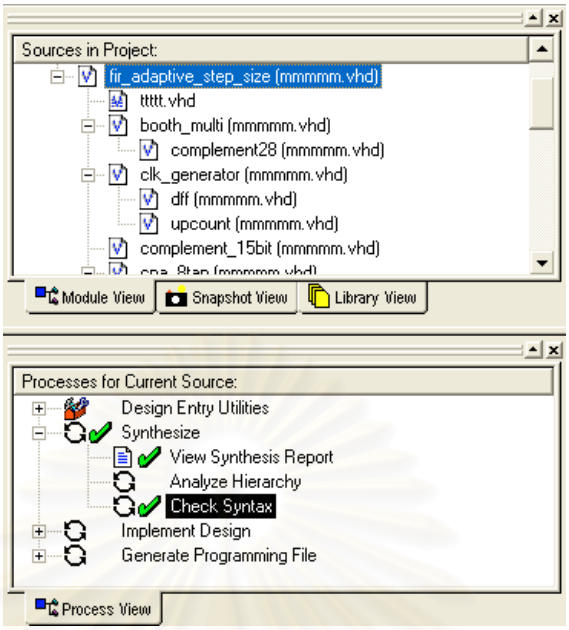


รูปที่ ก.2 หน้าต่างที่ใช้สำหรับสร้างวงจรและ Testbench



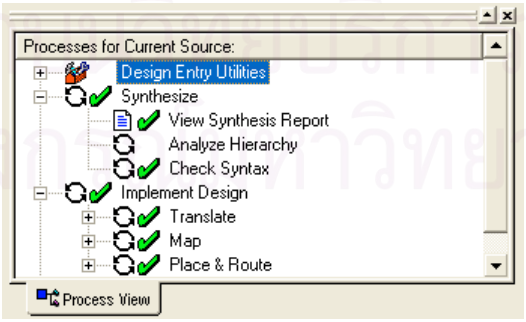
รูปที่ ก.3 Template ที่สร้างจากโปรแกรม

เมื่อได้ Template ของวงจรและตัว Testbench เรียบร้อยแล้ว จากนั้นทำการ Copy วงจรที่ทำการออกแบบไว้มาใส่ไว้ในตัววงจร ส่วนตัว Testbench ให้ทำการ Copy ใส่ไว้ในตัว Testbench (ถ้าต้องการจะออกแบบวงจรบน Template ที่โปรแกรมสร้างขึ้นมาก็ได้) เมื่อเสร็จสิ้นกระบวนการดังกล่าวแล้ว ต่อไปจะทำการ Synthesis วงจรที่ออกแบบด้วยภาษา VHDL ซึ่งจะแสดงหน้าต่างที่ใช้สำหรับการทำงานดังรูปที่ ก-4



รูปที่ ก.4 หน้าต่างของโปรแกรม Xilinx ISE ที่ใช้ในการสร้างจริง

การ Synthesis วงจรที่ออกแบบด้วยภาษา VHDL ต้องคลิกเมาส์เข้าไปที่ Synthesize ในหน้าต่าง Processes for Current Source ซึ่งจะตรวจสอบความถูกต้องของ Syntax ต่าง ๆ ที่ใช้ในการออกแบบวงจรกรองปรับตัว โดยที่ในหน้าต่างของ Sources in Project ต้องเลือกอยู่ที่วงจรที่ต้องการ Synthesis เมื่อผ่านขั้นตอนดังกล่าวแล้วบนหน้าต่างจะปรากฏเครื่องหมายถูกสีเขียวที่ตำแหน่ง Synthesize ต่อจากนั้นก็จะทำการ Implement Design โดยใช้เมาส์คลิกเข้าไปที่ Implement Design โปรแกรมจะทำกระบวนการ Translate ต่อด้วยกระบวนการ Map และกระบวนการ Place & Route ถ้าสามารถทำกระบวนเหล่านี้สำเร็จก็จะปรากฏเครื่องถูกสีเขียวหน้ากระบวนเหล่านี้ดังรูปที่ ก.5 แต่ถ้าขึ้นเครื่องหมายกากบาทสีแดง แสดงว่าไม่สามารถ Implement วงจรดังกล่าวที่ออกแบบได้



