โครงงานวิศวกรรมโทรคมนาคม

เรื่อง: การต่อวัดสัญญาณด้วยกล้องโทรทัศน์ชนิด ที.1120 (Like Giant)

ผู้ดำเนินงาน: นายสาวลศิริ อนุรักษ์ปัญญา (B3904602)

อาจารย์ที่ปรึกษา: รองศาสตราจารย์ ดร. สราวุฒิ สุจิตรา

อาจารย์ยงศักดิ์ วาณิชอนันต์ชัย

สาขาวิศวกรรมโทรคมนาคม

ภาคการศึกษาที่ 3/2542

รายงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาตรีสาขาวิชาวิศวกรรมโทรคมนาคม

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี พ.ศ. 2543

ลิขสิทธิ์เป็นของสำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

โครงงานวิศวกรรมโทรคมนาคม

เรื่อง: การอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

ผู้ดำเนินงาน: นางสาวศศิธร อนุรักษ์ปัญญา (B3904602)

อาจารย์ที่ปรึกษา: รองศาสตราจารย์ ดร. สราวุฒิ สุจิตจร

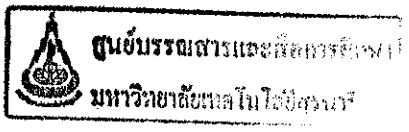
อาจารย์สมศักดิ์ วาณิชอนันต์ชัย

สาขาวิศวกรรมโทรคมนาคม

ภาคการศึกษาที่ 3/2542

รายงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาตรีสาขาวิชาวิศวกรรมโทรคมนาคม

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี พ.ศ. 2542



Telecommunication Engineering Project

" Compensator Implementation by BL1120 Controller Board (Little Giant)"

Author: Miss Sasithorn ANURAKPANYA (B3904602)

Adviser: Assoc. Prof. Sarawut SUJITJORN

Lect. Somsak WANICH-ANAN-CHAI

School of Telecommunication Engineering

Trimester 3/1999

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR

THE DEGREE OF BACHELOR OF ENGINEERING IN TELECOMMUNICATION ENGINEERING

THE INSTITUTE OF ENGINEERING, SURANAREE UNIVERSITY OF TECHNOLOGY, 1999

บทคัดย่อ

โครงงานวิศวกรรมโทรคมนาคมนี้ ศึกษาเกี่ยวกับการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) โดยมีการประยุกต์ทฤษฎีการแปลงทรานสเฟอร์ฟังก์ชันจากรูป s-domain ไปเป็น z-domain ด้วยเทคนิคของทุสติน เพื่อเตรียมทรานสเฟอร์ฟังก์ชันใน z-domain มาทำการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง โครงสร้างของตัวชดเชยที่อนุวัตขึ้นจะเหมือนกับโครงสร้างของตัวกรองความถี่แบบดิจิตอลชนิด IIR

เทคนิคการแปลงแบบทุสตินนั้นจะต้องมีการปรับ dc gain ของ H(s) ให้เท่ากับ H(z) และการปรับแก้อาการโก่งทางความถี่ (frequency prewarping) เพิ่มเติมด้วยในกรณีที่จำเป็น แล้วจะทำให้ค่าอัตราขยายและเฟส ของ H(s) ใกล้เคียงกับ H(z) มากยิ่งขึ้น โดยมีผลการทดสอบเป็นสิ่งยืนยัน

ABSTRACT

This Telecommunication Engineering Project is concerned with the implementation of digital compensator using the BL1120 (Little Giant) controller board. Preparation of H(z) for the first and second order compensator implementation utilizes transformation from H(s) to H(z) based on Tustin's approximation. Also, Structure of compensator is similar to IIR digital filter.

Dc gain adaptation and frequency prewarping may be added for Tustin's transformation because they make gain and phase in s-domain close to the ones in z-domain. Compensator testing shows this effects.

กิตติกรรมประกาศ

ผู้เขียนขอขอบพระคุณบุพการีที่อบรมสั่งสอนและเลี้ยงดู รวมทั้งให้การสนับสนุนอย่างดีมาโดยตลอด

ขอขอบพระคุณ รองศาสตราจารย์ ดร. สราวุฒิ สุจิตจร และ อาจารย์สมศักดิ์ วาณิชอนันต์ชัย ในฐานะอาจารย์ที่ปรึกษาโครงงานวิศวกรรมโทรคมนาคมที่ให้คำปรึกษาและแนะนำในการทำงานชิ้นนี้จนงานต่าง ๆ สำเร็จลุล่วงไปได้ด้วยดี

ขอขอบคุณ Mr. Allan Pentecost, Mr. David Terra and Mr. Richard Jenson (Technician Support Services Department) จากบริษัท Z-World ประเทศสหรัฐอเมริกา ที่ให้คำแนะนำเกี่ยวกับการใช้งานโปรแกรมไดนามิกส์ซี

ขอขอบคุณ คุณประพล จาระตะคุ สำหรับการอำนวยความสะดวกในการยืมอุปกรณ์การทดสอบ

นอกจากนี้ผู้เขียนขอขอบคุณ คุณศุภชาติ วราวุฒิ คุณณัฐนันท์ ทัดพิทักษ์กุล คุณฤดีรัตน์ ชินเวชกิจวานิชย์ และคุณเผด็จ เผ่าละออ สำหรับกำลังใจและให้ความช่วยเหลือทุกอย่างแก่ผู้เขียนรวมทั้งเพื่อน ๆ ทุกคนที่คอยเป็นกำลังใจจนกระทั่งรายงานฉบับนี้เสร็จสิ้นโดยสมบูรณ์

นางสาวศศิธร อนุรักษ์บัญญา

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

หน้า

รูปที่

สารบัญรูป (ต่อ)

หน้า

รูปที่

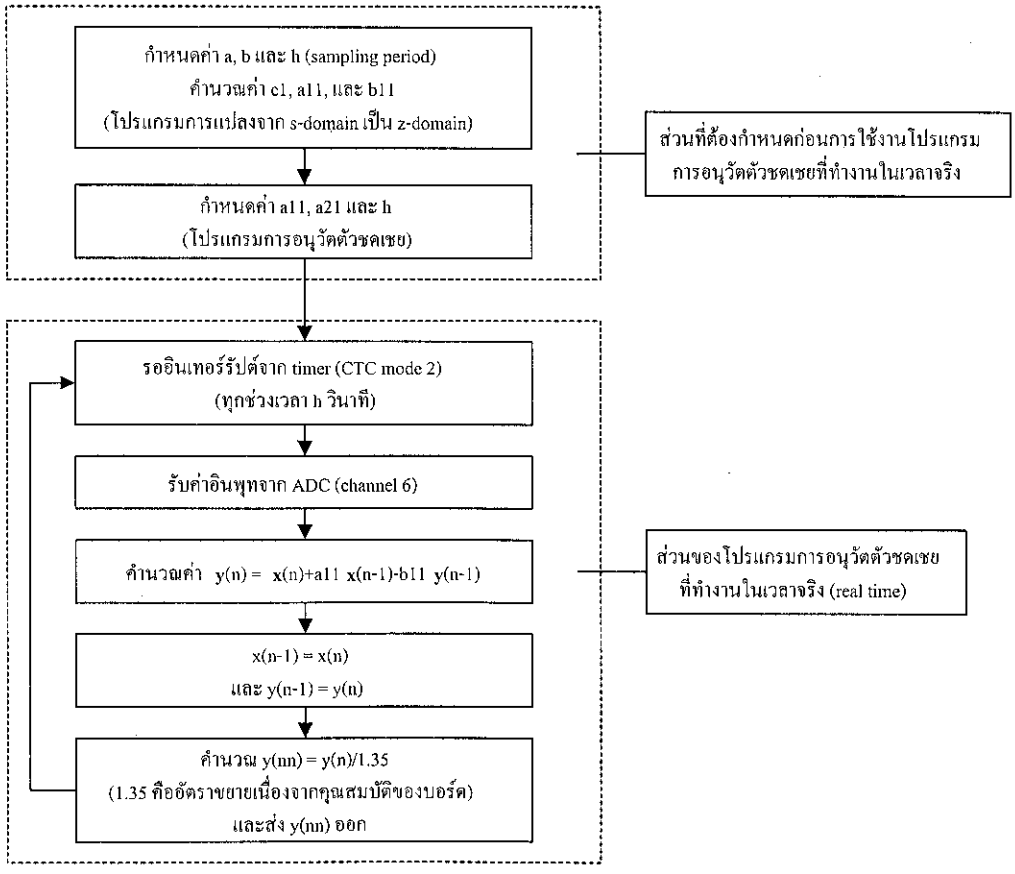
สารบัญรูป (ต่อ)

บทที่ 1

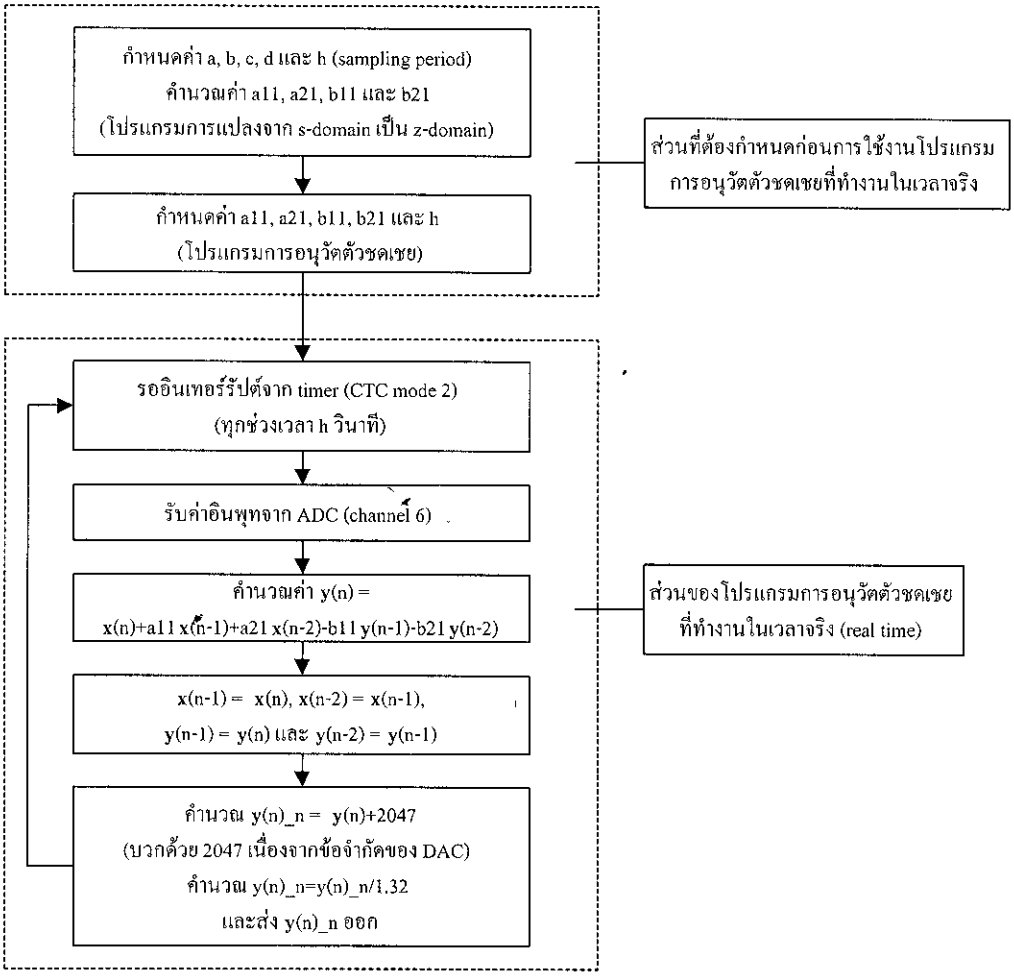
บทนำ

1.1 กล่าวนำ

โครงงานวิศวกรรมโทรคมนาคม (Telecommunication Engineering Project) ภายใต้รายวิชา 409469 เกี่ยวข้องกับการอนุวัตตัวชดเชย (compensator) หรือตัวควบคุม (controller) ด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) โปรแกรมการอนุวัตตัวชดเชยที่พัฒนาขึ้นในโครงงานนี้ประกอบด้วย โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง นอกจากนี้ยังมีโปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันจากรูป s-domain ไปเป็น z-domain โดยที่โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสองจะมีโครงสร้างการทำงานดังรูปที่ 1.1 และ 1.2 ตามลำดับ



รูปที่ 1.1 การทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง



รูปที่ 1.2 การทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

รูปที่ 1.1 อธิบายลักษณะการทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง การทำงานแบ่งออกเป็นสองส่วน ส่วนแรกเป็นการกำหนดค่าตัวแปรต่าง ๆ ที่ใช้ในโปรแกรมการอนุวัตตัวชดเชย ได้แก่ ตัวแปรที่ใช้กำหนดค่าเริ่มต้นของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) และค่าสัมประสิทธิ์ของตัวชดเชยซึ่งอาศัยความสัมพันธ์แบบทุสตินจากการแปลง s-domain ไปเป็น z-domain โดยใช้โปรแกรมที่เขียนขึ้นใน MATLAB ส่วนที่สองจะเป็นการทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

รูปที่ 1.2 อธิบายลักษณะการทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง การทำงานแบ่งออกเป็นสองส่วนเช่นกัน จะเห็นได้ว่าการทำงานจะคล้ายคลึงกัน เพียงแต่ในอันดับที่สองนี้จะมีการกำหนดค่าสัมประสิทธิ์ a21 และ b21 รวมทั้งผลจากข้อจำกัดของ DAC เพิ่มขึ้นมา รายละเอียดการทำงานของแต่ละส่วน จะอธิบายในบทต่อ ๆ ไป

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อศึกษาการทำงานของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)
2. เพื่อศึกษาการโปรแกรมในไดนามิกส์ซี รุ่น 5.24
3. เพื่อศึกษาทฤษฎีพื้นฐานการแปลงแบบทุสตินจาก s-domain เป็น z-domain
4. เพื่อศึกษาโครงสร้างการอนุวัตตัวชดเชย และวิธีการอนุวัตตัวชดเชย
5. เพื่อนำความรู้ในการศึกษา controller บอร์ด BL1120 (Little Giant) มาใช้ในการอนุวัตตัวชดเชย

1.3 ขอบเขตของโครงงาน

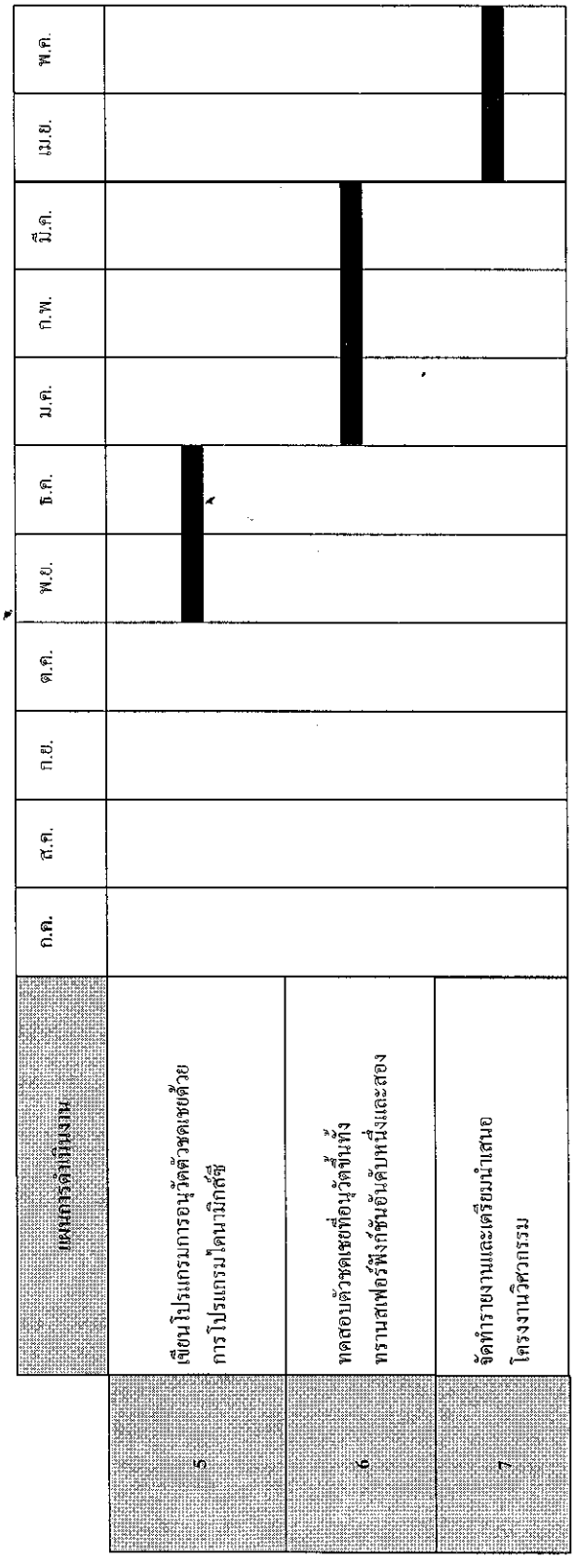
1. พัฒนาโปรแกรมการแปลง s-domain เป็น z-domain โดยใช้วิธีการแปลงแบบทุสติน
2. พัฒนาโปรแกรมการอนุวัตตัวชดเชยด้วยการโปรแกรมภาษาไดนามิกส์ซี
3. ทดสอบตัวชดเชยที่อนุวัตขึ้นทั้งหมด
4. วิเคราะห์ผลการทดลองจากการทดสอบตัวชดเชยที่อนุวัตขึ้น

1.4 ประโยชน์ที่คาดว่าจะได้รับ

- ได้รับความรู้เกี่ยวกับการอนุวัตตัวชดเชย
- ได้รับความรู้เกี่ยวกับคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)
- ได้รับความรู้เกี่ยวกับ Digital Signal Processing
- สามารถพัฒนาตัวชดเชยด้วยการเขียนโปรแกรมภาษาไดนามิกส์ซีในการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)
- สามารถพัฒนาโปรแกรมการแปลงจาก s-domain เป็น z-domain ด้วยการเขียนโปรแกรม MATLAB

1.5 แผนการดำเนินงาน

	แผนการดำเนินงาน	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
1	ศึกษาการทำงานของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)											
2	ศึกษาทฤษฎีการแปลงทรานสเฟอร์ฟังก์ชัน จาก s-domain เป็น z-domain และการหาค่าสัมประสิทธิ์ของทรานสเฟอร์ฟังก์ชัน											
3	ศึกษาโครงสร้างการอนุวัตด้วยคอมและวิธีการอนุวัตด้วยคเชย											
4	เขียนโปรแกรมการแปลงทรานสเฟอร์ฟังก์ชัน จาก s-domain เป็น z-domain พร้อมทั้งคำนวณค่าสัมประสิทธิ์											



	กิจกรรมที่ดำเนินการ	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
5	เขียนโปรแกรมการอนุวัตด้วยคอมพิวเตอร์ด้วย การโปรแกรมโดนามิกส์ซี											
6	ทดสอบตัวอย่างเพื่อนำวัตถุดิบทั้ง ทรานสฟอร์มกับอันดับหนึ่งและสอง											
7	จัดทำรายงานและเตรียมนำเสนอ โครงงานวิศวกรรม											

* ระยะเวลาการดำเนินงานประมาณ 10 เดือน โดยเริ่มตั้งแต่ ก.ค. 42 ถึง พ.ค. 43

1.6 การจัดรูปเล่ม

ในการศึกษาการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ได้แบ่งออกเป็น 5 บท ดังนี้

บทที่ 1 เป็นการกล่าวนำถึงการอนุวัตตัวชดเชย และส่วนประกอบในรายงานฉบับนี้

บทที่ 2 เป็นการกล่าวถึงเทคนิคการอนุวัตตัวชดเชยแบบดิจิตอล ทฤษฎีพื้นฐานการแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain โครงสร้างพื้นฐานของตัวกรองความถี่แบบดิจิตอลชนิด IIR(Infinite Impulse Response) การเลือกโครงสร้างของ IIR มาใช้ในการอนุวัตตัวชดเชย โครงสร้างของภาษาไดนามิกส์ซี นอกจากนี้จะกล่าวถึงโครงสร้างของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) และส่วนที่เกี่ยวข้องกับการอนุวัตตัวชดเชยโดยละเอียด รวมถึงการปรับตั้งค่าต่าง ๆ ก่อนการใช้งานและข้อกำหนดในการใช้บอร์ด

บทที่ 3 เป็นการกล่าวถึงรายละเอียดของส่วนต่าง ๆ ของโปรแกรมการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง และโปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันแบบทุสติน

บทที่ 4 เป็นการกล่าวถึงการทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) การทดสอบเพื่อหาค่าอัตราขยาย รวมทั้งการทดสอบโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและอันดับสอง

บทที่ 5 เป็นข้อสรุปเกี่ยวกับผลการแปลงจาก s-domain เป็น z-domain, การอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) สาระสำคัญเกี่ยวกับบอร์ด โปรแกรมภาษาไดนามิกส์ซี และข้อเสนอแนะในการพัฒนางานให้ดียิ่งขึ้น

บทที่ 2

การอนุวัตตัวชดเชย

2.1 กล่าวนำ

ด้วยเทคโนโลยีในปัจจุบันทำให้มีอุปกรณ์ต่าง ๆ ที่ถูกสร้างขึ้นและนำไปใช้ประโยชน์ในอุตสาหกรรม ทั้งในระบบอนาล็อกและดิจิตอล ซึ่งตัวชดเชยถือได้ว่าเป็นเทคโนโลยีที่สำคัญ ที่ช่วยให้สามารถปรับขนาดและเฟสของสัญญาณตามที่ต้องการได้ ในบทนี้จะกล่าวถึง หลักการการอนุวัตตัวควบคุมแบบดิจิตอลด้วยโปรแกรมภาษาไดน่ามิกส์ซีกับคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ทฤษฎีความสัมพันธ์ของทุสตินพร้อมทั้งแสดงขั้นตอนการแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain โครงสร้างของโปรแกรมภาษาไดนามิกส์ซี และโครงสร้างของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

2.2 เทคนิคการอนุวัตตัวชดเชยแบบดิจิตอล

สาเหตุที่ทำให้การอนุวัตแบบดิจิตอลเป็นที่นิยมมากกว่าแบบอนาล็อก มีหลายสาเหตุด้วยกัน เช่น การโปรแกรมด้วยระบบดิจิตอลจะมีความยืดหยุ่นมากกว่า ง่ายต่อการเปลี่ยนแปลงภายในระบบ สะดวกในการทำรีดีไซน์ การพิสูจน์และการทดสอบ

การใช้คอนโทรลเลอร์บอร์ดมาอนุวัตเป็นตัวชดเชยโดยอาศัยหลักการของ Digital Signal Processing ถือได้ว่าเป็นเทคโนโลยีที่สะดวกและง่ายต่อการนำไปใช้ เนื่องจากการอนุวัตโดยอาศัยหลักการของ Digital Signal Processing กับคอนโทรลเลอร์บอร์ดจะทำเพียงครั้งเดียวสำหรับตัวชดเชยที่มีโครงสร้างตายตัว ไม่มีการปรับจูนใด ๆ ไม่มีปัญหาในเรื่องการแม็ทช์ทางอิมพีแดนซ์ (Impedance Matching) และเมื่อระบบที่ต้องการควบคุมมีทรานสเฟอร์ฟังก์ชันเปลี่ยนแปลงไป แต่โครงสร้างของตัวชดเชยยังเป็นโครงสร้างแบบเดิมก็จะไม่ต้องทำการเขียนโปรแกรมใหม่ทั้งหมด เพียงแต่ทำการคำนวณค่าสัมประสิทธิ์ใหม่จากทรานสเฟอร์ฟังก์ชันที่เปลี่ยนไป แล้วกำหนดค่านั้นให้กับโปรแกรมการอนุวัตตัวชดเชย

การวิเคราะห์และออกแบบตัวควบคุมเพื่อให้ได้ผลดีนั้น นิยมทำใน s-domain แต่ในการอนุวัตตัวชดเชยนี้เป็นการอนุวัตแบบดิจิตอล จะทำใน z-domain ดังนั้นจึงต้องมีการแปลงทรานสเฟอร์ฟังก์ชันในรูป s-domain ไปเป็น z-domain ก่อนด้วยเทคนิคที่เหมาะสม โครงงานนี้จะใช้เทคนิคการแปลงโดยอาศัยความสัมพันธ์แบบทุสติน หลังจากนั้นจึงทำการอนุวัตตัวชดเชยใน z-domain ซึ่งตัวชดเชยจะมีโครงสร้างแบบเดียวกับตัวกรองความถี่แบบดิจิตอลชนิด IIR (Infinite Impulse Response)

2.3 ทฤษฎีพื้นฐานการแปลงแบบทุสตินจาก s-domain ไปเป็น z-domain

เทคนิคการประมาณโดยวิธีของทุสตินมีรากฐานมาจากการพิจารณาในทางเรขาคณิต (Geometrical considerations) ซึ่งก็คือการประมาณค่าโดยใช้สี่เหลี่ยมคางหมูโดยการลากจุดต่อจุด ซึ่งจะมีค่าใกล้เคียงมากกว่าการประมาณโดยใช้สี่เหลี่ยมผืนผ้าโดยวิธีของออยเลอร์ (Euler) ในการวิเคราะห์เชิงจำนวน (Numerical analysis) จะเรียกวิธีนี้ว่า กฎของสี่เหลี่ยมคางหมู (Trapezoidal Rule) ในทาง Signal Processing จะรู้จักกันดีในชื่อการแปลงแบบไบลิเนียร์ (Bilinear transformation) ส่วนในทางระบบควบคุมหรือที่กล่าวถึงในที่นี้จะเรียกว่าวิธีของทุสติน (Tustin's method)

วิธีของทุสตินเป็นที่นิยมกันมากในการแปลงจากระบบอนาล็อกไปเป็นระบบดิจิตอลโดยแปลงจากทรานสเฟอร์ฟังก์ชันใน s-domain G(s) ไปเป็น z-domain G(z) ด้วยความสัมพันธ์ตามสมการที่ 2.1

$$G(z) = G(s)\Big|_{s=\alpha(1-z^{-1})/(1+z^{-1})} \tag{2.1}$$

โดยค่า α ในสมการที่ 2.1 คือ 2/h และ h คือ คาบของการสุ่มสัญญาณ (หน่วยเป็นวินาที) ซึ่งความถี่ของการสุ่มสัญญาณ (sampling frequency) f_s = 1/h Hz หรืออาจเขียนในอีกรูปแบบหนึ่งได้ดังนี้

$$\frac{1}{s} \Leftrightarrow \frac{h}{2}\left(\frac{z+1}{z-1}\right) \quad or \quad s \Leftrightarrow \frac{2}{h}\left(\frac{z-1}{z+1}\right) \tag{2.2}$$

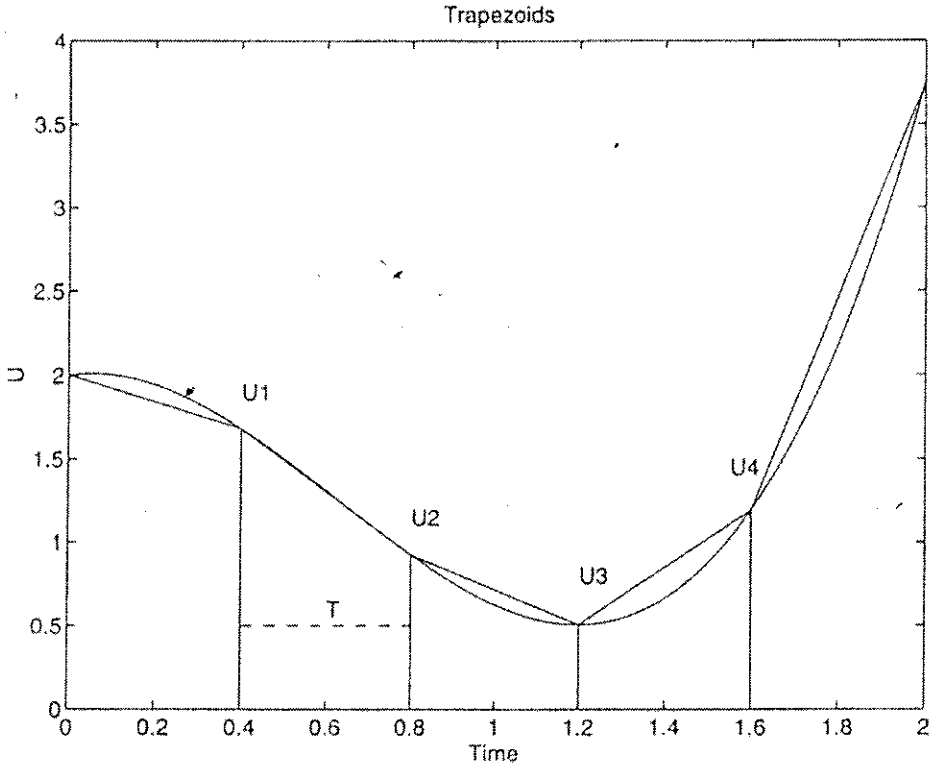
จากที่กล่าวในตอนต้นว่าการประมาณโดยวิธีของทุสตินมาจากผลรวมของสี่เหลี่ยมคางหมู ดังจะเห็นได้จากรูปที่ 2.1 ซึ่งสามารถแสดงได้ด้วยสมการผลต่างดังนี้

$$y_{n+1} = y_n + \frac{h}{2}(u_{n+1} + u_n) \tag{2.3}$$

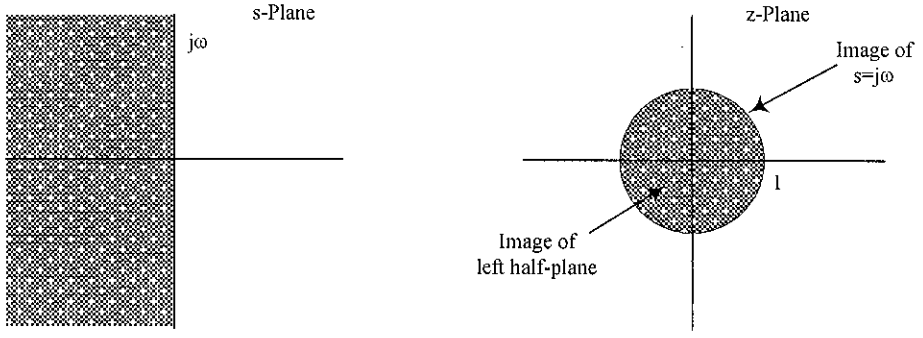
จากการแปลงทรานสเฟอร์ฟังก์ชันในสมการที่ 2.1 สามารถทำ Mapping จาก ระนาบใน s-domain ไปบนระนาบใน z-domain โดยสรุปได้ดังนี้ (ดูรูปที่ 2.2 ประกอบ)

1. ครึ่งซ้ายในระนาบ s $\rightarrow$ ภายในของวงกลมหนึ่งหน่วยในระนาบ z
2. ครึ่งขวาในระนาบ s $\rightarrow$ ภายนอกของวงกลมหนึ่งหน่วยในระนาบ z
3. แกนจินตภาพในระนาบ s = jω $\rightarrow$ เส้นรอบวงของวงกลมหนึ่งหน่วย ($z = e^{j\omega}$)

จะเห็นได้ว่าการแปลงแบบไบลิเนียร์จากตัวกรองความถี่ที่มีเสถียรภาพในทางอนาล็อกไปเป็นตัวกรองความถี่ที่มีเสถียรภาพในทางดิจิตอลสามารถทำได้ถึงแม้ความสัมพันธ์จะไม่เป็นเชิงเส้น (nonlinear) ก็ตาม



รูปที่ 2.1 การประมาณพื้นที่โดยการใช้สี่เหลี่ยมคางหมู



รูปที่ 2.2 Mapping จาก s-plane ไปบน z-plane โดยใช้การแปลงแบบทุสติน (ไบลิเนียร์)

เป็นที่ทราบกันเป็นอย่างดีว่าการแปลงแบบทุสตินนี้ก่อให้เกิดการโก่งทางความถี่ (frequency warping) ซึ่งก็คือ ความถี่ ω_c ในโดเมนต่อเนื่อง (continous) เมื่อแปลงไปเป็น ω_d ในโดเมนไม่ต่อเนื่อง (discrete) จะมีค่าไม่ตรงกัน ดังนั้นในการใช้ความสัมพันธ์ของทุสตินนี้จึงต้องมีการแก้อาการโก่งทางความถี่ (frequency prewarping) ก่อนที่จะประยุกต์ความสัมพันธ์ตามสมการที่ 2.2 หลังจากนั้นเมื่อเสร็จสิ้นกระบวนการทางพีชคณิตที่ประยุกต์ตามสมการที่ 2.2 แล้วจะต้องทำการตรวจสอบว่า dc gain ของทรานสเฟอร์ฟังก์ชันใน z-domain เท่ากันกับใน s-domain หรือไม่ หากไม่เท่ากันจะต้องทำการคำนวณค่าอัตราขยาย K_d ที่เหมาะสมที่จะคูณควบอยู่กับ G(z) แล้วทำให้ G(z) มี dc gain เท่ากับ G(s) เดิม

การแก้อาการโก่งทางความถี่สามารถทำได้โดยประยุกต์ความสัมพันธ์ 2.4 และ 2.5 สำหรับทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสองตามลำดับ

$$(s+\gamma) \rightarrow \left(s+\gamma'\right)\Big|_{\gamma' = \frac{2}{h}\tan\left(\frac{\gamma h}{2}\right)} \tag{2.4}$$

$$\left(s^2 + 2\zeta\omega_n s + \omega_n{}^2\right) \rightarrow \left(s^2 + 2\zeta\omega'_n s + \omega'_n{}^2\right)\Big|_{\omega'_n = \frac{2}{h}\tan\left(\frac{\omega_n h}{2}\right)} \tag{2.5}$$

<u>กรณีที่ 1</u> : ทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

$G(s) = \dfrac{s+a}{s+b}$ เมื่อทำการแก้อาการโก่งทางความถี่ จะได้

$G'(s) = \dfrac{s+\alpha}{s+\beta}$ เมื่อ $\alpha = \dfrac{2}{h}\tan\left(\dfrac{ah}{2}\right)$ และ $\beta = \dfrac{2}{h}\tan\left(\dfrac{bh}{2}\right)$

$$G(z) = G'(s)\Big|_{s = \frac{2}{h}\left(\frac{z-1}{z+1}\right)}$$

$$= \frac{(\alpha h + 2)z + (\alpha h - 2)}{(\beta h + 2)z + (\beta h - 2)}$$

$$= \frac{(\alpha h + 2) + (\alpha h - 2)z^{-1}}{(\beta h + 2) + (\beta h - 2)z^{-1}}$$

$$= \frac{a_0 + a_1 z^{-1}}{b_0 + b_1 z^{-1}}$$

<u>กรณีที่ 2</u> : ทรานสเฟอร์ฟังก์ชันอันดับสอง

$$G(s) = \frac{s^2 + cs + d}{s^2 + as + b} \qquad \text{ซึ่งมี} \quad \omega_{np} = \sqrt{b} \quad \text{และ} \quad \omega_{nz} = \sqrt{d}$$

