การแก้ความผิดเพี้ยนทางเวลาของสัญญาณลูมิแนนซ์ต่อโครมิแนนซ์ด้วยวิธีดีเลย์สโลปเชิงเส้น

โดย

นายถนิม ทิพย์ผ่อง 39012010

นายไพโรจน์ ปุรณวัฒนกุลชัย 39012019

นายรังสรรค์ ศรีศันสนีย 39012023

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรอุตสาหกรรมศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2540

EQUALIZATION OF THE LUMINANCE TO CHROMINANCE TIME DISTORTION BY LINEAR DELAY SLOPE

MR. TANIM TIPPHONG
MR. PAIROJ PURANAWATTANAKUNCHAI
MR. RANGSAN SRISANSANEE

PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE BACHELOR'S DEGREE
DEPARTMENT OF INDUSTRIAL TECHNOLOGY
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
1997

หัวข้อปริญญานิพนธ์	การแก้ความผิดเพี้ยนทางเวลาของสัญญาณลูมิแนนซ์ต่อโครมิแนนซ์ด้วยวิธีดีเลย์สโลปเชิงเส้น		
โดย	นายถนิม	ทิพย์ผ่อง	39012010
	นายไพโรจน์	ปุรณวัฒนกุลชัย	39012019
	นายรังสรรค์	ศรีศันสนีย	39012023
ภาควิชา	เทคนิคอุตสาหกรรม		
อาจารย์ที่ปรึกษา	รศ.ดร.กนก	เจนจิระพงศ์เวช	
	อาจารย์จักรี	ทีฆภาคย์วิศิษฎ์	

คณะวิศวกรรมศาสตร์สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้นับปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรอุตสาหกรรมศาสตรบัณฑิต

คณะกรรมการสอบปริญญานิพนธ์

--อาจารย์ที่ปรึกษา
(รศ.ดร.กนก เจนจิระพงศ์เวช)

--อาจารย์ที่ปรึกษา
(อาจารย์จักรี ทีฆภาคย์วิศิษฎ์)

--กรรมการ
(--)

--กรรมการ
(--)

PROJECT REPORT EQUALIZATION OF THE LUMINANCE TO CHROMINANCE TIME DISTORTION BY LINEAR DELAY SLOPE

BY MR. TANIM TIPPHONG
MR. PAIROJ PURANAWATTANAKULCHAI
MR. RANGSAN SRISANSANEE

DEPARTMENT OF INDUSTRIAL TECHNOLOGY

ADVISOR ASSOC. PROF. DR. KANOK JANCHITRAPONGVEJ
MR. CHAKREE TEEKAPAKVISIT

ACCEPTED BY THE FACULTY OF ENGINEERING , KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE BACHELOR'S DEGREE

PROJECT REPORT COMMITTEE

--CHAIRMAN
( )

-- COMMITTEE
( )

-- COMMITTEE
( )

-- COMMITTEE
( )

การแก้ความผิดเพี้ยนทางเวลาของสัญญาณลูมิแนนซ์ต่อโครมิแนนซ์โดยวิธีดีเลย์สโลปเชิงเส้น

Equalization of the Luminance to Chrominance Time Distortion by linear slope delay

โดย	นายถนิม	ทิพย์ผ่อง	39012010
	นายไพโรจน์	ปุรณวัฒนกุลชัย	39012019
	นายรังสรรค์	ศรีศันสนีย	39012023

อาจารย์ที่ปรึกษา	รศ.ดร.กนก	เจนจิระพงศ์เวช
	อาจารย์จักรี	ทีฆภาคย์วิศิษฎ์

บทคัดย่อ

เนื่องจากในการส่งสัญญาณภาพนั้น มักมีความผิดเพี้ยนทางด้านเวลาของสัญญาณสี ฉะนั้นจึงมีความจำเป็นที่จะต้องแก้ไขความผิดเพี้ยนทางเวลาของสัญญาณลูมิแนนซ์ต่อโครมิแนนซ์ ดังกล่าว ในโครงงานนี้เสนอการแก้ไขความผิดเพี้ยนโดยการประมาณดีเลย์สโลปเชิงเส้นและใช้วิธีการออปติไมซ์เซชั่น นอกจากนี้ยังใช้สัญญาณมอดูเลต 20T ซายน์กำลังสองพัลซ์ในการวิเคราะห์ผลของการตอบสนองทางของเน็ทเวิร์กอีกด้วย

จากผลการทดลองได้ผลลัพธ์สอดคล้องกับทฤษฎีการประมาณดีเลย์สโลปเชิงเส้นที่ใช้ในการแก้ไข

Abstract

In the picture signal transmission system, it is necessary to correct the time distortion of the luminance to chrominance signal. This is project equalize the above by linear slope delay approximation and used optimization method. In addition,the modulated 20T sine-squared pulse is used to investigate the effeft of the network response.

It is shown that the experimental results are quite agree with the theoretical result.

กิตติกรรมประกาศ

ขอขอบคุณ รศ.ดร. กนก เจนจิระพงศ์เวช และ อาจารย์จักรี ทีฆภาควิศิษฏ์ ที่ให้คำแนะนำในการออกแบบวงจรพร้อมทั้งให้ข้อมูลที่เป็นประโยชน์ในการทำโครงงาน โปรแกรมคำนวณ และเครื่องมือที่ใช้ในการทดสอบวงจรที่ได้ออกแบบให้เป็นผลสำเร็จได้อย่างดียิ่ง

คณะผู้จัดทำโครงงาน

สารบัญ

หน้า

บทนำ

ในการส่งสัญญาณโทรทัศน์นั้น สัญญาณภาพรวมจะถูกส่งผ่านวงจรต่าง ๆ รวมทั้งสายส่งก่อนที่จะถูกส่งออกอากาศทำให้สัญญาณโทรทัศน์มีอัตราการขยายทางขนาดของสัญญาณลูมิแนนซ์ไม่เท่ากับของสัญญาณโครมิแนนซ์ ขณะเดียวกันจะทำให้สัญญาณลูมิแนนซ์ถูกหน่วงเวลาต่างกันกับสัญญาณโครมิแนนซ์ในการที่จะทดสอบระบบการส่งทำได้หลายวิธี ในที่นี้จะใช้สัญญาณไซน์กำลังสองพัลซ์ เพราะสามารถทดสอบผลตอบสนองทางขนาด และการหน่วงสัญญาณภาพรวม โดยแก้ความผิดเพี้ยนเชิงเส้นทางเวลาของสัญญาณลูมิแนนซ์ต่อโครมิแนนซ์

วัตถุประสงค์

เพื่อทำการเสนอวิธีการแก้ความผิดเพี้ยนทางเวลาของสัญญาณลูมิแนนซ์ต่อโครมิแนนซ์ด้วยการประมาณค่าดีเลย์สโลปเชิงเส้น โดยได้นำเอาหลักการออปติไมซ์เซชั่นมาช่วยในการคำนวณและออกแบบ ซึ่งจะแก้ไขความผิดพลาดของการคำนวณได้

บทที่ 1
ทฤษฎีที่ใช้ในการออกแบบอ้างอิง

1.1 ทฤษฎีและการหาสูตรสำเร็จของความผิดเพี้ยนของอัตราการขยายและดีเลย์

การหาความผิดเพี้ยนของสัญญาณสี (Chrominance signal) ทั้งอัตราการขยาย (gain) ซึ่งใช้สัญลักษณ์ A และไทม์ดีเลย์ (τ) โดยผลจากการคำนวณหาความผิดเพี้ยนทั้ง A และ τ จะสัมพันธ์กับค่ายอดทั้งสองจากฐานของพัลส์ก็สามารถที่จะหาความผิดเพี้ยนของพัลส์ว่าในระบบทดสอบ (System Under Test) จะมีผลต่อสัญญาณสีทางด้านไหน เช่น ระบบทดสอบอาจมีผลทางด้านความผิดเพี้ยนของอัตราการขยาย (gain distortion) อย่างเดียวหรือความผิดเพี้ยนทางด้านไทม์ดีเลย์ (Time delay Distortion) เพียงอย่างเดียวหรือมีผลของความผิดเพี้ยนทั้งอตราขยาย ไทม์ ดีเลย์ ซึ่งจะแสดงผลการคำนวณค่าความผิดเพี้ยนทั้งอัตราการขยาย และ ไทม์ดีเลย์โดยละเอียด และการแสดงค่าโดยประมาณด้วยค่าที่กำหนดให้ของค่ายอดทั้งสองที่ฐานของพัลส์และอัตราการขยายที่ไม่เท่ากัน (A) และค่าดีเลย์อินอิควอลิตี้ (τ) ที่สัมพันธ์กัน

วิธีการคำนวณของอัตราขยาย และดีเลย์อินอิควอลิตี้ กำหนดสัญลักษณ์การคำนวณดังต่อไปนี้

f(t) = เบสไลน์ฟังก์ชั่นของโคไซน์กำลังสองพัลส์

U(t) = เอ็นวิโลป ทางด้านบนของโคไซน์กำลังสองพัลส์

L(t) = เอ็นวิโลป ด้านล่างของโคไซน์กำลังสองพัลส์

Y1 & Y2 = ขนาดของเบสไลน์ด้านล่างของโคไซน์กำลังสองพัลส์

A = อัตราการขยายของโคไซน์กำลังสองพัลส์ที่ได้รวบรวมกับเบสแบนด์ที่สัมพันธ์กัน

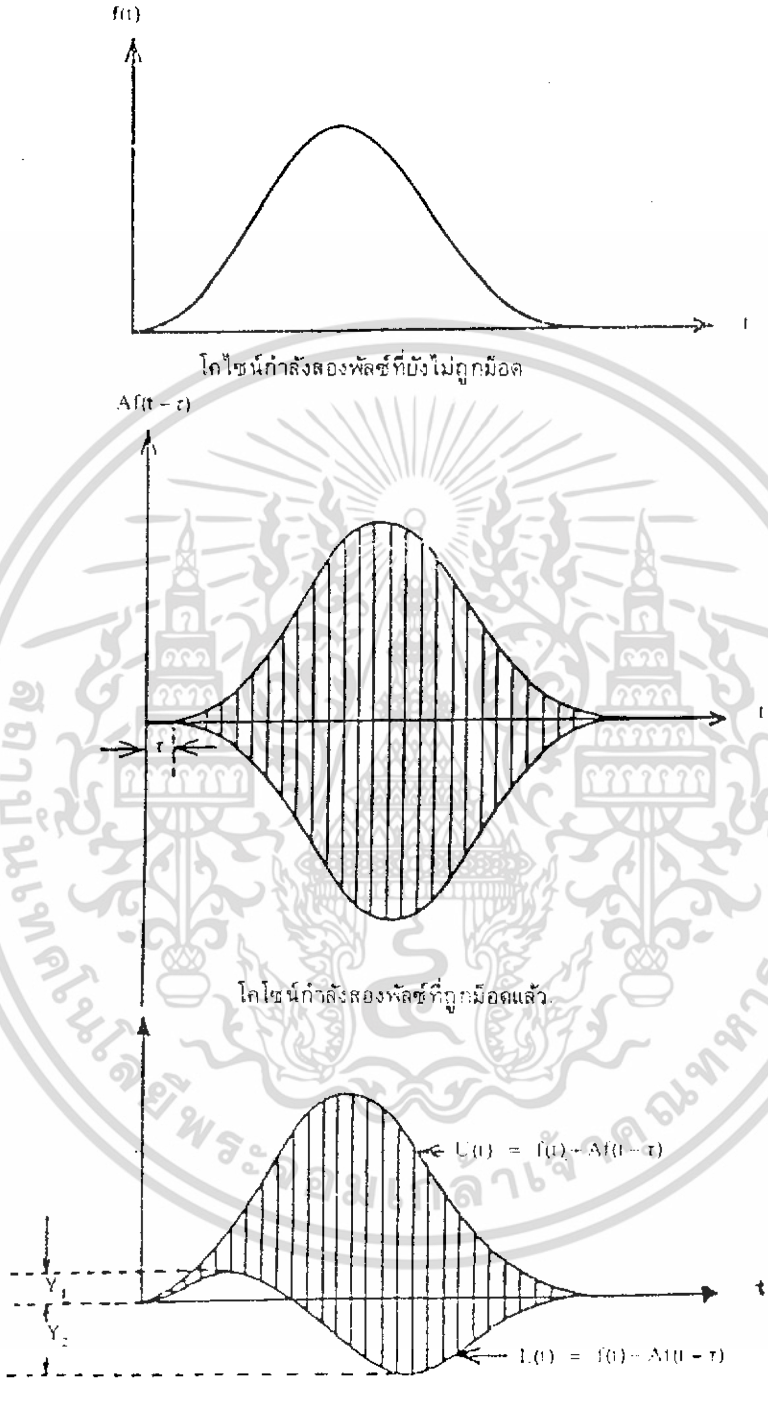
τ = ดีเลย์อินอิควอลิตี้ส่วนที่ถูกรวบช้าไป หรือนำหน้าสัญญาณที่ยังไม่มีการมอด

To = ระยะเวลาของขนาดของพัลส์ ในกรณีของ 20T pulse คือ 2 μs และ 12.5T คือ 1.25 ในระบบ PAL System ,20T ในระบบ NTSC, T= 0.125 μs

การคำนวณให้พิจารณารูปที่ 1 จะได้เอ็นวิโลปด้านบนและด้านล่าง

$$U(t) = f(t) + Af(t-\tau) \quad ....(1)$$

$$L(t) = f(t) - Af(t-\tau) \quad ....(2)$$



รูปที่ 1 สัญญาณรวมโคไซน์กำลังสองพัลส์

และค่าเบสแบน (base band) ก็คือ

$$f(t) = \frac{1}{2}\cos^2\frac{\pi}{2T_0} \qquad 0 < t < T_0 \qquad \text{....(3)}$$
$$= 0 \qquad t > T_0$$

การหาค่า เอ็นวิโลปด้านบนและด้านล่างได้ด้วยการแทนค่าสมการ (3) ลงในสมการ (1) และ (2) ตามลำดับ จากเอ็นวิโลปด้านล่างจะทำให้เราทราบค่า Y1 และ Y2 ส่วนเอ็นวิโลปด้านล่างจะทำให้เราทราบค่า Y_{max} ตามลำดับ แต่จากการวัดค่าที่แน่นอนของค่ายอดฐานของพัลส์ที่สูงสุด Y_{max} จากผลการคำนวณนี้จะทำให้เราสามารถหาสูตรความสำเร็จของความผิดเพี้ยนของการขยายที่แน่นอนแสดงได้ดังนี้

$$A = \frac{1-(y_1+y_2+y_1y_2)}{1+(y_1+y_2-y_1y_2)} \qquad \text{....(4)}$$

โดยกำหนดให้

$$y_1 = \frac{Y_1}{Y_{max}}, \quad y_2 = \frac{Y_2}{Y_{max}}$$

จากเอ็นวิโลปด้านล่างที่ฐานล่างที่ฐานของพัลส์เราสามารถจะหาค่าดีเลย์ อินอิควอลิตี้ (Delay Inequality) จากค่ายอดทั้งบวกและค่าลบของ Y1 และ Y2 ตามลำดับ ซึ่งทำให้ได้สูตรสำเร็จในการหาดีเลย์ อินอิควอลิตี้ ดังนี้

$$\tau = \frac{T_0}{\pi}\cos^{-1}\left\{1+\frac{8y_1y_2}{\{1-(y_1+y_2+y_1y_2)\}.\{1+(y_1+y_2-y_1y_2)\}}\right\} \qquad \text{....(5)}$$

ค่ายอดทั้งค่าบวกและค่าลบของเอ็นวิโลปด้านล่างจากฐานของพัลส์ Y1 และ Y2 ทำการ normalized ด้วยค่ายอดสูงสุดของเอ็นวิโลปด้านบน นั่นคือ

$$y_1 = \frac{Y_1}{Y_{max}} \qquad \text{และ} \qquad y_2 = \frac{Y_2}{Y_{max}}$$

สมการที่ (4) และ สมการ (5) เป็นสูตรสำเร็จที่ใช้ในการวัดความผิดเพี้ยนของอัตราการขยายและดีเลย์ อินอิควอลิตี้ ตามลำดับ

สาเหตุของการผิดเพี้ยนโดยความแตกต่างของอัตราการขยายอย่างเดียว (Distortion Cause by Gain Difference Only)

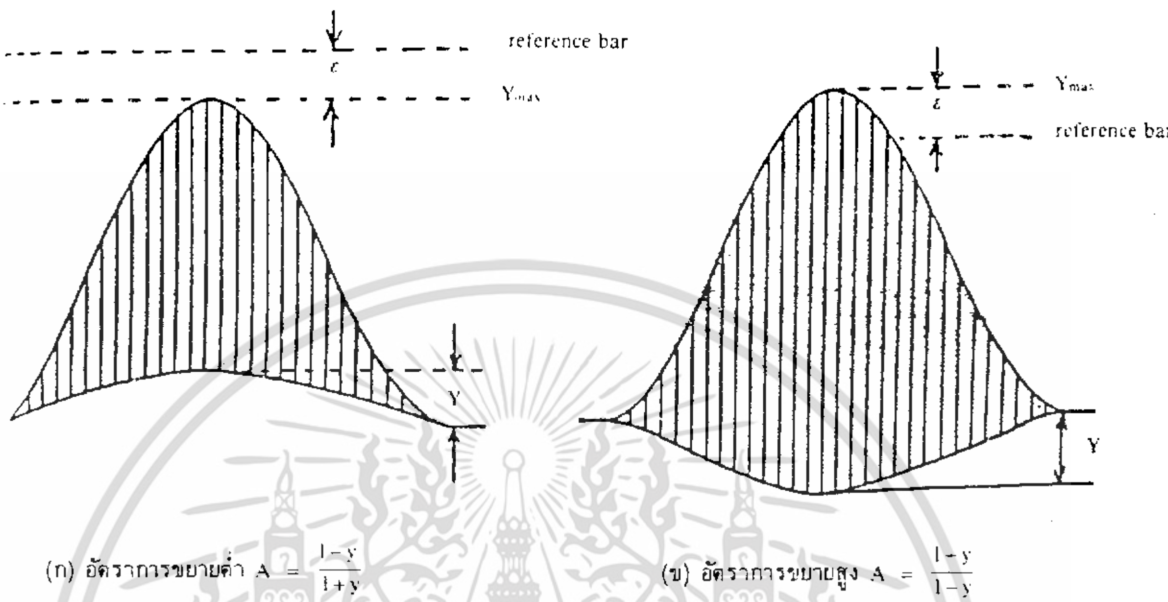
ในกรณีนี้แสดงว่าค่ายอด Y1 และ Y2 ค่าใดค่าหนึ่งเป็น 0 ดังนั้นจะไม่มีความผิดเพี้ยนทางดีเลย์เกิดขึ้นก็คือ $\tau = 0$ เพราะฉะนั้นความผิดเพี้ยนที่เกิดขึ้นจึงมีแต่ความผิดเพี้ยนของอัตราอย่างเดียวซึ่งสามารถแสดงให้เห็นจริงได้ดังนี้

$$\tau = \frac{T_0}{\pi} \cos^{-1}(1) = 0$$

และสมการ (4) จะกลายเป็น

$$A = \frac{1 - y_{1,2}}{1 + y_{1,2}} = \frac{1 - \frac{Y}{Y_{max}}}{1 + \frac{Y}{Y_{max}}} \qquad \text{....(6)}$$

ซึ่ง (y_1, y_2) หรือ Y อาจมีค่าเป็นบวกหรือลบ เนื่องมาจากขนาดของความผิดเพี้ยนที่สัมพันธ์กันดังรูป (2)



รูปที่ 2 แสดงความแตกต่างของความผิดเพี้ยนของอัตราอย่างเดียว

ความผิดเพี้ยนของอัตราการขยาย (ε) พิจารณาได้จากสมการ (6) ดังนี้

$$\varepsilon = 1 - A = \frac{2Y/Y_{max}}{1+Y/Y_{max}} \qquad ....(7)$$

ถ้า $1 >> Y/Y_{max}$ ดังนั้น $\varepsilon = 2Y/Y_{max}$

ซึ่ง $2Y/Y_{max}$ จะถูกเรียกว่า Relative Chrominance Level (RCL) และบางครั้งถูกเรียกว่า Relative Amplitude of the Color Sub-Carrier

ในกรณีที่ $Y1 = Y2$ ซึ่งฐานของพัลส์มีรูปร่างเป็นคลื่นไซน์ที่มีค่ายอดเท่ากันแต่อยู่ตรงข้ามแสดงว่าไม่มีความผิดเพี้ยนทางอัตราการขยาย ความผิดเพี้ยนที่เกิดขึ้นที่มีแต่ความผิดเพี้ยนทางด้านดีเลย์เพียงอย่างเดียวดังได้แสดงดังต่อไปนี้จากสมการ (4) แทนค่า $Y1 = - Y2$ จะได้ดังนี้

$$A = \frac{1+(Y/Y_{max})^2}{1+(Y/Y_{max})^2} = 1 = 0 \text{ dB}$$

จากสมการ (5) จะกลายเป็น

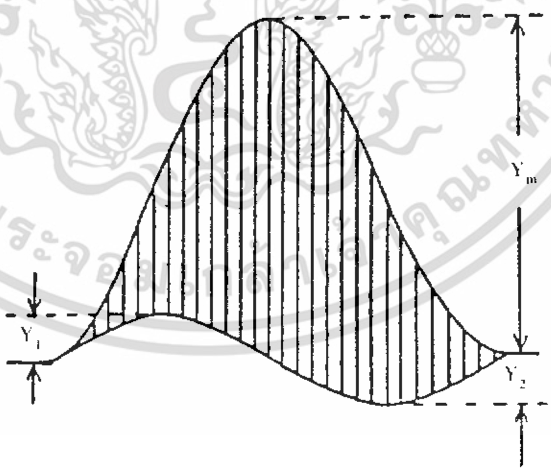
$$\tau = \frac{T_0}{\pi} \cos^{-1}\left[1-\frac{8(Y/Y_{max})^2}{\left\{1+(Y/Y_{max})^2\right\}^2}\right] \quad ....(8)$$

โดยอาศัยสูตรทางตรีโกณมิติ สมการ (8) จะกลายเป็น

$$\tau = \frac{T_0}{\pi} \tan^{-1}\left[1-\frac{8(Y/Y_{max})^2}{\left\{1+(Y/Y_{max})^2\right\}^2}\right] \quad ....(9)$$

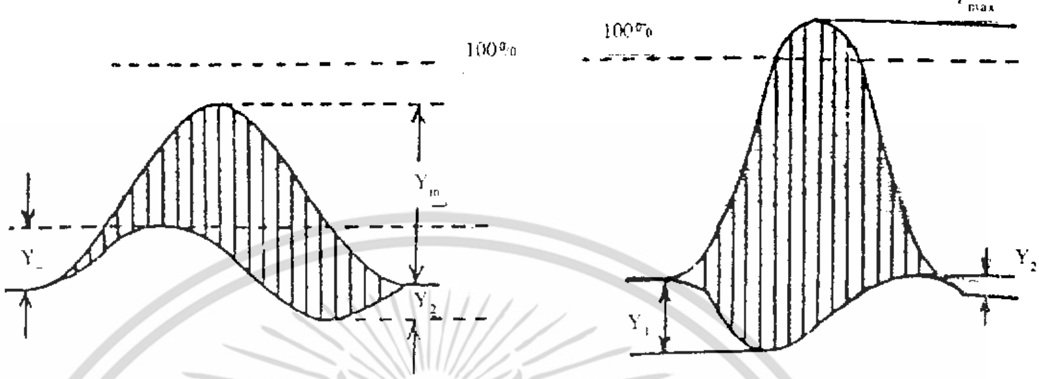
$$= \frac{4T_0}{\pi} \tan^{-1}\frac{Y}{Y_{max}}$$

จะเห็นได้ว่าเมื่อ Y1 = - Y2 จะได้ A = 0 dB ส่วนการผิดเพี้ยนทางด้านดีเลย์ดังแสดงในรูปที่ 3



รูปที่ 3 แแสดงการผิดเพี้ยนของดีเลย์เพียงอย่างเดียว (กรณี Y1 = - Y2)

ในกรณีที่เกิดความผิดเพี้ยนทั้งสองแบบ ค่ายอดทั้งบวกและลบ มีค่าแตกต่างกัน ดังแสดงในรูปที่ 4