รูปที่ ก.5 หน้าต่างแสดงลักษณะของวงจรที่ผ่านการ Implement

หลังจากที่สั่งให้โปรแกรม Xilinx ISE เข้าไปทำงานในกระบวนการ Implement Design ผลที่ได้จากขั้นตอนนี้ สามารถที่จะดูจำนวนของทรัพยากรต่าง ๆ ที่ถูกใช้ไป (ดูจาก Map Report)ใน

การ Implement วงจรดังกล่าวลงบน FPGA ซึ่งจากการสร้างจริงวงจรกรองปรับตัวสามารถแสดงรายละเอียดได้ดังนี้

Design Summary

Number of errors : 0

Number of warnings : 2

Number of Slices : 1,875 out of 2,352 79%

Number of Slices containing unrelated logic: 0 out of 1,875 0%

Total Number Slice Registers : 1,391 out of 4,704 29%

Number used as Flip Flops : 1,378

Number used as Latches : 13

Total Number 4 input LUTs : 2,874 out of 4,704 61%

Number used as LUTs : 2,870

Number used as a route-thru : 4

Number of bonded IOBs : 40 out of 140 28%

IOB Flip Flops : 23

IOB Latches : 1

Number of GCLKs : 1 out of 4 25%

Number of GCLKIOBs : 1 out of 4 25%

Total equivalent gate count for design : 29,239

Additional JTAG gate count for IOBs : 1,968

จากรายละเอียดที่แสดงสิ่งที่สำคัญที่ต้องพิจารณา คือ การใช้งานทรัพยากรต่าง ๆ บน FPGA จะต้องไม่มากเกินจำนวนที่มีอยู่ หมายความว่าไม่เกินร้อยละ 100 ของทรัพยากรที่มีอยู่มิฉะนั้นจะไม่สามารถทำงานบน FPGA ตัวที่กำลังใช้งานอยู่ได้ นอกจากสิ่งที่จำเป็นต้องดูอีกอย่าง คือ จำนวน Gate ทั้งหมดที่ใช้ในการสร้างจริงวงจรกรองปรับตัว สำหรับจำนวน Gate ที่ใช้ในการสร้างจริงวงจรต่าง ๆ สามารถสรุปได้ดังนี้