นอกจากนั้น

$$2\zeta_z \omega_{nz} = c \qquad \text{เมื่อ} \quad \omega_{nz} = \sqrt{d} \quad \text{ดังนั้น} \quad 2\zeta_z = \frac{c}{\sqrt{d}}$$

$$2\zeta_p \omega_{np} = a \qquad \text{เมื่อ} \quad \omega_{np} = \sqrt{b} \quad \text{ดังนั้น} \quad 2\zeta_p = \frac{a}{\sqrt{b}}$$

ทำการแก้อาการโก่งทางความถี่ด้วย

$$\omega'_{nz} = \frac{2}{h}\tan\left(\frac{h\sqrt{d}}{2}\right) \quad \text{และ} \quad \omega'_{np} = \frac{2}{h}\tan\left(\frac{h\sqrt{b}}{2}\right) \qquad \text{ทำให้มี}$$

$$G'(s) = \frac{s^2 + \frac{c}{\sqrt{d}}\omega'_{nz}s + (\omega'_{nz})^2}{s^2 + \frac{a}{\sqrt{b}}\omega'_{np}s + (\omega'_{np})^2}$$

$$G(z) = G'(s)\Big|_{s = \frac{2}{h}\left(\frac{z-1}{z+1}\right)}$$

$$= \frac{s^2 + \gamma s + \omega'^{\,2}_{nz}}{s^2 + \psi s + \omega'^{\,2}_{np}}\Bigg|_{s = \frac{2}{h}\left(\frac{z-1}{z+1}\right)} \quad (\text{กำหนดให้ } \gamma = \frac{c}{\sqrt{d}}\omega'_{nz},\ \psi = \frac{a}{\sqrt{b}}\omega'_{np})$$

$$= \left. \frac{s(s+\gamma)+\omega_{nz}^{'2}}{s(s+\psi)+\omega_{np}^{'2}} \right|_{s=\frac{2}{h}\left(\frac{z-1}{z+1}\right)}$$

$$= \frac{\left(\frac{2}{h}\right)\frac{(z-1)}{(z+1)}\left[\left(\frac{2}{h}\right)\frac{(z-1)}{(z+1)}+\gamma\right]+\omega_{nz}^{'2}}{\left(\frac{2}{h}\right)\frac{(z-1)}{(z+1)}\left[\left(\frac{2}{h}\right)\frac{(z-1)}{(z+1)}+\psi\right]+\omega_{np}^{'2}}$$

$$= \frac{2(z-1)[(\gamma h+2)z+(\gamma h-2)]+h^2(z+1)^2\omega_{nz}^{'2}}{2(z-1)[(\psi h+2)z+(\psi h-2)]+h^2(z+1)^2\omega_{np}^{'2}}$$

$$= \frac{\left(4+2\gamma h+h^2\omega_{nz}^{'2}\right)z^2+\left(2h^2\omega_{nz}^{'2}-8\right)z+\left(4-2\gamma h+h^2\omega_{nz}^{'2}\right)}{\left(4+2\psi h+h^2\omega_{np}^{'2}\right)z^2+\left(2h^2\omega_{np}^{'2}-8\right)z+\left(4-2\psi h+h^2\omega_{np}^{'2}\right)}$$

$$= \frac{a_0+a_1z^{-1}+a_2z^{-2}}{b_0+b_1z^{-1}+b_2z^{-2}}$$

เมื่อพิจารณาว่า h มีค่าน้อยมาก ๆ ส่งผลให้ $h^2 \approx 0$ จะได้

$$G(z) \approx \frac{(4+2\gamma h)-8z^{-1}+(4-2\gamma h)z^{-2}}{(4+2\psi h)-8z^{-1}+(4-2\psi h)z^{-2}} = G^*(z)$$

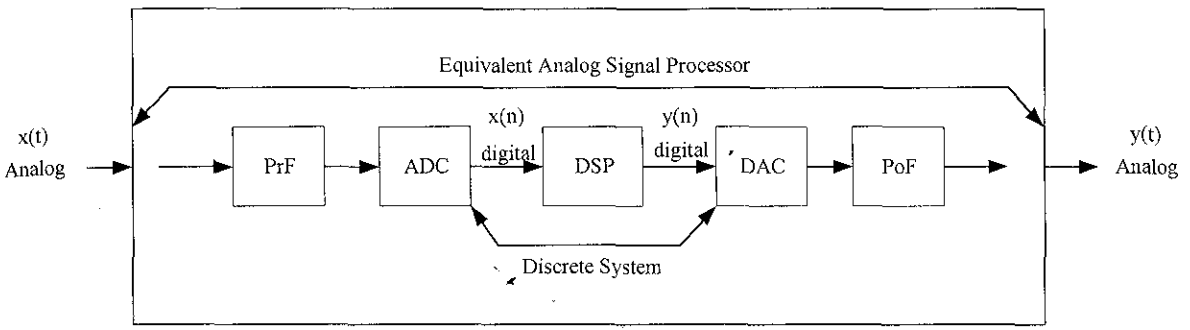
จากรายละเอียดการวิเคราะห์ที่นำเสนอผ่านมา อาจสรุปข้อมูลที่ใช้ประโยชน์สำหรับการแปลง $G(s)$ ไปเป็น $G(z)$ ที่มีการแก้อาการโก่งทางความถี่แล้วดังตารางที่ 2.1 อย่างไรก็ตามการประยุกต์ตารางที่ 2.1 จะต้องตามด้วยการตรวจสอบ dc gain ของ $G(s)$ กับ $G(z)$ เสมอ หากพบว่าแตกต่างกันจะต้องทำการวินิจฉัยอัตราขยาย K สำหรับ $G(z)$ ให้ถูกต้องด้วย

ตารางที่ 2.1 ความสัมพันธ์ในรูปทั่วไปสำหรับการแปลง $G(s)$ อันดับหนึ่งและสองไปเป็น $G(z)$ ด้วยการแปลงแบบทุสติน

$G(s)=\dfrac{s+a}{s+b}$	$G(s)=\dfrac{s^2+cs+d}{s^2+as+b}$
$G(z)=\dfrac{a_0+a_1z^{-1}}{b_0+b_1z^{-1}}$ $a_0=\alpha h+2$ $a_1=\alpha h-2$ $b_0=\beta h+2$ $b_1=\beta h-2$ เมื่อ $\alpha=\dfrac{2}{h}\tan\left(\dfrac{ah}{2}\right)$ $\beta=\dfrac{2}{h}\tan\left(\dfrac{bh}{2}\right)$	$G(z)=\dfrac{a_0+a_1z^{-1}+a_2z^{-2}}{b_0+b_1z^{-1}+b_2z^{-2}}$ **กรณี** $h^2>>\epsilon$ ซึ่ง $\epsilon\approx 0$ **กรณี** $h^2\approx 0$ $a_0=4+2\gamma h+h^2\omega_{nz}'^2$ $a_0=4+2\gamma h$ $a_1=2h^2\omega_{nz}'^2-8$ $a_1=-8$ $a_2=4-2\gamma h+h^2\omega_{nz}'^2$ $a_2=4-2\gamma h$ $b_0=4+2\psi h+h^2\omega_{np}'^2$ $b_0=4+2\psi h$ $b_1=2h^2\omega_{np}'^2-8$ $b_1=-8$ $b_2=4-2\psi h+h^2\omega_{np}'^2$ $b_2=4-2\psi h$ **ทั้งสองกรณีมี** $\gamma=\dfrac{c}{\sqrt{d}}\omega_{nz}'$ และ $\psi=\dfrac{a}{\sqrt{b}}\omega_{np}'$ $\omega_{nz}'=\dfrac{2}{h}\tan\left(\dfrac{h\sqrt{d}}{2}\right)$ $\omega_{np}'=\dfrac{2}{h}\tan\left(\dfrac{h\sqrt{b}}{2}\right)$

2.4 ทฤษฎีพื้นฐานของตัวกรองความถี่แบบดิจิตอล

กระบวนการดำเนินการกับระบบสัญญาณดิจิตอล (Digital Signal Processing, DSP) มีโครงสร้างดังแสดงได้ด้วยแผนภูมิข้างล่างดังนี้



รูปที่ 2.3 โครงสร้างของตัวกรองความถี่แบบดิจิตอลในเวลาจริง

PrF: คือ prefilter หรือ antialiasing filter ซึ่งจะช่วยปรับสภาพของสัญญาณอนาล็อกเพื่อป้องกันการเกิด aliasing

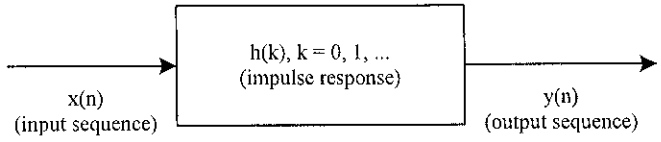
ADC: คืออุปกรณ์การแปลงสัญญาณอนาล็อกเป็นดิจิตอล (analog-to-digital converter) ซึ่งจะช่วยผลิตเลขฐานสอง (binary numbers) จากสัญญาณอนาล็อก

Digital Signal Processor: คือส่วนที่สำคัญที่สุดสำหรับ DSP ใช้แสดงวัตถุประสงค์ทั่วไปของคอมพิวเตอร์และวัตถุประสงค์หลักของตัวประมวลผล (processors) หรือ ฮาร์ดแวร์ทางดิจิตอล และอื่น ๆ

DAC: คืออุปกรณ์ที่ทำงานตรงกันข้ามกับ ADC โดยในขั้นแรกจะสร้างรูปสัญญาณที่เป็นขั้นบันไดจากลำดับของเลขฐานสองก่อนการเปลี่ยนไปเป็นสัญญาณอนาล็อก

PoF: คือ postfilter จะช่วยกรองรูปสัญญาณขั้นบันไดให้กลายเป็นสัญญาณอนาล็อกที่สวยงาม

ตัวกรองความถี่แบบดิจิตอลที่ใช้กันอย่างกว้างขวางในปัจจุบันมี 2 ชนิดด้วยกัน คือ Finite Impulse Response (FIR) และ Infinite Impulse Response (IIR) โดยทั้งสองชนิดสามารถแสดงได้ด้วย impulse response sequence, h(k) (โดยที่ k = 0, 1, ...) ดังรูปที่ 2.4 ซึ่งสัญญาณอินพุทและเอาท์พุทของตัวกรองความถี่จะมีความสัมพันธ์กับการบวกทางคอนโวลูชัน (convolution sum) ซึ่งสามารถแสดงได้ด้วยสมการที่ 2.6 และ 2.7 สำหรับตัวกรองความถี่ชนิด FIR และ IIR ตามลำดับ



รูปที่ 2.4 การนำเสนอข้อสรุปของตัวกรองความถี่แบบดิจิตอล

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k) \tag{2.6}$$

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k) \tag{2.7}$$

จะเห็นได้ว่าตัวกรองความถี่ชนิด IIR มี impulse response จำนวนมาก ไม่สามารถคำนวณค่าเอาท์พุทได้ ดังนั้นจึงนำสมการที่ 2.7 มาเขียนใหม่ในรูปของรีเคอร์ซีฟ (recursive form) ดังนี้

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k) = \sum_{k=0}^{N} a_k x(n-k) - \sum_{k=1}^{M} b_k y(n-k) \tag{2.8}$$

โดยที่ a_k และ b_k คือสัมประสิทธิ์ของตัวกรองความถี่ สมการที่ 2.8 ที่อยู่ในรูปสมการผลต่างนั้น จะเห็นว่า $y(n)$ เป็นฟังก์ชันของเอาท์พุทกับอินพุทในอดีต ซึ่งก็หมายความว่าระบบป้อนกลับบางชนิดจะเป็นตัวกรองความถี่ชนิด IIR ซึ่งทั้งสมการที่ 2.6 และ 2.7 สามารถเขียนในรูปของทรานสเฟอร์ฟังก์ชันใน z-domain ได้ดังสมการที่ 2.9 และ 2.10 สำหรับตัวกรองความถี่ชนิด FIR และ IIR ตามลำดับ

$$H(z) = \sum_{k=0}^{N-1} h(k)z^{-k} \tag{2.9}$$

$$H(z) = \sum_{k=0}^{N} a_k z^{-k} \Big/ \left(1 + \sum_{k=1}^{M} b_k z^{-k}\right) \tag{2.10}$$

ตัวกรองความถี่ทั้งชนิด FIR และ IIR ต่างก็มีข้อดีข้อด้อยต่างกันดังแสดงในตารางที่ 2.2

ตารางที่ 2.2 การเปรียบเทียบข้อดีและข้อด้อยของตัวกรองความถี่ชนิด IIR และ FIR

ตัวกรองความถี่ชนิด IIR	ตัวกรองความถี่ชนิด FIR
1. เมื่อต้องการ transition band แคบ ๆ จะใช้ order ต่ำกว่าชนิด FIR	1. เมื่อต้องการ transition band ในช่วงแคบ ๆ จะใช้ order สูงกว่าชนิด IIR
2. ความยุ่งยากในการสร้างตัวกรองความถี่ชนิด IIR มีน้อยกว่าชนิด FIR และใช้ memory น้อยกว่า (เนื่องจาก order ต่ำกว่า)	2. ความยุ่งยากในการสร้างตัวกรองความถี่ชนิด FIR มีมากกว่าชนิด IIR และใช้ memory มาก (เนื่องจาก order สูงกว่า)
3. เวลาในการหน่วง (delay time) น้อยกว่า (เนื่องจาก order ต่ำกว่า)	3. เวลาในการหน่วง (delay time) มากกว่า (เนื่องจาก order สูงกว่า)
4. ที่ order เท่ากัน จะมี sharp cutoff (transition band แคบ) และ lower side lobe ต่ำกว่าชนิด FIR	4. ที่ order เท่ากัน จะมี transition band กว้าง และ lower side lobe สูงกว่าชนิด IIR
5. Nonlinear phase response	5. Linear phase response
6. เนื่องจากเป็น Recursive จึงไม่สามารถแน่ใจได้ว่ามีเสถียรภาพหรือไม่	6. เนื่องจากเป็น non-recursive จึงมีเสถียรภาพ (Stability) มากกว่า
7. มีผลกระทบเนื่องมาจาก roundoff noise, coefficient quantization errors, finite wordlength effects มากกว่าชนิด FIR	7. มีผลกระทบเนื่องมาจาก roundoff noise, coefficient quantization errors, finite wordlength effects น้อยกว่าชนิด IIR

ขั้นตอนต่อไปจะกล่าวถึงโครงสร้างที่ใช้ในการสร้างตัวกรองความถี่แบบดิจิตอล แต่ก่อนที่จะกล่าวถึงโครงสร้างของตัวกรองความถี่จะต้องทำความเข้าใจองค์ประกอบพื้นฐานที่นำมาใช้ในแผนภาพแสดงโครงสร้างเสียก่อน

2.4.1 องค์ประกอบพื้นฐานของแผนภาพแสดงโครงสร้างตัวกรองความถี่

องค์ประกอบพื้นฐานในแผนภาพมีด้วยกัน 3 องค์ประกอบคือ

- องค์ประกอบคูณ (Multiplier element)
- องค์ประกอบบวก (Adder/accumulator element)
- องค์ประกอบหน่วงในหนึ่งหน่วยเวลา (Delay of 1 unit of time)

โดยที่แต่ละหน่วยจะใช้สัญลักษณ์ดังรูปที่ 2.5 ซึ่งความหมายของแต่ละองค์ประกอบจะอธิบายในรายละเอียดต่อไปนี้

องค์ประกอบคูณ (Multiplier element)

จะทำหน้าที่นำสัญญาณเวลาเต็มหน่วยเข้ามาคูณกับค่าคงที่ (k) ที่อยู่ภายในองค์ประกอบคูณแล้วส่งผลลัพธ์ออกไปที่ด้านออกในเวลาเดียวกัน ถ้าให้สัญญาณด้านเข้าแทนด้วย x(n) และสัญญาณด้านออกแทนด้วย y(n) สามารถเขียนความสัมพันธ์ระหว่างสัญญาณด้านออกกับด้านเข้าโดยใช้องค์ประกอบคูณได้ดังนี้

$$y(n) = kx(n)$$

เมื่อ k คือ ค่าคงที่ภายในองค์ประกอบคูณ ดูสัญลักษณ์ขององค์ประกอบนี้ได้ในรูปที่ 2.5.1

องค์ประกอบบวก(Adder/accumulator element)

จะทำหน้าที่รวมสัญญาณเวลาเต็มหน่วยตั้งแต่ 2 สัญญาณขึ้นไป ซึ่งในการรวมกันอาจจะเป็นการบวกหรือการลบก็ได้ขึ้นอยู่กับการกำหนด ถ้าให้สัญญาณด้านเข้าเป็น x(n) และ w(n) และให้สัญญาณด้านออกเป็น y(n) สามารถเขียนความสัมพันธ์ของสัญญาณด้านออกกับสัญญาณด้านเข้าโดยใช้องค์ประกอบบวกได้ดังนี้

$$y(n) = x(n) + w(n)$$

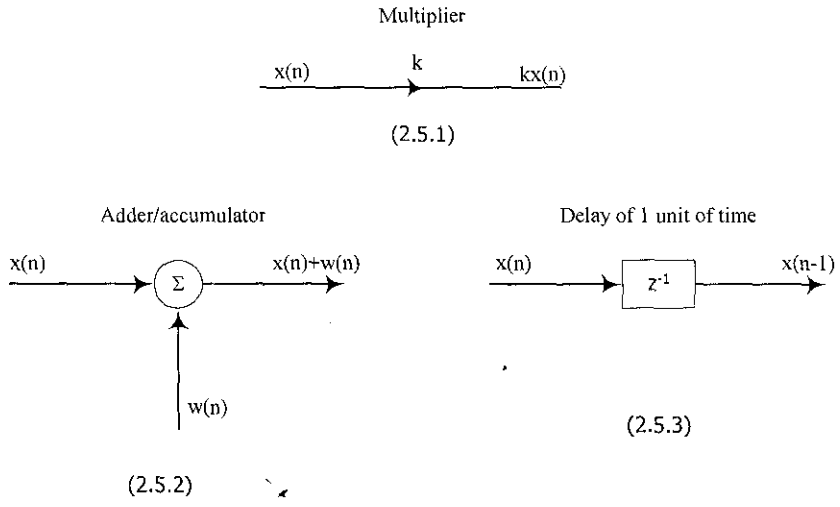
ดูสัญลักษณ์ขององค์ประกอบนี้ได้ในรูปที่ 2.5.2

องค์ประกอบหน่วงในหนึ่งหน่วยเวลา (Delay of 1 unit of time)

จะทำหน้าที่เก็บสัญญาณเวลาเต็มหน่วยที่รับเข้ามานั้นเอาไว้จนกว่าสัญญาณเวลาเต็มหน่วยต่อไปจะเข้ามาจึงจะส่งสัญญาณนั้นออกมาทางด้านออก ดังนั้นสัญญาณด้านออกที่เวลาปัจจุบันจึงเท่ากับสัญญาณด้านเข้าตัวที่แล้ว ถ้าให้สัญญาณด้านเข้าเป็น x(n) และสัญญาณด้านออกเป็น y(n) สามารถเขียนความสัมพันธ์ของสัญญาณด้านออกกับสัญญาณด้านเข้าได้โดยใช้องค์ประกอบหน่วงในหนึ่งหน่วยเวลาได้ดังนี้

$$y(n) = x(n-1)$$

ดูสัญลักษณ์ขององค์ประกอบนี้ได้ในรูปที่ 2.5.3



รูปที่ 2.5 องค์ประกอบพื้นฐานที่ใช้ในโครงสร้างตัวกรองความถี่

จากองค์ประกอบพื้นฐานดังกล่าว สามารถนำมาประกอบให้เป็นโครงสร้างของตัวกรองความถี่แบบดิจิตอลได้ดังจะกล่าวต่อไป

2.4.2 โครงสร้างของตัวกรองความถี่แบบดิจิตอลชนิด IIR (Infinite Impulse Response)

เนื่องจากตัวชดเชยที่ใช้งานกันส่วนใหญ่จะอยู่ในรูปฟังก์ชันของโพลและซีโร่ ซึ่งมีโครงสร้างแบบเดียวกันกับโครงสร้างของตัวกรองความถี่แบบดิจิตอลชนิด IIR ดังนั้นในรายงานฉบับนี้ จะกล่าวถึงเฉพาะโครงสร้างของตัวกรองความถี่แบบดิจิตอลชนิด IIR เท่านั้น

สำหรับตัวกรองความถี่แบบดิจิตอลชนิด IIR จากความสัมพันธ์ในสมการที่ 2.8 ซึ่งสามารถเขียนเป็นทรานสเฟอร์ฟังก์ชันในรูป z-domain ได้ตามสมการที่ 2.10 สามารถนำมาเขียนให้อยู่ในรูปที่มีอัตราขยาย (K) คูณอยู่ด้วย พร้อมทั้งแสดงโพลและซีโร่ได้ดังสมการที่ 2.11

$$H(z) = \frac{K(z - z_1)(z - z_2)\ldots(z - z_N)}{(z - p_1)(z - p_2)\ldots(z - p_M)} \tag{2.11}$$

โดยที่ $z_1, z_2, \ldots$ คือ zeros ของทรานสเฟอร์ฟังก์ชัน $H(z)$ และ $p_1, p_2, \ldots$ คือ poles ของทรานสเฟอร์ฟังก์ชัน $H(z)$

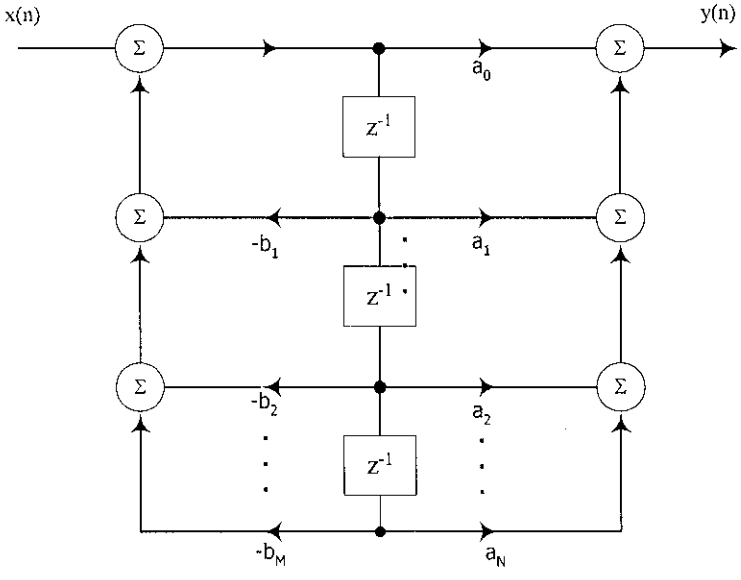
โครงสร้างแบบ IIR สามารถแบ่งออกได้เป็น 3 รูปแบบใหญ่ ๆ คือ โครงสร้างแบบตรง (Direct Form) โครงสร้างแบบต่อเรียงกัน (Cascade Form) และโครงสร้างแบบต่อขนาน (Parallel Form)

2.4.2.1 โครงสร้างแบบตรง (Direct Form)

โครงสร้างแบบตรงตามสมการที่ 2.12 แสดงได้ด้วยรูปที่ 2.6 โดยกำหนดให้ N = M เพื่อง่ายต่อการวิเคราะห์ ซึ่งสัมประสิทธิ์ที่ใช้ในแผนภาพคือค่าสัมประสิทธิ์ในทรานสเฟอร์ฟังก์ชัน แต่เครื่องหมายของสัมประสิทธิ์ในส่วนดีโนมิเนเตอร์ (denominator) ในสมการที่ 2.12a จะตรงกันข้าม

$$H(z) = \sum_{k=0}^{N} a_k z^{-k} \Big/ \left(1 + \sum_{k=1}^{M} b_k z^{-k}\right) \qquad M \geq N \tag{2.12a}$$

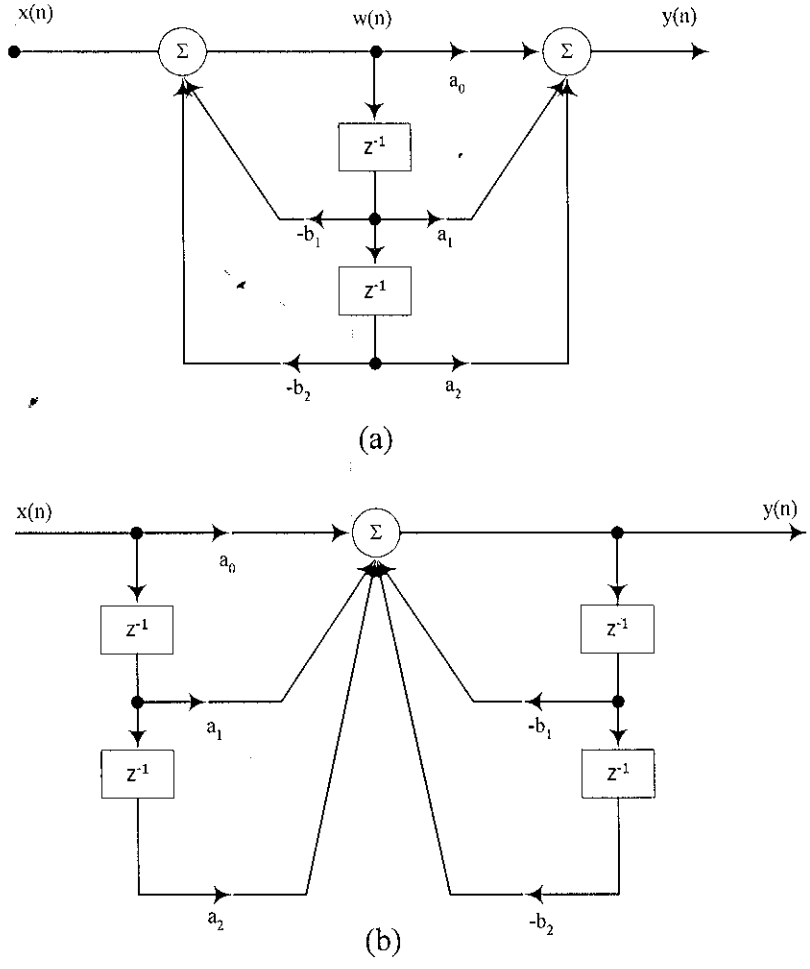
$$y(n) = \sum_{k=0}^{N-1} a_k x(n-k) - \sum_{k=1}^{M} b_k y(n-k) \tag{2.12b}$$



รูปที่ 2.6 โครงสร้างแบบตรง (Direct Form)

เมื่อ order ของตัวกรองความถี่สูงขึ้น เช่น กรณีที่ M > 3 โครงสร้างแบบตรงในรูปที่ 2.6 นี้จะได้รับผลกระทบมากจาก finite wordlength effects ซึ่งในกรณีเช่นนี้ควรที่จะหลีกเลี่ยง ในทางปฏิบัติจะทำการลดรูป H(z) ให้เล็กลง คือให้อยู่ในรูป second-order หรือ first-order แล้วนำองค์ประกอบเหล่านี้มาเรียงต่อกัน (cascade) หรือ นำมาต่อแบบขนานกัน (parallel) ซึ่งจะกล่าวในรายละเอียดต่อไป

สำหรับแผนภาพที่ใช้ในการสร้าง second-order ในทางปฏิบัติ เพื่อใช้ใน order ที่สูงขึ้น (higher order) สามารถดูได้จากรูปที่ 2.7



รูปที่ 2.7 โครงสร้างของ second-order ที่ใช้ในการสร้างตัวกรองความถี่แบบดิจิตอลชนิด IIR ในทางปฏิบัติ: (a) canonic second-order section; (b) direct form second-order section.

โครงสร้างแบบ (a) เรียกว่า canonic section หรือ direct form 2 เนื่องจากมีองค์ประกอบหน่วงเวลาน้อยที่สุด ซึ่งสามารถแสดงได้ด้วยสมการที่ 2.13 ดังนี้

$$w(n) = \sum_{k=0}^{2} a_k x(n) - \sum_{k=1}^{2} b_k w(n-k) \qquad (2.13a)$$

$$y(n) = \sum_{k=0}^{2} a_k w(n-k) \tag{2.13b}$$

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \tag{2.13c}$$

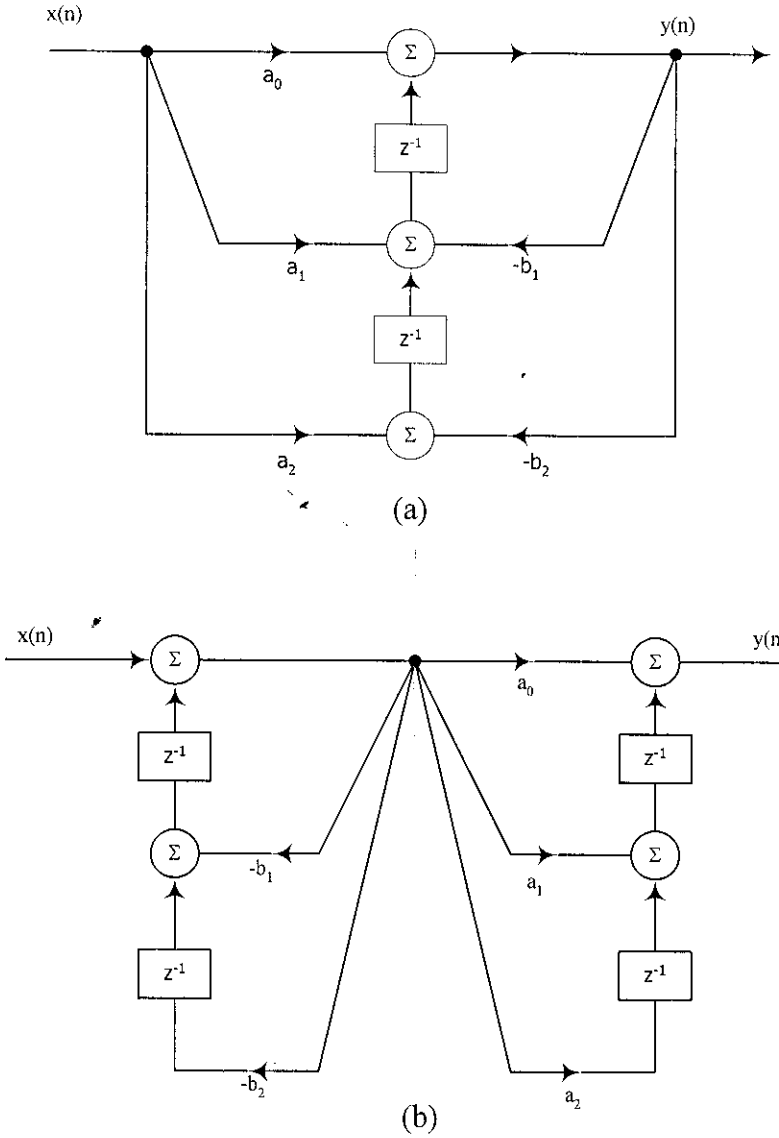
ส่วนโครงสร้างแบบ (b) คือโครงสร้างแบบตรงของ second-order แบบ IIR ซึ่งสามารถแสดงได้ด้วยสมการที่ 2.14

$$y(n) = \sum_{k=0}^{2} a_k x(n-k) - \sum_{k=1}^{2} b_k y(n-k) \tag{2.14a}$$

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \tag{2.14b}$$

โครงสร้างแบบ canonic ในรูปที่ 2.7 (a) เป็นโครงสร้างที่มีคนนิยมใช้มากที่สุด เนื่องจากมีผลจาก roundoff noise ค่อนข้างน้อยและใช้องค์ประกอบในการเก็บค่าน้อยที่สุด แต่อาจจะเกิด overflow ภายในได้ ซึ่งสามารถหลีกเลี่ยงการเกิด overflow ได้โดยการทำ scaling นอกจากนั้นทั้งสองโครงสร้างสามารถทำเป็น first-order ได้โดยให้ $a_2 = b_2 = 0$

ในรูปที่ 2.8 แสดงทรานสโพส (transpose) ของ second-order canonic และ direct form โดยการเปลี่ยน node เป็น adder adder เป็น node และกลับทิศทางหัวลูกศร ถึงแม้ทรานสเฟอร์ฟังก์ชันที่ทรานสโพสของทั้งสองโครงสร้างในรูปที่ 2.8 จะเหมือนกัน แต่คุณสมบัติที่เกี่ยวกับ finite wordlength ของทั้งสองโครงสร้างจะแตกต่างกัน โครงสร้างแบบอื่น ๆ อาจจะได้รับผลกระทบจาก finite wordlength น้อยกว่าแต่ในขณะเดียวกันโครงสร้างเหล่านั้นก็จะมีความซับซ้อนมากกว่าด้วย



รูปที่ 2.8 โครงสร้างทรานสโพส (a) transpose canonic second-order; (b) transpose direct form second-order.