รูปที่ 4 แสดงความผิดเพี้ยนทางอัตราการขยายและดีเลย์ พร้อมกัน

เมื่อความผิดเพี้ยนมีค่าน้อย ดังนั้นผลคูณของ Y1 Y2 สามารถตัดทิ้งได้ดังสมการ (4) และ (5) อาจเขียนได้ใหม่ดังนี้

$$A = \frac{1-(Y1/Y2)}{1+(Y1/Y2)} \quad ....(10)$$

$$\tau = \frac{T_0}{\pi}\cos^{-1}\left[1+\frac{8Y1Y2}{1-(Y1+Y2)^2}\right]$$

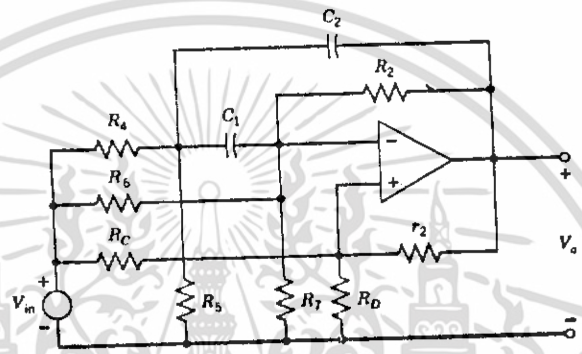
และถ้า $(y_1+y_2)^2 << 1$ ความผิดเพี้ยนทางด้านดีเลย์อาจเขียนได้ใหม่ คือ

$$\tau = \frac{4T_0}{\pi}\sqrt{-Y_1Y_2} \quad ....(11)$$

ความผิดเพี้ยนทางอัตราการขยายคำนวณด้วยสมการ (4) กับ (10) จะได้ค่าใกล้เคียงกันมาก และความผิดเพี้ยนทางด้านดีเลย์ เมื่อคำนวณด้วยสมการ (5) และ (11) ค่าที่ได้แตกต่างกันเล็กน้อย เมื่อผลคูณ Y1 Y2 มีค่าน้อยมาก

1.2 Single- Amplifier General Biquad

ในที่นี้เราจะพิจารณา Negative feedback RC Amplifier Network ซึ่งสมารถทำรูปแบบทั่ว ๆ ไปของ Biquadratic transfer function ได้ network นั้นแสดงได้ดังรูปที่ 5



รูปที่ 5

ในการวิเคราะห์เราจะพิจารณา equivalent network ในรูปที่ 6 เมื่อค่า K_1 ,K_2 และ K_3 ซึ่งเกี่ยวพันกันในรูป เป็นการปรับ divider ได้คือ

$$K_1 = \frac{R_5}{R_4 + R_5}, \qquad R_1 = \frac{R_4 R_5}{R_4 + R_5}$$

$$K_2 = \frac{R_D}{R_C + R_D}, \qquad r_2 = \frac{R_C R_D}{R_C + R_D}$$

$$K_3 = \frac{R_7}{R_6 + R_7}, \qquad R_3 = \frac{R_6 R_7}{R_6 + R_7}$$

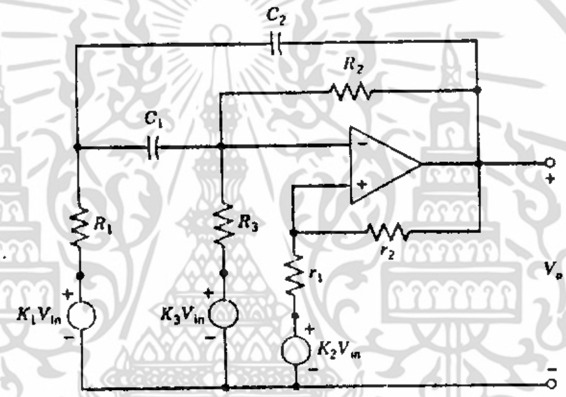
โดยหลักการของ SUPERPOSITION นั้น V_O จะเท่ากับผลรวมของ K_1V_{in} ,K_2V_{in} และ K_3V_{in} ซึ่งจะได้ดังนี้

$$V_o(s) = H_1(s)K_1V_{in}(s) + H_2(s)K_2V_{in}(s) + H_3(s)K_3V_{in}(s)$$

$$\frac{V_o(s)}{V_{in}(s)} = H(s) = K_1H_1(s) + K_2H_2(s) + K_3H_3(s)$$

เขียน H ให้อยู่ในฟอร์ม

$$H(s) = \frac{V_o(s)}{V_{in}(s)} = K\frac{b_2s^2+b_1s+b_0}{s^2+a_1s+a_0}$$



รูปที่ 6

เมื่อพบว่า K = 1 สัมประสิทธิ์จะเป็น

$$b_2 = K_2$$

$$b_1 = \frac{K_2}{C_2}\left(\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_3}\right)+\frac{K_2}{C_1}\left(\frac{1}{R_2}+\frac{1}{R_3}\right)-\left(1+\frac{r_1}{r_2}\right)\left[\frac{K_1}{R_1C_2}+\frac{K_3}{R_3}\left(\frac{1}{C_1}+\frac{1}{C_2}\right)\right]$$

$$b_0 = \frac{1}{C_1C_2}\left[\frac{K_2}{R_1}\left(\frac{1}{R_2}+\frac{1}{R_3}\right)-\frac{K_3}{R_1R_3}\left(1+\frac{r_1}{r_2}\right)\right]$$

$$a_1 = \frac{C_1 + C_2}{C_1 C_2}\left(\frac{1}{R_2} - \frac{r_1}{R_3 r_2}\right) - \frac{r_1}{R_1 r_2 C_2}$$

$$a_0 = \frac{1}{R_1 C_1 C_2}\left(\frac{1}{R_2} - \frac{r_1}{R_3 r_2}\right)$$

ใน 5 สมการข้างต้นมี 9 ตัวแปรที่ไม่รู้คือ $C_1, C_2, R_1, R_2, R_3, K_1, K_2, K_3$, และ $\frac{r_1}{r_2}$ ถึงแม้ว่า 4 ตัวแปรที่คงที่อาจจะได้ผลที่หาไม่ได้ฉะนั้นจะกำหนดค่าตัวเก็บประจุ โดยปกติจะให้เท่ากัน ดังนั้นจากสมการข้างต้นจะได้

$$R_1 = \frac{2r_1}{r_2 C_2\left[-a_1 + \sqrt{a_1^2 + 4a_0(1 + C_1 / C_2) r_1 / r_2}\right]} \quad ....(12.1)$$

$$K_1 = \frac{b_2 + b_0(1 + C_1 / C_2) R_1^2 C_2^2 - b_1 R_1 C_2}{1 + r_1 / r_2} \quad ....(12.2)$$

$$R_3 = \frac{(1 + r_1 / r_2)(b_2 - K_3)}{R_1 C_1 C_2 (b_0 - a_0 b_2)} \quad ....(12.3)$$

$$R_2 = \frac{R_3}{R_1 R_3 C_1 C_2 a_0 + r_1 / r_2} \quad ....(12.4)$$

สำหรับการออกแบบเราจะกำหนดให้

$$C_1 = C_2 = 1\text{F} \;, \quad 0 \le K_3 \le 1 \;, \quad \gamma = r_1/r_2$$

เพราะฉะนั้นจากสมการ (12) จะได้

$$R_1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}} \quad ....(13.1)$$

$$K_1 = \frac{b_2 + 2b_0R_1^2 - b_1R_1}{1+\gamma} \qquad ....(13.2)$$

$$R_3 = \frac{(1+\gamma)(b_2 - K_3)}{R_1a_0(b_0 / a_0 - b_2)} \qquad ....(13.3)$$

$$R_2 = \frac{R_3}{R_1R_3a_0 + \gamma} \qquad ....(13.4)$$

ถ้าพบว่า K_1 มากกว่า 1 เราต้อง scale สัมประสิทธิ์ b_2, b_1 และ b_0 ลงโดย

$$H(s) = Kk_1\frac{(b_2 / k_1)s^2 + (b_1 / k)s + b_0 / k_1}{s^2 + a_1 + a_0} \qquad ....(13.5)$$

1.3 Scaling

ในการ scaling นั้นเราจะทำการ scale หาค่าอุปกรณ์และ response ที่มีผลต่อ network โดยที่คุณสมบัติของ network ไม่มีการเปลี่ยนแปลง การ scale มีดังนี้

- magnitude scaling
- frequency sclaing
- time scaling
- delay scaling

ซึ่งสูตรสำเร็จในการ scaling คือ

- magnitude scaling

$L_{new} = k_mL_{old}$

$C_{new} = \frac{1}{k_m} C_{old}$

$R_{new} = k_m R_{old}$

เมื่อ k_m = magnitude scale factor

- frequency scaling

$$L_{new} = \frac{1}{k_f} L_{old}$$

$$C_{new} = \frac{1}{k_f} C_{old}$$

$$R_{new} = R_{old}$$

เมื่อ k_f = frequency scaling factor

- time scaling

$$L_{new} = k_t L_{old}$$

$$C_{new} = k_t C_{old}$$

$$R_{new} = R_{old}$$

เมื่อ k_t = time scaling factor

- delay scaling

$$L_{new} = k_D L_{old}$$

$$C_{new} = k_D C_{old}$$

$$R_{new} = R_{old}$$

$$D_0 = \frac{-\theta}{\omega}$$

เมื่อ k_D = delay scaling factor

Bairstow's Method

* เป็นวิธีการที่ใช้หาค่า root จาก Polynomail
* นำ Deflated Polynomail ที่เหลือมาทำตามวิธีการของ Bairstow ก็จะได้ Quadratic Factor ของ Polynomial ทั้งหมด
* Quadratic Factor คือ $x^2 + \bar{p}x + \bar{q} = 0$

ซึ่งสามารถนำไปหา Root จะได้เป็น Complex Conjugate Root ซึ่งจะมีค่า Real เหมือนกัน

Imagine จะเป็น Conjugate ซึ่งกันและกัน

การหา Root ------> $x^2 + \bar{p}x + \bar{q} = 0$

$$x = \frac{-\bar{p} \pm \sqrt{\bar{p}^2 - 4\bar{q}}}{2}$$

*ตัวอย่าง Polynomail Order N คือ

$$y = a_0 + a_1 x + a_2 x^2 + \ldots + a_N x^N$$

หรือ

$$y = \left(x^2 + px + q\right)G(x) + R(x)$$

จะพบว่า $b_0 = f(p,q)$ และ $b_1 = f(p,q)$

ถ้า p,q คือ root < $\bar{p},\bar{q}$ > เราจะสามารถเขียน b_0 , b_1 ให้อยู่ในรูปของ Taylor Series ได้คือ

$$b_0(\bar{p},\bar{q}) = b_0(p,q) + \Delta p\left(\frac{\partial b_0}{\partial p}\right) + \Delta q\left(\frac{\partial b_0}{\partial q}\right) + \ldots \qquad \text{....(a)}$$

$$b_0(\bar{p},\bar{q}) = b_1(p,q) + \Delta p\left(\frac{\partial b_1}{\partial p}\right) + \Delta q\left(\frac{\partial b_1}{\partial q}\right) + \ldots \qquad \text{....(b)}$$

เมื่อ $\Delta p = \bar{p} - p$; $\Delta q = \bar{q} - q$

จากสมการ (a) และ (b) ทางซ้ายมือมีค่าเท่ากับ 0 และถ้าพิจารณาเฉพาะ 1^{st} order จะได้

$$\frac{\partial b_0}{\partial p}\Delta p+\frac{\partial b_0}{\partial q}\Delta q=-b_0(p,q)$$

$$\frac{\partial b_1}{\partial p}\Delta p+\frac{\partial b_1}{\partial q}\Delta q=-b_1(p,q)$$

จากสมการ ทำการ Solving เพื่อหาค่า $\Delta p, \Delta q$ โดยนำค่าของสมการ b_i มาทำ partial derivative เทียบกับ p,q จะได้

จากสมการ :

$$y = a_0 + a_1x + a_2x^2 + \ldots + a_Nx^N$$

$$y = (x^2 + px + q)G(x) + R(x)$$

เทียบสัมประสิทธิ์จากสมการทั้งสองจะได้

$$\begin{aligned}
a_N &= b_N \\
a_{N-1} &= b_{N-1} + pb_N \\
a_{N-2} &= b_{N-2} + pb_{N-1} + qb_N \\
&\vdots \\
a_2 &= b_2 + pb_3 + qb_4 \\
a_1 &= b_1 + pb_2 + qb_3 \\
a_0 &= b_0 + \quad + qb_2
\end{aligned}$$

แต่ค่าที่เราต้องคำนวณคือ b ดังนั้นจึงจัดสมการใหม่

$$\begin{aligned}
b_N &= a_N \\
b_{N-1} &= a_{N-1} + pb_N \\
b_{N-2} &= a_{N-2} + pb_{N-1} + qb_N
\end{aligned}$$

:

$$\boxed{b_i \quad = \quad a_i \quad pb_{i+1} \quad qb_{i+2}}$$

:

$$b_2 \quad = \quad a_2 + pb_3 + qb_4$$

$$b_1 \quad = \quad a_1 + pb_2 + qb_3$$

$$b_0 \quad = \quad a_0 + \quad + qb_2$$

p , q เป็นตัวเลข ใด ๆ

G(x) ----> Polynamial order N-2

R(x) -----> Remainder Polynomial function

ถ้าเลือกค่า p และ q แล้วทำให้ R(x) = 0 นั่นก็แสดงว่า $x^2 + px + q$ เป็น Quadratic factor ซึ่งจะสามารถหา Roots ได้จาก

$$\frac{-p \pm \sqrt{p^2 - 4q}}{2}$$

สามารถเขียน Polynomial ของ G(x) และ R(x) ได้คือ

$$G(x) = b_2 + b_3x + b_4x^2 + \ldots + b_Nx^{N-2}$$

$$R(x) = b_0 + b_1x$$

ซึ่งจะพบว่าค่า b_0 และ b_1 ขึ้นอยู่กับ p และ q ดังนั้นเราสามารถเขียน function ของ b_0 , b_1 ได้คือ

$$b_0(p,q) = b_0$$

$$b_1(p,q) = b_1$$

ถ้าค่า p,q ที่เลือก คือ root $<\bar{p},\bar{q}>$ จะได้

$$b_0(\bar{p},\bar{q}) = b_1(\bar{p},\bar{q}) = 0$$

$$(b_N)_p = 0$$

$(b_{N-1})_p$ = $-b_N$ $-p(b_N)_p$

$(b_{N-2})_p$ = $-b_{N-1}$ $-p(b_{N-1})_p$ - $q(b_N)_p$

$(b_i)_p$ = $-b_{i+1}$ $-p(b_{i+1})_p$ - $q(b_{i+2})_p$

$(b_2)_p$ = $-b_3$ $-p(b_{N-1})_p$ - $q(b_4)p$

สรุป Bairstow 's method

* set initial p , q and calculate b_0 , b_N
* calculate $(b_0)_p$, $(b_1)_p$, $(b_0)_q$, $(b_1)_p$
* solve for Δp , Δq
* obtain $\bar{p} = p + \Delta p$, $\bar{q} = q + \Delta q$
* check $(|\Delta p| + |\Delta q|) \leq \varepsilon$
* Root for quadratic factor

If $(p^2 - 4q) > 0$ ===>real root

$(p^2 - 4q) < 0$ ===>imagine root

let zz =p*p - Δq

print real root : $\frac{-p+\sqrt{zz}}{2}, -\frac{-p-\sqrt{zz}}{2}$

print imagine root

$$\frac{-p}{2} + j\frac{\sqrt{-zz}}{2}$$

$$\frac{-p}{2} - j\frac{\sqrt{-zz}}{2}$$

coefficient of deflated polynomail > G(x)

order	copefficint
0	b_2
1	b_3
.	.

จากนั้นก็นำค่าของ G(x) มาเข้าขบวนการใหม่จนกว่าจะได้ครบ

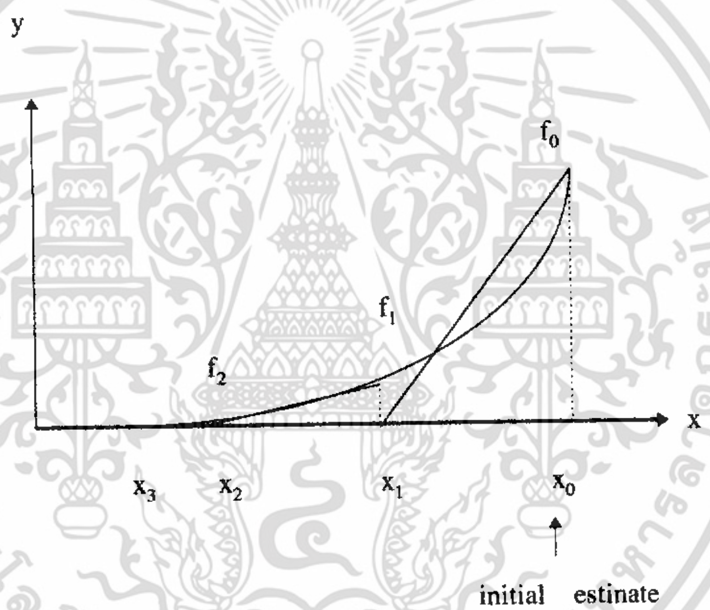
Newton's method

(Newton-Raphson Method)

- ใช้หาค่าที่ใกล้เคียงกับค่ารากจริงมากที่สุด เมื่อทราบค่าเริ่มต้น(Initial Values)โดยประมาณ
- ใช้หา Complex Roots
- ใช้หลักการของ Taylor expansion : ที่ $x=x_0$

$$f(x) = f(x_0) + hf'(x_0)+O(h^2) \; ; \; h= x-x_0$$

- หา ROOT : $y = f(x_0) = 0 = f(x_0) +(x+x_0 f'(x_0) + O(h^2)$



- พิจารณา tangential line ผ่านจุด $x_0, f'(x_0)$

$$g(x) = f'(x_0)\,(x-x_0)+f(x_0)$$

โดยที่ ROOT ของ $g(x) = 0$ คือ x_1

$$\therefore \quad g(x_1) = 0 = f'(x_0)\,(x_1-x_0) + f(x_0)$$

ดังนั้น $x_1 = x_0 - f(x_0)/f'(x_0)$

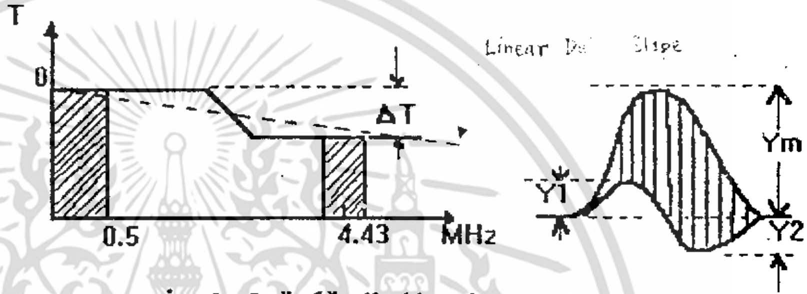
สรุป $x_i = f(x_{i=1})/f'(x_{i-1})$

บทที่ 2

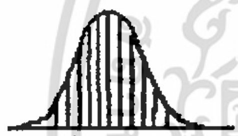
การออกแบบ

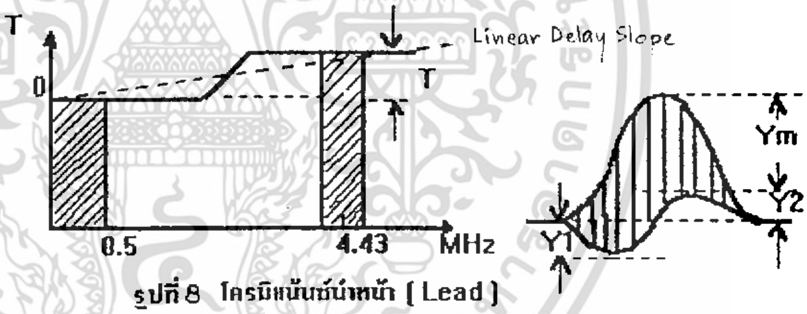
2.1 แนวทางการออกแบบ

ความผิดเพี้ยนทางด้านดีเลย์ของสัญญาณลูมิแนนซ์-โครมิแนนซ์การผิดเพี้ยนทางด้านดีเลย์ ซึ่งเป็นไปได้ทั้งแบบสัญญาณโครมิแนนซ์นำหน้า (Chrominance lead) หรือโครมิแนนซ์ล้าหลัง (Chrominance lag) แสดงดังรูปที่ 7 และรูปที่ 8



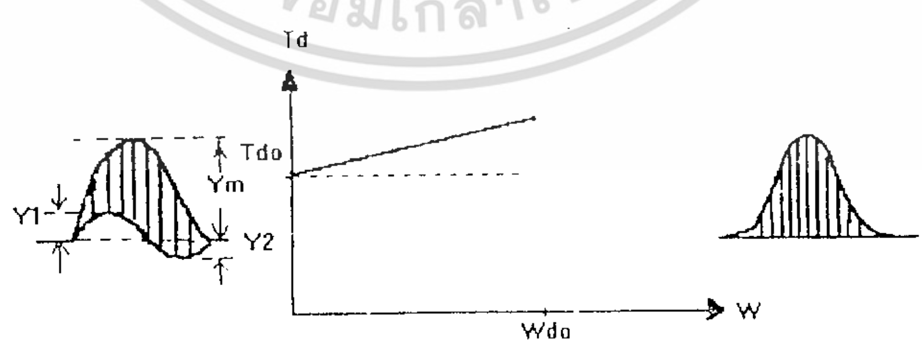
รูปที่ 7 โครมิแนนซ์ล้าหลัง (Lag)





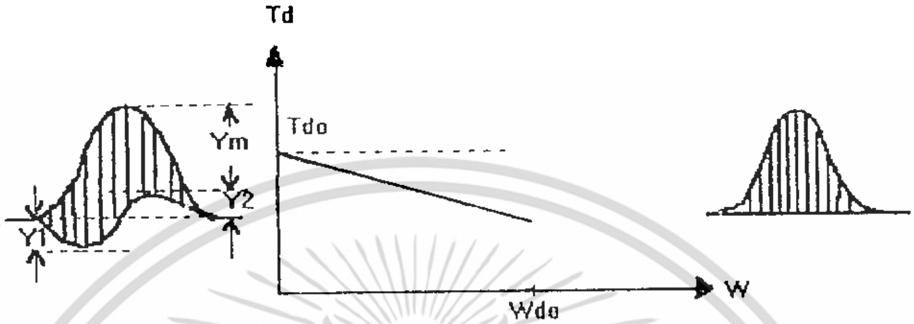
รูปที่ 8 โครมิแนนซ์นำหน้า (Lead)

การแก้ไขในรูปที่ 7 จะแก้ด้วยดีเลย์สโลปบวกเชิงเส้นดังรูปที่ 9



รูปที่ 9 สโลปบวกเชิงเส้น

การแก้ไขในรูปที่ 8 จะแก้ด้วยดีเลย์สโลปลบเชิงเส้นดังรูปที่ 10



รูปที่ 10สโลปลบเชิงเส้น

2.2 การออกแบบวิธีการแก้ปัญหาการดีเลย์ของสัญญาณลูมิแนนซ์โครมิแนนซ์

การออกแบบวิธีการแก้ปัญหาการดีเลย์ของสัญญาณลูมิแนนซ์โครมิแนนซ์โดยวิธีการดังนี้

จากทรานสเฟอร์ฟังก์ชั่นของ All pass filter

$$H(s) = \frac{Pn(-s)}{Pn(s)}$$

P(s) = เฮอร์วิทโพลิโนเมียลกำลังคู่

$= M(s) + kN(s)$

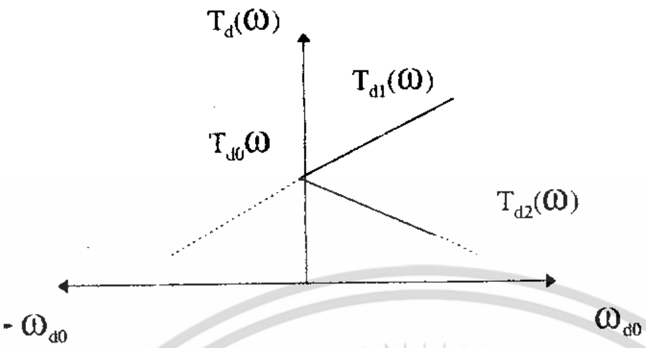
M(s) เป็นโพลิโนเมียลกำลังคู่

N(s) เป็นโพลิโนเมียลกำลังคี่

$$P(s) = (s^2 + \omega_1^2)(s^2+\omega_3^2)\ldots + ks(s^2+\omega_2^2)(s^2+\omega_4^2)\ldots$$
$$= s^n + a_{n-1}s^{n-1} + \ldots + a_1 s + a_0$$

เราจะต้องทำการหาค่าสัมประสิทธิ์โพลิโนเมียลเพื่อที่จะได้ดีเลย์สโลปเชิงเส้นที่ประมาณค่าเข้าใกล้เส้นดีเลย์สโลปเชิงเส้น

จากสมการของกราฟดีเลย์เชิงเส้น



จาก $y = mx + b$

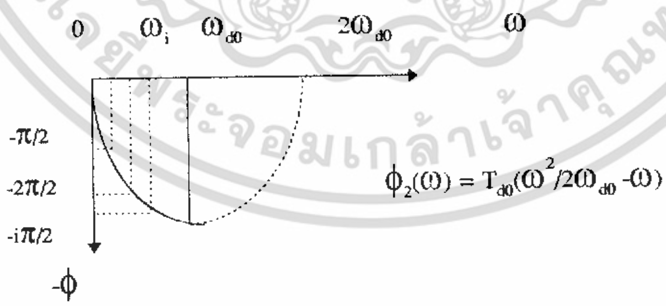
$m = T_{d0}/\omega_{d0}$; $x = \omega$

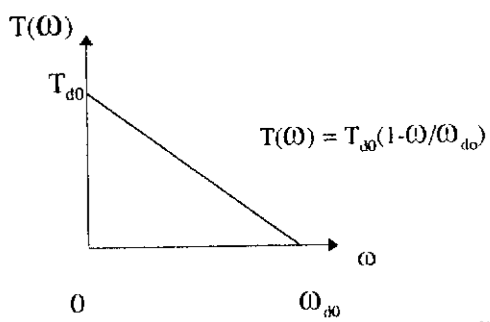
$$T_{d1}(\omega) = \frac{\omega}{\omega d_0} Td_0(\omega) + Td_0$$