Register ขนาด 1 บิต ใช้จำนวน Gate เท่ากับ 8 Gate

Register ขนาด 8 บิต ใช้จำนวน Gate เท่ากับ 64 Gate

Register ขนาด 15 บิต ใช้จำนวน Gate เท่ากับ 120 Gate

วงจร 2's Complement ขนาด 8 บิต ใช้จำนวน Gate เท่ากับ 54 Gate

วงจร 2's Complement ขนาด 15 บิต ใช้จำนวน Gate เท่ากับ 120 Gate

วงจร 2's Complement ขนาด 29 บิต ใช้จำนวน Gate เท่ากับ 246 Gate

วงจรบวกแบบ Carry Propagate Adder ขนาด 15 บิต ใช้จำนวน Gate เท่ากับ 168 Gate

วงจรบวกแบบ Carry Propagate Adder ขนาด 22 บิต ใช้จำนวน Gate เท่ากับ 252 Gate

วงจรบวกแบบ Carry Propagate Adder ขนาด 32 บิต ใช้จำนวน Gate เท่ากับ 372 Gate

วงจรบวกแบบ Carry Save Adder ขนาด 32 บิต ใช้จำนวน Gate เท่ากับ 378 Gate

วงจรกำเนิดสัญญาณนาฬิกา (Clock Generator) ใช้จำนวน Gate เท่ากับ 1,054 Gate

วงจรนับ (Counter) ใช้จำนวน Gate เท่ากับ 130 Gate

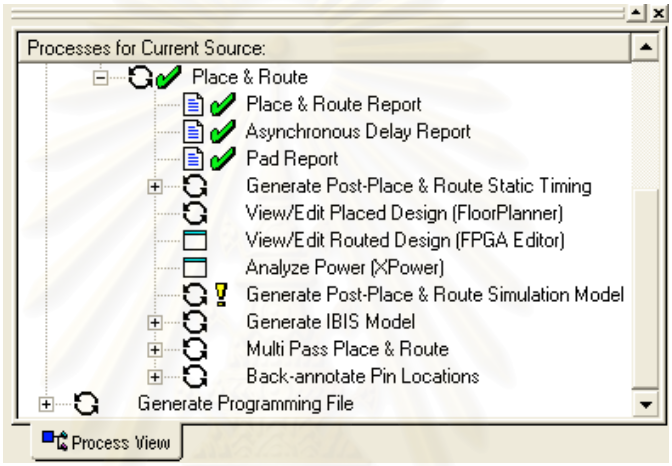
วงจรคูณ Booth Encode ใช้จำนวน Gate เท่ากับ 2,122 Gate

วงจรคูณแบบ Array ใช้จำนวน Gate เท่ากับ 1,230 Gate

วงจรปรับค่าช่วงก้าวแบบค่าผิดพลาดกำลังสองใช้จำนวน Gate เท่ากับ 1,814 Gate และ

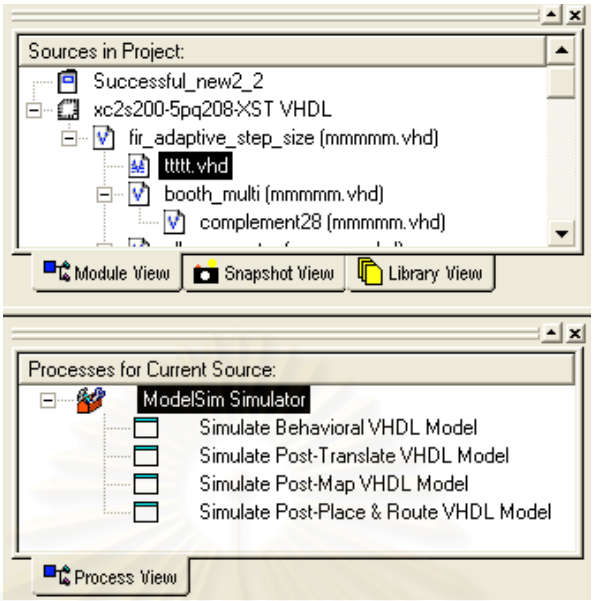
วงจรปรับค่าสัมประสิทธิ์ของวงจรกรองปรับตัวที่ใช้ขั้นตอนวิธีกำลังสองเฉลี่ยน้อยสุดประเภท Sign-Sign ใช้จำนวน Gate เท่ากับ 543 Gate

ต่อจากนั้นคลิกเมาส์เข้าไปที่ Place & Route หลังจากนั้นคลิกเมาส์เข้าไปที่ Generate Post-Place & Route Simulation Model เมื่อทำงานที่กระบวนการดังกล่าวเรียบร้อยแล้วจะปรากฏเครื่องหมายตกใจขึ้นมาหน้ากระบวนการดังกล่าวดังแสดงในรูปที่ ก.6 ซึ่งในกระบวนการนี้โปรแกรม Xilinx ISE จะสร้างไฟล์ที่สำคัญขึ้นมา 2 ตัวดังนี้คือ ตัวแรกชื่อ File_name_timesim.sdf และตัวที่สองชื่อ File_name_timesim.vhd ซึ่งไฟล์ทั้งสองนี้จะเป็นไฟล์ที่สำคัญมากที่จะนำไปใช้ในการจำลองผลการทำงาน เพราะในการจำลองผลการทำงานในระดับ Gate จะต้องนำผลของ Delay Time ต่าง ๆ ที่จะเกิดขึ้นของ Gate แต่ละตัวมาประเมินผลในการจำลองผลการทำงาน