2.4.2.2 โครงสร้างแบบต่อเรียงกัน (Cascade Form)

ในทางปฏิบัติทรานสเฟอร์ฟังก์ชันที่มีอันดับสูงจะใช้โครงสร้างแบบต่อเรียงกันหรือต่อขนานกัน โดยโครงสร้างดังกล่าวเกิดจากการรวมกันของโครงสร้างแบบตรงที่เป็น second-order และ first-order สำหรับโครงสร้างแบบต่อเรียงกันสามารถเขียนแทนด้วยทรานสเฟอร์ฟังก์ชันได้ดังสมการที่ 2.15

$$H(z) = \prod_{k=1}^{N/2}\left[\frac{a_{0k} + a_{1k}z^{-1} + a_{2k}z^{-2}}{1 + b_{1k}z^{-1} + b_{2k}z^{-2}}\right]$$

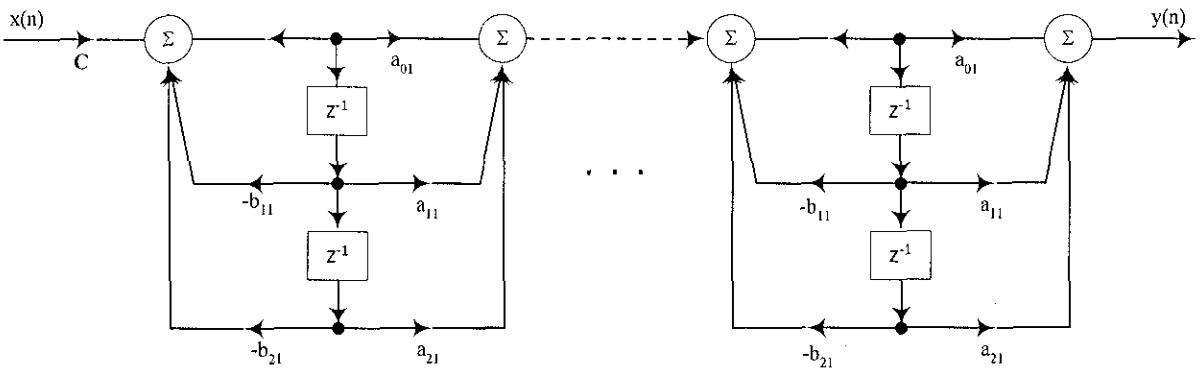
$$= \prod_{k=1}^{N/2}\frac{N_k(z)}{D_k(z)} \qquad (2.15a)$$

$$N_k(z) = a_{0k} + a_{1k}z^{-1} + a_{2k}z^{-2}$$

$$D_k(z) = 1 + b_{1k}z^{-1} + b_{2k}z^{-2} \qquad (2.15b)$$

N คือ order ของตัวกรองความถี่ โดยให้ N เป็นจำนวนคู่ แต่ถ้า N เป็นจำนวนคี่จะทำให้มีส่วนย่อยที่เป็น first-order หนึ่งส่วนใน $H_k(z)$

แต่ละส่วนย่อยที่เป็น second-order จะนำมาต่อเรียงกันดังรูปที่ 2.9 ซึ่งความซับซ้อนในโครงสร้างแบบต่อเรียงกันก็คือ วิธีที่จะจับคู่พจน์เศษกับพจน์ส่วน order ของแต่ละส่วนย่อยที่ควรนำมาเชื่อมต่อและความต้องการที่จะทำ scaling ระดับสัญญาณที่หลาย ๆ จุดในตัวกรองความถี่เพื่อที่จะหลีกเลี่ยงระดับสัญญาณที่สูงหรือต่ำเกินไป



รูปที่ 2.9 โครงสร้างแบบต่อเรียงกัน (Cascade Form)

จากรูปที่ 2.9 สามารถเขียนแสดงได้ด้วยสมการที่ 2.16 โดยที่ C คือค่าคงที่

$$H(z) = \mathbf{c}\prod_{k=1}^{n}\frac{a_{0k} + a_{1k}z^{-1} + a_{2k}z^{-2}}{1 + b_{11}z^{-1} + b_{21}z^{-2}} \qquad (2.16)$$

2.4.2.3 โครงสร้างแบบต่อขนาน (Parallel Form)

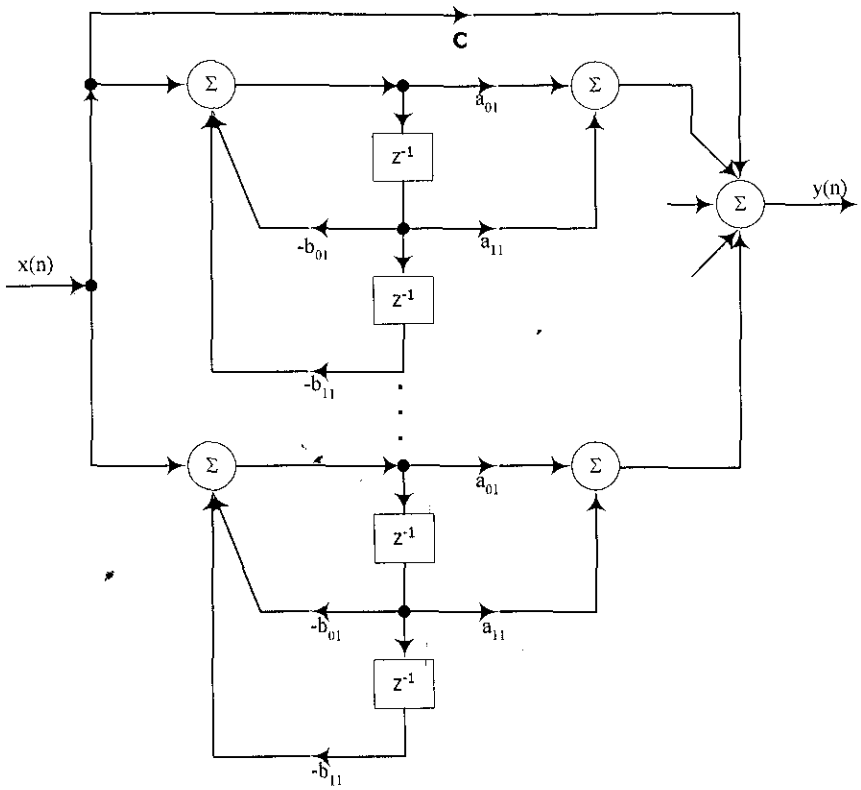
โครงสร้างแบบต่อขนานนี้ ทรานสเฟอร์ฟังก์ชันที่มี N^{th}-order H(z) จะถูกแยกออกโดยการใช้ partial fraction แสดงได้ด้วยสมการที่ 2.17

$$H(z) = \mathbf{c} + \sum_{k=1}^{N/2} H_k(z) \tag{2.17}$$

โดยที่

$$\mathbf{c} = \frac{a_N}{b_N} \quad , \quad H_k(z) = \frac{a_{0k} + a_{1k} z^{-1}}{1 + b_{1k} z^{-1} + b_{2k} z^{-2}}$$

แต่ละส่วนย่อยที่เป็น second-order จะถูกนำมาต่อกันดังในรูปที่ 2.10 สัมประสิทธิ์ในพจน์เศษของ z^{-2} ก็คือ zero ในโครงสร้างแบบต่อขนานนี้ order ที่นำมาเชื่อมต่อยังไม่ค่อยมีความสำคัญมากนัก ยิ่งไปกว่านั้นการทำ scaling ก็ง่ายกว่า สามารถแยกแต่ละส่วนย่อยมาพิจารณาได้อย่างอิสระ และ SNRsของโครงสร้างนี้ก็สามารถเทียบได้กับการต่อแบบเรียงกัน (cascade) ที่ดีที่สุด อย่างไรก็ตามซีโร่ของโครงสร้างแบบต่อขนานยังคงได้รับผลกระทบจาก quantization errors มากกว่าแบบต่อเรียงกัน



รูปที่ 2.10 โครงสร้างแบบต่อขนาน (Parallel Form)

สมการที่ 2.18 แสดงความสัมพันธ์ของโครงสร้างแบบต่อขนานในรูปที่ 2.10

$$H(z) = \mathbf{c} + \sum_{k=1}^{n} \frac{a_{0k} + a_{1k} z^{-1}}{1 + b_{1k} z^{-1} + b_{2k} z^{-2}} \tag{2.18}$$

2.4.3 โครงสร้างการอนุวัตตัวชดเชย

โครงสร้างของตัวกรองความถี่แบบ IIR มีหลายแบบ อย่างไรก็ตามไม่มีโครงสร้างใดที่สมบูรณ์แบบ ผู้ใช้จึงต้องตัดสินใจเลือกโครงสร้างให้เหมาะสมกับการใช้งานของตนมากที่สุด การประยุกต์ที่ใช้การคำนวณแบบ floating point มักจะพบปัญหาน้อยกว่าการใช้ fixed point แต่การประยุกต์ในด้านระบบควบคุม ยังคงมีข้อจำกัดทางด้าน hardware ในเรื่องจำนวนบิตของหน่วยความจำ ข้อจำกัดนี้ส่งผลให้เกิดการแกว่งแบบดำรงตัว (limit-cycle oscillation) ในเอาท์พุทจากตัวชดเชย เมื่อพิจารณาประเด็นที่ว่า จะต้องป้องกันไม่ให้เกิดการแกว่งแบบดำรงตัว หรือมีโอกาสเกิดขึ้นน้อยที่สุด วิธีปฏิบัติที่เหมาะสมคือ ใช้การตัดทศนิยม (truncation) และใช้โครงสร้างแบบต่อเรียงกันในกรณีของ high-order โดยใช้ผลการอนุวัตทรานสเฟอร์ฟังก์ชันอันดับหนึ่งหรืออันดับสองด้วยโครงสร้างแบบ

direct form 2 (canonic) วิธีการดังกล่าวจะให้ความสะดวกและง่ายต่อการอนุวัตมากที่สุด เพราะถ้าต้องการอนุวัต ทรานสเฟอร์ฟังก์ชันที่มีอันดับสูงกว่าสอง ก็สามารถแยกองค์ประกอบย่อยให้อยู่ในรูปผลคูณของอันดับสองหลาย ๆ พจน์กับอันดับหนึ่งได้ ดังแผนภาพโครงสร้างแบบต่อเรียงกัน (cascade) ในรูปที่ 2.9 การอนุวัตในโครงงานนี้ได้ทำการอนุวัตทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง ซึ่งเป็นโครงสร้างในองค์ประกอบย่อยที่สำคัญมากโดยใช้โครงสร้างแบบ direct form 2 ที่กล่าวข้างต้น สำหรับ ทรานสเฟอร์ฟังก์ชัน high-order ผู้ใช้สามารถนำการอนุวัตในโครงงานนี้ไปใช้ประยุกต์ได้ดังที่กล่าวไปแล้ว นอกจากนั้นในการอนุวัตครั้งนี้ยังใช้ทั้งการคำนวณแบบ floating point ร่วมกับ fixed point โดยแสดงโครงสร้างการอนุวัตทรานสเฟอร์ฟังก์ชันอันดับหนึ่งในรูปที่ 2.11 และโครงสร้างการอนุวัตทรานสเฟอร์ฟังก์ชันอันดับสองในรูปที่ 2.12 ส่วนสมการที่ 2.19 และ 2.20 แสดงความสัมพันธ์ของทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสองตามลำดับ

$$H(z) = \frac{Y(z)}{X(z)} = \frac{a_0 + a_1 z^{-1}}{b_0 + b_1 z^{-1}} \tag{2.19a}$$

$$H(z) = C_1\left(\frac{1 + a_{11} z^{-1}}{1 + b_{11} z^{-1}}\right)$$

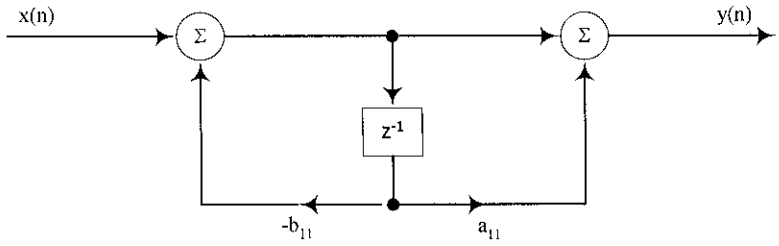
$$y(n) = C_1\left(x(n) - \frac{b_1}{b_0} y(n-1) + \frac{a_1}{a_0} x(n-1)\right) \tag{2.19b}$$

ถ้าให้ $a_{11} = \frac{a_1}{a_0}$ และ $b_{11} = \frac{b_1}{b_0}$

จะได้ความสัมพันธ์ดังในสมการที่ 2.19c ซึ่งสมการนี้นำไปใช้ในโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

$$y(n) = C_1\left(x(n) + a_{11} x(n-1) - b_{11} y(n-1)\right) \tag{2.19c}$$

ค่าคงที่ C_1 นี้จะนำมาคูณเข้าทีหลังเพื่อป้องกันการเกิด overflow ในการประมวลผลของโปรแกรมการอนุวัตตัวชดเชย



รูปที่ 2.11 โครงสร้างที่ใช้ในการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

$$H(z) = \frac{Y(z)}{X(z)} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{b_0 + b_1 z^{-1} + b_2 z^{-2}} \tag{2.20a}$$

$$H(z) = c_2 \left(\frac{1 + a_{11} z^{-1} + a_{21} z^{-2}}{1 + b_{11} z^{-1} + b_{21} z^{-2}} \right)$$

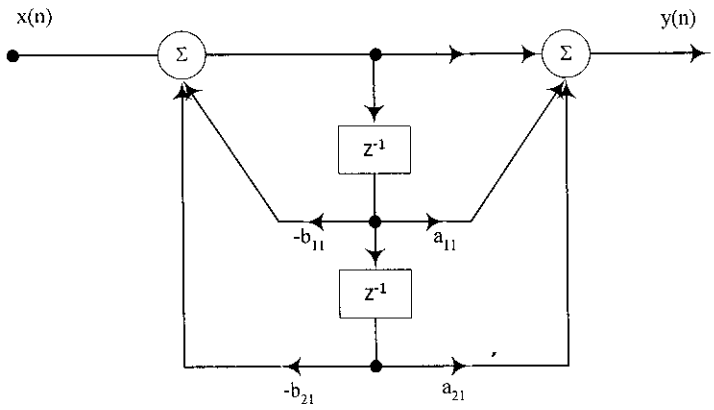
$$y(n) = c_2 \left(x(n) + \frac{a_1}{a_0} x(n-1) + \frac{a_2}{a_0} x(n-2) - \frac{b_1}{b_0} y(n-1) - \frac{b_2}{b_0} y(n-2) \right) \tag{2.20b}$$

ถ้าให้ $a_{11} = \frac{a_1}{a_0}$, $a_{21} = \frac{a_2}{a_0}$, $b_{11} = \frac{b_1}{b_0}$ และ $b_{21} = \frac{b_2}{b_0}$

จะได้ความสัมพันธ์ดังในสมการที่ 2.20c ซึ่งสมการนี้นำไปใช้ในโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

$$y(n) = c_2 \left[x(n) + a_{11} x(n-1) + a_{21} x(n-2) - b_{11} y(n-1) - b_{21} y(n-2) \right] \tag{2.20c}$$

เช่นเดียวกับกรณีของทรานสเฟอร์ฟังก์ชันอันดับหนึ่งที่จะต้องดึงค่าคงที่ออกก่อนเพื่อป้องกันการเกิด overflow แต่สำหรับทรานสเฟอร์ฟังก์ชันอันดับสองนี้ค่าคงที่ที่ดึงออกมาคือ C_2



รูปที่ 2.12 โครงสร้างที่ใช้ในการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

2.5 โครงสร้างของภาษาไดนามิกส์ซี (Dynamics C)

โปรแกรมที่ใช้ติดต่อกับคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) คือโปรแกรมไดนามิกส์ซี ดังนั้นจึงจำเป็นที่จะต้องทำความเข้าใจและศึกษาเกี่ยวกับโครงสร้างของภาษาไดนามิกส์ซี ซึ่งมีโครงสร้างคล้ายคลึงกับภาษาซีแต่ก็มีหลายส่วนที่แตกต่างกันดังจะอธิบายในรายละเอียดต่อไป

2.5.1 บทนำเกี่ยวกับไดนามิกส์ซี

ไดนามิกส์ซีคือระบบพัฒนารวมที่สามารถทำงานได้บนพีซี โดยถูกออกแบบมาให้ใช้กับ Z-world คอนโทรลเลอร์ และผลิตภัณฑ์ที่เกี่ยวกับระบบควบคุม ซึ่ง Z-world คอนโทรลเลอร์นี้เกิดมาจาก Zilog's Z180 ไมโครโพรเซสเซอร์และประกอบด้วยอนาล็อกอินพุท อนาล็อกเอาท์พุท ดิจิตอลอินพุท ดิจิตอลเอาท์พุท เอาท์พุทกระแสสูง การติดต่อแบบอนุกรม สัญญาณนาฬิกา และ ไทม์เมอร์ Z-world คอนโทรลเลอร์จะใช้กับโปรแกรมภาษาซี ที่เรียกว่า ไดนามิกส์ซี

สาเหตุที่เลือกใช้ภาษาซีเนื่องมาจากว่าตัวควบคุมที่สามารถโปรแกรมได้นั้น จะต้องมีความยืดหยุ่นมากที่สุดเพื่อที่จะได้พัฒนาระบบควบคุม ภาษาซีเป็นภาษาที่ได้รับการนิยมมากสำหรับการโปรแกรมระบบตรึงแน่น ซึ่งเป็นที่รู้จักกันอย่างแพร่หลาย ทั้งสามารถผลิตโค้ดที่กระชับและมีประสิทธิภาพได้ นอกจากนี้ภาษาซียังเป็นภาษาระดับสูง โค้ดสามารถพัฒนาได้เร็วกว่าภาษาแอสเซมบลี้ และยังสามารถโปรแกรมเป็นภาษาเครื่องได้อีกด้วย

2.5.2 ส่วนประกอบของไดนามิกส์ซี

ไดนามิกส์ซีประกอบด้วยฟังก์ชันดังนี้ Editing, Compiling, Linking, Loading, และ Debugging ไดนามิกส์ซีสามารถเขียนได้ง่ายใน text editor สามารถประมวลผลและแก้ส่วนที่ผิดได้ในซ็อสโค้ด และสามารถสร้างไฟล์ EPROM หรือไฟล์ที่สามารถดาวน์โหลดได้สำหรับโปรแกรมที่จะใช้ประมวลผลเพียงลำพังในตัวคอนโทรลเลอร์ พูลดาวน์เมนูและคีย์บอร์ดช็อตคัทของคำสั่งต่าง ๆ ช่วยทำให้ไดนามิกส์ซีมีประสิทธิภาพมากยิ่งขึ้น เนื่องจากได้มีการนำฟังก์ชันที่ได้รับการพัฒนาแล้วมารวมไว้ สามารถเปลี่ยนการทำงานจากฟังก์ชันหนึ่งไปสู่อีกฟังก์ชันหนึ่งได้โดยใช้คีย์สโตก นอกจากนั้นไดนามิกส์ซียังสนับสนุนการโปรแกรมภาษาแอสเซ็มบลี้ (Assembly) โดยสามารถใช้ภาษาแอสเซ็มบลี้รวมกับไดนามิกส์ซีได้ โดยไม่จำเป็นต้องออกจากส่วนของไดนามิกส์ซีก่อน

ส่วนการแก้ไขข้อบกพร่อง (Debugging) ไดนามิกส์ซีได้จัดให้มีหน้าต่างมาตรฐานอินพุท/เอาท์พุท (Standard I/O window) ทำให้โปรแกรมในคอนโทรลเลอร์สามารถพิมพ์ข้อความออกทางหน้าจอได้ หน้าต่างแอสเซ็มบลี้ (Assembly window) จะแสดงแอสเซ็มบลี้โค้ดที่ผ่านการคอมไพล์แล้ว หน้าต่างการมอง (Watch window) ช่วยให้โปรแกรมเมอร์สามารถพิมพ์หรือกำหนดส่วนที่แสดงผลทางจอภาพ หรือปรับตั้งค่าตัวแปร รวมทั้งการเรียกใช้ฟังก์ชันต่าง ๆ ได้ นอกจากนั้นยังมีหน้าต่างรีจิสเตอร์ (Register window) และหน้าต่างแสต็ค (Stack window) ให้ใช้งานอีกด้วย ดีบักเกอร์ (Debugger) ของไดนามิกส์ซีสามารถปรับตั้งค่าและลบค่าจุดหยุด (Break point) ของโปรแกรมได้

นอกจากนั้นไดนามิกส์ซียังมีส่วนที่เพิ่มเติมจากภาษาซีธรรมดาเช่น การใช้ตัวแปรร่วมกันและการป้องกันตัวแปร ซึ่งจะสนับสนุนระบบในเวลาจริง (Real-time) สามารถเขียนโปรแกรมบริการอินเทอร์รัปต์ (Interrupt Service Routines) ในภาษาซีไดนามิกส์ซี นอกจากนั้นไดนามิกส์ซียังสนับสนุน Real-time multitasking ด้วยฟังก์ชัน Real-time kernel และฟังก์ชัน Costatement ไดนามิกส์ซีประกอบด้วยฟังก์ชันไลบราลีมากมายที่อยู่ในรูปซ็อสโค้ด ฟังก์ชันไลบราลีเหล่านี้ต่างก็สนับสนุนการโปรแกรมแบบระบบเวลาจริง อินพุท/เอาท์พุทระดับแม็ชชีน การจัดเตรียมระบบข้อความและฟังก์ชันคณิตศาสตร์

ในเรื่องของความเร็ว ไดนามิกส์ซีจะคอมไพล์โดยตรงที่หน่วยความจำของ Z180 ฟังก์ชันและไลบราลีจะถูกคอมไพล์ ลิ้งค์ และดาวน์โหลดในทันที ถ้าพีซีที่ใช้เชื่อมต่อมีความเร็วสูง ไดนามิกส์ซีสามารถคอมไพล์ซ็อสโค้ดได้มากกว่า 250 บรรทัดต่อวินาที และสร้างแม็ชชีนโค้ดประมาณ 2,500 ไบต์ต่อวินาที ถ้าเป็นโปรแกรมใหญ่ (ในที่นี้หมายถึงโปรแกรมที่มีซ็อสโค้ดประมาณ 8,000 บรรทัด) อาจจะสร้างแม็ชชีนโค้ดได้ 80 กิโลไบต์ และใช้เวลา 30 วินาทีในการคอมไพล์และดาวน์โหลด ส่วนแอพพลิเคชั้นโค้ดอาจจะมีเพียง 400 บรรทัด

2.5.3 ข้อแตกต่างของภาษาไดนามิกส์ซีกับภาษาซี

ไดนามิกส์ซีแตกต่างจากภาษาซีแบบเดิมไม่ว่าจะทำการรันด้วยพีซี (PC) หรือภายใต้ระบบยูนิกซ์ (UNIX) ด้วยการกระตุ้นเพื่อให้เกิดความแตกต่างที่ดีกว่า โดยช่วยให้ผู้ใช้สามารถเขียนซอฟแวร์ควบคุมระบบตรึงแน่นที่มีความน่าเชื่อถือมากที่สุดได้

ข้อแตกต่างทั้งหมดของภาษาไดนามิกส์ซีจากภาษาซีสามารถสรุปได้ดังนี้

- ตัวแปรที่ถูกกำหนดขึ้นเมื่อมีการประกาศตัวแปรจะถูกนำไปเก็บไว้ในส่วน ROM ส่วนนี้คือ Error ที่พยายามเปลี่ยนแปลง
- Class ปกติที่ใช้ในการเก็บค่า (Storage class) คือ Static ไม่ใช่ Auto
- ไม่มีไดเรคทีฟ ***#include*** แต่จะใช้ไดเรคทีฟ ***#use*** แทน
- ไดนามิกส์ซีจะไม่สนับสนุน Enumerated types
- ***Extern*** กับ ***Register*** จะมีความหมายเปลี่ยนไป
- Function chaining ทำให้สามารถรวมส่วนที่พิเศษของโค้ดในหนึ่งฟังก์ชันหรือมากกว่าได้ เมื่อถูก Execute ส่วนที่เป็นของฟังก์ชันนั้นทั้งหมดก็จะถูก Execute ด้วย ทำให้ซอฟต์แวร์สามารถทำการกำหนดค่าเริ่มต้น กู้คืนข้อมูล หรือหน้าที่อื่นได้ตามต้องการ
- "***Costatements***" ทำให้กระบวนการหลายกระบวนการสามารถจำลองผลได้ในโปรแกรมเดียว
- ไดนามิกส์ซีทำให้ผู้เขียนโปรแกรมสามารถเขียนโปรแกรมบริการอินเทอร์รัปต์ในภาษาซีได้
- ไดนามิกส์ซีสนับสนุนโค้ดภาษาแอสเซมบลี้
- ไดนามิกส์ซีมีคีย์เวิร์ด ***Shared*** และ ***Protected*** ซึ่งจะช่วยป้องกันข้อมูลไม่ให้หายได้
- มีลักษณะที่ให้โปรแกรมเมอร์สามารถใช้หน่วยความจำในส่วนที่มีการขยายออกได้
- มี 2 รูปแบบสำหรับ Argument passing โดยใช้ IX index register กับการใช้ Stack pointer (SP)
- มีโครงสร้างของ Subfunction ที่จะช่วย Optimize โค้ดที่มีการใช้งานบ่อย ๆ

2.5.4 การใช้งานโปรแกรมไดนามิกส์ซีในเบื้องต้น

ก่อนใช้งานโปรแกรมไดนามิกส์ซีจะต้องมีการติดตั้งโปรแกรมก่อน สิ่งที่ต้องทราบก็คือข้อกำหนดต่าง ๆ ก่อนการติดตั้ง ดังจะกล่าวในรายละเอียดข้างล่างนี้

2.5.4.1 ข้อกำหนดก่อนการติดตั้งโปรแกรมไดนามิกส์ซี

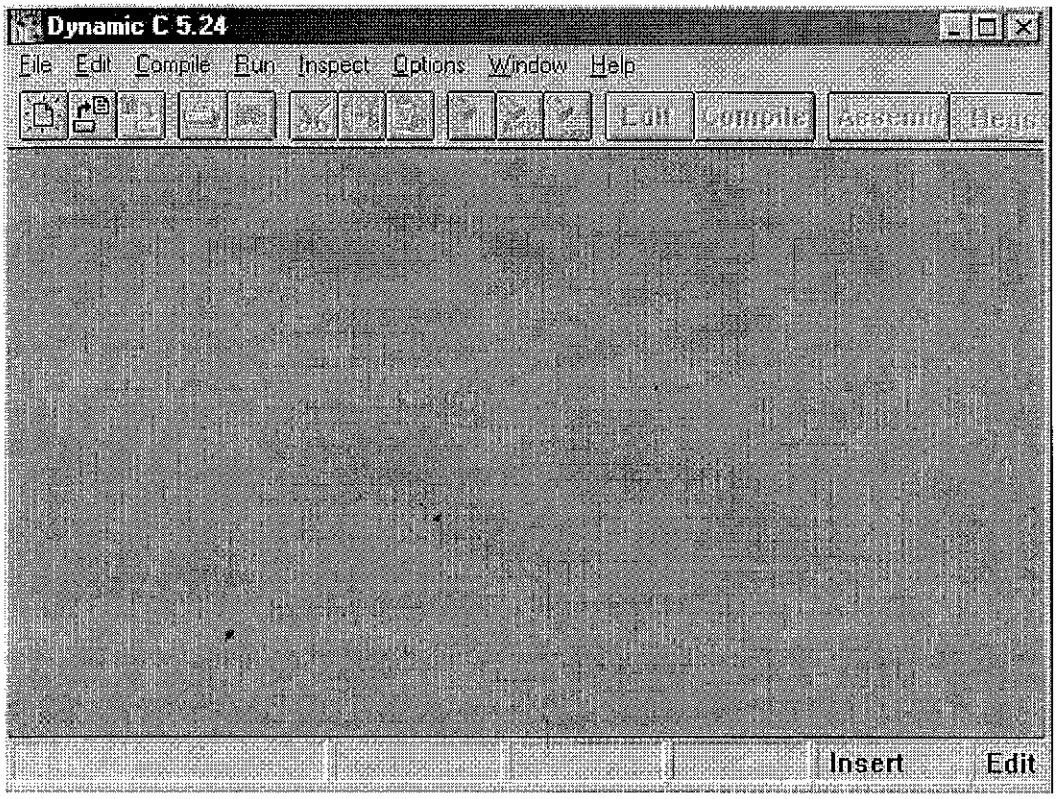
- มีเนื้อที่ว่างในฮาร์ดดิสก์อย่างน้อย 4 เมกะไบต์
- เครื่องพีซีต้องมีโปรเซสเซอร์ตั้งแต่ 386 ขึ้นไป
- มีการติดตั้งวินโดวส์ 3.1, วินโดวส์ เอ็นที หรือ วินโดวส์ 95 แล้ว
- มีเนื้อที่ว่างในหน่วยความจำอย่างน้อย 4 เมกะไบต์
- มีพอร์ทอนุกรมที่ว่างอย่างน้อย 1 พอร์ทเพื่อใช้ในการติดต่อกับคอนโทรลเลอร์บอร์ด

2.5.4.2 การติดตั้งโปรแกรมไดนามิกส์ซี

แผ่นดิสก์ที่ใช้ในการติดตั้งโปรแกรมมีอยู่ด้วยกัน 2 แผ่น โดยทำตามขั้นตอนดังนี้

1. นำแผ่นติดตั้งแผ่นที่ 1 ใส่ในไดร์ฟ A
2. พิมพ์ A:\setup.exe
3. เมื่อทำไปได้สักระยะหนึ่งโปรแกรมการติดตั้งก็จะเตือนให้เปลี่ยนเป็นแผ่นติดตั้งแผ่นที่ 2

เมื่อทำการติดตั้งเสร็จเรียบร้อยแล้วจะมีไอคอนของโปรแกรมไดนามิกส์ซีปรากฏขึ้น เมื่อคลิกเข้าไปที่ตัวโปรแกรมจะมีหน้าต่างดังรูปที่ 2.13



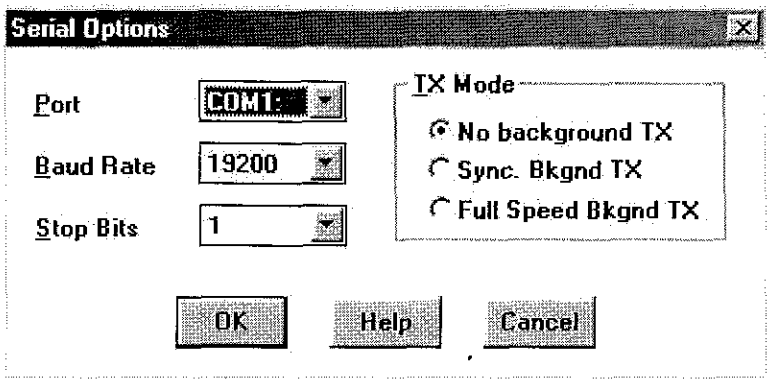
รูปที่ 2.13 หน้าต่างของโปรแกรมไดนามิกส์ซี

2.5.4.3 การปรับตั้งค่าเพื่อให้โปรแกรมติดต่อกับบอร์ด

ก่อนที่จะใช้งานโปรแกรมนี้จะต้องทำการปรับตั้งค่าต่าง ๆ เพื่อให้โปรแกรมไดนามิกส์ซีสามารถติดต่อกับคอนโทรลเลอร์บอร์ดได้ ถ้าไม่เช่นนั้นแล้วโปรแกรมนี้จะไม่สามารถติดต่อกับบอร์ดได้เลย

ทำการปรับตั้งค่าที่ Serial Options

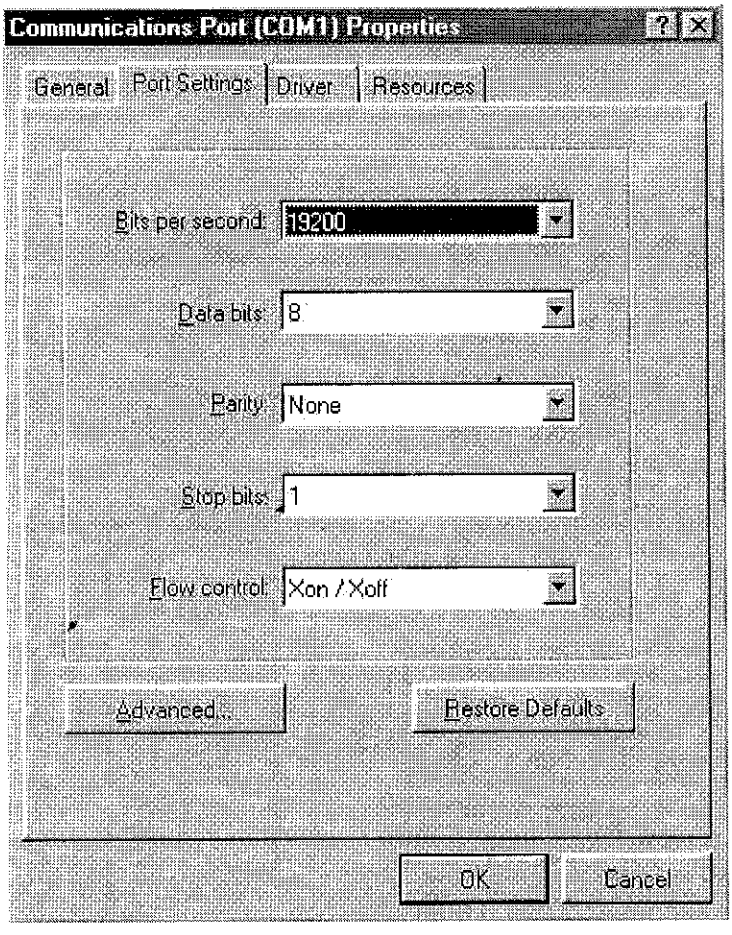
ในขั้นตอนที่ทำการติดตั้งโปรแกรมไดนามิกส์ซีจะมีให้เลือก Serial port ที่ใช้ในการติดต่อซึ่งในที่นี้เลือกที่พอร์ท Com1 และจากข้อกำหนดของคอนโทรลเลอร์บอร์ดจึงต้องทำการปรับตั้งค่า Baud rate ที่ 19,200 baud ดังรูปที่ 2.14



รูปที่ 2.14 การปรับตั้งค่าที่ Serial Options

ทำการปรับตั้งค่าที่ Communications port (Com1) properties

ในส่วนการปรับตั้งค่านี้จะต้องเข้าไปที่ Control Panel ในวินโดวส์ แล้วเข้าไปที่ System จะปรากฏหน้าต่างของ System Properties แล้วคลิ้กที่ Device Manager เมื่อเข้ามาใน Device Manager แล้วให้เลือก View devices by type ไปที่ Ports (Com & LPT) แล้วคลิ้กที่ Communications Port (Com1) จะปรากฏหน้าต่างของ Communications Port (Com1) Properties เลือกที่ Port Settings แล้วทำการปรับตั้งค่าตามรูปที่ 2.15 เหตุที่ต้องทำการปรับตั้งค่าตรงส่วนนี้ด้วยก็เพราะว่าเพื่อให้สามารถติดต่อกับบอร์ดได้นั่นเอง เนื่องจากว่าจะต้องปรับตั้งค่าให้เข้ากันทั้งสามส่วนคือ คอนโทรลเลอร์บอร์ด โปรแกรมไดนามิกส์ซี และพีซี โดยได้ทำการต่อ J7 ของบอร์ดเข้ากับ Com1 ของพีซีแล้ว



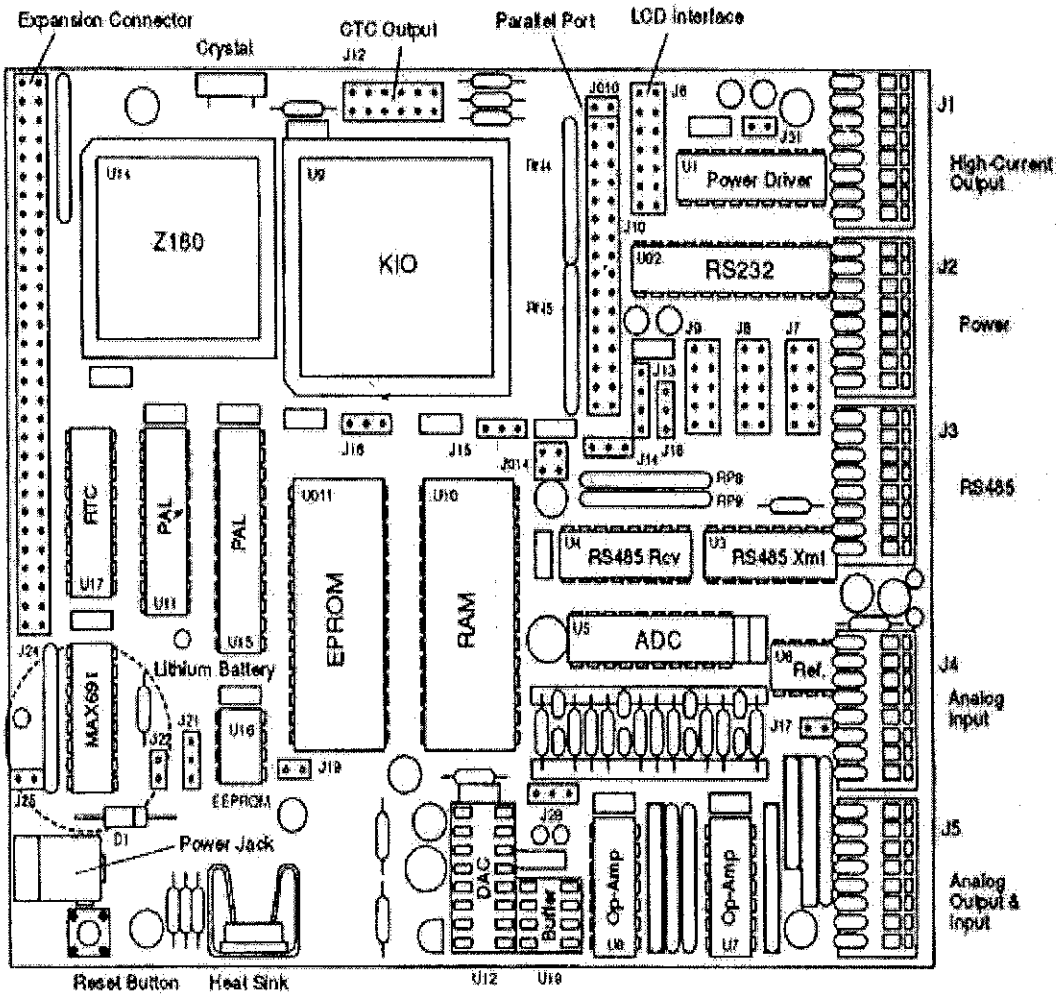
รูปที่ 2.15 การปรับตั้งค่าที่ Communications port (Com1) properties