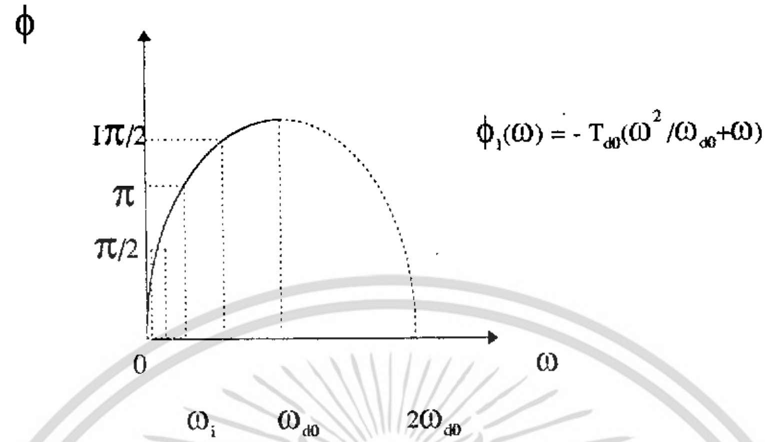
$$= Td_0\left(1 + \frac{\omega}{\omega_{d_0}}\right) \quad \text{;สโลปบวก} \qquad ....(14)$$

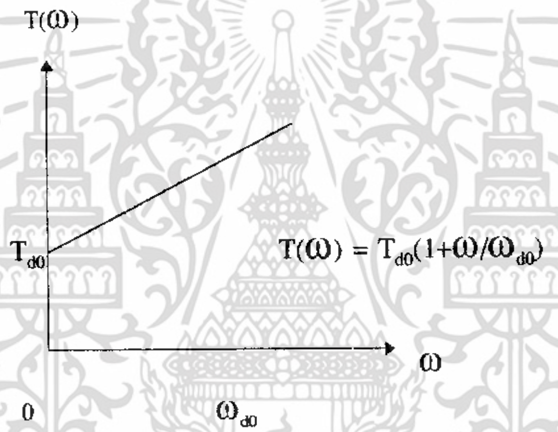
$$T_{d1}(\omega) = -T_{d0}\left(\frac{\omega}{\omega_{d0}}\right) + T_{d0}$$

$$= T_{d0}\left(1 - \frac{\omega}{\omega_{d0}}\right) \quad \text{;สโลปลบ} \qquad ....(15)$$









กำหนดให้กรุ๊ปดีเลย์ $T_{d0}(\omega) = -\dfrac{d\phi(\omega)}{d\omega}$

ค่าที่เฟสสอดคล้องกับ (14) และ (15)

$$\phi_1(\omega) = -T_{d0}\left[\frac{\omega^2}{2\omega_{d0}} + \omega\right] \qquad \text{....(16)}$$

$$\phi_2(\omega) = T_{d0}\left[\frac{\omega}{2\omega_{d0}} - \omega\right] \qquad \text{....(17)}$$

เมื่อ $$-\frac{d\phi_1}{d\omega}(\omega) = -\frac{d}{d\omega}\left(-T_{d0}(\omega + \frac{\omega^2}{2\omega_{d0}})\right)$$

$$= \quad T_{d1}(\omega)$$

$$T_{d1}(\omega) \quad = \quad T_{d0}\left(1+\frac{\omega}{\omega_{d0}}\right)$$

เช่นเดียวกับการหา $T_{d2}(\omega)$

$$T_{d2}(\omega) = -\frac{d\phi_2}{d\omega}(\omega)$$

$$= -\frac{d}{d\omega}\left(T_{d0}\left(\frac{\omega^2}{2\omega_{d0}}-\omega\right)\right)$$

$$= T_{d0}\left(1-\frac{\omega}{\omega d_0}\right)$$

จาก (16) และ (17)

$$\phi(\omega_i) = T_{d0}\frac{\omega^2}{2\omega_{d0}} - \omega T_{d0}$$

$$0 = \frac{T_{d0}}{2\omega d_0}\omega^2 - T_{d0}\omega - \phi(\omega_i) \qquad ....(18)$$

จากสูตรแยกแฟกเตอร์

$$X = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$$

เพราะฉะนั้นจากสมการ (18)

$$\omega = \frac{-(-T_{d0}) \pm \sqrt{(-T_{d0})^2 - 4(T_{d0}/2\omega_{d0})(-\phi(\omega_i))}}{2(T_{d0}/2\omega_{d0})}$$

$$= \frac{T_{d0}}{T_{d0}/\omega_{d0}} \pm \frac{1}{T_{d0}/\omega_{d0}} \sqrt{(T_{d0})^2 + 2\frac{T_{d0}}{\omega_{d0}}(\phi(\omega_i))}$$

$$= \omega_{d0} \pm \frac{\omega d_0}{T_{d0}} \sqrt{(T_{d0})^2 + \frac{2T_{d0}\phi(\omega_i)}{\omega d_0}}$$

$$= \omega_{d0} \pm \omega_{d0} \sqrt{(T_{d0})^2 / (T_{d0})^2 + \frac{2T_{d0}(\phi(\omega_i))}{(T_{d0})^2 \omega_{d0}}}$$

$$= \omega_{d0} \pm \omega_{d0} \sqrt{1 + \frac{2(\phi(\omega_i))}{T_{d0}\omega_{d0}}}$$

ฉะนั้นที่สโลปบวกจะได้

$$\omega_i = \omega_{d0}\left[\sqrt{1 + \frac{2(\phi(\omega_i))}{T_{d0}\omega_{d0}}} - 1\right] \qquad ....(19)$$

และที่สโลปลบ

$$\omega_i = \omega_{d0}\left[1 - \sqrt{1 + \frac{2(\phi(\omega_i))}{T_{d0}\omega_{d0}}}\right] \qquad ....(20)$$

เพื่อให้ค่าเฟสที่เหมาะสมในทางอุดมคติกำหนดที่จุด $0, \pm\pi/2, \pm 2\pi/2, \pm 3\pi/2, \ldots\ldots$ โดยให้ $\phi(\omega) = -\frac{i\pi}{2}$ ที่สโลปลบและ $\phi(\omega_i) = \frac{i\pi}{2}$ ที่สโลปบวก โดยแทนใน (19) และ (20) จะได้

ที่สโลปบวก

$$\omega_i = \omega_{d0}\left(\sqrt{1 + \frac{2(i\pi/2)}{T_{d0}\omega_{d0}}} - 1\right)$$

$$= \omega_{d0}\left[\sqrt{1 + \frac{i\pi}{T_{d0}\omega_{d0}}} - 1\right] \qquad ....(21)$$

และที่สโลปลบ

$$\omega_i = \omega_{d0}\left(1 - \sqrt{1 + \frac{2(-i\pi/2)}{T_{d0}\omega_{d0}}}\right)$$

$$= \omega_{d0}\left[1 - \sqrt{1 - \frac{i\pi}{T_{d0}\omega_{d0}}}\right] \quad \text{....(22)}$$

$i \;=\; 1,2,3,\ldots\ldots\ldots,(n-1)$

ในการกำหนดสโลปเชิงเส้นที่เหมาะสมให้ $\omega_{n-1} = \omega_{d0}$ ใน (20) และ (21) คือ

$$i = n-1$$

$$\omega_{n-1} = \omega_{d0}$$

$$\omega_{d0} = \omega_{d0}\left[\sqrt{1 + \frac{(n-1)\pi}{T_{d0}\omega_{d0}}} - 1\right]$$

$$\frac{\omega_{d0}}{\omega_{d0}} + 1 = \sqrt{1 + \frac{(n-1)\pi}{T_{d0}\omega_{d0}}}$$

$$2 = \sqrt{1 + \frac{(n-1)\pi}{T_{d0}\omega_{d0}}}$$

$$2^2 = 1 + \frac{(n-1\pi)}{T_{d0}\omega_{d0}}$$

$$4-1 = \frac{(n-1)\pi}{T_{d0}\omega_{d0}}$$

$$T_{d0}(\omega) = \frac{(n-i)\pi}{3\omega_{d0}} \qquad \text{สโลปบวก}$$

ทำนองเดียวกัน

$$\omega_{d0} = \omega_{d0}\left(1-\sqrt{1-\frac{i\pi}{T_{d0}\omega_{d0}}}\right)$$

$$1-1 = -\sqrt{1-\frac{i\pi}{T_{d0}\omega_{d0}}}$$

$$0 = \left(-\sqrt{1-\frac{(n-1)\pi}{T_{d0}\omega_{d0}}}\right)^2$$

$$1 = \frac{(n-1)\pi}{T_{d0}\omega_{d0}}$$

$$T_{d0} = \frac{(n-1)\pi}{\omega_{d0}}$$ สโลปลบ

ที่สโลปลบหาค่า k เพื่อจะให้ผลเส้นดีเลย์สโลปที่ทำการหาเข้าใกล้เส้นดีเลย์สโลปทางอุดมคติมากที่สุด

$$T_{ideal} = T_0\left(1-\frac{\omega_i}{\omega_{d0}}\right)$$

$$Td(\omega) = \frac{N\frac{dM}{d\omega} - M\frac{dN}{d\omega}}{M^2+N^2}$$

$$k = \frac{T_{ideal}}{T_d(\omega)}$$ เมื่อ $\omega = \omega_1, \omega_3, \omega_5, \ldots$

$$k = \frac{T_d(\omega)}{T_{ideal}}$$ เมื่อ $\omega = \omega_2, \omega_4, \omega_6, \ldots$

จากสมการ $\omega_i = \omega_{d0}\left[1-\sqrt{1-\frac{i\pi}{T_{d0}\omega_{d0}}}\right]$ for I = 0,1,2,3,....,(n-1)

$$\omega_i = \omega_{d0}\left[1-\sqrt{1-\frac{i}{n-1}}\right] \qquad ;\mathbf{T_{d0}} = \quad (n\text{-}1);\ T_{d0}(n-1)\pi/\omega_{d0}$$

$$k = \frac{T_{d0}(1-(\omega_i/\omega_{d0}))}{\left(N\frac{dM}{d\omega}-M\frac{dN}{d\omega}\right)/\left(M^2+N^2\right)}$$

เมื่อ $M(\omega) = 0$

$$k = \frac{T_{d0}(1-(\omega_i/\omega_{d0}))}{\frac{N\frac{dM}{d\omega}}{N^2}}$$

$$= \frac{NT_{d0}(1-(\omega_i/\omega_{d0}))}{\frac{dM}{d\omega}} \qquad ....(23)$$

แทน ω_i ใน (23)

$$k = \frac{NT_{d0}}{\frac{dM}{d\omega}}\left(1-\frac{\omega_{d0}}{\omega_{d0}}\left(1-\sqrt{1-\frac{i}{n-1}}\right)\right)$$

$$= \frac{NT_{d0}}{\frac{dM}{d\omega}}\left(\sqrt{1-\frac{i}{n-1}}\right) \qquad ....(24)$$

ในทำนองเดียวกันเมื่อ $N(\omega) = 0$

$$k = -\frac{\frac{dN}{d\omega}}{M(\omega)T_{d0}\sqrt{1-\frac{i}{n-1}}} \qquad ....(25)$$

และค่าที่สโลปบวกการหาค่า k

จาก

$$T_{d0} = \frac{(n-1)\pi}{3\omega_{d0}}$$

$$T_{ideal} = T_0\left(1+\frac{\omega_i}{\omega_{d0}}\right)$$

$$T_d(\omega) = \frac{N\frac{dM}{d\omega} - M\frac{dN}{d\omega}}{M^2+N^2}$$

$$\omega_I = \omega_{d0}\left[\sqrt{1+\frac{3i}{(n-1)}}-1\right]$$

$$k = \frac{NT_{d0}}{\frac{dM}{d\omega}}\left(\sqrt{1+\frac{3i}{n-1}}\right) \quad \text{....(26)}$$

เมื่อ $\omega = \omega_1, \omega_3, \omega_5, \ldots$

$$k = -\frac{\frac{dN}{d\omega}}{M(\omega)T_{d0}\sqrt{1+\frac{3i}{n-1}}} \quad \text{....(27)}$$

เมื่อ $\omega = \omega_2, \omega_4, \omega_6, \ldots$

หาค่า k, T_0, ω_i

(หมายเหตุ อยากได้สโลปลบหรือบวกก็นำ T_{d0} ของสโลปเหล่านั้นไปแทนในสมการ k ได้)

ที่สโลปบวก เมื่อกำหนดให้ $\omega_{d0} = 2.3$ และ $N = 6$

$$\text{จาก}\quad T_{d0}(\omega) = (n-1)\pi / (3)(2.3)$$

$$= (5-1)\pi / (3)(2.3)$$

$$= 2.27$$

จาก

$$\omega_i = \omega_{d0}\left(\sqrt{1+\frac{3i}{(n-1)}}-1\right)$$

$$\omega_1 = (2.3)\left[\sqrt{1+\frac{3(1)}{6-1}}-1\right]$$

$$= 0.60092$$

$$\omega_2 = (2.3)\left[\sqrt{1+\frac{3(2)}{6-1}}-1\right]$$

$$= 1.11145$$

$$\omega_3 = (2.3)\left[\sqrt{1+\frac{3(3)}{6-1}}-1\right]$$

$$= 1.548636$$

$$\omega_4 = (2.3)\left[\sqrt{1+\frac{3(4)}{6-1}}-1\right]$$

$$= 1.9409$$

$$\omega_5 = (2.3)\left[\sqrt{1+\frac{3(5)}{6-1}}-1\right]$$

$$= 2.3$$

สรุป $\omega_{(n-1)} = \omega_{d0}$

คือ $\omega_5 = 2.3$

นำค่า k มาหา ส.ป.ส. Pn(s)

$$Pn(s) = M(s) + KN(s)$$

$$= (s^2+\omega_1^2)(s^2+\omega_3^2)\ldots+ks(s^2+\omega_2^2)(s^2+\omega_4^2)\ldots$$

การหาค่า k เมื่อ $N = 6$; $T_{d0} = 2.27$

$I = 1$; $\omega = \omega_1$

$$k = \frac{N(\omega)}{\frac{dM(\omega)}{d\omega}} T_{d0} \sqrt{1+\frac{3i}{(n-1)}}$$

$$= \frac{j\omega(\omega_2^2-\omega^2)(\omega_4^2-\omega^2)}{\frac{d\left[\left(\omega_1^2-\omega^2\right)\left(\omega_3^2-\omega^2\right)\left(\omega_5^2-\omega^2\right)\right]}{d\omega}} \times 2.27\sqrt{1+\frac{3(1)}{(6-1)}}$$

$$k = \frac{j\omega(\omega_2^2-\omega^2)(\omega_4^2-\omega^2)\cdot(2.871)}{\frac{d}{d\omega}\left(\omega_1^2\omega_3^2\omega_5^2-\left(\omega_1^2\omega_3^2+\omega_1^2\omega_5^2+\omega_5^2\omega_3^2\right)\omega^2+\left(\omega_1^2+\omega_3^2+\omega_5^2\right)\omega^4-\omega^6\right)}$$

$$= \frac{j\omega\left(\omega_2^2-\omega^2\right)\left(\omega_4^2\right)\cdot(2.871)}{0-2\omega\left(\omega_1^2\omega_3^2+\omega_1^2\omega_5^2+\omega_3^2\omega_5^2\right)+4\omega^3\left(\omega_1^2+\omega_3^2+\omega_5^2\right)-6\omega^5}$$

เมื่อ $\omega = \omega_1$

$= j(0.609295)(0.86419)(3.396)(2.871)/(-18.19+7.288-0.503)$

$= j\ 5.134\ /\ -12.145$

$= -j0.4227$

$|k| = 0.4227$

$I = 2\ ;\quad \omega = \omega 2$

$$k = \frac{-\frac{dN(\omega)}{d\omega}}{M(\omega)\cdot T_{d0}\sqrt{1+\frac{3i}{n-1}}}$$

$$= \frac{-j\frac{d\omega\left(\omega_2^2-\omega^2\right)\left(\omega_4^2-\omega^2\right)}{d\omega}}{\left(\omega_1^2-\omega^2\right)\left(\omega_3^2-\omega^2\right)\left(\omega_5^2-\omega^2\right)\left(2.27\sqrt{1+6/5}\right)}$$

$$= \frac{-j\dfrac{d\left(\omega_2^2\omega_4^2\omega\right)\left(\omega_2^2+\omega_4^2\right)\left(\omega^3+\omega^5\right)}{d\omega}}{\left(\omega_1^2-\omega^2\right)\left(\omega_3^2-\omega^3\right)\left(\omega_5^2-\omega(3.366)\right)}$$

$$= \frac{-j\left(\omega_2^2\omega_4^2-3\omega^2\left(\omega_2^2+\omega_4^2\right)+5\omega^4\right)}{\left(\omega_1^2-\omega^2\right)\left(\omega_3^2-\omega^3\right)\left(\omega_5^2-\omega^2\right)(3.366)}$$

$\omega = \omega_2$

$= -j(-6.255)/(-0.864)(1.162)(4.05)(3.366)$

$k = -j(-6.255)/(-13.702)$

$= -j0.456$

$|k| = 0.456$

$I = 3 \;;\; \omega = \omega_3$

$$k = \frac{N(\omega)\cdot T_{d0}\sqrt{1+3i/(n-1)}}{\dfrac{dM(\omega)}{d\omega}}$$

$$k = \frac{j\omega(\omega_2^2-\omega^2)(\omega_4^2-\omega^2)\cdot(2.27\sqrt{1+9/(6-1)}}{-2\omega(\omega_1^2\omega_3^2+\omega_1^2\omega_5^2+\omega_3^2\omega_5^2)+4\omega^3(\omega_1^2+\omega_3^2+\omega_5^2)-6\omega^5}$$

$= -j9.364 / (-48.129+119.72-53.437)$

$= -j9.364 / 18.154$

$= -j0.515$

$|k| = 0.515$

$I = 4 \;;\; \omega = \omega_4$

$$k = \frac{\dfrac{-dN(\omega)}{d\omega}}{M(\omega)\cdot T_{d0}\sqrt{1+3i/(n-1)}}$$

$$= \frac{-j(\omega_2^2\omega_4^2 - 3\omega^2(\omega_2^2+\omega_4^2)+5\omega^4}{(\omega_1^2-\omega^2)(\omega_3^2-\omega^2)(\omega_5^2-\omega^2)(4.1856)}$$

$= -j(4.654 - 56.54 + 70.968)/(7.080)(4.1856)$

$= -j\ (19.082)\ /\ 29.634$

$= -j0.6439$

$|k| \quad = \quad 0.6739$

$$k = \frac{N(\omega)}{\frac{dM(\omega)}{d(\omega)}} \cdot T_{d0}\sqrt{1+3i/(n-1)}$$

$$k = \frac{j\omega(\omega_2^2-\omega^2)(\omega_4^2-\omega^2)\cdot\left(2.227\sqrt{1+3(5)/(6-1)}\right)}{-2\omega(\omega_1^2\omega_3^2+\omega_1^2\omega_5^2+\omega_3^2\omega_5^2)+4\omega^3(\omega_1^2+\omega_3^2+\omega_5^2)-6\omega^5}$$

$$= \frac{j\omega(\omega_2^2-\omega^2)(\omega_4^2-\omega^2)\cdot(4.54)}{-2\omega(\omega_1^2\omega_3^2+\omega_1^2\omega_5^2+\omega_3^2\omega_5^2)+4\omega^3(\omega_1^2+\omega_3^2+\omega_5^2)-6\omega^5}$$

$\omega = \omega_5$

$= j(2.3)(6.1734)(4.541)\ /\ (-71.48 + 392.229 - 386.18)$

$= j(64.46)\ /\ (-65.431)$

$= -j0.985$

$|k| \quad = 0.985$

ในที่นี้เมื่อ $n = 6$

$$P_6(s) = a_6s^6 + a_{6-1}s^5 + a_4s^4 + a_3s^3 + a_2s^2 + a_1s^1 + a_0$$

$$= (s^2+\omega_1^2)(s^2+\omega_3^2)(s^2+\omega_5^2) + ks(s^2+\omega_2^2)(s^2+\omega_4^2)$$

$$= \quad s^6 + (\omega_1^2 + \omega_3^2)s^4 + (\omega_1^2 + \omega_3^2)s^2 + \omega_5^2 s^4$$
$$+\omega_5^2(\omega_1^2 + \omega_3^2)s^2 + \omega_1^2\omega_3^2\omega_5^2 + s^5 + (\omega_2^2\omega_4)s^3 + (\omega_2^2\omega_4^2)s$$

$$s^6 + ks^5 + [(\omega_1^2 + \omega_3^2) + \omega_5^2]s^4 + [(\omega_2^2 + \omega_4^2)s^3]k$$
$$= \quad +[(\omega_1^2\omega_3^2) + \omega_5^2(\omega_1^2 + \omega_3^2)]s^2 + k(\omega_2^2\omega_4^2)s$$
$$+\omega_1^2\omega_3^2\omega_5^2$$

เมื่อ $k = 1$

$a_6 = 1$

$a_5 = 1$

$a_4 = 8.05969$

$a_3 = 5.00276$

$a_2 = 14.065171$

$a_1 = 4.65357$

$a_0 = 4.71024$

แทนค่า $k = 0.4236$

$$= s^6 + 0.4236s^5 + 8.05969s^4 + 2.11917s^3 + 14.65171s^2$$
$$+ 1.97125s + 4.17024$$

แทนค่า $k = 0.4548$

$$= s^6 + 0.4548s^5 + 8.05969s^4 + 2.275255s^3 + 14.6517s^2$$
$$+2.116444s + 4.71024$$

แทนค่า $k = 0.5127$

$$= s^6 + 0.5127s^5 + 8.05969s^4 + 2.587428s^3 + 14.65171s^2$$
$$+2.988057s + 4.71024$$

แทนค่า $k = 0.9878$

$$= s^6 + 0.9878s^5 + 8.05969s^4 + 4.941726s^3 + 14.65171s^2$$
$$+ 4.596796s + 4.71024$$

จาก $H(s) = \frac{P(-s)}{P(s)}$; ฟังก์ชั่นมาตรฐานของ all pass filter

$n = 6$

$$Pn(-s) = a_6s^6 - a_5s^5 + a_4s^4 - a_3s^3 + a_2s^2 - a_1s + a_0$$

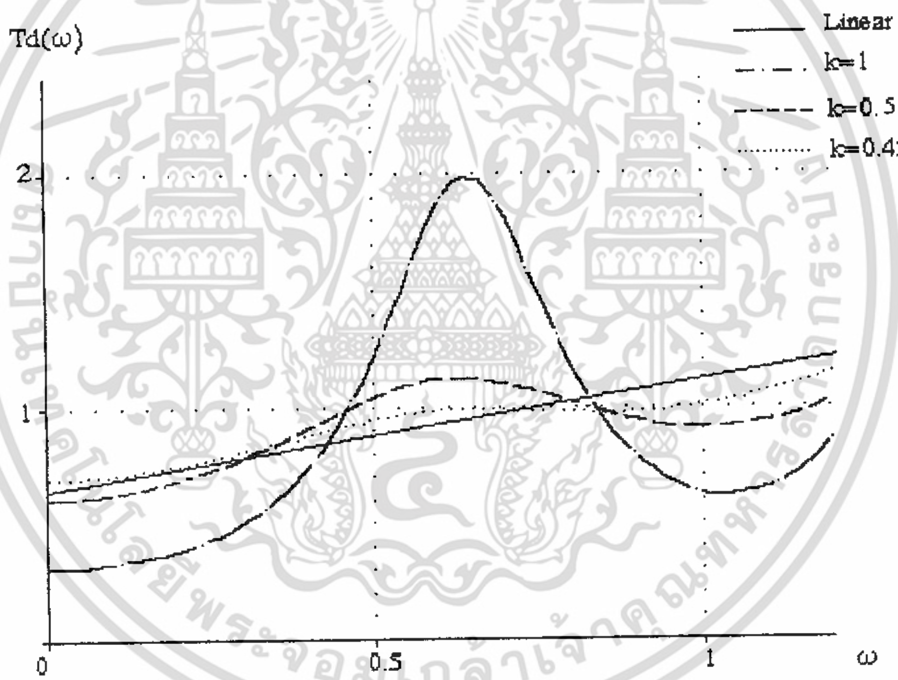
$$Pn(s) = a_6s^6 + a_5s^5 + a_4s^4 + a_3s^3 + a_2s^2 + a_1s + a_0$$