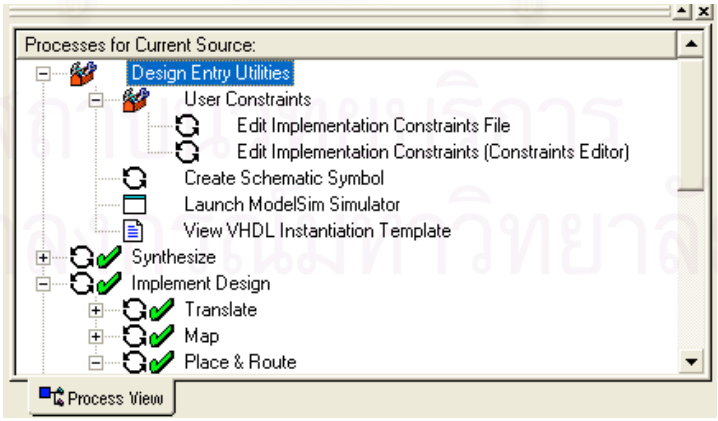
รูปที่ ก.6 หน้าต่างแสดงผลที่ได้จากการทำขั้นตอน Generate Post-Place & Route Simulation Model

หลังจากนั้นก็จะทดสอบวงจรกรองปรับตัวที่ทำการออกแบบด้วยภาษา VHDL โดยจะทำการจำลองผลการทำงานของวงจรในระดับ Function (Simulate Behavioral VHDL Model) และในระดับ Gate (Simulate Post-Place & Route VHDL Model) โดยที่ในกรอบหน้าต่างของ Sources in Project จะต้องคลิกเมาส์เลือกอยู่ที่ตำแหน่งของไฟล์ที่เป็นตัว Testbench จากนั้นทำการจำลองผลทั้งในระดับ Function และระดับ Gate โดยเลือกในกรอบหน้าต่าง Processes for Current Source ดังแสดงในรูปที่ ก.7 ต่อจากนั้นพิจารณาผลที่ได้จากการจำลองผลการทำงาน โดยจะเริ่มพิจารณาการจำลองผลในระดับ Fucntion ก่อนเป็นอันดับแรก ซึ่งการทำงานจะต้องถูกต้องก่อนที่จะไปทำการจำลองการทำงานของวงจรในระดับ Gate ต่อไป ซึ่งถ้าหากการจำลองผลในระดับ Gate ถูกต้องก็สามารถนำวงจรดังกล่าวไปทำการทดสอบจริงบน FPGA ได้ แต่ถ้าผลการจำลองการทำงานปรากฏว่าวงจรไม่สามารถทำงานได้ ก็ต้องทำการแก้ไขและเริ่มต้นกระบวนการใหม่



รูปที่ ก.7 หน้าต่างแสดงส่วนที่ใช้ในการจำลองผลทั้งในระดับ Function และระดับ Gate

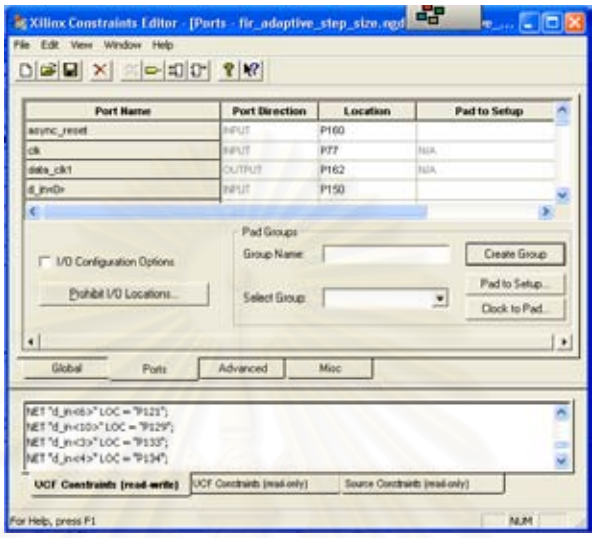
หลังจากที่วงจรกรองปรับตัวสามารถจำลองผลการทำงานในระดับ Gate ได้ถูกต้อง ต่อจากนั้นก็ทำการกำหนดขาของ FPGAให้กับวงจรกรองปรับตัวที่จะทำการ Download วงจรลงบน FPGA โดยคลิกเมาส์เข้าไปที่ Place & Route และเลือกคลิกเมาส์ไปที่ Back-annotate Pin Locations ดังรูปที่ ก.6 โปรแกรม Xilinx ISE จะทำการกำหนดขาของวงจรกรองปรับตัวลงบน FPGA ให้ (จะได้ไฟล์ชื่อ File_name.UCF) เพื่อนำไปใช้สำหรับป้อนสัญญาณเข้าและวัดสัญญาณออกจาก FPGA ซึ่งการกำหนดขาของโปรแกรมสามารถใช้งานได้ แต่ถ้าต้องการเปลี่ยนตำแหน่งขาของ FPGA ที่จะนำมาใช้เป็น Input Port และ Output Port ให้กับวงจรกรองปรับตัวก็สามารถทำได้