หลังจากที่ทำการปรับตั้งค่าต่าง ๆ เสร็จเรียบร้อยแล้ว จะสามารถทำการติดต่อระหว่างคอนโทรลเลอร์บอร์ดกับพีซีได้แล้วโดยผ่านโปรแกรมไดนามิกส์ซี ในส่วนของการใช้งานกับโปรแกรมการอนุวัตตัวชดเชยจะกล่าวในบทที่ 3

2.6 คอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

คอนโทรลเลอร์บอร์ดนี้มีชื่อเรียกย่อ ๆ ว่า Little Giant เป็นผลิตภัณฑ์หนึ่งของบริษัท Z-World ซึ่งเป็นคอนโทรลเลอร์บอร์ดขนาดเล็ก โดยบอร์ดนี้ใช้งานกับโปรแกรมไดนามิกส์ซีเวอร์ชัน 5.24 (Dynamics C 5.24) เป็นเวอร์ชันมาตรฐานเนื่องจาก Run ด้วย DCW.EXE ภายใต้วินโดวส์ โดยมีลักษณะที่สำคัญดังจะกล่าวในรายละเอียดต่อไปนี้

2.6.1 ส่วนประกอบของคอนโทรลเลอร์บอร์ด



รูปที่ 2.16 ส่วนประกอบของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

- บอร์ดขนาด 5.6" x 4.8" หรือ 142x122 มิลลิเมตร
- Z180 ไมโครโปรเซสเซอร์ รันที่ 12.288 MHz ด้วย partial wait state
- มีส่วนการตรวจจับและเตือนการลดลงของ Power โดยจะมี นอนแมสเคเบิลอินเทอร์รัปต์ (non-maskable interrupt) เกิดขึ้นเมื่อ power ลดลงต่ำกว่าระดับปกติ โดยโปรแกรมจะใช้เวลาประมาณ 2-3 มิลลิวินาที เพื่อที่จะทำการปิดโปรแกรม
- มี Watchdog timer ถ้าปรับตั้งค่าให้ watchdog ทำงาน (enable) จะทำการปรับตั้งค่าบอร์ดใหม่โดยอัตโนมัติถ้าระบบ Watchdog timer ทำงานผิดปกติไป นี่คือ

ลักษณะที่แสดงถึงความน่าเชื่อถืออย่างหนึ่งที่จะช่วยในการกู้คืนจากความผิดพลาดของซอฟต์แวร์หรือฮาร์ดแวร์

- มี 256 Kbytes EPROM ซึ่งยอมรับ ROMs ทั้ง 28-pin หรือ 32-pin
- มี battery-backed static RAM 512 Kbytes ซึ่งสามารถใช้ RAMsได้ทั้ง 28-pin หรือ 32-pin แบตเตอรี่ลิเทียมบนบอร์ดมีอายุประมาณ 10 ปี
- 512 bytes EEPROM ครึ่งหนึ่งของหน่วยความจำ EEPROM (256 bytes) สามารถป้องกันไม่ให้มีการลบได้โดยใช้ jumper 21 (J21) ซึ่งหน่วยความจำส่วนนี้ใช้สำหรับการเก็บรักษาค่า Baud rate ที่ปรับตั้งค่าเอาไว้, ค่าที่คาลิเบรท (calibrate) หรือค่าคงที่การปรับตั้งค่าต่าง ๆ นอกจากนั้นผู้ใช้ยังสามารถเปลี่ยน EEPROM ใหม่ได้ถึง 2048 bytes
- Battery-backed real-time clock มีพื้นฐานมาจากชิพ Epson 72421 สามารถรันได้ถึง 10 ปีบนแบตเตอรี่ลิเทียม
- มี 4 พอร์ทอนุกรม (serial ports) โดยที่มี 2 พอร์ทที่มีพื้นฐานมาจากพอร์ทที่สร้างขึ้นภายใน Z180 ส่วนอีก 2 พอร์ทมีรากฐานมาจากพอร์ทอนุกรมของ Zilog KIO peripheral chip ซึ่งพอร์ท KIO เข้ากันได้กับ Zilog SIO และสามารถติดต่อได้ทั้งแบบซินโครนัส (Synchronous) และอะซินโครนัส (Asynchronous) ซึ่งระดับศักย์ไฟฟ้าทั้งแบบ RS485 (balanced) และ RS232 (unbalanced) สามารถหาได้ไม่ยาก
- มี 1 พอร์ทขนาน 16 บิตซึ่งมีรากฐานมาจาก Zilog KIO/PIO พอร์ทนี้สามารถเชื่อมต่อโดยตรงได้กับอุปกรณ์ชนิดอื่น ๆ หลายชนิด
- มี 1 พอร์ทศักย์/กระแสสูงซึ่งอยู่บนพื้นฐานของ ***Sprague 5841 type driver*** ช่องรับสัญญาณ (channels) ที่เห็นในตัวบอร์ดในรูป 2.16 สามารถนำมาต่อขนานกันเพื่อให้กระแสสูงขึ้นได้
- มี Liquid crystal display interface LCDs มาตรฐาน เช่น Optrex part number DMC20481 สามารถต่อเข้าโดยตรงกับคอนเนคเตอร์ที่มี 14-pin
- ตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (A/D converter) 8 ช่องรับสัญญาณ (channels) ที่มีตัวขยายอินพุทรวมอยู่ด้วย ซึ่ง A/D converter นี้มีความละเอียด (resolution) ที่ 10 บิตถ้าเป็น LTC1094 หรือ 12 บิตถ้าเป็น LTC1294 โดยที่ค่าความละเอียดนี้สามารถเปลี่ยนแปลงได้ คอนโทรลเลอร์บอร์ดตัวนี้สนับสนุน single-ended หรือ differential input ด้วย

- ตัวแปลงสัญญาณดิจิตอลเป็นอนาล็อก DAC (Digital to Analog Converter) มีความละเอียดที่ 12 บิตและเอาท์พุทอยู่ในช่วง 0-2.5 โวลต์ ส่วน Maxim AD7543 DAC จะใช้กับ MAX400 buffer amplifier สำหรับเอาท์พุทที่มากที่สุดที่ 2 mA
- ใช้กับแหล่งจ่ายที่เป็นเชิงเส้น (linear power supply) ที่สามารถจ่ายแรงดันได้ตั้งแต่ 9-16 โวลต์สำหรับ wall transformer หรือแหล่งจ่ายอื่น
- คอนโทรลเลอร์บอร์ด Little Giant นี้มีแหล่งจ่ายในตัวแบบเชิงเส้น –5V (10 mA) และจะมีสภาวะปกติที่ –10 V จาก driver RS232 ซึ่งจะมีตัวแปลงแบบไฟตรง-ไฟตรง ที่สร้างขึ้นภายในบอร์ด

2.6.2 สิ่งที่ควรทำก่อนการใช้งานบอร์ด

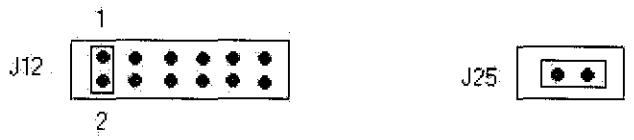
ก่อนที่จะเริ่มใช้งานคอนโทรลเลอร์บอร์ดควรมีการปรับตั้งค่าต่าง ๆ เหล่านี้ก่อนเพื่อให้สามารถติดต่อกับบอร์ดโดยใช้โปรแกรมไดนามิกส์ซีผ่านพีซีได้

- ทำการเชื่อมต่อ J7 (Jumper 7) ที่อยู่บนคอนโทรลเลอร์บอร์ดกับพอร์ทอนุกรมคอม 1 (Com1) ของพีซี โดยใช้สายเคเบิลที่ให้มาพร้อมกับบอร์ด และจะต้องใช้หม้อแปลงต่อเข้ากับพาวเวอร์แจ็ค (Power jack) ของตัวบอร์ดเพื่อจ่ายไฟให้กับบอร์ด เมื่อทำตามขั้นตอนนี้แล้วจะเห็นว่า reset LED จะกระพริบและ LD2 จะสว่าง
- จากคู่มือเนื่องจากทำการเชื่อมต่อที่ J7 (RS232) ดังนั้นจึงต้องทำการปรับตั้งค่า Hardware reset เป็นโหมดศูนย์ (Mode 0) โดยการเชื่อมต่อ J25 และ J12 pin 1-2 โดยจะเป็นการบูทบอร์ดโดยไม่ต้องสนใจ EEPROM เพื่อติดตั้ง Baud rate ที่เข้ากันกับ Clock crystal ในที่นี้ปรับตั้งค่า Baud rate ที่ 19,200 baud (เนื่องจากถ้าปรับตั้งค่าที่ค่าอื่นแล้วไม่สามารถเข้ากับบอร์ดได้เลย เห็นได้จากจะมีไดอะล็อกบอกซ์ขึ้นมาเตือนว่า “Target communication error” หรือ “Target not responding” แสดงการปรับตั้งค่า J25 และ J12:1-2 ในรูปที่ 2.17
- เมื่อต้องการใช้ Watchdog timer จะต้องทำการเชื่อมต่อ J22 (ในที่นี้ทำการปรับตั้งค่า J22 เพื่อรองรับการปรับตั้งค่า timer)
- ส่วน EEPROM จะปรับตั้งค่าที่ J21 มีการปรับตั้งค่า 2 รูปแบบซึ่งก็คือ
  - ถ้าทำการเชื่อมต่อ J21:1-2 จะหมายถึงการป้องกันไม่ให้มีการเขียน EEPROM ส่วนบน
  - ถ้าทำการเชื่อมต่อ J21:2-3 จะหมายถึงการยอมให้มีการเขียน EEPROM ส่วนบนได้

ในที่นี้จะทำการปรับตั้งค่าโดยป้องกันไม่ให้มีการเขียนที่ส่วนบนของ EEPROM จะยอมให้มีการเขียน EEPROM ส่วนบนเมื่อจำเป็นเท่านั้น

- ในส่วนของการปรับตั้งค่า RAM จะทำการปรับตั้งค่าที่ J15 โดยจะมีการปรับตั้งค่า 2 รูปแบบคือ
  - กรณี Static RAM น้อยกว่า 256K bytes จะทำการเชื่อมต่อ J15:1-2
  - กรณี Static RAM มากกว่าหรือเท่ากับ 256K bytes จะทำการเชื่อมต่อ J15:2-3

  ในที่นี้จะทำการปรับตั้งค่าที่ Static RAM น้อยกว่า 256K bytes เนื่องจาก Little Giant มี SRAM 32K bytes
- ทำการเชื่อมต่อ J16:2-3 เนื่องจาก EPROM > 32K bytes โดยใช้ 27C020 256K 32 pins
- จากการปรับตั้งค่า J16:2-3 ทำให้ต้องทำการเชื่อมต่อ J20:1-2 เนื่องจาก J20 เป็นจัมเปอร์ที่ใช้ในการกำหนดขนาดของ EPROM
- เชื่อมต่อ J17 ซึ่งจัมเปอร์นี้ช่วยให้มีการอีนาเบิลตัวต้านทานสำหรับออปแอมป์ด้านอินพุท
- เชื่อมต่อ J19 เนื่องจากใช้ไฟเลี้ยงภายในบอร์ด



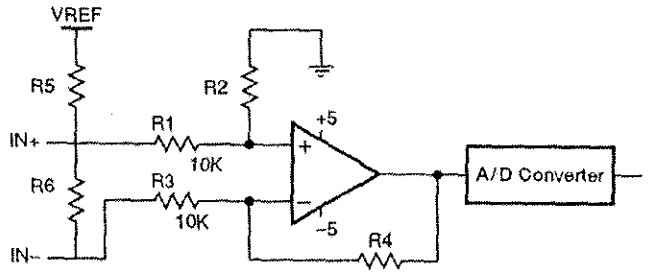
รูปที่ 2.17 การปรับตั้งค่า J25 และ J12:1-2 บนคอนโทรลเลอร์บอร์ด

2.6.3 การกำหนดค่าเริ่มต้นให้กับส่วนประกอบต่าง ๆ ของคอนโทรลเลอร์บอร์ดเพื่อใช้ในการอนุวัตตัวชดเชย

ในส่วนนี้จะอธิบายถึงข้อกำหนดที่ควรทราบและรายละเอียดของอุปกรณ์ที่สำคัญรวมไปถึงการกำหนดค่าเริ่มต้นให้กับอุปกรณ์เหล่านั้นก่อนที่จะใช้งานบอร์ดในการอนุวัตตัวชดเชย

2.6.3.1 Signal Conditioning

เป็นการกล่าวถึงข้อกำหนดเกี่ยวกับการขยายสัญญาณ ดูรูปที่ 2.18 ประกอบการพิจารณา



รูปที่ 2.18 วงจรของตัวขยายสัญญาณก่อนที่สัญญาณอินพุทจะเข้าไปยังส่วนของ ADC

อัตราขยาย (gain, g) สามารถหาได้จากความสัมพันธ์ในสมการที่ 2.21

$$g = \frac{IN+}{IN-} = \frac{R_2}{R_1 + R_2} \div \frac{R_3}{R_4 + R_3} = \frac{R_2(R_4 + R_3)}{R_3(R_1 + R_2)} \tag{2.21}$$

ซึ่งจะมีการหาค่าอัตราขยายอยู่ด้วยกัน 4 รูปแบบดังนี้

<u>รูปแบบที่1</u>

ช่องรับสัญญาณนั้นต่ออยู่กับตัวขยายสัญญาณที่ปลายด้านนอนอินเวอร์สติง (Non-inverting) โดยไม่มี R_5 และ R_6 อินพุทลบต่ออยู่กับกราวด์ อินพุทบวกใช้สุ่มตัวอย่าง และ R_2 มีค่าใหญ่มาก ประมาณ 100K หรือสามารถตัดทิ้งได้ เมื่อเป็นดังรูปแบบนี้แล้ว สามารถหาอัตราขยายได้จากสูตรในสมการที่ 2.22

$$g = R_2 \times (R_4 + R_3)/(R_3 \times (R_1 + R_2)) \tag{2.22}$$

กรณีที่ไม่คิด R_2 จะได้ความสัมพันธ์ดังนี้

$$g = (R_4 + R_3)/R_3 = R_4/R_3 + 1 \tag{2.23}$$

รูปแบบที่2

ช่องรับสัญญาณนั้นต่ออยู่กับตัวขยายสัญญาณที่ปลายด้านอินเวอร์สติง (Inverting) โดยไม่มี R_5 และ R_6 อินพุทบวกต่ออยู่กับกราวด์ อินพุทลบใช้สุ่มตัวอย่าง จะสามารถหาอัตราขยายได้ดังนี้

$$g = R_4 / R_3 \tag{2.24}$$

R_2 สามารถตัดทิ้งได้โดยที่ไม่มีผลใด ๆ

รูปแบบที่3

ช่องรับสัญญาณใช้วัดความต้านทาน (Resistance) โดยที่ R_6 คือตัวต้านทานที่ไม่ทราบค่า อินพุทลบต่ออยู่กับกราวด์ สามารถหาตัวต้านทานที่ไม่ทราบค่าได้จาก

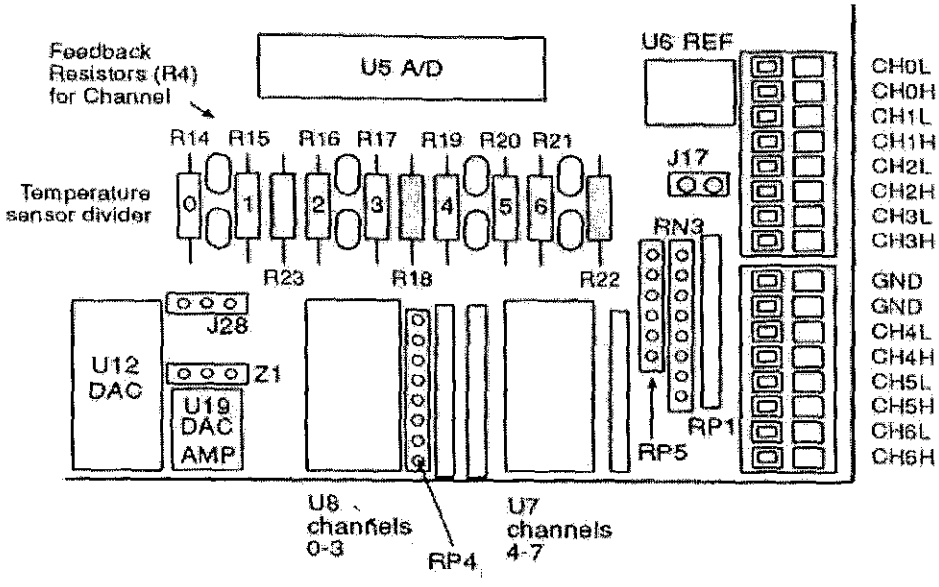
$$R_6 = R_5 \times (REF - V)/V \tag{2.25}$$

โดยที่ V คือแรงดันที่ขั้วอินพุทบวก อัตราขยายจากขั้วอินพุทบวกจะเหมือนรูปแบบที่ 1 ส่วนแรงดันอ้างอิง (REF) จะเป็น 2.5V หรือเป็นไฟเลี้ยง +5V ถ้าไม่ต้องการความถูกต้องมากนัก

รูปแบบที่4

ช่องรับสัญญาณใช้วัดกระแส โดยที่รู้ค่า R6 และเมื่อต้องการวัดกระแสผ่านตลอด ซึ่งหาได้จาก $i = V / R_6$ การปรับตั้งค่าเหมือนกับรูปแบบที่3

ทั้งสี่รูปแบบที่กล่าวมา สามารถทำการติดตั้งได้โดยการเปลี่ยน ตัวต้านทาน R_1, R_2, R_3, R_4 และ R_5 ตำแหน่งต่าง ๆ แสดงดังรูปที่ 2.19



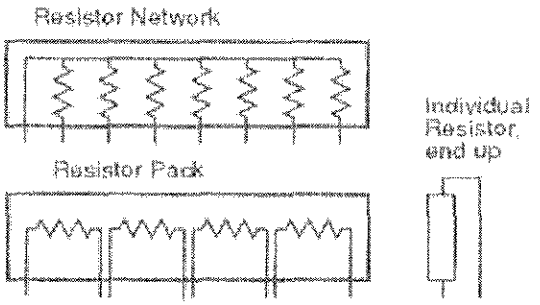
รูปที่ 2.19 ตำแหน่งของตัวต้านทานบนบอร์ด

โดยปกติชื่อของตัวต้านทานในแผนภาพวงจรกับในแผงวงจรจริงบนบอร์ดจะแตกต่างกัน ดังแสดงในตารางที่ 2.3

ตารางที่ 2.3 ชื่อของตัวต้านทานบนบอร์ดกับบนแผนภาพวงจร

Channel	R_1	R_2	R_3	R_4	R_5
0	RP1A	RP4A	RP7A	R14	RN3pin2
1	RP7B	RP4B	RP7C	R15	RN3pin3
2	RP2A	RP4C	RP2B	R16	RN3pin4
3	RP2C	RP4D	RP2D	R17	RN3pin5
4	RP3B	RP5A	RP3A	R19	RN3pin6
5	RP3C	RP5B	RP3D	R20	RN3pin7
6	RP1C	RP5C	RP1D	R21	RN3pin8

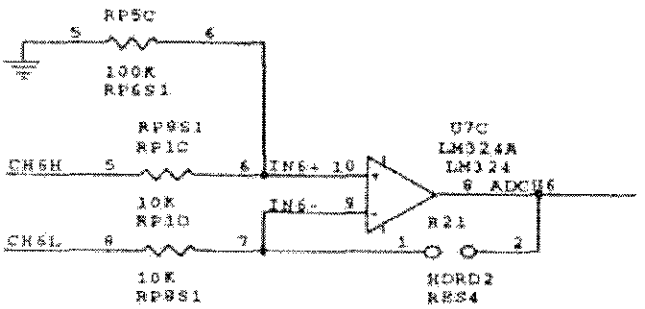
ตัวต้านทานแบบแพ็ค แบบเครือข่าย และแบบเดี่ยว ซึ่งตัวต้านทานแบบต่าง ๆ เหล่านี้ สามารถทำการปรับตั้งค่าได้โดยซอคเก็ตที่อยู่ในบอร์ด โดยปกติ RN3 จะไม่ได้ทำการติดตั้งมาให้ จะใช้ J17 เพื่อติดตั้ง RN3 (R5s) เพื่อเชื่อมต่อกับ VCC



รูปที่ 2.20 ตัวต้านทานแบบต่าง ๆ

จากแผนภาพในรูปที่ 2.21 ซึ่งเป็นแผนภาพจริงของวงจรบนบอร์ดที่เลือกใช้ (ch6) รูปแบบของการหาค่าอัตราขยายจึงเป็นรูปแบบที่สอง ซึ่งค่าอัตราขยายที่หาได้คือ

$$g = R_4 / R_3 = \frac{22.1\ k\Omega}{10\ k\Omega} = 2.21$$



รูปที่ 2.21 รูปแบบของวงจรขยายสัญญาณก่อนที่สัญญาณอินพุทจะผ่านไปยัง ADC

2.6.3.2 ตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (Analog to Digital Converter, ADC)

ตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล คือ U5 ในรูปที่ 2.16 โดยจะมี 8 ช่องรับสัญญาณอินพุท มีตัวขยายในตัวที่ไว้ปรับค่าให้กับ ADC โดยที่ช่องรับสัญญาณที่7 (ch7) จะใช้สำหรับวัดอุณหภูมิของสัญญาณที่จ่ายโดยชิพอ้างอิง LTC1019 ส่วน ch6 จะใช้สำหรับอินพุทหรือเป็นบัฟเฟอร์เพื่อเตรียมสำหรับอินพุทอ้างอิงจากภายนอก แต่อย่างไรก็ตามก็ต้องขึ้นอยู่กับการย้ายตัวเชื่อมต่อ Wago และการเพิ่มของ headers

A/D converter ใช้ชิพเดี่ยว (single chip) ของ Linear Technologies LTC1094 (10 บิตที่ 120 ไมโครวินาที) หรือ LTC1294 (12 บิตที่ 120 ไมโครวินาที) ฟังก์ชันไลบราลีที่ใช้อ่าน A/D converter คือ

```
int ad_rd10 (int channel)        //10-bit 1094
int ad_rd12 (int channel)        //12-bit 1294
```

ถ้าช่องรับสัญญาณอยู่ในช่วง 0-7 ฟังก์ชันจะส่งค่าไบโพลาร์ (bipolar) อยู่ในช่วง –2048 ถึง +2047โดยแสดงถึงแรงดันอินพุทตั้งแต่ –2.500 V ถึง +2.499 V แต่ถ้าเป็นช่องรับสัญญาณ 8 จะทำการแปลงให้เป็นยูนิโพลาร์ (unipolar) ซึ่งจะส่งค่าในช่วง 0 ถึง 4095 โดยแสดงแรงดันอินพุทตั้งแต่ 0.000 V ถึง 2.499 V โดยที่ค่าที่ส่งกลับมาเหล่านั้นจะถูกสเกลไม่ว่าจะใช้ตัวแปลงแบบ 10 บิตหรือ 12 บิต แต่สำหรับตัวแปลงแบบ 10 บิตนั้นจะมีอย่างน้อยที่สุด 2 บิตที่ส่งค่าเป็นศูนย์ เนื่องจากค่าที่ส่งกลับมาจะเป็นค่าแบบ 12 บิต

ในการอนุวัตจะเลือกความละเอียด ADC ที่ 12 บิต (LTC1294) ดังนั้นจึงต้องทำการปรับตั้งค่าฟังก์ชันไลบราลีเป็น

```
int ad_rd12 (int channel)        //12-bit 1294
```

โดยนำสัญญาณอินพุทเข้าที่ช่องรับสัญญาณ 6 (ch6) ดังนั้นค่าที่ต้องกำหนดให้กับโปรแกรมจึงเป็น

```
ad_rd12 (6);      //12-bit 1294
```

2.6.3.3 ตัวแปลงสัญญาณดิจิตอลเป็นอนาล็อก (Digital to Analog Converter, DAC)

ก่อนที่จะทำการติดตั้ง DAC จะต้องมีการติดอุปกรณ์เหล่านี้บนบอร์ดเสียก่อน ซึ่งได้แก่

- U12 Maxim AD7543 D/A converter
- U19 Maxim OP07 operational amplifier
- Z1 National LM285 voltage reference

ส่วน R3 กับ C012 ควรจะติดตั้งที่บริษัท

DAC จะเชื่อมต่อกับ jumper J28 ไปยัง connector J5-1 ถ้าทำการปรับตั้งค่าที่ J28:1-2 (J28 pin 1-2) บอร์ดจะถูกปรับตั้งค่าให้เป็น DAC output บน J5 แต่ถ้าปรับตั้งค่าที่ J28:2-3 (J28 pin 2-3) จะทำให้ J5-1 มีสถานะเป็นกราวด์ (ground) และ DAC output จะไม่ถูกส่งไปยัง J5

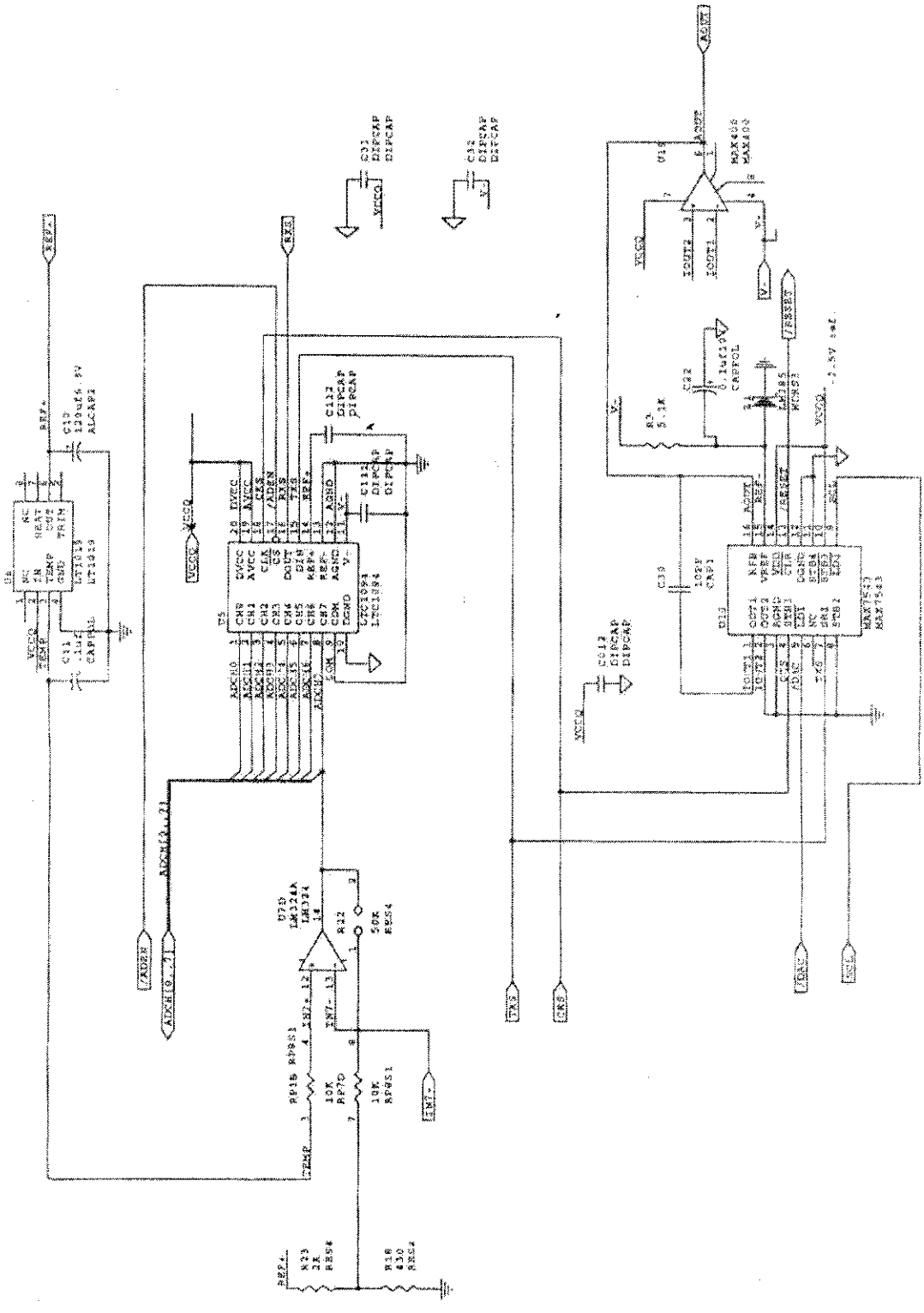
ส่วนฟังก์ชันที่ใช้กับ DAC คือ wdac ใน Little Giant Library ซึ่งฟังก์ชันนี้จะใช้เขียนค่าที่เป็นดิจิตอล 12 บิต ไปยัง DAC ค่าของเอาท์พุทจะอยู่ระหว่าง 0 ถึง 4095 ซึ่งค่าช่วงนี้จะอยู่ในช่วงแรงดันเอาท์พุทระหว่าง 0 V ถึง 2.499 V (เป็นเพียงค่าโดยประมาณ) ในงานที่ต้องการความถูกต้องสูงควรมีการปรับอัตราขยายทางซอฟต์แวร์ เนื่องจากการอ้างอิงค่าแรงดันจะมีความผิดพลาดมากถึง 80 counts และ DAC จะมีอัตราขยายความผิดพลาดมากที่สุดถึง 15 counts ในการแก้ไขข้อผิดพลาด DAC ควรทำงานด้วยความถูกต้อง 1-2 บิต (bit accuracy) ในช่วงและอุณหภูมิที่กำหนด นอกจากนั้นความสามารถของกระแสขับเคลื่อนเอาท์พุทมีค่าประมาณ 2 มิลลิแอมแปร์

ส่วนรูปแบบการใช้ฟังก์ชันไลบราลีของ DAC คือ

void wdac (int value)

โดยที่ value คือ ค่าที่ต้องการเขียนลงใน DAC ซึ่ง value ควรมีค่าอยู่ในช่วง 0 ถึง 4095 ซึ่งสามารถแปลงเป็นค่าแรงดันได้ด้วยความสัมพันธ์ในสมการที่ 2.26

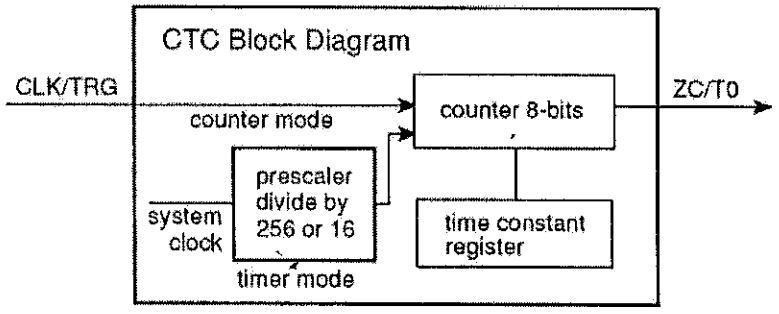
$$output\ voltage = 2.5 * value / 4096 \quad volts \tag{2.26}$$



รูปที่ 2.22 วงจรของตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (ADC) และตัวแปลงสัญญาณดิจิตอลเป็นอนาล็อก (DAC)

2.6.3.4 CTC Counter/Timers

CTC ของ KIO chip มี Counter/timer 4 หน่วย มี 2 ไทม์เมอร์ (Timer) ที่สามารถโปรแกรมได้ (เรียกว่า PRTs) ซึ่งเป็นส่วนหนึ่งของ Z180 ซึ่งจะอธิบายในคู่มือของ Z180



รูปที่ 2.23 แผนภาพของ CTC counter/timer

เคาท์เตอร์/ไทม์เมอร์แต่ละตัวจะมีตัวนับและรีจิสเตอร์ค่าคงที่ทางเวลา (Time-constant) ผู้ใช้สามารถโหลดค่าคงที่ทางเวลาและตัวนับได้ โดยตัวนับจะนับลงจนถึงศูนย์ พอถึงจุดนี้กระบวนการต่าง ๆ จะเริ่มทำซ้ำ ซึ่งก็คือจะสร้างอินเทอร์รัปต์หรือพัลส์เป็นรายคาบ ซึ่งค่าคงที่ทางเวลานี้จะมีค่าเริ่มตั้งแต่ 0-255 นอกจากนั้นสัญญาณนาฬิกาที่เข้ามาสามารถหาได้จากสัญญาณนาฬิกาของระบบ (System clock) โดยผ่านการทำพรีสเกล หรือมาจากสัญญาณนาฬิกานอกระบบ

CTC จะมีด้วยกัน 3 โหมด โดย 2 โหมดจะเป็นเคาท์เตอร์โหมด และอีกหนึ่งโหมดจะเป็นไทม์เมอร์โหมด ในการอนุวัตตัวชดเชยนี้จะทำการปรับตั้งค่าให้เป็นไทม์เมอร์โหมด ซึ่งจะตรงกับ CTC2 (Timer mode) ต้องใช้ฟังก์ชันที่มีรูปแบบดังนี้

```
int setctc(int ctc, int mode, byte timer, int intr)
```

ในการกำหนดค่าเริ่มต้นให้กับฟังก์ชันจะต้องทราบความหมายของพารามิเตอร์แต่ละตัว ซึ่งมีรายละเอียดดังนี้

ctc	CTC โหมด 0-3
mode	0 timer mode sysclock/16
	1 timer mode sysclock/256
	2 counter mode

	4 timer mode sysclock/16 trigger rising
	5 timer mode sysclock/256 trigger rising
	6 timer mode sysclock/16 trigger falling
	7 timer mode sysclock/256 trigger falling
timer	timer reload value 0-255
intr	0 disable interrupt
	1 enable interrupt

ซึ่งในการอนุวัตตัวชดเชยจะทำการปรับตั้งค่า โดย **ctc** ปรับตั้งค่าเป็นโหมด 2 **mode** ปรับตั้งค่าเป็น 1 **timer** ปรับตั้งค่าเพื่อกำหนด sampling rate ที่ต้องการ **intr** ปรับตั้งค่าเป็น enable เพื่อใช้อินเทอร์รัปต์