จากการ plot graph ค่า k = 0.4236 จะเข้าใกล้ slope delay ดีที่สุด

$$H(s) = \frac{s^6 - 0.4236s^5 + 8.05969s^4 - 2.11917s^3 + 14.65171s^2 - 1.97125s + 4.71024}{s^6 + 0.4236s^5 + 8.05969s^4 + 2.11917s^3 + 14.65171s^2 + 1.97125s + 4.71024}$$

2.8 การพิจารณาค่า k ที่เหมาะสมจากกราฟเพื่อนำไปใช้ในการออกแบบวงจร

แสดงการพล็อตกราฟจากส่วนของทรานสเฟอร์ฟังก์ชั่นที่ k ค่า 0.4236 ,0.5127 และ 1 เพื่อดูกราฟที่มีลักษณะใกล้เคียงกับเส้นกราฟเชิงเส้นในสโลปด้านบวกมากที่สุด และนำค่า ω ที่ได้ไปทำการหาทรานสเฟอร์ฟังก์ชั่น เพื่อนำไปใช้ในการออกแบบวงจรต่อไปซึ่งผลจากการพล็อตกราฟจากโปรแกรมได้กราฟที่มีลักษณะแตกต่างกันดังต่อไปนี้



2.4 การออกแบบวงจรโดยใช้วิธีการออปติไมซ์เซชั่น

จากการใช้หลักการปรับแก้ค่าผิดพลาดในการออกแบบของซึ่งจะพิจารณาได้จากเส้นกราฟที่ได้จากการนำค่าที่พล็อตจากรากของส่วนจากทรานสเฟอร์ฟังก์ชั่นเดิมเราจะเลือกเส้นกราฟที่มีค่า k=0.4236 เพราะมีลักษณะใกล้เคียงกับเส้นสโลปสมมติที่ใช้ในการแก้ปัญหาทางด้านดีเลย์ไทม์มากที่สุด ซึ่งวิธีการปรับแก้ค่าความผิดพลาดโดยการออปติไมซ์ด้วยคอมพิวเตอร์นั้นสามารถสรุปขั้นตอนได้ดังต่อไปนี้

ขั้นตอนในการออปติไมซ์มีดังนี้ คือ

1. ทำการหาโพลของทรานสเฟอร์ฟังก์ชั่นโดยการนำไปหารากด้วยวิธีการของ Bairstow Method ซึ่งได้ผลลัพธ์จากการประมวลผลของโปรแกรมดังนี้

```
Order of Polynomial = 6
 Enter Coefficient of equation ( Max to min order)
a1 = 1
a2 = 0.42360
a3 = 8.05969
a4 = 2.11917
a5 = 14.65171
a6 = 1.97125
a7 = 4.71024
 Admissible Error = 0.0000000000001
 max. iteration = 200
Root1  -6.9487132001E-02   6.3989054353E-01i
Iteration = 6
Root2  -6.9487132001E-02  -6.3989054353E-01i
Iteration = 6
Root3  -5.1667605733E-02   1.4326942020E+00i
Iteration = 6
Root4  -5.1667605733E-02  -1.4326942020E+00i
Iteration = 6
Root5  -9.0645262266E-02   2.3502369035E+00i
Iteration = 2
Root6  -9.0645262266E-02  -2.3502369035E+00i
Iteration = 2
```

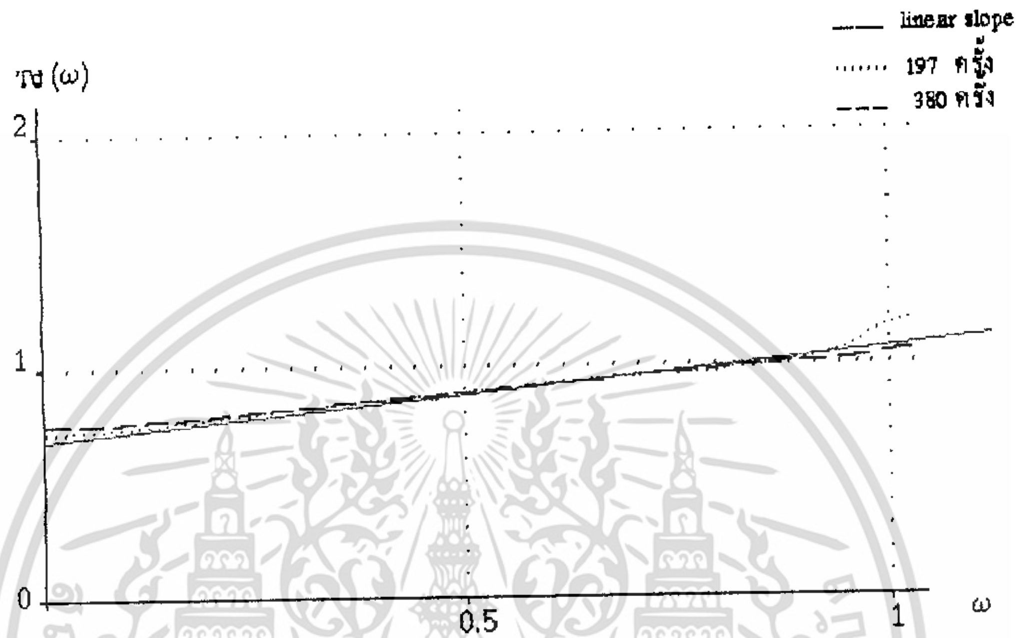
ผลการ RUN โปรแกรม Bair โดยใช้หลักการ Bairstow's method

2. Iterate ค่าของรากที่ได้จากการใช้โปรแกรม bairstow ถอดรากสมการโพลิโนเมียลซึ่งกำหนดค่าที่ยอมเกิดความผิดพลาดได้ $1 * 10^{-13}$ เท่านั้น ซึ่งค่ารากที่ได้ดังกล่าวจะนำไปใช้ในการปรับค่าให้มีความถูกต้องมากขึ้นโดยใช้หลักการของ Newton Raphson ซึ่งจะทำการ iterate ค่ารากให้มีค่าใกล้เคียงสโลปเชิงเส้นมากที่สุด โดยจะทำการ iterate ช่วงแรก 197 ครั้งและนำมาพิจารณาผลด้วยกราฟ จนได้กราฟเป็นที่น่าพอใจซึ่งพบว่าที่ 380 ครั้ง จะมีค่ารากเหมาะสมที่สุด ซึ่งผลที่ได้จากการ iterate และการพล็อตกราฟ ของแต่ละช่วงมีผลดังต่อไปนี้

ผลจากการ iterate 197 ครั้งและ 380 ครั้ง

```
                              Iterate 0

a1 =   0.0694871320                 b1 =   0.6398905435
a2 =   0.0516676057                 b2 =   1.4326942020
a3 =   0.0906452623                 b3 =   2.3502369035

w 1 =    0.0000000000E+00           delta D =    5.8934977033E-01
w 2 =    6.3989054353E-01           delta D =   -3.8128478041E+00
w 3 =    1.0362923728E+00           delta D =    6.1831096161E-01
w 4 =    1.4326942020E+00           delta D =   -5.2881645841E+00
w 5 =    1.8914655527E+00           delta D =    7.9388253875E-01
w 6 =    2.3502369035E+00           delta D =   -2.4469335536E+00

First[197]                    Iterate 197

a1 =   0.8475033141                 b1 =   0.6439796803
a2 =   0.7272263085                 b2 =   1.4169018385
a3 =   0.4695584874                 b3 =   2.1126060195

w 1 =    0.0000000000E+00           delta D =   -1.8189894035E-12
w 2 =    6.4397968034E-01           delta D =   -3.5879788071E-12
w 3 =    1.0304407594E+00           delta D =   -1.8689894035E-12
w 4 =    1.4169018385E+00           delta D =    9.1449470177E-12
w 5 =    1.7647539290E+00           delta D =    1.5411409930E-11
w 6 =    2.1126060195E+00           delta D =   -8.9080480774E-11

First[197]                    Iterate 380

a1 =   0.9856317019                 b1 =   0.6323788308
a2 =   0.8555511488                 b2 =   1.3692956359
a3 =   0.5723911274                 b3 =   2.0063312996

w 1 =    2.1079294360E-01           delta D =   -5.0693981912E-03
w 2 =    6.3237883079E-01           delta D =   -5.9108496266E-05
w 3 =    8.1509951142E-01           delta D =   -9.5020427629E-05
w 4 =    1.1523229269E+00           delta D =   -1.9813030571E-04
w 5 =    1.5054969612E+00           delta D =    2.5171188763E-03
w 6 =    1.7267906720E+00           delta D =   -2.4699541513E-04
```

กราฟที่ทำการ iterate 197 ครั้งและ 380 ครั้ง



3. ทำการหาทรานสเฟอร์ฟังก์ชั่นใหม่จาก ω ของการ iterate ครั้งที่ 380

ω_0 = 0.21079294360

ω_1 = 0.63237883079

ω_2 = 0.81509951142

ω_3 = 1.1862526363

ω_4 = 1.5054969612

ω_5 = 1.7267906720

ทำการแทนค่าและคำนวณหาทรานเฟอร์ฟังก์ชั่นใหม่ได้ดังนี้ คือ

$$H(s) = \frac{s^6 - 0.4326s^5 + 4.78890s^4 - 1.241532761s^3 + 5.951158212s^2 - 0.6378770595s + 1.677986318}{s^6 + 0.4326s^5 + 4.78890s^4 + 1.241532761s^3 + 5.951158212s^2 + 0.6378770595s + 1.677986318}$$

4. การออกแบบวงจรนี้เราจะใช้รูปแบบของ Single - Amplifier generation Biquad (SAB) ซึ่งเป็นรูปแบบของ 2nd order ในลักษณะของ negative feedback RC amplifier network จากทรานสเฟอร์ฟังก์ชั่นที่ได้จะเห็นว่ามี 6th order เราจะใช้ SAB ต่อในลักษณะ CASCADE ที่สโลปบวกทรานสเฟอร์ฟังก์ชั่นที่ได้

$$H(s) = \frac{s^6 - 0.4326s^5 + 4.78890s^4 - 1.241532761s^3 + 5.951158212s^2 - 0.6378770595s + 1.677986318}{s^6 + 0.4326s^5 + 4.78890s^4 + 1.241532761s^3 + 5.951158212s^2 + 0.6378770595s + 1.677986318}$$

จากทรานสเฟอร์ฟังก์ชั่น H(s) order 6th ของ Linear slope ด้านบวก ทำการออกแบบวงจรโดยการแยกแฟคเตอร์ออกเป็น H(s) order 2nd 3 ชุด โดยใช้โปรแกรม

$$H(s) = H_1(s) \cdot H_2(s) \cdot H_3(s)$$

ซึ่งมีผลการ RUN โปรแกรมดังนี้คือ

```
The order of polynomial ?  6
a(0)?    1.677986318
a(1)?    0.6378770595
a(2)?    5.951158212
a(3)?    1.241532761
a(4)?    4.788904327
a(5)?    0.4236
a(6)?    1
Tolerance ?    0.0000000000001
---------------------------------------------------
 p =0.0806202      q =0.407707
Quadratic coefficient = x^2 + (0.0806202 x) + (0.407707 )
Roots of the quadratic factor:
   -0.0403101 + 0.637246i
   -0.0403101 - 0.637246i
Coefficients of deflated polynomial
       order   coefficients
         0          4.11566
         1          0.75071
         2          4.35355
         3          0.34298
         4          1.00000
---------------------------------------------------
Type 1 to continue, or 0 to stop.
```

จากผลการ Run โปรแกรมเป็นการแยกแฟกเตอร์โพลิโนเมียลออกทีละกำลัง 2 จากโพลิโนเมียลอันดับที่ 6 ออกเป็น โพลิโนเมียลอันดับที่ 2 คูณกับโพลิโนเมียลอันดับที่ 4 และเราจะทำการแยกโพลิโนเมียล 4 ออกเป็น อันดับที่ 2 คูณกันซึ่งจะได้ผลลัพธ์ดังนี้ คือ

$$H(s) = \left(\frac{s^2 - 0.080620s + 0.407707}{s^2 + 0.0806202s + 0.407707}\right)\bullet\left(\frac{s^2 - 0.177614s + 1.41445}{s^2 + 0.177614s + 1.41445}\right)\bullet\left(\frac{s^2 - 0.165375s + 2.90973}{s^2 + 0.165375s + 2.90973}\right)$$

จะเห็นว่าการแยกแฟคเตอร์ของทรานสเฟอร์ฟังก์ชั่นมีวัตถุประสงค์เพื่อความสะดวกในการหาค่าอุปกรณ์ของวงจรจากทรานสเฟอร์ฟังก์ชั่นย่อยในแต่ละชุดนั่นเอง

การออกแบบวงจร Linear slope จากทรานสเฟอร์ฟังก์ชั่นทางด้านบวก

การหาคำนวณหาค่าความต้านทานและค่าตัวเก็บประจุของวงจรชุดที่ 1 จาก $H_1(s)$

$$H_1(s) = \left(\frac{s^2 - 0.086202 + 0.407707}{s^2 + 0.086202 + 0.407707}\right)$$

K=1, b_2= 1, b_1=-0.086202, b_0=0.407707, a_1=0.086202, a_0=0.407707

กำหนดให้ $C_1 = C_2 = 1F$, $\gamma = 0.1$, $0 \le K_3 \le 1$

จากสมการที่ (13) จะได้

$$R1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}}$$

$$R1 = \frac{2*0.1}{-0.0.086202 + \sqrt{0.086202^2 + (8 \times 0.407707 \times 0.1)}}$$

$$= \frac{0.2}{0.491376}$$

$$= 0.407020 \ \Omega$$

$$R_4 = \frac{R_1}{K_1} = 0.407020\ \Omega$$

$$R_5 = \frac{R_1}{1-K_1} = \infty$$

$$R_D = \frac{r_1}{K_2}$$

$$= \frac{r_1}{0.94}$$

$$= 1.0636\ r_1$$

$$R_D = \frac{r_1}{1-K_2}$$

$$= 16.666\ r_1$$

$$R_6 = \frac{R_3}{K_3} = \infty$$

$$R_7 = \frac{R_3}{1-K_3} = \infty$$

ทำการ scale ค่า RC ใช้ magnitude -scale โดยค่าแฟคเตอร์เท่ากับ 10^6 จะได้

$R_1 = 407\ K\Omega$, $R_2 = 6.026\ M\Omega$, $R_3 = \infty$

$R_4 = 407\ K\Omega$, $R_5 = \infty$

$C_1 = C_2 = 1\ uF$

จาก $\gamma = \frac{r_1}{r_2}$ เมื่อ $\gamma = 0.1$ กำหนดให้ $r_1 = 1K\Omega$, $r_2 = 10\ K\Omega$

$$R_C = 1.0638 K\Omega$$
$$R_D = 16.666 K\Omega$$

เมื่อ K= 1.0638 โดยใช้ gain enhancement คือ

$$K_1 = \frac{b_2 + 2b_0R_1^2 - b_1R_1}{1+\gamma}$$

$$= \frac{1 + (2 \times 0.407707 \times 0.4070201^2) - (-0.08620 \times 0.407707)}{1+0.1}$$

$$= 1.0638 \quad > \quad 1$$

ในเมื่อ K_1 ไม่สามารถเกิน 1 ได้ เราต้องทำการ scale สัมประสิทธิ์ b_2, b_1 และ b_0 ลงโดยแฟคเตอร์ k_1 ที่น้อยกว่า 1.0638 และการเพิ่ม K สอดคล้องกัน โดยกำหนด $k_1 = 1.0638$ ได้สัมประสิทธิ์ใหม่คือ

$$H(s) = \frac{Kk_1(b_2/k_1)s^2 + (b_1/k_1)s + b_0/k_1}{s^2 + a_1s + a_0}$$

$b_2 = 0.94$, $b_1 = -0.081032$, $b_0 = 0.248905$

$k = 1.0638$, $a_1 = 0.086202$, $a_0 = 0.407707$

$$R_3 = \frac{(1+\gamma)(b_2 - k_3)}{R_1a_0\left[\left(b_0/a_0\right) - b_2\right]}$$

$$= \frac{(1+0.1)(0.94 - K_3)}{R_1a_0(0.2489/0.407707 - 0.94)}$$

$$= \infty$$

$$R_2 = \frac{R_3}{R_1R_3a_0 + \gamma} = \frac{1}{R_1a_0}$$

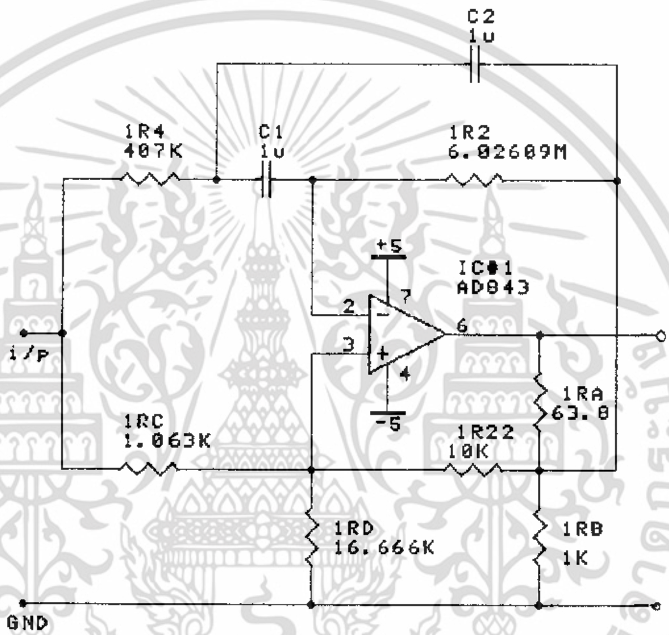
$$= \frac{1}{(0.407020 \times 0.407707)}$$

$$= 6.02609\ \Omega$$

เมื่อ $K_1 = 1$ และ $K_2 = b_2 = 0.94$ ได้

$$K = 1 + \frac{R_a}{R_b} = 1.0638$$

กำหนดให้ $R_b = 1\ K\Omega$ จะได้ $R_a = 63.8\ \Omega$



วงจรที่ได้จากการคำนวณของทรานสเฟอร์ฟังก์ชั่นชุดที่ 1

การหาคำนวณหาค่าความต้านทานและค่าตัวเก็บประจุของวงจรชุดที่ 2 จาก $H_2(s)$

$$H_2(s) = \left(\frac{s^2 - 0.177614s + 1.41445}{s^2 + 0.177614s + 1.41445}\right)$$

K=1, $b_2 = 1$, $b_1 = -0.177614$, $b_0 = 1.41445$, $a_1 = 0.177614$, $a_0 = 1.41445$

กำหนดให้ $C_1 = C_2 = 1F$, $\gamma = 0.1$, $0 \leq K_3 \leq 1$

จากสมการที่ (13) จะได้

$$R1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}}$$

$$R1 = \frac{2 * 0.1}{-0.177614 + \sqrt{(0.177614)^2 + (8 \times 1.41445 \times 0.1)}}$$

$$= \frac{0.2}{0.900860262}$$

$$= 0.2220100014 \ \Omega$$

$$K_1 = \frac{b_2 + 2b_0 R_1^2 - b_1 R_1}{1+\gamma}$$

$$= \frac{1 + (2 \times 1.41445 \times 0222010014^2) - (-0.177614 \times 0.222010014)}{1+0.1}$$

$$= 1.071694703 \quad > \quad 1$$

ในเมื่อ K_1 ไม่สามารถเกิน 1 ได้ เราต้องทำการ scale สัมประสิทธิ์ b_2, b_1 และ b_0 ลงโดยแฟคเตอร์ k_1 ที่น้อยกว่า 1.071694703 และการเพิ่ม K สอดคล้องกัน โดยกำหนด k_1 = 1.071694703 ได้สัมประสิทธิ์ใหม่คือ

$$H(s) = \frac{Kk_1(b_2 / k_1)s^2 + (b_1 / k_1)s + b_0 / k_1}{s^2 + a_1 s + a_0}$$

b_2 = 0.93310156 , b_1 = -0.1657319 , b_0 = 1.319825503

k = 1.071694703 , a_1 = 0.177614 , a_0 = 1.41445

$$R_3 = \frac{(1+\gamma)(b_2 - k_3)}{R_1 a_0 \left[(b_0 / a_0) - b_2 \right]}$$

$$= \frac{(1+0.1)(0.93310156 - K_3)}{R_1 a_0 (1.319825503 / 1.41445 - 0.93310156)}$$

$$= \infty$$

$$R_2 = \frac{R_3}{R_1 R_3 a_0 + \gamma} \qquad = \frac{1}{R_1 a_0}$$

$$= \frac{1}{(0.222010014 \times 1.41445)}$$

$$= 3.184489607\ \Omega$$

เมื่อ $K_1 = 1$ และ $K_2 = b_2 = 0.93310156$ ได้

$$R_4 = \frac{R_1}{K_1} = 0.22210014\ \Omega$$

$$R_5 = \frac{R_1}{1-K_1} = \infty$$

$$R_D = \frac{r_1}{K_2}$$

$$= \frac{r_1}{0.93310156}$$

$$= 1.071694704\ r_1$$

$$R_D = \frac{r_1}{1-K_2} = \frac{r_1}{1-0.93310156}$$

$$= 14.94803167 r_1$$

$$R_6 = \frac{R_3}{K_3} = \infty$$

$$R_7 = \frac{R_3}{1-K_3} = \infty$$

ทำการ scale ค่า RC ใช้ magnitude -scale โดยค่าแฟคเตอร์เท่ากับ 10^6 จะได้

$R_1 = 222.010\ K\Omega$, $R_2 = 3.1844896\ M\Omega$, $R_3 = \infty$

$R_4 = 222.010\ K\Omega$; $R_5 = \infty$

$C_1 = C_2 = 1\ uF$

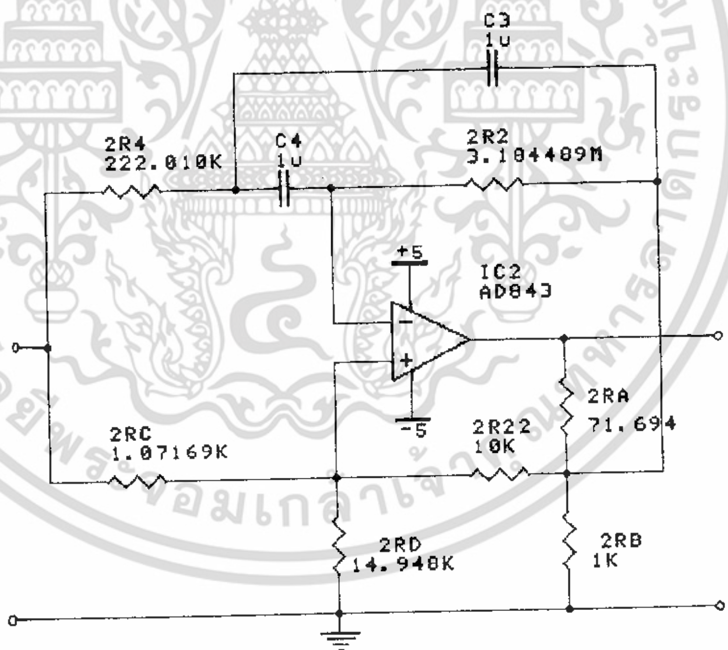
จาก $\gamma = \frac{r_1}{r_2}$ เมื่อ $\gamma = 0.1$ กำหนดให้ $r_1 = 1K\Omega$, $r_2 = 10\ K\Omega$

$$R_C = 1.07169 K\Omega$$
$$R_D = 14.948 K\Omega$$

เมื่อ K= 1.071694703 โดยใช้ gain enhancement คือ

$$K = 1 + \frac{R_a}{R_b} = 1.071694703$$

กำหนดให้ $R_b = 1\ K\Omega$ จะได้ $R_a = 71.694\ \Omega$



วงจรที่ได้จากการคำนวณของทรานสเฟอร์ฟังก์ชั่นชุดที่ 2

คำนวณหาค่าความต้านทานและค่าตัวเก็บประจุของวงจรชุดที่ 3 จาก $H_3(s)$

$$H_3(s) = \left(\frac{s^2 - 0.16537s + 2.90973}{s^2 + 0.16537s + 2.90973}\right)$$

K=1, b_2= 1, b_1=-0.16537, b_0=2.90973, a_1=0.16537, a_0=2.90973

กำหนดให้ $C_1 = C_2 = 1F$, $\gamma = 0.1$, $0 \le K_3 \le 1$

จากสมการที่ (13) จะได้

$$R1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}}$$

$$R1 = \frac{2 * 0.1}{-0.16537 + \sqrt{(0.16537)^2 + (8 \times 2.90973 \times 0.1)}}$$

$$= \frac{0.2}{1.369273}$$

$$= 0.146014 \ \Omega$$

$$K_1 = \frac{b_2 + 2b_0R_1^2 - b_1R_1}{1+\gamma}$$

$$= \frac{1 + (2 \times 2.90973 \times 0.146014^2) - (-0.16537 \times 0.146014)}{1 + 0.1}$$

$$= 1.0438 > 1$$

ในเมื่อ K_1 ไม่สามารถเกิน 1 ได้ เราต้องทำการ scale สัมประสิทธิ์ b_2 , b_1 และ b_0 ลงโดยแฟคเตอร์ k_1 ที่น้อยกว่า 1.0438 และการเพิ่ม K สอดคล้องกัน โดยกำหนด k_1 = 1.0438 ได้สัมประสิทธิ์ใหม่คือ

$$H(s) = \frac{Kk_1(b_2 / k_1)s^2 + (b_1 / k_1)s + b_0 / k_1}{s^2 + a_1 s + a_0}$$

b_2 = 0.958 , b_1 = -0.158430 , b_0 = 2.787631

k = 1.0438 , a_1 = 0.16537 , a_0 = 2.90973

$$R_3 = \frac{(1+\gamma)(b_2 - k_3)}{R_1 a_0\left[\left(b_0 / a_0\right) - b_2\right]}$$

$$= \frac{(1+0.1)(0.958 - K_3)}{R_1 a_0((2.787631 / 2.90973) - 0.958)}$$

$$= \infty$$

$$R_2 = \frac{R_3}{R_1 R_3 a_0 + \gamma} = \frac{1}{R_1 a_0}$$

$$= \frac{1}{(0.146014 \times 2.90973)}$$

$$= 2.35370\ \Omega$$

เมื่อ $K_1 = 1$ และ $K_2 = b_2 = 0.958$ ได้

$$R_4 = \frac{R_1}{K_1} = 0.146014\ \Omega$$

$$R_5 = \frac{R_1}{1-K_1} = \infty$$

$$R_C = \frac{r_1}{K_2}$$

$$= \frac{r_1}{0.958}$$

$$= 1.0438\ r_1$$

$$R_D = \frac{r_1}{1-K_2} = \frac{r_1}{1-0.958}$$

$$= 23.809\ r_1$$

$$R_6 = \frac{R_3}{K_3} = \infty$$

$$R_7 = \frac{R_3}{1-K_3} = \infty$$

ทำการ scale ค่า RC ใช้ magnitude -scale โดยค่าแฟคเตอร์เท่ากับ 10^6 จะได้

$R_1 = 146.014\ K\Omega$, $R_2 = 2.35370\ M\Omega$, $R_3 = \infty$

$R_4 = 146.014\ K\Omega$, $R_5 = \infty$

$C_1 = C_2 = 1\ uF$

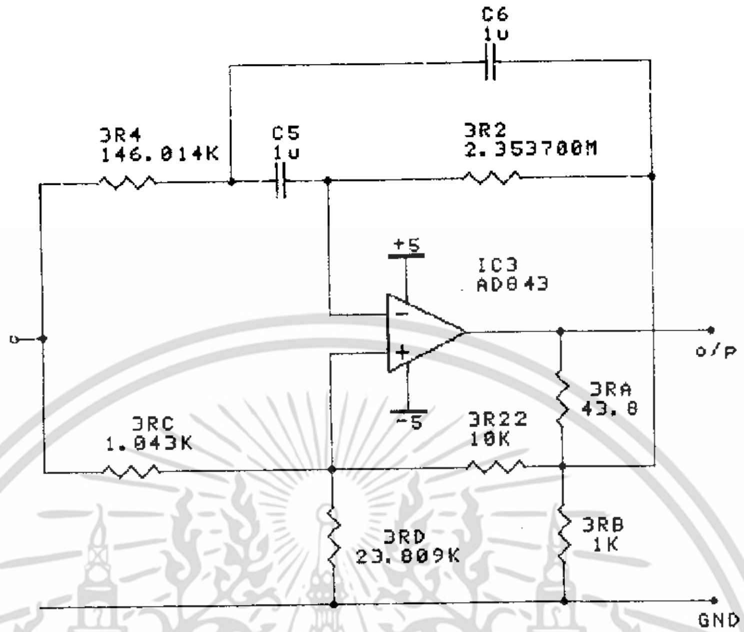
จาก $\gamma = \frac{r_1}{r_2}$ เมื่อ $\gamma = 0.1$ กำหนดให้ $r_1 = 1K\Omega$, $r_2 = 10\ K\Omega$

$$R_C = 1.0438K\Omega$$
$$R_D = 23.809K\Omega$$

เมื่อ K= 1.0438 โดยใช้ gain enhancement คือ

$$K = 1+\frac{R_a}{R_b} = 1.0438$$

กำหนดให้ $R_b = 1\ K\Omega$ จะได้ $R_a = 43.8\ \Omega$



วงจรที่ได้จากการคำนวณของทรานสเฟอร์ฟังก์ชั่นชุดที่ 3

การออกแบบการคำนวณทางด้านลบ

$$\omega_i = \omega_{d0}\left(1-\sqrt{1-\frac{i}{n-1}}\right)$$

$$T_{d0} = \frac{(n-1)\pi}{\omega_{d0}}$$

ในกรณีหาค่า k ได้จากสมการ (24) และ (25)

กำหนดค่า ω_{d0} = 4.3

n = 6

จะได้

T_{d0} = 3.5458

ω_1 = 0.4677

ω_2 = 0.9985

ω_3 = 1.6282

ω_4 = 2.4488

ω_5 = 4.43

ที่ k ที่เหมาะสมคือ สัมประสิทธิ์โพลิโนเมียลได้

a_6 = 1

a_5 = 0.15029

a_4 = 22.4947

a_3 = 1.05114

a_2 = 56.9

a_1 = 0.8936

a_0 = 1.3801

มาตรฐาน All Pass filter

$$H(s) = \frac{Pn(-s)}{Pn(s)}$$

n=6

$$pn(-s) = a_6s^6 - a_5s^5 + a_4s^4 - a_3s^3 + a_2s^2 - a_1s + a_0$$

$$pn(s) = a_6s^6 + a_5s^5 + a_4s^4 + a_3s^3 + a_2s^2 + a_1s + a_0$$

ค่า k = 0.15029 จะเข้าใกล้ slope delay มากที่สุด

$$H(s) = \frac{s^6 - 0.15029s^5 + 22.494s^4 - 1.051s^3 + 56.9s^2 - 0.8986s + 11.3801}{s^6 + 0.15029s^5 + 22.494s^4 + 1.051s^3 + 56.9s^2 + 0.8986s + 11.3801}$$

การออกแบบวงจรทางด้านสโลปลบก็จะมีวิธีการออกแบบเช่นเดียวกันกับการออกแบบสโลปบวก คือขั้นแรกจะทำการถอดรากจากทรานเฟอร์ฟังก์ชั่นซึ่งมีค่า k = 0.15029 โดยใช้โปรแกรมก่อน แล้วจึงนำค่ารากที่ได้ไปทำการ iterate เพื่อให้ได้ค่า ω ที่ใกล้เคียงกับเส้นของสโลปเชิงเส้นมากที่สุด ซึ่งผลการ RUN โปรแกรม มีดังต่อไปนี้

ผลการถอดรากฟังก์ชั่นโดยใช้โปรแกรม

```
Order of Polynomial = 6
 Enter Coefficient of equation ( Max to min order)
a1 = 1
a2 = 0.15029
a3 = 22.494
a4 = 1.051
a5 = 56.9
a6 = 0.8986
a7 = 11.3801
 Admissible Error = 0.0000000000001
 max. iteration = 200
Root1  -7.1607305886E-03    4.6768038706E-01i
Iteration = 6
Root2  -7.1607305886E-03   -4.6768038706E-01i
Iteration = 6
Root3  -1.0071011955E-02    1.6282414409E+00i
Iteration = 5
Root4  -1.0071011955E-02   -1.6282414409E+00i
Iteration = 5
Root5  -5.7913257456E-02    4.4290312677E+00i
Iteration = 2
Root6  -5.7913257456E-02   -4.4290312677E+00i
Iteration = 2
```

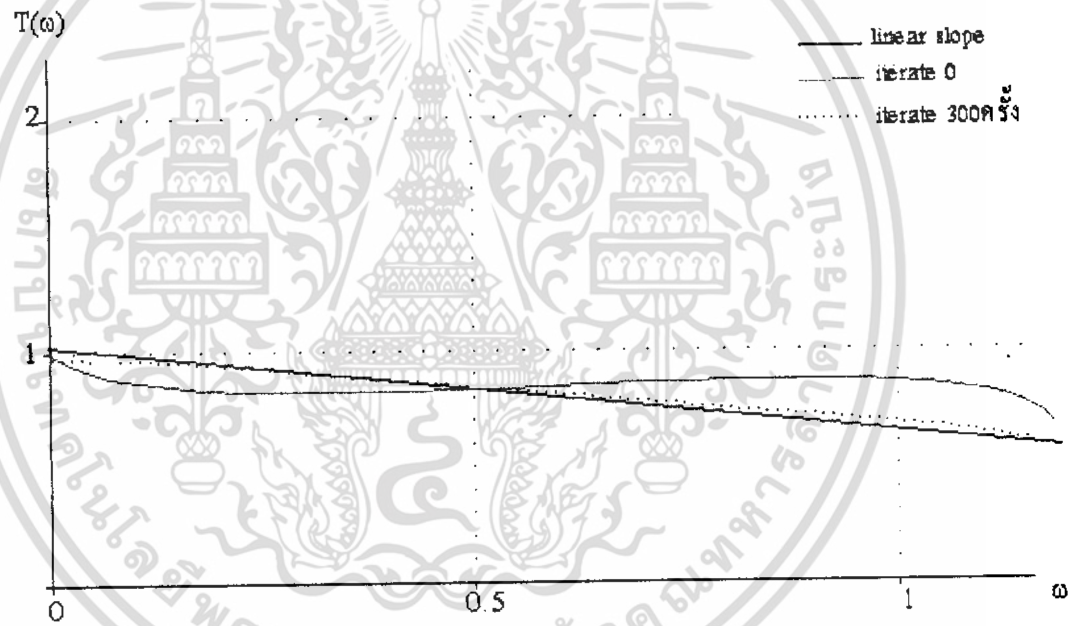
ผลจากการ iterate ค่ารากที่ถอดได้เป็นจำนวน 0 และ 300 ครั้ง

```
                     Iterate 0

a1 =  0.0071607306                b1 =  0.4678038706
a2 =  0.0100710120                b2 =  1.6282414409
a3 =  0.0579132575                b3 =  4.4290312677

w 1 =   0.0000000000E+00          delta D =   6.9743992574E-01
w 2 =   4.6780387060E-01          delta D =  -4.3662792392E+01
w 3 =   1.0480226557E+00          delta D =   8.6940637842E-01
w 4 =   1.6282414409E+00          delta D =  -3.0630203147E+01
w 5 =   3.0286363543E+00          delta D =   1.1925541299E+00
w 6 =   4.4290312677E+00          delta D =  -4.0698271732E+00
```

Iterate 300

```
a1 =  0.9462580000                  b1 =  0.3253000000
a2 =  1.0377364000                  b2 =  1.2081141600
a3 =  0.9661633000                  b3 =  2.2035115000

w 1 =   0.0000000000E+00            delta D =  -2.4581851544E-01
w 2 =   3.2530000000E-01            delta D =  -1.7566940251E-01
w 3 =   7.6670708000E-01            delta D =  -3.6167686462E-02
w 4 =   1.2081141600E+00            delta D =   1.0343826241E-01
w 5 =   1.7058128300E+00            delta D =   2.6330475765E-01
w 6 =   2.2035115000E+00            delta D =   4.3355435243E-01
```

กราฟที่ทำการ iterate จำนวน 300 ครั้ง



จะได้ค่า ω ที่นำไปใช้ในการหาทรานสเฟอร์ฟังก์ชั่น

$\omega_0 = 0.0000000000$

$\omega_1 = 0.3253000000$

$\omega_2 = 0.7667070800$

$\omega_3 = 1.2081141600$

$\omega_4 = 1.7058128300$

$\omega_5 = 2.2035110000$

4. การออกแบบวงจรนี้เราจะใช้รูปแบบของ Single - Amplifier generation Biquad (SAB) ซึ่งเป็นรูปแบบของ 2^{nd} order ในลักษณะของ negative feedback RC amplifier network จากทรานสเฟอร์ฟังก์ชั่นที่ได้จะเห็นว่ามี 6^{th} order เราจะใช้ SAB ต่อในลักษณะ CASCADE ที่สโลปลบทรานสเฟอร์ฟังก์ชั่นที่ได้

$$H(s) = \frac{s^6 - 0.15029s^5 + 6.420822844s^4 - 0.525659888s^3 + 7.754995669s^2 - 0.257070229s + 0.749919624}{s^6 + 0.15029s^5 + 6.420822844s^4 + 0.525659888s^3 + 7.754995669s^2 + 0.257070229s + 0.749919624}$$

จากทรานสเฟอร์ฟังก์ชั่น H(s) order 6^{th} ของ Linear slope ด้านลบทำการออกแบบวงจรโดยการแยกแฟคเตอร์ออกเป็น H(s) order 2^{nd} 3 ชุด โดยใช้โปรแกรม ได้คือ

$$H(s) = H_1(s) \cdot H_2(s) \cdot H_3(s)$$

ซึ่งมีผลการ RUN โปรแกรมดังนี้ คือ

```
The order of polynomial ?  6
a(0)?   0.749919624
a(1)?   0.257070229
a(2)?   7.754995669
a(3)?   0.525659888
a(4)?   6.420822844
a(5)?   0.15029
a(6)?   1
Tolerance ?   0.0000000000001
---------------------------------------------------------------
 p =0.0316344      q =0.105977
Quadratic coefficient = x^2 + (0.0316344 x) + (0.105977 )
Roots of the quadratic factor:
   -0.0158172 + 0.325157i
   -0.0158172 - 0.325157i
Coefficients of deflated polynomial
       order   coefficients
         0          7.07625
         1          0.31344
         2          6.31109
         3          0.11866
         4          1.00000
---------------------------------------------------------------
Type 1 to continue, or 0 to stop.
```

จากผลการ Run โปรแกรม เราสามารถแยกแฟคเตอร์ออกเป็น 3 วงเล็บดังนี้ คือ

$$H(s) = \left(\frac{s^2 - 0.0316344s + 0.105977}{s^2 + 0.0316344s + 0.105977}\right) \bullet \left(\frac{s^2 - 0.0413868s + 1.45951}{s^2 + 0.0413868s + 1.45951}\right) \bullet \left(\frac{s^2 - 0.07727s + 4.84839}{s^2 + 0.07727s + 4.84839}\right)$$

จะเห็นว่าการแยกแฟคเตอร์ของทรานสเฟอร์ฟังก์ชั่นมีวัตถุประสงค์เพื่อความสะดวกในการหาค่าอุปกรณ์ของวงจรจากทรานสเฟอร์ฟังก์ชั่นย่อยในแต่ละชุดนั่นเอง

การออกแบบวงจร Linear slope จากทรานสเฟอร์ฟังก์ชั่นทางด้านลบ

การหาคำนวณหาค่าความต้านทานและค่าตัวเก็บประจุของวงจรชุดที่ 1 จาก $H_1(s)$

$$H_1(s) = \left(\frac{s^2 - 0.0316344 + 0.105977}{s^2 + 0.0316344 + 0.105977}\right)$$

K=1, b_2= 1, b_1=-0.0316344, b_0=0.105977, a_1=0.031344, a_0=0.105977

กำหนดให้ $C_1 = C_2 = 1F$, $\gamma = 0.1$, $0 \le K_3 \le 1$

จากสมการที่ (13) จะได้

$$R1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}}$$

$$R1 = \frac{2 * 0.1}{-0.0316344 + \sqrt{0.031644^2 + (8 \times 0.105977 \times 0.1)}}$$

$$= \frac{0.2}{0.261252852}$$

$$= 0.765541882 \ \Omega$$

$$K_1 = \frac{b_2 + 2b_0R_1^2 - b_1R_1}{1+\gamma}$$

$$= \frac{1 + (2 \times 0.105977 \times 0.765541882^2) - (-0.0316344 \times 0.065541882)}{1 + 0.1}$$

$= 1.044060934 \quad > \quad 1$

ในเมื่อ K_1 ไม่สามารถเกิน 1 ได้ เราต้องทำการ scale สัมประสิทธิ์ b_2, b_1 และ b_0 ลงโดยแฟคเตอร์ k_1 ที่น้อยกว่า 1.044060934 และการเพิ่ม K สอดคล้องกัน โดยกำหนด $k_1 = 1.044060934$ ได้สัมประสิทธิ์ใหม่คือ

$$H(s) = \frac{Kk_1(b_2/k_1)s^2 + (b_1/k_1)s + b_0/k_1}{s^2 + a_1 s + a_0}$$

$b_2 = 0.957826025$, $b_1 = -0.030300251$, $b_0 = 0.101507528$

$k = 1.044030934$, $a_1 = 0.0316344$, $a_0 = 0.105977$

$$R_3 = \frac{(1+\gamma)(b_2 - k_3)}{R_1 a_0\left[\left(b_0/a_0\right) - b_2\right]}$$

$$= \frac{(1+0.1)(0.957826025 - K_3)}{R_1 a_0(0.101507528/0.105977 - 0.957826025)}$$

$$= \infty$$

$$R_2 = \frac{R_3}{R_1 R_3 a_0 + \gamma} = \frac{1}{R_1 a_0}$$

$$= \frac{1}{(0.765541882 \times 0.105977)}$$

$$= 12.31592223\ \Omega$$

เมื่อ $K_1 = 1$ และ $K_2 = b_2 = 0.957826025$ ได้

$$R_4 = \frac{R_1}{K_1} = 0.765541882\ \Omega$$

$$R_5 = \frac{R_1}{1-K_1} = \infty$$

$$R_C = \frac{r_1}{K_2}$$

$$= \frac{r_1}{0.957826025}$$

$$= 1.044030935\ r_1$$

$$R_D = \frac{r_1}{1-K_2}$$

$$= 23.71130537 r_1$$

$$R_6 = \frac{R_3}{K_3} = \infty$$

$$R_7 = \frac{R_3}{1-K_3} = \infty$$

ทำการ scale ค่า RC ใช้ magnitude -scale โดยค่าแฟคเตอร์เท่ากับ 10^6 จะได้

$R_1 = 765.541\ K\Omega$, $R_2 = 12.32592\ M\Omega$, $R_3 = \infty$

$R_4 = 765.541\ K\Omega$, $R_5 = \infty$

$C_1 = C_2 = 1\ uF$

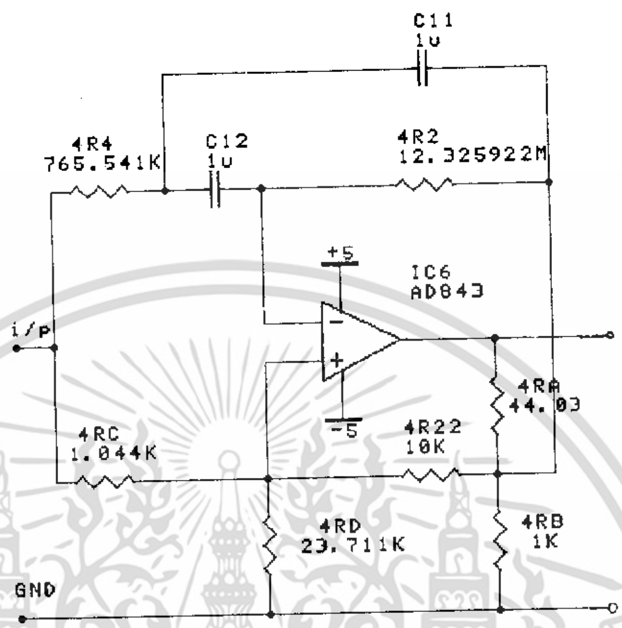
จาก $\gamma = \frac{r_1}{r_2}$ เมื่อ $\gamma = 0.1$ กำหนดให้ $r_1 = 1K\Omega$, $r_2 = 10\ K\Omega$

$$R_C = 1.04403 K\Omega$$
$$R_D = 23.7113 K\Omega$$

เมื่อ K= 1.044030934 โดยใช้ gain enhancement คือ

$$K = 1 + \frac{R_a}{R_b} = 1.044030934$$

กำหนดให้ R_b = 1 KΩ จะได้ R_a = 44.03 Ω



วงจรที่ได้จากการคำนวณของทรานสเฟอร์ฟังก์ชั่นชุดที่ 1

การหาคำนวณหาค่าความต้านทานและค่าตัวเก็บประจุของวงจรชุดที่ 2 จาก $H_2(s)$

$$H_2(s) = \left(\frac{s^2 - 0.01431868s + 1.45951}{s^2 + 0.01431868s + 1.45951}\right)$$

K=1, b_2= 1, b_1=-0.0413868, b_0=1.45951, a_1=0.0413868, a_0=1.45951

กำหนดให้ $C_1 = C_2 = 1F$, $\gamma = 0.1$, $0 \leq K_3 \leq 1$

จากสมการที่ (13) จะได้

$$R1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}}$$

$$R1 = \frac{2 * 0.1}{-0.0413868 + \sqrt{(0.0413868)^2 + (8 \times 1.45951 \times 0.1)}}$$

$$= \frac{0.2}{1.039964608}$$

$$= 0.192314236\ \Omega$$

$$K_1 \quad = \quad \frac{b_2 + 2b_0R_1^2 - b_1R_1}{1+\gamma}$$

$$= \frac{1 + (2\times 1.45951 \times 0.192314236^2) - (-0.0413868 \times 0.192314236)}{1+0.1}$$

$$= 1.01447401 \quad > \quad 1$$

ในเมื่อ K_1 ไม่สามารถเกิน 1 ได้ เราต้องทำการ scale สัมประสิทธิ์ b_2 , b_1 และ b_0 ลงโดยแฟคเตอร์ k_1 ที่น้อยกว่า 1.01447401 และการเพิ่ม K สอดคล้องกัน โดยกำหนด k_1 = 1.01447401 ได้สัมประสิทธิ์ใหม่คือ

$$H(s) \quad = \frac{Kk_1(b_2/k_1)s^2 + (b_1/k_1)s + b_0/k_1}{s^2 + a_1s + a_0}$$

b_2 = 0.985732498 , b_1 = -0.040796313 , b_0 = 1.438691983

k = 1.0144701 , a_1 = 0.0413868 , a_0 = 1.45951

$$R_3 \quad = \frac{(1+\gamma)(b_2 - k_3)}{R_1a_0\left[\left(b_0/a_0\right) - b_2\right]}$$

$$= \frac{(1+0.1)(0.93310156 - K_3)}{R_1a_0(1.4386919831/1.45951 - 0.985732498)}$$

$$= \infty$$

$$R_2 \quad = \frac{R_3}{R_1R_3a_0 + \gamma} \qquad = \frac{1}{R_1a_0}$$

$$= \frac{1}{(0.192314236 \times 1.45691)}$$

$= 3.563718354\ \Omega$

เมื่อ $K_1 = 1$ และ $K_2 = b_2 = 0.985732498$ ได้

$$R_4 = \frac{R_1}{K_1} = 0.192314236\ \Omega$$

$$R_5 = \frac{R_1}{1-K_1} = \infty$$

$$R_C = \frac{r_1}{K_2}$$

$$= \frac{r_1}{0.985732498}$$

$$= 1.01447401\ r_1$$

$$R_D = \frac{r_1}{1-K_2} = \frac{r_1}{1-0.985732498}$$

$$= 70.08935411 r_1$$

$$R_6 = \frac{R_3}{K_3} = \infty$$

$$R_7 = \frac{R_3}{1-K_3} = \infty$$

ทำการ scale ค่า RC ใช้ magnitude -scale โดยค่าแฟคเตอร์เท่ากับ 10^6 จะได้

$R_1 = 192.3142\ K\Omega$, $R_2 = 3.5627183\ M\Omega$, $R_3 = \infty$

$R_4 = 192.3142\ K\Omega$, $R_5 = \infty$

$C_1 = C_2 = 1\ uF$

จาก $\gamma = \frac{r_1}{r_2}$ เมื่อ $\gamma = 0.1$ กำหนดให้ $r_1 = 1K\Omega$, $r_2 = 10\ K\Omega$

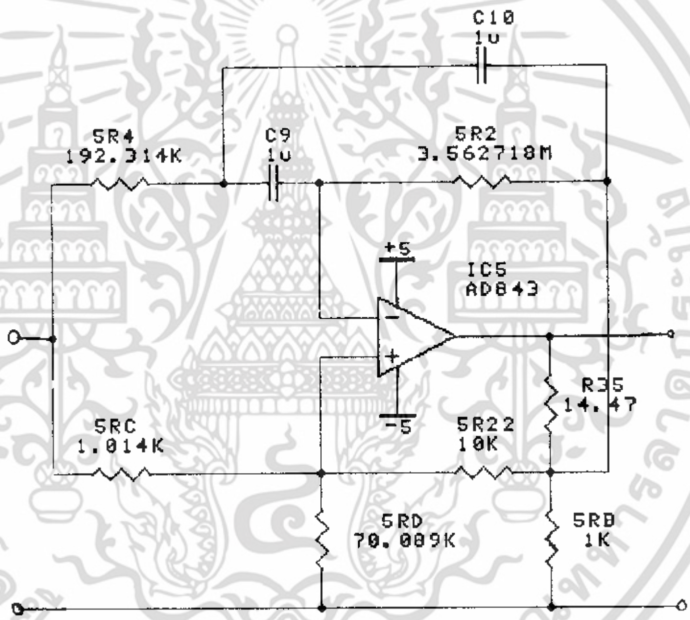
$R_C = 1.01447\Omega$

$R_D = 70.08935K\Omega$

เมื่อ K= 1.01447101 โดยใช้ gain enhancement คือ

$$K = 1+\frac{R_a}{R_b} = 1.01447401$$

กำหนดให้ R_b = 1 KΩ จะได้ R_a = 14.474 Ω



วงจรที่ได้จากการคำนวณของทรานสเฟอร์ฟังก์ชั่นชุดที่ 2

การหาคำนวณหาค่าความต้านทานและค่าตัวเก็บประจุของวงจรชุดที่ 3 จาก $H_3(s)$

$$H_3(s) = \left(\frac{s^2 - 0.07727s + 4.8459}{s^2 + 0.07727s + 4.8459}\right)$$

K=1, b_2= 1, b_1=-0.07727, b_0=4.84839, a_1=0.07727, a_0=4.84839

กำหนดให้ $C_1 = C_2 = 1F$, $\gamma = 0.1$, $0 \leq K_3 \leq 1$
จากสมการที่ (13) จะได้

$$R1 = \frac{2\gamma}{-a_1 + \sqrt{a_1^2 + 8a_0\gamma}}$$

$$R1 = \frac{2 * 0.1}{-0.07727 + \sqrt{(0.07727)^2 + (8 \times 4.84839 \times 0.1)}}$$

$$= \frac{0.2}{1.89368983}$$

$$= 0.105613916 \ \Omega$$

$$K_1 = \frac{b_2 + 2b_0R_1^2 - b_1R_1}{1+\gamma}$$

$$= \frac{1 + (2 \times 4.84839 \times 0.105613916^2) - (-0.7727 \times 0.105613916)}{1 + 0.1}$$

$$= 1.014837794 \quad > \quad 1$$

ในเมื่อ K_1 ไม่สามารถเกิน 1 ได้ เราต้องทำการ scale สัมประสิทธิ์ b_2 , b_1 และ b_0 ลงโดยแฟคเตอร์ k_1 ที่น้อยกว่า 1.014837794 และการเพิ่ม K สอดคล้องกัน โดยกำหนด k_1 = 1.014837794 ได้สัมประสิทธิ์ใหม่คือ

$$H(s) = \frac{Kk_1(b_2 / k_1)s^2 + (b_1 / k_1)s + b_0 / k_1}{s^2 + a_1 s + a_0}$$

b_2 = 0.985379147 , b_1 = -0.076140246 , b_0 = 4.777413719

k = 1.014837794 , a_1 = 0.07727 , a_0 = 4.84839

$$R_3 = \frac{(1+\gamma)(b_2 - k_3)}{R_1 a_0\left[\left(b_0 / a_0\right) - b_2\right]}$$

$$= \frac{(1+0.1)(0.986379147 - K_3)}{R_1 a_0((4.777413719 / 4.84839) - 0.985379147)}$$

$$= \infty$$

$$R_2 = \frac{R_3}{R_1 R_3 a_0 + \gamma} = \frac{1}{R_1 a_0}$$

$$= \frac{1}{(0.105613916 \times 4.8483)}$$

$$= 1.320962882\ \Omega$$

เมื่อ $K_1 = 1$ และ $K_2 = b_2 = 0.985379147$ ได้

$$R_4 = \frac{R_1}{K_1} = 0.105613916\ \Omega$$

$$R_5 = \frac{R_1}{1 - K_1} = \infty$$

$$R_C = \frac{r_1}{K_2}$$

$$= \frac{r_1}{0.985379147}$$

$$= 1.014837794\ r_1$$

$$R_D = \frac{r_1}{1 - K_2} = \frac{r_1}{1 - 0.985379147}$$

$$= 68.39546229\ r_1$$

$$R_6 = \frac{R_3}{K_3} = \infty$$

$$R_7 = \frac{R_3}{1 - K_3} = \infty$$

ทำการ scale ค่า RC ใช้ magnitude -scale โดยค่าแฟคเตอร์เท่ากับ 10^6 จะได้

R_1 = 105.6139 KΩ , R_2 = 1.3209628 MΩ , $R_3 = \infty$

R_4 = 105.6139 KΩ , $R_5 = \infty$

$C_1 = C_2 =$ 1 uF

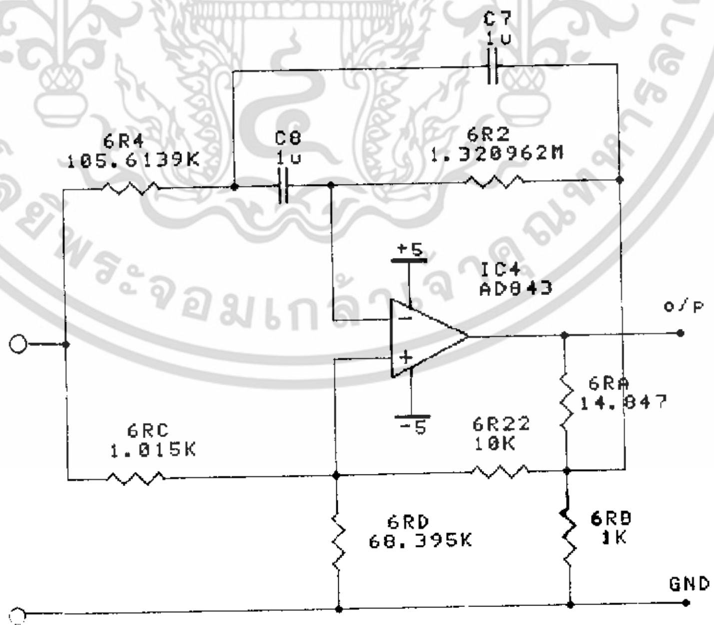
จาก $\gamma = \frac{r_1}{r_2}$ เมื่อ γ = 0.1 กำหนดให้ r_1 = 1KΩ , r_2 = 10 KΩ

$$R_C = 1.01483 K\Omega$$
$$R_D = 68.3954 K\Omega$$

เมื่อ K= 1.014837794 โดยใช้ gain enhancement คือ

$$K = 1 + \frac{R_a}{R_b} = 1.01483$$

กำหนดให้ R_b = 1 KΩ จะได้ R_a = 14.83 Ω



วงจรที่ได้จากการคำนวณของทรานสเฟอร์ฟังก์ชั่นชุดที่ 3

บทที่ 3

การทดลอง ผลการทดลองและบทสรุป

3.1 การทดลอง

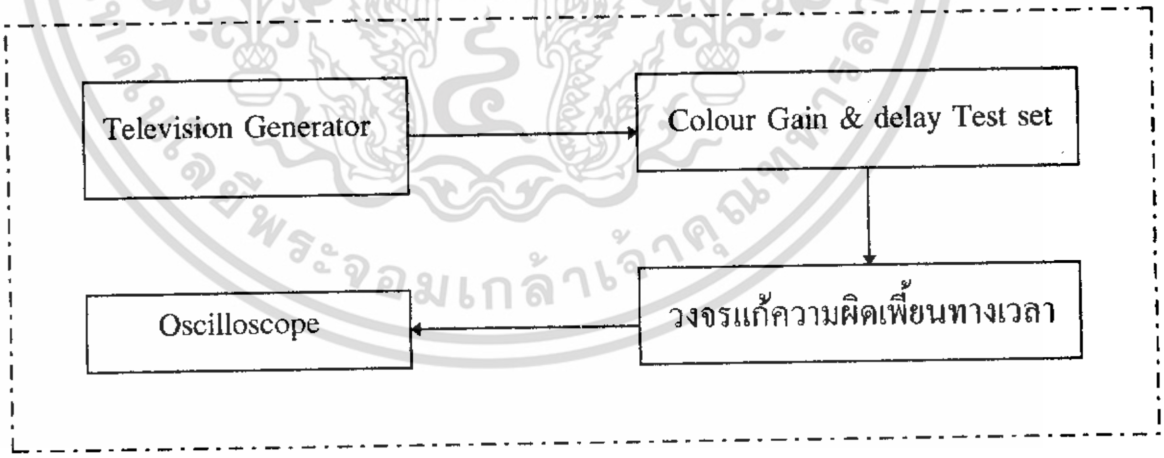
อุปกรณ์ในการทดลอง

1. Oscilloscope
2. Colour Gain & Delay Test set
3. PAL Television Generator
4. วงจรแก้ความผิดเพี้ยนทางเวลา

ขั้นตอนในการทดลอง

1.ต่อวงจรตามรูปเพื่อทำการสร้างสัญญาณทดสอบที่ใช้ในการกับวงจรแก้ไขดีเลย์สโลปเชิงเส้นทั้งด้านบวกและลบ

2. ตั้งค่าเครื่อง Colour Gain & Delay Test Set ให้เป็นสัญญาณ 20T ไซน์กำลังสองพัลส์เพื่อทำให้เกิดการผิดเพี้ยนทางเวลาทั้งนำหน้าและล้าหลังได้



รูปที่ 3.1 แสดงการวัดการตอบสนองการทำงานของวงจร

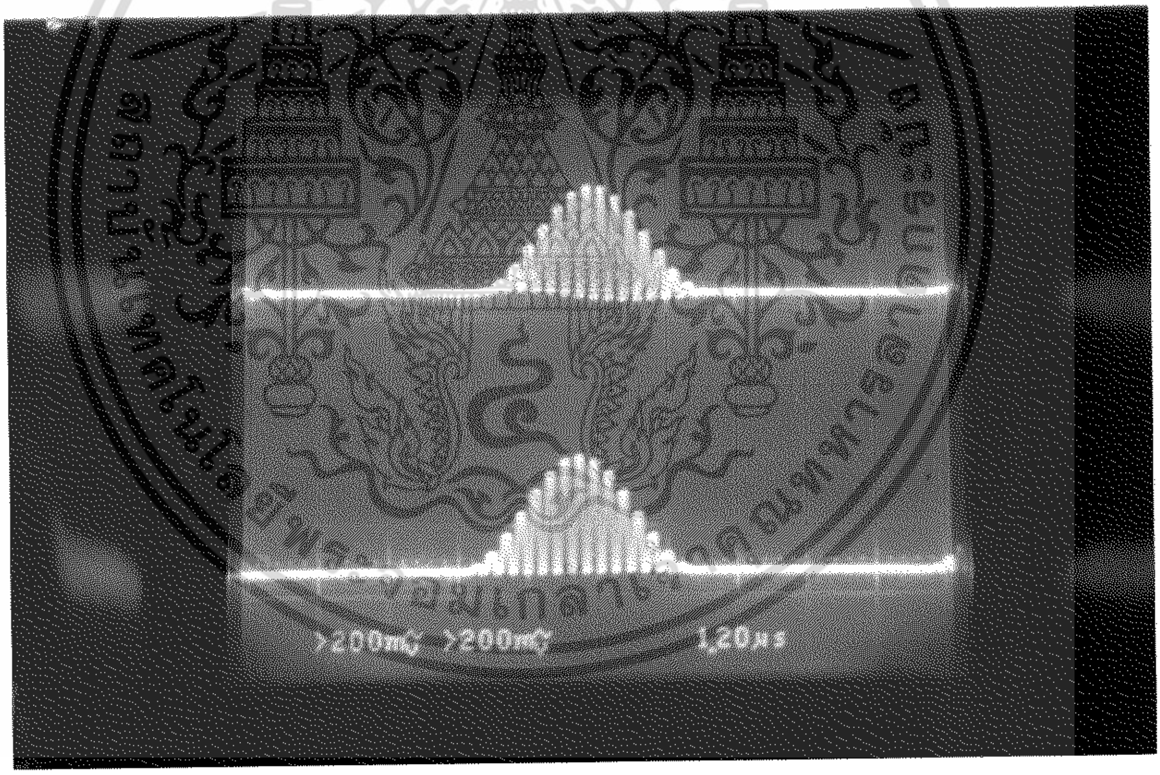
3. ถ้าสัญญาณที่ออกมาจากเครื่อง Colour Gain & Delay Test set มีฐานของสัญญาณไม่เรียบคือเกิดการเพี้ยนไม่ว่าจะเป็นแอมปลิจูดหรือ group delay เพื่อทำให้สัญญาณทดสอบโดยการปรับแต่งให้ฐานของสัญญาณมีความเรียบเพื่อนำไปใช้เป็นค่าอ้างอิงในการวัด

4. นำสัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์เข้าต่อเพื่อทดสอบวงจรแก้ความผิดเพี้ยนทางเวลาที่ ±60 ns

รูปที่ 3.2 สัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์ที่ปรับแต่งแล้ว

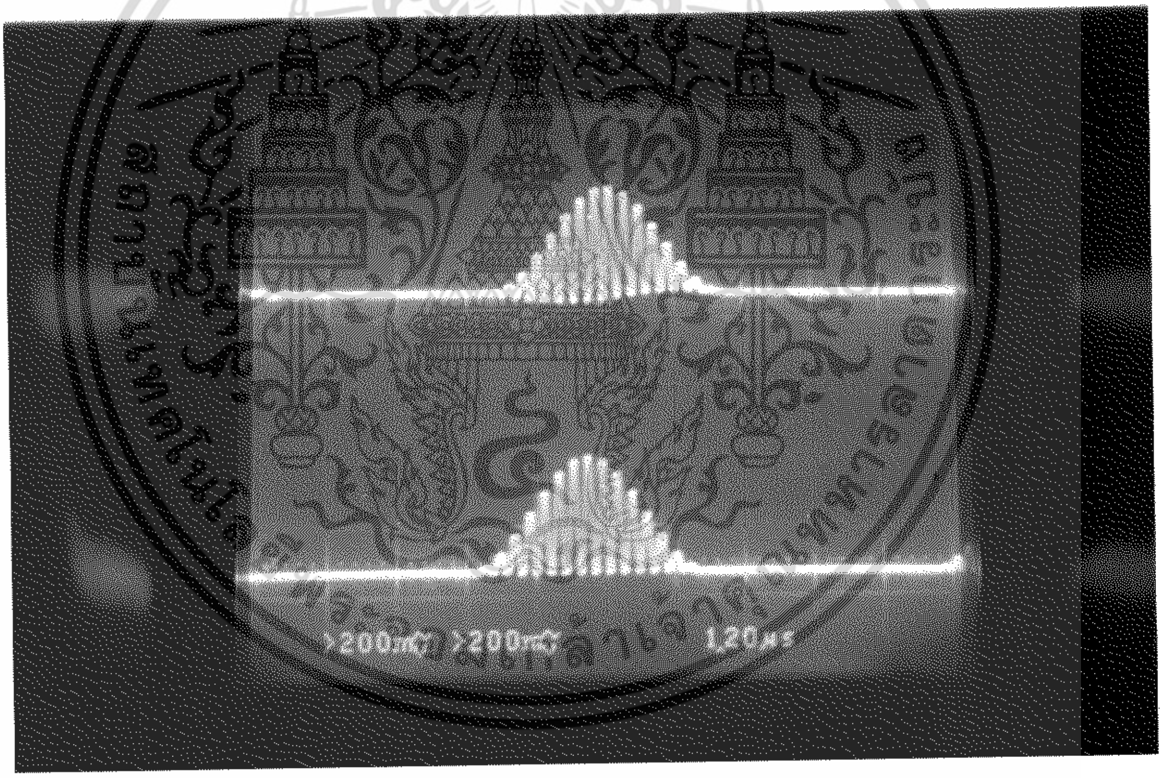
3.2 ผลการทดลอง

ผลการทดลองที่ใช้ทดสอบ 20T ไซน์กำลังสองพัลซ์ที่ล้าหลัง 60 ns ป้อนให้กับวงจรแก้ความผิดเพี้ยนทางเวลาสโลปบวกที่ได้ทำการออกแบบโดยการออปติไมซ์เซชั่นมาใช้แก้ปัญหาความผิดพลาดที่เกิดจากการออกแบบในโครงงานเดิม โดยใช้หลักการของ Numerical มาใช้ในการแก้ไขความผิดพลาดเนื่องจากการคำนวณ สำหรับสัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์ล้าหลัง 60 ns (สัญญาณรูปบน) และผลที่ได้จากการแก้ไขความผิดเพี้ยนทางเวลาทางด้านสโลปบวก



รูปที่ 3.3 สัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์ที่ล้าหลัง 60 ns และสัญญาณเอาต์พุตเมื่อผ่านวงจรแก้ความผิดเพี้ยนสโลปบวก

ผลการทดลองที่ใช้ทดสอบ 20T ไซน์กำลังสองพัลซ์ที่นำหน้า 60 ns ป้อนให้กับวงจรแก้ความผิดเพี้ยนทางเวลาสโลปลบที่ได้ทำการออกแบบโดยการออปติไมซ์เซชั่นมาใช้แก้ปัญหาความผิดพลาดที่เกิดจากการออกแบบในโครงงานเดิม โดยใช้หลักการของ Numerical มาใช้ในการแก้ไขความผิดพลาดเนื่องจากการคำนวณ สำหรับสัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์นำหน้า 60 ns (สัญญาณรูปบน) และผลที่ได้จากการแก้ไขความผิดเพี้ยนทางเวลาทางด้านสโลปลบ



รูปที่ 3.4 สัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์ที่นำหน้า 60 ns และสัญญาณเอาต์พุตเมื่อผ่านวงจรแก้ความผิดเพี้ยนสโลปลบ

.3.3 บทสรุป

จากการทดลองจะเห็นว่าสัญญาณที่เราใช้ทดสอบไม่ว่าสโลบบวกหรือลบเราต้องมีการตั้งค่า Gain และ Phase เพื่อชดเชยค่าความเพี้ยนของเครื่องมือทดสอบและการสูญเสียของสัญญาณในสายวัดก่อนเสมอเพื่อความถูกต้องในการทดสอบ และผลของสัญญาณทดสอบ 20T ไซน์กำลังสองพัลซ์ที่ล้าหลัง 60 ns และนำหน้า 60 ns จะเห็นว่าวงจรแก้ความผิดเพี้ยนทางเวลาสโลป บวกและสโลปลบที่ผ่านการออปติไมซ์เซชั่นมาใช้ในการออกแบบจะมีรูปสัญญาณที่ดีกว่าเดิมมากซึ่งทำให้เราทราบว่าการออปติไมซ์เซชั่นโดยใช้หลักการ Nummerical ช่วยลดค่าความผิดพลาดในการหาค่ารากของสมการซึ่งเป็นส่วนที่สำคัญที่สุดในการออกแบบวงจร ซึ่งเราสามารถปรับปรุงวงจรแก้ความผิดเพี้ยนทางเวลาทั้งในด้านสโลปบวกและสโลบลบมีความเป็นเชิงเส้นมากขึ้นจนใกล้เคียงเส้นในทางอุดมคติมาก โดยเราจะสามารถสังเกตจากเบสไลน์ของสัญญาณ 20T ไซน์กำลังสองพัลซ์นั่นเอง

บรรณานุกรม

1. K. Janchirapongvej and T. Ikeda , "Equalization of the relative Luminance to Chrominance Gain in Video Signal Transmission"Proc. of IL ITS Vol. 40 No. 8 1986

2. I.I. Trifonov ., "Synthesis of Low pass Filter with Near -monotonic Transient response" Telecommunication June 9,1963

3. C.A.Siocos., "Chrominance to Lumnance Ratio and timing Measurement " IEEE Trans BC-14,1 PP.1-4 March 1968

4. R.Kenmedy., "Sine Square Pulse in television System Analysis" RCA Review Vol.21 No.2 june 1968

5. Janez valand .,"On the linear Phase Approximation" Proceeding of IEEE Sep.1967 p.1627-1928

6. J.Valand., "Vriation of the linear Phase Approx Technical" Proc. of IEEE Letter March 1968 p.1141-1143

7. J.Valand., "On the Linear slope Delay approximate" Proc.IEEE Nov.1967 p.2059-2060

8. Wai-Kai chen., "Passive and Active filter" p.281-297

9. Siocos , C.A.,Chrominance to Luminance ratio and Timing Mesurment in Color Television ,IEEE Trans. Onbroadcasing ,vol. BC-14 NO.1 pp.1-4,1968

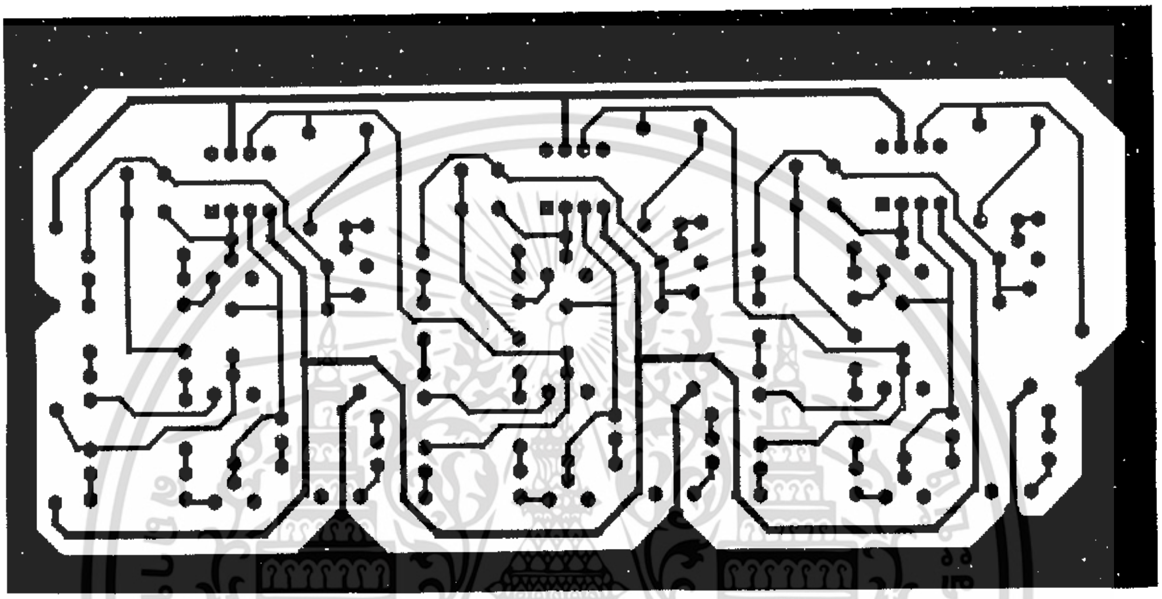
10. Hillburn,J.L. and johnson , D.E., Manual of Active Filter Design McGraw-Hill,1973

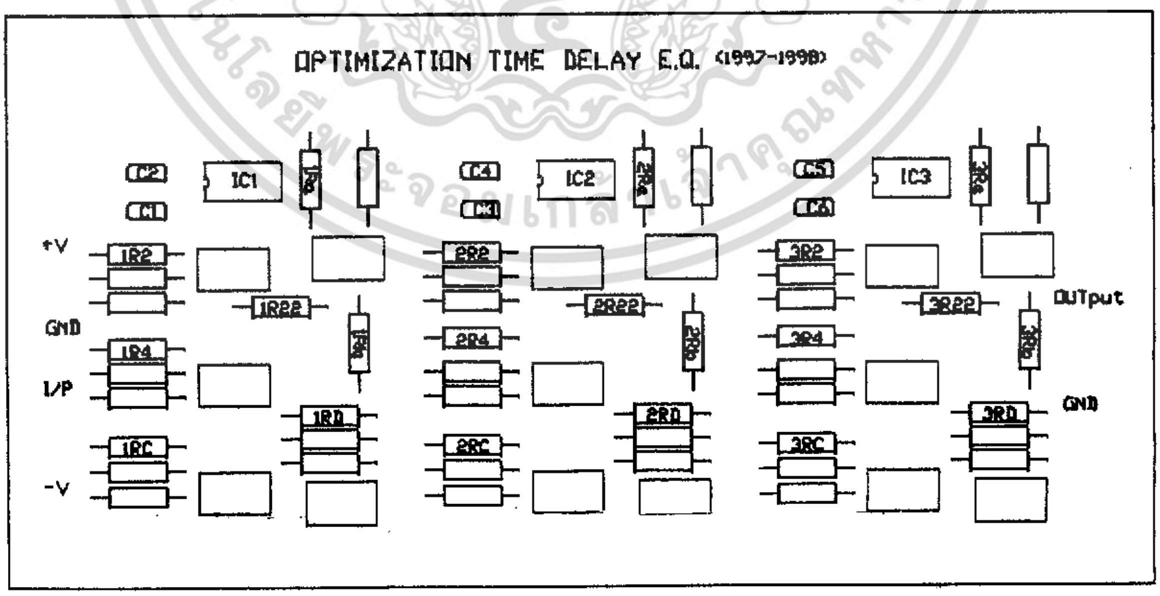
11. จินตนา ม่วงศรีจันทร์ และกนก เจนจิระพงศ์เวช., การวัดสัญญาณความผิดเพี้ยนเชิงเส้นโดยใช้ไซน์กำลังสองพัลส์.,การประชุมวิชาการวิศวกรรมไฟฟ้า ครั้งที่ 1 ณ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

12. กนก เจนจิระพงส์เวช .,การตรวจสอบสัญญาณวิดีโอ.,ตำราชุดวิศวกรรมศาสตร์ **คณะวิศกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง**

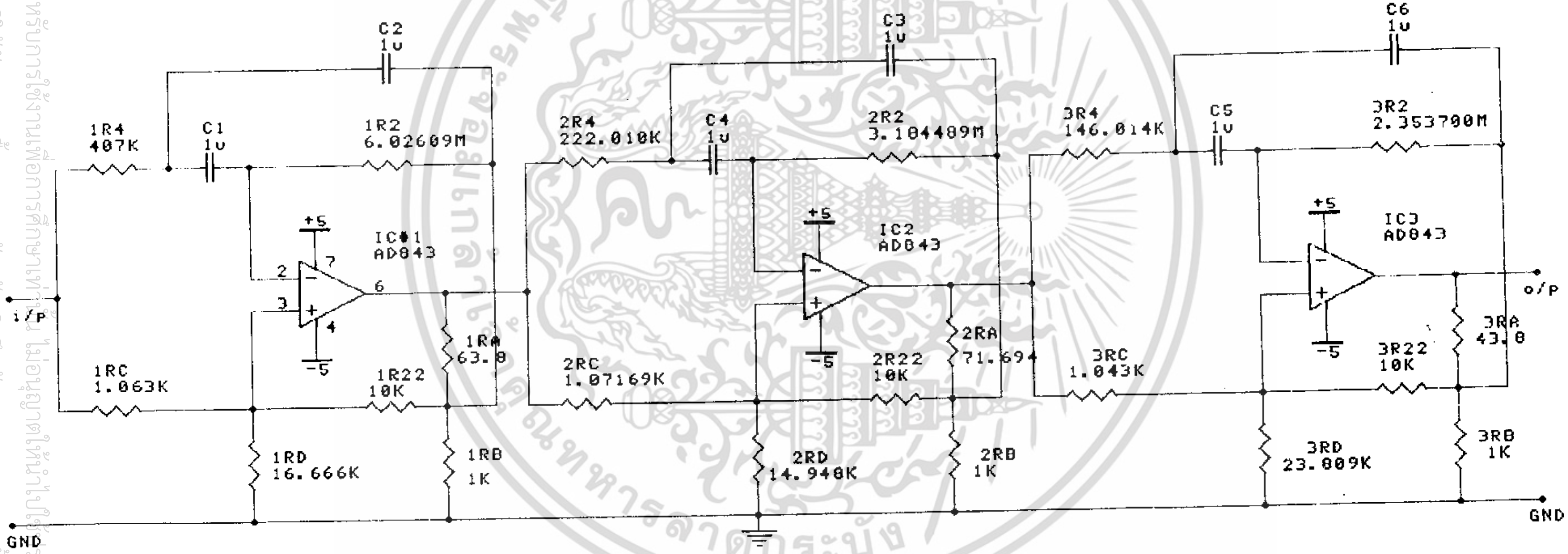
ภาคผนวก

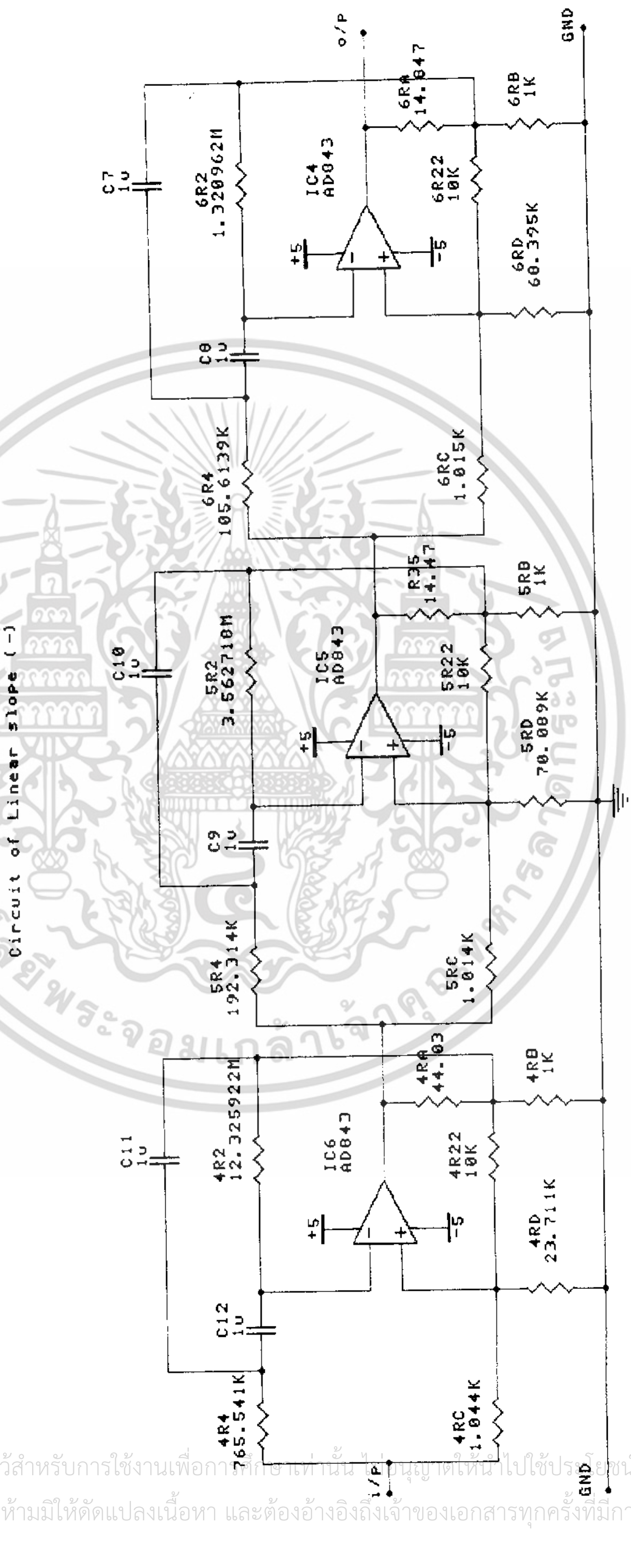
แบบปรินท์และตำแหน่งการวางอุปกรณ์ของวงจร linear slope (+)





Circuit of Linear Slope (+)





Circuit of Linear slope (-)

โปรแกรม EQ2.PAS สำหรับหาค่ารากของสมการโพลิโนเมียล

```
program roots_polynomial;

const ArraySize = 60;              { Maximum number of equations }
type complex    = record
                   Re,Im : real;
                  end;
CompVector = array[0..ArraySize] of complex;
IntVector  = array[0..ArraySize] of integer;
RealVector = array[1..ArraySize] of real;
matrix = array[1..ArraySize] of RealVector;
var DFile  : text;

 function power(x,n:real):real;    { function power(base,order) }
   begin
    if x=0 then power := 0
    else
     if x>0 then  power := exp(ln(x)*n)
     else
      if frac(n)=0 then
       begin
        if odd(trunc(n)) then power := -exp(ln(abs(x))*n)
        else power := exp(ln(abs(x))*n);
       end
      else
       begin
        writeln('*** error power ***');
        read;
       end;
   end;

 procedure Sort(var ary: CompVector; n: integer);

  procedure swapping(var first,second: complex);
   var tem : complex;
   begin
    tem := first;
    first := second;
    second := tem;
   end;

  var d,i  : integer;
      Done : boolean;
  begin
   d := n;
   while d>1 do
    begin
     d := d div 2;
     repeat
      done := true;
      for i := 1 to (n-d) do
       if ary[i].Re>ary[i+d].Re then
        begin
         swapping(ary[i],ary[i+d]);
         done := false;
        end;
     until done;
    end;
  end;

  procedure Laguerre(var Degree    : integer;
                     var Poly      : CompVector;
                         InitGuess : complex;
                         Tol       : real;
                         MaxIter   : integer;
                     var NumRoots  : integer;
                     var Roots     : CompVector;
```

```
                  var yRoots    : CompVector;
                  var Iter      : IntVector;
                  var Error     : byte);
const NearlyZero = 1E-10;
type  quadratic = record
                    A,B,C : real;
                   end;
var AddIter    : integer;
    InitDegree : integer;
    InitPoly   : CompVector;
    GuessRoot  : complex;

procedure Conjugate(var C1,C2 : complex);
 begin
  C2.Re := C1.Re;
  C2.Im := -C1.Im;
 end;

function Modulus(var C1 : complex) : real;
 begin
  Modulus := Sqrt(Sqr(C1.Re) + Sqr(C1.Im));
 end;

procedure Add(var C1,C2,C3 : complex);
 begin
  C3.Re := C1.Re + C2.Re;
  C3.Im := C1.Im + C2.Im;
 end;

procedure Sub(var C1,C2,C3 : complex);
 begin
  C3.Re := C1.Re - C2.Re;
  C3.Im := C1.Im - C2.Im;
 end;

procedure Mult(var C1,C2,C3 : complex);
 begin
  C3.Re := C1.Re * C2.Re - C1.Im * C2.Im;
  C3.Im := C1.Im * C2.Re + C1.Re * C2.Im;
 end;

procedure Divide(var C1,C2,C3 : complex);
 var Dum1,Dum2 : complex;
     E : real;
 begin
  Conjugate(C2,Dum1);
  Mult(C1,Dum1,Dum2);
  E := Sqr(Modulus(C2));
  C3.Re := Dum2.Re / E;
  C3.Im := Dum2.Im / E;
 end;

procedure SquareRoot(var C1,C2 : complex);
 const NearlyZero = 1E-10;
 var   R,Theta : real;
 begin
  R := Sqrt(Sqr(C1.Re) + Sqr(C1.Im));
  if ABS(C1.Re) < NearlyZero then
   begin
    if C1.Im < 0 then Theta := Pi / 2
    else Theta := -Pi / 2;
   end
  else
   if C1.Re < 0 then Theta := ArcTan(C1.Im / C1.Re) + Pi
   else Theta := ArcTan(C1.Im / C1.Re);
  C2.Re := Sqrt(R) * Cos(Theta / 2);
```

```
  C2.Im := Sqrt(R) * Sin(Theta / 2);
 end;

procedure InitAndTest(var Degree     : integer;
                      var Poly       : CompVector;
                          Tol        : real;
                          MaxIter    : integer;
                          InitGuess  : complex;
                      var NumRoots   : integer;
                      var Roots      : CompVector;
                      var yRoots     : CompVector;
                      var Iter       : IntVector;
                      var GuessRoot  : complex;
                      var InitDegree : integer;
                      var InitPoly   : CompVector;
                      var Error      : byte);
 var Term : integer;
 begin
  Error := 0;
  if Degree <= 0 then Error := 2;
  if Tol <= 0 then Error := 3;
  if MaxIter < 0 then Error := 4;

  if Error = 0 then
   begin
    NumRoots := 0;
    GuessRoot := InitGuess;
    InitDegree := Degree;
    InitPoly := Poly;
    while (Degree > 0) and (Modulus(Poly[Degree]) < NearlyZero) do
     Degree := Pred(Degree);
    while (Modulus(Poly[0]) = 0) and (Degree > 0) do
     begin
      NumRoots := Succ(NumRoots);
      Roots[NumRoots].Re := 0;
      Roots[NumRoots].Im := 0;
      yRoots[NumRoots].Re := 0;
      yRoots[NumRoots].Im := 0;
      Iter[NumRoots] := 0;
      Degree := Pred(Degree);
      for Term := 0 to Degree do
       Poly[Term] := Poly[Term + 1];
     end;
   end;
  end;

 procedure FindOneRoot(Degree     : integer;
                       Poly       : CompVector;
                       GuessRoot  : complex;
                       Tol        : real;
                       MaxIter    : integer;
                   var Root       : complex;
                   var yValue     : complex;
                   var Iter       : integer;
                   var Error      : byte);

   var Found : boolean;
       Dif : complex;
       yPrime,yDoublePrime : complex;

   procedure EvaluatePoly(Degree       : integer;
                          Poly         : CompVector;
                          X            : complex;
                      var yValue       : complex;
                      var yPrime       : complex;
                      var yDoublePrime : complex);
```

```
var Loop : integer;
    Dummy,yDPdummy : complex;
    Deriv,Deriv2 : CompVector;
begin
 Deriv[Degree] := Poly[Degree];
 for Loop := Degree - 1 downto 0 do
 begin
  Mult(Deriv[Loop + 1],X,Dummy);
  Add(Dummy,Poly[Loop],Deriv[Loop]);
 end;
 yValue := Deriv[0];

 Deriv2[Degree] := Deriv[Degree];
 for Loop := Degree - 1 downto 1 do
  begin
   Mult(Deriv2[Loop + 1],X,Dummy);
   Add(Dummy,Deriv[Loop],Deriv2[Loop]);
  end;
 yPrime := Deriv2[1];

 yDPdummy := Deriv2[Degree];
 for Loop := Degree - 1 downto 2 do
  begin
   Mult(yDPdummy,X,Dummy);
   Add(Dummy,Deriv2[Loop],yDPdummy);
  end;
 yDoublePrime.Re := 2 * yDPdummy.Re;
 yDoublePrime.Im := 2 * yDPdummy.Im;
end;

procedure ConstructDifference(Degree       : integer;
                              yValue       : complex;
                              yPrime       : complex;
                              yDoublePrime : complex;
                          var Dif          : complex);
 var yPrimeSQR,yTimesyDPrime,Sum,SRoot,Numer1,Numer2,Numer,Denom : complex;
 begin
  Mult(yPrime,yPrime,yPrimeSQR);
  yPrimeSQR.Re := Sqr(Degree - 1) * yPrimeSQR.Re;
  yPrimeSQR.Im := Sqr(Degree - 1) * yPrimeSQR.Im;
  Mult(yValue,yDoublePrime,yTimesyDPrime);
  yTimesyDPrime.Re := (Degree - 1) * Degree * yTimesyDPrime.Re;
  yTimesyDPrime.Im := (Degree - 1) * Degree * yTimesyDPrime.Im;
  Sub(yPrimeSQR,yTimesyDPrime,Sum);
  SquareRoot(Sum,SRoot);
  Add(yPrime,SRoot,Numer1);
  Sub(yPrime,SRoot,Numer2);
  if Modulus(Numer1) > Modulus(Numer2) then Numer := Numer1
  else Numer := Numer2;
  Denom.Re := Degree * yValue.Re;
  Denom.Im := Degree * yValue.Im;
  if Modulus(Numer) < NearlyZero then
   begin
    Dif.Re := 0;
    Dif.Im := 0;
   end
  else
   Divide(Denom,Numer,Dif);
 end;

 function TestForRoot(X,Dif,Y,Tol : real) : boolean;
 begin
  TestForRoot := (ABS(Y) <= NearlyZero) or (ABS(Dif) < ABS(X * Tol))
 end;

 begin { procedure FindOneRoot }
```

```
Root := GuessRoot;
Found := false;
Iter := 0;
EvaluatePoly(Degree,Poly,Root,yValue,yPrime,yDoublePrime);
while (Iter < MaxIter) and not(Found) do
 begin
  Iter := Succ(Iter);
  ConstructDifference(Degree,yValue,yPrime,yDoublePrime,Dif);
  Sub(Root,Dif,Root);
  EvaluatePoly(Degree,Poly,Root,yValue,yPrime,yDoublePrime);
  Found := TestForRoot(Modulus(Root),Modulus(Dif),Modulus(yValue),Tol);
 end;
 if not(Found) then Error := 1;
end;

procedure ReducePoly(var Degree : integer;
                     var Poly   : CompVector;
                     Root       : complex);
var Term : integer;
    NewPoly : CompVector;
    Dummy : complex;
begin
 NewPoly[Degree - 1] := Poly[Degree];
 for Term := Degree - 1 downto 1 do
  begin
   Mult(NewPoly[Term],Root,Dummy);
   Add(Dummy,Poly[Term],NewPoly[Term - 1]);
  end;
 Degree := Pred(Degree);
 Poly := NewPoly;
end;

begin  { procedure Laguerre }
 InitAndTest(Degree,Poly,Tol,MaxIter,InitGuess,NumRoots,Roots,
           yRoots,Iter,GuessRoot,InitDegree,InitPoly,Error);
 while (Degree > 0) and (Error = 0) do
  begin
   FindOneRoot(Degree,Poly,GuessRoot,Tol,MaxIter,
              Roots[NumRoots + 1],yRoots[NumRoots + 1],
              Iter[NumRoots + 1],Error);
   if Error = 0 then
    begin
     FindOneRoot(InitDegree,InitPoly,Roots[NumRoots + 1],
                Tol,MaxIter,Roots[NumRoots + 1],
                yRoots[NumRoots + 1],AddIter,Error);
     Iter[NumRoots + 1] := Iter[NumRoots + 1] + AddIter;
     NumRoots := Succ(NumRoots);
     ReducePoly(Degree,Poly,Roots[NumRoots]);
    end;
   GuessRoot := Roots[NumRoots];
  end;
end;

procedure Initial(var Poly : CompVector);
begin
 FillChar(Poly,SizeOf(Poly),0);
end;

procedure UserInput(var Degree  : integer;
                    var Poly    : CompVector;
                    var Guess   : complex;
                    var Tol     : real;
                    var MaxIter : integer);
 var Ch : char;

 procedure GetCoefficientsFromKeyboard(var Degree : integer;
```

```
                                       var Poly   : CompVector);
var Term : integer;
begin
 Write('Degree of the polynomial = ');
 Readln(Degree);
 Writeln;
 for Term := 0 to Degree do
  begin
   Write(' Poly[',Term:2,']   Re : ',Poly[Term].Re,'  = ');
   Readln(Poly[Term].Re);
   Write('            Im : ',Poly[Term].Im,'  = ');
   Readln(Poly[Term].Im);
  end;
 end;

begin { procedure UserInput }
 GetCoefficientsFromKeyboard(Degree,Poly);
 Guess.Re := 1;
 Guess.Im := 1;
 Tol := 1E-10;
 MaxIter := 1000;
end;

procedure Results(NumRoots   : integer;
                  Answer     : CompVector;
                  MaxIter    : integer;
                  Error      : byte);
 var Term : integer;
 begin
  Writeln;
  Sound(3000);
  Delay(2);
  NoSound;
  case Error of
   1 : Writeln('This will take more than ',MaxIter,' iterations.');
   2 : Writeln('The degree of the polynomial must be greater than zero.');
   3 : Writeln('The tolerance must be greater than zero.');
   4 : Writeln('The maximum number of iterations must be greater than zero.');
  end;
  if Error <= 1 then
   begin
    for Term := 1 to NumRoots do
     begin
      Write(' Root ',Term:2,'    Re = ',Answer[Term].Re:15:8);
      Writeln('        Im = ',Answer[Term].Im:15:8);
     end;
    read;
     end;
 end;

 procedure polmul(Poly1        : CompVector;
                  Poly2        : CompVector;
                  InitDegree1  : integer;
                  InitDegree2  : integer;
              var Poly         : CompVector;
              var Degree       : integer);
  var i,k,j :integer;
   begin
    Degree := InitDegree1+InitDegree2;
    for i := 0 to Degree do
     begin
      poly[i].Re := 0;  poly[i].Im := 0;
      for k := 0 to InitDegree2 do
       begin
        j := i-k;
        if (j>=0) and(j<=InitDegree1) then
```

```
      begin
       poly[i].Re := poly[i].Re+
       poly1[j].Re*poly2[k].Re-poly1[j].Im*poly2[k].Im;
       poly[i].Im := poly[i].Im
       +poly1[j].Re*poly2[k].Im+poly1[j].Im*poly2[k].Re;
      end;
    end;
  end;
 end;

procedure ResultPoly(NumRoots   : integer;
                     Answer     : CompVector);
 var a,b,c    : CompVector;
     i,j,k,n  : integer;
 begin
  writeln;
  Sound(3000);
  Delay(2);
  NoSound;
  writeln('Result:');
  writeln;
  j := 0;
  n := -1;
  for i := 1 to NumRoots do
   if Answer[i].Re<0 then
    begin
     j := j+1;
     if j=1 then
      begin
       n := 1;
       a[1].Re := 1;
       a[1].Im := 0;
       a[0].Re := -Answer[i].Re;
       a[0].Im := -Answer[i].Im;
      end
     else
      begin
       b[1].Re := 1;
       b[1].Im := 0;
       b[0].Re := -Answer[i].Re;
       b[0].Im := -Answer[i].Im;
       polmul(a,b,n,1,c,n);
       for k := 0 to n do
        begin
         a[k].Re := c[k].Re;  a[k].Im := c[k].Im;
        end;
      end;
    end;
   for i := 0 to n do
    begin
     write(' Poly[',i:2,']   Re = ',a[i].Re:15:8);
     writeln('        Im = ',a[i].Im:15:8);
    end;
  end;

 procedure MulPoly;
  var a,b,c         : CompVector;
      n,n1,n2,i,j   : integer;
  begin
   n := 0;
   n1 := 0;
   Initial(a);
   Write('Number of polynomial = '); Readln(n);
   Writeln;
   Write('Degree of the polynomial 1 = '); readln(n1);
    for i := 0 to n1 do
```

```
  begin
   Write(' Poly[',i:2,']   Re = ');  Readln(a[i].Re);
   Write('            Im = ');  Readln(a[i].Im);
  end;

writeln;
for j := 2 to n do
 begin
  n2 := 0;
  Initial(b);
  Write('Degree of the polynomial ',j,'  = ');  readln(n2);
  for i := 0 to n2 do
   begin
    Write(' Poly[',i:2,']   Re = ');  Readln(b[i].Re);
    Write('            Im = ');  Readln(b[i].Im);
   end;

  polmul(a,b,n1,n2,c,n1);

  writeln;
  for i := 0 to n1 do
   begin
    a[i].Re := c[i].Re;  a[i].Im := c[i].Im;
   end;
 end;
 Sound(3000);
 Delay(2);
 NoSound;
 writeln('Result:');
 writeln;
 for i := 0 to n1 do
  begin
   write(' Poly[',i:2,']   Re = ',a[i].Re:15:8);
   writeln('         Im = ',a[i].Im:15:8);
  end;
end;

procedure Delay_equalizer;
var Coefficients               : matrix;
    Constants,Solution,a,b,w   : RealVector;
    data,d1,d2                 : complex;
    e,D0,D,Tol,min,max,mm      : Real;
    Dimen,Row,Column,i,iter    : integer;
    Error                      : byte;
    IterM                      : boolean;
label loop,exitd;

 function sinh(x:real):real;
  begin
   sinh := (exp(x)-exp(-x))/2;
  end;

 function cosh(x:real):real;
  begin
   cosh := (exp(x)+exp(-x))/2;
  end;

  procedure Add_C(     C1_Re,C1_Im,C2_Re,C2_Im  : real;
                 var C3                         : complex);
   begin
    C3.Re := C1_Re + C2_Re;
    C3.Im := C1_Im + C2_Im;
   end;

  procedure Sub_C(     C1_Re,C1_Im,C2_Re,C2_Im  : real;
                 var C3                         : complex);
```

```
begin
 C3.Re := C1_Re - C2_Re;
 C3.Im := C1_Im - C2_Im;
end;

procedure Mul_C(    C1_Re,C1_Im,C2_Re,C2_Im   : real;
               var C3                         : complex);
 begin
  C3.Re := C1_Re * C2_Re - C1_Im * C2_Im;
  C3.Im := C1_Im * C2_Re + C1_Re * C2_Im;
 end;

procedure Div_C(     C1_Re,C1_Im,C2_Re,C2_Im  : real;
               var C3                         : complex);
 var Dum  : complex;
     E    : real;
 begin
  Dum.Re := C1_Re * C2_Re + C1_Im * C2_Im;
  Dum.Im := C1_Im * C2_Re - C1_Re * C2_Im;
  E := Sqr(C2_Re) + Sqr(C2_Im);
  C3.Re := Dum.Re / E;
  C3.Im := Dum.Im / E;
 end;

 function EquF(w:real):real;
 var D  : real;
     i  : integer;
 begin
{
   D := -(0.54367/( sqr(0.54367)+sqr(w*5.573e-2*2*pi))
        -0.42738/( sqr(0.42738)+sqr(w*5.573e-2*2*pi)))/2.856;
}

   D := -w;



   for i := 1 to (Dimen shr 1) do
     D := D + ( ( 2*a[i] ) / ( sqr(a[i])+sqr(w-b[i]) ) +
               ( 2*a[i] ) / ( sqr(a[i])+sqr(w+b[i]) ) );
   EquF := D/(2*pi);

  end;

  function DifF(x: real):real;
  const Tolerance = 1e-10;
  type  vector = array[1..100] of real;
  var   Term,Iter,TwoToTheIterMinus2,Extrap  : integer;
        DeltaX,FourToTheExtrapMinus1         : real;
        OldEstimate,NewEstimate              : vector;

   function EvaluateFirstDeriv(X      : real;
                               DeltaX : real):real;
   var LeftPoint,RightPoint : real;
   begin
     LeftPoint := EquF(X - DeltaX);
     RightPoint := EquF(X + DeltaX);
     EvaluateFirstDeriv := (RightPoint - LeftPoint)/(2 * DeltaX);
   end;

  begin
      if ABS(X) < Tolerance then
        DeltaX := Sqrt(Tolerance)
      else
        DeltaX := ABS(X * Sqrt(Tolerance));
      OldEstimate[1] := EvaluateFirstDeriv(X,DeltaX);
```

```
  Iter := 1;
  TwoToTheIterMinus2 := 1;
  repeat
    Iter := Succ(Iter);
    DeltaX := DeltaX / 2;
    NewEstimate[1] := EvaluateFirstDeriv(X,DeltaX);
    TwoToTheIterMinus2 := TwoToTheIterMinus2 * 2;
    FourToTheExtrapMinus1 := 1;
    for Extrap := 2 to Iter do
     begin
      FourToTheExtrapMinus1 := FourToTheExtrapMinus1 * 4;
      NewEstimate[Extrap] :=
         (FourToTheExtrapMinus1 * NewEstimate[Extrap - 1]
         - OldEstimate[Extrap - 1]) / (FourToTheExtrapMinus1 - 1);
     end;
    OldEstimate := NewEstimate;
  until (ABS(NewEstimate[Iter - 1] - NewEstimate[Iter]) <=
         ABS(Tolerance * NewEstimate[Iter])) or (ABS(DeltaX) < Tolerance);
  DiffF := NewEstimate[Iter];
end;

procedure MinMax(LeftEndPoint  : real;
                 RightEndPoint : real;
             var Answer        : real;
             var Error         : byte);
 const NearlyZero = 1E-10;
 var Tol,yLeft,yRight,MidPoint,yMidPoint : real;
     Iter, MaxIter                       : integer;
     Found                               : boolean;

 function TestForRoot(X,OldX,Y,Tol : real) : boolean;
 begin
   TestForRoot := (ABS(Y) <= NearlyZero) or (ABS(X - OldX) < ABS(OldX * Tol))
 end;

 begin
  MaxIter := 700;
  Tol := 1E-10;
  Error := 0;
  Found := false;
  yLeft := DiffF(LeftEndpoint);
  yRight := DiffF(RightEndpoint);
  if ABS(yLeft) <= NearlyZero then
  begin
    Answer := LeftEndpoint;
    Found := true;
  end;
  if ABS(yRight) <= NearlyZero then
  begin
    Answer := RightEndpoint;
    Found := true;
  end;
  if not Found then   { Test for errors }
  begin
    if yLeft * yRight > 0 then
      Error := 2;
    if Tol <= 0 then
      Error := 3;
    if MaxIter < 0 then
      Error := 4;
  end;
  if (Error = 0) and (Found = false) then
  begin
    Iter := 0;
    yLeft := DiffF(LeftEndPoint);
    while not(Found) and (Iter < MaxIter) do
```

```
    begin
     Iter := Succ(Iter);
     MidPoint := (LeftEndpoint + RightEndpoint) / 2;
     yMidPoint := DifF(MidPoint);
     Found := TestForRoot(MidPoint, LeftEndpoint, yMidPoint, Tol);
     if (yLeft * yMidPoint) < 0 then
        RightEndpoint := MidPoint
     else
       begin
         LeftEndpoint := MidPoint;
         yLeft := yMidPoint;
       end;
    end;
   Answer := MidPoint;
   if Iter >= MaxIter then
     Error := 1;
  end;
end;

procedure Partial_Pivoting(Dimen        : integer;
                           Coefficients : matrix;
                           Constants    : RealVector;
                       var Solution     : RealVector;
                       var Error        : byte);
const NearlyZero = 1E-10;
var Sum,Multiplier,Dummy              : real;
    Term,Row,ReferenceRow,PivotRow    : integer;
    DummyRow                          : RealVector;

 procedure EROmultAdd(Multiplier   : real;
                      Dimen        : integer;
                  var ReferenceRow : RealVector;
                  var ChangingRow  : RealVector);
 var Term : integer;
 begin
   for Term := 1 to Dimen do
     ChangingRow[Term] := ChangingRow[Term] + Multiplier*ReferenceRow[Term];
 end;

 begin
   Error := 0;
   if Dimen < 1 then
     Error := 1
   else
     if Dimen = 1 then
       if ABS(Coefficients[1, 1]) < NearlyZero then
         Error := 2
       else
         Solution[1] := Constants[1] / Coefficients[1, 1];
   if Dimen > 1 then
   begin
     { Make Coefficients matrix upper triangular }
     ReferenceRow := 0;
     while (Error = 0) and (ReferenceRow < Dimen - 1) do
     begin
       ReferenceRow := Succ(ReferenceRow);
       { Find row with largest element in this column }
       { and switch this row with the ReferenceRow    }
       PivotRow := ReferenceRow;
       for Row := ReferenceRow + 1 to Dimen do
         if ABS(Coefficients[Row, ReferenceRow]) >
            ABS(Coefficients[PivotRow, ReferenceRow]) then
           PivotRow := Row;
       if PivotRow <> ReferenceRow then
       begin
         DummyRow := Coefficients[PivotRow];
```

```
        Coefficients[PivotRow] := Coefficients[ReferenceRow];
        Coefficients[ReferenceRow] := DummyRow;
        Dummy := Constants[PivotRow];
        Constants[PivotRow] := Constants[ReferenceRow];
        Constants[ReferenceRow] := Dummy;
      end
    else { If the diagonal element is zero, no solution exists }
      if ABS(Coefficients[ReferenceRow, ReferenceRow]) < NearlyZero then
        Error := 2;      { No solution }
    if Error = 0 then
      for Row := ReferenceRow + 1 to Dimen do
        { Make the ReferenceRow element of these rows zero }
        if ABS(Coefficients[Row, ReferenceRow]) > NearlyZero then
        begin
          Multiplier := -Coefficients[Row, ReferenceRow] /
                         Coefficients[ReferenceRow,ReferenceRow];
          EROmultAdd(Multiplier, Dimen,
                     Coefficients[ReferenceRow], Coefficients[Row]);
          Constants[Row] := Constants[Row] +
                            Multiplier * Constants[ReferenceRow];
        end;
   end;
   if ABS(Coefficients[Dimen, Dimen]) < NearlyZero then
     Error := 2;    { No solution }
   if Error = 0 then
   begin
     Term := Dimen;
     while Term >= 1 do
     begin
       Sum := 0;
       for Row := Term + 1 to Dimen do
         Sum := Sum + Coefficients[Term, Row] * Solution[Row];
       Solution[Term] := (Constants[Term] - Sum) / Coefficients[Term, Term];
       Term := Pred(Term);
     end;
   end;
  end;
 end;

begin
  Tol := 1E-10;
  iter := -1;
  IterM := false;
  FillChar(Coefficients, SizeOf(Coefficients), 0);
  FillChar(Constants, SizeOf(Constants), 0);


  Dimen := 6;
  e := 0.01;
  D0 := 2.27/pi;
  a[1] := 0.069487132001;  b[1] := 0.63989054353;
  a[2] := 0.051667605733;  b[2] := 1.4326942020;
  a[3] := 0.090645262266;  b[3] := 2.3502369035;


  a[4] := 1.3575;  b[4] := 4.6667;
  a[5] := 1.086;   b[5] := 6;


 loop:
  iter := iter+1;
  gotoxy(25,1);  writeln('Iterate ',iter); writeln;
  for i := 1 to (Dimen shr 1) do
    Writeln('a',i,' = ',a[i]:13:10,'b':15,i,' = ',b[i]:13:10);
  writeln;
```

```
min := 0;
max := 0;
for Row := 1 to Dimen do
 begin
  if odd(Row) then
   begin              (* Minimum *)
    if Row=1 then
     begin
      if not IterM then
        w[Row] := 0
      else
       begin
        w[Row] := b[1]/3;
        if DifF(w[Row])>=0 then
          MinMax(0,w[Row],mm,Error)
        else
          MinMax(w[Row],b[1],mm,Error);
        if Error=0 then w[Row] := mm;
       end;
     end
    else
     begin
      w[Row] := (b[(Row shr 1)]+b[(Row shr 1)+1])/2;
      if IterM then
       begin
        if DifF(w[Row])>=0 then
          MinMax(b[(Row shr 1)],w[Row],mm,Error)
        else
          MinMax(w[Row],b[(Row shr 1)+1],mm,Error);
        if Error=0 then w[Row] := mm;
       end;
     end;
   end
  else
   begin               (* Maximum *)
    w[Row] := b[Row shr 1];
    if IterM then
     begin
      if DifF(w[Row])<=0 then
        MinMax((b[(Row shr 1)-1]+b[(Row shr 1)])/2,w[Row],mm,Error)
      else
        MinMax(w[Row],(b[(Row shr 1)]+b[(Row shr 1)+1])/2,mm,Error);
      if Error=0 then w[Row] := mm;
     end;
   end;
  D := EquF(w[Row]);
  if odd(Row) then
   begin
    if (Row=1) and not IterM then
     begin
      Constants[Row] := D0-D;
      if min<abs(D-D0) then  min := abs(D-D0);
     end
    else
     begin
      Constants[Row] := D0-D-e/2;
      if min<abs(D-D0+e/2) then  min := abs(D-D0+e/2);
     end;
   end
  else
   begin
    Constants[Row] := D0-D+e/2;
    if max<abs(D-D0-e/2) then  max := abs(D-D0-e/2);
   end;
  Writeln('w',Row:2,' = ',w[Row],'delta D':15,' = ',Constants[Row]);
 end;
```

```
 if iter=0 then read;
 if (min<Tol) and (max<Tol) then
  begin
   if IterM then
     goto exitd
   else
    begin
     gotoxy(1,1);  write('First[',iter,']');
     IterM := true;
     read;
    end;
  end;

 for Row := 1 to Dimen do
  begin
   for Column := 1 to Dimen do
    begin
      if Column<=(Dimen shr 1) then
       begin
        i := Column;
        Coefficients[Row, Column] :=
         2*( ( sqr(w[Row]-b[i]) - sqr(a[i]) )
             / sqr( sqr(a[i]) + sqr(w[Row]-b[i]) )
           + ( sqr(w[Row]+b[i]) - sqr(a[i]) )
             / sqr( sqr(a[i]) + sqr(w[Row]+b[i]) ) );
       end
      else
       begin
        i := Column - (Dimen shr 1);
        Coefficients[Row, Column] :=
         4*a[i]*( ( w[Row]-b[i] ) / sqr( sqr(a[i]) + sqr(w[Row]-b[i]) )
                - ( w[Row]+b[i] ) / sqr( sqr(a[i]) + sqr(w[Row]+b[i]) ) );
       end;
    end;
  end;
  Partial_Pivoting(Dimen,Coefficients,Constants,Solution,Error);
  if Error=0 then
   begin
    for i := 1 to (Dimen shr 1) do
     begin
      a[i] := a[i]+Solution[i];
      b[i] := b[i]+Solution[(Dimen shr 1)+i];
     end;
   end
  else
   begin
    Writeln;
    Write('***** Error matrix *****'); read;
   end;
  goto loop;
 exitd:
end;

var Guess                                 : complex;
    InitDegree,Degree,MaxIter,NumRoots : integer;
    Tol                                   : real;
    Iter                                  : IntVector;
    InitPoly,Poly,Answer,yAnswer          : CompVector;
    Error                                 : byte;
    ch                                    : char;
begin
  InitDegree := 0;
  Initial(InitPoly);
  repeat
   ClrScr;
   gotoxy(24,9);  write('1) Multiple Polynomial');
```

```
gotoxy(24,10); write('2) Roots & Polynomial');
gotoxy(24,11); write('3) Delay-equalizer');
gotoxy(24,12); write('4) ...');
gotoxy(24,13); write('5) ...');
gotoxy(24,14); write('6) ...');
gotoxy(24,15); write('7) ...');
gotoxy(24,16); write('8) ...');
gotoxy(24,17); write('9) ...');
gotoxy(23,18); write('ESC Quit');
gotoxy(27,20); write('Select No. ');
repeat
 gotoxy(38,20); read(kbd,ch); write(ch);
until ch in [#27,'1'..'3'];
if ch<>#27 then
 begin
  ClrScr;
  Assign(DFile,'SOLUTION');
  Rewrite(DFile);
  case ch of
   '1' : MulPoly;
   '2' : UserInput(InitDegree,InitPoly,Guess,Tol,MaxIter);
   '3' : Delay_equalizer;
  end;
  if ch in ['2'] then
   begin
    Degree := InitDegree;
    Poly := InitPoly;
    Laguerre(Degree,Poly,Guess,Tol,MaxIter,
             NumRoots,Answer,yAnswer,Iter,Error);
    Sort(Answer,Numroots);
    Results(NumRoots,Answer,MaxIter,Error);
    if Error <= 1 then ResultPoly(NumRoots,Answer);
   end;
  read;
  Close(DFile);
 end;
until ch=#27;
end.
```

34 MHz, CBFET Fast Settling Op Amp

AD843

ANALOG DEVICES

FEATURES

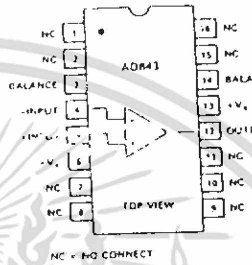
AC PERFORMANCE
Unity Gain Bandwidth: 34 MHz
Fast Settling: 135 ns to 0.01%
Slew Rate: 250 V/µs
Stable at Gains of 1 or Greater
Full Power Bandwidth: 3.9 MHz

DC PERFORMANCE
Input Offset Voltage: 1 mV max (AD843K/B)
Input Bias Current: 0.6 nA typ
Input Voltage Noise: 19 nV/√Hz
Open Loop Gain: 30 V/mV into a 500 Ω Load
Output Current: 50 mA min
Supply Current: 13 mA max
Available in 8-Pin Plastic Mini-DIP & Cerdip, 16-Pin SOIC, 20-Pin LCC and 12-Pin Hermetic Metal Can Packages
Available in Tape and Reel in Accordance with EIA-481A Standard
Chips and MIL-STD-883B Parts Also Available

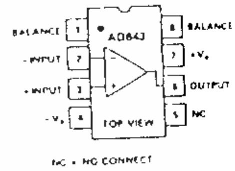
APPLICATIONS
High Speed Sample-and-Hold Amplifiers
High Bandwidth Active Filters
High Speed Integrators
High Frequency Signal Conditioning

CONNECTION DIAGRAMS

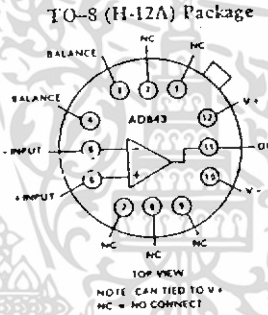
16-Pin SOIC (R-16) Package



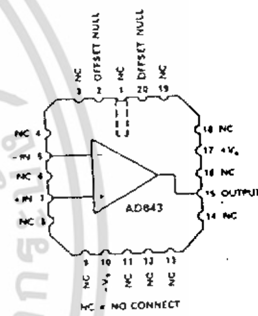
Plastic (N-8) and Cerdip (Q-8) Package



TO-8 (H-12A) Package



LCC (E-20A) Package



PRODUCT DESCRIPTION

The AD843 is a fast settling, 34 MHz, CBFET input op amp. The AD843 combines the low (0.6 nA) input bias currents characteristic of a FET input amplifier while still providing a 34 MHz bandwidth and a 135 ns settling time (to within 0.01% of final value for a 10 volt step). The AD843 is a member of the Analog Devices' family of wide bandwidth operational amplifiers. These devices are fabricated using Analog Devices' junction isolated complementary bipolar (CB) process. This process permits a combination of dc precision and wideband ac performance previously unobtainable in a monolithic op amp.

The 250 V/µs slew rate and 0.6 nA input bias current of the AD843 ensure excellent performance in high speed sample-and-hold applications and in high speed integrators. This amplifier is also ideally suited for high bandwidth active filters and high frequency signal conditioning circuits.

Unlike many high frequency amplifiers, the AD843 requires no external compensation and it remains stable over its full operating temperature range. It is available in five performance grades: the AD843J and AD843K are rated over the commercial temperature range of 0°C to +70°C. The AD843A and AD843B are rated over the industrial temperature range of −40°C to +85°C. The AD843S is rated over the military temperature range of −55°C to +125°C and is available processed to MIL-STD-883B, Rev. C.

The AD843 is offered in either 8-pin plastic DIP or hermetic cerdip packages, in 16-pin SOIC, 20-Pin LCC, or in a 12-pin metal can. Chips are also available.

PRODUCT HIGHLIGHTS

1. The high slew rate, fast settling time and low input bias current of the AD843 make it the ideal amplifier for 12-bit D/A and A/D buffers, for high speed sample-and-hold amplifiers and for high speed integrator circuits. The AD843 can replace many FET input hybrid amplifiers such as the LH0032, LH4104 and OPA600.
2. Fully differential inputs provide outstanding performance in all standard high frequency op amp applications such as signal conditioning and active filters.
3. Laser wafer trimming reduces the input offset voltage to 1 mV max (AD843K and AD843B).
4. Although external offset nulling is unnecessary in many applications, offset null pins are provided.
5. The AD843 does not require external compensation at closed loop gains of 1 or greater.

This is an abridged data sheet. To obtain the most recent version or complete data sheet, call our fax retrieval system at 1-800-446-6212.

REV. C

SPECIFICATIONS (@ T_A +25°C and ±15 V dc, unless otherwise noted)

Model	Conditions	AD843J/A Min	AD843J/A Typ	AD843J/A Max	AD843K/B Min	AD843K/B Typ	AD843K/B Max	AD843S[1] Min	AD843S[1] Typ	AD843S[1] Max	Units
INPUT OFFSET VOLTAGE[1]			1.0	2.0		0.5	1.0		1.0	2.0	mV
	T_{min}–T_{max}		1.7	4.0		1.2	2.0		3.0	4.5	mV
Offset Drift			12			12	35		12		μV/°C
INPUT BIAS CURRENT	Initial (T_J = +25°C)		50			40			50		pA
	Warmed-Up[2]		0.8	2.5		0.6	1.0		0.8	2.5	nA
	T_{min}–T_{max}			60/160			23/65			2600	nA
INPUT OFFSET CURRENT	Initial (T_J = +25°C)		30			20			30		pA
	Warmed-Up[2]		0.25	1.0		0.2	0.4		0.25	1.0	nA
	T_{min}–T_{max}			23/64			9/26			1025	nA
INPUT CHARACTERISTICS											
Input Resistance			10^{10}			10^{10}			10^{10}		Ω
Input Capacitance			6			6			6		pF
INPUT VOLTAGE RANGE											
Common Mode		±10	+12, −13		±10	+12, −13		±10	+12, −13		V
COMMON MODE REJECTION	V_{CM} = ±10 V	60	72		70	76		60	72		dB
	T_{min}–T_{max}	60	72		68	76		60	72		dB
INPUT VOLTAGE NOISE	f = 10 kHz		19			19			19		nV/√Hz
Wideband Noise	10 Hz to 10 MHz		60			60			60		μV rms
OPEN LOOP GAIN	V_O = ±10 V, R_{LOAD} ≥ 500 Ω	15	25		20	30		15	30		V/mV
	T_{min}–T_{max}	10	20		10	25		10	25		V/mV
OUTPUT CHARACTERISTICS											
Voltage	R_{LOAD} ≥ 500 Ω	±10	+11.5, −12.6		±10	+11.5, −12.6		±10	+11.5, −12.6		V
Current	V_{OUT} = ±10 V	50			50			50			mA
Output Resistance	Open Loop		12			12			12		Ω
FREQUENCY RESPONSE											
Unity Gain Bandwidth	V_{OUT} = 90 mV p-p		34			34			34		MHz
Full Power Bandwidth[3]	V_O = 20 V p-p, Rl ≥ 500 Ω	2.5	3.9		2.5	3.9		2.5	3.9		MHz
Rise Time	A_{VCL} = −1		10			10			10		ns
Overshoot	A_{VCL} = −1		15			15			15		%
Slew Rate	A_{VCL} = −1	160	250		160	250		160	250		V/μs
Settling Time	10 V Step, A_{VCL} = −1										
	to 0.1%		95			95			95		ns
	to 0.01%		135			135			135		ns
Overdrive Recovery	−Overdrive		200			200			200		ns
	+Overdrive		700			700			700		ns
Differential Gain	f = 4.4 MHz		0.025			0.025			0.025		%
Differential Phase	f = 4.4 MHz		0.025			0.025			0.025		Degree
POWER SUPPLY											
Rated Performance			±15			±15			±15		V
Operating Range		±4.5		±18	±4.5		±18	±4.5		±18	V
Quiescent Current			12	13		12	13		12	13	mA
	T_{min}–T_{max}		12.3	14		12.3	14		12.5	16	mA
Rejection Ratio	±5 V to ±18 V	65	76		70	80		65	76		dB
Rejection Ratio	T_{min}–T_{max}	62	76		68	80		62	76		dB
TEMPERATURE RANGE											
Operating, Rated Performance											
Commercial (0 to +70°C)			AD843J			AD843K					
Industrial (−40°C to +85°C)			AD843A			AD843B					
Military (−55°C to +125°C)[4]									AD843S		
PACKAGE OPTIONS[5]											
Plastic (N-8)			AD843JN			AD843KN					
Cerdip (Q-8)			AD843AQ			AD843BQ			AD843SQ, AD843SQ/883B		
Metal Can (H-12A)						AD843BH			AD843SH, AD843SH/883B		
LCC (E-20A)									AD843SE/883B		
SOIC (R-16)			AD843JR								
Tape & Reel			AD843JR-REEL								
Chips			AD843JChips						AD843SChips		

NOTES
[1]Standard Military Drawings Available: 5962-9098001M2A (SE/883B), 5962-9098001MXA (SH/883B), 5962-9098001MPA (SQ/883B).
[2]Specifications are guaranteed after 5 minutes at T_A = +25°C.
[3]Full power bandwidth = Slew Rate/2 πV peak.
[4]All "S" grade T_{min}–T_{max} specifications are tested with automatic test equipment at T_A = −55°C and T_A = +125°C.
[5]For outline information see Package Information section.

Specifications subject to change without notice.

Specifications in boldface are tested on all production units at final electrical test. Results from those tests are used to calculate outgoing quality levels. All min and max specifications are guaranteed although only those shown in boldface are tested on all production units.

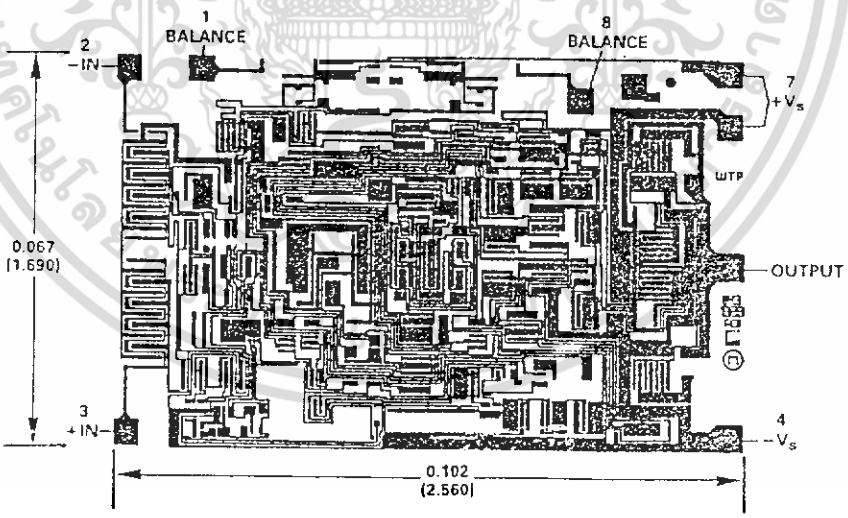
ABSOLUTE MAXIMUM RATINGS[1]

Supply Voltage . . . . ±18 V
Internal Power Dissipation[2]
  Plastic Package . . . . 1.50 Watts
  Cerdip Package . . . . 1.35 Watts
  12-Pin Header Package . . . . 1.80 Watts
  16-Pin SOIC Package . . . . 1.50 Watts
  20-Pin LCC Package . . . . 1.00 Watt
Input Voltage . . . . ±V_S
Output Short Circuit Duration . . . . Indefinite
Differential Input Voltage . . . . +V_S and −V_S
Storage Temperature Range (N, R) . . . . −65°C to +125°C
Storage Temperature Range (Q, H, E) . . . . . −65°C to +150°C
Operating Temperature Range
  AD843J/K . . . . 0 to +70°C
  AD843A/B . . . . −40°C to +85°C
  AD843S . . . . −55°C to +125°C
Lead Temperature Range (Soldering 60 sec) . . . . +300°C
ESD Rating . . . . 500 V

NOTES
[1]Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.
[2]8-Pin Plastic Package: θ_{JA} = 100°C/Watt
8-Pin Cerdip Package: θ_{JA} = 110°C/Watt
12-Pin Header Package: θ_{JA} = 80°C/Watt
16-Pin SOIC Package: θ_{JA} = 100°C/Watt
20-Pin LCC Package: θ_{JA} = 150°C/Watt

METALIZATION PHOTOGRAPH

Contact factory for latest dimensions.
Dimensions shown in inches and (mm).



แบบปรินท์และตำแหน่งการวางอุปกรณ์ของวงจร linear slope (-)