รูปที่ ก.8 หน้าต่างแสดงกระบวนที่ใช้ในการเปลี่ยนตำแหน่งขาบน FPGA ที่จะใช้ในการสร้างจริง

การเปลี่ยนตำแหน่งขาของ FPGA ที่จะนำมาใช้เป็น Input Port และ Output Port ทำได้โดยการคลิกเมาส์เข้าไปที่ User Constraints และเลือกคลิกเมาส์เข้าไปที่ Edit Implementation

Constraints (Constraints Editor) จะปรากฏหน้าต่างดังรูปที่ ก.9 ถ้าต้องการแก้ตำแหน่งขา FPGA ที่นำมาใช้เป็น Input Port และ Output Port ของวงจรกรองปรับตัวก็สามารถทำได้ แต่ถ้าไม่ต้องการแก้ตำแหน่งขาต่าง ๆ จะเป็นไปตามที่โปรแกรมกำหนดที่แสดงรายละเอียดอยู่ใน File_name.UCF



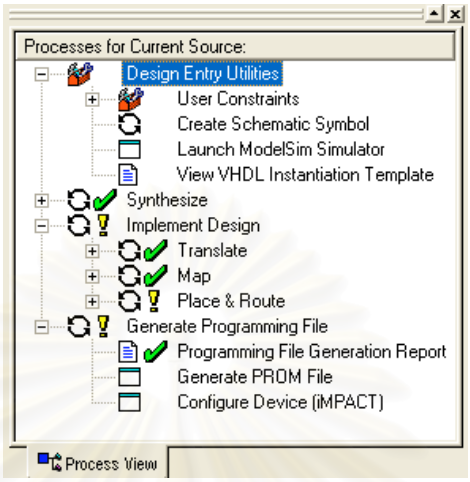
รูปที่ ก.9 หน้าต่างที่ใช้สำหรับแก้ตำแหน่งขาของ FPGA ที่จะนำไปใช้งานจริง

ตัวอย่างรายละเอียดที่ปรากฏใน File_name.UCF ไฟล์มีดังนี้

```
#### UCF file created by Project Navigator

#PINLOCK_BEGIN

#Wed Apr 02 00:37:05 2003

NET "x_in<6>"       LOC =  "P14";

NET "x_in<7>"       LOC =  "P10";
                    |
                    |
                    |
NET "d_in<0>"       LOC =  "P150";

#PINLOCK_END
```

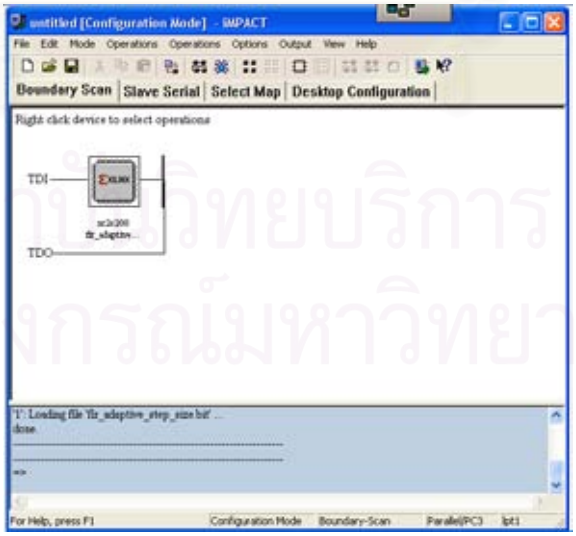
เมื่อทำกระบวนการต่าง ๆ เรียบร้อยแล้ว ต่อไปจะเป็นกระบวนการในการ Download วงจรกรองปรับตัวที่ได้ออกแบบไว้ลงบน FPGA โดยเริ่มจากคลิกขวาที่เมาส์และเลือกไปที่ Properties

ซึ่งจะทำการกำหนด Start-Up Clock เป็น JTAG Clock เนื่องจากใช้สาย Download แบบขนาน (JTAG) ในการ Download วงจรลงบน FPGA



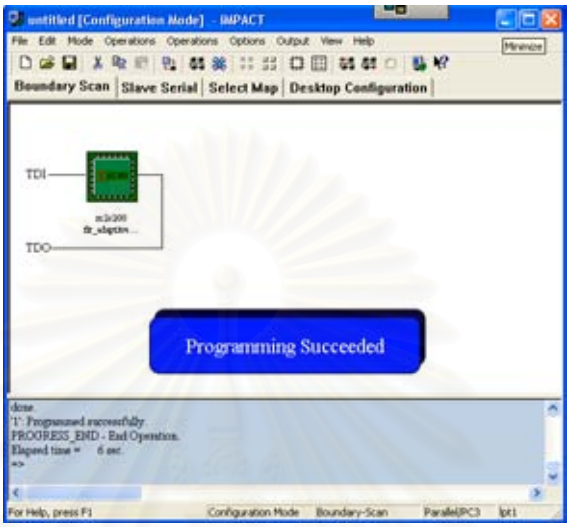
รูปที่ ก.10 หน้าต่างแสดงการ Download วงจรที่ออกแบบไว้ลงบน FPGA

ขั้นตอนต่อไปจะเป็นขั้นตอนในการ Download วงจรกรองปรับตัวลงบน FPGA ซึ่งจะต้องต่อบอร์ด FPGA เข้ากับคอมพิวเตอร์โดยผ่านสาย JTAG และป้อนไฟให้กับบอร์ด FPGA ให้เรียบร้อย ต่อจากนั้นเลื่อนเมาส์ไปคลิกที่ Configure Device (iMPACT) (อยู่ในรูปที่ ก.10) ขั้นตอนดังกล่าวจะเป็นการตรวจสอบการเชื่อมต่อของระบบ ถ้าการเชื่อมต่อเสร็จสมบูรณ์แล้วจะแสดงผลดังรูปที่ ก.11



รูปที่ ก.11หน้าต่างแสดงความพร้อมในการ Download วงจรลงบน FPGA

เมื่อระบบพร้อมที่จะทำการ Download วงจรกรองปรับตัวลงบน FPGA จากนั้นเลื่อนเมาส์ไปคลิกที่ตัว FPGA ซึ่งอยู่บนรูปที่ ก.11 และทำการโปรแกรมวงจรกรองปรับตัวลงบน FPGA เมื่อการโปรแกรมวงจรกรองปรับตัวลงบน FPGA เสร็จสมบูรณ์แล้วจะได้ผลลัพธ์ดังรูปที่ ก.12



รูปที่ ก.12 หน้าต่างแสดงความสำเร็จในการโปรแกรมวงจรลงบน FPGA

เมื่อทำการโปรแกรมวงจรกรองปรับตัวสำหรับแก้ปัญหาการเกิดเสียงหอน เนื่องจากการป้อนกลับทางเสียงลงบน FPGA เรียบร้อยแล้ว สามารถนำเอาวงจรที่อยู่บน FPGA ไปใช้งานได้โดยการป้อนสัญญาณขาเข้าไปที่ขาของ FPGA ที่เป็น Input Port ของวงจรกรองปรับตัว และทำการวัดสัญญาณขาออกที่ขาของ FPGA ที่เป็น Output Port ของวงจรกรองปรับตัว

ประวัติผู้เขียนวิทยานิพนธ์

นายสมบูรณ์ กลิ่นจันทร์กลั่น เกิดวันที่ 8 พฤศจิกายน พ.ศ. 2517 ที่กรุงเทพมหานคร สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าฯ เจ้าคุณทหารลาดกระบัง ในปีการศึกษา 2540 หลังจากสำเร็จการศึกษาในระดับปริญญาตรีได้เข้ารับราชการที่สำนักงานพลังงานปรมาณูเพื่อสันติ และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต ภาควิชาวิศวกรรมไฟฟ้า สาขาวิศวกรรมไฟฟ้า ที่จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2543