ตัวอย่าง การปรับตั้งค่าอินเทอร์รัปต์รูทีนให้ทำงานทุก ๆ 5 มิลลิวินาที (timer period = 5 milliseconds)

```
#INT_VEC CTC2_VEC myroutine
        setctc(2,1,180,1);              // every  5  milliseconds
```

บทที่ 3

โปรแกรมการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

3.1 กล่าวนำ

การอนุวัตตัวชดเชยแบบดิจิตอลในโครงงานนี้จะใช้โครงสร้างแบบ direct form 2 (canonic form) สำหรับการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันทั้งอันดับหนึ่งและสองดังที่ได้กล่าวไว้ในบทที่ 2 ส่วนการประยุกต์ใช้งานทรานสเฟอร์ฟังก์ชันอันดับที่สูงขึ้นไป สามารถนำโครงสร้างแบบ direct form 2 ที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสองที่จะกล่าวถึงในบทนี้ไปต่อเรียงกัน (cascade form) หรือนำไปต่อขนานกัน (parallel form) ขึ้นอยู่กับการประยุกต์ใช้งาน ส่วนเรื่องการคำนวณหาค่าสัมประสิทธิ์ของตัวชดเชยจะใช้วิธีการแปลงจาก s-domain เป็น z-domain โดยอาศัยความสัมพันธ์ของทุสติน ในบทที่ 3 นี้จะกล่าวถึง โปรแกรมการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ซึ่งโปรแกรมการอนุวัตจะเขียนด้วยภาษาไดนามิกส์ซี และสามารถแบ่งออกได้เป็น 2 ส่วนด้วยกันคือ

<u>ส่วนแรก</u>

จะเป็นส่วนของโปรแกรมการคำนวณค่าสัมประสิทธิ์ด้วยการแปลงแบบทุสติน โดยใช้โปรแกรม MATLAB เพื่อที่จะนำค่าสัมประสิทธิ์ดังกล่าวมาใช้กับโปรแกรมการอนุวัตในโปรแกรมไดนามิกส์ซี

<u>ส่วนที่สอง</u>

จะเป็นโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสองในโปรแกรมไดนามิกส์ซี

3.2 ภาพรวมของการโปรแกรมการอนุวัตตัวชดเชย

โปรแกรมที่เขียนขึ้นในไดนามิกส์ซีสำหรับตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสองคือ **order1.c** และ **order2.c** ตามลำดับ (แสดงรายละเอียดในภาคผนวก ก และ ข ตามลำดับ) โดยจะกล่าวถึงทรานสเฟอร์ฟังก์ชันอันดับหนึ่งก่อนและตามด้วยอันดับสองดังรายละเอียดต่อไปนี้

3.2.1 โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

ทรานสเฟอร์ฟังก์ชันอันดับหนึ่งของตัวชดเชยเขียนในรูป s-domain ได้ดังสมการที่ 3.1

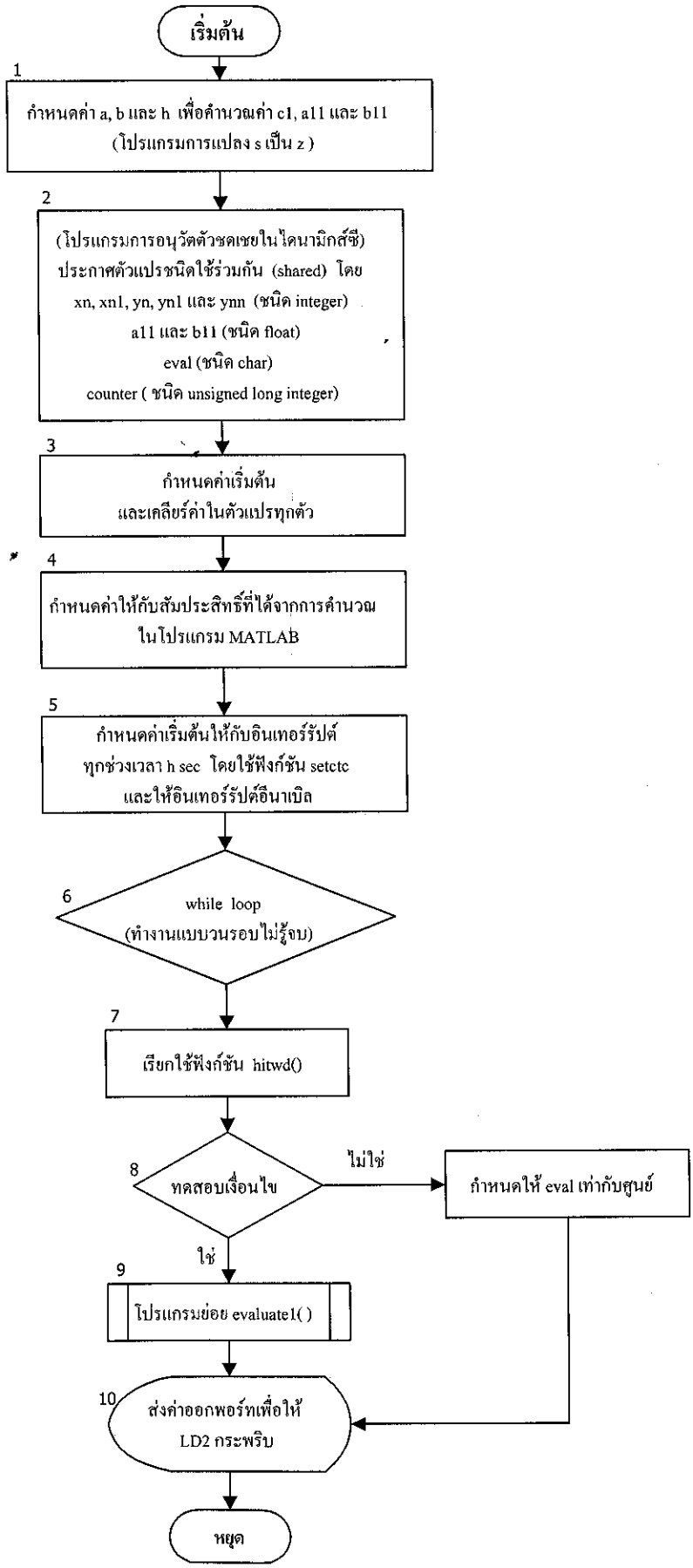
$$H(s) = \frac{s+a}{s+b} \tag{3.1}$$

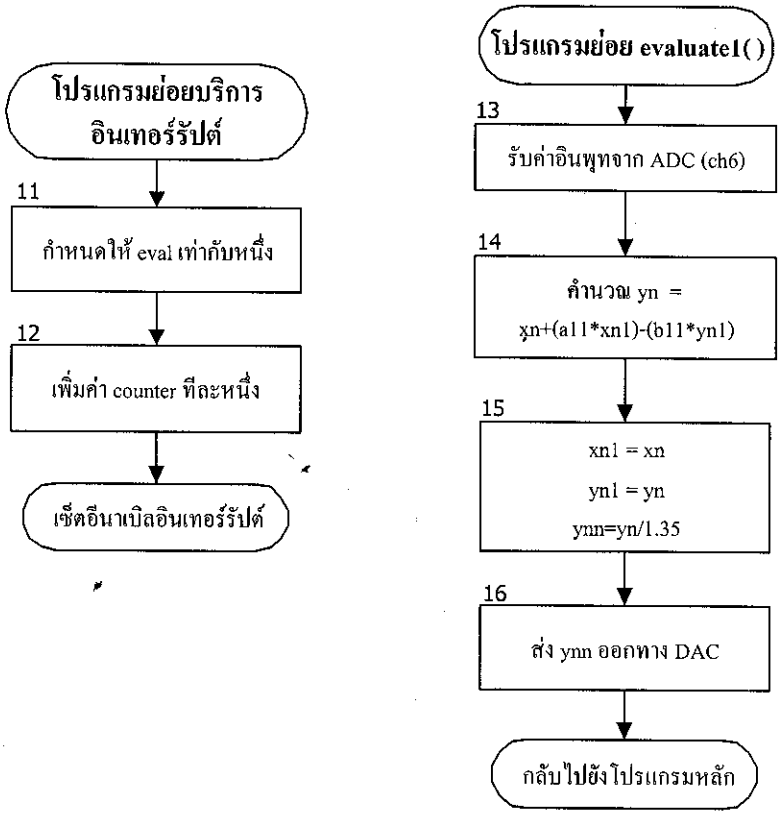
เมื่อทำการแปลงให้อยู่ในรูป z-domain โดยอาศัยความสัมพันธ์แบบทุสติน สามารถเขียนได้ดังสมการที่ 2.19a และเมื่อเขียนในรูปสมการผลต่างจะสามารถเขียนได้ดังสมการที่ 2.19b และเมื่อนำสมการที่ 2.19b มาเขียนใหม่เพื่อใช้ในการอนุวัตตัวชดเชย จะเป็นดังสมการที่ 2.19c

$$y(n) = x(n) + a_{11}x(n-1) - b_{11}y(n-1)$$

ซึ่งสมการ 2.19c นี้จะเป็นสมการหลักในการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

โปรแกรมการอนุวัตตามที่ได้กล่าวมาข้างต้นนี้สามารถเขียนแสดงกระบวนการทำงานดังรูปที่ 3.1


เริ่มต้น
1
กำหนดค่า a, b และ h เพื่อคำนวณค่า c1, a11 และ b11
(โปรแกรมการแปลง s เป็น z)
2
(โปรแกรมการอนุวัตตัวชดเชยในไดนามิกส์ซี)
ประกาศตัวแปรชนิดใช้ร่วมกัน (shared) โดย
xn, xn1, yn, yn1 และ ynn (ชนิด integer)
a11 และ b11 (ชนิด float)
eval (ชนิด char)
counter (ชนิด unsigned long integer)
3
กำหนดค่าเริ่มต้น
และเคลียร์ค่าในตัวแปรทุกตัว
4
กำหนดค่าให้กับสัมประสิทธิ์ที่ได้จากการคำนวณ
ในโปรแกรม MATLAB
5
กำหนดค่าเริ่มต้นให้กับอินเทอร์รัปต์
ทุกช่วงเวลา h sec โดยใช้ฟังก์ชัน setctc
และให้อินเทอร์รัปต์อีนาเบิล
6
while loop
(ทำงานแบบวนรอบไม่รู้จบ)
7
เรียกใช้ฟังก์ชัน hitwd()
8
ทดสอบเงื่อนไข
ไม่ใช่
กำหนดให้ eval เท่ากับศูนย์
ใช่
9
โปรแกรมย่อย evaluate1( )
10
ส่งค่าออกพอร์ทเพื่อให้
LD2 กระพริบ
หยุด



รูปที่ 3.1 แผนภูมิการทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

อธิบายการทำงาน

บล็อกที่หนึ่ง จะเป็นการทำงานของโปรแกรมการแปลงสัมประสิทธิ์ของทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain โดยอาศัยความสัมพันธ์แบบทุสติน เป็นโปรแกรมที่ดำเนินการในโปรแกรม MATLAB โดยจะต้องทำการป้อนค่าสัมประสิทธิ์ใน s-domain ซึ่งก็คือ ค่า a และ b รวมทั้ง h (Sampling interval) เพื่อให้โปรแกรมคำนวณค่า c1, a11 และ b11

บล็อกที่สอง จะเป็นส่วนเริ่มต้นของโปรแกรมการอนุวัตตัวชดเชยในไดนามิกส์ซี ในขั้นตอนแรกจะต้องทำการประกาศตัวแปรทั้งหมดที่ต้องใช้ โดยจะประกาศตัวแปรเป็นชนิดใช้ร่วมกัน (shared) และจะต้องประกาศชนิดของตัวแปรที่ต้องการนำไปใช้ด้วยซึ่งก็คือ

- ตัวแปร xn, xn1, yn, yn1 และ ynn ประกาศเป็นชนิด integer
- ตัวแปร a11 และ b11 ประกาศเป็นชนิด float
- ตัวแปร eval ประกาศเป็นชนิด char
- ตัวแปร counter ประกาศเป็นชนิด unsigned long integer

บล็อกที่สาม จะเป็นการกำหนดค่าเริ่มต้นให้กับพอร์ทที่ใช้ในการอินเทอร์รัปต์ และกำหนดค่าเริ่มต้นให้กับตัวแปรทุกตัวมีค่าเป็นศูนย์

บล็อกที่สี่ จะเป็นการกำหนดค่าที่ได้จากการคำนวณในโปรแกรมการแปลงแบบทุสตินให้กับ a11 และ b11

บล็อกที่ห้า เป็นการกำหนดค่าเริ่มต้นให้กับอินเทอร์รัปต์ให้ทำงานในช่วงเวลา h วินาทีที่กำหนด โดยใช้ฟังก์ชัน setctc

บล็อกที่หก คือส่วนของ while loop ที่ทำงานวนรอบแบบไม่รู้จบเพื่อให้โปรแกรมทำงานได้ตลอด

บล็อกที่เจ็ด เป็นการเรียกใช้ฟังก์ชัน hitwd () เพื่อป้องกันไม่ให้มีการปรับตั้งค่าใหม่ กรณีที่โปรแกรมมีการทำงานแบบวนรอบไม่รู้จบ ถ้าไม่ใช้ฟังก์ชันนี้อาจเกิดความคลาดเคลื่อนในการรันโปรแกรมได้

บล็อกที่แปด เป็นฟังก์ชัน if เพื่อทำการตรวจสอบเงื่อนไข ถ้าเป็นจริงจะไปทำโปรแกรมย่อยที่ชื่อ evaluate1 () แต่ถ้าเป็นเท็จจะไปกำหนดให้ eval เป็นศูนย์

บล็อกที่เก้า เรียกโปรแกรมย่อยชื่อ evaluate1( ) ซึ่งเป็นโปรแกรมการอนุวัตตัวชดเชยสำหรับทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

บล็อกที่สิบ ส่งค่าออกพอร์ทเพื่อให้ LD2 กระพริบทุกครั้งที่มีการอินเทอร์รัปต์

บล็อกที่สิบเอ็ด เป็นส่วนแรกของโปรแกรมย่อยบริการอินเทอร์รัปต์ โดยจะทำการปรับตั้งค่าให้ eval มีค่าเท่ากับหนึ่งเมื่อมีการอินเทอร์รัปต์เกิดขึ้น

บล็อกที่สิบสอง เป็นส่วนที่สองของโปรแกรมย่อยบริการอินเทอร์รัปต์ จะทำการเพิ่มค่า counter ขึ้นทีละหนึ่ง โดย counter นี้จะเป็นตัวแปรที่ใช้ในการนับอินเทอร์รัปต์

บล็อกที่สิบสาม เป็นส่วนแรกของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง โดยจะทำการรับค่าอินพุทจากตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (ADC) ผ่านทางช่องรับสัญญาณที่ 6 (ch6)

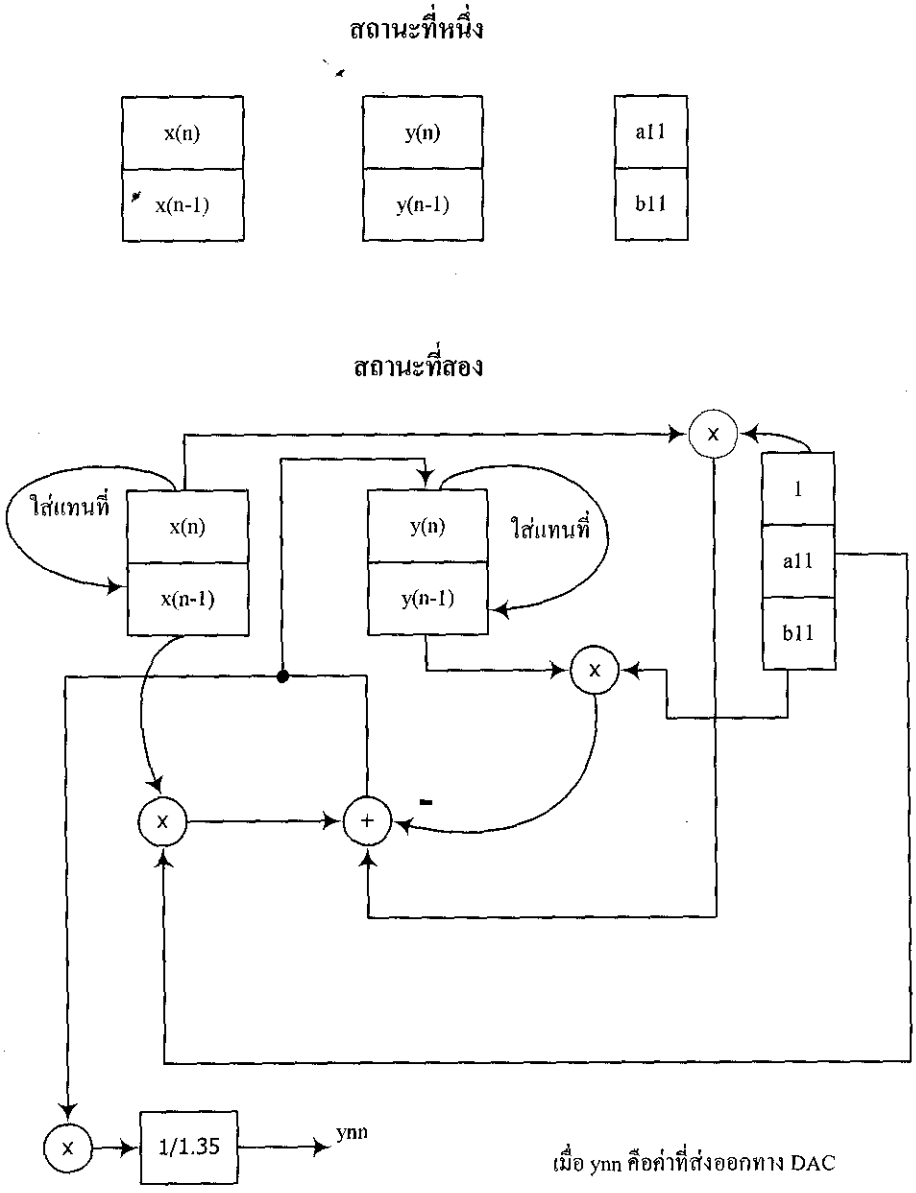
บล็อกที่สิบสี่ เป็นส่วนที่สองของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง โดยจะทำการคำนวณค่า yn จากสมการการอนุวัตดังนี้

$$yn = xn+(a11*xn1)-(b11*yn1)$$

บล็อกที่สิบห้า เป็นส่วนที่สามของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง โดยจะทำการเก็บค่าของอินพุทและเอาท์พุทไว้เพื่อหน่วงเวลาหนึ่งหน่วยเวลา ซึ่งก็คือการนำค่าอินพุท x(n) มาเก็บใน x(n-1) สำหรับด้านเอาท์พุทจะนำค่า y(n) เก็บใน y(n-1) ส่วนตอนที่

จะส่งค่า yn ออก จะทำการหารออกด้วย 1.35 เพื่อลดทอนให้เท่ากับค่าจริง เนื่องจาก 1.35 คืออัตราขยายเนื่องจากคุณสมบัติของคอนโทรลเลอร์บอร์ด แล้วส่งออกโดยใช้ตัวแปรชื่อ ynn เพื่อให้ค่าที่ส่งเท่ากับค่าจริงเท่านั้น ไม่ได้ทำการเปลี่ยนแปลงค่าในสมการการอนุวัตตัวชดเชยแต่อย่างใด การทำงานของบล็อกที่สิบสี่และบล็อกนี้แสดงได้ด้วยรูปที่ 3.2

<u>บล็อกที่สิบหก</u> เป็นส่วนสุดท้ายของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง โดยจะทำการส่ง ynn ออกทางตัวแปลงสัญญาณดิจิตอลเป็นอนาล็อก (DAC)



รูปที่ 3.2 กระบวนการทำงานของบล็อกที่สิบสี่และสิบห้า

3.2.2 โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

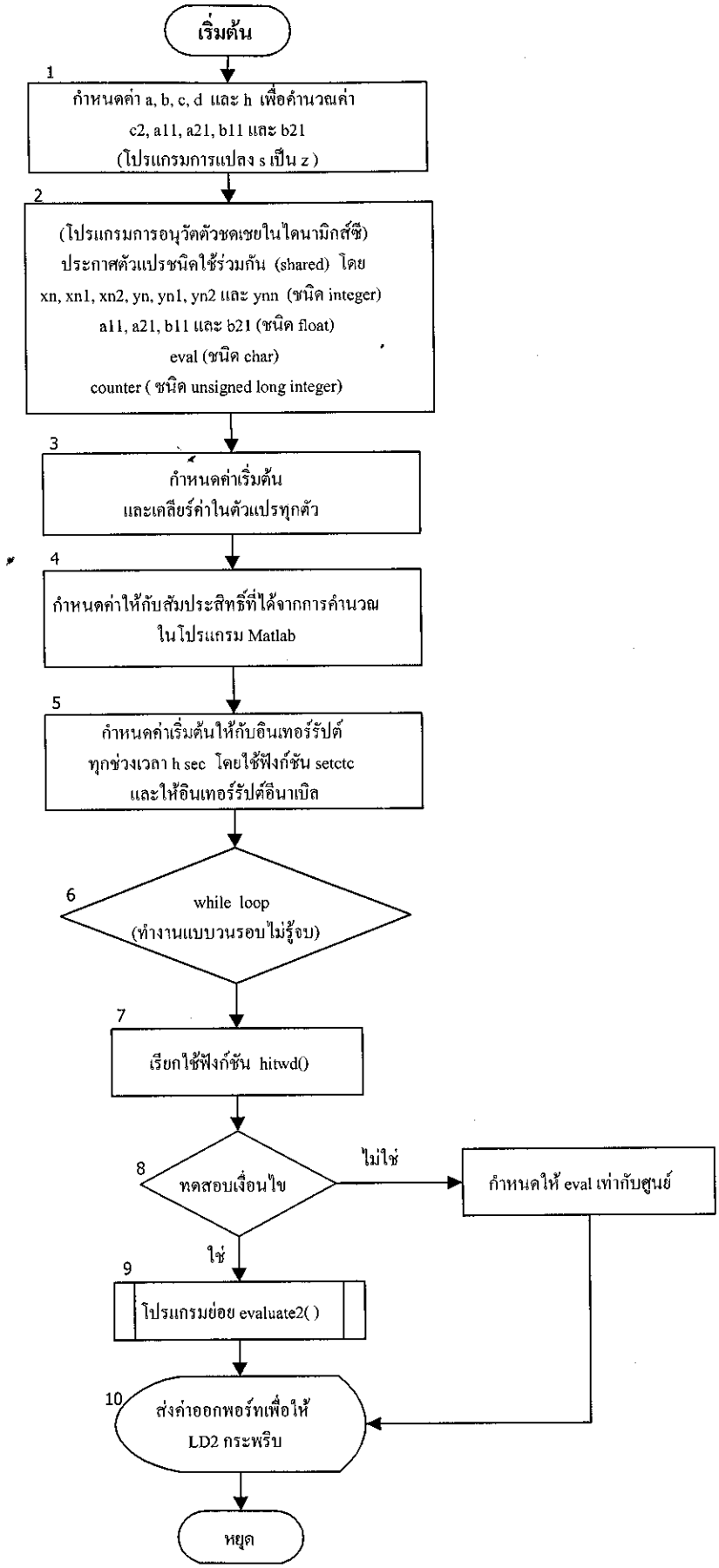
สำหรับทรานสเฟอร์ฟังก์ชันอันดับสองที่ใช้ในการอนุวัตตัวชดเชยสามารถเขียนในรูป s-domain ดังสมการที่ 3.2

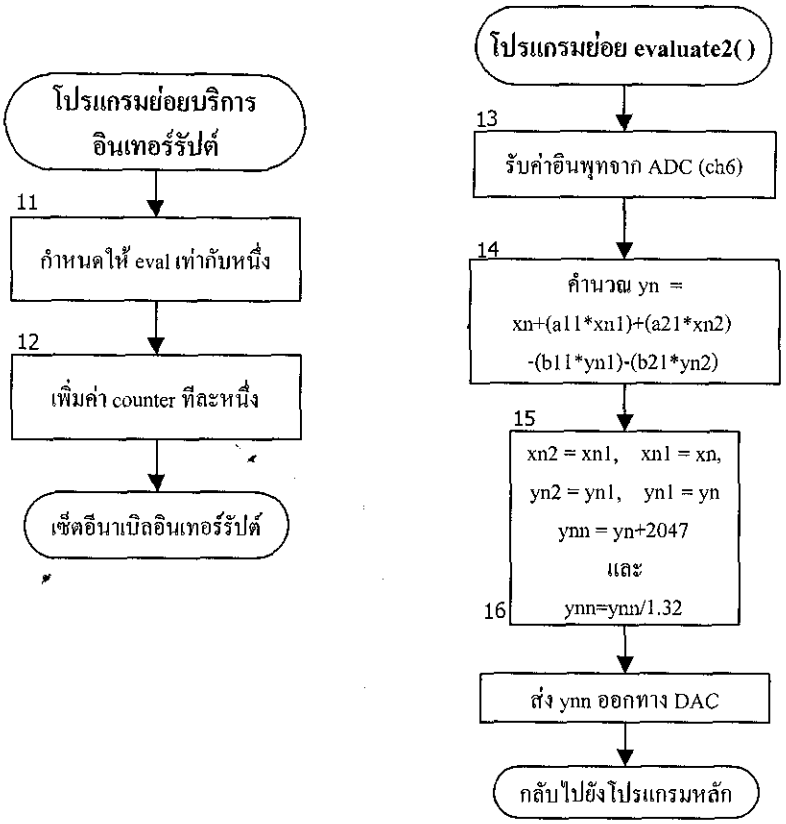
$$H(s) = \frac{s^2 + cs + d}{s^2 + as + b} \tag{3.2}$$

จากสมการใน s-domain สามารถเขียนในรูป z-domain ได้ดังสมการที่ 2.20a และสามารถเขียนในรูปสมการผลต่างได้ดังสมการที่ 2.20b ส่วนสมการที่ใช้ในการอนุวัตตัวชดเชยคือ สมการที่ 2.20c

$$y(n) = x(n) + a_{11}x(n-1) + a_{21}x(n-2) - b_{11}y(n-1) - b_{21}y(n-2)$$

ซึ่งสมการ 2.20c นี้จะเป็นสมการหลักในการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง แสดงกระบวนการทำงานในรูปที่ 3.3


เริ่มต้น
1
กำหนดค่า a, b, c, d และ h เพื่อคำนวณค่า
c2, a11, a21, b11 และ b21
(โปรแกรมการแปลง s เป็น z)
2
(โปรแกรมการอนุวัตตัวชดเชยในไดนามิกส์ซี)
ประกาศตัวแปรชนิดใช้ร่วมกัน (shared) โดย
xn, xn1, xn2, yn, yn1, yn2 และ ynn (ชนิด integer)
a11, a21, b11 และ b21 (ชนิด float)
eval (ชนิด char)
counter (ชนิด unsigned long integer)
3
กำหนดค่าเริ่มต้น
และเคลียร์ค่าในตัวแปรทุกตัว
4
กำหนดค่าให้กับสัมประสิทธิ์ที่ได้จากการคำนวณ
ในโปรแกรม Matlab
5
กำหนดค่าเริ่มต้นให้กับอินเทอร์รัปต์
ทุกช่วงเวลา h sec โดยใช้ฟังก์ชัน setctc
และให้อินเทอร์รัปต์อีนาเบิล
6
while loop
(ทำงานแบบวนรอบไม่รู้จบ)
7
เรียกใช้ฟังก์ชัน hitwd()
8
ทดสอบเงื่อนไข
ไม่ใช่
กำหนดให้ eval เท่ากับศูนย์
ใช่
9
โปรแกรมย่อย evaluate2( )
10
ส่งค่าออกพอร์ทเพื่อให้
LD2 กระพริบ
หยุด



รูปที่ 3.3 แผนภูมิการทำงานของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

<u>อธิบายการทำงาน</u>

การทำงานในขั้นตอนต่าง ๆ โดยทั่วไปจะเหมือนกับโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง แต่มีบางส่วนที่แตกต่างกัน ซึ่งในที่นี้จะอธิบายเฉพาะในส่วนที่แตกต่างกันเท่านั้น ได้แก่

<u>บล็อกที่หนึ่ง</u> เนื่องจากเป็นทรานสเฟอร์ฟังก์ชันอันดับสอง ทำให้มีพจน์ของสัมประสิทธิ์เพิ่มขึ้น ซึ่งค่าสัมประสิทธิ์ที่คำนวณได้ในโปรแกรม MATLAB ก็คือ c2, a11, a21, b11 และ b21 การคำนวณค่าในโปรแกรม MATLAB ก็เช่นกัน ต้องป้อนค่าสัมประสิทธิ์ใน s-domain เป็น a, b, c และ d และป้อนค่า h (Sampling interval) ที่เหมาะสม

<u>บล็อกที่สอง</u> ในส่วนของการประกาศตัวแปร เป็นส่วนเริ่มต้นของโปรแกรมการอนุวัตตัวชดเชยในไดนามิกส์ซี จะประกาศตัวแปรดังนี้

- ตัวแปร xn, xn1, xn2, yn, yn1, yn2 และ ynn ประกาศเป็นชนิด integer

- ตัวแปร a11, a21, b11 และ b21 ประกาศเป็นชนิด float
- ตัวแปร eval ประกาศเป็นชนิด char
- ตัวแปร counter ประกาศเป็นชนิด unsigned long integer

โดยที่ตัวแปรทั้งหมดยังคงเป็นตัวแปรชนิดใช้ร่วมกัน (shared)

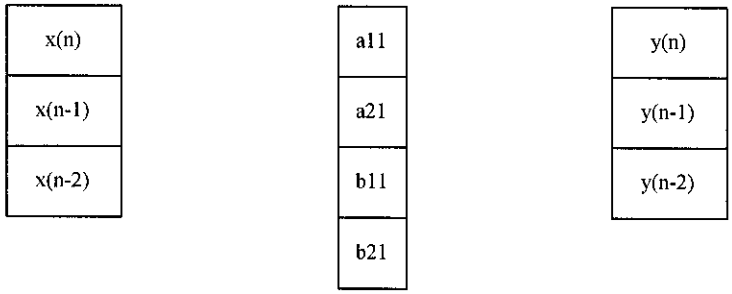
บล็อกที่สิบสี่ ซึ่งเป็นส่วนที่สองของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง โดยการคำนวณค่า yn จากสมการการอนุวัตจะเปลี่ยนเป็น

yn = xn+(a11*xn1)+(a21*xn2)-(b11*yn1)-(b21*yn2)

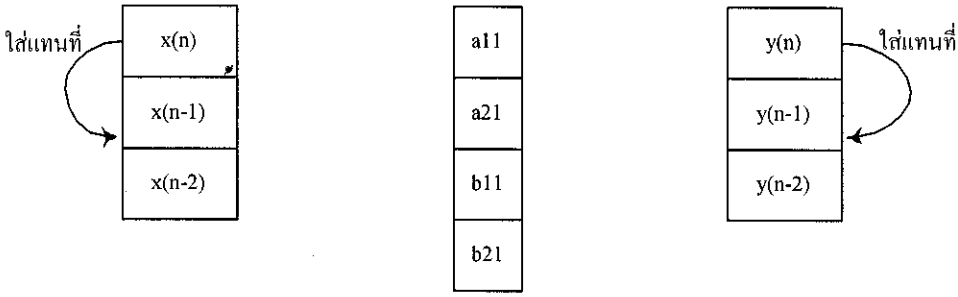
บล็อกที่สิบห้า เป็นส่วนที่สามของโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง โดยจะทำการเก็บค่าของอินพุทและเอาท์พุทไว้เพื่อหน่วงเวลาหนึ่งหน่วยเวลา ซึ่งก็คือการนำค่าอินพุท x(n-1) มาเก็บใน x(n-2) และในทำนองเดียวกันก็จะนำค่า x(n) มาเก็บใน x(n-1) ใน state ถัดมา สำหรับด้านเอาท์พุทจะนำค่า y(n-1) มาเก็บใน y(n-2) และนำค่า y(n) เก็บใน y(n-1) ใน state ถัดมา ส่วนตอนที่จะส่งค่า yn ออก จะทำการบวกด้วย 2047 เพื่อทำการปรับค่าของเอาท์พุทเนื่องมาจากข้อจำกัดของตัวแปลงสัญญาณดิจิตอลเป็นอนาล็อก (DAC) แล้วส่งออกโดยใช้ตัวแปรชื่อ ynn เพื่อให้ค่าที่ส่งออกเป็นเพียงการปรับขนาดของสัญญาณตามข้อจำกัดของ DAC เท่านั้น ไม่ได้ไปทำการเปลี่ยนแปลงในสมการการอนุวัตตัวชดเชยแต่อย่างใด การทำงานของบล็อกที่สิบสี่กับบล็อกนี้แสดงได้ด้วยรูปที่ 3.4

ส่วนบล็อกที่ไม่ได้กล่าวถึงแสดงว่ามีการทำงานที่คล้ายคลึงกับอันดับหนึ่ง

สถานะที่หนึ่ง

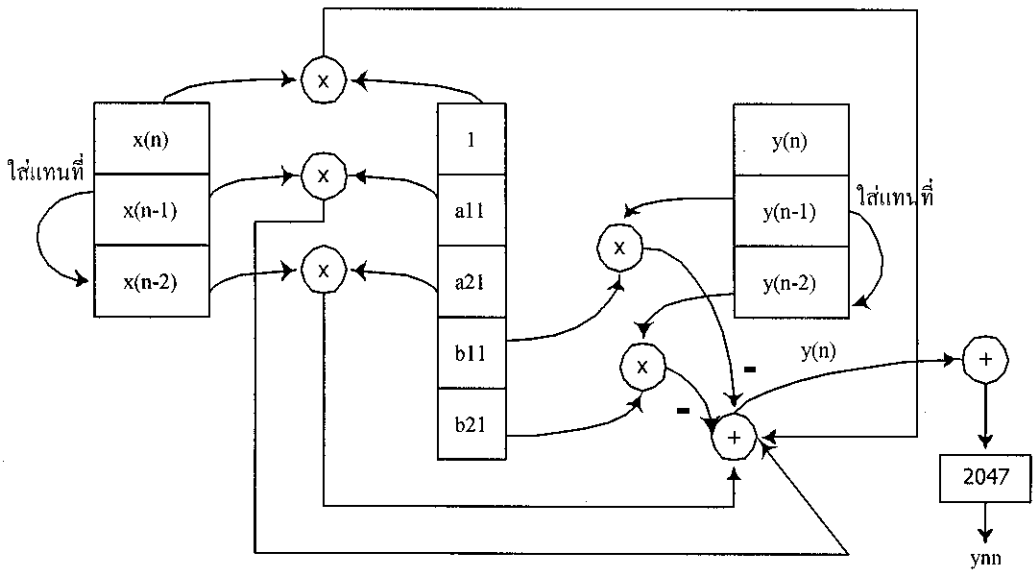


สถานะที่สอง



เมื่อ x(n) คือ อินพุท
y(n) คือ เอาท์พุท
และ
ynn คือค่าที่ส่งออกทาง DAC

สถานะที่สาม



รูปที่ 3.4 กระบวนการทำงานของบล็อกที่สิบสี่และสิบห้า

3.2.3 การคำนวณเพื่อกำหนดค่าความถี่ที่ใช้ในการสุ่มสัญญาณ (Sampling frequency, f_s)

การกำหนดค่าความถี่ที่ใช้ในการสุ่มสัญญาณนั้นควรทราบเวลาการทำงานในแต่ละส่วนของโปรแกรมเสียก่อน

- **การคำนวณค่าความถี่ที่ใช้ในการสุ่มสัญญาณสำหรับโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง**

ตารางที่ 3.1 เวลาที่ใช้ในการทำงานของแต่ละส่วนในโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

ส่วนต่าง ๆ ของโปรแกรมการอนุวัตตัวชดเชย	เวลาที่ใช้ในการประมวลผล (μ sec)
1. การกำหนดค่าเริ่มต้นและเคลียร์ค่าตัวแปร	4
2. การกำหนดค่าให้กับสัมประสิทธิ์ที่ได้จากการคำนวณในโปรแกรม MATLAB	3
3. การกำหนดค่าเริ่มต้นให้กับโปรแกรมบริการอินเทอร์รัปต์	4
4. การทำงานของ while loop (เมื่อคิดเพียง 1 รอบ)	67.13
5. โปรแกรมย่อยบริการอินเทอร์รัปต์	2
6. โปรแกรมย่อย evaluate1( )	53.5

เมื่อทราบเวลาที่ใช้ในการทำงานของแต่ละส่วน จะทำให้สามารถเลือกความถี่ในการสุ่มที่เหมาะสมได้ โดยพิจารณาจากค่าเวลาที่ใช้ในการประมวลผล ซึ่งค่าที่เลือกจะต้องมากกว่าเวลาที่ใช้ในการประมวลผลทั้งหมด เมื่อเลือกค่าที่เหมาะสมได้แล้วจึงนำค่าเวลานั้นมาคำนวณกลับหาค่าความถี่ที่ใช้ในการสุ่ม (f_s) โดยอาศัยความสัมพันธ์ในสมการ 3.3

$$f_s = \frac{1}{h} \quad \text{sec} \tag{3.3}$$

เมื่อ h คือคาบของการสุ่มสัญญาณ (Sampling interval) ในที่นี้ค่า h ที่เลือกใช้ในการอนุวัตตัวชดเชยสำหรับทรานสเฟอร์ฟังก์ชันอันดับหนึ่งคือ 1 มิลลิวินาที (จะเห็นว่าคาบการสุ่มที่เลือกนี้จะทำให้โปรแกรมสามารถทำงานได้ทัน) ทำให้ได้ค่า f_s เท่ากับ 1,000 Hz

เมื่อทำการกำหนดความถี่ที่ใช้ในการสุ่มได้แล้ว จะต้องนำค่านี้ไปกำหนดให้กับ CTC Counter/Timer ในโปรแกรมไดนามิกส์ซี โดยพารามิเตอร์ที่ใช้ในโปรแกรมไดนามิกส์ซีคือ Timer reload (มีค่าตั้งแต่ 0-255) ดังนั้นจึงต้องทำการคำนวณจากค่าความถี่ในการสุ่มเพื่อคำนวณค่า Timer reload ที่ต้องใช้ในโปรแกรมการอนุวัตโดยอาศัยความสัมพันธ์ในสมการ 3.4

$$Timer\ reload = \frac{system\ clock\ (12.288\ MHz)}{prescaler \times f_s} \tag{3.4}$$

การอนุวัตในครั้งนี้จะเลือกใช้ Prescaler ที่ 256 (คือ Timer mode ที่นำ System clock มาแบ่งออกเป็น 256 ส่วน) ทั้งการอนุวัตทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง ทำให้สามารถหาค่า Timer reload ได้จากสมการต่อไปนี้

$$Timer\ reload = \frac{12.288\ MHz}{256 \times 1000} = 48 \tag{3.5}$$

หลังจากนั้นจึงนำค่านี้ไปปรับตั้งค่าในฟังก์ชัน **setctc** ซึ่งอยู่ในส่วนที่ใช้กำหนดค่าเริ่มต้นให้กับโปรแกรมย่อยบริการอินเทอร์รัปต์ดังนี้

```
setctc (2, 1, 48, 1);        // ปรับตั้งค่าอินเทอร์รัปต์ทุก ๆ 1 มิลลิวินาที
```

- **การหาค่าความถี่ที่ใช้ในการสุ่มสัญญาณสำหรับโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง**

สำหรับทรานสเฟอร์ฟังก์ชันอันดับสองการคำนวณความถี่ที่ใช้ในการสุ่มเหมือนกับทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง แต่ค่าความถี่ที่ได้จะแตกต่างกันเนื่องจากเวลาที่ใช้ในการประมวลผลต่างกัน ดังรายละเอียดต่อไปนี้

ตารางที่ 3.2 เวลาที่ใช้ในการทำงานของแต่ละส่วนในโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

ส่วนต่าง ๆ ของ โปรแกรมการอนุวัตตัวชดเชย	เวลาที่ใช้ในการประมวลผล (μ sec)
1. การกำหนดค่าเริ่มต้นและเคลียร์ค่าตัวแปร	6
2. การกำหนดค่าให้กับสัมประสิทธิ์ที่ได้จากการคำนวณในโปรแกรม MATLAB	4
3. การกำหนดค่าเริ่มต้นให้กับโปรแกรมบริการอินเทอร์รัปต์	4
4. การทำงานของ while loop (เมื่อคิดเพียง 1 รอบ)	57.72
5. โปรแกรมย่อยบริการอินเทอร์รัปต์	2
6. โปรแกรมย่อย evaluate2( )	49.8

สำหรับทรานสเฟอร์ฟังก์ชันอันดับสองนี้ ค่า h ที่เลือกใช้คือ 2 มิลลิวินาที จากความสัมพันธ์ในสมการที่ 3.3 ทำให้ได้ค่า f_s คือ 500 Hz และจากความสัมพันธ์ในสมการที่ 3.4 ค่า Timer reload คือ

$$Timer\ reload = \frac{12.288\,MHz}{256 \times 500} = 96 \qquad (3.6)$$

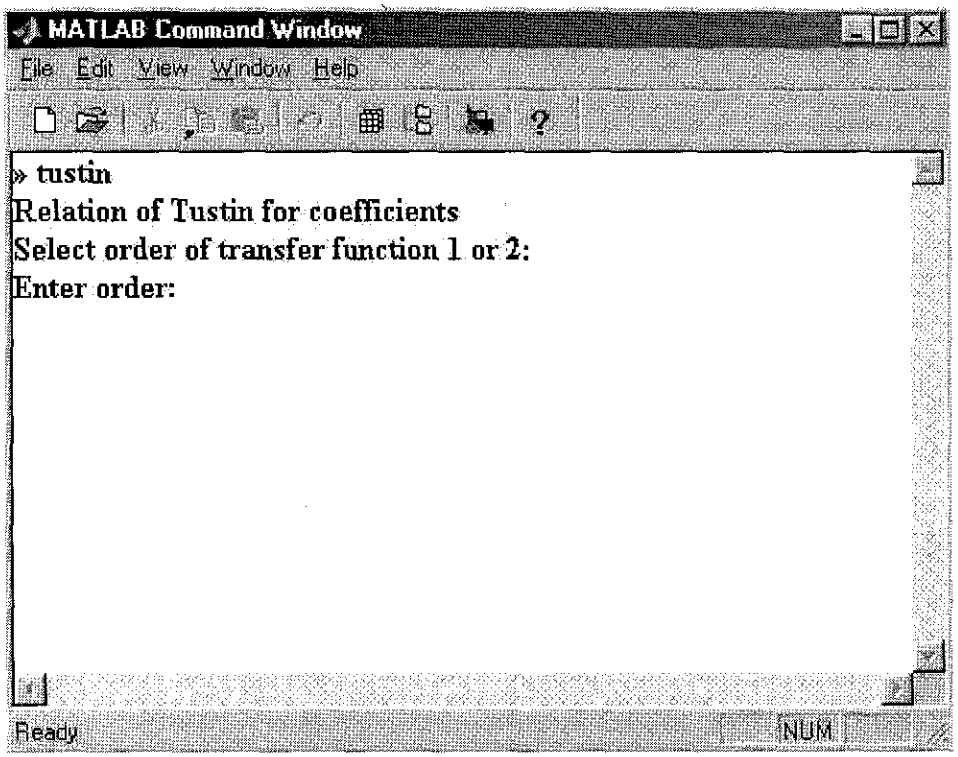
ดังนั้นการกำหนดค่า timer reload ให้กับฟังก์ชัน setctc จึงเป็น

```
setctc (2, 1, 96, 1);        // ปรับตั้งค่าอินเทอร์รัปต์ทุก ๆ 2 มิลลิวินาที
```

3.3 โปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain แบบทุสติน

การแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain โดยอาศัยความสัมพันธ์แบบทุสตินได้กล่าวโดยละเอียดในหัวข้อทฤษฎีพื้นฐานการแปลงแบบทุสตินในบทที่ 2 แล้ว ในบทนี้จึงนำสูตรดังกล่าวมาประยุกต์ใช้ในการโปรแกรมการแปลงโดยโปรแกรมนี้เขียนขึ้นใน MATLAB ซึ่งรายละเอียดของโปรแกรมแสดงไว้ในภาคผนวก ค วิธีการใช้งานโปรแกรมการแปลง มีรายละเอียดดังนี้

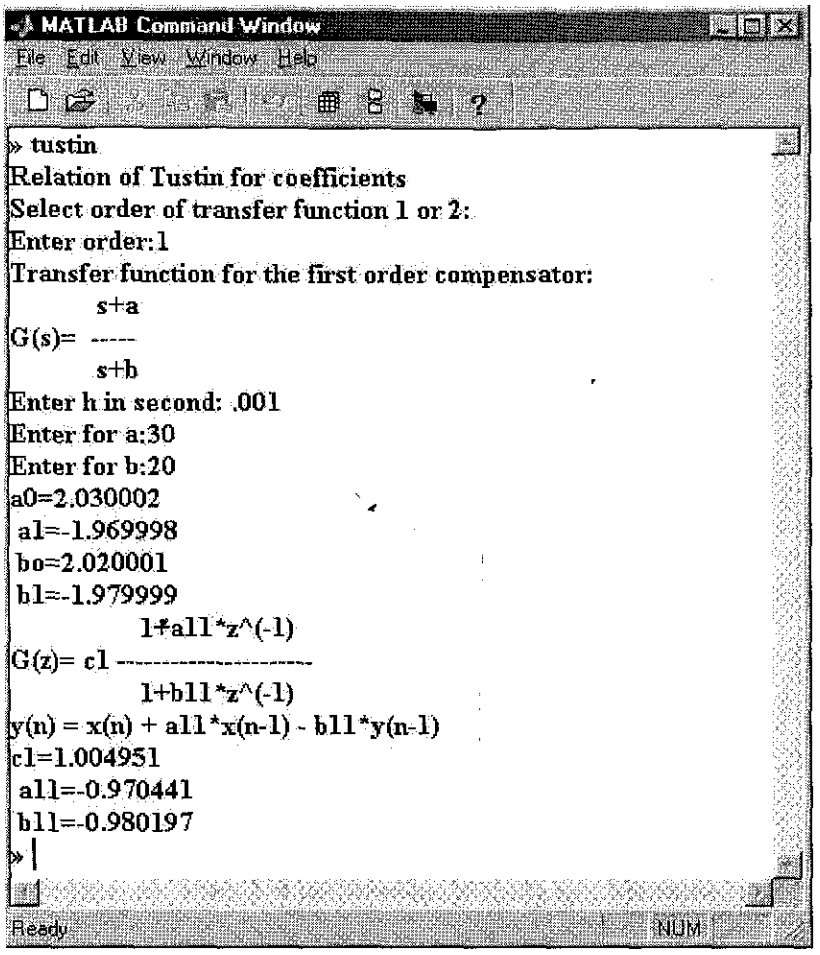
โปรแกรมนี้ออกแบบเพื่อให้ผู้ใช้เลือกทรานสเฟอร์ฟังก์ชันที่ต้องการแล้วป้อนข้อมูลตามลำดับขั้นตอนที่ปรากฏ โดยในขั้นแรกจะทำการเรียกโปรแกรมการแปลงที่มีชื่อว่า **tustin** ขึ้นมาหลังจากนั้นจะมีข้อความปรากฏดังรูปที่ 3.5



รูปที่ 3.5 การใช้งานโปรแกรมการแปลงแบบทุสตินใน MATLAB

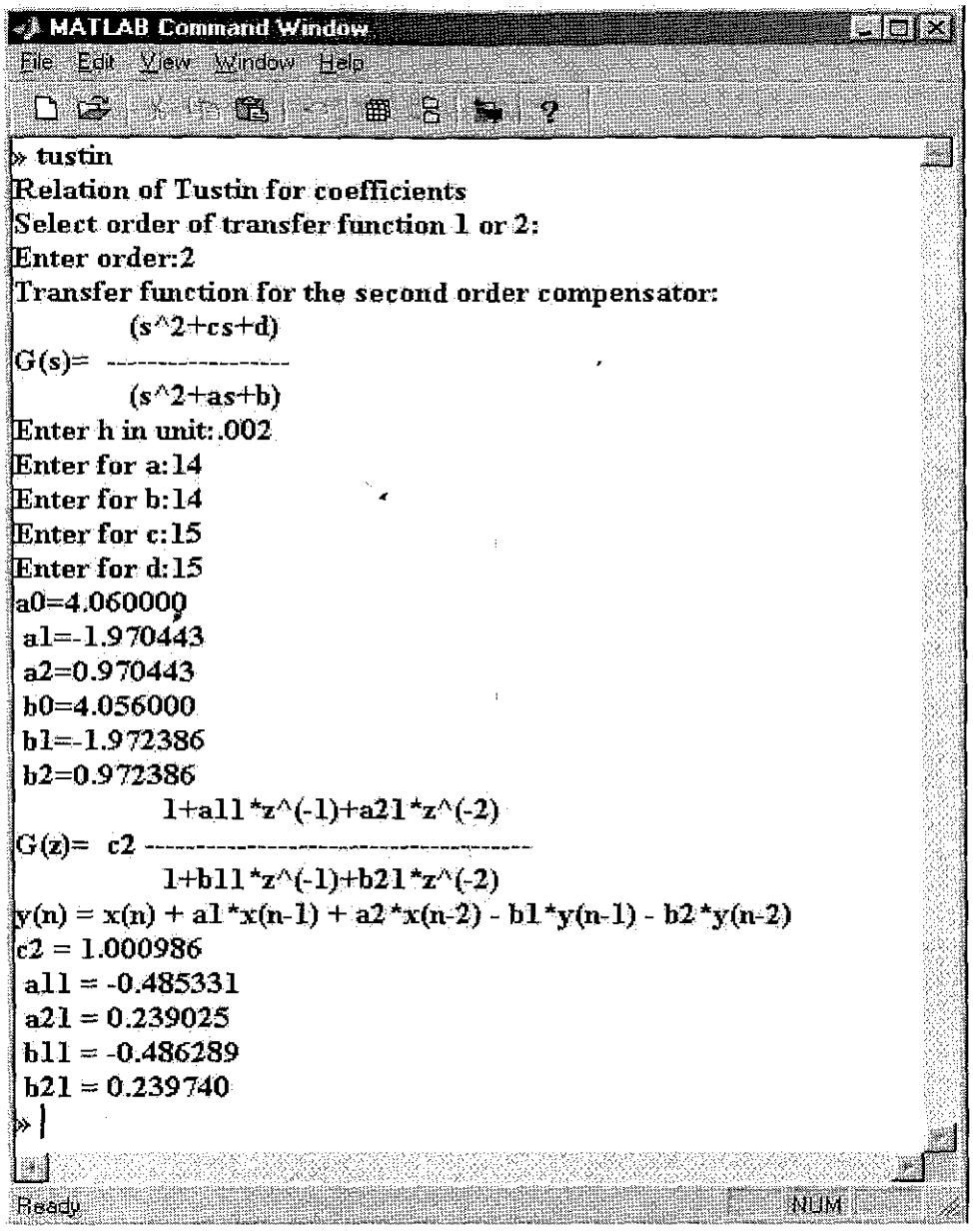
หลังจากที่ทำการเลือก order ของทรานสเฟอร์ฟังก์ชันแล้ว โปรแกรมจะให้ผู้ใช้ป้อนค่าคาบเวลาการสุ่มสัญญาณ (h), a และ b หลังจากนั้นโปรแกรมจะทำการคำนวณค่าสัมประสิทธิ์ให้

ในการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งนี้ ทรานสเฟอร์ฟังก์ชันที่เลือกใช้ มีค่า a = 30 และ b = 20 ส่วนคาบในการสุ่มสัญญาณคือ 1 msec หาค่าสัมประสิทธิ์แสดงในรูปที่ 3.6



รูปที่ 3.6 ผลการทำงานของโปรแกรมการแปลงแบบทุสตินสำหรับทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

สำหรับการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง ทรานสเฟอร์ฟังก์ชันที่เลือกใช้มีค่า a = 14, b = 14, c = 15 และ d = 15 ส่วนคาบในการสุ่มสัญญาณคือ 2 msec การหาค่าสัมประสิทธิ์แสดงในรูปที่ 3.7



รูปที่ 3.7 ผลการทำงานของโปรแกรมการแปลงแบบทุสตินสำหรับทรานสเฟอร์ฟังก์ชันอันดับสอง

บทที่ 4

การทดสอบโปรแกรมการอนุวัตตัวชดเชยด้วย คอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

4.1 กล่าวนำ

ในบทที่ผ่านมาได้กล่าวถึงโปรแกรมที่ใช้ในการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง และโปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันใน s-domain ไปเป็น z-domain โดยใช้ความสัมพันธ์แบบทุสตินโดยอธิบายถึงขั้นตอนการทำงานของโปรแกรมต่าง ๆ อย่างละเอียด ดังนั้นในบทที่ 4 นี้จะกล่าวถึง วิธีการทดสอบโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง ด้วยวิธีการทดสอบทางความถี่ (Frequency response test) พร้อมทั้งแสดงผลการทดสอบ นอกจากนั้นยังมีการทดสอบโปรแกรมการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ดโดยใช้โปรแกรมไดนามิกส์ซี

4.2 วิธีทดสอบโปรแกรมการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

การทดสอบโดยหาผลตอบสนองทางความถี่ (Frequency response) นั้นจะทำการวัดสัญญาณเอาท์พุททั้งอัตราขยายและเฟส โดยมีรายละเอียดดังต่อไปนี้

4.2.1 อุปกรณ์ที่ใช้ในการทดสอบ

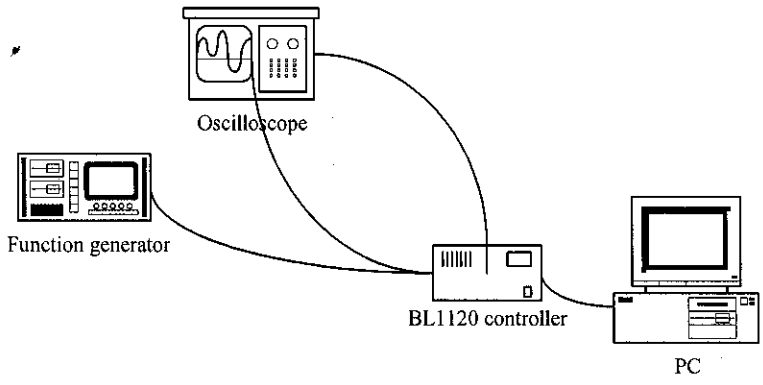
อุปกรณ์ที่ใช้ในการทดสอบมีดังนี้

- Oscilloscope
- Function generator
- คอนโทรลเลอร์บอร์ด
- พีซี
- สาย probe 2 เส้น
- AC line 2 เส้น

4.2.2 การวัดอัตราขยายและเฟสของสัญญาณเอาท์พุท

การวัดอัตราขยายและเฟสมีขั้นตอนดังนี้

1. ก่อนทำการวัดจะต้องต่ออุปกรณ์ดังรูปที่ 4.1 โดยจะต่อออสซิลโลสโคปเข้ากับบอร์ด ช่องรับสัญญาณที่ 1 ของออสซิลโลสโคปต่อเข้ากับช่องรับสัญญาณอินพุทช่องที่ 6 ของตัวบอร์ด และต่อช่องรับสัญญาณที่ 2 ของออสซิลโลสโคปเข้ากับเอาท์พุทของ DAC บนบอร์ดและต่อ Function generator เข้ากับช่องรับสัญญาณอินพุทช่องรับสัญญาณที่ 6 ของตัวบอร์ด เพื่อป้อนอินพุทให้กับช่องรับสัญญาณอินพุทของบอร์ด



รูปที่ 4.1 การต่ออุปกรณ์เพื่อใช้ในการทดสอบ

2. เข้าไปในโปรแกรมไดนามิกส์ซีและเปิดโปรแกรมการอนุวัตที่ต้องการทดสอบ (ไฟล์ในไดนามิกส์ซีจะใช้นามสกุล .c) หลังจากนั้นทำการคอมไพล์ (compile) โปรแกรมโดยการกดปุ่ม COMPILE หรือกดปุ่ม F3 โปรแกรมไดนามิกส์ซีจะทำการคอมไพล์โปรแกรมการอนุวัตให้ หลังจากคอมไพล์เสร็จเรียบร้อยแล้ว ก็จะต้องทำการรันโปรแกรมโดยการกดปุ่ม RUN หรือกดปุ่ม F9 (ถ้าต้องการคอมไพล์และรันทันทีหลังจากคอมไพล์เสร็จ สามารถกดปุ่ม F9 เพียงปุ่มเดียวก็ได้ โปรแกรมจะทำการคอมไพล์และรันให้ทันที)
3. เมื่อทำการรันโปรแกรมเสร็จสิ้นแล้ว ผลการรันโปรแกรมจะปรากฏขึ้นที่หน้าจอของออสซิลโลสโคปโดยแสดงภาพสัญญาณทั้งส่วนของอินพุทและเอาท์พุท เมื่อต้องการอ่านค่าแรงดันอินพุทและเอาท์พุทให้กดปุ่ม MEASURE บนออสซิลโลสโคป จะเห็นภาพสัญญาณพร้อมทั้งค่าแรงดันอินพุทและเอาท์พุทดังรูปที่ 4.2

4. สำหรับการคำนวณค่าอัตราขยายนั้นจะใช้ความสัมพันธ์ดังนี้คือ

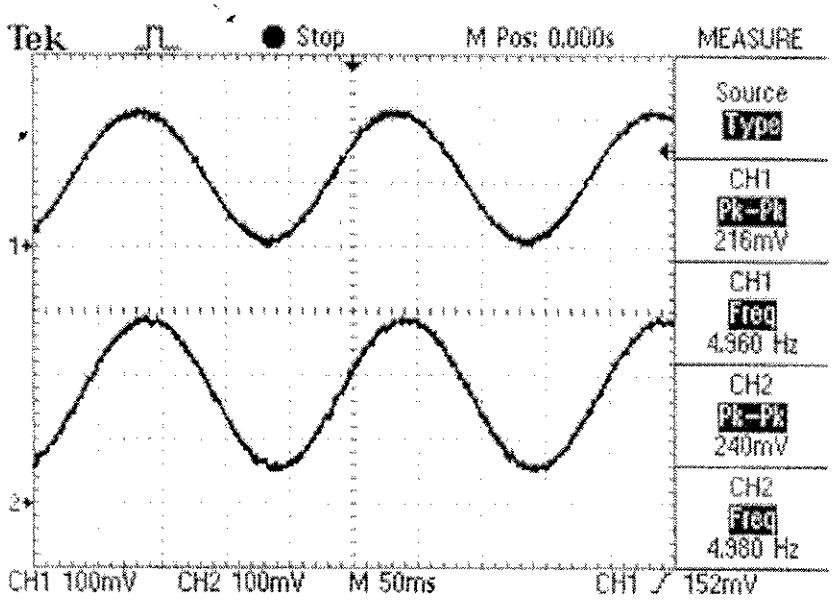
อัตราขยาย (gain, G) = ค่าแรงดันเอาท์พุท (โวลต์) / ค่าแรงดันอินพุท (โวลต์)

อัตราขยาย (gain, dB) = 20*log(G)

จากรูปที่ 4.2 สามารถคำนวณหาค่าอัตราขยายได้ดังนี้

อัตราขยาย = 240 mV/216 mV = 1.11

= 20*log(1.11) = 0.906 dB



รูปที่ 4.2 ภาพสัญญาณอินพุทและเอาท์พุทพร้อมการอ่านค่าแรงดันและความถี่

5. เมื่อทำการวัดค่าแรงดันได้แล้ว หลังจากนั้นก็จะทำการวัดเฟสของสัญญาณ ในการวัดเฟสของสัญญาณนั้นจะวัดจากช่วงเวลาที่ต่างกันแล้วนำมาคำนวณหาค่าเฟสที่ต่างกัน โดยจะทำการเปลี่ยนโหมดการวัด ซึ่งทำได้ดังนี้คือ กดปุ่ม CURSOR ที่หน้าจอจะปรากฏฟังก์ชันของโหมดต่าง ๆ ดังแสดงในรูปที่ 4.3 ให้เลือกที่โหมด Type แล้วกดเลือก Time จะปรากฏ Cursor มาให้ 2 อัน หลังจากนั้นใช้ปุ่ม position แบบ vertical ทั้งของช่องรับสัญญาณที่ 1 และ 2 เพื่อปรับตำแหน่งของ cursor ให้มาอยู่ในตำแหน่งที่ต้องการดังแสดงในรูปที่ 4.3 คือการวัดช่วงเวลาในหนึ่งคาบของสัญญาณ (ซึ่งสัญญาณอินพุทและเอาท์พุทจะมีคาบเวลาเท่ากัน) โดยนำ cursor

1 จับที่ส่วนที่สูงของสัญญาณและนำ cursor 2 จับที่ส่วนที่สูงที่สุดของสัญญาณในลูกคลื่นถัดไป และอ่านค่าเวลาในช่อง Delta

6. ความสัมพันธ์ที่ใช้ในการคำนวณหาค่าความต่างเฟสคือ

เฟสที่ต่างกัน (θ, degree) = (d*360)/D

โดยที่ d คือ ค่าเวลาที่ต่างกันของสัญญาณอินพุทกับเอาท์พุท (วินาที)

D คือ คาบของสัญญาณ (วินาที)

โดยเครื่องหมาย + แสดงถึงเฟสของสัญญาณเอาท์พุทนำหน้าอินพุท

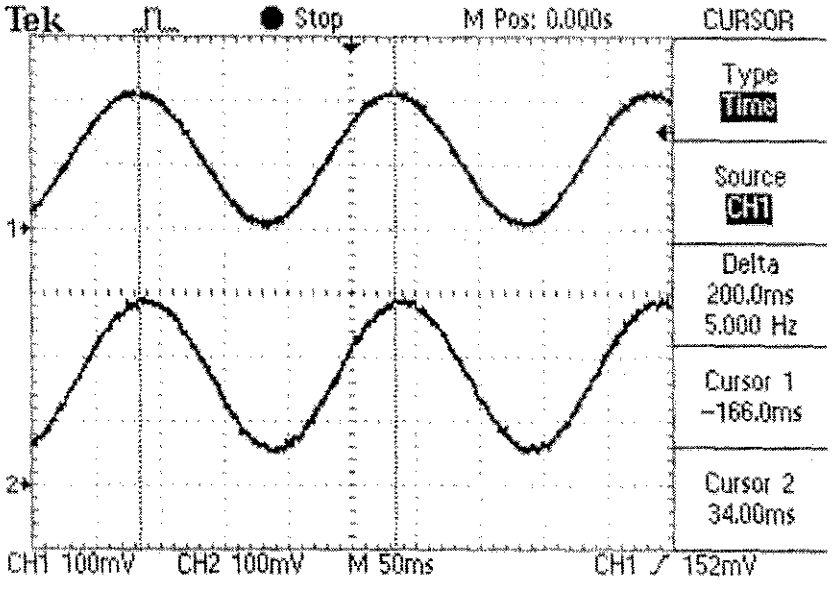
\- แสดงถึงเฟสของสัญญาณเอาท์พุทล้าหลังอินพุท

จากรูปที่ 4.4 สามารถคำนวณหาค่าเฟสที่ต่างกัน (θ) ได้ดังนี้

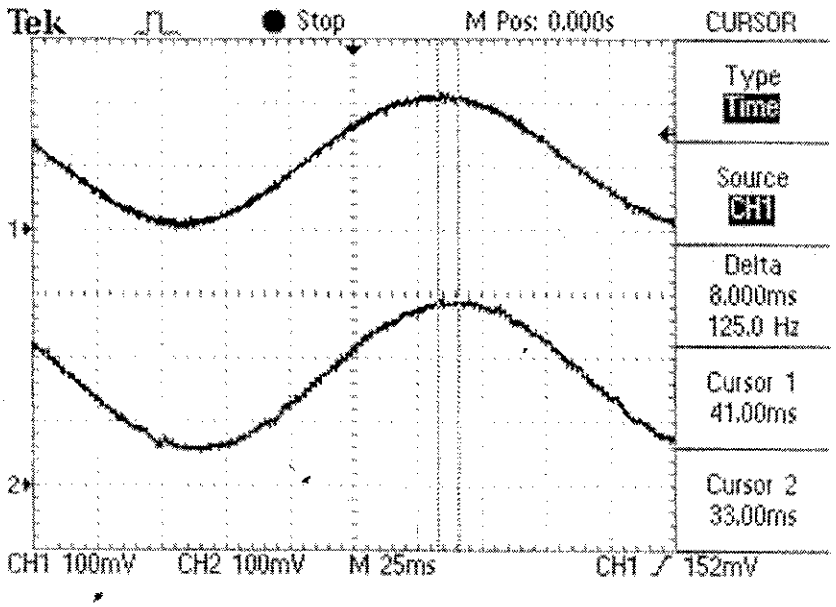
เฟสที่ต่างกัน (θ) = (8 msec*360)/200 msec

= 14.4 องศา

∴ เฟสที่ต่างกันคือ –14.4 องศา เนื่องจากเฟสของสัญญาณเอาท์พุทล้าหลังอินพุท



รูปที่ 4.3 การวัดคาบสัญญาณของสัญญาณอินพุท



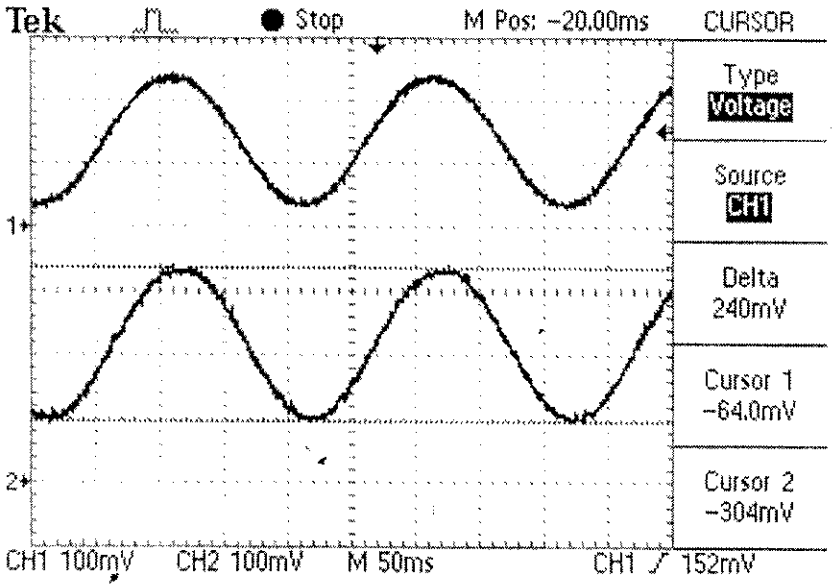
รูปที่ 4.4 การวัดเวลาที่ต่างกันของอินพุทกับเอาท์พุท

หมายเหตุ

การวัดสัญญาณที่ค่าความถี่ต่ำ ๆ อาจทำให้ไม่สามารถวัดค่าแรงดันได้โดยอาศัยผลการวัดจากการใช้ปุ่ม MEASURE วิธีแก้ไขก็คือ ให้มาใช้วิธีการวัดเหมือนการหาค่าเฟสที่ต่างกัน แต่ที่โหมด Type ให้เลือก Voltage ค่า Delta ที่อ่านได้ก็คือ ค่าแรงดันของสัญญาณนั้น วิธีการวัดแสดงในรูปที่ 4.5 แต่มีข้อที่ควรระวังคือ ควรปรับ VOLTS/DIV ให้สามารถอ่านค่าแรงดันได้อย่างชัดเจนเพื่อจะได้อ่านค่าได้อย่างถูกต้อง และที่สำคัญ VOLTS/DIV ของทั้งสองช่องรับสัญญาณควรจะเท่ากันเพื่อจะได้วัดค่าได้อย่างถูกต้องและไม่สับสน

ข้อสังเกต

บางครั้งการวัดโดยใช้โหมด CURSOR จะได้ค่าที่แน่นอนและถูกต้องมากกว่าค่าที่แสดงการวัดโดยอัตโนมัติ เนื่องจากค่าที่แสดงทางจอออสซิลโลสโคปไม่คงที่ (ไม่นิ่ง) จึงไม่สามารถอ่านค่าที่แน่นอนได้



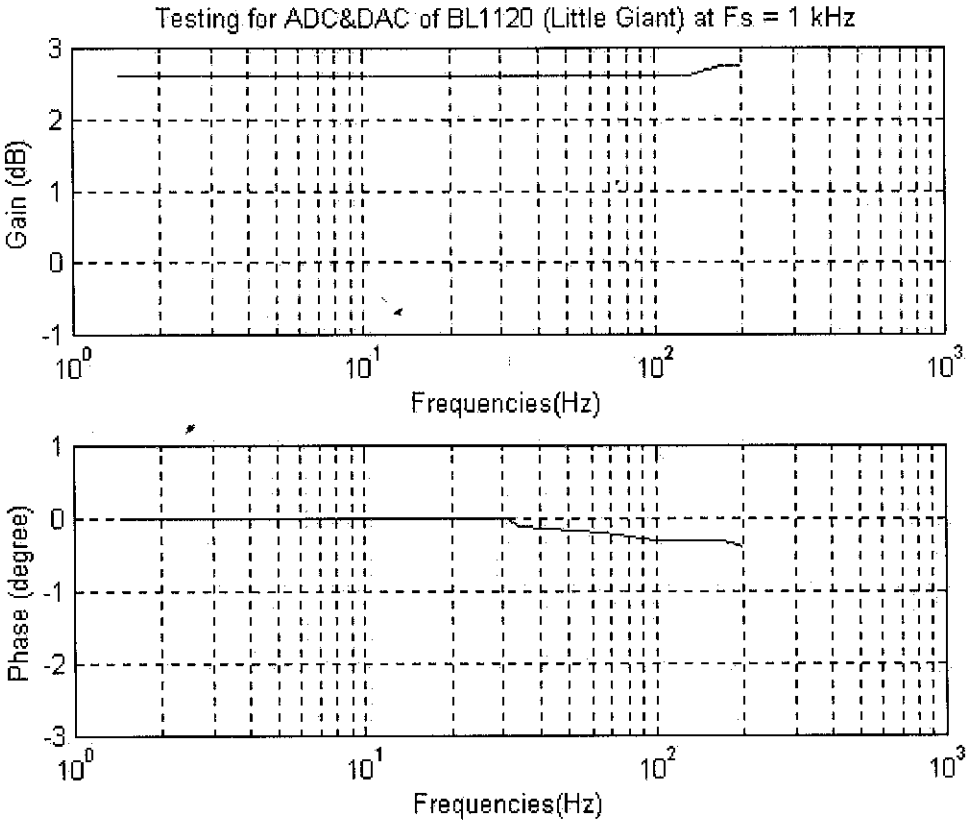
รูปที่ 4.5 การวัดค่าแรงดันกรณีใช้ CURSOR

หลังจากทำการวัดค่าอัตราขยายและเฟสเสร็จเรียบร้อยแล้ว จะต้องนำค่าเหล่านั้นมาคูณกลับด้วยค่าคงที่ (C_1) แล้วจึงนำมาพล็อตกราฟเพื่อดูผลตอบสนองทางความถี่ (Frequency Response) โปรแกรมที่ใช้ในการพล็อตกราฟคือ **test.m** ซึ่งเขียนขึ้นใน MATLAB รายละเอียดต่าง ๆ ของโปรแกรมแสดงไว้ในภาคผนวก ฉ ส่วนการใช้งานโปรแกรมนั้นจะคล้ายคลึงกับโปรแกรมการแปลง s-domain ไปเป็น z-domain แบบทุสตินที่ได้นำเสนอไปแล้วในบทที่ 3

4.3 การทดสอบโปรแกรมการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

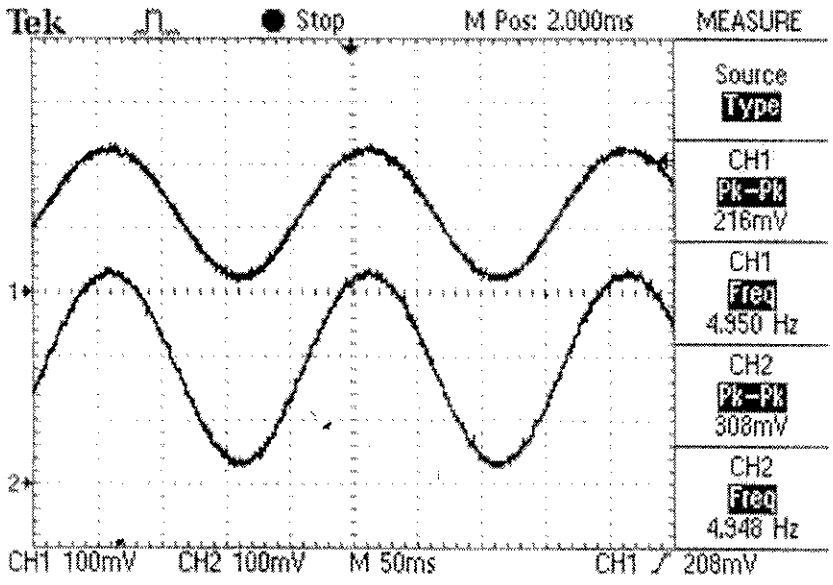
การทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด จะใช้ตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (ADC) และตัวแปลงสัญญาณดิจิตอลเป็นอนาล็อก (DAC) โดยจะทำการเขียนโปรแกรมในไดนามิกส์ซีให้รับอินพุทเข้ามาทาง ADC และส่งออกทันทีทาง DAC ความถี่ของการสุ่มสัญญาณที่เลือกใช้คือ 1,000 Hz และ 500 Hz ซึ่งค่าความถี่ทั้งสองค่านี้ได้นำไปใช้ในการอนุวัตตัวชดเชยดังที่ได้กล่าวไปแล้ว โดยขนาดของสัญญาณอินพุทที่ใช้ในการทดสอบคือ 216 mV_{pp} (เนื่องจากที่สัญญาณอินพุทขนาดต่าง ๆ จะมีค่าเฟสเท่ากันและขนาดของสัญญาณใกล้เคียงกันมาก จึงทำการทดสอบเพียงค่าเดียว) ส่วนโปรแกรมที่ใช้ทดสอบคือ **ADCDAC.c** รายละเอียดของโปรแกรมแสดงไว้ในภาคผนวก ง และผลการทดสอบแสดงไว้ในภาคผนวก จ

ผลการทดสอบที่ความถี่การสุ่มสัญญาณเท่ากับ 1 kHz แสดงในตาราง จ.3 สามารถสรุปได้ดังรูปที่ 4.6

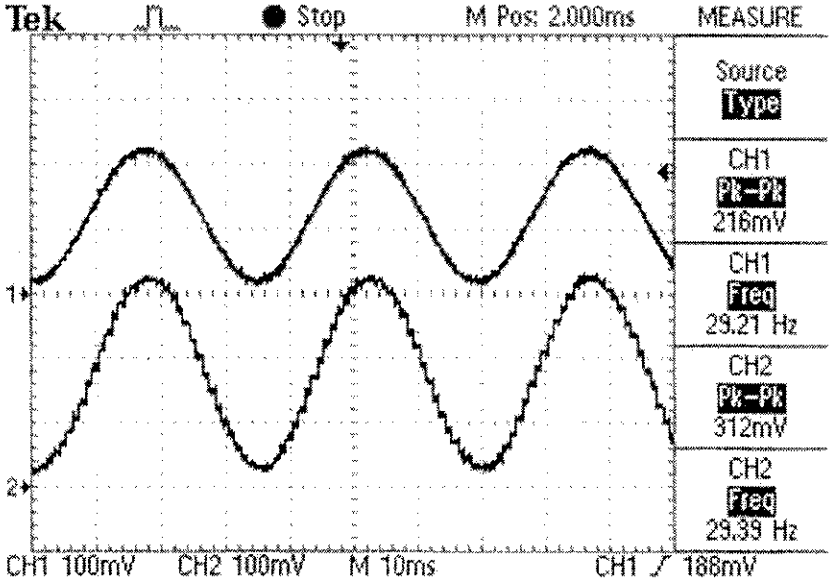


รูปที่ 4.6 อัตราขยายและเฟสของสัญญาณจากการทดสอบโปรแกรมการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด ที่ความถี่ของการสุ่มสัญญาณ 1 kHz และใช้สัญญาณอินพุท 0.216 V_{pp}

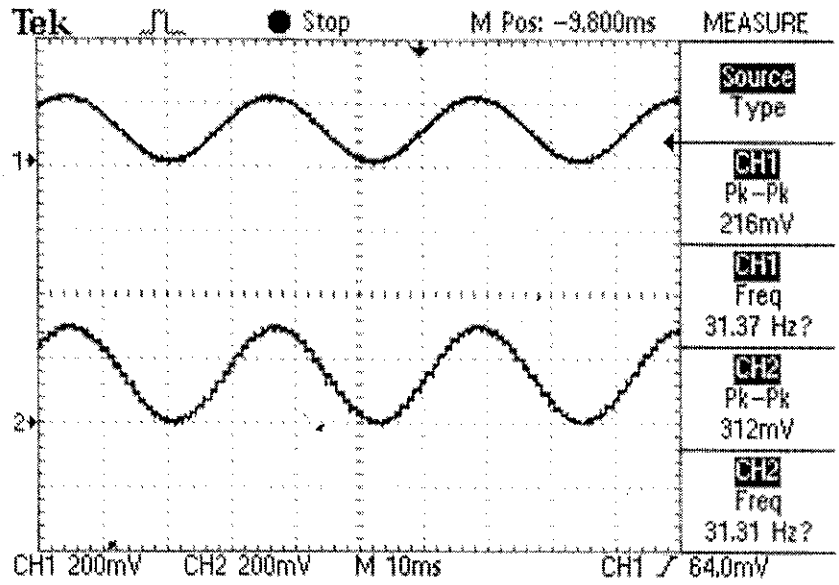
จากรูปที่ 4.6 แสดงให้เห็นว่าช่วงความถี่ที่ให้อัตราขยายเท่ากันประมาณ 0 – 200 Hz และค่าอัตราขยายโดยเฉลี่ยคือ 1.35 หรือประมาณ 2.64 dB



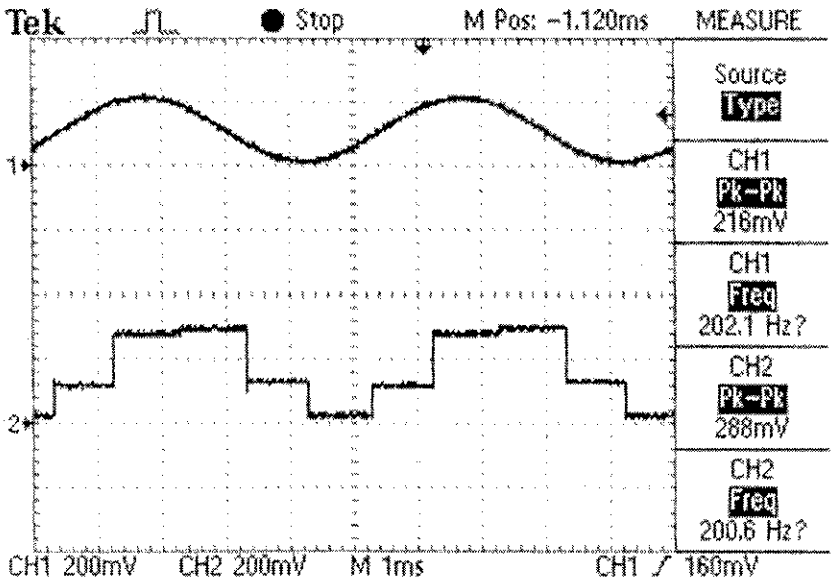
รูปที่ 4.7 รูปสัญญาณที่ความถี่ 4.950 Hz โดยมีความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz



รูปที่ 4.8 รูปสัญญาณที่ความถี่ 29.07 Hz โดยมีความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz

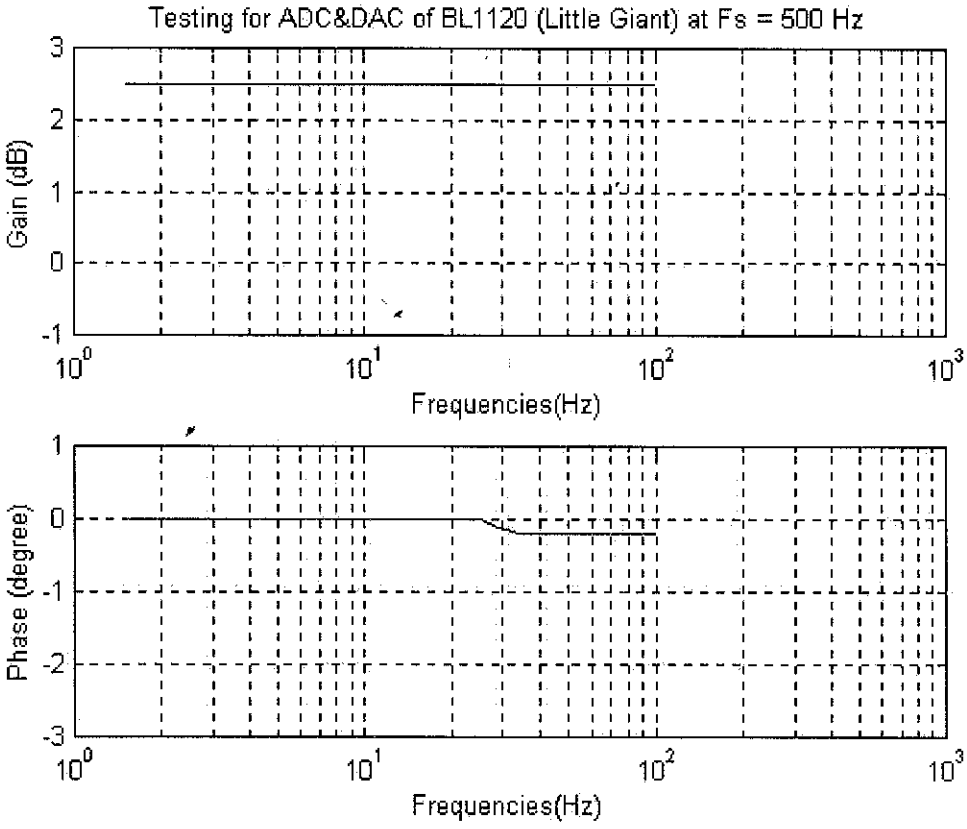


รูปที่ 4.9 รูปสัญญาณที่ความถี่ 31.25 Hz ซึ่งเป็นค่าความถี่ที่มากที่สุดที่สัญญาณอินพุทยังมีเฟสเท่ากับเอาท์พุท โดยความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz



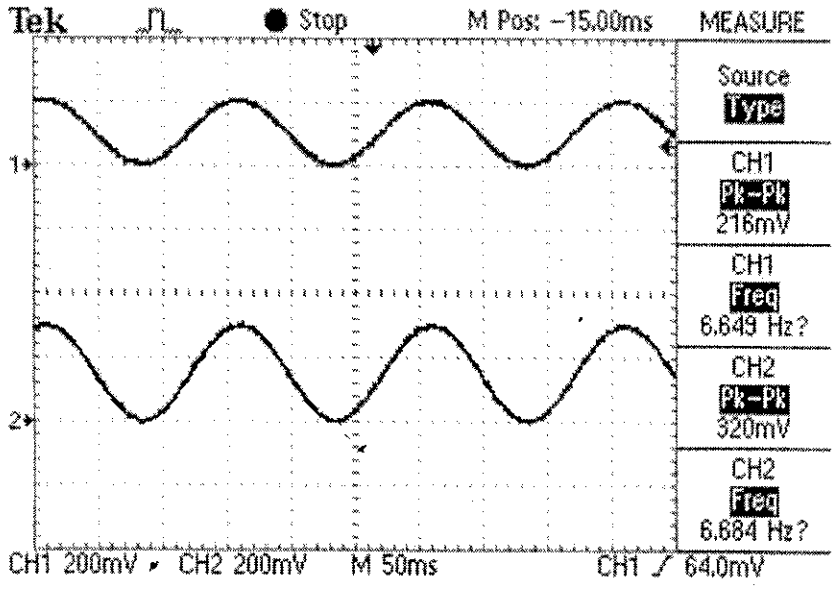
รูปที่ 4.10 รูปสัญญาณที่ความถี่ 200 Hz ซึ่งเป็นค่าความถี่ที่มากที่สุดที่สัญญาณเอาท์พุทยังไม่ผิดเพี้ยนไปจากสัญญาณอินพุท โดยความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz

ส่วนผลการทดสอบที่ความถี่การสุ่มสัญญาณเท่ากับ 500 Hz แสดงในตาราง จ.4 สามารถสรุปได้ดังรูปที่ 4.11

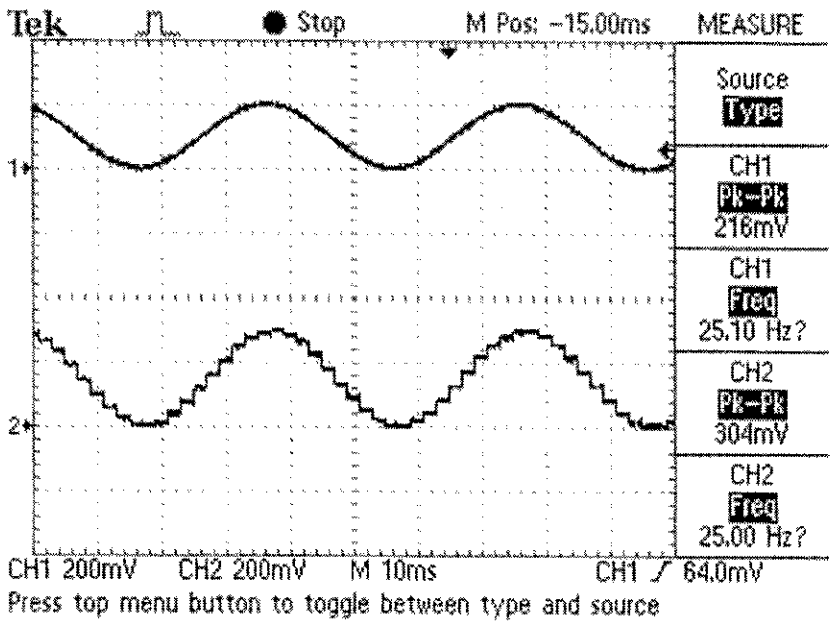


รูปที่ 4.11 อัตราขยายและเฟสของสัญญาณจากการทดสอบโปรแกรมการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด ที่ความถี่ของการสุ่มสัญญาณ 500 Hz และใช้สัญญาณอินพุท 0.216 V_{pp}

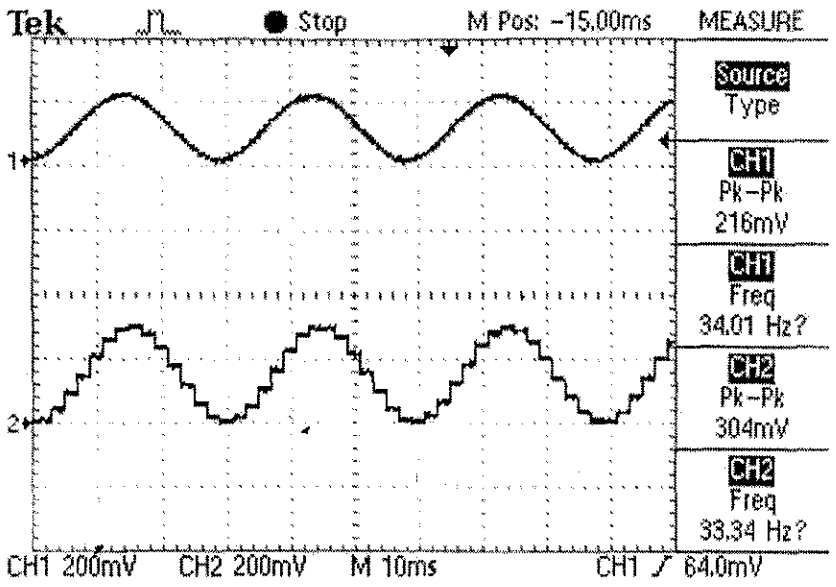
จากรูปที่ 4.11 แสดงให้เห็นว่าช่วงความถี่ที่ให้อัตราขยายเท่ากันประมาณ 0 – 100 Hz และค่าอัตราขยายโดยเฉลี่ยคือ 1.33 หรือประมาณ 2.50 dB



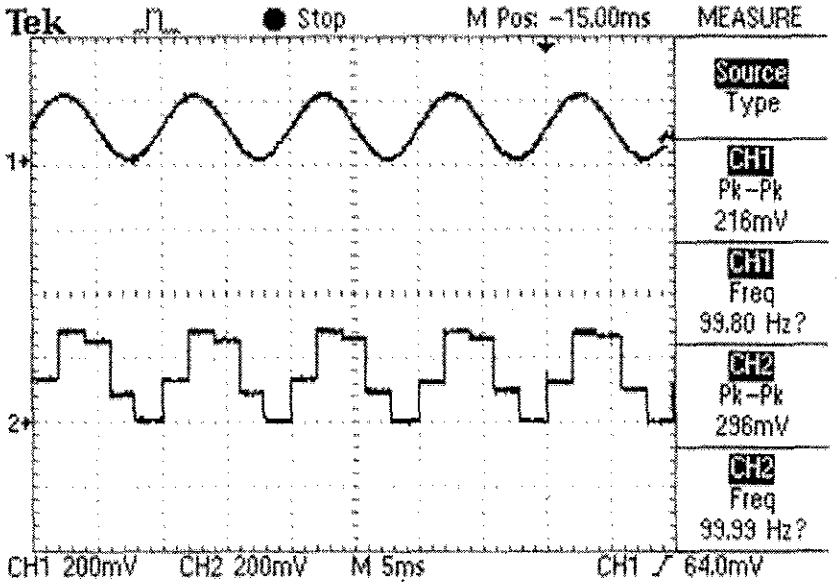
รูปที่ 4.12 รูปสัญญาณที่ความถี่ 6.623 Hz โดยความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz



รูปที่ 4.13 รูปสัญญาณที่ความถี่ 25.25 Hz ซึ่งเป็นค่าความถี่ที่มากที่สุดที่สัญญาณอินพุทยังมีเฟสเท่ากับเอาท์พุท โดยความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz



รูปที่ 4.14 รูปสัญญาณที่ความถี่ 33.78 Hz โดยความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz



รูปที่ 4.15 รูปสัญญาณที่ความถี่ 100 Hz ซึ่งเป็นค่าความถี่ที่มากที่สุดที่สัญญาณเอาท์พุทยังไม่ผิดเพี้ยนไปจากสัญญาณอินพุท โดยความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz

อภิปรายผลการทดสอบ

จากรูปที่ 4.6 และรูปที่ 4.11 จะเห็นได้ว่าอัตราขยายค่อนข้างคงที่ที่ค่า 2.64 dB และ 2.51 dB ตามลำดับ แสดงว่าคอนโทรลเลอร์บอร์ดมีการขยายสัญญาณก่อนที่จะส่งออก ส่วนเฟสในช่วงที่ความถี่ต่ำ ๆ มีค่าเท่ากับศูนย์ และเริ่มมีการเปลี่ยนแปลงเมื่อความถี่มีค่ามากกว่า 31.25 Hz สำหรับรูปที่ 4.6 และ 25.25 Hz สำหรับรูปที่ 4.11 ซึ่งการเปลี่ยนแปลงดังกล่าวมีค่าไม่ถึง 1 องศา (ถือว่ามีการเปลี่ยนแปลงน้อยมากจนอาจประมาณได้ว่ามีค่าเป็นศูนย์) อาจเป็นผลเนื่องมาจากอุปกรณ์ที่ใช้ในการวัด ***ดังนั้นในการทดสอบทั้งหมดจะทำการควบคุมอุปกรณ์ที่ใช้ในการวัดโดยจะใช้อุปกรณ์ชุดเดียวกันตลอดการทดสอบ เพื่อป้องกันการเปลี่ยนแปลงของอัตราขยายและเฟสที่อาจเกิดขึ้นได้***

จากรูปที่ 4.7 ถึงรูปที่ 4.10 และรูปที่ 4.12 ถึงรูปที่ 4.15 พบว่า***ค่าความถี่เพิ่มมากขึ้นจะทำให้สัญญาณเอาท์พุทมีลักษณะเป็นขั้นบันไดชัดเจนขึ้น เนื่องจากไม่มี Low-pass filter ที่ด้านส่งสัญญาณออก*** จึงไม่ได้รูปสัญญาณที่เรียบเมื่อความถี่สูงขึ้น ส่วนที่ความถี่ต่ำยังเห็นรูปสัญญาณที่เป็นคลื่นไซน์ชัดเจนเนื่องมาจากพัลส์ที่สุ่มสัญญาณได้อยู่ชิดกันมาก จึงทำให้มองดูเหมือนรูปสัญญาณที่ได้นั้นเรียบไม่เป็นขั้นบันได

รูปที่ 4.10 และรูปที่ 4.15 แสดงค่าความถี่มากที่สุดที่รูปสัญญาณเอาท์พุทที่ได้ยังไม่ผิดเพี้ยนไปจากสัญญาณอินพุท ซึ่งค่าความถี่นั้นคือ 200 Hz และ 100 Hz ตามลำดับ แสดงให้เห็นว่า***ช่วงความถี่ที่เหมาะสำหรับการทำงานของคอนโทรลเลอร์บอร์ดนี้คือ 0 - 200 Hz กรณีที่ทำการสุ่มสัญญาณที่ 1 kHz และ 0-100 Hz กรณีที่ทำการสุ่มสัญญาณที่ 500 Hz***

จากตาราง จ1 และ จ2 จะเห็นได้ว่าค่าอัตราขยายเฉลี่ยเมื่อทำการทดสอบที่แรงดันอินพุทค่าต่าง ๆ คือ 1.351 หรือ 2.613 dB และ 1.321 หรือ 2.414 dB ตามลำดับ ส่วนช่วงความถี่ที่มีอัตราขยายเท่ากันคือ 0 – 200.8 Hz และ 0 – 101.5 Hz ตามลำดับ แสดงให้เห็นว่าที่ค่าความถี่ในการสุ่มมากขึ้นช่วงความถี่ที่ใช้งานได้จะมากขึ้นด้วย

ดังนั้นในการทดสอบโปรแกรมการอนุวัตตัวชดเชยจะทำการทดสอบที่ความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz ทดสอบในช่วงความถี่ 0 – 200 Hz สำหรับตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและที่ความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz ทดสอบในช่วงความถี่ 0 – 100 Hz สำหรับตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง เนื่องจากที่ค่าความถี่ในการสุ่มสัญญาณดังกล่าวเป็นค่าที่เหมาะสมกับทรานสเฟอร์ฟังก์ชันอันดับนั้น ๆ ส่วนในเรื่องอัตราขยายที่เกิดขึ้นเนื่องจากการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด จึงจำเป็นต้องนำค่าอัตราขยายนั้นไปหารออกจากสัญญาณเอาท์พุทที่จะส่งออก

4.4 การทดสอบโปรแกรมการอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

โปรแกรมการอนุวัตตัวชดเชยแบ่งออกเป็น 2 โปรแกรมด้วยกันคือ โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง ส่วนโปรแกรมที่ใช้ทดสอบจะใช้เพียงโปรแกรมเดียว เป็นโปรแกรมที่เขียนขึ้นใน MATLAB ชื่อ **test.m** วิธีที่ใช้ในการทดสอบคือ ในขั้นแรกจะทำการวัดขนาดและเฟสของสัญญาณทั้งอินพุทและเอาท์พุทและนำมาคำนวณค่าอัตราขยาย หลังจากนั้นจะใช้การพล็อตกราฟดูผลตอบสนองทางความถี่ ซึ่งในตัวโปรแกรมจะใส่ค่าอัตราขยายและเฟสที่ได้หลังจากการทดสอบโดยการวัดลงไปโดยตรง จากผลการทดสอบในภาคผนวก ช (ถ้าต้องการทดสอบกับตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันต่างออกไป สามารถนำค่าที่วัดได้มาใส่ในส่วนการกำหนดค่าอัต์ราขยายและเฟส ดูรายละเอียดได้จากภาคผนวก ฉ) การรันโปรแกรมจะให้เลือกอันดับของทรานสเฟอร์ฟังก์ชัน แล้วโปรแกรมจะทำการพล็อตผลตอบสนองทางความถี่มาให้ โดยมีรายละเอียดดังนี้

4.4.1 การทดสอบโปรแกรมอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

ทรานสเฟอร์ฟังก์ชันอันดับหนึ่งของตัวชดเชยใน s-domain คือ

$$H(s) = \frac{s + 30}{s + 20}$$

หลังจากทำการแปลงให้อยู่ใน z-domain โดยอาศัยความสัมพันธ์ของทุสติน จะได้ทรานสเฟอร์ฟังก์ชันในรูป z-domain ดังนี้

$$H(z) = \frac{2.030002 - 1.969998z^{-1}}{2.020001 - 1.979999z^{-1}}$$

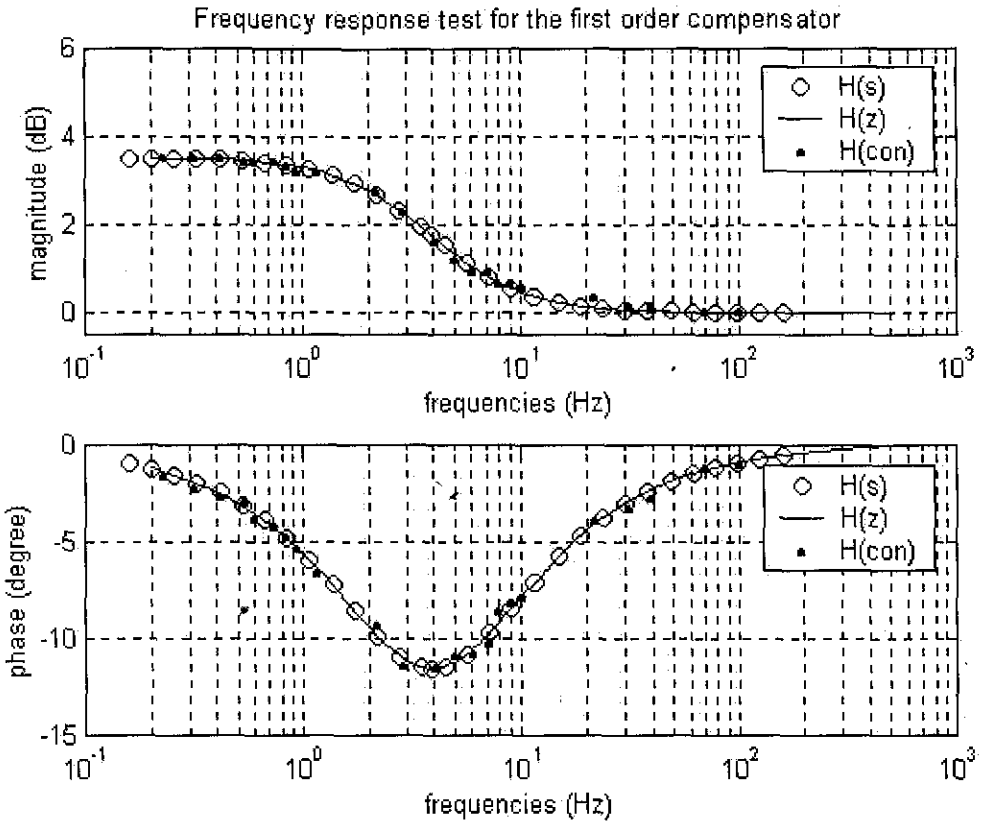
$$H(z) = 1.004950\left(\frac{1 - 0.970442z^{-1}}{1 - 0.980197z^{-1}}\right)$$

เมื่อนำมาเปลี่ยนเป็นค่าสัมประสิทธิ์ของตัวชดเชยที่ใช้ในการอนุวัต จะมีค่าดังนี้

$$a_{11} = -0.970442$$
$$b_{11} = -0.980197$$

สาเหตุที่ใช้สัมประสิทธิ์เพียง 2 ตัวในการอนุวัตตัวชดเชยเนื่องมาจากค่าสัมประสิทธิ์ตัวอื่นมีค่าเป็น 1 ดังที่กล่าวในบทที่ 3 หลังจากทำการวัดสัญญาณเสร็จสิ้นแล้วจะต้องนำค่าอัตราขยายที่วัดได้ที่เรียกว่า G_{disp} (Display gain) มาคูณกับค่าคงที่ที่ดึงออกมาจากทรานสเฟอร์ฟังก์ชัน (C_1) ก่อนนำไปใช้ในโปรแกรมการอนุวัตซึ่งมีค่าเท่ากับ 1.004951 ค่าอัตราขยายที่ได้ใหม่จะเรียกว่า G_{act} (Actual gain) หลังจากนั้นจึงนำค่านี้ไปพล็อตกราฟเพื่อดูผลตอบสนองทางความถี่

ผลการทดสอบตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) โดยโปรแกรม order1.c ในไดนามิกส์ซี และนำค่าที่ได้จากการทดสอบมาใช้ในโปรแกรม test.m โดยเลือกทดสอบกรณีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง ที่ความถี่ของการสุ่มสัญญาณเท่ากับ 1 kHz แสดงไว้ในรูปที่ 4.16 ส่วนค่าอัตราขยายและเฟสของ H(con) ที่ได้จากการทดสอบแสดงไว้ใน ภาคผนวก ช ในตาราง ช.1



รูปที่ 4.16 ผลตอบสนองทางความถี่ของตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง ที่ความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz

<u>อภิปรายผลการทดสอบ</u>

จากรูปที่ 4.16 จะเห็นได้ว่าอัตราขยายของทรานสเฟอร์ฟังก์ชันใน s-domain H(s) และ z-domain H(z) มีค่าเท่ากัน เนื่องจาก H(z) มี dc gain เท่ากับ H(s) คือ 1 (dc gain ของ H(z) = 1.004951 $\approx$ 1) ดังที่ได้กล่าวแล้วว่าการแปลงแบบทุสตินจะต้องมีการปรับ dc gain (K_d) ที่คูณควบอยู่กับ H(z) เพื่อทำให้ H(z) มี dc gain เท่ากับ H(s) ดั้งเดิม ส่วนอัตราขยายที่ได้จากการอนุวัตโดยใช้คอนโทรลเลอร์บอร์ด H(con) จะใกล้เคียงกับ H(s) และ H(z) มาก

นอกจากนี้จะเห็นว่าเฟสของ H(con) จะกระจายเล็กน้อย แต่แนวโน้มยังเกาะกลุ่มอยู่กับ H(s) และ H(z) ลักษณะการกระจายดังกล่าวเกิดขึ้นเนื่องจากความไม่แน่นอน (uncertainty) ของค่าเฟสที่วัดได้ ซึ่งอาจมีสาเหตุมาจาก phase jitter ในอุปกรณ์อิเลคทรอนิคส์ที่ใช้เป็นส่วนประกอบของ Function generator

4.4.2 การทดสอบโปรแกรมอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

ทรานสเฟอร์ฟังก์ชันอันดับสองของตัวชดเชยใน s-domain คือ

$$H(s) = \frac{s^2 + 15s + 15}{s^2 + 14s + 14}$$

หลังจากทำการแปลงให้อยู่ใน z-domain โดยอาศัยคววมสัมพันธ์ของทุสตินแล้ว จะได้ทรานสเฟอร์ฟังก์ชันในรูป z-domain ดังนี้

$$H(z) = \frac{4.060000 - 1.970443z^{-1} + 0.970443z^{-2}}{4.056000 - 1.972386z^{-1} + 0.972386z^{-2}}$$

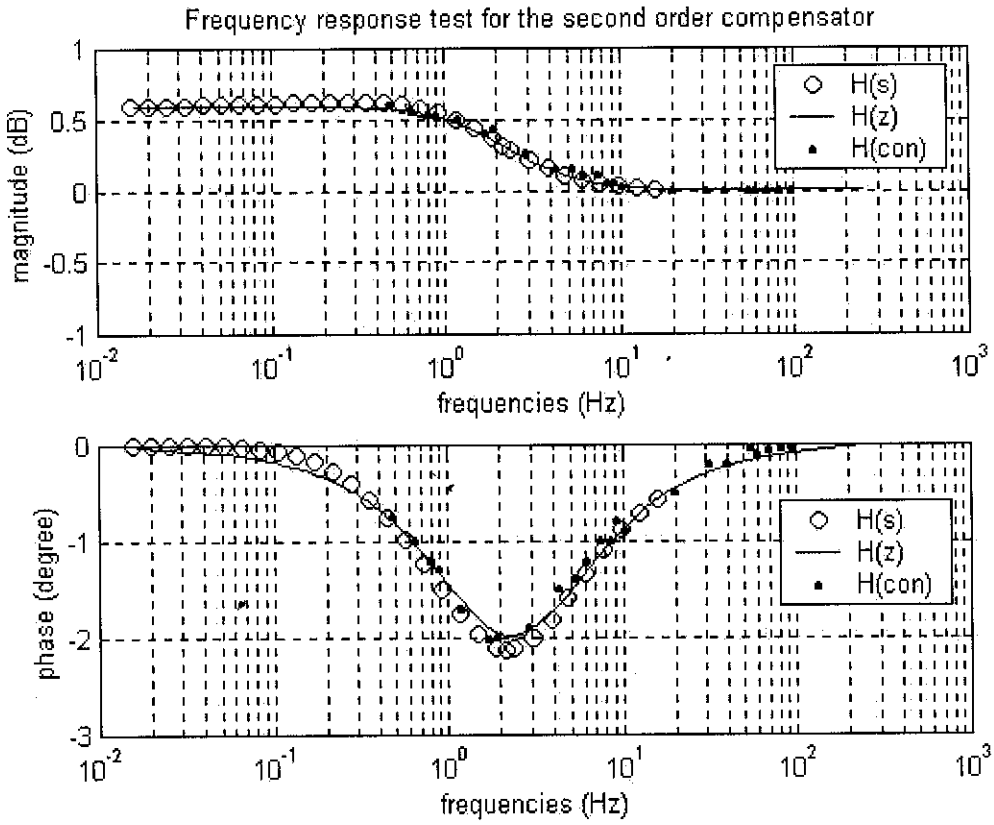
$$H(z) = 1.000986\left(\frac{1 - 0.485331\,z^{-1} + 0.239025\,z^{-2}}{1 - 0.486289z^{-1} + 0.239740z^{-2}}\right)$$

เมื่อนำมาเปลี่ยนเป็นค่าสัมประสิทธิ์ของตัวชดเชยที่ใช้ในการอนุวัต จะมีค่าดังนี้

$$a_{11} = -0.485331$$
$$a_{21} = 0.239025$$
$$b_{11} = -0.486289$$
$$b_{21} = 0.239740$$

สาเหตุที่ใช้สัมประสิทธิ์เพียง 4 ตัวในการอนุวัตเนื่องมาจากค่าสัมประสิทธิ์ตัวอื่นมีค่าเป็น 1 หลังจากทำการวัดสัญญาณเสร็จสิ้นแล้วจะต้องนำค่าอัตราขยาย G_{disp} มาคูณกับค่าคงที่ที่ดึงออกมาจากทรานสเฟอร์ฟังก์ชัน (C_2) ก่อนนำไปใช้ในโปรแกรมการอนุวัตตัวชดเชยซึ่งมีค่าเท่ากับ 1.000986 ค่าอัตราขยายที่ได้คือ G_{act} หลังจากนั้นจึงนำค่านี้ไปพล็อตกราฟเพื่อดูผลตอบสนองทางความถี่

ผลการทดสอบตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสองด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) โดยโปรแกรม order2.c ในไดนามิกส์ซี และนำค่าที่ได้จากการทดสอบมาใช้ในโปรแกรม test.m โดยเลือกทดสอบกรณีทรานสเฟอร์ฟังก์ชันอันดับสอง ที่ความถี่ของการสุ่มสัญญาณเท่ากับ 500 Hz แสดงไว้ในรูปที่ 4.17 ส่วนค่าอัตราขยายและเฟสของ H(con) ที่ได้จากการทดสอบแสดงไว้ใน ภาคผนวก ช ในตาราง ช.2



รูปที่ 4.17 ผลตอบสนองทางความถี่ของตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง ที่ความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz

<u>อภิปรายผลการทดสอบ</u>

จากรูปที่ 4.17 จะเห็นได้ว่าอัตราขยายของทรานสเฟอร์ฟังก์ชันใน s-domain H(s) และ z-domain H(z) มีค่าเท่ากัน เนื่องจาก H(z) มี dc gain เท่ากับ H(s) คือ 1 (dc gain ของ H(z) = 1.000986 ≈ 1) ดังที่ได้กล่าวแล้วว่าการแปลงแบบทุสตินจะต้องมีการปรับ dc gain (K_d) ที่คูณควบอยู่กับ H(z) เพื่อทำให้ H(z) มี dc gain เท่ากับ H(s) ดั้งเดิม ส่วนอัตราขยายที่ได้จากการอนุวัตโดยใช้คอนโทรลเลอร์บอร์ด H(con) จะใกล้เคียงกับ H(s) และ H(z) มาก เหมือนกับกรณีของตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

จากรูปเดียวกันจะเห็นได้ว่าเฟสของ H(con) จะเกาะกลุ่มกระจายอยู่กับ H(z) มากกว่า ลักษณะการกระจายได้กล่าวแล้วในส่วนของการอภิปรายผลการทดสอบในหัวข้อ 4.4.2

สรุปผลการทดสอบ

การอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ให้อัตราขยายและเฟสใกล้เคียงกับ H(z) มาก เนื่องมาจากการอนุวัตตัวชดเชยนั้นมีรากฐานมาจาก H(z) และ H(z) นี้ได้มาจากการแปลงโดยใช้เทคนิคของทุสตินที่ทำการปรับแก้อาการโก่งทางความถี่แล้ว ซึ่งในการแปลงแบบทุสตินนี้จะต้องมีการปรับ dc gain ของ H(s) ให้เท่ากับ H(z) ก่อนนำมาใช้งานจริง นอกจากนั้นผลการทดสอบได้แสดงให้เห็นว่าหลังจากทำการปรับแก้อาการโก่งทางความถี่และปรับ dc gain แล้ว การแปลงแบบทุสตินจะให้ผลที่ใกล้เคียงกับ s-domain มาก

บทที่ 5

ข้อสรุปและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงข้อสรุปและข้อเสนอแนะต่างๆ ที่เกี่ยวกับการอนุวัตตัวชดเชย โดยมีรายละเอียดดังต่อไปนี้

5.1 ข้อสรุป

1. โครงสร้างที่ใช้ในการอนุวัตตัวชดเชย จะเหมือนกับโครงสร้างของตัวกรองความถี่แบบดิจิตอลชนิด IIR (Infinite Impulse Response) เนื่องมาจากตัวชดเชยที่ใช้งานกันส่วนใหญ่จะอนุวัตมาจากทรานสเฟอร์ฟังก์ชันที่อยู่ในรูปที่มีโพลและซีโร่
2. เทคนิคการแปลงทรานสเฟอร์ฟังก์ชันใน s-domain ไปเป็น z-domain แบบทุสตินนั้นจะต้องทำการปรับค่า dc gain ก่อนนำมาใช้งานจริง โดยจะต้องทำให้ค่า dc gain ที่คูณควบอยู่กับ H(z) มีค่าเท่ากับ dc gain ใน H(s) นอกจากนั้นการแก้อาการโก่งทางความถี่ (frequency prewarping) เนื่องมาจากการแปลงแบบทุสตินทำให้ค่า H(s) ใกล้เคียงกับ H(z) มากขึ้น
3. การอนุวัตตัวชดเชยด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ให้อัตราขยายและเฟสใกล้เคียงกับ H(z) มากกว่า เนื่องจากโครงสร้างที่นำมาใช้ในการอนุวัตมีรากฐานมาจากทรานสเฟอร์ฟังก์ชันใน z-domain
4. ช่วงความถี่ที่ใช้งานได้ของการอนุวัตตัวชดเชยถูกจำกัดด้วยคุณสมบัติของคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ที่ให้อัตราขยายเท่ากันและรูปสัญญาณไม่ผิดเพี้ยนในช่วงความถี่หนึ่ง

5.2 ข้อเสนอแนะ

การพัฒนาโปรแกรมการอนุวัตตัวชดเชยให้มีประสิทธิภาพและความถูกต้องมากยิ่งขึ้นเพื่อให้สามารถนำไปใช้งานได้จริงนั้นสามารถทำการปรับปรุงและเพิ่มเติมได้ดังนี้

1. ถ้าต้องการให้รูปสัญญาณเรียบขึ้นหรือการเป็นขั้นบันไดลดน้อยลง สามารถเพิ่ม Low-pass filter ที่ด้านเอาท์พุทก่อนที่จะส่งสัญญาณออก แต่ในขณะเดียวกันเมื่อทำการเพิ่ม Low-pass filter แล้วย่อมมีผลกระทบทางด้านอัตราขยายและเฟสตามมาจากตัว Low-pass filter เอง ดังนั้นทางด้านส่งสัญญาณออกควรมีอุปกรณ์ที่ช่วยทำให้อัตราขยายและเฟสของสัญญาณเท่ากับอัตราขยายที่แท้จริง (actual gain, G_{act}) และเฟสที่แท้จริง ที่ความถี่ในการสุ่มสัญญาณ
2. เทคนิคการแปลงทรานสเฟอร์ฟังก์ชันมีด้วยกันหลายเทคนิค ในการนำไปประยุกต์ใช้งานทางเวลาจริง (real-time) อาจใช้เทคนิคอื่นที่เหมาะสมก็ได้ เช่น Schneider's rule เป็นต้น
3. การกวัดแกว่งแบบดำรงตัว (Sustained oscillation) จะมีผลค่อนข้างมากในการใช้งานทรานสเฟอร์ฟังก์ชันอันดับสูง ๆ ผู้ใช้อาจต้องทำการเพิ่มควอนไทเซอร์เข้าไปในโครงสร้างการอนุวัตเพื่อลดการกวัดแกว่งดังกล่าว

บรรณานุกรม

Andreas A. (1938). Digital Filters: analysis, design, and applications, 2nd edition. New York : McGraw-Hill, Inc.

Bishop, R. H. and Dorf, R. C. Modern Control Systems. New York: Addison Wesley, Inc.

Dynamics C 5.x: Technical Reference. (1998). Z-World, Inc.: The United States of America.

Dynamics C 5.x: Function Reference. (1998). Z-World, Inc.: The United States of America.

Dynamics C 5.x: Application Framework. (1998). Z-World, Inc.: The United States of America.

Hartley, T. T., Beale, G. O. and Chicatelli, S. P. (1964). Digital Simulation of Dynamics System: a control theory approach. Englewood Cliffs NJ: Prentice-Hall

Ifeachor, E. C. and Jervis, B. W. (1993). Digital Signal Processing: a practical approach. Dorset: Addison-Wesley Publishing Company Inc.

Ingle, V. K. and Proakis, J. G. Digital Signal Processing Using MATLAB V.4. Boston: PWS Publishing Company.

Kuo, B. C. (1930). Digital Control Systems. New York: Holt, Rinehart and Winston

Little Giant: C-Programmable Miniature Controller. (1995). Z-World, Inc.: The United States of America.

Mitra, S. K. and Kaiser, J. F. (1993). Handbook for digital signal processing. New York: J. Wiley & Sons.

ภาคผนวก

ภาคผนวก ก

โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง

(โปรแกรม order1.c ในไดนามิกส์ซี)

```
*************************************************************************
*   โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง
*   จัดทำโดย : นางสาวศศิธร อนุรักษ์ปัญญา B3904602
*   อาจารย์ที่ปรึกษา:  รศ.ดร. สราวุฒิ สุจิตจร
*                       อาจารย์สมศักดิ์ วาณิชอนันต์ชัย
*   ลิขสิทธิ์เป็นของมหาวิทยาลัยเทคโนโลยีสุรนารี , พ.ศ. 2543
*************************************************************************
#use drivers.lib
#use stdio.lib
#use bl11xx.lib
#use vdriver.lib

shared int yn1,yn,xn1,xn,ynn;        /* used by interrupt routine and main */
shared float a11,b11;
shared char eval;
shared unsigned long int counter;    /* shared between different interrupt levels */

main()
{
        outport(DCNTL,0x30);

        yn1=0;
        yn=0;
        xn1=0;
        xn=0;
        ynn=0;
        a11=0;
        b11=0;
        eval=0;

        a11 = -0.970442;              /* initialize coefficients */
        b11 = -0.980197;             /* sampling interval at 1 msec */

        setdaisy(3);                  /* setup KIO daisy chain */
        setctc(2,1,48,1);             /* initialize ctc,enable interrupts */
        counter=0;
        while(1)
        {
                hitwd();
                if(eval)
                {
                evaluate1();
                eval=0;
```

```
            }
            outport(ENB485,!(counter&64));      /* flash light */
        }
}
/*.....*/

#INT_VEC CTC2_VEC ccc           /* set interrupt vector for interrupt routine */

/* this interrupt routine sets eval to 1 */

interrupt reti ccc()
{
        eval=1;
        counter++;
}

/******* FILTER_1 ********/
evaluate1()
{
        xn=ad_rd12(6);                  /* 12 bit ADC  ( channel 6, bipolar) */
        yn=xn+(a11*xn1)-(b11*yn1);      /* difference equation */
        xn1=xn;                         /* shift and save delay data */
        yn1=yn;
        ynn=yn/1.35;
        wdac(ynn);                      /* DAC (output) */
}

/*.....*/
```

ภาคผนวก ข

โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง

(โปรแกรม order2.c ในไดนามิกส์ซี)

```
*************************************************************************
*   โปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง
*   จัดทำโดย : นางสาวศศิธร อนุรักษ์ปัญญา B3904602
*   อาจารย์ที่ปรึกษา:  รศ.ดร. สราวุฒิ สุจิตจร
*                       อาจารย์สมศักดิ์ วาณิชอนันต์ชัย
*   ลิขสิทธิ์เป็นของมหาวิทยาลัยเทคโนโลยีสุรนารี , พ.ค. 2543
*************************************************************************
#use drivers.lib
#use stdio.lib
#use bl11xx.lib
#use vdriver.lib

shared int yn1,yn,xn1,xn,yn2,xn2,ynn;
                        /* declare variables that used by interrupt routine and main */
shared float a11,b11,a21,b21;
shared char eval;
shared unsigned long int counter;       /* shared between different interrupt levels */

main()
{
        outport(DCNTL,0x30);

        xn=0;
        yn=0;
        xn1=0;
        xn2=0;
        yn1=0;
        yn2=0;
        ynn=0;
        a11=0;
        a21=0;
        b11=0;
        b21=0;
        eval=0;

        a11 = -0.485331;        /* initialize coefficients at 1 kHz sampling rate */
        a21 = 0.239025;
        b11 = -0.486289;
        b21 = 0.239740;

        setdaisy(3);            /* setup KIO daisy chain */
        setctc(2,1,96,1);       /* initialize ctc,enable interrupts */
        counter=0;
```

```
        while(1)
        {
                hitwd();
                if(eval)
                {
                evaluate2();
                eval=0;
                }
                outport(ENB485,!(counter&64));      /* flash light */
        }
}
/*.....*/

#INT_VEC CTC2_VEC ccc           /* set interrupt vector for interrupt routine */

/* this interrupt routine sets eval to 1 */

interrupt reti ccc()
{
        eval=1;
        counter++;
}

/******* FILTER_2 ********/
evaluate2()
{
        xn=ad_rd12(6);                  /* 12 bit ADC  ( channel 6, bipolar) */

        yn=xn+(a11*xn1)+(a21*xn2)-(b11*yn1)-(b21*yn2);
                                        /* difference equation */
        xn2=xn1;                        /* shift and save delay data */
        xn1=xn;
        yn2=yn1;
        yn1=yn;

        ynn=yn+2047;
        ynn=ynn/1.32;
        wdac(ynn);                      /* DAC (output) */
        hitwd();
}

/*.....*/
```

ภาคผนวก ค

โปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain แบบทุสติน

โปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain แบบทุสติน

(โปรแกรม tustin.m ใน Matlab)

```
**************************************************************************
*   โปรแกรมการแปลงทรานสเฟอร์ฟังก์ชันจาก s-domain ไปเป็น z-domain
*   จัดทำโดย : นางสาวศศิธร อนุรักษ์ปัญญา B3904602
*   อาจารย์ที่ปรึกษา:  รศ.ดร. สราวุฒิ สุจิตจร
*                      อาจารย์สมศักดิ์ วาณิชอนันต์ชัย
*   ลิขสิทธิ์เป็นของมหาวิทยาลัยเทคโนโลยีสุรนารี , พ.ค. 2543
**************************************************************************
disp('Relation of Tustin for coefficients');
disp('Select order of transfer function 1 or 2:');
n=input('Enter order:');

if(n==1)
  disp('Transfer function for the first order compensator:');
  disp('          s+a');
  disp('G(s)=  -----');
  disp('          s+b');
  h=input('Enter h in second: ');
  a=input ('Enter for a:');
  b=input ('Enter for b:');

  m=2/h;
  alpha=m*tan(a*h/2);
  beta=m*tan(b*h/2);
  e=alpha*h;
  f=beta*h;
  a0=e+2;
  a1=(e-2);
  b0=f+2;
  b1=(f-2);
  fprintf('a0=%f\n a1=%f\n bo=%f\n b1=%f\n',a0,a1,b0,b1);
  c1=a0./b0;
  a11=a1./a0;
  b11=b1./b0;

  disp('          1+a11*z^(-1)');
  disp('G(z)= c1 ----------------------');
  disp('          1+b11*z^(-1)');

  disp('y(n) = x(n) + a11*x(n-1) - b11*y(n-1)');
  fprintf('c1=%f\n a11=%f\n b11=%f\n',c1,a11,b11);

else
  disp('Transfer function for the second order compensator:');
```

```
disp('          (s^2+cs+d)');
disp('G(s)= ----------------');
disp('          (s^2+as+b)');
h=input('Enter h in unit:');
a=input('Enter for a:');
b=input('Enter for b:');
c=input('Enter for c:');
d=input('Enter for d:');
bn=sqrt(b);
dn=sqrt(d);
wp=h*bn/2;
wz=h*dn/2;
m=2/h;
wp=m*tan(wp);
wz=m*tan(wz);

r=c./dn*wz;
w=a./bn*wp;
a_0=2*r*h;
b_0=2*w*h;
a0=4+a_0;
a1=-8./a0;
a2=(4-a_0)./a0;
b0=4+b_0;
b1=-8./b0;
b2=(4-b_0)./b0;

fprintf('a0=%f\n a1=%f\n a2=%f\n b0=%f\n b1=%f\n b2=%f\n',a0,a1,a2,b0,b1,b2);

c2=a0./b0;
a11=a1./a0;
a21=a2./a0;
b11=b1./b0;
b21=b2./b0;

disp('          1+a11*z^(-1)+a21*z^(-2)');
disp('G(z)=  c2 ------------------------------------');
disp('          1+b11*z^(-1)+b21*z^(-2)');

disp('y(n) = x(n) + a1*x(n-1) + a2*x(n-2) - b1*y(n-1) - b2*y(n-2)');
fprintf('c2 = %f \n a11 = %f \n a21 = %f \n b11 = %f \n b21 = %f\n'
,c2,a11,a21,b11,b21);
end
```

ภาคผนวก ง

โปรแกรมทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

โปรแกรมทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

(โปรแกรม ADCDAC.c ในไดนามิกส์ซี)

```
*************************************************************************
*  โปรแกรมทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)
*  จัดทำโดย : นางสาวศศิธร อนุรักษ์ปัญญา B3904602
*  อาจารย์ที่ปรึกษา:  รศ.ดร. สราวุฒิ สุจิตจร
*                      อาจารย์สมศักดิ์ วาณิชอนันต์ชัย
*  ลิขสิทธิ์เป็นของมหาวิทยาลัยเทคโนโลยีสุรนารี , พ.ค. 2543
*************************************************************************
#use drivers.lib
#use stdio.lib
#use bl11xx.lib
#use vdriver.lib

shared int xn;                          /* used by interrupt routine and main */
shared char eval;
shared unsigned long int counter;       /* shared between different interrupt levels */

main()
{
        outport(DCNTL,0x30);
        xn=0;

        setdaisy(3);                    /* setup KIO daisy chain */
        setctc(2,1,96,1);               /* initialize ctc,enable interrupts */
        counter=0;
        while(1)
        {
                hitwd();
                if(eval)
                {
                    test();
                    eval=0;
                }
                outport(ENB485,!(counter&64)); /* flash light */
        }
}

/*.....*/

#INT_VEC CTC2_VEC ccc                   /* set interrupt vector for interrupt routine */

/* this interrupt routine sets eval to 1 */

interrupt reti ccc()
```

```
{
        eval=1;
        counter++;
}

test()
{
        xn=ad_rd12(6);                  /* 12 bit ADC  ( channel 6, bipolar) */
        wdac(xn);                       /* DAC (output) */
}

/*.....*/
```

ภาคผนวก จ

ผลการทดสอบโปรแกรมการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant)

ตาราง จ.1 ค่าอัตราขยายและช่วงความถี่ที่มีค่าอัตราขยายเท่ากันจากการทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด ที่ความถี่ในการสุ่มสัญญาณ 1 kHz โดยทำการเปลี่ยนแปลงขนาดของสัญญาณอินพุท พร้อมแสดงค่าเฉลี่ยที่นำไปใช้ในการอนุวัต

แรงดันอินพุท (V_{pp})	แรงดันเอาท์พุท (V_{pp})	อัตราขยาย	อัตราขยาย (dB)	ช่วงความถี่ที่มีอัตราขยายเท่ากัน	
				f_{min}	f_{max}
0.216	0.292	1.352	2.619	0	200.0
0.224	0.304	1.357	2.653	0	203.0
0.232	0.315	1.358	2.656	0	201.6
0.240	0.320	1.333	2.499	0	199.7
0.248	0.336	1.355	2.638	0	199.5
	ค่าเฉลี่ย	1.351	2.613	0	200.8

ตาราง จ.2 ค่าอัตราขยายและช่วงความถี่ที่มีค่าอัตราขยายเท่ากันจากการทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด ที่ความถี่ในการสุ่มสัญญาณ 500 Hz โดยทำการเปลี่ยนแปลงขนาดของสัญญาณอินพุท พร้อมแสดงค่าเฉลี่ยที่นำไปใช้ในการอนุวัต

แรงดันอินพุท (V_{pp})	แรงดันเอาท์พุท (V_{pp})	อัตราขยาย	อัตราขยาย (dB)	ช่วงความถี่ที่มีอัตราขยายเท่ากัน	
				f_{min}	f_{max}
0.216	0.296	1.352	2.619	0	104.3
0.224	0.296	1.321	2.421	0	99.8
0.232	0.296	1.276	2.116	0	101.2
0.240	0.320	1.333	2.499	0	100.8
	ค่าเฉลี่ย	1.321	2.414	0	101.5

หมายเหตุ: f_{min} คือความถี่ต่ำสุดที่มีอัตราขยายเท่ากัน

f_{max} คือความถี่สูงสุดที่มีอัตราขยายเท่ากัน

ตารางที่ จ.3 ค่าอัตราขยายและเฟสที่ได้จากการโปรแกรมทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ที่ความถี่ในการสุ่มสัญญาณเท่ากับ 1 kHz และใช้สัญญาณอินพุทที่มีขนาด 0.216 V_{pp}

ความถี่ (Hz)	แรงดันอินพุท (V)	แรงดันเอาท์พุท (V)	อัตราขยาย	อัตราขยาย (dB)
1.449	0.216	0.292	1.352	2.619
4.95	0.216	0.292	1.352	2.619
6.173	0.216	0.292	1.352	2.619
13.51	0.216	0.292	1.352	2.619
25	0.216	0.292	1.352	2.619
29.07	0.216	0.292	1.352	2.619
31.25	0.216	0.292	1.352	2.619
33.56	0.216	0.292	1.352	2.619
65.79	0.216	0.292	1.352	2.619
100	0.216	0.292	1.352	2.619
131.6	0.216	0.292	1.352	2.619
169.5	0.216	0.296	1.370	2.737
200	0.216	0.296	1.370	2.737
		ค่าเฉลี่ย	1.35	2.64

ตารางที่ จ.4 ค่าอัตราขยายและเฟสที่ได้จากการโปรแกรมทดสอบการรับและส่งสัญญาณด้วยคอนโทรลเลอร์บอร์ด BL1120 (Little Giant) ที่ความถี่ในการสุ่มสัญญาณเท่ากับ 500 Hz และใช้สัญญาณอินพุทที่มีขนาด 0.216 V_{pp}

ความถี่ (Hz)	แรงดันอินพุท (V)	แรงดันเอาท์พุท (V)	อัตราขยาย	อัตราขยาย (dB)
1.52	0.216	0.288	1.333	2.499
4.95	0.216	0.288	1.333	2.499
6.22	0.216	0.288	1.333	2.499
14.3	0.216	0.288	1.333	2.499
25.25	0.216	0.288	1.333	2.499
29.00	0.216	0.288	1.333	2.499
33.2	0.216	0.288	1.333	2.499
40.25	0.216	0.288	1.333	2.499
65.79	0.216	0.288	1.333	2.499
101.23	0.216	0.288	1.333	2.499
		ค่าเฉลี่ย	1.33	2.50

ภาคผนวก ฉ

โปรแกรมที่ใช้ทดสอบโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง

โปรแกรมที่ใช้ทดสอบโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง

(โปรแกรม test.m ใน Matlab)

```
**************************************************************************
*   โปรแกรมทดสอบการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง
*   จัดทำโดย : นางสาวศศิธร อนุรักษ์ปัญญา B3904602
*   อาจารย์ที่ปรึกษา:  รศ.ดร. สราวุฒิ สุจิตจร
*                       อาจารย์สมศักดิ์ วาณิชอนันต์ชัย
*   ลิขสิทธิ์เป็นของมหาวิทยาลัยเทคโนโลยีสุรนารี , พ.ศ. 2543
**************************************************************************
disp('Select order of the compensator 1 or 2');
n=input('Enter order:');

if(n==1)
  disp('        s+a');
  disp('G(s)= -----');
  disp('        s+b');

  h=input('Enter h in second unit: ');
  a=input ('Enter for a:');
  b=input ('Enter for b:');

  disp('              1+a11*z^(-1)');
  disp('G(z)= c1 ----------------------');
  disp('              1+b11*z^(-1)');

 disp('y(n) = x(n) + a11*x(n-1) - b11*y(n-1)');

  m=2/h;
  alpha=m.*tan(a*h/2);
  beta=m.*tan(b*h/2);
  e=alpha.*h;
  f=beta.*h;
  a0=e+2;
  a1=(e-2);
  b0=f+2;
  b1=(f-2);

  c1=a0./b0;
  a11=a1./a0;
  b11=b1./b0;

  zero=[a0 a1]; pole=[b0 b1];
  [H,F]=freqz(zero,pole,2500,(1/h));

  nums=[1 a]; dens=[1 b];
```

```
[mag pha w]= bode(nums,dens);
db=[3.522 3.522 3.522 3.522 3.414 3.414 3.414 3.305 3.194 3.194 2.737 2.254
1.610 1.200 0.915 0.915 0.621 0.621 0.545 0.316 0.159 0.159 0.000 0.000];

f=[0.0980 0.2315 0.3165 0.4167 0.5319 0.6024 0.7463 0.8333 0.9346 1.1630
2.174 2.89 4.032 5.051 6.024 7.143 8 9.174 10 21.74 31.25 39.68 70.42 100];

subplot(2,1,1); semilogx(w/(2*pi),20*log10(mag),'bo',F,20*log10(H),'r-',f,db,'m.');
                                                        %H(s),H(z),H(con)
axis([1/10 1000 -0.5 6]);
grid on;

title('Frequency response test for the first order compensator');
xlabel('frequencies (Hz)'); ylabel('magnitude (dB)');
legend('H(s)','H(z)','H(con)',0);

phase=-[1.058 1.667 2.279 2.700 3.064 3.904 4.299 4.800 5.383 6.699 9.392
11.444 11.612 10.910 10.843 10.286 8.640 8.257 7.920 3.913 3.375 2.857
1.268 1.080];

subplot(2,1,2); semilogx(w/(2*pi),pha,'bo',F,angle(H)*180/pi,'r-',f,phase,'m.');
                                                        %phase of H(s),H(dsp)
axis([1/10 1000 -15 0]);

xlabel('frequencies (Hz)'); ylabel('phase (degree)');
legend('H(s)','H(z)','H(con)',0);
grid;
else
disp('          (s^2+cs+d)');
disp('G(s)= ----------');
disp('          (s^2+as+b)');
h=input('Enter h in second unit:');
a=input('Enter for a:');
b=input('Enter for b:');
c=input('Enter for c:');
d=input('Enter for d:');

disp('              1+a11*z^(-1)+a21*z^(-2)');
disp('G(z)= c2 ------------------------------------');
disp('              1+b11*z^(-1)+b21*z^(-2)');

disp('y(n) = x(n) + a1*x(n-1) + a2*x(n-2) - b1*y(n-1) - b2*y(n-2)');

bn=sqrt(b);
dn=sqrt(d);
wp=h*bn/2;
wz=h*dn/2;
m=2/h;
```

```
wp=m.*tan(wp);
wz=m.*tan(wz);

r=c./dn*wz;
w=a./bn*wp;
a_0=2*r*h;
b_0=2*w*h;
a0=4+a_0;
a1=-8;
a2=(4-a_0);
b0=4+b_0;
b1=-8;
b2=(4-b_0);

c2=a0./b0;
a11=a1./a0;
a21=a2./a0;
b11=b1./b0;
b21=b2./b0;

zeros=[a0 a1 a2]; poles=[b0 b1 b2];
[H,F]=freqz(zeros,poles,15000,(1/h));

num=[1 c d]; den=[1 a b];
[mag pha w]= bode(num,den);

db=[0.962 0.962 0.962 0.962 0.962 0.962 0.889 0.816 0.742 0.668 0.592 0.592
0.517 0.363 0.363 0.206 0.127 0.127 0.047 0.047 0.047 0.047 0.047];

f=[0.4854 0.6579 0.8065 0.9009 1.19 1.736 1.953 2.941 4.31 5.263 6.173 7.463
8.475 9.346 10.42 20 31.65 39.68 54.95 60.21 70.85 82.31 97.09];

subplot(2,1,1); semilogx(w/(2*pi),20*log10(mag),'bo',F,20*log10(H),'r-',f,db,'m.');
                                                    %H(s),H(z),H(con)
axis([1/100 1000 -1 2]);
grid on;

title('Frequency response test for the second order compensator');
xlabel('frequencies (Hz)'); ylabel('magnitude (dB)');
legend('H(s)','H(z)','H(con)',0);

phase=-[0.280 0.474 0.581 0.649 0.857 0.937 1.406 1.694 2.483 2.653 2.667
2.955 2.929 2.927 2.813 2.232 1.595 1.143 1.187 0.867 0.765 0.593 0.350];

subplot(2,1,2); semilogx(w/(2*pi),pha,'bo',F,angle(H)*180/pi,'r-',f,phase,'m.');
                                                    %phase of H(s),H(dsp)
xlabel('frequencies (Hz)'); ylabel('phase (degree)');
legend('H(s)','H(z)','H(con)',0);
```

```
axis([1/100 1000 -3 0]);
grid;
end
```

ภาคผนวก ช

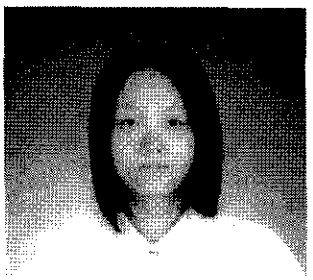
ค่าอัตราขยายและเฟสที่ได้จากการทดสอบโปรแกรมการอนุวัตตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่งและสอง

ตารางที่ ช.1 ค่าอัตราขยายและเฟสที่ได้จากการทดสอบตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับหนึ่ง ที่ความถี่การสุ่มสัญญาณเท่ากับ 1 kHz

ความถี่ (Hz)	G_{disp}	G_{act}	G_{act} (dB)	ชนิดของเฟส	เฟส (degree)
0.0980	1.493	1.5000	3.522	lag	1.058
0.2315	1.493	1.5000	3.522	lag	1.667
0.3165	1.493	1.5000	3.522	lag	2.279
0.4167	1.493	1.5000	3.522	lag	2.700
0.5319	1.474	1.4815	3.414	lag	3.064
0.6024	1.474	1.4815	3.414	lag	3.904
0.7463	1.474	1.4815	3.414	lag	4.299
0.8333	1.456	1.4630	3.305	lag	4.800
0.9346	1.437	1.4444	3.194	lag	5.383
1.1630	1.437	1.4444	3.194	lag	6.699
2.174	1.364	1.3704	2.737	lag	9.392
2.890	1.290	1.2963	2.254	lag	11.444
4.032	1.198	1.2037	1.610	lag	11.612
5.051	1.142	1.1481	1.200	lag	10.910
6.024	1.106	1.1111	0.915	lag	10.843
7.143	1.106	1.1111	0.915	lag	10.286
8.000	1.069	1.0741	0.621	lag	8.640
9.174	1.069	1.0741	0.621	lag	8.257
10.000	1.060	1.0648	0.545	lag	7.920
21.74	1.032	1.0370	0.316	lag	3.913
31.25	1.013	1.0185	0.159	lag	3.375
39.68	1.013	1.0185	0.159	lag	2.857
70.42	0.995	1.0000	0.000	lag	1.268
100.00	0.995	1.0000	0.000	lag	1.080

ตารางที่ ช.2 ค่าอัตราขยายและเฟสที่ได้จากการทดสอบตัวชดเชยที่มีทรานสเฟอร์ฟังก์ชันอันดับสอง ที่ความถี่การสุ่มสัญญาณเท่ากับ 500 Hz

ความถี่ (Hz)	G_{disp}	G_{act}	G_{act} (dB)	ชนิดของเฟส	เฟส (degree)
0.4854	1.0657	1.0715	0.600	lag	0.750
0.6579	1.0596	1.0654	0.550	lag	1.000
0.8065	1.0560	1.0617	0.520	lag	1.200
0.9009	1.0560	1.0617	0.520	lag	1.300
1.1900	1.0535	1.0593	0.500	lag	1.700
1.736	1.0415	1.0471	0.400	lag	2.020
1.953	1.0451	1.0508	0.430	lag	2.000
2.941	1.0248	1.0304	0.260	lag	1.900
4.310	1.0119	1.0174	0.150	lag	1.500
5.263	1.0119	1.0174	0.150	lag	1.400
6.173	1.0061	1.0116	0.100	lag	1.200
7.463	1.0061	1.0116	0.100	lag	1.000
8.475	1.0003	1.0058	0.050	lag	1.000
9.346	1.0003	1.0058	0.050	lag	0.800
10.42	0.9969	1.0023	0.020	lag	0.900
20.00	0.9946	1.0000	0.000	lag	0.500
31.65	0.9946	1.0000	0.000	lag	0.200
39.68	0.9946	1.0000	0.000	lag	0.200
54.95	0.9946	1.0000	0.000	lag	0.050
60.21	0.9946	1.0000	0.000	lag	0.100
70.85	0.9946	1.0000	0.000	lag	0.060
82.31	0.9946	1.0000	0.000	lag	0.050
97.09	0.9946	1.0000	0.000	lag	0.050



นางสาวศศิธร อนุรักษ์ปัญญา
เกิดวันที่ 15 กันยายน พ.ศ. 2521
ภูมิลำเนาอยู่ที่จังหวัดนนทบุรี

ประวัติการศึกษา

- สำเร็จการศึกษาชั้นประถมศึกษาจากโรงเรียนวัดมกุฎกษัตริยาราม กรุงเทพมหานคร
- สำเร็จการศึกษาชั้นมัธยมศึกษาตอนต้นจากโรงเรียนเบญจมราชาลัย กรุงเทพมหานคร
- สำเร็จการศึกษาชั้นมัธยมศึกษาตอนปลายจากศูนย์การศึกษานอกโรงเรียน โรงเรียน กรุงเทพมหานคร 2
- ปัจจุบัน เป็นนักศึกษาชั้นปีที่ 4 สาขาวิศวกรรมโทรคมนาคม สำนักวิศวกรรมโทรคมนาคม มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา