PASCAL MATRICES AND THEIR APPLICATIONS TO DIGITAL SIGNAL PROCESSING

SORAWAT CHIVAPREECHA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
DOCTOR OF ENGINEERING IN ELECTRICAL ENGINEERING
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2008

KMITL-2008-EN-D-018-169

หัวข้อวิทยานิพนธ์	ปาสคาลเมตริกซ์และการประยุกต์ใช้งานด้านการประมวลผลสัญญาณเชิงเลข
นักศึกษา	นายศรวัฒน์ ชิวปรีชา
รหัสประจำตัว	45160305
ปริญญา	วิศวกรรมศาสตรดุษฎีบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้า
พ.ศ.	2551
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ. ดร. กอบชัย เดชหาญ
อาจารย์ผู้ควบคุมวิทยานิพนธ์ร่วม	Prof. Dr. Tian-Bo Deng

บทคัดย่อ

วิทยานิพนธ์นี้นำเสนอเมตริกซ์ชนิดพิเศษที่เรียกว่าปาสคาลเมตริกซ์และการประยุกต์ใช้งานด้านการประมวลผลสัญญาณเชิงเลขทั้งในแง่ของการกรองสัญญาณและการแปลงสัญญาณซึ่งเป็นแขนงหลักของงานด้านการประมวลผลสัญญาณเชิงเลข การนำเสนอแรกเป็นเรื่องของเมตริกซ์ที่เรียกว่าเมตริกซ์ปาสคาลแบบทั่วไปที่ใช้สำหรับการแปลงตัวแปรเอสไปเป็นตัวแปรแซดซึ่งใช้ในงานด้านการออกแบบวงจรกรองสัญญาณ เมตริกซ์ดังกล่าวจะใช้สำหรับการแม็ปตัวแปรของฟังก์ชันถ่ายโอนระหว่างโดเมนเอสไปยังโดเมนแซด และในทางกลับกันด้วยโดยใช้อินเวิร์สปาสคาลเมตริกซ์ ตัวของเมตริกซ์ปาสคาลแบบทั่วไปนั้นจะประกอบไปด้วยการแปลงตัวแปร ดังนี้คือการแปลงแบบผลต่างไปข้างหน้า การแปลงแบบผลต่างย้อนกลับ การแปลงเชิงเส้นคู่ และการแปลงแบบผลต่างย้อนกลับ-เชิงเส้นคู่ที่กำหนดพารามิเตอร์ ในการจะทำให้เมตริกซ์ปาสคาลแบบโดยทั่วไปนั้นทำหน้าที่ในการแปลงตัวแปรแบบใดจะอาศัยการกำหนดจากพารามิเตอร์ซึ่งจะเป็นตัวกำหนดรูปแบบในการแปลงตัวแปร การนำเสนอที่สองเป็นการนำเมตริกซ์ปาสคาลที่ใช้สำหรับการแปลงเชิงเส้นคู่มาใช้ในการออกแบบวงจรกรองสัญญาณแบบผลตอบสนองอิมพัลส์ไม่จำกัดร่วมกับการแปลงความถี่แบบอัตโนมัติจากฟังก์ชันถ่ายโอนต้นแบบบรรทัดฐานในเชิงอุปมานไปเป็นวงจรกรองสัญญาณแบบความถี่ต่ำผ่าน ความถี่สูงผ่าน แถบความถี่ผ่าน และแถบความถี่หยุดโดยใช้สิ่งที่เรียกว่าเมตริกซ์สัมประสิทธิ์สำหรับการสเกลความถี่ จากวิธีการนี้ทำให้ได้ซึ่งวิธีการใหม่สำหรับการออกแบบวงจรกรองสัญญาณเชิงเลขโดยใช้การคำเนินการทางเมตริกซ์แต่เพียงอย่างเดียว ประการถัดไปพิจารณาไปที่ฟังก์ชันถ่ายโอนมาตรฐานแบบไบควอเดติกในเชิงอุปมานที่ถูกใช้กันโดยปกติในการออกแบบวงจรกรองชนิดไบควอด อย่างไรก็ตามในวิทยานิพนธ์นี้ได้นำเสนอเป็นวงจรกรองสัญญาณชนิดไบควอดแบบเชิงเลขทั้งในเรื่องของวิธีการออกแบบและโครงสร้างของวงจรกรองสัญญาณซึ่งสามารถให้ผลตอบสนอง 5 เอาท์พุทในเวลาเดียวกัน สมการการออกแบบสำหรับวงจรกรองชนิดนี้ใช้เพียงสมการเมตริกซ์เพียงสมการเดียวที่ดัดแปลงมาจากเมตริกซ์ปาสคาล

สำหรับการแปลงเชิงเส้นคู่ จึงเรียกเป็นปาสคาลเมตริกซ์ดัดแปลงสำหรับการออกแบบวงจรกรองสัญญาณชนิดไบควอดแบบเชิงเลข สำหรับในแง่ของการแปลงสัญญาณที่ใช้เมตริกซ์ปาสคาล ได้ทำการศึกษาถึงสิ่งที่เรียกว่าการแปลงปาสคาลเต็มหน่วย ตัวเมตริกซ์ปาสคาลที่ใช้สำหรับการแปลงนี้แตกต่างจากเมตริกซ์ปาสคาลที่ใช้สำหรับการแปลงตัวแปรเอส-แซด เมตริกซ์ปาสคาลที่ใช้สำหรับการแปลงนี้ สมาชิกภายในทั้งหมดมาจากค่าสัมประสิทธิ์ทวินามจากสามเหลี่ยมปาสคาลโดยตรง คุณลักษณะทางความถี่ได้ถูกทำการตรวจสอบเพื่อยืนยันถึงผลตอบสนองทางความถี่ของการแปลงปาสคาลเต็มหน่วยที่กระทำต่อสัญญาณ นอกจากนั้นชนิดของเมตริกซ์ปาสคาลที่ใช้สำหรับการแปลงปาสคาลเต็มหน่วยได้ถูกแบ่งเป็น 2 ชนิด ซึ่งถูกแบ่งจากผลตอบสนองทางความถี่ที่ได้มาจากการตรวจสอบ คือเมตริกซ์ปาสคาลชนิดความถี่สูงผ่านและเมตริกซ์ปาสคาลชนิดความถี่ต่ำผ่าน ยิ่งไปกว่านั้นโครงสร้างทางฮาร์ดแวร์ของวงจรที่ใช้ในการแปลงปาสคาลเต็มหน่วยทั้งชนิดความถี่สูงผ่านและชนิดความถี่ต่ำผ่านโดยอาศัยหน่วยผีเสื้อของการแปลงปาสคาลเต็มหน่วยได้ถูกแสดงให้เห็น โครงสร้างที่ใช้ในการคำนวณของการแปลงปาสคาลเต็มหน่วยนั้นสร้างขึ้นมาจากเมตริกซ์เลขฐานสองซึ่งทำการแยกองค์ประกอบมาจากตัวเมตริกซ์การแปลงปาสคาล เป็นผลให้โครงสร้างทั้งหมดสำหรับการแปลงปาสคาลเต็มหน่วยไม่ต้องการตัวคูณ ใช้เพียงตัวบวกหรือตัวลบเท่านั้นสำหรับการคำนวณ ผลการทดลองของการแปลงปาสคาลเต็มหน่วยที่กระทำต่อสัญญาณทั้งกรณีสัญญาณ 1 มิติ และสัญญาณ 2 มิติ ได้นำไปสู่การพัฒนาไปเป็นสิ่งที่เรียกว่าตัวกรองปาสคาลเต็มหน่วย ตัวกรองปาสคาลเต็มหน่วยมีสองชนิดเช่นเดียวกันกับการแบ่งชนิดของการแปลงปาสคาลเต็มหน่วย คือเป็นชนิดความถี่ต่ำผ่านและความถี่สูงผ่าน ผลของสัญญาณเอาท์พุทของตัวกรองปาสคาลเต็มหน่วยสามารถที่จะเห็นประโยชน์ในการนำไปใช้งานได้อย่างชัดเจนมากกว่าผลที่ได้จากการแปลงปาสคาลเต็มหน่วยในความรู้สึกของการกรองสัญญาณ การสร้างเป็นฮาร์ดแวร์จริงของตัวกรองปาสคาลเต็มหน่วยทั้งชนิดความถี่สูงผ่านและความถี่ต่ำผ่านจะถูกออกแบบรวมกันและสามารถเลือกการทำงานได้โดยใช้บิทสำหรับการเลือก ฮาร์ดแวร์ทั้งหมดจะถูกสร้างลงไปบนอุปกรณ์ FPGA โดยเฉพาะในกรณีของสัญญาณ 2 มิติหรือภาพเชิงเลข สามารถที่จะแสดงภาพที่เป็นผลลัพธ์บนมอนิเตอร์แบบ VGA ได้โดยตรงจากตัว FPGA โดยปราศจากการเชื่อมต่อคอมพิวเตอร์ส่วนบุคคลสำหรับหน่วยการแสดงผล

Thesis Title	Pascal Matrices and Their Applications to Digital Signal Processing
Student	Mr. Sorawat Chivapreecha
Student ID.	45160305
Degree	Doctor of Engineering
Program	Electrical Engineering
Year	2008
Thesis Advisor	Assoc. Prof. Dr. Kobchai Dejhan
Thesis Co-Advisor	Prof. Dr. Tian-Bo Deng

ABSTRACT

This thesis proposes the special matrices called Pascal matrices and applications to digital signal processing of both the signal filtering field and the signal transformation field, which are the main fields in digital signal processing area. The first proposal is called the generalized Pascal matrices for s-z transformation that is used in the field of filter design, the matrix used for variable mapping between s-domain to z-domain transfer function and vice versa by using an inverse Pascal matrix. The generalized Pascal matrices consist of forward difference (FD), backward difference (BD), bilinear (BL), and parametric BD-BL transformations, which depends on some parameters setting. The second, bilinear Pascal matrix is used for IIR digital filter design along with automatic frequency transformations from analog prototype normalized transfer function to lowpass, highpass, bandpass or bandstop filter by using the so-called frequency scaling coefficients (FSC) matrix. In this way, we can yield a new method for designing digital filters based on matrix operations. Next, consider the standard biquadratic transfer function that is normally used in analog domain for biquad filter design. However, in this thesis, the both design method and filter structure of the biquad digital filter is proposed, which can give five outputs simultaneously. The design equation for this type of filter uses only one matrix equation that is modified from the bilinear Pascal matrix, so called modified Pascal matrix for biquad digital filter design. For a signal transformation field that uses Pascal matrices, the so-called discrete Pascal transform (DPT) is studied. The Pascal matrices for this transform are different from the Pascal matrices used for s-z transformations. All the entries of these Pascal transformation matrices come from the binomial coefficients from Pascal's triangle directly. Also, this thesis investigates the frequency characteristic of DPT. In addition, the Pascal matrices for DPT are divided into two

types, i.e., highpass type, and lowpass type Pascal matrices, which are based on obtained frequency responses. Moreover, the hardware structure of DPT both of highpass and lowpass type based on DPT butterfly unit is shown. The DPT computational structure stems from the binary matrices, which are factorized from the DPT Pascal matrix and results in the overall structure for DPT and does not need any multipliers, only adders or subtractors are used for computation. The experiment of DPT on both 1-D and 2-D signal transformations leads to the development of the so-called discrete Pascal filter (DPF). There are two types of DPF the same as a division in DPT as lowpass and highpass type. The effect on output signal of DPF clearly shows more useful results of DPF than DPT in the sense of signal filtering. The real hardware implementation of both highpass and lowpass types of DPF will be integrated and can select the operation by selection bit. All hardware will be implemented on a filed programmable gate array (FPGA) especially, 2-D signal or digital image processing results can be shown on a VGA monitor directly from FPGA without any PC connection to a display unit.

ACKNOWLEDGEMENTS

This thesis is completed with various kind helps and supports from many persons. First, I would like to express my sincere thanks to my thesis advisor Assoc. Prof. Dr. Kobchai Dejhan for his teaching, advice, comment, warning, and practice until I realized that I could survive in my research fields. And I appreciate him very much for supporting me in every aspect since I started my study for the master degree at KMITL in 1999 until now, he provided me the opportunity to work in my desired job, and also took me to many international conferences, which opened my vision on international researches.

I would like to sincerely thank Prof. Dr. Tian-Bo Deng, who is my thesis co-advisor. The luck took me to meet him, first time that we met at Songkhla in 2003 at the international conference IEEE ISCIT 2003 and in next year again in Sendai at the conference ITC-CSCC 2004, where we had some discussions on my presented paper and from this point we started to know each other. In that year, we continued meeting by chance without appointment 2 times in Sapporo at ISCIT 2004 and by a great surprise at Chiang-Mai Night Bazaar with different conference ACRS 2004 for me and IEEE TENCON 2004 for him. Hence, we continue contact until ITC-CSCC 2006 (Chiang-Mai). After this conference, we started our joint research on Pascal matrices, and he supported me from that time until now. I thank him very much for his kind advices and comments to me about scientific research. He is my prototype to be a good researcher, as he often said to me that we should focus on one concentrated topic (problem) one-by-one and solve it step-by-step until it is completely finished, and should not be interrupted by others. These are his teachings that he often talks to me. He is more than a friend for me (he often said to me, we are friends). I want to say he is my teacher too.

I also would like to thank Assoc. Prof. Dr. Fusak Cheevasuvit, Assoc. Prof. Dr. Kanok Janchitrapongvej, Assoc. Prof. Dr. Pitikhate Sooraksa, and Assist. Prof. Dr. Phaophak Sirisuk who are my doctoral thesis defense committee members for their valuable comments and suggestions, which helped me improve my thesis greatly.

I am also grateful to my students for making me fun and exciting on some experiments in our projects and having our pleasant lab life's style. Many thanks are also go to master students of Assoc. Prof. Dr. Kobchai under my supervision as Mr. Wasan for helping me in many things, Mr. Ussanai for discussion and doing some simulations on Pascal transform for me based on my concept and idea, Ms. Theetima for helping me to arrange the paper format of my thesis.

I wish to thank Ms. Sudawadee Teocharoen very much for the long period of our friendship since 2003. Her voices over a very long distance from the so far places can make me have cheerfulness and powers in order to have the better and better in my life.

Finally, my sincere thanks are due to my family for their kind encouragements and supports on my education and my loving work. The usefulness that can happen from this thesis is dedicated to my passed away mother Somsri Chivapreecha who is my greatest love.

Sorawat Chivapreecha

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

Page

TABLE OF CONTENTS (continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF TABLES

Chapter 1

Introduction

1.1 Background and Literature Reviews

In this chapter, the background and literature reviews for each chapter of this thesis will be described. This thesis concerns Pascal matrices and applications of these matrices to digital signal processing both of using Pascal matrices for digital filter design and signal operations.

Chapter 2 : Generalized Pascal matrices for s-z Transformations will be proposed, the so-called s-z transformations provide a simple way for analyzing and designing discrete-time (DT) linear system in the z-domain through utilizing continuous-time (CT) linear systems in the s-domain, and vise versa [1-7]. For designing infinite-impulse-response (IIR) DT linear filters, it is the most common practice and successful way to convert a CT filter to a DT filter [1]. This is because CT filter design techniques have been highly advanced, thus the existing closed-form solutions and well-documented tables can be adopted directly for designing DT filters. One can design a DT filter with specified design requirements by performing s-z transformations, and the resulting DT filters can preserve the desired characteristics of the original CT filters [8-12]. On the other hand, CT filters are still important system components and useful in the applications where both CT and DT systems have to co-exist. If the required CT filter is difficult to design in the s-domain, a DT filter can be first designed, and then inverse conversion from z-to-s domain is applied to get a CT filter. Therefore, both s-to-z and z-to-s transformations, i.e., the mutual transformations between s-domain and z-domain are required.

In [2-4], it is shown that the bilinear (BL) transformation leads to a unique relation between the coefficients of the s-domain and z-domain polynomials (transfer function)

$$H_{CT}(s)=\sum_{n=0}^{N}A_n s_n \ , \qquad H(z)=\sum_{n=0}^{N}a_n z^{-n}$$

as

$$\mathbf{A}=\mathbf{P}\mathbf{a}\ , \qquad \mathbf{A}=\begin{bmatrix}A_0\\A_1\\\vdots\\A_N\end{bmatrix}, \qquad \mathbf{a}=\begin{bmatrix}a_0\\a_1\\\vdots\\a_N\end{bmatrix}$$

where $\mathbf{P}$ is called *Pascal matrix* that performs a one-to-one mapping between the z-domain coefficient vector $\mathbf{a}$ and s-domain coefficient vector $\mathbf{A}$. The inner elements of the Pascal matrix $\mathbf{P}$ can be recursively computed from the boundary elements by using recurrence formula [3]. For such a polynomial case, the s-z BL transformation also guarantees the one-to-one mapping between the polynomial values of $H_{CT}(s)$ and $H(z)$. However, for the IIR case, i.e., the rational polynomial case, assume that the s-domain and z-domain transfer function are

$$H_{CT}(s)=\frac{N_{CT}(s)}{D_{CT}(s)}=\frac{\sum_{n=0}^{N}A_n s^n}{\sum_{n=0}^{N}B_n s^n}$$

$$H(z)=\frac{N(z)}{D(z)}=\frac{\sum_{n=0}^{N}a_n z^{-n}}{\sum_{n=0}^{N}b_n z^{-n}}$$

respectively, the mapping between the numerators $N_{CT}(s)$ and $N(z)$ only [13], or the mapping between the denominators $D_{CT}(s)$ and $D(z)$ only does not yield a one-to-one mapping between the coefficients of s-domain and z-domain IIR transfer functions. Consequently, for inverse s-z transformation, the original coefficients cannot be recovered from the transformed coefficients through using the inverse Pascal matrix. That is, the forward and backward s-z transformations are not simply related by the Pascal matrix and its inverse, which will be stated in chapter 2.

Starting from the general model of the first-order s-z transformations introduced in [13], chapter 2 proposes a one-to-one coefficient mapping between the coefficients of the numerator pair $\{N_{CT}(s),N(z)\}$ or the denominator pair $\{D_{CT}(s),D(z)\}$, which has the following advantages.

1) The coefficients of the IIR CT filter $H_{CT}(s)$ can be uniquely mapped to the coefficients of a DT filter $H(z)$, and vice versa. The coefficients $\{A_n,B_n\}$ are related to $\{a_n,b_n\}$ in a closed-form through using the so-called generalized Pascal matrices and their inverses.

2) The inverses of the generalized Pascal matrices for various first-order s-z transformations can be analytically derived by utilizing the one-to-one mapping. Such proofs and derivations of the inverse generalized Pascal matrices cannot be found in the

literature. Although some of the inverse generalized Pascal matrices are given by a few researchers through observing some examples of the generalized Pascal matrices, no rigorous proofs are provided to support the results and convince the readers. Some observations even lead to incorrect conclusions, see the inverse of the generalized Pascal matrix for the FD case [13]. Moreover, the inverse generalized Pascal matrix for the parametric BD-BL case is not given and also apply the one-to-one coefficient mapping to derive closed-form expressions for the inverse generalized Pascal matrices and provide explicit formulas for various cases such as the bilinear (BL), backward-difference (BD), forward-difference (FD), and parametric BD-BL transformations.

Besides the above, a recurrence formula for computing the generalized Pascal matrix is also derived, which uses the general first-order s-z transformation model [13] and thus embeds the above BL, BD, FD, and parametric BD-BL transformations. As a result, the FFT-based numerical method proposed in [14] can be substituted by the new recurrence formula because the latter is analytical and more computationally efficient. The recurrence formula starts from the boundary elements (first row and first column) of the generalized Pascal matrix, and then computes the inner elements recursively, where only the neighboring elements are used.

Chapter 3 : Further Results on Generalized Pascal Matrices and Inverses Using a Unified s-z Transformation Model will be proposed, a more general first-order s-z transformation model in [17] is used to derive a more general Pascal matrix, which includes both lowpass-to-lowpass and lowpass-to-highpass s-z transformations. For simplicity, we call this s-z transformation model the unified s-z transformation model, and the resulting generalized Pascal matrix the *unified Pascal matrix*. As compared with the s-z transformation model in [13,16], the unified one uses one more parameter (three parameters in total) to incorporate the lowpass-to-highpass transformation. As a result, the unified Pascal matrix is more general than the generalized Pascal matrix [13,16], and parameterizes the backward-difference (BD), forward-difference (FD), lowpass-to-lowpass bilinear (BL1), lowpass-to-highass bilinear (BL2), and parametric BD-BL1 transformations. Although some inverses are mentioned in [17] through observing various unified Pascal matrices, no theoretical proofs are given, which even leads to an incorrect inverse for the FD case. In this chapter, using the unified s-z transformation model, rigorously prove the inverses of the unified Pascal matrices for the BD, FD, BL1, BL2, and parametric BD-BL1

transformations on the basis of one-to-one coefficient mapping of the CT domain and DT domain IIR transfer functions.

Chapter 4 : Pascal Matrix Operations for Unified Bilinear s-z Transformation will be proposed, the most successful approach of obtaining the coefficients of IIR filters is converting from analog filters that there already exists a wealth of information in the literature which can be utilized. The important method used to convert analog filters into equivalent digital filters is the bilinear transform [1, 18]. In [13-14,16] proposed using generalized Pascal matrix for various s-z transforms. This proposed method can obtain digital transfer function coefficients by easy matrix operations. In order to substitute the relationship between s-z into the transfer function directly, it is difficult to rearrange the transfer function to obtain coefficients when high orders and this only suitable to work out on paper, not suitable for computer programming. Also, this chapter proposes using the Pascal matrix based on bilinear transform for s-z transformation. Moreover, it can transform the frequency from normalized analog filter prototype to others which does not appear in [13-14] and covers more than in [8-9].

Chapter 5 : Modified Pascal Matrix for Biquad Digital Filter Design and Its Filter Structure Realization will be proposed, biquad filter is a filter which can give multiple-outputs at the same time. The biquad transfer functions are in the form of bi-quadratic equations. Generally, when we think of this type of filter, we often mean analog biquad filter [19] which can be implemented by using Op-Amp circuits or other analog devices. The previous papers [20-21] show that multiple-outputs digital filter in [20] has many different points compared to this chapter, such as the form of filter structure, analog prototype transfer functions and lack of ability to control a quality factor. In [21], the concept is based on combinations of FIR lowpass filter coefficients to give multiple-outputs and the filter structure is based on non-recursive scheme.

Chapter 5 proposes a new one-digital-type biquad filter. Both design concepts and the particular digital biquad filter structure for hardware realization in recursive scheme are introduced. The new ideas, which are used and proposed in this chapter, are s-z transformation from analog to digital transfer functions using bilinear-based Pascal matrix operations [8-14, 22] in order to compute the biquad digital filter coefficients, and a new proposed biquad digital filter structure which is modified from an ordinary 2^{nd} order IIR filter structure.

Chapter 6 : Discrete Pascal Transform and Its Hardware Realization will be proposed, the discrete Pascal transform (DPT) that was presented by [23] is one of many discrete transforms such as DFT (Discrete Fourier Transform), FFT (Fast Fourier Transform), DCT (Discrete Cosine

Transform), DWT (Discrete Wavelet Transform), etc. Most of these discrete transforms can be operated in the form of matrix operation as same as DPT. The interesting problem is that matrix operation consumes many multiplications and many adders for implementation, all of these depend on the size or dimension of matrix operator. With some properties of Pascal transform matrix where the elements in matrix operator are the binomial coefficients and related to the Pascal's triangle, we can factorize the Pascal transform matrix into binary (1,0,-1) matrices. Then we can represent Pascal transform matrix in the form of binary (1,0,-1) matrices, the transformed output, which is computed from Pascal transform matrix operation, can be computed without any multipliers. Only adders are used by using an efficient hardware structure that is obtained from these factorized binary (1,0,-1) matrices [24,26].

The Pascal transform matrix in this chapter is divided into two types that are called highpass type and lowpass type, respectively [25]. In [26] proposes only the efficient hardware realization of highpass type DPT that uses highpass type Pascal transform matrix, but in this chapter also proposes the efficient hardware realization of lowpass type DPT. Moreover, we also propose the elimination matrix that is used for formulating the factorized binary (1,0,-1) matrices both highpass and lowpass type which do not appear in [24,26]. Finally the efficient hardware realization of two-dimensional (2-D) DPT can be shown in the form of one-dimensional (1-D) DPT structure, and from the DPT hardware structure we will show the improvement of discrete Pascal filter (DPF) in both 1-D and 2-D case and both highpass and lowpass type.

Chapter 7 : The Discrete Pascal Filter (DPF) from Frequency Characteristic in DPT will be proposed, the discrete Pascal transform (DPT), that was proposed in [23-25] and in chapter 6 has an attraction for signal processing applications, especially application of the transform in digital image processing. Moreover, many previous papers introduced the advantages of the Pascal matrix for designing digital filter. As mentioned in [23-24], the DPT was used in bump and edge detection. So, it makes us think of the high frequency component in an image. Therefore, we make an assumption about the hidden highpass filtering characteristic in DPT. This chapter will show the method to investigate the frequency characteristic that is hidden in DPT. The DPT that uses Pascal matrix as shown in [23], we will call the highpass type DPT, and we also present another one that we will call the lowpass type DPT. All frequency characteristic can be shown in the frequency response of both 1-D and 2-D filtering. Moreover, we found that the operation of direct DPT, as proposed in [23-24] has weakness. From the frequency or filtering characteristic investigation, we can improve the operation method from DPT to discrete Pascal

filtering (DPF). The simulation results will show the differences between DPT and DPF output signal both highpass and lowpass 1-D and 2-D signal filtering and also proposes the applications of 2-DPF on hardware implementation.

1.2 Thesis Outlines

The organization of this thesis consists of 8 chapters. Chapter 1 is about research background and literature reviews that used in this thesis and the outlines of thesis. In chapter 2, the generalized Pascal matrices for s-z transformation topic, mathematic proofs shown on generalized Pascal matrices and their inverses in 4 cases as backward difference (BD), forward difference (FD), bilinear (BL) and parametric BD-BL transformation. Chapter 3 shows further results on generalized Pascal matrices for s-z transformation such as in chapter 2, but use a different model which is called unified Pascal matrix for derive. Chapter 4, the Pascal matrix operation for unified bilinear s-z transformation explains using of bilinear Pascal matrix for IIR filter design from analog prototype normalized transfer function with automatic frequency transformation by frequency scaling coefficients (FSC) matrix. Chapter 5, modified Pascal matrix used to design equation for biquad digital filter design and the proposed biquad digital filter structure can be shown which can give 5 outputs simultaneously. Chapter 6, the transform topic that called discrete Pascal transform (DPT), the basis function for this type of transform will be shown and the efficient way to create the hardware structure for DPT by Pascal matrix factorization into binary matrices is also shown. Chapter 7 is developed from chapter 6 to the so-called discrete Pascal filter (DPF), the frequency investigation of DPT leads to the development of DPF, the DPF hardware structure is same as the DPT only some input modification. Chapter 8 is the conclusion of overall research in this thesis.

Chapter 2

Generalized Pascal Matrices for s-z Transformations

2.1 Introduction

This chapter proposes a one-to-one mapping between the coefficients of continuous-time (s-domain) and discrete-time (z-domain) IIR transfer functions such that the s-domain numerator/denominator coefficients can be uniquely mapped to the z-domain numerator/denominator coefficients. The one-to-one mapping provides a firm basis for proving the inverses of the so-called generalized Pascal matrices from various first-order s-z transformations. Also, derive recurrence formulas for recursively determining the inner elements of the generalized Pascal matrices from their boundary ones. Consequently, all the elements of the whole generalized Pascal matrix can be easily generated through utilizing their neighborhood, which can be exploited for further simplifying the Pascal matrix generations. Finally, reveal and prove some interesting properties of the generalized Pascal matrices.

2.2 No One-to-One Mapping Problem

In this section, first consider why one-to-one mapping between the coefficients of IIR s-domain and z-domain transfer functions are necessary. If not, the inverse transformation (forward or backward s-z transformation) cannot restore the original coefficients. For simplicity, denote the s-to-z conversion as forward transformation, and z-to-s conversion as backward transformation. To derive correct one-to-one coefficient mapping between the coefficients of s-domain and z-domain IIR transfer functions and correct inverses of the generalized Pascal matrices for various first-order s-z transformations, need to perform both forward and backward s-z transformations first, which will be detailed in the following two subsections.

2.2.1 Forward s-z Transformation

Assume that the s-domain transfer function

$$H_{CT}(s)=\frac{\sum_{i=0}^{N}A_i s^i}{\sum_{i=0}^{N}B_i s^i}=\frac{N_{CT}(s)}{D_{CT}(s)} \tag{2.1}$$

with

$$N_{CT}(s)=\sum_{i=0}^{N}A_i s^i\,,\quad D_{CT}(s)=\sum_{i=0}^{N}B_i s^i$$

and z-domain transfer function

$$H(z)=\frac{\sum_{k=0}^{N}a_k z^{-k}}{\sum_{k=0}^{N}b_k z^{-k}}=\frac{\sum_{k=0}^{N}a_k z^{N-k}}{\sum_{k=0}^{N}b_k z^{N-k}}=\frac{N(z)}{D(z)} \tag{2.2}$$

with

$$N(z)=\sum_{k=0}^{N}a_k z^{N-k}\,,\quad D(z)=\sum_{k=0}^{N}b_k z^{N-k} \tag{2.3}$$

are related to each other through the first-order s-z transformation

$$s=(-u)\cdot\frac{z-1}{wz-v}\Leftrightarrow z=\frac{u+vs}{u+ws} \tag{2.4}$$

where $u,v\neq w$ are real constants [13]. Substituting (2.4) into (2.1) yields

$$\begin{aligned}\widehat{H}(z)&=\frac{\sum_{i=0}^{N}A_i(-u)^i\left(\frac{z-1}{wz-v}\right)^i}{\sum_{i=0}^{N}B_i(-u)^i\left(\frac{z-1}{wz-v}\right)^i}\\&=\frac{\sum_{i=0}^{N}u^iA_i\cdot(z-1)^i(v-wz)^{N-i}}{\sum_{i=0}^{N}u^iB_i\cdot(z-1)^i(v-wz)^{N-i}}\\&=\frac{\widehat{N}(z)}{\widehat{D}(z)}\end{aligned} \tag{2.5}$$

with

$$\widehat{N}(z)=\sum_{i=0}^{N}u^{i}A_{i}\cdot(-1)^{N-i}P_{i}(z)$$
$$\widehat{D}(z)=\sum_{i=0}^{N}u^{i}B_{i}\cdot(-1)^{N-i}P_{i}(z) \tag{2.6}$$
$$P_{i}(z)=(z-1)^{i}(wz-v)^{N-i}$$

Since $P_i(z)$ is the N th degree polynomial in z, it can be expressed as

$$P_{i}(z)=\sum_{k=0}^{N}p_{i,k}z^{N-k} \tag{2.7}$$

By using the binomial theorem, can expand $(z-1)^{i}$ and $(wz-v)^{N-i}$ as

$$(z-1)^{i}=\sum_{n=0}^{i}\binom{i}{n}z^{i-n}\cdot(-1)^{n}$$
$$(wz-v)^{N-i}=\sum_{n=0}^{N-i}\binom{N-i}{n}(wz)^{(N-i)-n}(-v)^{n}$$

To determine $p_{i,k}$, i.e., the coefficient of z^{N-k} in (2.7), need the consider the following combinations of the exponents of z :

$$z^{N-k}=\begin{cases} z^{N-i}\cdot z^{i-k} \\ z^{(N-i)-1}\cdot z^{i-(k-1)} \\ \vdots \\ z^{(N-i)-k}\cdot z^{i-(k-k)} \end{cases}$$

Thus,

$$\begin{aligned} p_{i,k}&=\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k-n}(-v)^{n}w^{(N-i)-n} \\ &=\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k}v^{n}w^{(N-i)-n} \end{aligned} \tag{2.8}$$

and

$$(-1)^{N-i}p_{i,k} = \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{N-i+k}v^n w^{(N-i)-n}$$
$$= \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}v^n w^{k-n}\cdot(-w)^{N-i-k}(-1)^{2k}$$
$$= \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}v^n w^{k-n}\cdot(-w)^{N-i-k}$$

Consequently, the numerator $\widehat{N}(z)$ in (2.5) can be expressed as

$$\widehat{N}(z) = \sum_{k=0}^{N}\hat{a}_k z^{N-k} \tag{2.9}$$

with

$$\hat{a}_k = \sum_{i=0}^{N}u^i A_i \cdot \underbrace{\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}v^n w^{k-n}}_{M(k,i)}\cdot(-w)^{N-i-k}$$
$$= \sum_{i=0}^{N}M(k,i)\widehat{A}_i\cdot(-w)^{N-i-k} \tag{2.10}$$

and

$$M(k,i) = \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}v^n w^{k-n} \tag{2.11}$$

$$\widehat{A}_i = u^i A_i$$

Using (2.11), can prove that

$$M(k,i) = \begin{cases} 1 \\ v^{N-i}w^i \\ \binom{N}{k}v^k \\ \binom{N}{k}w^k \end{cases} \tag{2.12}$$

Therefore, the matrix $\mathbf{M}$ takes the form

$$\mathbf{M}=\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ \binom{N}{1}v^1 & \cdots & \cdots & \cdots & \binom{N}{1}w^1 \\ \binom{N}{2}v^2 & \cdots & \cdots & \cdots & \binom{N}{2}w^2 \\ \vdots & \cdots & \cdots & \cdots & \vdots \\ \binom{N}{N}v^N & v^{N-1}w^1 & \cdots & v^1w^{N-1} & \binom{N}{N}w^N \end{bmatrix} \tag{2.13}$$

In [13], the matrix $\mathbf{M}$ is called the generalized Pascal matrix. For the BL, BD, and parametric BD-BL transformations ($w=-1$), (2.10) can be rewritten as

$$\hat{a}_k=\sum_{i=0}^{N}M(k,i)\hat{A}_i$$

or can express the above equation in matrix form as

$$\hat{\mathbf{a}}=\mathbf{M}\hat{\mathbf{A}} \tag{2.14}$$

with

$$\hat{\mathbf{a}}=\begin{bmatrix}\hat{a}_0\\ \hat{a}_1\\ \vdots\\ \hat{a}_N\end{bmatrix},\quad \hat{\mathbf{A}}=\begin{bmatrix}\hat{A}_0\\ \hat{A}_1\\ \vdots\\ \hat{A}_N\end{bmatrix}=\begin{bmatrix}u^0A_0\\ u^1A_1\\ \vdots\\ u^NA_N\end{bmatrix} \tag{2.15}$$

If we just equate the numerator $\hat{N}(z)$ in (2.9) to the numerator $N(z)$ in (2.3) as [13], i.e., $\hat{a}_k=a_k$, then the numerator coefficient vector $\mathbf{a}$ is related to the vector $\hat{\mathbf{A}}$ as

$$\hat{\mathbf{a}}=\mathbf{a}=\begin{bmatrix}a_0\\ a_1\\ \vdots\\ a_N\end{bmatrix}=\mathbf{M}\hat{\mathbf{A}} \tag{2.16}$$

we will show later that equating the numerators only will cause no one-to-one coefficient mapping and incorrect inverses of the generalized Pascal matrices.

2.2.2 Backward s-z Transformation

Conversely, we want to transform the z-domain transfer function

$$H(z)=\frac{\sum_{i=0}^{N} a_i z^{-i}}{\sum_{i=0}^{N} b_i z^{-i}}=\frac{\sum_{i=0}^{N} a_i z^{N-i}}{\sum_{i=0}^{N} b_i z^{N-i}}=\frac{N(z)}{D(z)} \tag{2.17}$$

to the s-domain transfer function

$$H_{CT}(s)=\frac{\sum_{k=0}^{N} A_k s^k}{\sum_{k=0}^{N} B_k s^k}=\frac{N_{CT}(s)}{D_{CT}(s)} \tag{2.18}$$

by using the first-order s-z transformation (2.4). Substituting

$$z=\frac{u+vs}{u+ws}$$

into (2.17) yields

$$\widehat{H}_{CT}(s)=\frac{\sum_{i=0}^{N} a_i \left(\frac{u+vs}{u+ws}\right)^{N-i}}{\sum_{i=0}^{N} b_i \left(\frac{u+vs}{u+ws}\right)^{N-i}}=\frac{\widehat{N}_{CT}(s)}{\widehat{D}_{CT}(s)} \tag{2.19}$$

with

$$\begin{aligned}\widehat{N}_{CT}(s)&=\sum_{i=0}^{N} a_i (u+vs)^{N-i}(u+ws)^{i}\\ \widehat{D}_{CT}(s)&=\sum_{i=0}^{N} b_i (u+vs)^{N-i}(u+ws)^{i}\end{aligned} \tag{2.20}$$

Using binomial series expansion of $(u+vs)^{N-i}$ and $(u+ws)^{i}$, can obtain

$$\widehat{N}_{CT}(s) = \sum_{k=0}^{N} \widetilde{A}_k s^k \tag{2.21}$$

with

$$\widetilde{A}_k = u^{N-k} \sum_{i=0}^{N} M(k,i) a_i \tag{2.22}$$

In [13], the numerator $\widehat{N}_{CT}(s)$ is equated to $N_{CT}(s)$ in (2.18), i.e., $\widetilde{A}_k = A_k$, which leads to

$$u^{-N}(u^k A_k) = \sum_{i=0}^{N} M(k,i) a_i$$

i.e.,

$$u^{-N} \widehat{A}_k = \sum_{i=0}^{N} M(k,i) a_i$$

with $\widehat{A}_k = u^k A_k$. Thus,

$$u^{-N} \widehat{\mathbf{A}} = \mathbf{M}\mathbf{a} \tag{2.23}$$

i.e.,

$$\mathbf{a} = u^{-N}(\mathbf{M}^{-1} \widehat{\mathbf{A}}) \tag{2.24}$$

For the BL transformation, comparing (2.16) with (2.24) leads to

$$\mathbf{M}_{BL}^{-1} = u^N \mathbf{M}_{BL} \tag{2.25}$$

where $\mathbf{M}_{BL}$ denotes the generalized Pascal matrix for the BL case, and its elements are

$$M_{BL}(k,i) = \sum_{n=0}^{k} \binom{N-i}{n} \binom{i}{k-n} (-1)^{k-n} \tag{2.26}$$

It should be pointed out here that the inverse (2.25) is incorrect, the correct one is

$$\mathbf{M}_{BL}^{-1} = 2^{-N}\mathbf{M}_{BL} \tag{2.27}$$

The proof is given in proof of inverse Pascal matrix section. The reason for the incorrect relation (2.25) between the generalized Pascal matrix and its inverse is due to the direct mapping between the numerators only or between the denominators only. This means that we cannot restore the original s-domain coefficients $\{A_n, B_n\}$ from the z-domain coefficients $\{a_n, b_n\}$, or vice versa, i.e., the coefficient mapping is not one-to-one mapping. More importantly, it misleads to wrong inverses of the generalized Pascal matrices for various transformations such as BD, FD cases and cannot rigorously prove the inverse generalized Pascal matrix for the BD-BL parametric transformation, which will be shown later. For example, see the example presented in [8], where the transfer function of the third-order ($N = 3$) s-domain lowpass filter is

$$H_{CT}(s) = \frac{5.153 + s^2}{5.153 + 4.344s + 2.781s^2 + 0.929s^3} \tag{2.28}$$

For simplicity, assume $u = 2/T = 1$, which leads to

$$\widehat{\mathbf{A}} = \mathbf{A} = \begin{bmatrix} 5.153 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \widehat{\mathbf{B}} = \mathbf{B} = \begin{bmatrix} 5.153 \\ 4.344 \\ 2.781 \\ 0.929 \end{bmatrix}$$

as shown in (2.15). If (2.16) is used, then

$$\begin{aligned} [\mathbf{a} \quad \mathbf{b}] &= \mathbf{M}_{BL}\left[\widehat{\mathbf{A}} \quad \widehat{\mathbf{B}}\right] \\ &= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 3 & 1 & -1 & -3 \\ 3 & -1 & -1 & 3 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 5.153 & 5.153 \\ 0 & 4.344 \\ 1 & 2.781 \\ 0 & 0.929 \end{bmatrix} \\ &= \begin{bmatrix} 6.153 & 13.207 \\ 14.459 & 14.235 \\ 14.459 & 11.121 \\ 6.153 & 2.661 \end{bmatrix} \end{aligned} \tag{2.29}$$

i.e., the z-domain transfer function is

$$\widehat{H}(z)=\frac{6.153+14.459z^{-1}+14.459z^{-2}+6.153z^{-3}}{13.207+14.235z^{-1}+11.121z^{-2}+2.661z^{-3}}$$

For the backward s-z transformation, if (2.23) is used as [13], then

$$\begin{aligned}\left[\widehat{\mathbf{A}}\quad\widehat{\mathbf{B}}\right]&=\mathbf{M}_{BL}[\mathbf{a}\quad\mathbf{b}]\\&=\begin{bmatrix}1&1&1&1\\3&1&-1&-3\\3&-1&-1&3\\1&-1&1&-1\end{bmatrix}\begin{bmatrix}6.153&13.207\\14.459&14.235\\14.459&11.121\\6.153&2.661\end{bmatrix}\\&=\begin{bmatrix}41.224&41.224\\0&34.752\\8&22.248\\0&7.432\end{bmatrix}\end{aligned}$$

i.e.,

$$\widehat{H}_{CT}(s)=\frac{41.224+8s^2}{41.224+34.752s+22.248s^2+7.432s^3}$$

Clearly, both the numerator and denominator coefficients of $\widehat{H}_{CT}(s)$ are different from those of $H_{CT}(s)$ in (2.28). In contrast, if (2.24) is used, then

$$\begin{aligned}[\mathbf{a}\quad\mathbf{b}]&=\mathbf{M}_{BL}^{-1}\left[\widehat{\mathbf{A}}\quad\widehat{\mathbf{B}}\right]\\&=2^{-3}\mathbf{M}_{BL}[\mathbf{A}\quad\mathbf{B}]\\&=\begin{bmatrix}0.7691&1.6509\\1.8074&1.7794\\1.8074&1.3901\\0.7691&0.3326\end{bmatrix}\end{aligned}\qquad(2.30)$$

i.e., the z-domain transfer function becomes

$$\widehat{H}(z)=\frac{0.7691+1.8074z^{-1}+1.8074z^{-2}+0.7691z^{-3}}{1.6509+1.7794z^{-1}+1.3901z^{-2}+0.3326z^{-3}}$$

and conversely

$$\left[\widehat{\mathbf{A}}\quad\widehat{\mathbf{B}}\right]=\mathbf{M}_{BL}[\mathbf{a}\quad\mathbf{b}]=\begin{bmatrix}5.153 & 5.153\\ 0 & 4.344\\ 1 & 2.781\\ 0 & 0.929\end{bmatrix}$$

i.e.,

$$\widehat{H}_{CT}(s)=\frac{5.153+s^2}{5.153+4.344s+2.781s^2+0.929s^3}$$

Obviously, the coefficients of $\widehat{H}_{CT}(s)$ here are identical to those of $H_{CT}(s)$ in (2.28). Therefore, the original s-domain coefficients can be completely restored.

2.3 Inverse Matrices Using One-To-One Mapping

The incorrect conclusion of the inverse Pascal matrix (2.25) is caused by equating only numerators. Let us go back to (2.19), where a new s-domain transfer function $\widehat{H}_{CT}(s)$ is transformed from the z-domain transfer function $H(z)$ in (2.17). To perform the first-order s-z transformation (2.4), the constraint

$$s=0 \Leftrightarrow z=1$$

is imposed, i.e., the point $s=0$ in the s-plane must be transformed into the $z=1$ in the z-plane [13]. It follows from (2.19) that

$$\widehat{H}_{CT}(s)\Big|_{s=0}=\frac{\sum_{i=0}^{N}a_iu^{N-i}\cdot u^i}{\sum_{i=0}^{N}b_iu^{N-i}\cdot u^i}=\frac{\sum_{i=0}^{N}a_iu^N}{\sum_{i=0}^{N}b_iu^N} \tag{2.31}$$

On the other hand, obtain from (2.17) that

$$H(z)\big|_{z=1} = \frac{\sum_{i=0}^{N} a_i}{\sum_{i=0}^{N} b_i} \tag{2.32}$$

Comparing (2.31) with (2.32) indicates that both the numerator and denominator coefficients of $H(z)$ are scaled by a factor u^N after the backward *s*-*z* transformation. However, if we want to derive a one-to-one mapping between the coefficients of the numerators $N_{CT}(s)$ in (2.18) and $N(z)$ in (2.17), the correspondence

$$\begin{aligned} N(z)\big|_{z=1} &= \sum_{i=0}^{N} a_i z^{N-i}\Big|_{z=1} = \sum_{i=0}^{N} a_i \\ &\Updownarrow \\ N_{CT}(s)\big|_{s=0} &= \sum_{k=0}^{N} A_k s^k\Big|_{s=0} = \sum_{k=0}^{N} A_k 0^k = A_0 \end{aligned} \tag{2.33}$$

must be satisfied. Therefore, need to divide $\widetilde{A}_k$ in (2.22) by u^N and then set the result to A_k as

$$\frac{\widetilde{A}_k}{u^N} = u^{-k}\sum_{i=0}^{N} M(k,i)\,a_i = A_k \tag{2.34}$$

As a result, we have

$$\widehat{A}_k = u^k A_k = \sum_{i=0}^{N} M(k,i) a_i$$

or in matrix form

$$\widehat{\mathbf{A}} = \mathbf{M}\mathbf{a} \tag{2.35}$$

namely,

$$\mathbf{a} = \mathbf{M}^{-1}\widehat{\mathbf{A}} \tag{2.36}$$

Next, go back to (2.5), where a new *z*-domain transfer function $\widehat{H}_{CT}(z)$ is transformed from the *s*-domain transfer function $H_{CT}(s)$ in (2.1). Since

$$\widehat{H}(z)\Big|_{z=1} = \frac{\sum_{i=0}^{N} u^i A_i \cdot 0^i (v-w)^{N-i}}{\sum_{i=0}^{N} u^i B_i \cdot 0^i (v-w)^{N-i}} = \frac{A_0 (v-w)^N}{B_0 (v-w)^N} \tag{2.37}$$

and

$$H_{CT}(s)\Big|_{s=0} = \frac{\sum_{i=0}^{N} A_i 0^i}{\sum_{i=0}^{N} B_i 0^i} = \frac{A_0}{B_0} \tag{2.38}$$

we know that the numerator and denominator coefficients of $H_{CT}(s)$ are scaled by a factor $(v-w)^N$ after the forward *s*-*z* transformation. Therefore, the coefficients $\hat{a}_k$ in (2.10) are related to the coefficients a_k in (2.2) as

$$\frac{\hat{a}_k}{(v-w)^N} = a_k \tag{2.39}$$

2.3.1 Bilinear (BL) Transformation

For the BL transformation, the parameters u, v, and w take the values

$$\begin{cases} u = \frac{2}{T}, & T : \textit{sampling period} \\ v = 1 \\ w = -1 \end{cases}$$

and the generalized Pascal matrix reduces to the so-called Pascal matrix, whose elements are defined as (2.26). For example, if $N = 4$, then the Pascal matrix is

$$\mathbf{M}_{BL} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 0 & -2 & -4 \\ 6 & 0 & -2 & 0 & 6 \\ 4 & -2 & 0 & 2 & -4 \\ 1 & -1 & 1 & -1 & 1 \end{bmatrix} \tag{2.40}$$

As for the inverse Pascal matrix, substituting $v=1$ and $w=-1$ into (2.39) yields $a_k = 2^{-N}\hat{a}_k$, i.e.,

$$\mathbf{a} = 2^{-N}\hat{\mathbf{a}} \tag{2.41}$$

Substituting (2.14) into (2.41) results in

$$\mathbf{a} = 2^{-N}\mathbf{M}_{BL}\hat{\mathbf{A}} \tag{2.42}$$

and comparing (2.36) and (2.42) yields the inverse Pascal matrix (2.27) for the BL case.

2.3.2. Backward-Difference (BD) Transformation

For the BD transformation, the parameters u, v, and w take the values

$$\begin{cases} u = \frac{2}{T}, & T: \textit{sampling period} \\ v = 0 \\ w = -1 \end{cases}$$

Using the general definition (2.11), can compute the elements of the generalized Pascal matrix as

$$M_{BD}(k,i) = \sum_{n=0}^{k} \binom{N-i}{n}\binom{i}{k-n} 0^n (-1)^{k-n} = \binom{i}{k}(-1)^k \tag{2.43}$$

For $k > i$, $M_{BD}(k,i) = 0$, and for $k = i$, $M_{BD}(k,i) = (-1)^k$. For example, if $N = 4$, then

$$\mathbf{M}_{BD} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & -1 & -2 & -3 & -4 \\ 0 & 0 & 1 & 3 & 6 \\ 0 & 0 & 0 & -1 & -4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.44}$$

As for the inverse of the generalized Pascal matrix, substituting $v=0$ and $w=-1$ into (2.39) yields $\hat{a}_k = a_k$, i.e.,

$$\mathbf{a} = \hat{\mathbf{a}} \tag{2.45}$$

Substituting (2.14) into (2.45) obtains

$$\mathbf{a} = \mathbf{M}_{BD}\,\widehat{\mathbf{A}} \tag{2.46}$$

and comparing (2.36) and (2.46) yields the inverse matrix for the BD case as

$$\mathbf{M}_{BD}^{-1} = \mathbf{M}_{BD} \tag{2.47}$$

2.3.3. Forward-Difference (FD) Transformation

For the FD transformation, the parameters u, v, and w take the values

$$\begin{cases} u = \frac{2}{T}, & T : \textit{sampling period} \\ v = 1 \\ w = 0 \end{cases}$$

By using the general definition (2.11), can express the elements of the generalized Pascal matrix as

$$M_{FD}(k,i) = \sum_{n=0}^{k} \binom{N-i}{n}\binom{i}{k-n} \cdot 1^{n} \cdot 0^{k-n} = \binom{N-i}{k} \tag{2.48}$$

For $k > N-i$, i.e., $k+i > N$, $M_{FD}(k,i) = 0$, and for $k+i = N$, $M_{FD}(k,i) = 1$. As for the inverse matrix, substituting $v = 1$ and $w = 0$ into (2.39) yields $\hat{a}_k = a_k$. The $\hat{a}_k$ in (2.10) can be further manipulated as

$$\begin{aligned} \hat{a}_k &= \sum_{i=0}^{N} \widehat{A}_i \cdot \sum_{n=0}^{k} \binom{N-i}{n}\binom{i}{k-n} v^{n} w^{(N-i)-n} \cdot (-1)^{N-(k+i)} \\ &= \sum_{i=0}^{N} \widehat{A}_i \cdot \sum_{n=0}^{k} \binom{N-i}{n}\binom{i}{k-n} \cdot 0^{(N-i)-n} \cdot (-1)^{N-(k+i)} \\ &= \sum_{i=0}^{N} \widehat{A}_i \cdot \binom{N-i}{N-i}\binom{i}{k-(N-i)} (-1)^{(k+i)-N} \\ &= \sum_{i=0}^{N} \widehat{A}_i \cdot \binom{i}{N-k} (-1)^{(k+i)-N} \\ &= \sum_{i=0}^{N} W(k,i)\,\widehat{A}_i \end{aligned} \tag{2.49}$$

with

$$W(k,i)=\binom{i}{N-k}(-1)^{(k+i)-N} \tag{2.50}$$

The relation (2.49) can be expressed in matrix form as

$$\hat{\mathbf{a}}=\mathbf{a}=\mathbf{W}\hat{\mathbf{A}} \tag{2.51}$$

Comparing (2.36) with (2.51) yields the inverse Pascal matrix

$$\mathbf{M}_{FD}^{-1}=\mathbf{W} \tag{2.52}$$

To get a closed-form solution for $\mathbf{W}$, compare $M_{FD}(k,i)$ in (2.48) with $W(k,i)$ in (2.50) and conclude that the matrix $\mathbf{W}$ can be obtained from the matrix $\mathbf{M}_{FD}$ by the following steps.

Step-1: Flip $\mathbf{M}_{FD}$ in up/down direction to get a new matrix $\mathbf{U}$;

Step-2: Flip $\mathbf{U}$ in left/right direction to get a new matrix $\mathbf{V}$;

Step-3: Multiply the element of the matrix $\mathbf{V}$ in (k,i) position by $(-1)^{(k+i)-N}$ to get the final matrix $\mathbf{W}$, i.e.,

$$W(k,i)=(-1)^{(k+i)-N}V(k,i) \tag{2.53}$$

where (k,i) represent the row and column positions, respectively, $k,i=0,1,\ldots,N$.

For example, if $N=4$, then the generalized Pascal matrix and its inverse are

$$\mathbf{M}_{FD}=\begin{bmatrix}1&1&1&1&1\\4&3&2&1&0\\6&3&1&0&0\\4&1&0&0&0\\1&0&0&0&0\end{bmatrix},\quad \mathbf{M}_{FD}^{-1}=\begin{bmatrix}0&0&0&0&1\\0&0&0&1&-4\\0&0&1&-3&6\\0&1&-2&3&-4\\1&-1&1&-1&1\end{bmatrix}$$

It should be noted here that the inverse Pascal matrix for the FD case is mentioned in [13], where $W(k,i)$ is computed by

$$W(k,i) = (-1)^{k+i+1} V(k,i) \tag{2.55}$$

Unfortunately, the conclusion (2.55) is incorrect for the general case, which is only valid for odd N. This is because

$$(-1)^{(k+i)-N} = (-1)^{(k+i+1)} \cdot (-1)^{-(N+1)} = \begin{cases} (-1)^{k+i+1} & \text{for odd } N \\ -(-1)^{k+i+1} & \text{for even } N \end{cases}$$

2.3.4. Parametric BD-BL Transformation

For the BD-BL transformation, the parameters u, v, and w take the values

$$\begin{cases} u = \frac{1+r}{T}, & T : \textit{sampling period} \\ v = r \\ w = -1 \end{cases}$$

By using the general definition (2.11), can compute the elements of the generalized Pascal matrix as

$$M_p(k,i) = \sum_{n=0}^{k} \binom{N-i}{n} \binom{i}{k-n} r^n (-1)^{k-n} \tag{2.56}$$

Also, substituting $v = r$ and $w = -1$ into (2.39) yields $a_k = (r+1)^{-N} \hat{a}_k$, i.e.,

$$\mathbf{a} = (r+1)^{-N} \hat{\mathbf{a}} \tag{2.57}$$

Substituting (2.14) into (2.57) yields

$$\mathbf{a} = (r+1)^{-N} \mathbf{M}_p \widehat{\mathbf{A}} \tag{2.58}$$

and comparing (2.36) and (2.58) obtains the inverse matrix for the parametric BD-BL case as

$$\mathbf{M}_P^{-1} = (r+1)^{-N}\mathbf{M}_P \tag{2.59}$$

As special cases, if $r = 0$, then the parametric BD-BL transformation becomes the BD transformation, and if $r = 1$, it becomes the BL transformation. For other r, the inverse matrix $\mathbf{M}_P^{-1}$ can be obtained by scaling the matrix $\mathbf{M}_P$ by a factor $(r+1)^{-N}$. As an example, if $N = 4$ and $r = 0.5$, then the generalized Pascal matrix is

$$\mathbf{M}_P = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 2 & 0.5 & -1 & -2.5 & -4 \\ 1.5 & -0.75 & -0.75 & 1.5 & 6 \\ 0.5 & -0.625 & 0.5 & 0.5 & -4 \\ 0.0625 & -0.125 & 0.25 & -0.5 & 1 \end{bmatrix} \tag{2.60}$$

and its inverse is

$$\mathbf{M}_P^{-1} = \begin{bmatrix} 0.1975 & 0.1975 & 0.1975 & 0.1975 & 0.1975 \\ 0.3951 & 0.0988 & -0.1975 & -0.4938 & -0.7901 \\ 0.2963 & -0.1481 & -0.1481 & 0.2963 & 1.1852 \\ 0.0988 & -0.1235 & 0.0988 & 0.0988 & -0.7901 \\ 0.0123 & -0.0247 & 0.0494 & -0.0988 & 0.1975 \end{bmatrix}$$

Thus, we can verify

$$\mathbf{M}_P\mathbf{M}_P^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

2.4 Lowpass-To-Highpass Transformation

So far, we detailed the first-order *s*-*z* transformation (2.4), which is a model for lowpass-to-lowpass mapping, i.e., $s = 0 \Leftrightarrow z = 1$. In this section, consider the lowpass-to-highpass BL transformation, i.e., $s = 0 \Leftrightarrow z = -1$. Therefore, the *s*-*z* transformation model is

$$z=-\frac{u+s}{u-s}\Leftrightarrow s=u\cdot\frac{z+1}{z-1} \tag{2.61}$$

where u is a real constant. In the following subsections, first consider the one-to-one coefficient mapping from the s-domain transfer function to the z-domain transfer function (forward s-z transfer function), and then consider that from the z-domain to s-domain (backward s-z transformation). Based on the s-to-z and z-to-s one-to-one mapping, can derive a closed-form inverse Pascal matrix for the lowpass-to-highpass case. It will be seen that the inverse Pascal matrix for this lowpass-to-highpass BL case can be easily obtained by modifying the Pascal matrix in the lowpass-to-lowpass BL case (2.26).

2.4.1. s-to-z Transformation

Substituting $s=u(z+1)/(z-1)$ into (2.1) yields

$$\begin{aligned}\widehat{H}(z)&=\frac{\sum_{i=0}^{N}u^iA_i\left(\frac{z+1}{z-1}\right)^i}{\sum_{i=0}^{N}u^iA_i\left(\frac{z+1}{z-1}\right)^i}\\&=\frac{\sum_{i=0}^{N}u^iA_i(z+1)^i(z-1)^{N-i}}{\sum_{i=0}^{N}u^iB_i(z+1)^i(z-1)^{N-i}}\\&=\frac{\widehat{N}(z)}{\widehat{D}(z)}\end{aligned} \tag{2.62}$$

with

$$\widehat{N}(z)=\sum_{i=0}^{N}u^iA_iP_i(z),\quad \widehat{D}(z)=\sum_{i=0}^{N}u^iB_iP_i(z)$$

and

$$P_i(z)=(z+1)^i(z-1)^{N-i}$$

Using the binomial expansions

$$(z+1)^i=\sum_{n=0}^{i}\binom{i}{n}z^{i-n}$$

$$(z-1)^{N-i}=\sum_{n=0}^{N-i}\binom{N-i}{n}z^{(N-i)-n}\cdot(-1)^n$$

the numerator $\widehat{N}(z)$ can be expressed as

$$\widehat{N}(z)=\sum_{k=0}^{N}\hat{a}_k z^{N-k} \tag{2.63}$$

with

$$\hat{a}_k=\sum_{i=0}^{N}u^i A_i\cdot\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^n$$

Here, we want to perform the one-to-one mapping from the numerator $N_{CT}(s)$ in (2.1) to $\widehat{N}(z)$, i.e.,

$$N_{CT}(s)\ \Leftrightarrow\ \widehat{N}(z)$$

on the basis $s=0\ \Leftrightarrow\ z=-1$. Since

$$H_{CT}(s)\big|_{s=0}=\frac{\sum_{i=0}^{N}A_i\cdot 0^i}{\sum_{i=0}^{N}B_i\cdot 0^i}=\frac{A_0}{B_0} \tag{2.64}$$

and

$$\begin{aligned}\widehat{H}(z)\big|_{z=-1}&=\frac{\sum_{i=0}^{N}u^i A_i\cdot 0^i\cdot(-2)^{N-i}}{\sum_{i=0}^{N}u^i B_i\cdot 0^i\cdot(-2)^{N-i}}\\&=\frac{A_0\cdot(-2)^N}{B_0\cdot(-2)^N}\\&=\frac{A_0\cdot 2^N}{B_0\cdot 2^N}\end{aligned} \tag{2.65}$$

it is clear that both the numerator and denominator coefficients of $H_{CT}(s)$ are scaled by a factor 2^N after the s-to-z transformation. Hence, we have

$$\frac{\hat{a}_k}{2^N}=2^{-N}\hat{a}_k=a_k$$

i.e.,

$$2^{-N}\sum_{i=0}^{N}u^{i}A_{i}\cdot\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{n}=a_{k} \tag{2.66}$$

Further, since $(-1)^{n}=(-1)^{k-n}\cdot(-1)^{k}$, the relation (2.66) can be rewritten as

$$2^{-N}\sum_{i=0}^{N}(-1)^{k}M_{BL}(k,i)\widehat{A}_{i}=a_{k} \tag{2.67}$$

with $\widehat{A}_{i}=u^{i}A_{i}$, and $M_{BL}(k,i)$ is defined in (2.26) for the lowpass-to-lowpass BL case. If we let

$$\widetilde{M}(k,i)=(-1)^{k}M_{BL}(k,i) \tag{2.68}$$

then (2.67) can be rewritten in the matrix form

$$\mathbf{a}=2^{-N}\widetilde{\mathbf{M}}\widehat{\mathbf{A}}$$

or

$$\widehat{\mathbf{A}}=2^{N}\widetilde{\mathbf{M}}^{-1}\mathbf{a} \tag{2.69}$$

Eq. (2.68) indicates that the matrix $\widetilde{\mathbf{M}}$ can be easily obtained from $\mathbf{M}_{BL}$ by multiplying the rows of $\mathbf{M}_{BL}$ with $(-1)^{k}$, $k=0,1,\ldots,N$. As an example, for $N=3$, we have the Pascal matrix

$$\mathbf{M}_{BL}=\begin{bmatrix}1&1&1&1\\3&1&-1&-3\\3&-1&-1&3\\1&-1&1&-1\end{bmatrix} \tag{2.70}$$

then matrix $\widetilde{\mathbf{M}}$ can be obtained by changing the signs of the second row $(k=1)$ and fourth row $(k=3)$ of $\mathbf{M}_{BL}$ as

$$\widetilde{\mathbf{M}}=\begin{bmatrix}1&1&1&1\\-3&-1&1&3\\3&-1&-1&3\\-1&1&-1&1\end{bmatrix} \tag{2.71}$$

Instead of changing the signs, the generalized Pascal matrix $\widetilde{\mathbf{M}}$ can also be obtained by flipping the columns of $\mathbf{M}_{BL}$. This can be proved as follows. It follows from (2.66) that

$$\widetilde{M}(k,i)=\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^n$$

Thus,

$$\widetilde{M}(k,N-i)=\sum_{n=0}^{k}\binom{i}{n}\binom{N-i}{k-n}(-1)^n$$

Let $n'=k-n$, then

$$\begin{aligned}\widetilde{M}(k,N-i)&=\sum_{n'=k}^{0}\binom{i}{k-n'}\binom{N-i}{n'}(-1)^{k-n'}\\&=\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k-n}\\&=M_{BL}(k,i)\end{aligned}$$

Therefore, flipping the columns of $\mathbf{M}_{BL}$ can also yield $\widetilde{\mathbf{M}}$.

2.4.2. z-to-s Transformation

Substituting $z=-(u+s)/(u-s)$ into (2.17) leads to

$$\begin{aligned}\widehat{H}_{CT}(s)&=H(z)\Big|_{z=-\frac{u+s}{u-s}}\\&=\frac{\sum_{i=0}^{N}(-1)^{N-i}a_i\left(\frac{u+s}{u-s}\right)^{N-i}}{\sum_{i=0}^{N}(-1)^{N-i}b_i\left(\frac{u+s}{u-s}\right)^{N-i}}\\&=\frac{\sum_{i=0}^{N}(-1)^{i}a_i(u+s)^{N-i}(u-s)^{i}}{\sum_{i=0}^{N}(-1)^{i}b_i(u+s)^{N-i}(u-s)^{i}}\\&=\frac{\widehat{N}_{CT}(s)}{\widehat{D}_{CT}(s)}\end{aligned}$$

with

$$\widehat{N}_{CT}(s)=\sum_{i=0}^{N}(-1)^i a_i Q_i(s), \qquad \widehat{D}_{CT}(s)=\sum_{i=0}^{N}(-1)^i b_i Q_i(s) \tag{2.72}$$

and

$$Q_i(s)=(u+s)^{N-i}(u-s)^i$$

Here, we want to expand the Nth degree polynomial $Q_i(s)$ as

$$Q_i(s)=\sum_{k=0}^{N} q_{i,k}s^k$$

Since

$$\begin{aligned}(u+s)^{N-i}&=\sum_{n=0}^{N-i}\binom{N-i}{n}u^{(N-i)-n}s^n\\ (u-s)^i&=\sum_{n=0}^{i}\binom{i}{n}u^{i-n}(-s)^n\end{aligned} \tag{2.73}$$

the coefficient $q_{i,k}$ can be determined by considering the following combinations of the exponents of s:

$$s^k=\begin{cases}s^0\cdot s^k\\ s^1\cdot s^{k-1}\\ s^2\cdot s^{k-2}\\ \vdots\\ s^k\cdot s^{k-k}\end{cases}$$

Thus, we can get

$$\begin{aligned}q_{i,k}&=\sum_{n=0}^{k}\binom{N-i}{n}u^{(N-i)-n}\cdot\binom{i}{k-n}u^{i-(k-n)}(-1)^{k-n}\\ &=u^{N-k}\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k-n}\\ &=u^{N-k}M_{BL}(k,i)\end{aligned}$$

Consequently, the numerator $\widehat{N}_{CT}(s)$ in (2.72) can be expressed as

$$\widehat{N}_{CT}(s)=\sum_{k=0}^{N}\widetilde{A}_k s^k$$

with

$$\widetilde{A}_k = \sum_{i=0}^{N} (-1)^i a_i q_{i,k}$$
$$= u^{N-k} \sum_{i=0}^{N} (-1)^i M_{BL}(k,i) a_i$$

Consider the mapping $s = 0 \Leftrightarrow z = -1$, we have

$$\widehat{H}_{CT}(s)\Big|_{s=0} = \frac{\sum_{i=0}^{N} (-1)^i a_i \cdot u^N}{\sum_{i=0}^{N} (-1)^i b_i \cdot u^N}$$

$$H(z)\Big|_{z=-1} = \frac{\sum_{i=0}^{N} (-1)^{N-i} a_i}{\sum_{i=0}^{N} (-1)^{N-i} b_i} = \frac{\sum_{i=0}^{N} (-1)^i a_i}{\sum_{i=0}^{N} (-1)^i b_i}$$

Therefore, both the numerator and denominator coefficients of $H(z)$ in (2.17) have been scaled by a factor u^N after the z-to-s transformation, which implies

$$\frac{\widetilde{A}_k}{u^N} = u^{-k} \sum_{i=0}^{N} (-1)^i M_{BL}(k,i) a_i = A_k$$

i.e.,

$$\widehat{A}_k = u^k A_k = \sum_{i=0}^{N} (-1)^i M_{BL}(k,i) a_i \tag{2.74}$$

If we let

$$\widehat{M}_{k,i} = (-1)^i M_{BL}(k,i) \tag{2.75}$$

then (2.74) can be expressed in matrix form as

$$\widehat{\mathbf{A}} = \widehat{\mathbf{M}}\mathbf{a} \tag{2.76}$$

Eq. (2.75) indicates that the matrix $\widehat{\mathbf{M}}$ can be easily obtained by multiplying the columns of $\mathbf{M}_{BL}$ with $(-1)^i, i = 0,1,\ldots,N$, i.e., the even columns (odd i) will change their signs. For example, if $N = 3$, we have $\mathbf{M}_{BL}$ given in (2.70) and $\widehat{\mathbf{M}}$ can be obtained by changing the signs of the second column (i=1) and fourth column (i=3) of $\mathbf{M}_{BL}$ as

$$\widehat{\mathbf{M}} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 3 & -1 & -1 & 3 \\ 3 & 1 & -1 & -3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.77}$$

Instead of changing the signs of even columns of the Pascal matrix $\mathbf{M}_{BL}$ for the lowpass-to-lowpass BL case, the generalized Pascal matrix $\widehat{\mathbf{M}}$ can also be obtained by flipping the rows of $\mathbf{M}_{BL}$, which can be proved as follows. Replacing k in (2.26) with $N-k$ yields

$$M_{BL}(N-k,i) = \sum_{n=0}^{N-k} \binom{N-i}{n} \binom{i}{(N-k)-n} (-1)^{(N-k)-n}$$

Since $i \geq (N-k)-n$, i.e., $n \geq (N-k)-i, M_{BL}(N-k,i)$ can be written as

$$M_{BL}(N-k,i) = \sum_{n=(N-k)-i}^{N-k} \binom{N-i}{n} \binom{i}{(N-k)-n} (-1)^{(N-k)-n}$$

Let $n' = (N-k)-n$, then $n = (N-k)-n'$, and

$$\begin{aligned} M_{BL}(N-k,i) &= \sum_{n'=i}^{0} \binom{N-i}{(N-k)-n'} \binom{i}{n'} (-1)^{n'} \\ &= \sum_{n=0}^{i} \binom{N-i}{k+n-i} \binom{i}{i-n} (-1)^{n} \end{aligned} \tag{2.78}$$

On the other hand, since

$$\begin{aligned} \widehat{M}(k,i) &= (-1)^i M_{BL}(k,i) \\ &= \sum_{n=0}^{k} \binom{N-i}{n} \binom{i}{k-n} (-1)^{k-n+i} \end{aligned}$$

and $i \geq k-n$, i.e., $n \geq k-i$, then $\widehat{M}(k,i)$ can be written as

$$\widehat{M}(k,i)=\sum_{n=k-i}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k-n+i}$$

Let $n'=k-n$, then $n=k-n'$, and

$$\begin{aligned}\widehat{M}(k,i)&=\sum_{n'=i}^{0}\binom{N-i}{k-n'}\binom{i}{n'}(-1)^{n'+i}\\&=\sum_{n=0}^{i}\binom{N-i}{k-n}\binom{i}{n}(-1)^{n+i}\end{aligned}$$

Further, let $n'=i-n$, then $n=i-n'$, and

$$\begin{aligned}\widehat{M}(k,i)&=\sum_{n'=i}^{0}\binom{N-i}{k+n'-i}\binom{i}{i-n'}(-1)^{2i-n'}\\&=\sum_{n=0}^{i}\binom{N-i}{k+n-i}\binom{i}{i-n}(-1)^{n}\end{aligned}\qquad(2.79)$$

Comparing (2.78) with (2.79), we know

$$\widehat{M}(k,i)=M_{BL}(N-k,i)$$

That is, the generalized Pascal matrix $\widehat{\mathbf{M}}$ can be obtained by flipping the rows of the Pascal matrix $\mathbf{M}_{BL}$. By comparing (2.69) with (2.76), can get

$$\widetilde{\mathbf{M}}^{-1}=2^{-N}\widehat{\mathbf{M}}\qquad(2.80)$$

Moreover, $\widehat{M}(k,i)$ in (2.75) can be rewritten as

$$\begin{aligned}\widehat{M}(k,i)&=(-1)^{k}M_{BL}(k,i)\cdot(-1)^{i-k}\\&=(-1)^{k}M_{BL}(k,i)\cdot(-1)^{k+i}\\&=\widetilde{M}(k,i)\cdot(-1)^{k+i}\end{aligned}\qquad(2.81)$$

As a result, we can obtain

$$\widetilde{M}^{-1}(k,i)=2^{-N}\widetilde{M}(k,i)\cdot(-1)^{k+i}\qquad(2.82)$$

Therefore, if we want to compute the $\widetilde{\mathbf{M}}^{-1}$ from $\widetilde{\mathbf{M}}$ itself, we can first multiply the elements $\widetilde{M}(k,i)$ by $(-1)^{k+i}$ and then divide the resulting matrix by 2^N.

2.5 Recurrent Generation Formulas

In [14], it is shown that the elements $M(k,i)$ of the generalized Pascal matrix $\mathbf{M}$ in (2.13) can be computed through using the FFT and IFFT. Although this idea is interesting, the final results are numerical solutions and not in a closed-form. In this section, we derive a general recurrence formula for computing the elements $M(k,i)$, where the parameters v and w can take any real values, but $v \neq w$. Consequently, the BL, BD, FD, and parametric BD-BL transformations can be regarded as special cases. As compared with the DFT-based numerical algorithm [14], the recurrence formula can recursively compute the elements $M(k,i)$ in a closed-form, and the computational complexity can also be reduced significantly.

2.5.1 Lowpass-to-Lowpass Case

The BL, BD, FD, and parametric BD-BL transformations belong to this lowpass-to-lowpass case, where the first-order *s*-*z* transformation model (2.4) is used. Start from the element expression

$$M(k,i) = \sum_{n=0}^{k} \binom{N-i}{n} \binom{i}{k-n} v^n w^{k-n} \tag{2.83}$$

As shown in (2.13), if $k = 0$, i.e., the first row of matrix $\mathbf{M}$, Then $M(0,i) = 1$, and if $i = 0$, i.e., the first column, then

$$M(k,0) = \binom{N}{k} v^k$$

Starting from the first row and first column, can recursively compute all the elements $M(k,i)$ using the closed-form recurrence formula

$$M(k,i) = M(k,i-1) + wM(k-1,i-1) - vM(k-1,i) \tag{2.84}$$

which can be proved as follows. Since

$$\binom{N-(i-1)}{n}=\binom{N-i+1}{n}=\binom{N-i}{n-1}+\binom{N-i}{n}$$

then

$$\binom{N-i}{n}=\binom{N-(i-1)}{n}-\binom{N-i}{n-1}$$

and

$$\binom{N-i}{n}\binom{i}{k-n}=\binom{N-(i-1)}{n}\binom{i}{k-n}-\binom{N-i}{n-1}\binom{i}{k-n} \tag{2.85}$$

Substituting the identity

$$\binom{i}{k-n}=\binom{i-1}{k-n-1}+\binom{i-1}{k-n}$$

into (2.85) leads to

$$\binom{N-i}{n}\binom{i}{k-n}=\binom{N-(i-1)}{n}\binom{i-1}{k-n}+\binom{N-(i-1)}{n}\binom{i-1}{(k-1)-n}-\binom{N-i}{n-1}\binom{i}{k-n} \tag{2.86}$$

Thus, can obtain

$$\begin{aligned} M(k,i) &= \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}v^n w^{k-n} \\ &= \alpha_1+\alpha_2-\alpha_3 \end{aligned} \tag{2.87}$$

with

$$\alpha_1=\sum_{n=0}^{k}\binom{N-(i-1)}{n}\binom{i-1}{k-n}v^n w^{k-n}=M(k,i-1)$$

$$\begin{aligned} \alpha_2 &= \sum_{n=0}^{k}\binom{N-(i-1)}{n}\binom{i-1}{(k-1)-n}v^n w^{k-n} \\ &= \sum_{n=0}^{k-1}\binom{N-(i-1)}{n}\binom{i-1}{(k-1)-n}v^n w^{(k-1)-n}\cdot w \\ &= wM(k-1,i-1) \end{aligned}$$

and

$$\begin{aligned} \alpha_3 &= \sum_{n=0}^{k}\binom{N-i}{n-1}\binom{i}{k-n}v^n w^{k-n} \\ &= \sum_{n=1}^{k}\binom{N-i}{n-1}\binom{i}{k-n}v^n w^{k-n} \end{aligned}$$

If we let $n' = n-1$, then $n = n'+1$, and α_3 can be rewritten as

$$\begin{aligned}\alpha_3 &= \sum_{n'=0}^{k-1}\binom{N-i}{n'}\binom{i}{(k-1)-n'}v^{n'+1}w^{k-(n'+1)} \\ &= \sum_{n'=0}^{k-1}\binom{N-i}{n'}\binom{i}{(k-1)-n'}v^{n'}w^{(k-1)-n'}\cdot v \\ &= vM(k-1,i)\end{aligned}$$

Substituting α_1, α_2, and α_3 into (2.87) yields (2.84). This recurrence formula indicates that once the constants v and w are given, all the elements $M(k,i)$ can be recursively computed by using their left and top neighbourhood. For computing each $M(k,i)$, at most 2 multiplications and 2 additions are required. Using the recurrence formula (2.84), the BL, BD, FD, and parametric BD-BL transformations can be simplified as follows.

For BL transformation, $v=1$, $w=-1$, then

$$M_{BL}(k,i) = M_{BL}(k,i-1) - M_{BL}(k-1,i-1) - M_{BL}(k-1,i) \tag{2.88}$$

For BD transformation, $v=0$, $w=-1$, then we have

$$M_{BD}(k,i) = M_{BD}(k,i-1) - M_{BD}(k-1,i-1) \tag{2.89}$$

For FD transformation, $v=1$, $w=0$, then

$$M_{FD}(k,i) = M_{FD}(k,i-1) - M_{FD}(k-1,i) \tag{2.90}$$

For parametric BD-BL transformation, $v=r$, $w=-1$, then

$$M_P(k,i) = M_P(k,i-1) - M_P(k-1,i-1) - rM_P(k-1,i) \tag{2.91}$$

As an example, let us consider $N=3$ for the parametric BD-BL case. Using the recurrence formula (2.91), can verify

$$\mathbf{M}_P = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 3r & 2r-1 & r-2 & -3 \\ 3r^2 & r^2-2r & -2r+1 & 3 \\ r^3 & -r^2 & r & -1 \end{bmatrix} \tag{2.92}$$

Finally, some remarks should be given on the computational complexity (computation time). If the FFT-based algorithm [14] is used, while $(N + 1)$ is not equal to an integer power of two, then zero-padding is necessary before performing the FFT. Therefore, the computational complexity also depends on how far the number $(N+1)$ from the smallest integer power of two towards infinity. The most efficient case for the algorithm [14] is that $(N + 1)$ is exactly equal to the integer power of two. Assume that $(N + 1) = 2^p$, p is an integer. First, the vectors $\mathbf{a}_i$ and $\mathbf{b}_i$ in [14] must be computed before the FFT.

After generating the vectors $\mathbf{a}_i$ and $\mathbf{b}_i$, the FFT-IFFT process requires

$$3(N+1)\log_2(N+1)+(N+1)=(N+1)(3p+1)$$

complex multiplications and

$$3(N+1)\log_2(N+1)+N\approx(N+1)(3p+1)$$

complex additions. Since one complex multiplication requires 4 real multiplications, and one complex addition requires 2 real additions, thus the FFT-based algorithm [14] requires $4(N + 1)(3p + 1)$ real multiplications and $2(N + 1)(3p + 1)$ real additions. In contrast, the recurrence formula (2.84) only requires at most $2N$ real multiplications and $2N$ real additions. As a result, the ratio of real multiplications is

$$\frac{2N}{4(N+1)(3p+1)}\approx\frac{1}{2(3p+1)} \tag{2.93}$$

and the ratio of real additions

$$\frac{2N}{2(N+1)(3p+1)}\approx\frac{1}{3p+1} \tag{2.94}$$

Therefore, the recurrence formula requires lower computational complexity than the FFT-based algorithm [14]. For example, if $N = 16$, and $v = 0.5$, $w = -1$, then 15 zeros must be padded before the FFT-IFFT process, and the FFT-based algorithm [14] requires 137002 Flops

including all the multiplications and additions to complete the computation of the generalized Pascal matrix $\mathbf{M}$, while the recurrence formula only requires 3355 Flops, the ratio of the multiplications is about 0.0245, i.e., only about 2.45% of the multiplications are required by using the recurrence formula. Taking the $\mathbf{M}$ computed directly from (2.83) as the true result, then the FFT-based algorithm [14] and the recurrence formula (2.84) lead to normalized root-mean-squared (RMS) approximation errors $1.97\times10^{-13}\%$, and 0, respectively, i.e., no errors are caused. Moreover, it follows from (2.93) and (2.94) that the computational complexity can be further reduced as N becomes large. With the first, last columns and first, last rows pre-computed, i.e., once the boundary elements of $\mathbf{M}$ in (2.13) are pre-computed, the inner elements of $\mathbf{M}$ can also be recursively computed from the four corners independently. This recursive computation from four corners almost does not change the computational complexity because only the last column and last row need to be further pre-computed before starting the recurrent computation.

2.5.2 Lowpass-to-Highpass Case

The generalized Pascal matrix $\widetilde{\mathbf{M}}=\left[\widetilde{M}(k,i)\right]$ is used for the first-order lowpass-to-highpass s-z transformation (2.61). The elements $\widetilde{M}(k,i)$ can also be computed by the recurrence formula

$$\widetilde{M}(k,i)=\widetilde{M}(k,i-1)+\widetilde{M}(k-1,i-1)+\widetilde{M}(k-1,i) \tag{2.95}$$

which can be proved as follows. Since

$$\widetilde{M}(k,i)=\sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^n$$

substituting (2.86) into the above equation leads to

$$\begin{aligned}\widetilde{M}(k,i)&=\sum_{n=0}^{k}\binom{N-(i-1)}{n}\binom{i-1}{k-n}(-1)^n+\sum_{n=0}^{k}\binom{N-(i-1)}{n}\binom{i-1}{(k-1)-n}(-1)^n\\&\quad-\sum_{n=0}^{k}\binom{N-i}{n-1}\binom{i}{k-n}(-1)^n\\&=\widetilde{M}(k,i-1)+\widetilde{M}(k-1,i-1)-\sum_{n=1}^{k}\binom{N-i}{n-1}\binom{i}{k-n}(-1)^n\end{aligned} \tag{2.96}$$

If we let $n' = n-1$, then $n = n'+1$, and

$$\sum_{n=1}^{k}\binom{N-i}{n-1}\binom{i}{k-n}(-1)^n = -\sum_{n'=0}^{k-1}\binom{N-i}{n'}\binom{i}{(k-1)-n'}(-1)^{n'} = -\widetilde{M}(k-1,i) \tag{2.97}$$

Combining (2.96) with (2.97) arrives at the recurrence formula (2.95). As for the generalized Pascal matrix $\widehat{\mathbf{M}}$, it can be determined from (2.75) or (2.81) if $\mathbf{M}_{BL}$ or $\widetilde{\mathbf{M}}$ is known. As an alternative way, can also use the recurrence formula

$$\widehat{M}(k,i) = \widehat{M}(k-1,i-1) - \widehat{M}(k-1,i) - \widehat{M}(k,i-1) \tag{2.98}$$

2.6 Properties of Generalized Pascal Matrices

In this section, we prove some interesting properties that the sum of each column (except the first or last column) elements of the generalized Pascal matrices $\mathbf{M}$ (for $w=-1$), $\widetilde{\mathbf{M}}$, and $\widehat{\mathbf{M}}$ is zero. First, let us consider $\mathbf{M}$ in (2.13), where we assume that S_i denotes the sum of the (i + 1)th column elements, i.e.,

$$S_i = \sum_{k=0}^{N} M(k,i) \quad ; i = 0,1,\ldots,N$$

then can prove the following interesting property.

Property-1: If $w=-1$, $v \neq w$, then

$$S_i = \begin{cases} (1+v)^N & for\ i=0 \\ 0 & for\ i=1,2,\ldots,N \end{cases} \tag{2.99}$$

Proof: For $w=-1$, the recurrence formula (2.84) can be modified to

$$M(k,i) + vM(k-1,i) = M(k,i-1) - M(k-1,i-1)$$

For $i \geq 1$, summing up the two sides separately leads to

$$\sum_{k=0}^{N}[M(k,i) + vM(k-1,i)] = \sum_{k=0}^{N}[M(k,i-1) - M(k-1,i-1)]$$

The left-hand side can be rewritten as

$$\sum_{k=0}^{N}[M(k,i)+vM(k-1,i)]=(1+v)S_i-vM(N,i)$$

and the right-hand side becomes

$$\sum_{k=0}^{N}[M(k,i-1)-M(k-1,i-1)]=M(N,i-1)$$

where assume that the boundary elements $M(-1,i)$ and $M(-1,i-1)$ are zero. Thus,

$$(1+v)S_i-vM(N,i)=M(N,i-1)$$

i.e.,

$$S_i=\frac{vM(N,i)+M(N,i-1)}{1+v} \tag{2.100}$$

Moreover, as mentioned in (2.12), since

$$M(N,i)=v^{N-i}w^i=v^{N-i}(-1)^i$$

and

$$\begin{aligned}M(N,i-1)&=v^{N-(i-1)}(-1)^{i-1}\\&=-v\cdot v^{N-i}(-1)^i\\&=-vM(N,i)\end{aligned}$$

thus (2.100) becomes

$$S_i=\frac{vM(N,i)-vM(N,i)}{1+v}=0 \quad ;i=1,2,\ldots,N \tag{2.101}$$

If $i=0$, then

$$M(k,i)=M(k,0)=\binom{N}{k}v^k$$

thus

$$S_0 = \sum_{k=0}^{N} M(k,0) = \sum_{k=0}^{N} \binom{N}{k} v^k = (1+v)^N \tag{2.102}$$

Combining (2.101) with (2.102) together completes the proof of the *property-1* (2.99). Obviously, the BL, BD, and parametric BD-BL transformations ($w=-1$) have the property (2.99). As an example, let us check the generalized Pascal matrix $\mathbf{M}_P$ in (2.60) for the parametric BD-BL case, where $N=4$ and $v=r=0.5$. Clearly,

$$S_i = \begin{cases} (1+r)^N = 1.5^N = 5.0625 & for\ i=0 \\ 0 & for\ i=1,2,3,4 \end{cases}$$

Further, the Pascal matrix (2.40) for the BL case, the generalized Pascal matrix (2.44) for the BD case, and (2.92) for the parametric BD-BL case also have the property (2.99).

Next, consider the property of $\widetilde{\mathbf{M}}$ in (2.69), whose elements are

$$\widetilde{M}(k,i) = (-1)^k M_{BL}(k,i) \tag{2.103}$$

and $M_{BL}(k,i)$ denote the elements for the BL case (2.26). Manipulating the recurrence formula (2.95) as

$$\widetilde{M}(k,i) - \widetilde{M}(k-1,i) = \widetilde{M}(k,i-1) + \widetilde{M}(k-1,i-1)$$

and summing up the two sides, can obtain

$$\sum_{k=0}^{N} [\widetilde{M}(k,i) - \widetilde{M}(k-1,i)] = \sum_{k=0}^{N} [\widetilde{M}(k,i-1) + \widetilde{M}(k-1,i-1)]$$

Since the left-hand side becomes

$$\sum_{k=0}^{N} [\widetilde{M}(k,i) - \widetilde{M}(k-1,i)] = \widetilde{M}(N,i)$$

and the right-hand side reduces to

$$\sum_{k=0}^{N}\left[\widetilde{M}(k,i-1)+\widetilde{M}(k-1,i-1)\right]=2\tilde{S}_{i-1}-\widetilde{M}(N,i-1)$$

we have

$$2\tilde{S}_{i-1}-\widetilde{M}(N,i-1)=\widetilde{M}(N,i)$$

thus

$$\tilde{S}_{i-1}=\frac{\widetilde{M}(N,i-1)+\widetilde{M}(N,i)}{2}$$

Since it follows from (2.103) that

$$\widetilde{M}(N,i)=(-1)^{N}\cdot M_{BL}(M,i)=(-1)^{N+i}$$
$$\widetilde{M}(N,i-1)=(-1)^{N}\cdot(-1)^{i-1}=-\widetilde{M}(N,i)$$

can obtain

$$\tilde{S}_{i-1}=0 \qquad ;i=1,2,\ldots,N \tag{2.104}$$

i.e., $\tilde{S}_0,\tilde{S}_1,\ldots,\tilde{S}_{N-1}$ are zero. For the last column of $\widetilde{\mathbf{M}}$, since $M_{BL}(k,N)=\binom{N}{k}(-1)^k$, and

$$\widetilde{M}(k,N)=(-1)^{k}M_{BL}(k,N)=\binom{N}{k}$$

then

$$\tilde{S}_N=\sum_{k=0}^{N}\widetilde{M}(k,N)=\sum_{k=0}^{N}\binom{N}{k}=2^N$$

As a result, can arrive at the *Property-2*

$$\tilde{S}_i=\begin{cases}0 & for\ i=0,1,\ldots,(N-1)\\ 2^N & for\ i=N\end{cases} \tag{2.105}$$

As an example, the $\widetilde{\mathbf{M}}$ in (2.71) satisfies (2.105).

Finally, we consider the generalized Pascal matrix $\widehat{\mathbf{M}}$ in (2.76), whose elements are given as

$$\widehat{M}(k,i) = (-1)^i M_{BL}(k,i)$$

Thus,

$$\hat{S}_i = \sum_{k=0}^{N} \widehat{M}(k,i) = (-1)^i \sum_{k=0}^{N} M_{BL}(k,i) = (-1)^i S_i$$

where S_i is given in (2.99) with $v=1$. Hence, I have the *Property-3*

$$\hat{S}_i = \begin{cases} 2^N & for\ i=0 \\ 0 & for\ i=1,2,\ldots,N \end{cases} \tag{2.106}$$

As an example, can see that the $\widehat{\mathbf{M}}$ in (2.77) satisfies (2.106).

2.7 Proof of Inverse Pascal Matrix

For BL transformation, substituting $v=1$ and $w=-1$ into (2.11) yields

$$M_{BL}(k,i) = \sum_{n=0}^{k} \binom{N-i}{n} \binom{i}{k-n} (-1)^{k-n} \tag{2.107}$$

Substituting $n' = k-n$ into (2.107), can get

$$\begin{aligned} M_{BL}(k,i) &= \sum_{n'=k}^{0} \binom{N-i}{k-n'} \binom{i}{n'} (-1)^{n'} \\ &= \sum_{n=0}^{k} (-1)^n \binom{i}{n} \binom{N-i}{k-n} \end{aligned}$$

Using the binomial expansions

$$(1-x)^i = \sum_{n=0}^{i} \binom{i}{n} (-x)^n$$

$$(1+x)^{N-i} = \sum_{n=0}^{N-i} \binom{N-i}{n} x^n$$

can get the expansion

$$(1-x)^i(1+x)^{N-i} = \sum_{k=0}^{N} M_{BL}(k,i)x^k \tag{2.108}$$

Now, consider the Nth degree polynomial

$$f_N(x) = [(1+x)-(1-x)]^i[(1+x)+(1-x)]^{N-i}$$

which can be expanded as

$$\begin{aligned} f_N(x) &= [(1+x)-(1-x)]^i[(1+x)+(1-x)]^{N-i} \\ &= \left[\sum_{j=0}^{i}(-1)^j\binom{i}{j}(1+x)^{i-j}(1-x)^j\right]\times\left[\sum_{l=0}^{N-i}\binom{N-i}{l}(1+x)^{(N-i)-l}(1-x)^l\right] \\ &= \sum_{j=0}^{i}\sum_{l=0}^{N-i}(-1)^j\binom{i}{j}\binom{N-i}{l}(1-x)^{j+l}(1+x)^{N-(j+l)} \\ &= \sum_{j=0}^{i}\sum_{l=0}^{N-i}(-1)^j\binom{i}{j}\binom{N-i}{l}\cdot\sum_{k=0}^{N}M_{BL}(k,j+l)x^k \\ &= \sum_{k=0}^{N}\left[\sum_{j=0}^{i}\sum_{l=0}^{N-i}(-1)^j\binom{i}{j}\binom{N-i}{l}M_{BL}(k,j+l)\right]x^k \end{aligned} \tag{2.109}$$

Performing substitution $n = j+l$ yields

$$\sum_{j=0}^{i}\sum_{l=0}^{N-i}(-1)^j\binom{i}{j}\binom{N-i}{l}M_{BL}(k,j+l) = \sum_{n=0}^{N}\sum_{j=0}^{i}(-1)^j\binom{i}{j}\binom{N-i}{n-j}M_{BL}(k,n) \tag{2.110}$$

Furthermore, since

$$\sum_{j=0}^{i}(-1)^j\binom{i}{j}\binom{N-i}{n-j} = \sum_{j=0}^{n}(-1)^j\binom{i}{j}\binom{N-i}{n-j} = M_{BL}(n,i)$$

can simplify (2.110) as

$$\sum_{j=0}^{i}\sum_{l=0}^{N-i}(-1)^j\binom{i}{j}\binom{N-i}{l}M_{BL}(k,j+l) = \sum_{n=0}^{N}M_{BL}(k,n)M_{BL}(n,i) \tag{2.111}$$

Substituting (2.111) into (2.109) yields

$$f_N(x) = \sum_{k=0}^{N}\left[\sum_{n=0}^{N} M_{BL}(k,n)M_{BL}(n,i)\right]x^k$$

Moreover, since

$$\begin{aligned} f_N(x) &= [(1+x)-(1-x)]^i[(1+x)+(1-x)]^{N-i} \\ &= (2x)^i \cdot 2^{N-i} \\ &= 2^N x^i \end{aligned}$$

we have

$$\sum_{k=0}^{N}\left[\sum_{n=0}^{N} M_{BL}(k,n)M_{BL}(n,i)\right]x^k = 2^N x^i \tag{2.112}$$

which leads to

$$\sum_{n=0}^{N} M_{BL}(k,i)M_{BL}(n,i) = \begin{cases} 2^N & \text{if } k=i \\ 0 & \text{if } k\neq i \end{cases}$$

Consequently, we know

$$\mathbf{M}_{BL}\mathbf{M}_{BL} = \begin{bmatrix} 2^N & 0 & \cdots & 0 \\ 0 & 2^N & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 2^N \end{bmatrix}$$

i.e.,

$$\mathbf{M}_{BL}^{-1} = 2^{-N}\mathbf{M}_{BL}$$

2.8 Proof of Recurrence Formula

Since $\widehat{M}(k,i)$ can be rewritten as

$$\begin{aligned} \widehat{M}(k,i) &= (-1)^i M_{BL}(k,i) \\ &= \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k-n}\cdot(-1)^i \\ &= \sum_{n=0}^{k}\binom{N-i}{n}\binom{i}{k-n}(-1)^{k+i}\cdot(-1)^n \end{aligned} \tag{2.113}$$

substituting (2.86) into (2.113) leads to

$$\widehat{M}(k,i) = \beta_1 + \beta_2 - \beta_3 \tag{2.114}$$

with

$$\begin{aligned}
\beta_1 &= \sum_{n=0}^{k} \binom{N-(i-1)}{n} \binom{i-1}{k-n} (-1)^{k+i} \cdot (-1)^n \\
&= -\sum_{n=0}^{k} \binom{N-(i-1)}{n} \binom{i-1}{k-n} (-1)^{k+(i-1)} \cdot (-1)^n \\
&= -\widehat{M}(k,i-1)
\end{aligned}$$

$$\begin{aligned}
\beta_2 &= \sum_{n=0}^{k} \binom{N-(i-1)}{n} \binom{i-1}{(k-1)-n} (-1)^{k+i} \cdot (-1)^n \\
&= -\sum_{n=0}^{k-1} \binom{N-(i-1)}{n} \binom{i-1}{(k-1)-n} (-1)^{(k-1)+(i-1)} \cdot (-1)^n \\
&= -\widehat{M}(k-1,i-1)
\end{aligned}$$

and

$$\begin{aligned}
\beta_3 &= \sum_{n=0}^{k} \binom{N-i}{n-1} \binom{i}{k-n} (-1)^{k+i} \cdot (-1)^n \\
&= \sum_{n=1}^{k} \binom{N-i}{n-1} \binom{i}{k-n} (-1)^{k+i} \cdot (-1)^n
\end{aligned}$$

If we let $n' = n-1$,then $n = n'+1$, and β_3 can be rewritten as

$$\begin{aligned}
\beta_3 &= \sum_{n'=0}^{k-1} \binom{N-i}{n'} \binom{i}{(k-1)-n'} (-1)^{k+i} \cdot (-1)^{n'+1} \\
&= \sum_{n'=0}^{k-1} \binom{N-i}{n'} \binom{i}{(k-1)-n'} (-1)^{(k-1)+i} \cdot (-1)^{n'} \\
&= \widehat{M}(k-1,i)
\end{aligned}$$

Substituting $\beta_1, \beta_2,$ and β_3 into (2.114) yields (2.98).

2.9 Conclusion

In this chapter, we have proposed a one-to-one mapping method for transforming the *s*-domain IIR transfer functions to the *z*-domain ones or vice versa using the first-order *s*-*z* transformations. The original prototype IIR filter can be completely restored from the transformed one. Based on the one-to-one mapping, we have proved various inverses of the generalized Pascal matrices, such rigorous proofs cannot be found in the literature. Also, recurrence formulas have been derived for recursively computing the generalized Pascal matrices from their boundary elements, which simplifies the generations of the generalized Pascal matrices. Finally, we have proved some interesting properties of the generalized Pascal matrices. Various examples have been included to verify and illustrate the generalized Pascal matrices, inverses, recursive computations and interesting properties.

Chapter 3

Further Results on Generalized Pascal Matrices and Inverses Using a Unified s-z Transformation Model

3.1 Introduction

This chapter derives an explicit expression for a new generalized Pascal matrix called *unified Pascal matrix* from a unified first-order s-z transformation model and rigorously proves the inverses for various first-order s-z transformations. After deriving a recurrence formula for recursively generating the inner elements of the unified Pascal matrix from its boundary elements, we also show that the recurrence formula leads to computationally unstable solutions for high-order systems due to the so-called catastrophic cancellation in numerical computation, but the unstable problem can be solved through partitioning the whole unified Pascal matrix into several small matrices (sub-matrices) and then using the recurrence formula to compute the sub-matrices individually from their boundary elements. This operation almost retains the same computational complexity while guarantees numerically stable solutions. Moreover, an interesting property of the unified Pascal matrix is proved.

3.2 Unified Pascal Matrix

Continuous-time (CT) linear systems (filters) can be transformed to discrete-time (DT) ones through s-to-z transformations for easy system analysis, and infinite-impulse-response (IIR) digital filters can be indirectly designed through transforming CT linear filters into DT filters [1-12]. Conversely, DT linear systems (filters) can also be converted to CT ones through z-to-s transformations. In such transformations, the bilinear (BL) transformation leads to a transformation matrix called *Pascal matrix* [8-12], which is a special case of the *generalized Pascal matrix* [13-14]. In [16], the inverses, recurrence formulas, and some interesting properties of the generalized Pascal matrices using two types of first-order s-z transformation (lowpass-to-lowpass and lowpass-to-highpass) models have been proved on the basis of one-to-one coefficient mapping between the CT and DT domain rational transfer functions

$$H_{CT}(s)=\frac{N_{CT}(s)}{D_{CT}(s)}=\frac{\sum_{k=0}^{N}A_k s^k}{\sum_{k=0}^{N}B_k s^k} \tag{3.1}$$

$$H(z)=\frac{N(z)}{D(z)}=\frac{\sum_{k=0}^{N}a_k z^{-k}}{\sum_{k=0}^{N}b_k z^{-k}}. \tag{3.2}$$

In this section, we use the more general s-z transformation model (unified s-z transformation model) in [17] to derive a more general Pascal matrix, we call it the *unified Pascal matrix* for short. Besides the BD, FD, BL1, and BL2 transformations discussed in [17], we also include the parametric BD-BL1 transformation. Further, an explicit expression for the elements of the unified Pascal matrix is also derived.

3.2.1 Transformation Model and Transformation Matrix

Assume that the rational s-domain (CT) transfer function (3.1) and the z-domain (DT) rational transfer function (3.2) are one-to-one mutually transformed through the first-order s-z transformation model

$$s=c\frac{1+\alpha z^{-1}}{\mu+\beta z^{-1}} \tag{3.3}$$

where c, μ, α, β are real constants [17], $\beta \neq \alpha\mu$. The BD, FD, BL1, BL2, and parametric BD-BL1transformations can be parameterized as Table 3.1, where the constants $c1$, $c2$, $c3$, $c4$, $c5$ are related to the sampling period T, and $\beta \neq -\mu$ for lowpass-to-lowpass transformations $s = 0 \leftrightarrow z = 1$ (BD, FD, BL1, parametric BD-BL1), and $\beta \neq \mu$ for lowpass-to-highpass transformation $s = 0 \leftrightarrow z = -1$ (BL2).

Table 3.1 Parameters for Various s-z Transformations

Transforms	Formula	c	(μ,α,β)
BD	$s=c_1(1-z^{-1})$	c_1	(1,-1,0)
FD	$s=c_2\dfrac{1-z^{-1}}{z^{-1}}$	c_2	(0,-1,1)
BL1	$s=c_3\dfrac{1-z^{-1}}{1+z^{-1}}$	c_3	(1,-1,1)
BL2	$s=c_4\dfrac{1+z^{-1}}{1-z^{-1}}$	c_4	(1,1,-1)
Parametric BD-BL1	$s=c_5\dfrac{1-z^{-1}}{1+rz^{-1}}$	c_5	(1,-1,r)

For simplicity, we set $x=z^{-1}$ and rewrite the transformation model (3.3) as

$$s=c\frac{1+\alpha x}{\mu+\beta x} \tag{3.4}$$

Thus, $H(z)$ in (3.2) becomes

$$H(x)=\frac{\sum_{k=0}^{N}a_k x^k}{\sum_{k=0}^{N}b_k x^k} \tag{3.5}$$

Substituting (3.4) into (3.1) yields the s-to-z transformation

$$\begin{aligned}\hat{H}(x)&=\frac{\sum_{k=0}^{N}A_k\left(c\dfrac{1+\alpha x}{\mu+\beta x}\right)^k}{\sum_{k=0}^{N}B_k\left(c\dfrac{1+\alpha x}{\mu+\beta x}\right)^k}\\&=\frac{\sum_{k=0}^{N}A_k c^k(1+\alpha x)^k(\mu+\beta x)^{N-k}}{\sum_{k=0}^{N}B_k c^k(1+\alpha x)^k(\mu+\beta x)^{N-k}}\end{aligned} \tag{3.6}$$

Based on the assumption that $H(x)$ is transformed from $H_{CT}(s)$ using the unified s-z transformation model (3.4), we can set $\hat{H}(x)=H(x)$, i.e.,

$$\frac{\sum_{k=0}^{N} A_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}} = \frac{\sum_{k=0}^{N} a_k x^k}{\sum_{k=0}^{N} b_k x^k} \tag{3.7}$$

However, the mappings between the numerator coefficients $A_k \leftrightarrow a_k$ and denominator coefficients $B_k \leftrightarrow b_k$ are not one-to-one. In this paper, we want to find the one-to-one mappings $A_k \leftrightarrow a_k$ and $B_k \leftrightarrow b_k$.

Clearly, it is incorrect to set

$$\begin{cases} \sum_{k=0}^{N} A_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k} = \sum_{k=0}^{N} a_k x^k \\ \sum_{k=0}^{N} B_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k} = \sum_{k=0}^{N} b_k x^k \end{cases}$$

because (3.7) can be modified to

$$\frac{\sum_{k=0}^{N} A_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}} = \frac{\sum_{k=0}^{N} (\lambda a_k) x^k}{\sum_{k=0}^{N} (\lambda b_k) x^k} \tag{3.8}$$

for an arbitrary non-zero scaling factor λ, $\lambda \neq 0$. Denoting the numerator and denominator coefficients of $\hat{H}(x)$ by $\hat{a}_k = \lambda a_k$ and $\hat{b}_k = \lambda b_k$, respectively, then we have

$$\hat{H}(x) = \frac{\sum_{k=0}^{N} A_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}} = \frac{\sum_{k=0}^{N} \hat{a}_k x^k}{\sum_{k=0}^{N} \hat{b}_k x^k} \tag{3.9}$$

Therefore, the constant λ must be determined in order to get the one-to-one mappings $A_k \leftrightarrow a_k$ and $B_k \leftrightarrow b_k$. Since the numerator and denominator of $\hat{H}(x)$ have the same form, we only focus on the numerator, i.e.,

$$\sum_{k=0}^{N} A_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k} = \sum_{k=0}^{N} \hat{a}_k x^k \tag{3.10}$$

Denoting

$$\mathbf{A}^T = \begin{bmatrix} A_0 & A_1 & A_2 & \cdots & A_N \end{bmatrix}$$

$$\mathbf{D}_c = \begin{bmatrix} c^0 & & & & 0 \\ & c^1 & & & \\ & & c^2 & & \\ & & & \ddots & \\ 0 & & & & c^N \end{bmatrix}$$

$$\tilde{\mathbf{A}}^T = \begin{bmatrix} A_0 c^0 & A_1 c^1 & A_2 c^2 & \cdots & A_N c^N \end{bmatrix} = \mathbf{A}^T \mathbf{D}_c \tag{3.11}$$

$$\mathbf{a}^T = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_N \end{bmatrix}$$

$$\mathbf{x}^T = \begin{bmatrix} 1 & x & x^2 & \cdots & x^N \end{bmatrix}$$

and

$$\mathbf{P}^T(\mu,\alpha,\beta,x) = \begin{bmatrix} P_0(x) & P_1(x) & P_2(x) & \cdots & P_N(x) \end{bmatrix}$$

with

$$P_k(x) = (1+\alpha x)^k (\mu+\beta x)^{N-k}, \quad k = 0,1,\cdots,N \tag{3.12}$$

the numerator relation (3.10) can be rewritten as

$$\mathbf{P}^T(\mu,\alpha,\beta,x)\tilde{\mathbf{A}} = \mathbf{x}^T\hat{\mathbf{a}} \tag{3.13}$$

Moreover, since the Nth degree polynomial $P_k(x)$ in (3.12) can be expressed as

$$P_k(x) = \sum_{i=0}^{N} p_{i,k} x^i = \mathbf{x}^T \mathbf{p}_k \tag{3.14}$$

with

$$\mathbf{p}_k = \begin{bmatrix} p_{0,k} & p_{1,k} & p_{2,k} & \cdots & p_{N,k} \end{bmatrix}^T$$

we have

$$\mathbf{P}^T(\mu,\alpha,\beta,x) = \mathbf{x}^T \begin{bmatrix} \mathbf{p}_0 & \mathbf{p}_1 & \mathbf{p}_2 & \cdots & \mathbf{p}_N \end{bmatrix} \tag{3.15}$$

Substituting (3.15) into (3.13) yields

$$\hat{\mathbf{a}} = \mathbf{P}\tilde{\mathbf{A}} \tag{3.16}$$

with

$$\mathbf{P} = [\mathbf{p}_0 \quad \mathbf{p}_1 \quad \mathbf{p}_2 \quad \cdots \quad \mathbf{p}_N]$$

Similarly, the relation between the denominator coefficients in (3.9) can be written as

$$\hat{\mathbf{b}} = \mathbf{P}\tilde{\mathbf{B}} \tag{3.17}$$

Combining (3.16) with (3.17) yields

$$\begin{aligned} [\hat{\mathbf{a}} \quad \hat{\mathbf{b}}] &= \mathbf{P}[\tilde{\mathbf{A}} \quad \tilde{\mathbf{B}}] \\ &= \mathbf{P}\mathbf{D}_c[\mathbf{A} \quad \mathbf{B}] \end{aligned} \tag{3.18}$$

i.e., both the numerator and denominator coefficients of the s-domain transfer function $H_{CT}(s)$ are mapped to those of the z-domain transfer function $\hat{H}(z)$.

3.2.2 Explicit Element Expression

It is clear from (3.15) that each column of the transformation matrix **P** corresponds to the coefficients $P_k(x)$ in (3.12). Here, we want to find an explicit expression for the elements of **P**. Since

$$(1+\alpha x)^k = \sum_{n=0}^{k} \binom{k}{n} \alpha^n x^n = \sum_{n=0}^{k} f_n x^n \tag{3.19}$$

with $f_n = \binom{k}{n}\alpha^n$, and

$$\begin{aligned} (\mu+\beta x)^{N-k} &= \sum_{n=0}^{N-k} \binom{N-k}{n} \mu^{(N-k)-n} (\beta x)^n \\ &= \sum_{n=0}^{N-k} \binom{N-k}{n} \mu^{(N-k)-n} \beta^n x^n \\ &= \sum_{n=0}^{N-k} g_n x^n \end{aligned}$$

with $g_n = \binom{N-k}{n}\mu^{(N-k)-n}\beta^n$, we can compute the coefficients $p_{i,k}$ (elements of **P**) in (3.12) as

$$
\begin{aligned}
p_{i,k} &= f_n * g_n \\
&= \sum_{n=0}^{i} f_n g_{i-n} \\
&= \sum_{n=0}^{i} \binom{k}{n} \alpha^n \binom{N-k}{i-n} \mu^{N-k-i+n} \beta^{i-n} \\
&= \sum_{n=0}^{i} \binom{N-k}{i-n} \binom{k}{n} \mu^{N-k-i+n} \alpha^n \beta^{i-n}
\end{aligned}
\tag{3.20}
$$

where "*" denotes the convolution operation. Substituting $n' = i - n$ into (3.20) arrives at

$$
\begin{aligned}
p_{i,k} &= \sum_{n'=i}^{0} \binom{N-k}{n'} \binom{k}{i-n'} \mu^{N-k-n'} \alpha^{i-n'} \beta^{n'} \\
&= \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{N-k-n} \alpha^{i-n} \beta^{n}
\end{aligned}
\tag{3.21}
$$

If $k = N$, then the elements of the last column of **P** are

$$
p_{i,N} = \sum_{n=0}^{i} \binom{0}{n} \binom{N}{i-n} \mu^{-n} \alpha^{i-n} \beta^n = \binom{N}{i} \alpha^i \tag{3.22}
$$

If $i = 0$, then the elements of the first row of **P** are

$$
\begin{aligned}
p_{0,k} &= \sum_{n=0}^{0} \binom{N-k}{n} \binom{k}{0-n} \mu^{N-k-n} \alpha^{-n} \beta^n \\
&= \mu^{N-k}
\end{aligned}
$$

If $k = 0$, then the elements of the first column of **P** are

$$
\begin{aligned}
p_{i,0} &= \sum_{n=0}^{i} \binom{N-0}{n} \binom{0}{i-n} \mu^{N-n} \alpha^{i-n} \beta^n \\
&= \binom{N}{i} \mu^{N-i} \beta^i
\end{aligned}
\tag{3.23}
$$

If $i = N$, then the elements of the last row of **P** are

$$p_{N,k} = \sum_{n=0}^{N} \binom{N-k}{n} \binom{k}{N-n} \mu^{N-k-n} \alpha^{N-n} \beta^{n}$$

since

$$\binom{N-k}{n} \neq 0, \qquad for\ N-k \geq n$$

and

$$\binom{k}{N-n} \neq 0, \qquad for\ k \geq N-n$$

then

$$\binom{N-k}{n} \binom{k}{N-n} \neq 0, \qquad for\ n = N-k$$

and

$$p_{N,k} = \alpha^{k} \beta^{N-k} \tag{3.24}$$

Consequently, the transformation matrix **P** takes the form

$$\mathbf{P} = \left[p_{i,k} \right] = \begin{bmatrix} \mu^{N} & \mu^{N-1} & \cdots & \mu & 1 \\ \binom{N}{1} \mu^{N-1} \beta & \cdots & \cdots & \cdots & \binom{N}{1} \alpha^{1} \\ \binom{N}{2} \mu^{N-2} \beta^{2} & \cdots & \cdots & \cdots & \binom{N}{2} \alpha^{2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \beta^{N} & \alpha \beta^{N-1} & \cdots & \alpha^{N-1} \beta & \alpha^{N} \end{bmatrix} \tag{3.25}$$

For simplicity, we refer to **P** as the *unified Pascal matrix.*

3.2.3 Recurrence Formula

Using the explicit element expression (3.21), we can derive the recurrence formula

$$p_{i,k} = \mu p_{i,k+1} - \alpha p_{i-1,k} + \beta p_{i-1,k+1} \tag{3.26}$$

as follows. From

$$\binom{k+1}{i-n} = \binom{k}{i-n} + \binom{k}{i-n-1}$$

we have

$$\binom{k}{i-n} = \binom{k+1}{i-n} - \binom{k}{i-n-1} \tag{3.27}$$

Further, since

$$\binom{N-k}{n} = \binom{N-k-1}{n} + \binom{N-k-1}{n-1} \tag{3.28}$$

we obtain

$$\begin{aligned}\binom{N-k}{n}\binom{k}{i-n} &= \binom{N-k}{n}\binom{k+1}{i-n} - \binom{N-k}{n}\binom{k}{i-n-1} \\ &= \binom{N-k-1}{n}\binom{k+1}{i-n} + \binom{N-k-1}{n-1}\binom{k+1}{i-n} \\ &\quad - \binom{N-k}{n}\binom{k}{i-n-1}\end{aligned} \tag{3.29}$$

Substituting (3.29) into (3.21) yields

$$\begin{aligned}p_{i,k} &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\mu^{N-k-n}\alpha^{i-n}\beta^{n} \\ &= \sum_{n=0}^{i}\binom{N-k-1}{n}\binom{k+1}{i-n}\mu^{N-k-n}\alpha^{i-n}\beta^{n} \\ &\quad + \sum_{n=0}^{i}\binom{N-k-1}{n-1}\binom{k+1}{i-n}\mu^{N-k-n}\alpha^{i-n}\beta^{n} \\ &\quad - \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n-1}\mu^{N-k-n}\alpha^{i-n}\beta^{n} \\ &= \lambda_1 + \lambda_2 + \lambda_3\end{aligned} \tag{3.30}$$

where

$$\begin{aligned}\lambda_1 &= \sum_{n=0}^{i}\binom{N-k-1}{n}\binom{k+1}{i-n}\mu^{N-k-n}\alpha^{i-n}\beta^n \\ &= \sum_{n=0}^{i}\binom{N-k-1}{n}\binom{k+1}{i-n}\mu^{N-(k+1)-n}\alpha^{i-n}\beta^n\cdot\mu \\ &= \mu p_{i,k+1}\end{aligned}$$

$$\begin{aligned}\lambda_3 &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n-1}\mu^{N-k-n}\alpha^{i-n}\beta^n \\ &= \sum_{n=0}^{i-1}\binom{N-k}{n}\binom{k}{i-n-1}\mu^{N-k-n}\alpha^{(i-1)-n}\beta^n\cdot\alpha \\ &= \alpha p_{i-1,k}\end{aligned}$$

and

$$\begin{aligned}\lambda_2 &= \sum_{n=0}^{i}\binom{N-k-1}{n-1}\binom{k+1}{i-n}\mu^{N-k-n}\alpha^{i-n}\beta^n \\ &= \sum_{n=1}^{i}\binom{N-k-1}{n-1}\binom{k+1}{i-n}\mu^{N-k-n}\alpha^{i-n}\beta^n \\ &= \alpha p_{i-1,k}\end{aligned}$$

Let $n' = n-1$, then $n = n'+1$, and

$$\begin{aligned}\lambda_2 &= \sum_{n'=0}^{i-1}\binom{N-k-1}{n'}\binom{k+1}{i-1-n'}\mu^{N-k-n'-1}\alpha^{i-n'-1}\beta^{n'-1} \\ &= \sum_{n=0}^{i-1}\binom{N-k-1}{n}\binom{k+1}{i-n-1}\mu^{N-(k+1)-n}\alpha^{(i-1)-n}\beta^n\cdot\beta \\ &= \beta p_{i-1,k+1}\end{aligned}$$

Substituting λ_1, λ_2, and λ_3 into (3.30) gets the recurrence formula (3.26). The recurrence formula (3.26) can also be proved as follows. Since

$$\begin{aligned}P_k(x) &= (1+\alpha x)^k(\mu+\beta x)^{N-k} = \sum_{i=0}^{N} p_{i,k}x^i \\ P_{k+1}(x) &= (1+\alpha x)^{k+1}(\mu+\beta x)^{N-k-1} = \sum_{i=0}^{N} p_{i,k+1}x^i\end{aligned} \tag{3.31}$$

we have

$$\begin{aligned}\frac{p_k(x)}{p_{k+1}(x)} &= \frac{(1+\alpha x)^k(\mu+\beta x)^{N-k}}{(1+\alpha x)^{k+1}(\mu+\beta x)^{N-k-1}} \\ &= \frac{\mu+\beta x}{1+\alpha x} \\ &= \frac{\sum_{i=0}^{N} p_{i,k}x^i}{\sum_{i=0}^{N} p_{i,k+1}x^i}\end{aligned} \tag{3.32}$$

Hence,

$$(1+\alpha x)\sum_{i=0}^{N} p_{i,k}x^i = (\mu+\beta x)\sum_{i=0}^{N} p_{i,k+1}x^i$$

$$\sum_{i=0}^{N}\left[p_{i,k}+\alpha p_{i-1,k}\right]x^i = \sum_{i=0}^{N}\left[\mu p_{i,k+1}+\beta p_{i-1,k+1}\right]x^i$$

i.e.,

$$p_{i,k}+\alpha p_{i-1,k} = \mu p_{i,k+1}+\beta p_{i-1,k+1} \tag{3.33}$$

but $p_{-1,k}=0$, $p_{-1,k+1}=0$. As a result, we arrive at (3.26).

For the BD, FD, BL1, BL2, and parametric BD-BL1 transformations, the explicit element expressions and recurrence formulas can be further simplified as follows:

♦ For the BD case, $(\mu,\alpha,\beta)=(1,-1,0)$, the element expression becomes

$$\begin{aligned}p_{i,k} &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}(-1)^{i-n}\cdot 0^n \\ &= \binom{k}{i}(-1)^i\end{aligned} \tag{3.34}$$

and the recurrence formula becomes

$$p_{i,k} = p_{i,k+1}+p_{i-1,k} \tag{3.35}$$

♦ For the FD case, $(\mu,\alpha,\beta)=(0,-1,1)$, the element expression is

$$\begin{aligned} p_{i,k} &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\cdot 0^{N-k-n}\cdot(-1)^{i-n}\cdot 1^{n} \\ &= \binom{k}{i-(N-k)}\cdot(-1)^{i-(N-k)} \\ &= \binom{k}{N-i}\cdot(-1)^{N-(i+k)} \end{aligned} \tag{3.36}$$

and the recurrence formula becomes

$$p_{i,k} = p_{i-1,k} + p_{i-1,k+1} \tag{3.37}$$

♦ For the BL1 case, $(\mu,\alpha,\beta)=(1,-1,1)$, the element expression becomes

$$\begin{aligned} p_{i,k} &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\cdot 1^{N-k-n}\cdot(-1)^{i-n}\cdot 1^{n} \\ &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\cdot(-1)^{i-n} \end{aligned} \tag{3.38}$$

and the recurrence formula is

$$p_{i,k} = p_{i,k+1} + p_{i-1,k} + p_{i-1,k+1} \tag{3.39}$$

♦ For the BL2 case, $(\mu,\alpha,\beta)=(1,1,-1)$, the element expression becomes

$$\begin{aligned} p_{i,k} &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\cdot 1^{N-k-n}\cdot 1^{i-n}\cdot(-1)^{n} \\ &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\cdot(-1)^{n} \end{aligned} \tag{3.40}$$

and the recurrence formula becomes

$$p_{i,k} = p_{i,k+1} - p_{i-1,k} - p_{i-1,k+1} \tag{3.41}$$

♦ For the parametric BD-BL1 case, $(\mu,\alpha,\beta)=(1,-1,r)$, then the element expression becomes

$$\begin{aligned} p_{i,k} &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}\cdot 1^{N-k-n}\cdot(-1)^{i-n}\cdot r^{n} \\ &= \sum_{n=0}^{i}\binom{N-k}{n}\binom{k}{i-n}(-1)^{i-n}\cdot r^{n} \end{aligned} \tag{3.42}$$

and the recurrence formula is

$$p_{i,k} = p_{i,k+1} + p_{i-1,k} + rp_{i-1,k+1} \tag{3.43}$$

3.3 Inverses of Unified Pascal Matrices

In this section, we prove the inverses of unified Pascal matrices for the BD, FD, BL1, parametric BD-BL1, and BL2 transformations, which is based on the one-to-one coefficient mappings between the rational CT and DT transfer functions. To derive the inverses, we need to perform both *s*-to-*z* and *z*-to-*s* transformations, and then find the relations between the numerator/denominator coefficients of CT and DT transfer functions. For simplicity, we call the *s*-to-*z* transformation the *forward s-z transformation*, and *z*-to-*s* transformation the *backward s-z transformation*.

It follows from the unified transformation model (3.4) that x can be expressed as

$$x = c\frac{1+\alpha' s}{\mu'+\beta s} \tag{3.44}$$

with $\alpha' = -\dfrac{\mu}{c}$, and $\mu' = -\alpha c$. Therefore, the *s*-to-*z* transformation (forward *s*-*z* transformation) is defined as

$$H_{CT}(s) \xrightarrow{s=c\frac{1+\alpha x}{\mu+\beta x}} \hat{H}(x)$$

and the *z*-to-*s* transformation (backward *s*-*z* transformation) is defined as

$$H(x) \xrightarrow{x=c\frac{1+\alpha' s}{\mu'+\beta s}} \hat{H}_{CT}(s)$$

We will show that although

$$\hat{H}(x) = H(x)$$

and

$$\hat{H}_{CT}(s) = H_{CT}(s)$$

their numerator and denominator coefficients are not necessarily equal. In other words, both numerator and denominator coefficients may be scaled by a non-zero scaling factor. Without considering such a scaling factor, the inverses of the unified Pascal matrices cannot be rigorously derived.

3.3.1 s-to-z Transformation

The resulting z-domain transfer function (3.9) after the s-to-z transformation can be rewritten as

$$\begin{aligned}\hat{H}(x) &= \frac{\sum_{k=0}^{N} A_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1+\alpha x)^k (\mu+\beta x)^{N-k}} \\ &= \frac{\sum_{k=0}^{N} A_k c^k P_k(x)}{\sum_{k=0}^{N} B_k c^k P_k(x)} \\ &= \frac{\sum_{k=0}^{N} \hat{a}_i x^i}{\sum_{k=0}^{N} \hat{b}_i x^i}\end{aligned} \tag{3.45}$$

with

$$P_k(x) = (1+\alpha x)^k (\mu+\beta x)^{N-k} = \sum_{i=0}^{N} p_{i,k} x^i$$

and $p_{i,k}$ is defined in (3.21). The numerator of $\hat{H}(x)$ can be further rewritten as

$$\sum_{i=0}^{N} \hat{a}_i x^i = \sum_{k=0}^{N} A_k c^k P_k(x)$$
$$= \sum_{k=0}^{N} A_k c^k \left[\sum_{i=0}^{N} p_{i,k} x^i \right]$$
$$= \sum_{i=0}^{N} \left[\sum_{k=0}^{N} A_k c^k p_{i,k} \right] x^i$$

thus

$$\hat{a}_i = \sum_{k=0}^{N} p_{i,k} A_k c^k = \sum_{k=0}^{N} p_{i,k} \tilde{A}_k$$

with $\tilde{A}_k = A_k c^k$. The relation between the numerator coefficients $\hat{a}_i$ and A_i can be expressed in matrix form

$$\hat{\mathbf{a}} = \mathbf{P}\tilde{\mathbf{A}} \tag{3.46}$$

or

$$\tilde{\mathbf{A}} = \mathbf{P}^{-1}\hat{\mathbf{a}} \tag{3.47}$$

where

$$\hat{\mathbf{a}} = \begin{bmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_N \end{bmatrix}, \quad \tilde{\mathbf{A}} = \begin{bmatrix} \tilde{A}_0 \\ \tilde{A}_1 \\ \tilde{A}_2 \\ \vdots \\ \tilde{A}_N \end{bmatrix} = \begin{bmatrix} A_0 c^0 \\ A_1 c^1 \\ A_2 c^2 \\ \vdots \\ A_N c^N \end{bmatrix}$$

Similarly, the relation between the denominator coefficients $\hat{b}_i$ and B_i can be expressed as

$$\hat{\mathbf{b}} = \mathbf{P}\tilde{\mathbf{B}}$$

with $\tilde{B}_k = B_k c^k$.

3.3.2 z-to-s Transformation

Substituting $x = c(1+\alpha' s)/(\mu' + \beta s)$ into $H(x)$ in (3.5) yields

$$\hat{H}_{CT}(s) = H(x)\Big|_{x=c\frac{1+\alpha' s}{\mu'+\beta s}}$$
$$= \frac{\sum_{k=0}^{N} a_k c^k (1+\alpha' s)^k (\mu'+\beta s)^{N-k}}{\sum_{k=0}^{N} b_k c^k (1+\alpha' s)^k (\mu'+\beta s)^{N-k}}$$
$$= \frac{\sum_{k=0}^{N} a_k c^k Q_k(s)}{\sum_{k=0}^{N} b_k c^k Q_k(s)} \qquad (3.48)$$
$$= \frac{\sum_{i=0}^{N} \hat{A}_i s^i}{\sum_{i=0}^{N} \hat{B}_i s^i}$$

where

$$Q_k(s) = (1+\alpha' s)^k (\mu+\beta s)^{N-k} = \sum_{i=0}^{N} q_{i,k} s^i$$
$$q_{i,k} = \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (\mu')^{N-k-n} (\alpha')^{i-n} \beta^n$$
$$= \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-\alpha c)^{N-k-n} (-\frac{\mu}{c})^{i-n} \beta^n \qquad (3.49)$$
$$= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{i-n} \alpha^{N-k-n} \beta^n$$

The numerator of $\hat{H}_{CT}(s)$ can be rewritten as

$$\sum_{i=0}^{N} \hat{A}_i s^i = \sum_{k=0}^{N} a_k c^k Q_k(s)$$
$$= \sum_{k=0}^{N} a_k c^k \left[\sum_{i=0}^{N} q_{i,k} s^i \right]$$
$$= \sum_{i=0}^{N} \left[\sum_{k=0}^{N} a_k c^k q_{i,k} \right] s^i$$

Thus,

$$\hat{A}_i = \sum_{k=0}^{N} q_{i,k} a_k c^k \qquad (3.50)$$

Similarly, we can get

$$\hat{B}_i = \sum_{k=0}^{N} q_{i,k} b_k c^k$$

To derive one-to-one coefficient mappings between the numerator coefficients $A_k \leftrightarrow a_k$ and denominator coefficients and $B_k \leftrightarrow b_k$, the one-to-one mapping

$$s = 0 \Leftrightarrow z = 1$$

is imposed on the numerator/denominator values of $H_{CT}(s)$ and $H(x)$ in the lowpass-to-lowpass (BD,FD, BL1, parametric BD-BL1) transformations, i.e.,

$$\sum_{i=0}^{N} A_i s^i \Big|_{s=0} = \sum_{i=0}^{N} a_i x^i \Big|_{x=1}$$
$$\sum_{i=0}^{N} B_i s^i \Big|_{s=0} = \sum_{i=0}^{N} b_i x^i \Big|_{x=1}$$

which leads to

$$A_0 = \sum_{i=0}^{N} a_i, \quad B_0 = \sum_{i=0}^{N} b_i \tag{3.51}$$

Likewise, the one-to-one mapping

$$s = 0 \Leftrightarrow z = -1$$

is imposed on the lowpass-to-highpass (BL2) transformation, i.e.,

$$\sum_{i=0}^{N} A_i s^i \Big|_{s=0} = \sum_{i=0}^{N} a_i x^i \Big|_{x=-1}$$
$$\sum_{i=0}^{N} B_i s^i \Big|_{s=0} = \sum_{i=0}^{N} b_i x^i \Big|_{x=-1}$$

which leads to

$$A_0 = \sum_{i=0}^{N} a_i (-1)^i, \quad B_0 = \sum_{i=0}^{N} b_i (-1)^i \tag{3.52}$$

Substituting $s = 0$ into (3.48) yields

$$\hat{H}_{CT}(s)\Big|_{s=0} = \frac{\sum_{k=0}^{N} a_k c^k (\mu')^{N-k}}{\sum_{k=0}^{N} b_k c^k (\mu')^{N-k}}$$
$$= \frac{\sum_{k=0}^{N} a_k c^k (-\alpha c)^{N-k}}{\sum_{k=0}^{N} b_k c^k (-\alpha c)^{N-k}} \qquad (3.53)$$
$$= \frac{\sum_{k=0}^{N} a_k c^N (-\alpha)^{N-k}}{\sum_{k=0}^{N} b_k c^N (-\alpha)^{N-k}}$$

For lowpass-to-lowpass transformations ($\alpha = -1$), we have

$$\hat{H}_{CT}(s)\Big|_{s=0} = \frac{\sum_{k=0}^{N} a_k c^N}{\sum_{k=0}^{N} b_k c^N}$$

Considering

$$H_{CT}(s)\Big|_{s=0} = H(x)\Big|_{x=1} = \frac{\sum_{k=0}^{N} a_k}{\sum_{k=0}^{N} b_k}$$

we get the relation between $\hat{A}_i$ and A_i as

$$\frac{\hat{A}_i}{c^N} = A_i \qquad (3.54)$$

On the other hand, for lowpass-to-highpass transformation ($\alpha = 1$), we have

$$\hat{H}_{CT}(s)\Big|_{s=0} = \frac{\sum_{k=0}^{N} a_k \cdot (-1)^{N-k} \cdot c^N}{\sum_{k=0}^{N} b_k \cdot (-1)^{N-k} \cdot c^N}$$
$$= \frac{\sum_{k=0}^{N} a_k \cdot (-1)^k \cdot (-c)^N}{\sum_{k=0}^{N} b_k \cdot (-1)^k \cdot (-c)^N}$$

Considering

$$H_{CT}(s)\Big|_{s=0} = H(x)\Big|_{x=-1} = \frac{\sum_{k=0}^{N} a_k \cdot (-1)^k}{\sum_{k=0}^{N} b_k \cdot (-1)^k}$$

we obtain the relation

$$\frac{\hat{A}_i}{(-c)^N} = A_i \tag{3.55}$$

3.3.3 Inverse Matrix (BD Case)

For BD transformation, using the element expression (3.34) and substituting the parameter values $(\mu, \alpha, \beta) = (1, -1, 0)$ into (3.49) yields

$$q_{i,k} = (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{i-n} \alpha^{N-k-n} \beta^n$$
$$= \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-c)^{N-k-i} \mu^{i-n} (-1)^{N-k-n} 0^n$$
$$= \binom{k}{i} (-1)^{N-k} \cdot (-c)^{N-k-i} \tag{3.56}$$
$$= \binom{k}{i} (-1)^i \cdot c^{N-k-i}$$
$$= p_{i,k} \cdot c^{N-k-i}$$

Thus,

$$\begin{aligned}\hat{A}_i &= \sum_{k=0}^{N} a_k c^k \cdot q_{i,k} \\ &= \sum_{k=0}^{N} a_k c^k \cdot c^{N-k-i} p_{i,k} \\ &= c^{N-i} \cdot \sum_{k=0}^{N} p_{i,k} a_k\end{aligned}$$

and

$$\frac{\hat{A}_i}{c^N} = c^{-i} \cdot \sum_{k=0}^{N} p_{i,k} a_k = A_i$$

i.e.,

$$\sum_{k=0}^{N} p_{i,k} a_k = A_i c^i = \tilde{A}_i$$

which can be expressed in matrix form as

$$\mathbf{P}\mathbf{a} = \tilde{\mathbf{A}} \tag{3.57}$$

or

$$\mathbf{a} = \mathbf{P}^{-1}\tilde{\mathbf{A}} \tag{3.58}$$

Considering

$$\hat{H}(x)\Big|_{x=1} = \frac{\sum_{k=0}^{N} A_k c^k (1-1)^k (1+0)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1-1)^k (1+0)^{N-k}} = \frac{A_0}{B_0}$$

$$H(x)\Big|_{x=1} = H_{CT}(s)\Big|_{s=0} = \frac{A_0}{B_0}$$

we get the relation between $\hat{a}_i$ and a_i as

$$\hat{a}_i = a_i$$

i.e.,

$$\hat{\mathbf{a}} = \mathbf{a} \tag{3.59}$$

Substituting (3.59) into (3.46) yields

$$\mathbf{a} = \mathbf{P}\tilde{\mathbf{A}} \tag{3.60}$$

and combining (3.58) with (3.60) arrives at

$$\mathbf{P}^{-1} = \mathbf{P} \tag{3.61}$$

For example, if $N = 3$, using (3.34) yields

$$\mathbf{P} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & -2 & -3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

Then

$$\mathbf{P}^{-1} = \mathbf{P} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & -2 & -3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$\mathbf{P}\mathbf{P}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.3.4 Inverse Matrix (FD Case)

For FD transformation, using the element expression (3.36) and substituting $(\mu, \alpha, \beta) = (0, -1, 1)$ into (3.49) leads to

$$\begin{aligned} q_{i,k} &= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{i-n} \alpha^{N-k-n} \beta^{n} \\ &= \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-c)^{N-k-i} \cdot 0^{i-n} \cdot (-1)^{N-k-n} \cdot 1^{n} \\ &= \binom{N-k}{i} \cdot (-1)^{N-k-i} \cdot (-c)^{N-k-i} \\ &= c^{N-k-i} \cdot \binom{N-k}{i} \end{aligned} \tag{3.62}$$

Hence,

$$\begin{aligned}\hat{A}_i &= \sum_{k=0}^{N} a_k c^k \cdot q_{i,k} \\ &= \sum_{k=0}^{N} a_k c^k \cdot c^{N-k-i} \cdot \binom{N-k}{i} \\ &= c^{N-i} \cdot \sum_{k=0}^{N} a_k \cdot \binom{N-k}{i}\end{aligned}$$

and

$$\frac{\hat{A}_i}{c^N} = c^{-i} \cdot \sum_{k=0}^{N} a_k \cdot \binom{N-k}{i} = A_i$$

i.e.,

$$\sum_{k=0}^{N} \hat{p}_{i,k} a_k = A_i c^i = \tilde{A}_i \tag{3.63}$$

with

$$\hat{p}_{i,k} = \binom{N-k}{i} \tag{3.64}$$

Eq.(3.63) can be rewritten in matrix form as

$$\hat{\mathbf{P}}\mathbf{a} = \tilde{\mathbf{A}} \tag{3.65}$$

Considering

$$\hat{H}(x)\Big|_{x=1} = \frac{\sum_{k=0}^{N} A_k c^k (1-1)^k (1+0)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1-1)^k (1+0)^{N-k}} = \frac{A_0}{B_0}$$

$$H(x)\Big|_{x=1} = H_{CT}(s)\Big|_{s=0} = \frac{A_0}{B_0}$$

we get the relation between $\hat{a}_i$ and a_i as

$$\hat{a}_i = a_i$$

i.e.,

$$\hat{\mathbf{a}} = \mathbf{a}.$$

Substituting $\hat{\mathbf{a}} = \mathbf{a}$ into (3.47) yields

$$\mathbf{P}^{-1}\mathbf{a} = \tilde{\mathbf{A}} \tag{3.66}$$

and combining (3.65) with (3.66) yields the inverse matrix

$$\mathbf{P}^{-1} = \hat{\mathbf{P}} \tag{3.67}$$

It is clear from (3.36) and (3.64) that the matrix $\hat{\mathbf{P}}(\mathbf{P}^{-1})$ can be obtained by flipping the rows and columns of $\mathbf{P}$, and then multiplying the resulting elements by $(-1)^{N-(k+i)}$. For example, if $N = 3$, using (3.36) gets

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -3 \\ 0 & 1 & -2 & 3 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

Then, its inverse $\mathbf{P}^{-1}$ can be obtained through computing $\hat{\mathbf{P}}$ by flipping the rows and columns of $\mathbf{P}$, and changing the signs of elements as

$$\mathbf{P}^{-1} = \hat{\mathbf{P}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \\ 3 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.68}$$

Consequently, we can verify

$$\mathbf{P}\mathbf{P}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

It should be noted here that the inverse for the FD case given in [17] is incorrect for odd N, where

$$\hat{p}_{i,k} = (-1)^{i+k} p_{N-i,N-k} \tag{3.69}$$

For example, if $N = 3$, then (3.69) leads to

$$\mathbf{P}^{-1} = \hat{\mathbf{P}} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ -3 & -2 & -1 & 0 \\ -3 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix} \tag{3.70}$$

and

$$\mathbf{P}\mathbf{P}^{-1} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

The main reason for such an incorrect inverse is that the inverses mentioned in [17] are based on observing some concrete examples of the unified Pascal matrices, but no rigorous proofs are provided.

3.3.5 Inverse Matrix (BL1 Case)

For BL1 transformation, using the element expression (3.38) and substituting $(\mu, \alpha, \beta) = (1, -1, 1)$ into (3.49) yields

$$\begin{aligned} q_{i,k} &= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{i-n} \alpha^{N-k-n} \beta^{n} \\ &= \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-c)^{N-k-i} \cdot (-1)^{N-k-n} \\ &= c^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-1)^{i-n} \\ &= c^{N-k-i} \cdot p_{i,k} \end{aligned}$$

thus

$$\hat{A}_i = \sum_{k=0}^{N} a_k c^k \cdot q_{i,k}$$
$$= \sum_{k=0}^{N} a_k c^k \cdot c^{N-k-i} p_{i,k}$$
$$= c^{N-i} \cdot \sum_{k=0}^{N} a_k p_{i,k}$$

and

$$\frac{\hat{A}_i}{c^N} = c^{-i} \cdot \sum_{k=0}^{N} a_k p_{i,k} = A_i$$

i.e.,

$$\sum_{k=0}^{N} p_{i,k} a_k = A_i c^i = \tilde{A}_i$$

which can be expressed in matrix form as

$$\mathbf{Pa} = \tilde{\mathbf{A}} \tag{3.71}$$

or

$$\mathbf{a} = \mathbf{P}^{-1}\tilde{\mathbf{A}} \tag{3.72}$$

Considering

$$\hat{H}(x)\Big|_{x=1} = \frac{\sum_{k=0}^{N} A_k c^k (1-1)^k (1+0)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1-1)^k (1+0)^{N-k}} = \frac{A_0}{B_0}$$

$$H(x)\Big|_{x=1} = H_{CT}(s)\Big|_{s=0} = \frac{A_0}{B_0}$$

we get the relation between $\hat{a}_i$ and a_i as

$$\frac{\hat{a}_i}{2^N} = a_i$$

i.e.,

$$\hat{\mathbf{a}} = 2^N \mathbf{a}$$

Substituting $\hat{\mathbf{a}} = 2^N \mathbf{a}$ into (3.46) yields

$$\mathbf{a} = 2^{-N} \mathbf{P} \tilde{\mathbf{A}} \tag{3.73}$$

and comparing (3.72) with (3.73) gets the inverse matrix

$$\mathbf{P}^{-1} = 2^{-N} \mathbf{P} \tag{3.74}$$

3.3.6 Inverse Matrix (Parametric BD-BL1 Case)

For parametric BD-BL1 transformation, using the element expression (3.42) and substituting $(\mu, \alpha, \beta) = (1, -1, r)$ into (3.49) obtains

$$\begin{aligned} q_{i,k} &= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{i-n} \alpha^{N-k-n} \beta^n \\ &= (-c)^{N-k-i} \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-1)^{N-k-n} \cdot r^n \\ &= c^{N-k-i} \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-1)^{i-n} \cdot r^n \\ &= c^{N-k-i} \cdot p_{i,k} \end{aligned}$$

Thus,

$$\begin{aligned} \hat{A}_i &= \sum_{k=0}^{N} a_k c^k \cdot q_{i,k} \\ &= \sum_{k=0}^{N} a_k c^k \cdot c^{N-k-i} p_{i,k} \\ &= c^{N-i} \cdot \sum_{k=0}^{N} a_k p_{i,k} \end{aligned}$$

and

$$\frac{\hat{A}_i}{c^N} = c^{-i} \cdot \sum_{k=0}^{N} a_k p_{i,k} = A_i$$

i.e.,

$$\sum_{k=0}^{N} p_{i,k} a_k = A_i c^i = \tilde{A}_i$$

$$\mathbf{P}\mathbf{a} = \tilde{\mathbf{A}}$$

or

$$\mathbf{a} = \mathbf{P}^{-1}\tilde{\mathbf{A}} \tag{3.75}$$

Considering

$$\hat{H}(x)\Big|_{x=1} = \frac{\sum_{k=0}^{N} A_k c^k (1-1)^k (1+r)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1-1)^k (1+r)^{N-k}}$$

$$= \frac{A_0 \cdot (1+r)^N}{B_0 \cdot (1+r)^N}$$

$$H(x)\Big|_{x=1} = H_{CT}(s)\Big|_{s=0} = \frac{A_0}{B_0}$$

we get the relation between $\hat{a}_i$ and a_i as

$$\frac{\hat{a}_i}{(1+r)^N} = a_i$$

i.e.,

$$\hat{\mathbf{a}} = (1+r)^N \mathbf{a}$$

Substituting $\hat{\mathbf{a}} = (1+r)^N \mathbf{a}$ into (3.46) yields

$$\mathbf{a} = (1+r)^{-N} \mathbf{P}\tilde{\mathbf{A}} \tag{3.76}$$

and comparing (3.75) with (3.76) arrives at

$$\mathbf{P}^{-1} = (1+r)^{-N} \mathbf{P} \tag{3.77}$$

3.3.7 Inverse Matrix (BL2 Case)

For BL2 transformation, using the element expression (3.40) and substituting $(\mu, \alpha, \beta) = (1, 1, -1)$ into (3.49) obtains

$$
\begin{aligned}
q_{i,k} &= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} \mu^{i-n} \alpha^{N-k-n} \beta^{n} \\
&= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} 1^{i-n} \cdot 1^{N-k-n} \cdot (-1)^{n} \\
&= (-c)^{N-k-i} \cdot \sum_{n=0}^{i} \binom{N-k}{n} \binom{k}{i-n} (-1)^{n} \\
&= (-c)^{N-k-i} \cdot p_{i,k}
\end{aligned}
$$

thus

$$
\begin{aligned}
\hat{A}_i &= \sum_{k=0}^{N} a_k c^k \cdot q_{i,k} \\
&= \sum_{k=0}^{N} a_k c^k \cdot (-c)^{N-k-i} p_{i,k} \\
&= \sum_{k=0}^{N} a_k c^k \cdot (-c)^{N} \cdot (-c)^{-k-i} p_{i,k} \\
&= \sum_{k=0}^{N} a_k \cdot (-c)^{N} \cdot c^{k} \cdot c^{-k-i} \cdot (-1)^{-k-i} p_{i,k} \\
&= \sum_{k=0}^{N} a_k (-c)^{N} \cdot c^{-i} \cdot (-1)^{k+i} p_{i,k}
\end{aligned}
$$

and

$$
\frac{\hat{A}_i}{(-c)^N} = c^{-i} \cdot \sum_{k=0}^{N} a_k (-1)^{k+i} p_{i,k} = A_i
$$

Let

$$
\tilde{p}_{i,k} = (-1)^{k+i} p_{i,k} \tag{3.78}
$$

we get

$$
\sum_{k=0}^{N} \tilde{p}_{i,k} a_k = A_i c^i = \tilde{A}_i
$$

i.e.,

$$
\tilde{\mathbf{P}}\mathbf{a} = \tilde{\mathbf{A}} \tag{3.79}
$$

On the other hand, utilizing the one-to-one mapping $s = 0 \Leftrightarrow z = -1$ for the lowpass-to-highpass (BL2) transformation, we have

$$\hat{H}(x)\Big|_{x=-1} = \frac{\sum_{k=0}^{N} A_k c^k (1-1)^k (1+1)^{N-k}}{\sum_{k=0}^{N} B_k c^k (1-1)^k (1+1)^{N-k}} = \frac{A_0 \cdot 2^N}{B_0 \cdot 2^N}$$

$$H(x)\Big|_{x=-1} = H_{CT}(s)\Big|_{s=0} = \frac{A_0}{B_0}$$

Hence, the relation between $\hat{a}_i$ and a_i is

$$\frac{\hat{a}_i}{2^N} = a_i$$

i.e.,

$$\hat{\mathbf{a}} = 2^N \mathbf{a} \tag{3.80}$$

Substituting $\hat{\mathbf{a}} = 2^N \mathbf{a}$ into (3.46) yields

$$2^N \mathbf{a} = \mathbf{P}\tilde{\mathbf{A}}$$

i.e.,

$$\tilde{\mathbf{A}} = 2^N \mathbf{P}^{-1}\mathbf{a} \tag{3.81}$$

Comparing (3.79) with (3.81) yields the inverse matrix

$$\mathbf{P}^{-1} = 2^{-N}\tilde{\mathbf{P}} \tag{3.82}$$

Finally, let us see the example in [17], where $c_4 = 1$, and the CT transfer function is

$$H_{CT}(s) = \frac{5+s^2}{5+4s+3s^2+s^3} \tag{3.83}$$

i.e.,

$$\begin{bmatrix} \tilde{\mathbf{A}} & \tilde{\mathbf{B}} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix} = \begin{bmatrix} 5 & 5 \\ 0 & 4 \\ 1 & 3 \\ 0 & 1 \end{bmatrix} \tag{3.84}$$

The unified Pascal matrix for this $N = 3$ case

$$\mathbf{P} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -3 & -1 & 1 & 3 \\ 3 & -1 & -1 & 3 \\ -1 & 1 & -1 & 1 \end{bmatrix}$$

and $\tilde{\mathbf{P}}$ can be computed using (3.78) as

$$\tilde{\mathbf{P}} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 3 & -1 & -1 & 3 \\ 3 & 1 & -1 & -3 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

As a result, the inverse can be computed as

$$\mathbf{P}^{-1} = 2^{-N}\tilde{\mathbf{P}} = \begin{bmatrix} 0.125 & -0.125 & 0.125 & -0.125 \\ 0.375 & -0.125 & -0.125 & 0.375 \\ 0.375 & 0.125 & -0.125 & -0.375 \\ 0.125 & 0.125 & 0.125 & 0.125 \end{bmatrix}$$

Using (3.18), we yield

$$\begin{aligned} \begin{bmatrix} \hat{\mathbf{a}} & \hat{\mathbf{b}} \end{bmatrix} &= \mathbf{P}\mathbf{D}_c\begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 1 & 1 \\ -3 & -1 & 1 & 3 \\ 3 & -1 & -1 & 3 \\ -1 & 1 & -1 & 1 \end{bmatrix}\begin{bmatrix} 5 & 5 \\ 0 & 4 \\ 1 & 3 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 6 & 13 \\ -14 & -13 \\ 14 & 11 \\ -6 & -3 \end{bmatrix} \end{aligned} \tag{3.85}$$

and using the relation (3.80) leads to

$$[\mathbf{a} \quad \mathbf{b}] = 2^{-N}[\hat{\mathbf{a}} \quad \hat{\mathbf{b}}] = \begin{bmatrix} 0.75 & 1.625 \\ -1.75 & -1.625 \\ 1.75 & 1.375 \\ -0.75 & -0.375 \end{bmatrix} \tag{3.86}$$

Consequently, we obtain a DT transfer function

$$H(z) = \frac{0.75 - 1.75z^{-1} + 1.75z^{-2} - 0.75z^{-3}}{1.625 - 1.625z^{-1} + 1.375z^{-2} - 0.375z^{-3}}$$

Using (3.81), we can recover the original coefficient vectors $[\mathbf{A} \quad \mathbf{B}]$ of the CT transfer function from $[\mathbf{a} \quad \mathbf{b}]$ as

$$[\mathbf{A} \quad \mathbf{B}] = 2^{N}\mathbf{P}^{-1}[\mathbf{a} \quad \mathbf{b}] = \begin{bmatrix} 5 & 5 \\ 0 & 4 \\ 1 & 3 \\ 0 & 1 \end{bmatrix} \tag{3.87}$$

In contrast, the DT transfer function $H(z)$ obtained in [17] is

$$H(z) = \frac{6 - 14z^{-1} + 14z^{-2} - 6z^{-3}}{13 - 13z^{-1} + 11z^{-2} - 3z^{-3}}$$

i.e.,

$$[\mathbf{a} \quad \mathbf{b}] = \begin{bmatrix} 6 & 13 \\ -14 & -13 \\ 14 & 11 \\ -6 & -3 \end{bmatrix} \tag{3.88}$$

the original coefficient vectors $[\mathbf{A} \quad \mathbf{B}]$ cannot be recovered because the z-to-s relation (3.81) leads to

$$[\mathbf{A} \quad \mathbf{B}] = [\tilde{\mathbf{A}} \quad \tilde{\mathbf{B}}] = 2^N \mathbf{P}^{-1} [\mathbf{a} \quad \mathbf{b}] = \begin{bmatrix} 0 & 8 \\ -8 & -24 \\ 0 & 32 \\ -40 & -40 \end{bmatrix} \tag{3.89}$$

that are different from the original (3.84).

3.4 Property and Proof

In this section, we prove that the sum of all the elements of each column (except for the first column for BD, FD, BL1, and parametric BD-BL1 and the last column for BL2) is zero. Let S_k represent the sum of all the elements of the $(k+1)$th column of the unified Pascal matrix $\mathbf{P}$, i.e.,

$$S_k = \sum_{i=0}^{N} p_{i,k}, \quad k = 0,1,\cdots,N$$

It follows from (3.14) that

$$\begin{aligned} S_k &= P_k(x)\big|_{x=1} \\ &= (1+\alpha x)^k (\mu+\beta x)^{N-k}\big|_{x=1} \\ &= (1+\alpha)^k (\mu+\beta)^{N-k} \end{aligned}$$

Therefore, we obtain S_k for BD, FD, BL1, parametric BD-BL1, and BL2 transformations as follows:

♦ For the BD case, $(\mu,\alpha,\beta) = (1,-1,0)$, then

$$S_k = (1-1)^k (1+0)^{N-k} = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

♦ For the FD case, $(\mu,\alpha,\beta) = (0,-1,1)$, then

$$S_k = (1-1)^k (0+1)^{N-k} = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

♦ For the BL1 case, $(\mu,\alpha,\beta)=(1,-1,1)$, then

$$S_k=(1-1)^k(1+1)^{N-k}=\begin{cases}2^N, & k=0\\ 0, & k\neq 0\end{cases}$$

♦ For the parametric BD-BL1 case, $(\mu,\alpha,\beta)=(1,-1,r)$, then

$$S_k=(1-1)^k(1+r)^{N-k}=\begin{cases}(1+r)^N, & k=0\\ 0, & k\neq 0\end{cases}$$

♦ For the BL2 case, $(\mu,\alpha,\beta)=(1,1,-1)$, then

$$S_k=(1+1)^k(1-1)^{N-k}=\begin{cases}2^N, & k=0\\ 0, & k\neq 0\end{cases}$$

3.5 Catastrophic Cancellation Problem

The recurrence formulas (3.35), (3.37), (3.39), (3.41), and (3.43) provide a recurrent way for generating the inner elements of the unified Pascal matrices from their first rows and last columns for the BD, FD, BL1, BL2, and parametric BD-BL1 transformations, respectively. However, for high-order system transformations, such recurrence formulas lead to large errors due to the so-called catastrophic cancellation problem in numerical computation. Assume that the unified Pascal matrices computed using the explicit element expressions (3.34), (3.36), (3.38), (3.40), and (3.42) are true values, then the recurrence formulas generate unified Pascal matrices that are far away from the true ones and even useless for high-order system transformations.

To evaluate the generation errors, we assume that $\mathbf{P}_\mathbf{T}$ denotes the true unified Pascal matrix for some *s*-*z* transformation, and $\mathbf{P}_\mathbf{R}$ denotes the one generate from a recurrence formula, then the normalized root-mean-squared (RMS) error is

$$\varepsilon_2=\frac{\|\mathbf{P}_\mathbf{T}-\mathbf{P}_\mathbf{R}\|}{\|\mathbf{P}_\mathbf{T}\|}\times 100\% \tag{3.90}$$

where $\|\cdot\|$ denotes the Euclidean norm. As an example, let us consider the BL1 transformation. Using the recurrence formula (3.39), we obtain the error ε_2 versus the system order N as shown

in Table 3.2. It can be observed from Table 3.2 that the error ε_2 increases drastically as N increases. The errors are caused by the catastrophic cancellation and such errors are gradually accumulated as N increases, which makes the recurrent computation useless for high-order cases. To solve this problem, we can partition the whole unified Pascal matrix into several small matrices (sub-matrices) and then compute the first rows and last columns of the sub-matrices by using the explicit element expression (3.38). Finally, the recurrence formula (3.39) are used to generate the sub-matrices separately.

Another approach is to first compute all the boundary elements (first and last rows, first and last columns) as shown in (3.25), then the inner elements are computed individually from the four corners through modifying the recurrence formula (3.39). It follows from (3.39) that

$$\begin{aligned} p_{i,k+1} &= p_{i,k} - p_{i-1,k} - p_{i-1,k+1} \\ &\Downarrow \\ p_{i,k} &= p_{i,k-1} - p_{i-1,k-1} - p_{i-1,k} \end{aligned} \tag{3.91}$$

Thus, the recursive computation can be started from the elements of the first row and first column. Similarly, (3.39) can be modified to

$$\begin{aligned} p_{i-1,k+1} &= p_{i,k} - p_{i,k+1} - p_{i-1,k} \\ &\Downarrow \\ p_{i,k} &= p_{i+1,k-1} - p_{i+1,k} - p_{i,k-1} \end{aligned} \tag{3.92}$$

i.e., the recurrent computation can be started from the elements of the first column and last row. Finally, we can modify (3.39) to

$$\begin{aligned} p_{i-1,k} &= p_{i,k} - p_{i,k+1} - p_{i-1,k+1} \\ &\Downarrow \\ p_{i,k} &= p_{i+1,k} - p_{i+1,k+1} - p_{i,k+1} \end{aligned} \tag{3.93}$$

i.e., the recurrent computation can also be started from the elements of the last row and last column. Consequently, the whole unified Pascal matrix can be partitioned into four sub-matrices with almost the same size, and then the four sub-matrices are individually generated starting from the above four corners (directions). Such partitions reduce the number of iterations for each block

generation and thus avoid the error accumulation. Table 3.2 also shows the error ε_2 versus system order N when using this four direction approach. Clearly, the errors are small enough and neglectable. For other s-z transformations, the catastrophic cancellation problem can be similarly solved through modifying the corresponding recurrence formulas and using the explicit element expressions.

Table 3.2 Normalized RMS Error ε_2 (%) Versus System Order

Order N	Recurrence Formula (39)	Four-Direction
50	0.0000	0.0000
51	0.0000	0.0000
52	0.0000	0.0000
53	0.0000	0.0000
54	5.2252×10^{15}	0.0125×10^{-6}
55	9.0465×10^{15}	0.0016×10^{-6}
56	5.2199×10^{16}	0.0352×10^{-6}
57	1.0868×10^{17}	0.0055×10^{-6}
58	2.0446×10^{18}	0.8294×10^{-6}
⋮	⋮	⋮

3.6 Conclusion

A unified first-order s-z transformation model has been used to derive a more general Pascal matrix (unified Pascal matrix) so that both lowpass-to-lowpass and lowpass-to-highpass transformations are included in this single model. The main contributions of this chapter can be summarized as follows.

- The inverses of unified Pascal matrices for various s-z transformations have been rigorously proved by using the one-to-one coefficient mapping between the rational CT and DT transfer functions.
- An explicit formula for expressing the elements of the unified Pascal matrix has been derived, which provides a direct way for generating a unified Pascal matrix.
- An interesting property of the unified Pascal matrix has been proved.
- A partitioning method has been proposed for solving the catastrophic cancellation problem such that the recurrence formulas can still be efficiently utilized along with the explicit element expressions to generate numerically stable unified Pascal matrices.

Chapter 4

Pascal Matrix Operations for Unified Bilinear s-z Transformation

4.1 Introduction

This chapter presents an alternative method used for analog domain to digital domain transformation based on bilinear transformation which is obtained from the previous chapter. The Pascal matrix for bilinear transformation is used to transform analog transfer function on s plane to digital transfer function on z plane. Moreover, the frequency transformation from normalized analog lowpass filter prototype to digital lowpass, highpass, bandpass and bandstop filter will be considered to incorporate the Pascal matrix operation. The proposed method can clarify the computation complexity of original bilinear transform, especially to solve difficulties of the higher order of digital filter because all of the computations use matrix operations. Therefore, it is easy and appropriate to program this method on a personal computer or scientific calculator.

4.2 Pascal Matrix for Bilinear Transformation

This part reviews the concept of Pascal matrix operations from the previous chapter that used bilinear transformation which came from the binomial theorem as follows,

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k \tag{4.1}$$

where $\binom{n}{k} = \frac{n!}{(n-k)!k!}$ that called binomial coefficients and each n has binomial coefficients in the form of Pascal triangle. The important relationship between s-z of bilinear transform as

$$s = \frac{1-z^{-1}}{1+z^{-1}} \tag{4.2}$$

Consider the N^{th} order analog and digital transfer function as follows,

$$H(s) = \frac{\sum_{i=0}^{N} A_i s^i}{\sum_{i=0}^{N} B_i s^i} \tag{4.3}$$

and

$$H(z) = \frac{\sum_{i=0}^{N} a_i z^{-i}}{\sum_{i=0}^{N} b_i z^{-i}} \tag{4.4}$$

Using eq. (4.3) and the relationship in eq. (4.2) will give

$$H(z) = \frac{\sum_{i=0}^{N} A_i \left[\frac{1-z^{-1}}{1+z^{-1}}\right]^i}{\sum_{i=0}^{N} B_i \left[\frac{1-z^{-1}}{1+z^{-1}}\right]^i} = \frac{\sum_{i=0}^{N} A_i (1-z^{-1})^i (1+z^{-1})^{N-i}}{\sum_{i=0}^{N} B_i (1-z^{-1})^i (1+z^{-1})^{N-i}} \tag{4.5}$$

From eq. (4.5) can use the binomial theorem to distribute and consider only numerator part, also

$$\sum_{i=0}^{N} A_i (1-z^{-1})^i (1+z^{-1})^{N-i} = \sum_{i=0}^{N} A_i \left[\sum_{k=0}^{i} \binom{i}{k} (-1)^k z^{-k} \sum_{k=0}^{N-i} \binom{N-i}{k} z^{-k} \right] \tag{4.6}$$

As shown in [11-13] and in chapter 2 and 3, the terms in bracket will be formed as the matrix size $(N+1)\times(N+1)$ that called Pascal matrix for bilinear transform. With rearranging and changing, a new index can show the relationship between eq. (4.5) and eq. (4.4) as

$$\sum_{i=0}^{N} a_i z^{-i} = \sum_{i=0}^{N} A_i \left[P_{i,j}\right] z^{-i} \tag{4.7}$$

where the Pascal matrix $[P_{i,j}]$ can be defined as

$$P_{i,j} = \sum_{n=0}^{i} \binom{N-j}{n} \binom{j}{i-n} (-1)^{i-n} \qquad ; i, j = 0,1,\ldots,N \tag{4.8}$$

i and j means row and column index, respectively.

Also, the column vector of digital filter coefficients can be computed by vector product of Pascal matrix with column vector of analog filter coefficients as follows,

$$[a_i] = [P_{i,j}][A_i] \tag{4.9}$$

The denominator can be computed in the same manner as in eq. (4.9), only change $[a_i]$ and $[A_i]$ to $[b_i]$ and $[B_i]$, respectively.

4.3 The Frequency Transformation

The operations in eq. (4.9) is only bilinear transformation from analog to digital domain, but in practical use often uses the normalized analog filter prototype at the start of design. Therefore, the frequency scaling must be used to incorporate the Pascal matrix and this result in proposed bilinear transform involves two separate transformations: normal bilinear transform as shown in eq. (4.9) and the frequency transformation. At first, the normalized analog transfer function is frequency scaled by replacing as follows,

$$s = \frac{s}{\Omega_C} \qquad \text{for lowpass to lowpass} \tag{4.10}$$

$$s = \frac{\Omega_C}{s} \qquad \text{for lowpass to highpass} \tag{4.11}$$

$$s = \frac{s^2 + \Omega_{C1}\Omega_{C2}}{s(\Omega_{C2} - \Omega_{C1})} \qquad \text{for lowpass to bandpass} \tag{4.12}$$

$$s = \frac{s(\Omega_{C2} - \Omega_{C1})}{s^2 + \Omega_{C1}\Omega_{C2}} \qquad \text{for lowpass to bandstop} \tag{4.13}$$

Hence, apply the bilinear transform by replacing s follow by eq. (4.2) in the new transfer function and using the relationship between analog frequency and digital frequency is

$$\Omega = \tan\frac{\omega T}{2} \tag{4.14}$$

For computational efficiency, the two transformations can be combined into one as follows,

$$s = \frac{1-z^{-1}}{1+z^{-1}} \Big/ \tan\frac{\omega_C T}{2} = \cot\left(\frac{\omega_C T}{2}\right)\left(\frac{1-z^{-1}}{1+z^{-1}}\right)$$

given $T = 1/F_s$ and $\omega_C = 2\pi f_C$, it can be got

$$s = \cot\left(\frac{\pi f_C}{F_S}\right)\frac{1-z^{-1}}{1+z^{-1}} = c\frac{1-z^{-1}}{1+z^{-1}} \tag{4.15}$$

Eq. (4.15) is an example for analog lowpass to digital lowpass transformation where constant c is value of $\cot(\pi f_C / F_s)$. The others can be proven in the same manner that is summarized as follows,

Analog lowpass to digital highpass transformation is:

$$s = k\frac{1+z^{-1}}{1-z^{-1}} \quad \text{where } k = \tan(\pi f_C / F_s) \tag{4.16}$$

Analog lowpass to digital bandpass transformation is:

$$s = \hat{c}\left[\frac{1-z^{-1}}{1+z^{-1}} + \hat{k}\frac{1+z^{-1}}{1-z^{-1}}\right] \tag{4.17}$$

Analog lowpass to digital bandstop transformation is:

$$s = 1\Big/\hat{c}\left[\frac{1-z^{-1}}{1+z^{-1}} + \hat{k}\frac{1+z^{-1}}{1-z^{-1}}\right] \tag{4.18}$$

where $\hat{k} = k_1 k_2$ and $\hat{c} = 1/k_2 - k_1$, by $k_1 = \tan(\pi f_{C1} / F_s)$ and $k_2 = \tan(\pi f_{C2} / F_s)$, respectively.

In order to transform from analog to digital filter by eqs. (4.15)-(4.18), The Pascal matrix is the same as in eq. (4.8), except for bandpass and bandstop transformation in which the size of the matrix will double. Properties of the Pascal matrix, can be simplified in order to create the Pascal matrix from the first row and the first column of the matrix using the recurrence formula.

The first row: $$P_{0,j} = 1 \qquad \text{for all } j \tag{4.19}$$

The first column: $$P_{i,0} = \binom{N}{i} \qquad \text{for all } i \tag{4.20}$$

Each element in Pascal matrix can be obtained by

$$P_{i,j} = P_{i,j-1} - P_{i-1,j-1} - P_{i-1,j} \tag{4.21}$$

Except for highpass transformation, eq. (4.20) and (4.21) will change to

The first column: $$P_{i,0} = \binom{N}{i}(-1)^i \tag{4.22}$$

Each element in Pascal matrix can be obtained by

$$P_{i,j} = P_{i,j-1} + P_{i-1,j-1} + P_{i-1,j} \tag{4.23}$$

Moreover, each of the constants that happened from the frequency transformation will be combined in matrix operations, when compared with eq. (4.9), it can be rewritten as follows,

$$[a_i] = [P_{i,j}][\tilde{A}_i] = [P_{i,j}][Aux_{i,j}][A_i] \tag{4.24}$$

The matrix $[\tilde{A}_i]$ is the product between auxiliary or frequency scaling coefficient (FSC) matrix with original vector $[A_i]$. The auxiliary or FSC matrix occurs from the effect of frequency transformation which can be defined as follows,

For lowpass transformation:

$$Aux_{i,j}^{(LP)} = \begin{cases} c^i & ; i = j \\ 0 & ; others \end{cases} \tag{4.25}$$

For highpass transformation:

$$Aux_{i,j}^{(HP)} = \begin{cases} k^i & ; i = j \\ 0 & ; others \end{cases} \tag{4.26}$$

From eqs. (4.25) and (4.26), notice that these auxiliary matrix are diagonal matrix as shown in eqs. (4.27) and (4.28), respectively.

$$[Aux_{i,j}^{(LP)}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & c & 0 & 0 & 0 \\ 0 & 0 & c^2 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & c^N \end{bmatrix} \tag{4.27}$$

$$[Aux_{i,j}^{(HP)}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & k & 0 & 0 & 0 \\ 0 & 0 & k^2 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & k^N \end{bmatrix} \tag{4.28}$$

In case of bandpass and bandstop transformation the auxiliary matrix will be divided into 2 matrix as follows,

$$[Aux_{i,j}] = [Aux1_{i,j}][Aux2_{i,j}] \tag{4.29}$$

The matrix $[Aux1_{i,j}]$ have size $(2N+1)\times(N+1)$, since the order of bandpass and bandstop transformation will be double. The matrix $[Aux2_{i,j}]$ still has size $(N+1)\times(N+1)$.

For bandpass transformation: The non-zero elements in matrix $[Aux1_{i,j}^{(BP)}]$ can be obtained by

$$Aux1_{i,j}^{(BP)} = \frac{j!\hat{k}^{(j+N-i)/2}}{\left(\frac{j+N-i}{2}\right)!\left(\frac{i+j-N}{2}\right)!} \tag{4.30}$$

where $i = 0, 1, 2, \ldots, 2N$ and $j = 0, 1, 2, \ldots, N$

Example in case N=2

$$[Aux1_{i,j}^{(BP)}] = \begin{bmatrix} 0 & 0 & \hat{k}^2 \\ 0 & \hat{k} & 0 \\ 1 & 0 & 2\hat{k} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The matrix $[Aux2_{i,j}^{(BP)}]$ can be obtained by

$$Aux2_{i,j}^{(BP)} = \begin{cases} \hat{c}^i & ; i = j \\ 0 & ; others \end{cases} \tag{4.31}$$

For bandstop transformation: The non-zero elements in matrix $[Aux1_{i,j}^{(BS)}]$ can be obtained by

$$Aux1_{i,j}^{(BS)} = \frac{(N-j)!\hat{k}^{N-(i+j)/2}}{N-\left(\frac{i+j}{2}\right)!\left(\frac{i-j}{2}\right)!} \tag{4.32}$$

where $i = 0, 1, 2, \ldots, 2N$ and $j = 0, 1, 2, \ldots, N$

Example in case N=2

$$[Aux1_{i,j}^{(BS)}]=\begin{bmatrix}\hat{k}^2 & 0 & 0\\ 0 & \hat{k} & 0\\ 2\hat{k} & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{bmatrix}$$

The matrix $[Aux2_{i,j}^{(BS)}]$ can be found by

$$Aux2_{i,j}^{(BS)}=\begin{cases}\hat{c}^{N-i} & ;i=j\\ 0 & ;others\end{cases} \tag{4.33}$$

From eqs. (4.31) and (4.33), notice that these auxiliary matrix are diagonal matrix as shown in eqs. (4.34) and (4.35), respectively.

$$[Aux2_{i,j}^{(BP)}]=\begin{bmatrix}1 & 0 & 0 & 0 & 0\\ 0 & \hat{c} & 0 & 0 & 0\\ 0 & 0 & \hat{c}^2 & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & 0\\ 0 & 0 & 0 & 0 & \hat{c}^N\end{bmatrix} \tag{4.34}$$

$$[Aux2_{i,j}^{(BS)}]=\begin{bmatrix}\hat{c}^N & 0 & 0 & 0 & 0\\ 0 & \hat{c}^{N-1} & 0 & 0 & 0\\ 0 & 0 & \hat{c}^{N-2} & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & 0\\ 0 & 0 & 0 & 0 & 1\end{bmatrix} \tag{4.35}$$

Notice that, the Pascal matrix that is used for bandpass and bandstop transformation is still generated from eq. (4.8) but the upper limit is changed from N to $2N$.

4.4 Design Examples and Results

In the design, examples will show the use of the bilinear s-z with frequency transformation by Pascal matrix operation to transform from the normalized analog filter transfer function prototype to digital highpass filter and bandstop filter.

Let the normalized 2nd order Butterworth transfer function in s domain as follows,

$$H(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

For transformation to highpass digital filter by giving filter specifications as cut-off frequency 30 Hz and sampling frequency 250 Hz. Using eq. (4.16) can be found value k as follows,

$$k = \tan(\pi \times 30/250) = 0.39593$$

and use eqs. (4.19), (4.22), (4.23), (4.24) and (4.26) to compute the filter coefficients in z domain as follows,

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ -2 & 0 & 2 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.3959 & 0 \\ 0 & 0 & 0.1568 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

$$\therefore \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

and

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ -2 & 0 & 2 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.3959 & 0 \\ 0 & 0 & 0.1568 \end{bmatrix} \begin{bmatrix} 1 \\ \sqrt{2} \\ 1 \end{bmatrix}$$

$$\therefore \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 1.7166 \\ -1.6865 \\ 0.5968 \end{bmatrix}$$

Normally, the term b_0 must be equal to 1. Also, all coefficients must be scaled by b_0=1.7166. Therefore, the transfer function of highpass digital filter can be shown as

$$H(z) = \frac{0.5825 - 1.1651z^{-1} + 0.5825z^{-2}}{1 - 0.9825z^{-1} + 0.3477z^{-2}}$$

The amplitude response corresponds with this transfer function can be shown in Fig. 4.1

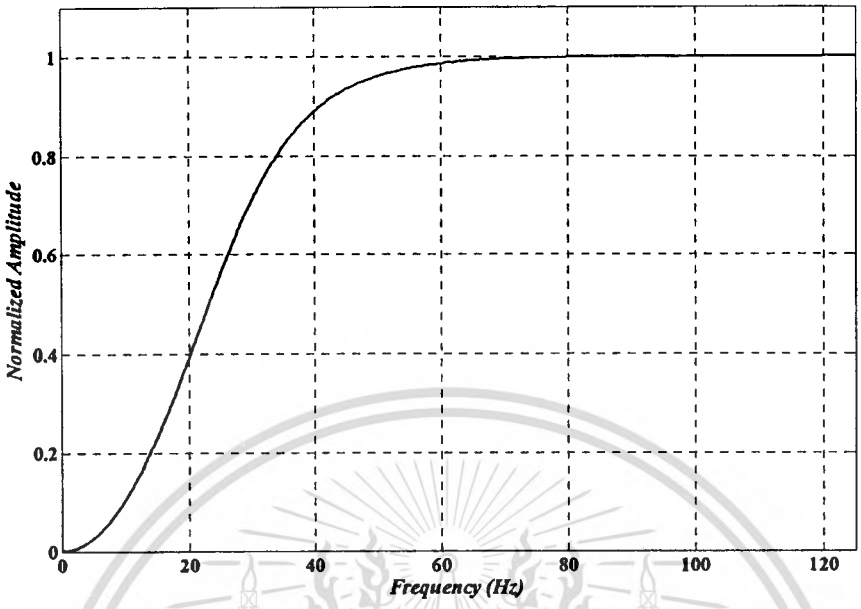


Fig. 4.1 Amplitude Response of Designed Highpass Digital Filter

For transformation to bandstop digital filter by giving filter specifications as lower cut-off frequency 30 Hz, upper cut-off frequency 50 Hz and sampling frequency 250 Hz. Using eq. (4.16) can be found value k_1 and k_2 as follows,

$$k_1 = \tan(\pi \times 30 / 250) = 0.39593$$

and

$$k_2 = \tan(\pi \times 50 / 250) = 0.72654$$

Also,

$$\hat{k} = k_1 k_2 = 0.28766 \quad \text{and} \quad \hat{c} = 1/k_2 - k_1 = 3.0247$$

and use eqs. (4.19), (4.20), (4.21), (4.24), (4.29), (4.32) and (4.33) to compute the filter coefficients in z domain as follows,

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 0 & -2 & -4 \\ 6 & 0 & -2 & 0 & 6 \\ 4 & -2 & 0 & 2 & -4 \\ 1 & -1 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0.0827 & 0 & 0 \\ 0 & 0.2877 & 0 \\ 0.5753 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 9.1486 & 0 & 0 \\ 0 & 3.0247 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\therefore \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} 15.1690 \\ -33.5664 \\ 48.9072 \\ -33.5664 \\ 15.1690 \end{bmatrix}$$

and

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 0 & -2 & -4 \\ 6 & 0 & -2 & 0 & 6 \\ 4 & -2 & 0 & 2 & -4 \\ 1 & -1 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 0.0827 & 0 & 0 \\ 0 & 0.2877 & 0 \\ 0.5753 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 9.1486 & 0 & 0 \\ 0 & 3.0247 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ \sqrt{2} \\ 1 \end{bmatrix}$$

$$\therefore \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 21.6762 \\ -39.6596 \\ 46.9072 \\ -27.4732 \\ 10.6619 \end{bmatrix}$$

Normally, the term b_0 must be equal to 1. Also, all coefficients must be scaled by b_0=21.6762.

Therefore, the transfer function of bandstop digital filter can be shown as

$$H(z) = \frac{0.6998 - 1.5485z^{-1} + 2.2563z^{-2} - 1.5485z^{-3} + 0.6998z^{-4}}{1 - 1.8296z^{-1} + 2.1640z^{-2} - 1.2674z^{-3} + 0.4919z^{-4}}$$

The amplitude response corresponds with this transfer function can be shown in Fig. 4.2

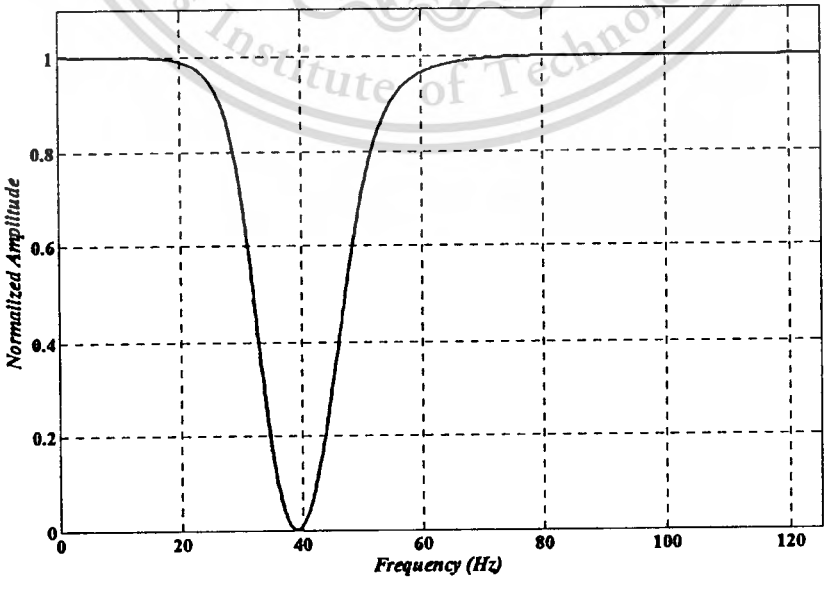


Fig. 4.2 Amplitude Response of Designed Bandstop Digital Filter

4.5 Conclusion

This chapter proposes an alternative method used for bilinear s-z transformation using Pascal matrix operations in cooperation with the frequency transformation. The proposed method has high efficiency in the case of high order filter design and is simpler than the directly substituting the relationship between s-z into the transfer function. The formulas used to generate each of matrices have been shown already. The design examples show the computation steps and the obtained transformation results.

Chapter 5

Modified Pascal Matrix for Biquad Digital Filter Design and Its Filter Structure Realization

5.1 Introduction

This chapter presents a new design method for designing a biquad digital filter which can give five frequency responses simultaneously, those are lowpass, highpass, bandpass, bandstop, and allpass filtering. The proposed method is based on Pascal matrix operation for bilinear s-z transformation, the Pascal matrix is used for transforming the normalized analog biquad transfer functions to digital biquad transfer functions. Moreover, the previous Pascal matrix will be modified to give only one matrix equation for proposed design method instead of six matrix equations in general case. Finally, the filter structure realization is proposed for biquad digital filter implementation, which corresponds to the proposed design matrix equation.

5.2 Pascal Matrix for Bilinear s-z Transformation

Transformation from *s*-domain to *z*-domain of analog transfer function to digital transfer function by using bilinear transform can replace the relationship between direct s-z into the equation in direct *s*-domain. Then we have to rearrange the equation to get the transfer function in *z*-domain. Generally, this method has some complexity since we have to rearrange and form the equation to obtain the digital coefficients. It will increase the computational complexity especially in a high order case. The method shown here same as described in 3 previous chapters and is presented in [8-17] which will enhance the bilinear *s-z* transform so that it can be easily computed by the matrix operation. The matrix used in bilinear transformation will be called *Pascal matrix for bilinear transformation*. The reason to call this matrix *Pascal matrix* is that the idea used to formulate this transformation matrix using the binomial theorem and the binomial coefficients are related to the Pascal's triangle. From the binomial theorem,

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k \tag{5.1}$$

where $\binom{n}{k} = \frac{n!}{(n-k)!k!}$ which is called binomial coefficients and can be shown in the form of Pascal's triangle as in Fig. 5.1

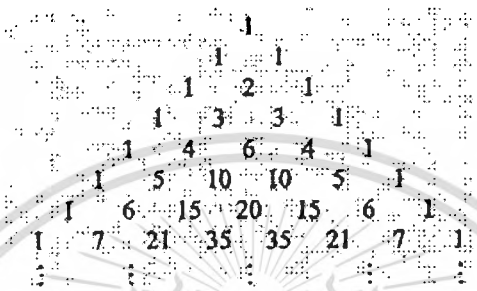


Fig. 5.1 Pascal's Triangle

Therefore, consider again for the linkage from the binomial theorem to Pascal matrix for bilinear transform creation. The relationship between s-z variable in bilinear transformation is

$$s = \frac{1 - z^{-1}}{1 + z^{-1}} \tag{5.2}$$

In addition, consider the N^{th} order analog and digital transfer function in eq. (5.3) and (5.4), respectively

$$H(s) = \frac{\sum_{i=0}^{N} A_i s^i}{\sum_{i=0}^{N} B_i s^i} \tag{5.3}$$

$$H(z) = \frac{\sum_{i=0}^{N} a_i z^{-i}}{\sum_{i=0}^{N} b_i z^{-i}} \tag{5.4}$$

Using eq. (5.3) and the relationship in eq. (5.2) would give

$$H(z)=\frac{\sum_{i=0}^{N}A_i\left[\frac{1-z^{-1}}{1+z^{-1}}\right]^i}{\sum_{i=0}^{N}B_i\left[\frac{1-z^{-1}}{1+z^{-1}}\right]^i}=\frac{\sum_{i=0}^{N}A_i(1-z^{-1})^i(1+z^{-1})^{N-i}}{\sum_{i=0}^{N}B_i(1-z^{-1})^i(1+z^{-1})^{N-i}} \tag{5.5}$$

From eq. (5.5), can see the digital transfer function in terms of analog coefficients. Use the binomial theorem to distribute and now consider only numerator part as below

$$\sum_{i=0}^{N}A_i(1-z^{-1})^i(1+z^{-1})^{N-i}=\sum_{i=0}^{N}A_i\left[\sum_{k=0}^{i}\binom{i}{k}(-1)^k z^{-k}\sum_{k=0}^{N-i}\binom{N-i}{k}z^{-k}\right] \tag{5.6}$$

After rearranging, change some new index and compare with eq. (5.4) to get digital coefficients. Also, from eq. (5.6) we would get eq. (5.7)

$$\sum_{i=0}^{N}A_i\left[\sum_{n=0}^{i}\binom{N-j}{n}\binom{j}{i-n}(-1)^{i-n}\right]z^{-i}=\sum_{i=0}^{N}a_i z^{-i} \tag{5.7}$$

The term in bracket will be formed into the matrix size $(N+1) \times (N+1)$ which is called Pascal matrix for bilinear transform. Therefore, the Pascal matrix $[P_{i,j}]$ can be defined as

$$P_{i,j}=\sum_{n=0}^{i}\binom{N-j}{n}\binom{j}{i-n}(-1)^{i-n} \quad ;i,j=0,1,\ldots,N \tag{5.8}$$

where i and j are row and column index, respectively.

Therefore, the column vector of digital coefficients can be computed by vector product of Pascal matrix with column vector of analog coefficients as follows,

$$\sum_{i=0}^{N}a_i z^{-i}=\sum_{i=0}^{N}A_i\left[P_{i,j}\right]z^{-i}$$

$$\therefore \quad [a_i]=[P_{i,j}][A_i] \tag{5.9}$$

The denominator part can be computed in the same manner. Not only the general form of Pascal matrix creation where each element can be determined from eq. (5.8), but also have the shortcut method or recurrence formula to generate each element in Pascal matrix using only the elements in the first row and column for other elements creation as shown in previous chapter.

5.3 Modified Pascal Matrix and Proposed Biquad Digital Filter Structure

The previous section helps us to understand how to create the Pascal matrix for bilinear transform. This method is an easy way to find the digital coefficients by using the analog coefficients from prototype transfer function multiplied by the Pascal matrix without replacing the relationship between *s*-*z* variable into analog prototype transfer function directly. Also, it can reduce the complexity in order to rearrange the equation to get the digital transfer function. In this chapter, we focus on using the principle of transform through Pascal matrix to design biquad digital filter. Therefore, we will start at the analog prototype transfer functions for biquad filter. The standard form in biqudratic equations for lowpass (LP), highpass (HP), bandpass (BP), bandstop (BS) and allpass (AP) filtering are shown in eqs. (5.10)-(5.14), respectively.

$$H_{LP}(s)=\frac{\Omega_0^2}{\Omega_0^2+\left(\Omega_0/Q\right)s+s^2} \tag{5.10}$$

$$H_{HP}(s)=\frac{s^2}{\Omega_0^2+\left(\Omega_0/Q\right)s+s^2} \tag{5.11}$$

$$H_{BP}(s)=\frac{\left(\Omega_0/Q\right)s}{\Omega_0^2+\left(\Omega_0/Q\right)s+s^2} \tag{5.12}$$

$$H_{BS}(s)=\frac{\Omega_0^2+s^2}{\Omega_0^2+\left(\Omega_0/Q\right)s+s^2} \tag{5.13}$$

$$H_{AP}(s)=\frac{\Omega_0^2-\left(\Omega_0/Q\right)s+s^2}{\Omega_0^2+\left(\Omega_0/Q\right)s+s^2} \tag{5.14}$$

If we use only Pascal matrix for bilinear transformation to obtain the digital coefficients, we have to use up to 6 matrix equations to transform analog coefficients to digital coefficients as 1 equation for a denominator term, which is common term for all equations from eqs. (5.10)-

(5.14), and 5 equations for each numerator term. The used Pascal matrix has the dimension 3×3 (for 2nd order transfer function).

For example, eq. (5.10) will be transformed from *s*-*z* by matrix equations as follows,

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 0 & -2 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \Omega_0^2 \\ 0 \\ 0 \end{bmatrix} \qquad \text{for numerator part}$$

and

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 0 & -2 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \Omega_0^2 \\ \Omega_0/Q \\ 1 \end{bmatrix} \qquad \text{for denominator part}$$

Therefore, from eq. (5.10) we would have

$$H_{LP}(z) = \frac{\Omega_0^2 + 2\Omega_0^2 z^{-1} + \Omega_0^2 z^{-2}}{(\Omega_0^2 + \Omega_0/Q + 1) + (2\Omega_0^2 - 2)z^{-1} + (\Omega_0^2 - \Omega_0/Q + 1)z^{-2}} \tag{5.15}$$

From eq. (5.15),

$$b_0 = \Omega_0^2 + \Omega_0/Q + 1, \quad b_1 = 2\Omega_0^2 - 2 \quad \text{and} \quad b_2 = \Omega_0^2 - \Omega_0/Q + 1$$

Practically, the value of b_0 must equal 1; also all coefficients in eq. (5.15) will be normalized by $(\Omega_0^2 + \Omega_0/Q + 1)$. Therefore, from eq. (5.15), the equation of digital lowpass transfer function can be shown as in eq. (5.16), and consider eq. (5.12) and (5.13) in the same manner to achieve digital transfer function as shown in eq. (5.17) and (5.18), respectively.

$$H_{LP}(z) = \frac{G_{LP}(1 + 2z^{-1} + z^{-2})}{1 + d_1 z^{-1} + d_2 z^{-2}} \tag{5.16}$$

$$H_{HP}(z) = \frac{G_{HP}(1 - 2z^{-1} + z^{-2})}{1 + d_1 z^{-1} + d_2 z^{-2}} \tag{5.17}$$

and

$$H_{BP}(z) = \frac{G_{BP}(1 - z^{-2})}{1 + d_1 z^{-1} + d_2 z^{-2}} \tag{5.18}$$

where

$$G_{LP} = \frac{1}{b_0}[\Omega_0^2], \qquad d_1 = \frac{b_1}{b_0}$$

$$G_{HP} = \frac{1}{b_0}, \qquad d_2 = \frac{b_2}{b_0}$$

$$G_{BP} = \frac{1}{b_0}\left[\frac{\Omega_0}{Q}\right]$$

In addition, we call G_{LP}, G_{HP} and G_{BP} "LP gain", "HP gain" and "BP gain", respectively.

The development to proposed design of biquad digital filter structure is considered by digital transfer function in eqs. (5.16)-(5.18). The denominator part of all equations is the same, which is a common term, as below

$$\frac{W(z)}{X(z)} = \frac{1}{1 + d_1 z^{-1} + d_2 z^{-2}} \tag{5.19}$$

$$\therefore \quad W(z)[1 + d_1 z^{-1} + d_2 z^{-2}] = X(z)$$

Take inverse z-transform yields

$$w(n) = x(n) - d_1 w(n-1) - d_2 w(n-2) \tag{5.20}$$

From eq.(5.20), the equation must be used likewise in every digital transfer function. In addition, the equation will become a common circuit which can be shared to all transfer functions and the structure of this circuit can be shown as in Fig.5.2

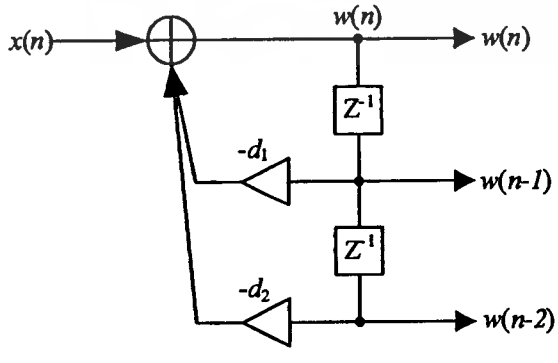


Fig. 5.2 Common Circuit Structure Realization

Thus, consider the numerator parts in eqs. (5.16)-(5.18) as follows,

For LP output;

Give $$\frac{Y_{LP}(z)}{W(z)} = G_{LP}(1 + 2z^{-1} + z^{-2})$$

$$\therefore \quad y_{LP}(n) = G_{LP}[w(n) + 2w(n-1) + w(n-2)] \tag{5.21}$$

For HP output;

Give $$\frac{Y_{HP}(z)}{W(z)} = G_{HP}(1 - 2z^{-1} + z^{-2})$$

$$\therefore \quad y_{HP}(n) = G_{HP}[w(n) - 2w(n-1) + w(n-2)] \tag{5.22}$$

For BP output;

Give $$\frac{Y_{BP}(z)}{W(z)} = G_{BP}(1 - z^{-2})$$

$$\therefore \quad y_{BP}(n) = G_{BP}[w(n) - w(n-2)] \tag{5.23}$$

From eqs. (5.21)-(5.23) we can see that the output of each transfer function is a function of signal $w(n)$, $w(n\text{-}1)$ and $w(n\text{-}2)$. Thus, the structure that is used for computing each output as LP, HP and BP output can be shown as follows,

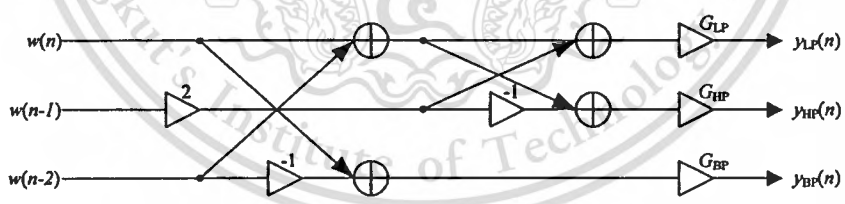


Fig. 5.3 Three Output Calculation Circuit Structure Realization

Moreover, we can evaluate the BS and AP output using analog prototype transfer function in eqs. (5.13)-(5.14), respectively. The digital transfer functions of both BS and AP filter can be shown as follows

For BS output;

$$H_{BS}(z) = H_{LP}(z) + H_{HP}(z)$$
$$= \frac{G_{LP}(1 + 2z^{-1} + z^{-2}) + G_{HP}(1 - 2z^{-1} + z^{-2})}{1 + d_1 z^{-1} + d_2 z^{-2}}$$

Like eqs. (5.16)-(5.18), we use the same denominator part. So, we would have

$$\frac{Y_{BS}(z)}{W(z)} = G_{LP}(1+2z^{-1}+z^{-2}) + G_{HP}(1-2z^{-1}+z^{-2})$$
$$\therefore \quad y_{BS}(n) = G_{LP}[w(n)+2w(n-1)+w(n-2)] + G_{HP}[w(n)-2w(n-1)+w(n-2)]$$

and we can notice from eq. (5.21) and (5.22), as below

$$\therefore \quad y_{BS}(n) = y_{LP}(n) + y_{HP}(n) \tag{5.24}$$

For AP output;

$$H_{AP}(z) = H_{LP}(z) + H_{HP}(z) - H_{BP}(z)$$
$$= H_{BS}(z) - H_{BP}(z)$$

Therefore, we can consider it in the same manner as the case of BS output and summarize it to

$$\frac{Y_{AP}(z)}{W(z)} = G_{LP}(1+2z^{-1}+z^{-2}) + G_{HP}(1-2z^{-1}+z^{-2}) - G_{BP}(1-z^{-2})$$
$$\therefore \quad y_{AP}(n) = G_{LP}[w(n)+2w(n-1)+w(n-2)] + G_{HP}[w(n)-2w(n-1)+w(n-2)] - G_{BP}[w(n)-w(n-2)]$$

and we can notice from eq. (5.23) and (5.24), as below

$$\therefore \quad y_{AP}(n) = y_{BS}(n) - y_{BP}(n) \tag{5.25}$$

Therefore, from eqs. (5.20)-(5.25), the new proposed biquad digital filter structure that gives five outputs simultaneously can be shown as in Fig. 5.4. The proposed structure is formulated from the direct II structure with some extended multipliers and adders in feed-forward path to give multiple outputs while the number of shift registers or delay units are the same as a single output version of ordinary 2nd order IIR filter. Five parameters, G_{LP}, G_{HP}, G_{BP}, d_1 and d_2, are

necessary for this filter structure. Therefore, the matrix equation for designing proposed biquad digital filter is

$$\begin{bmatrix} G_{LP} \\ G_{BP} \\ G_{HP} \\ d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 2 & 0 & -2 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} k\Omega_0^2 \\ k\Omega_0/Q \\ k \end{bmatrix} \tag{5.26}$$

where $k = \dfrac{1}{\Omega_0^2 + \Omega_0/Q + 1}$, Ω_0 = analog center frequency (rad/s), and Q = quality factor.

Consequently, we can use only 1 matrix equation as in eq. (5.26) for designing a biquad digital filter. The vector of design parameters (left hand side of eq. (5.26)) can be obtained from matrix transformation that transforms analog transfer function parameters (Ω_0, Q and k) to digital parameters corresponding to the new proposed filter structure in Fig. 5.4 This matrix transform will be called modified Pascal matrix for biquad digital filter design and, all of these, we will claim to be a new design suite for biquad digital filter.

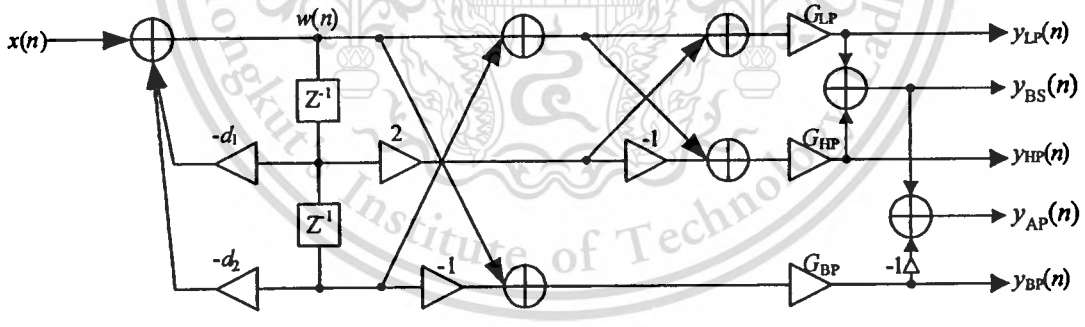


Fig. 5.4 The New Proposed Biquad Digital Filter Structure

5.4 Design Examples and Simulation Results

In design example of biquad digital filter, the filter specifications are as below,

$$f_0 = 2 \text{ kHz} \quad \text{and} \quad f_s = 10 \text{ kHz}$$

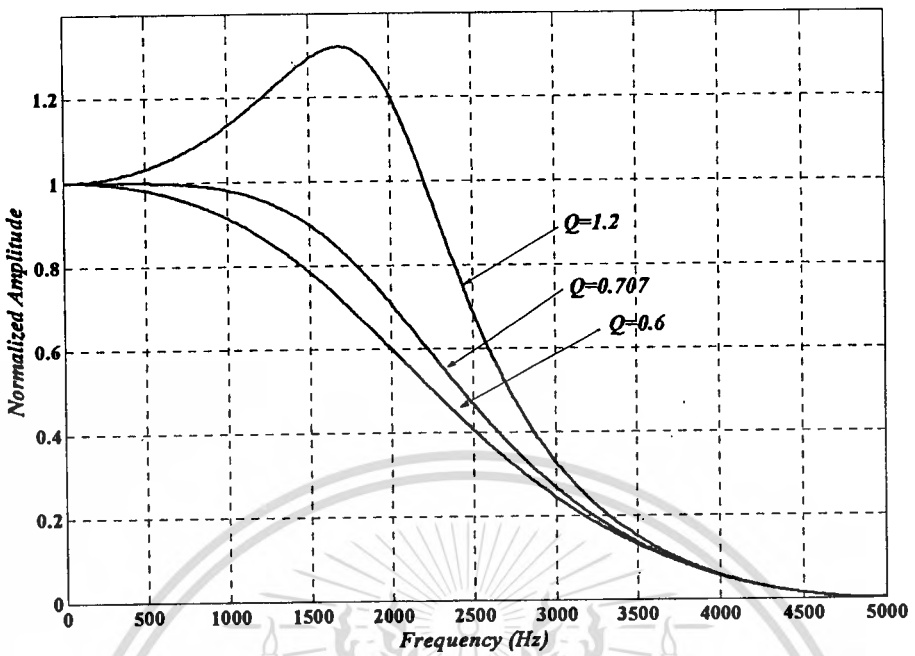
The analog frequency can be determined by the equation shown below

$$\Omega_0 = \tan\left(\frac{\pi f_0}{f_s}\right) = 0.7265 \; rad/s$$

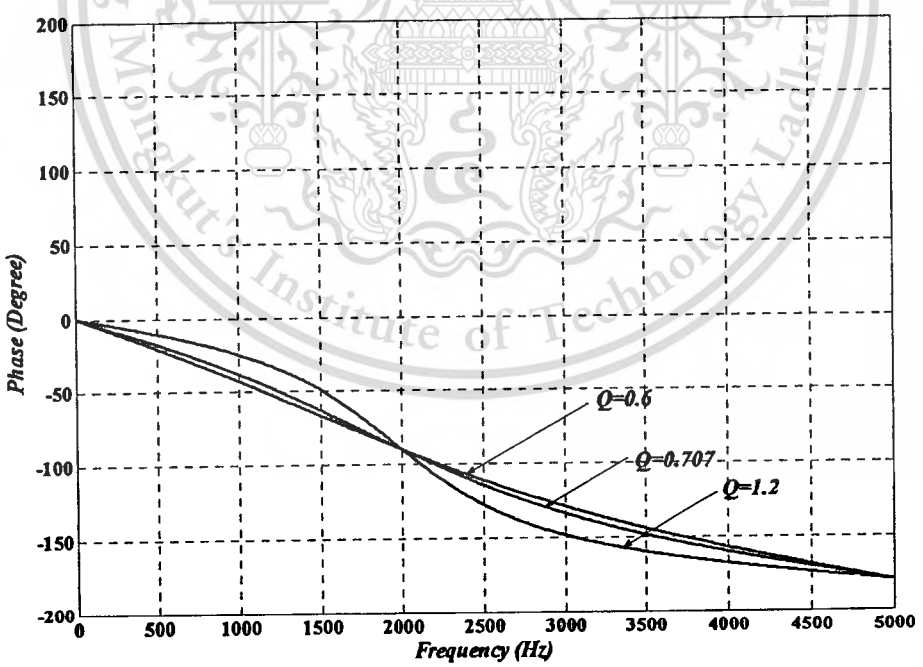
Next, determine 3 values of quality factor (Q) for this example as shown in Table 5.1 The summary of input parameters and the obtained output parameters, which are computed from eq. (5.26), is shown in Table 5.1. The simulation results shown in Fig. 5.5-5.9 are frequency response of five outputs respectively at the same time in 3 cases of quality factor.

Table 5.1 Input Parameters and Obtained Output Parameters for Design Examples

Analog Input Parameters			**Digital Output Parameters**				
Q	Ω_0	k	G_{LP}	G_{BP}	G_{HP}	d_1	d_2
0.6	0.7265rad/s	0.3651	0.1927	0.4421	0.3651	-0.3448	0.1157
0.707	0.7265rad/s	0.3913	0.2066	0.4021	0.3913	-0.3695	0.1957
1.2	0.7265rad/s	0.4688	0.2474	0.2838	0.4688	-0.4426	0.4324

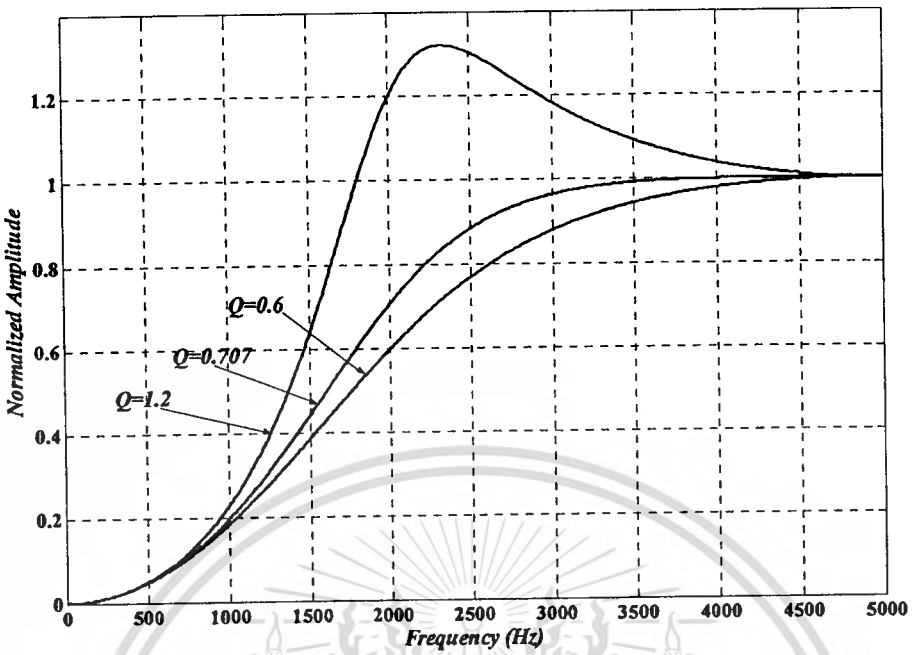


(a) Amplitude Response

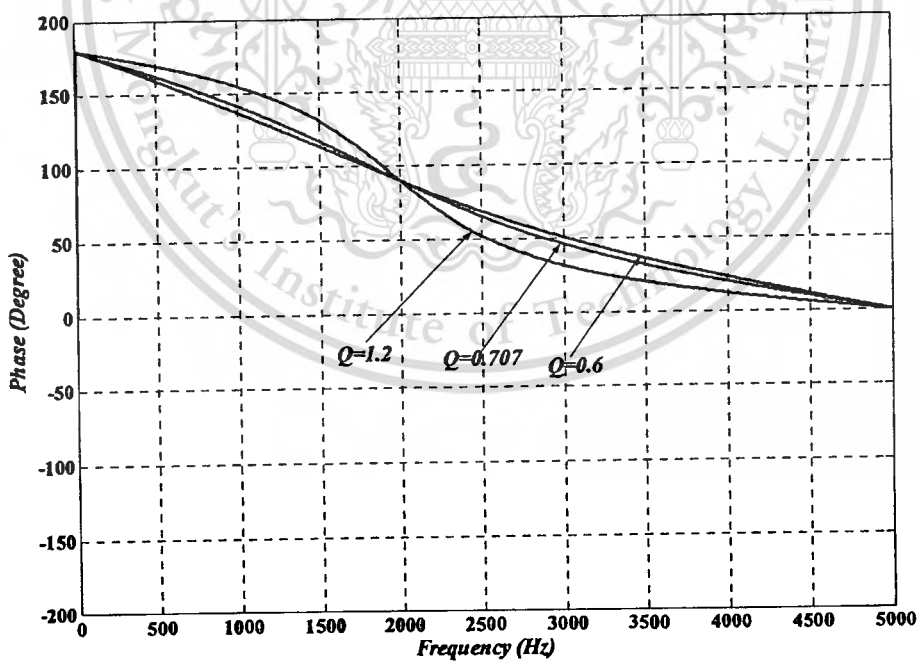


(b) Phase Response

Fig. 5.5 Frequency Response of Lowpass Output

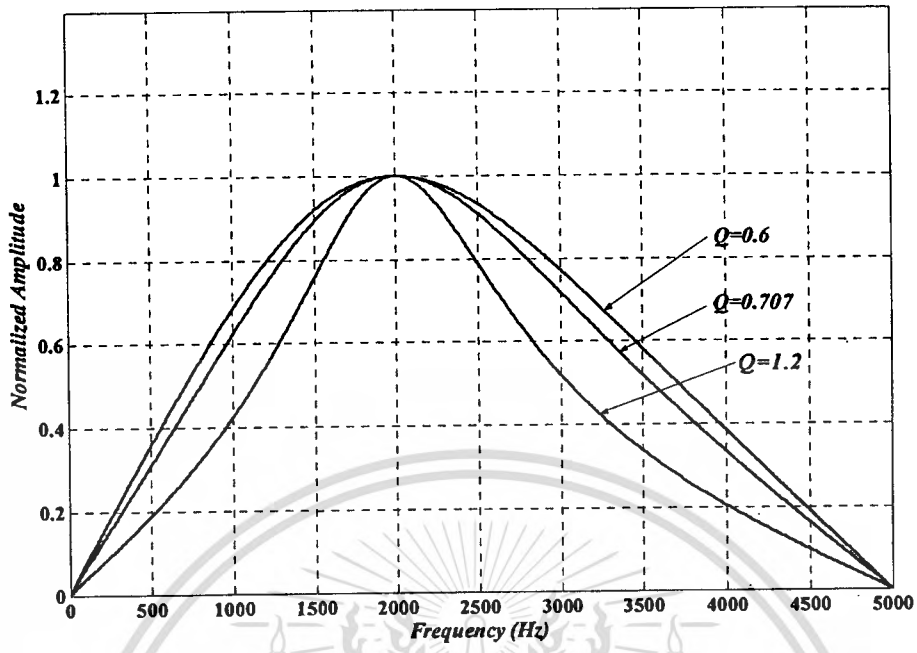


(a) Amplitude Response

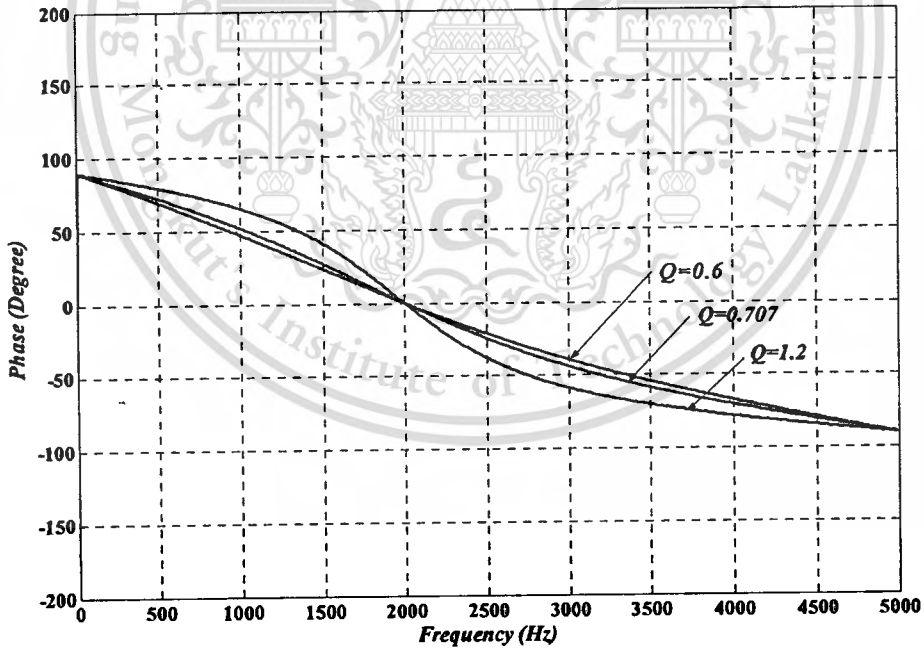


(b) Phase Response

Fig. 5.6 Frequency Response of Highpass Output

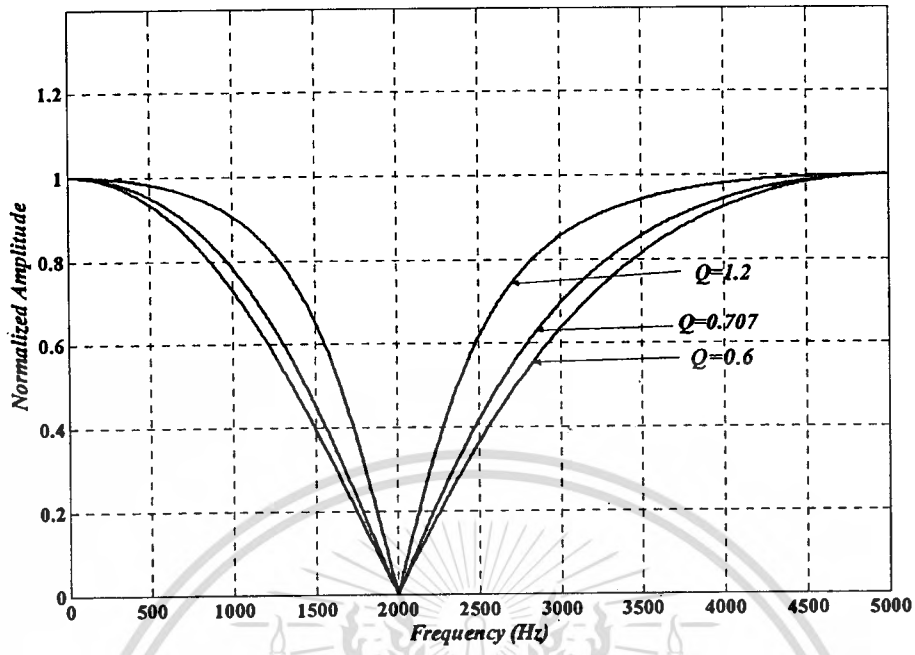


(a) Amplitude Response

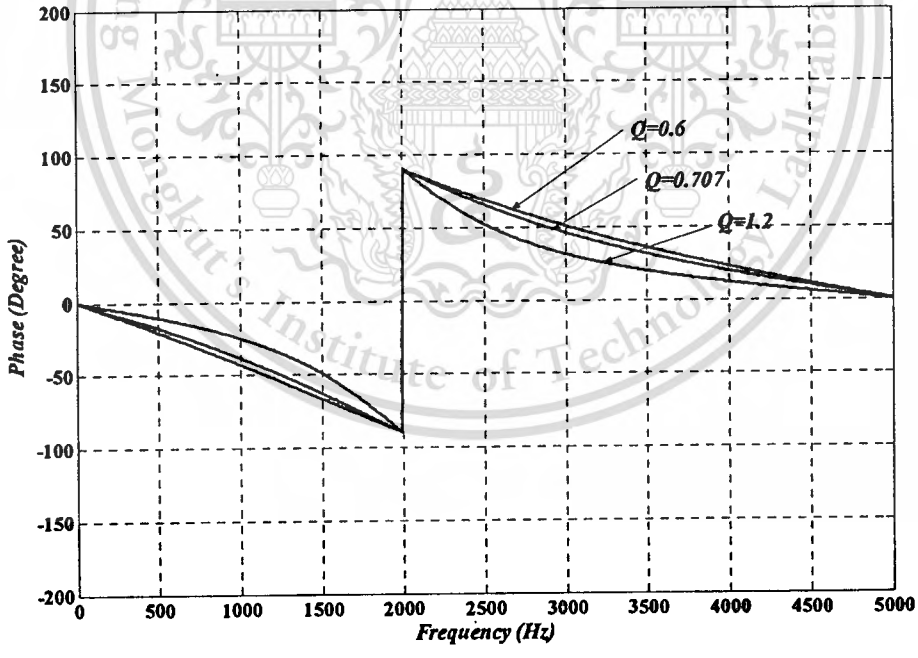


(b) Phase Response

Fig. 5.7 Frequency Response of Bandpass Output

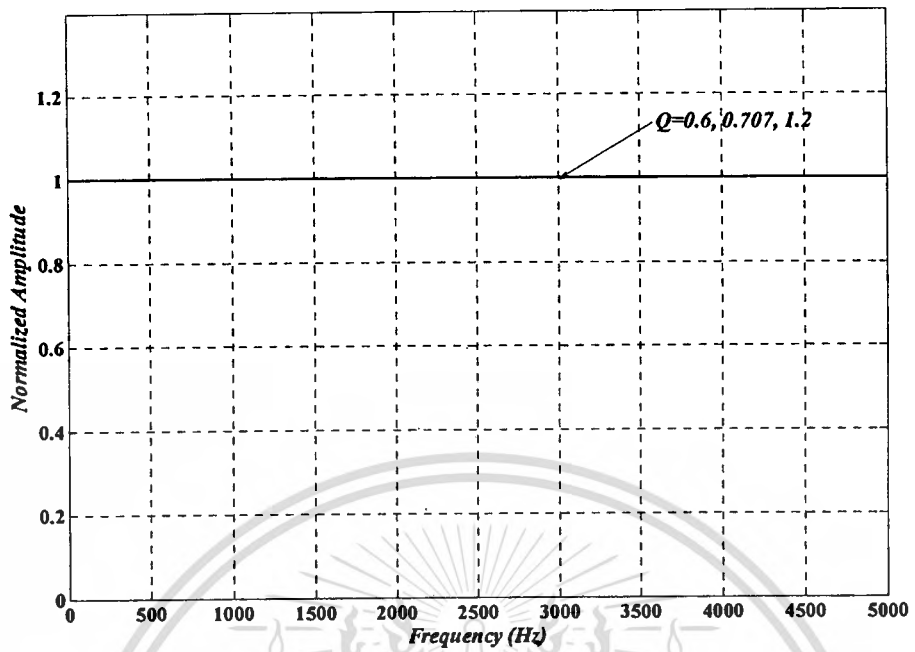


(a) Amplitude Response

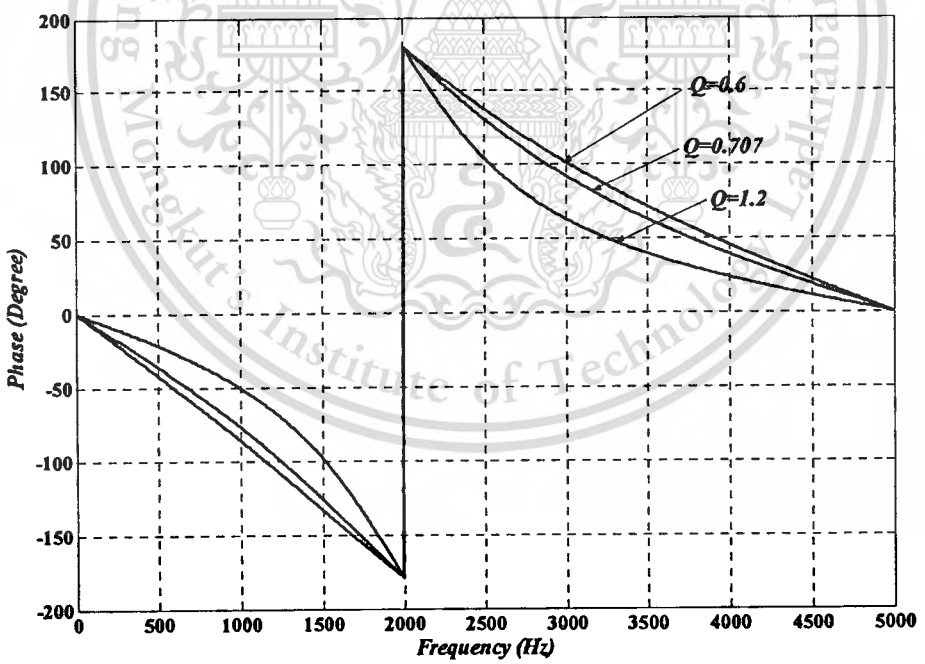


(b) Phase Response

Fig. 5.8 Frequency Response of Bandstop Output



(a) Amplitude Response



(b) Phase Response

Fig. 5.9 Frequency Response of Allpass Output

5.5 Conclusion

The new proposed method for designing a biquad digital filter based on modified Pascal matrix for bilinear *s-z* transformation is very easy since it needs only one matrix equation to compute the filter coefficients which correspond to the new proposed filter structure. Moreover, computing the denominator is the same as an ordinary 2^{nd} direct form II structure, and the number of delay units used is the same as a single output filter. With some modification in feed-forward path, this structure can give 5 outputs by an extension circuit compared to normal 2^{nd} order IIR filter that can give only 1 output.

Chapter 6

Discrete Pascal Transform and Its Hardware Realization

6.1 Introduction

This chapter presents the discrete Pascal transform for digital signal processing, the operations of discrete Pascal transform is based on Pascal transform matrix (this Pascal transform matrix is different from the Pascal matrix for s-z transformation that proposed in 4 previous chapters) which perform as the operator for signal transformation. The Pascal transform matrix used in the discrete Pascal transform which is proposed in this chapter is divided into 2 types. Generally, the matrix transformation is necessary to have many multipliers and adders, which is dependant on the dimension of the used matrix operator. The factorization of Pascal matrix into binary matrices will allow the transformation using Pascal matrix to operate without multipliers and only adders are used. Therefore, the hardware realization for transformation circuits can be efficiently designed by using the butterfly unit for discrete Pascal transform to establish the whole structure. Moreover, the hardware structure of two dimensional discrete Pascal transform will be proposed for a 2-D signal processing case

6.2 Basis Function of Discrete Pascal Transform

The discrete Pascal transform (DPT) **X** of the one-dimensional (1-D) signal vector **x** is defined as

$$\mathbf{X} = \mathbf{P}\mathbf{x} \tag{6.1}$$

where **P** is the Pascal transform matrix size $N \times N$, and **x** is input signal vector, **X** is transformed output signal vector size $N \times 1$. In this chapter, we divide the type of Pascal matrix into 2 types by basis function of the Pascal transform matrix. The first type, we call highpass type DPT which is the same as proposed in [23-26] and another type, we call lowpass type DPT. The reason for calling these type names was mentioned in [25] and in the next chapter.

A. Basis Function of Highpass Type DPT

From eq. (6.1), the Pascal matrix $\mathbf{P}$ in the case of highpass type DPT has the basis function as follows,

$$P_k^{(HP)}(x) = P^{(HP)}(x,k) = \frac{(-1)^k x^{(k)}}{k!} = (-1)^k \binom{x}{k} \qquad ;x,k = 0,1,2,...,N-1 \qquad (6.2)$$

B. Basis Function of Lowpass Type DPT

In the same manner as eq. (6.2), the basis function of lowpass type DPT can be shown as

$$P_k^{(LP)}(x) = P^{(LP)}(x,k) = \frac{x^{(k)}}{k!} = \binom{x}{k} \qquad ;x,k = 0,1,2,...,N-1 \qquad (6.3)$$

where N is dimension of Pascal transform matrix and

$$\binom{x}{k} = \frac{x!}{k!(x-k)!}$$

that we call binomial coefficient and is related to the Pascal's triangle. The only difference between eq. (6.2) and (6.3) is that the term $(-1)^k$ appears only in eq. (6.2). This term will make alternating the sign of the columns of highpass type Pascal transformation matrix. Both eq. (6.2) and eq. (6.3), $P(x,k)$ are the element in Pascal transformation matrix $\mathbf{P}$ as shown in eq. (6.1) depending on type needed, x^{th} is the row index and k^{th} is the column index.

The function $x^{(k)}$ is called falling factorial powers which can be shown as follows,

$$x^{(k)} = x(x-1)(x-2)...(x-k+2)(x-k+1) = \frac{x!}{(x-k)!}; k \geq 1 \qquad (6.4)$$

where $x^{(0)} = 1$ and k is also called the order of basis function.

For example, we can show the 3^{rd} order basis function as follows,

In the case of highpass type, it has 4 polynomials, these are

$$P_0^{(HP)}(x)=1$$
$$P_1^{(HP)}(x)=-x$$
$$P_2^{(HP)}(x)=\frac{1}{2}x(x-1)=-\frac{1}{2}(x-1)P_1^{(HP)}(x)$$
$$P_3^{(HP)}(x)=-\frac{1}{6}x(x-1)(x-2)=-\frac{1}{3}(x-2)P_2^{(HP)}(x)$$

and we can summarize to the recurrence formula as below

$$P_{k+1}^{(HP)}(x)=-\frac{1}{k+1}(x-k)P_k^{(HP)}(x) \tag{6.5}$$

In the case of lowpass type, it also has also 4 polynomials, these are

$$P_0^{(LP)}(x)=1$$
$$P_1^{(LP)}(x)=x$$
$$P_2^{(LP)}(x)=\frac{1}{2}x(x-1)=\frac{1}{2}(x-1)P_1^{(LP)}(x)$$
$$P_3^{(LP)}(x)=\frac{1}{6}x(x-1)(x-2)=\frac{1}{3}(x-2)P_2^{(LP)}(x)$$

So, the recurrence formula can be shown as,

$$P_{k+1}^{(LP)}(x)=\frac{1}{k+1}(x-k)P_k^{(LP)}(x) \tag{6.6}$$

Therefore, we can use these 4 polynomials that are obtained from each basis function to generate the element in Pascal transform matrix both of highpass and lowpass type as follows,

$$\mathbf{P}^{(HP)}=\left[P^{(HP)}(x,k)\right]=\begin{bmatrix}1&0&0&0\\1&-1&0&0\\1&-2&1&0\\1&-3&3&-1\end{bmatrix} \qquad \mathbf{P}^{(LP)}=\left[P^{(LP)}(x,k)\right]=\begin{bmatrix}1&0&0&0\\1&1&0&0\\1&2&1&0\\1&3&3&1\end{bmatrix}$$

The basic properties of these $\mathbf{P}$ *matrices are the following.*

1. The elements of the first column are equal to 1.
2. All matrices are lower triangular.
3. For highpass type $\mathbf{P}^{(HP)}$ matrix: the sum of the elements of each row (except the first row) is equal to zero.

   For lowpass type $\mathbf{P}^{(LP)}$ matrix: the sum of the elements of each row is equal to 2^x; x^{th} is the row index.
4. For highpass type $\mathbf{P}^{(HP)}$ matrix: the inverse $\left[\mathbf{P}^{(HP)}\right]^{-1}$ matrix is equal to the forward $\mathbf{P}^{(HP)}$ matrix, or $\left[\mathbf{P}^{(HP)}\right]^{-1} = \mathbf{P}^{(HP)}$.

   For lowpass type $\mathbf{P}^{(LP)}$ matrix: the inverse $\left[\mathbf{P}^{(LP)}\right]^{-1}$ matrix is equal to $(-1)^{x+k}$ multiply to the forward $\mathbf{P}^{(LP)}$ matrix, or $\left[\mathbf{P}^{(LP)}\right]^{-1} = (-1)^{x+k}\mathbf{P}^{(LP)}$.

6.3 Efficient Hardware Realization of DPT and Its Butterfly Unit

In [23] proposes only the basic of DPT in the form of matrix operation and does not focus on hardware realization. Meanwhile, [24] proposes the hardware implementation of DPT but the method used to decompose or factorize the Pascal transform matrix into binary (1,0,-1) matrices is not as compact as the one proposed in [26] which can give a very efficient hardware structure with all compact factorized binary (1,0,-1) matrices. This section will show the method to factorize the Pascal transform matrix of both highpass and lowpass type into binary (1,0,-1) matrices using elimination matrix while some details do not appear in [26].

6.3.1 Pascal Matrix Factorization to Binary (1,0,-1) Matrices

Consider matrix $\mathbf{P}^{(HP)}$ that has the dimension $N = 2$, which can be denoted to

$$\mathbf{P}_2^{(HP)} = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix}$$

and from equation of highpass type 2-point DPT we would get

$$\begin{bmatrix} X_0 \\ X_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$$

Therefore,

$$X_0 = x_0 \tag{6.7a}$$

$$X_1 = x_0 - x_1 \tag{6.7b}$$

from eq. (6.7a) and eq. (6.7b), we can make the data flow graph as below,

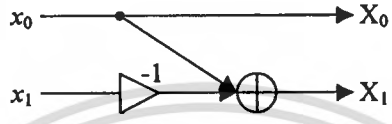


Fig. 6.1 Butterfly Unit of Highpass type DPT

The data flow graph that is obtained from eq. (6.7a) and (6.7b), we call a butterfly unit of highpass type DPT. Actually, we do not need the constant multiplier (-1) since we can design the subtractor instead of the adder. The reason to draw a data flow graph as shown in Fig.6.1 is that we want to distinguish the structure compared with a butterfly unit of lowpass type DPT and, in this chapter, both subtractor and adder will be classed the same.

Consider again to $\mathbf{P}^{(LP)}$ that has the dimension $N = 2$,

$$\mathbf{P}_2^{(LP)} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

and from equation of lowpass type 2-point DPT we would get

$$\begin{bmatrix} X_0 \\ X_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}$$

Therefore,

$$X_0 = x_0 \tag{6.8a}$$

$$X_1 = x_0 + x_1 \tag{6.8b}$$

from eq. (6.8a) and eq. (6.8b), we can make the data flow graph as below,

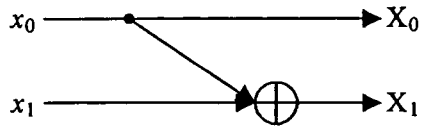


Fig. 6.2 Butterfly Unit of Lowpass type DPT

The data flow graph that is obtained from eq. (6.8a) and (6.8b), is also called a butterfly unit of lowpass type DPT. The butterfly unit shown in Fig.6.1 and Fig.6.2 will be the basic element that is used to establish any *N*-point DPT.

The method to factorize the Pascal transform matrix into binary (1,0,-1) matrices and to obtain the efficient hardware structure of any N-point DPT using Gaussian elimination [27] can be described as follows,

Lowpass type DPT

The lowpass type elimination matrix $\mathbf{E}^{(LP)}$ has entry $E_{x,x}^{(LP)} = 1$ and $E_{x,x-1}^{(LP)} = -1$ [27]. Assume $\mathbf{P}_4^{(LP)}$ for consideration,

First Step for matrix elimination:

$$\mathbf{E}_4^{(LP)}\mathbf{P}_4^{(LP)} = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & -1 & 1 & \\ & & -1 & 1 \end{bmatrix}\begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ 1 & 2 & 1 & \\ 1 & 3 & 3 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 2 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{P}_3^{(LP)} \end{bmatrix}$$

Second Step:

$$\begin{bmatrix} 1 & 0 \\ 0 & \mathbf{E}_3^{(LP)} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & \mathbf{P}_3^{(LP)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{P}_2^{(LP)} \end{bmatrix}$$

Third Step:

$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(LP)}\end{bmatrix}\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{P}_2^{(LP)}\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&-1&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&1&1\end{bmatrix}$$

$$=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}=[\mathbf{I}]$$

So that,

$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(LP)}\end{bmatrix}\begin{bmatrix}1&0\\0&\mathbf{E}_3^{(LP)}\end{bmatrix}\left[\mathbf{E}_4^{(LP)}\right]\left[\mathbf{P}_4^{(LP)}\right]=[\mathbf{I}] \tag{6.9}$$

$$\therefore\ \mathbf{P}_4^{(LP)}=\left[\mathbf{E}_4^{(LP)}\right]^{-1}\begin{bmatrix}1&0\\0&\mathbf{E}_3^{(LP)}\end{bmatrix}^{-1}\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(LP)}\end{bmatrix}^{-1} \tag{6.10}$$

Therefore, we can factorize the lowpass type Pascal transform matrix $\mathbf{P}_4^{(LP)}$ into binary (1,0) matrices as

$$\mathbf{P}_4^{(LP)}=\begin{bmatrix}1&&&\\1&1&&\\1&2&1&\\1&3&3&1\end{bmatrix}=\begin{bmatrix}1&&&\\1&1&&\\1&1&1&\\1&1&1&1\end{bmatrix}\begin{bmatrix}1&&&\\&1&&\\&1&1&\\&1&1&1\end{bmatrix}\begin{bmatrix}1&&&\\&1&&\\&&1&\\&&1&1\end{bmatrix}$$

This method of matrix decomposition or factorization is the same as the one used in [24] which cannot give the compact form of binary matrices. Using the property of inverse of lowpass type Pascal transform matrix as mentioned in section 6.2.

$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(LP)}\end{bmatrix}\begin{bmatrix}1&0\\0&\mathbf{E}_3^{(LP)}\end{bmatrix}\left[\mathbf{E}_4^{(LP)}\right]=(-1)^{x+k}\mathbf{P}_4^{(LP)} \tag{6.11}$$

we can show that

$$\mathbf{P}_4^{(LP)} = (-1)^{x+k}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{E}_2^{(LP)} \end{bmatrix} \times (-1)^{x+k}\begin{bmatrix} 1 & 0 \\ 0 & \mathbf{E}_3^{(LP)} \end{bmatrix} \times (-1)^{x+k}\left[\mathbf{E}_4^{(LP)}\right] \tag{6.12}$$

Finally, we can factorize $\mathbf{P}_4^{(LP)}$ into binary (1,0) matrices by this method as

$$\mathbf{P}_4^{(LP)} = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ 1 & 2 & 1 & \\ 1 & 3 & 3 & 1 \end{bmatrix} = \overset{Stage\,3}{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & 1 \end{bmatrix}} \overset{Stage\,2}{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \end{bmatrix}} \overset{Stage\,1}{\begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \end{bmatrix}} \tag{6.13}$$

and this is the compact form of binary matrices that is factorized from lowpass type Pascal transform matrix.

Highpass type DPT

The highpass elimination matrix $\mathbf{E}^{(HP)}$ has entry $E_{x,x}^{(HP)} = -1$ except $E_{0,0}^{(HP)} = 1$, and $E_{x,x-1}^{(HP)} = 1$. Assume $\mathbf{P}_4^{(HP)}$ for consideration,

First Step for matrix elimination:

$$\mathbf{E}_4^{(HP)}\mathbf{P}_4^{(HP)} = \begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix}\begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ 1 & -2 & 1 & \\ 1 & -3 & 3 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & -2 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 \\ 0 & \mathbf{P}_3^{(HP)} \end{bmatrix}$$

Second Step:

$$\begin{bmatrix} 1 & 0 \\ 0 & \mathbf{E}_3^{(HP)} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & \mathbf{P}_3^{(HP)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & -2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{P}_2^{(HP)} \end{bmatrix}$$

Third Step:

$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(HP)}\end{bmatrix}\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{P}_2^{(HP)}\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&1&-1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&1&-1\end{bmatrix}$$

$$=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}=[\mathbf{I}]$$

So that,

$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(HP)}\end{bmatrix}\begin{bmatrix}1&0\\0&\mathbf{E}_3^{(HP)}\end{bmatrix}\left[\mathbf{E}_4^{(HP)}\right]\left[\mathbf{P}_4^{(HP)}\right]=[\mathbf{I}] \tag{6.14}$$

$$\therefore\ \mathbf{P}_4^{(HP)}=\left[\mathbf{E}_4^{(HP)}\right]^{-1}\begin{bmatrix}1&0\\0&\mathbf{E}_3^{(HP)}\end{bmatrix}^{-1}\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(HP)}\end{bmatrix}^{-1}$$

Therefore, we can factorize the highpass type Pascal transform matrix $\mathbf{P}_4^{(HP)}$ into binary (1,0,-1) matrices as

$$\mathbf{P}_4^{(HP)}=\begin{bmatrix}1&&&\\1&-1&&\\1&-2&1&\\1&-3&3&-1\end{bmatrix}=\begin{bmatrix}1&&&\\1&-1&&\\1&-1&-1&\\1&-1&-1&-1\end{bmatrix}\begin{bmatrix}1&&&\\&1&&\\&1&-1&\\&1&-1&-1\end{bmatrix}\begin{bmatrix}1&&&\\&1&&\\&&1&\\&&1&-1\end{bmatrix}$$

Similar to the case of lowpass type, this form of binary matrices are not compact form. Using the property of inverse of highpass type Pascal transform matrix is equal to its matrix. Also, we can show that

$$\mathbf{P}_4^{(HP)}=\begin{bmatrix}1&0&0\\0&1&0\\0&0&\mathbf{E}_2^{(HP)}\end{bmatrix}\begin{bmatrix}1&0\\0&\mathbf{E}_3^{(HP)}\end{bmatrix}\left[\mathbf{E}_4^{(HP)}\right] \tag{6.15}$$

Finally, we can factorize $\mathbf{P}_4^{(HP)}$ into binary (1,0,-1) matrices as follows,

$$\mathbf{P}_{4*}^{(HP)} = \begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ 1 & -2 & 1 & \\ 1 & -3 & 3 & -1 \end{bmatrix} = \overset{Stage3}{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & -1 \end{bmatrix}} \overset{Stage2}{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix}} \overset{Stage1}{\begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix}} \tag{6.16}$$

and this is the compact form of binary matrices that is factorized from highpass type Pascal transformation matrix and is the same as binary (1,0,-1) matrices that is used in fast DPT [26]. The above method can be used to factorize any dimension N of both type of Pascal transform matrix into binary matrices with efficiency.

6.3.2 DPT Flow Graph

From previous section, we can factorize both of highpass and lowpass type Pascal transform matrix into binary matrices. The Pascal transform matrix in dimension $N = 4$ or $\mathbf{P}_4$ can be factorized to three binary matrices. We can summarize in general form as follows,

$$\begin{aligned} \mathbf{P} &= \prod_{l=N-1}^{1} [q_{xk}]_l \quad ; x,k = 0,1,2,\ldots,N-1 \\ &= [q_{xk}]_{N-1}[q_{xk}]_{N-2}\cdots[q_{xk}]_1 \end{aligned} \tag{6.17}$$

where $[q_{xk}]_l$ is binary matrix at stage l^{th}. Therefore, from eq. (6.13)

$$\begin{aligned} \mathbf{P}_4^{(LP)} &= [q_{xk}^{(LP)}]_3[q_{xk}^{(LP)}]_2[q_{xk}^{(LP)}]_1 \\ &= \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \end{bmatrix} \end{aligned} \tag{6.18}$$

and from eq. (6.16)

$$\mathbf{P}_4^{(HP)} = [q_{xk}^{(HP)}]_3[q_{xk}^{(HP)}]_2[q_{xk}^{(HP)}]_1$$

$$= \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix} \tag{6.19}$$

To simplify the method to factorize the Pascal transform matrix both lowpass and highpass type into binary matrices, we propose steps to create each binary matrix as follows,

For Lowpass type:

1. The main diagonal entries are '1' for all stage.
2. All entries in upper triangle are equal to '0'.

3. At stage l^{th}, consider for the rowth $x = N-1, N-2, ..., l$ by

$$\boxed{\text{for each } x \Rightarrow \textit{if } q_{x,x} = \text{'1'} \text{ then } q_{x,x-1} = \text{'1'}}$$

4. Otherwise are '0'.

For Highpass type:

1. Consider in the main diagonal,

At stage l^{th}, consider for the rowth $x = N-1, N-2, ..., l$ by

$$\boxed{q_{x,x} = \text{'}-1\text{', others are '1'}}$$

or consider in this manner

for stage $l = N-1$

$Diag([q_{xk}]_{N-1}) = [1\ 1\ 1 \ldots 1\ -1];$ '1' have N-1 elements, others are '-1'

$\vdots$

for stage $l = 2$

$Diag([q_{xk}]_2) = [1\ 1\ -1\ -1 \ldots -1];$ '1' have 2 elements, others are '-1'

for stage $l = 1$

$Diag([q_{xk}]_1) = [1\ -1\ -1\ -1 \ldots -1];$ '1' has 1 element, others are '-1'

2. All entries in upper triangle are '0'.

3. At stage l^{th}, consider in each row,

$$\boxed{\text{if } q_{x,x} = '-1' \text{ then } q_{x,x-1} = '1'}$$

4. Otherwise are '0'

The data flow graph showing hardware realization of DPT can be created from factorized binary matrices and its structure consists of many butterfly units which are the basic computational element in DPT.

The lowpass type 4-point DPT can be computed by

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ 1 & 2 & 1 & \\ 1 & 3 & 3 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \Rightarrow \begin{aligned} X_0 &= x_0 \\ X_1 &= x_0 + x_1 \\ X_2 &= x_0 + 2x_1 + x_2 \\ X_3 &= x_0 + 3x_1 + 3x_2 + x_3 \end{aligned} \tag{6.20}$$

$$= \underbrace{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & 1 \end{bmatrix}}_{Stage3} \underbrace{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \end{bmatrix}}_{Stage2} \underbrace{\begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & 1 & 1 & \\ & & 1 & 1 \end{bmatrix}}_{Stage1} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Therefore, the lowpass type 4-point DPT flow graph can be shown in Fig. 6.3

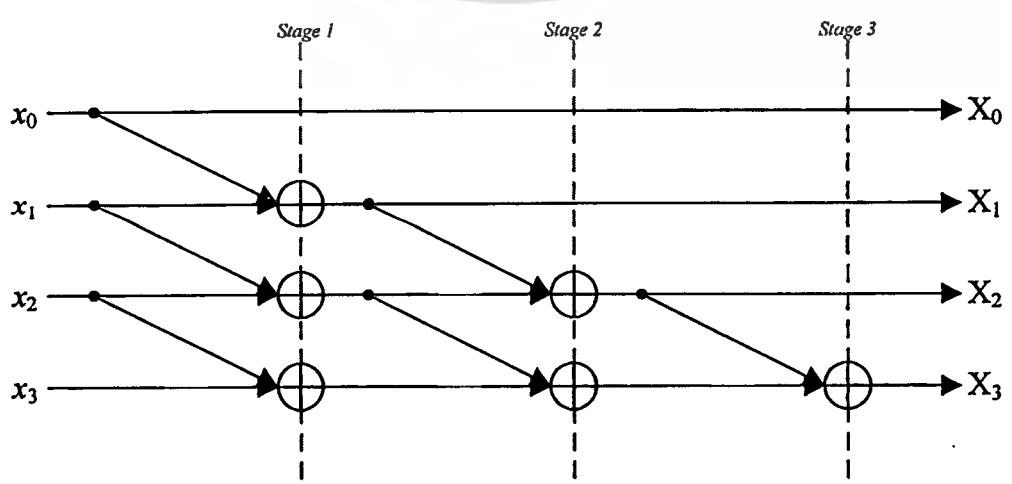


Fig. 6.3 The Lowpass type 4-point DPT flow graph

We use 6 lowpass type butterfly units for realizing lowpass type 4-point DPT. This structure is suitable for hardware implementation and very easy to implement. Especially, in VLSI design this structure is very compatible to the pipeline technique by placing the pipeline register on each stage as shown in Fig.6.3 to enhance the speed of computation of DPT.

The highpass type 4-point DPT can be computed by

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ 1 & -2 & 1 & \\ 1 & -3 & 3 & -1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \Rightarrow \begin{aligned} X_0 &= x_0 \\ X_1 &= x_0 - x_1 \\ X_2 &= x_0 - 2x_1 + x_2 \\ X_3 &= x_0 - 3x_1 + 3x_2 - x_3 \end{aligned} \tag{6.21}$$

$$= \underbrace{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & -1 \end{bmatrix}}_{Stage3} \underbrace{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & -1 \\ & & 1 & -1 \end{bmatrix}}_{Stage2} \underbrace{\begin{bmatrix} 1 & & & \\ 1 & -1 & & \\ & & 1 & -1 \\ & & 1 & -1 \end{bmatrix}}_{Stage1} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Therefore, the highpass type 4-point DPT flow graph can be shown in Fig. 6.4

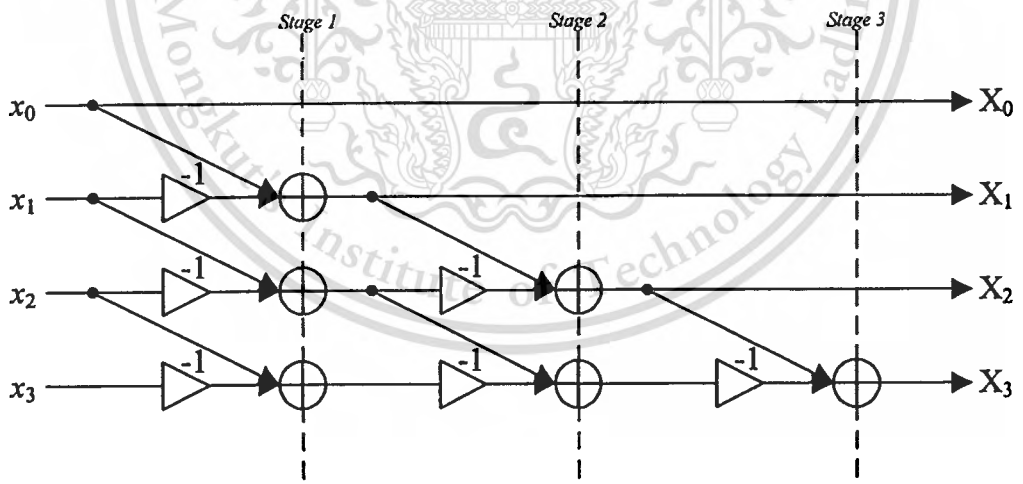


Fig. 6.4 The Highpass type 4-point DPT flow graph

This type of DPT also uses 6 highpass type butterfly units for realization.

6.3.3 Computational Complexity of DPT

If we consider the number of operation of direct DPT from definition matrix multiplication in eq. (6.1), the number of multiplications (M_N) and the number of additions (or subtractions in the case of highpass type) (A_N) will be

$$M_N = N^2 \qquad \text{and} \qquad A_N = (N-1)N$$

However, the Pascal transform matrix is a lower triangular matrix where all entries in the first column are '1' and the entries in main diagonal are '1' or '-1'. Also, no multiplications are needed for operating those entries. Then, the number of multiplications and number of additions would be

$$M_N = 1 + \frac{N(N-3)}{2} \qquad \text{and} \qquad A_N = \frac{(N-1)N}{2}$$

Finally, from DPT flow graph (butterfly based) which is based on Pascal matrix factorization into binary matrices can compute the transformed results without multiplications. Thus,

$$M_N = 0 \qquad \text{and} \qquad A_N = \frac{(N-1)N}{2}$$

The summary of computational complexity of any *N*-point DPT can be shown as in Table 6.1.

Table 6.1 Computational Complexity of *N*-point DPT

	Direct Matrix Multiplication	**Pascal Matrix Multiplication**	**Butterfly Based DPT**
M_N	N^2	$1+\frac{N(N-3)}{2}$	0
A_N	$(N-1)N$	$\frac{(N-1)N}{2}$	$\frac{(N-1)N}{2}$

6.4 Two-Dimensional DPT and Its Improvement to DPF structure

The two-dimension (2-D) DPT can be computed by [23-25]

$$\mathbf{X} = \mathbf{P}\mathbf{x}\mathbf{P}^{\mathrm{T}} \tag{6.22}$$

where $\mathbf{P}$ is the Pascal transform matrix and its transpose $\mathbf{P}^{\mathrm{T}}$, $\mathbf{X}$ is transformed output of an input image matrix $\mathbf{x}$, all of matrix size $N \times N$.

for example, we will consider highpass type 2-D DPT with $N = 3$

$$\mathbf{X} = \left[\mathbf{P}^{(\mathrm{HP})}\right]\left[\mathbf{x}\right]\left[\mathbf{P}^{(\mathrm{HP})}\right]^{\mathrm{T}} = \begin{bmatrix} 1 & & \\ 1 & -1 & \\ 1 & -2 & 1 \end{bmatrix}\begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{0,0} & x_{0,1} & x_{0,2} \\ x_{0,0} & x_{0,1} & x_{0,2} \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 \\ & -1 & -2 \\ & & 1 \end{bmatrix} \tag{6.23}$$

$$\therefore \begin{bmatrix} X_{0,0} & X_{0,1} & X_{0,2} \\ X_{1,0} & X_{1,1} & X_{1,2} \\ X_{2,0} & X_{2,1} & X_{2,2} \end{bmatrix} = \underbrace{\begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{0,0}-x_{1,0} & x_{0,1}-x_{1,1} & x_{0,2}-x_{1,2} \\ x_{0,0}-2x_{1,0}+x_{2,0} & x_{0,1}-2x_{1,1}+x_{2,1} & x_{0,2}-2x_{1,2}+x_{2,2} \end{bmatrix}}_{\text{This Matrix is a result of column operation for each input column vector, and assume to be } [y_{a,b}]} \times \begin{bmatrix} 1 & 1 & 1 \\ & -1 & -2 \\ & & 1 \end{bmatrix}$$

Thus,

$$\begin{bmatrix} X_{0,0} & X_{0,1} & X_{0,2} \\ X_{1,0} & X_{1,1} & X_{1,2} \\ X_{2,0} & X_{2,1} & X_{2,2} \end{bmatrix} = \begin{bmatrix} y_{0,0} & y_{0,1} & y_{0,2} \\ y_{1,0} & y_{1,1} & y_{1,2} \\ y_{2,0} & y_{2,1} & y_{2,2} \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 \\ & -1 & -2 \\ & & 1 \end{bmatrix} \tag{6.24}$$

$$= \underbrace{\begin{bmatrix} y_{0,0} & y_{0,0}-y_{0,1} & y_{0,0}-2y_{0,1}+y_{0,2} \\ y_{1,0} & y_{1,0}-y_{1,1} & y_{1,0}-2y_{1,1}+y_{1,2} \\ y_{2,0} & y_{2,0}-y_{2,1} & y_{2,0}-2y_{2,1}+y_{2,2} \end{bmatrix}}_{\text{This matrix is a result of row operation for each input row vector which is passed from column operation. Therefore, total results mean 2-D operation (2-D DPT)}}$$

Finally,

$$\begin{bmatrix} X_{0,0} & X_{0,1} & X_{0,2} \\ X_{1,0} & X_{1,1} & X_{1,2} \\ X_{2,0} & X_{2,1} & X_{2,2} \end{bmatrix} = \begin{bmatrix} x_{0,0} & x_{0,0}-x_{0,1} & x_{0,0}-2x_{0,1}+x_{0,2} \\ [x_{0,0}-x_{1,0}] & [x_{0,0}-x_{1,0}]-[x_{0,1}-x_{1,1}] & [x_{0,0}-x_{1,0}]-2[x_{0,1}-x_{1,1}]+[x_{0,2}-x_{1,2}] \\ [x_{0,0}-2x_{1,0}+x_{2,0}] & [x_{0,0}-2x_{1,0}+x_{2,0}]-[x_{0,1}-2x_{1,1}+x_{2,1}] & [x_{0,0}-2x_{1,0}+x_{2,0}]-2[x_{0,1}-2x_{1,1}+x_{2,1}]+[x_{0,2}-2x_{1,2}+x_{2,2}] \end{bmatrix} \tag{6.25}$$

We can summarize 2-D DPT flow graph with $N = 3$ as in Fig. 6.5

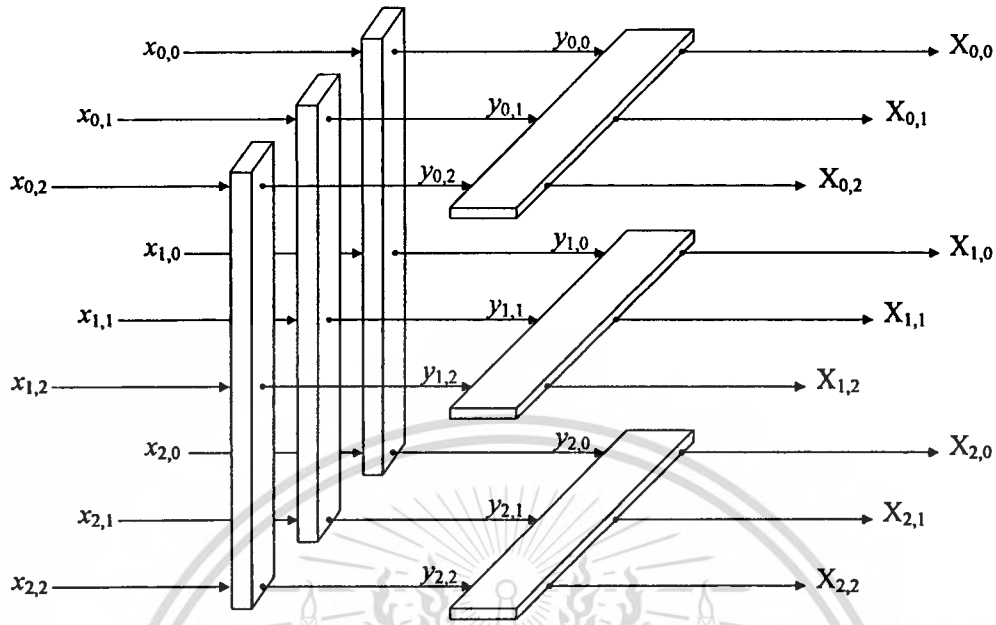
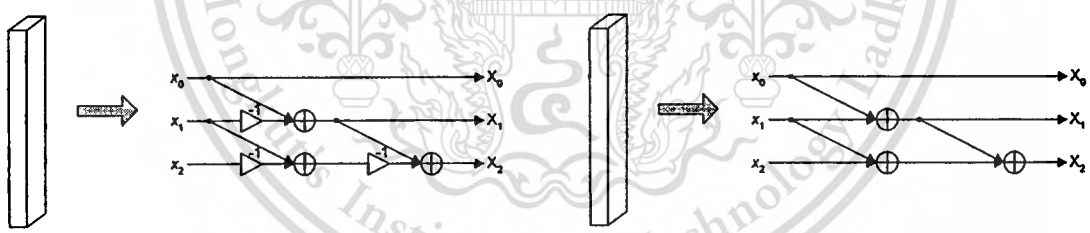


Fig. 6.5 The 2-D DPT flow graph

where the processing elements of highpass (left) and lowpass (right) type are as below, respectively



The DPF (Discrete Pascal Filter), which is mentioned in [25], can be formulated from the DPT by making the relation between the inputs and focus only on 1 output. Using DPF, the computational operation will be changed from matrix multiplication in DPT to convolution in DPF. With a little modification, we can obtain the hardware realization of 1-D DPF of both highpass type and lowpass type as follows

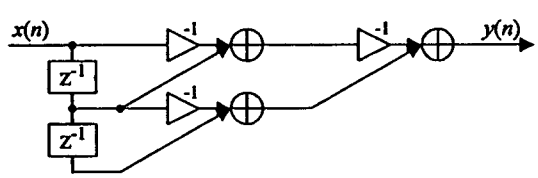


Fig. 6.6 The 2nd order Highpass type 1-D DPF

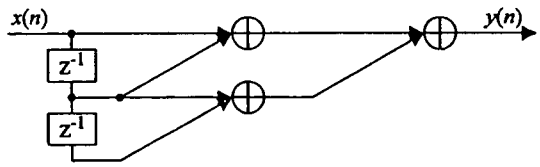


Fig. 6.7 The 2nd order Lowpass type 1-D DPF

The order of DPF is equal to the dimension of Pascal transform matrix minus 1, ($N-1$). 2-D DPF which is based on 2-D DPT flow graph in Fig.6.5, can be modified as shown in Fig. 6.8.

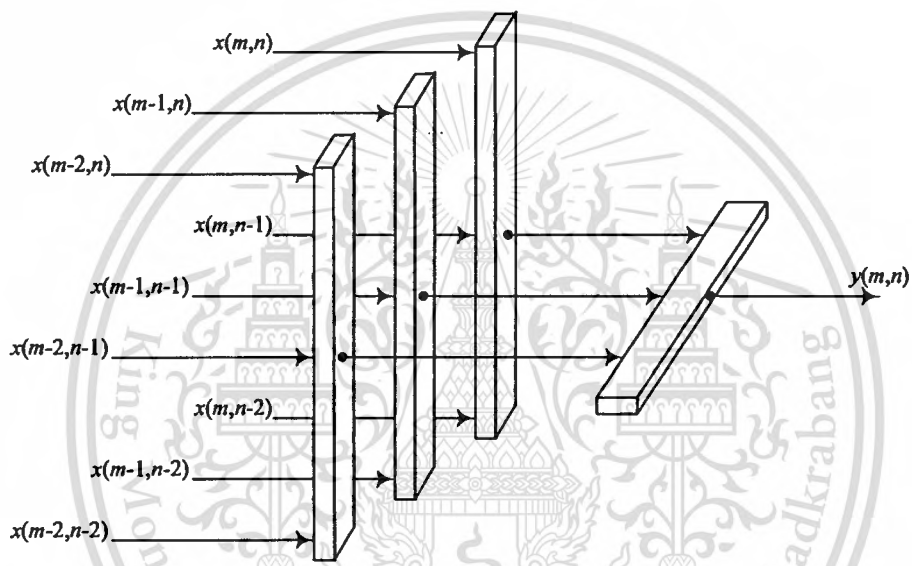
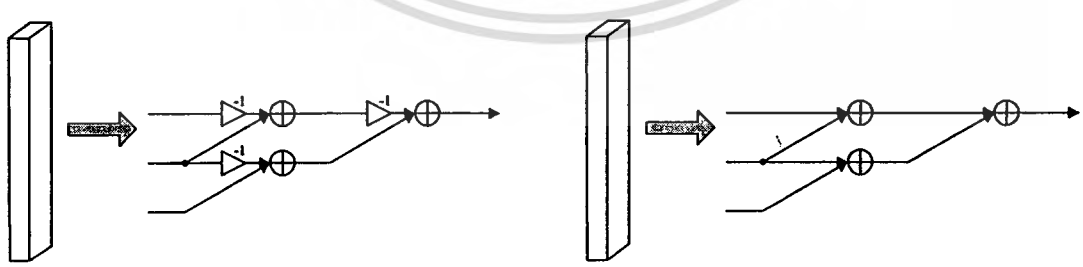


Fig. 6.8 The 2-D DPF with Convolution Mask size 3×3

The processing element of highpass (left) and lowpass (right) type are as below, respectively



Practically, the 2-D DPF in Fig. 6.8 used for processing image has to work along with input image arrangement circuit as shown in Fig. 6.9.

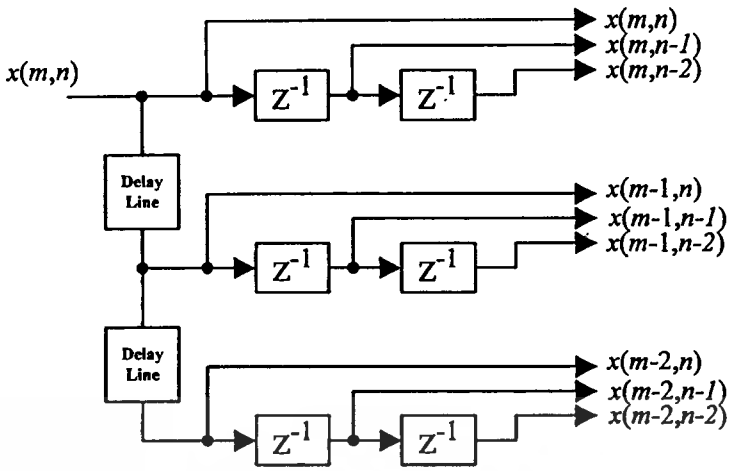


Fig. 6.9 The Input Image Arrangement Circuit

From all above described in sections 6.3 and 6.4 can be applied to higher order or higher dimension DPT or DPF applications.

6.5 Conclusion

The DPT hardware realization of both 1-D and 2-D and both highpass type and lowpass type is proposed. The hardware realization using Pascal matrix factorization or decomposition into binary (1,0-1) matrices can give the efficient structure based on butterfly unit for DPT. The obtained DPT flow graph can be used for transformation without multiplications, only additions are needed. Moreover, the DPF which is improved from DPT is also proposed.

Chapter 7

The Discrete Pascal Filter (DPF) from Frequency Characteristic in DPT

7.1 Introduction

This chapter presents the investigation of frequency or filtering characteristic that is hidden in the discrete Pascal transform. The Pascal matrix that is used for signal transformation by discrete Pascal transform in this chapter is divided into 2 types, the first type has hidden high-pass filtering characteristic and another type has hidden low-pass filtering characteristic. This chapter will show the method to investigate those characteristics by the transfer function of discrete Pascal transform systems formulated and used to analyze the frequency characteristic of such systems. Moreover, the relationship between the dimension of Pascal matrix and frequency characteristic will be shown. Finally, the result from the investigation will be applied for both 1-D and 2-D signal filtering applications.

7.2 The Hidden Frequency Characteristic in DPT and Its Discrete Pascal Filter

The discrete Pascal transform (DPT) X of the one-dimensional (1-D) signal vector x is defined as

$$X = \mathbf{P}x \tag{7.1}$$

where $\mathbf{P}$ is the Pascal transform matrix size $N \times N$, and x is input signal vector, X is a transformed output signal vector size $N \times 1$. In this chapter, we divide the type of Pascal matrix into 2 types by the basis function of the Pascal transform matrix. The first type, we call highpass type DPT which is the same as the one proposed in [23] and another type, we call lowpass type DPT. We will explain the reason for using these names in this section.

The two-dimensional (2-D) DPT can find the transformed output in the following [23]

$$\mathbf{X} = \mathbf{P}\mathbf{x}\mathbf{P}^{\mathbf{T}} \tag{7.2}$$

where $\mathbf{P}$ is the Pascal transform matrix in $N \times N$ dimension, and its transpose is $\mathbf{P}^T$, $\mathbf{X}$ is a transformed output of an input image matrix $\mathbf{x}$.

In [23-24] does not explain the practical usage, in which normally the size of input image must be bigger than the size or dimension of the matrix operator. Only the same size of input image and Pascal matrix operator is shown with small size of assumed input image. In the real applications, we use normal input image size such as 256×256 pixels. So, it is impossible to use the same size for Pascal matrix operator. The block processing is necessary for this case the same as when we use the DCT (Discrete Cosine Transform) matrix to transform the input image to the frequency domain. The DCT coefficients can be obtained by using DCT matrix size 8×8 for transforming image. Similar to the DCT, we can use the smaller size of Pascal transform matrix to operate with the real input image and the DPT coefficients can be obtained. Therefore, it is not a problem to apply the smaller size of Pascal transform matrix to operate with the bigger size of input image, but the obtained coefficients do not give a satisfactory transformed output image, and we cannot see the edge of an image in this way. If we focus on the effect that happens on the output image, the direct DPT cannot give good results and cannot give the results we expected at the edge or high frequency components of an image. We will show the transformed output of the direct DPT on both 1-D signal and 2-D signal or image in section 7.3.

Since the DPT has the Pascal transform matrix performing as the key operator, so if we consider such matrix as the system and the input of this system would be input vector or input matrix $\mathbf{x}$ depending on weather the case is 1-D or 2-D signal, and the output of this system would be the transformed output $\mathbf{X}$. Therefore, the system function or transform function of this system must be formulated for investigating the characteristic, especially the frequency or filtering characteristic of both the highpass type and lowpass type Pascal transform matrix and both 1-D and 2-D signal case.

7.2.1 Frequency Response of Highpass Type DPT

Consider the 1-D DPT with $N = 3$, from eq. (7.1) we would get

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & -1 & 0 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \tag{7.3}$$

In order to formulate the transfer function, it is necessary to make the relationship for each input data by giving

$x_2 = x(n)$; *present data*
$x_1 = x(n-1)$; 1 *sample delayed data*
$x_0 = x(n-2)$; 2 *samples delay data*

Also, from eq. (7.3) we would get

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & -1 & 0 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} x(n-2) \\ x(n-1) \\ x(n) \end{bmatrix} \tag{7.4}$$

Thus, we can summarize

$$X_0 = x(n-2) \tag{7.5a}$$
$$X_1 = -x(n-1) + x(n-2) \tag{7.5b}$$
$$X_2 = x(n) - 2x(n-1) + x(n-2) \tag{7.5c}$$

and we can show a system diagram to represent the highpass type DPT as below

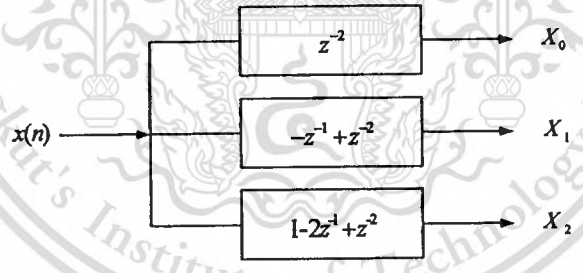


Fig. 7.1 Block Diagram of Highpass type DPT System

From Fig. 7.1, we can see that this system has 3 outputs. But since we assume that input x_2 is the present input data $x(n)$, so the output X_2 should be the present output $y(n)$. Then from eq. (7.5c) we would get

$$y(n) = x(n) - 2x(n-1) + x(n-2) \tag{7.6}$$

Take the z-transform, the transfer function of highpass type DPT can be shown as

$$H^{(HP)}(z) = 1 - 2z^{-1} + z^{-2} \tag{7.7}$$

We can notice that the transfer function coefficients comes from the last row of highpass type Pascal transform matrix in eq. (7.3). From the obtained transfer function as shown in eq. (7.7), we can investigate the frequency characteristic as shown in Fig. 7.2.

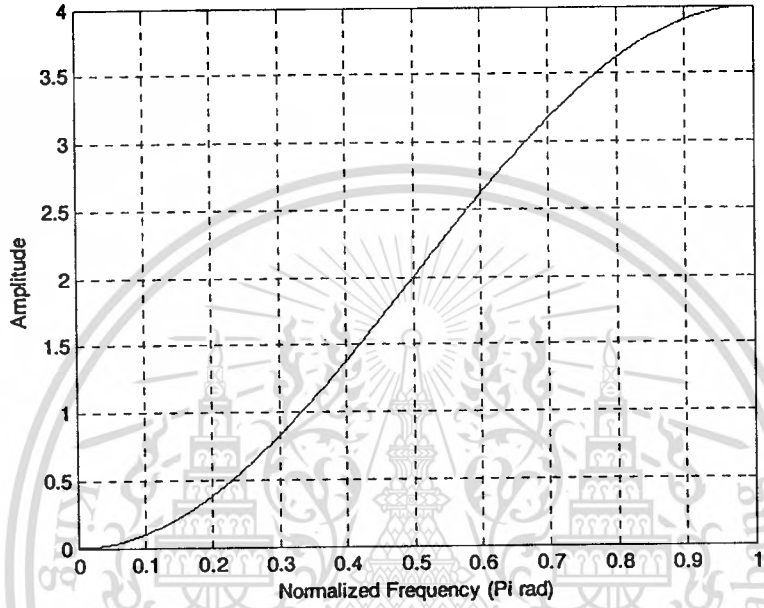


Fig. 7.2 Amplitude Response of Highpass type DPT at $N = 3$

So, we can observe that the frequency characteristic of such Pascal transform matrix is highpass filter and this is the reason why we call it highpass type DPT. The order of transfer function is equal to the dimension of Pascal transform matrix minus 1. Therefore, from eq. (7.7), we will call 2^{nd} order highpass type discrete Pascal filter (DPF) instead of DPT because the process of the operating signal has changed from matrix multiplication as in eq. (7.1) and eq. (7.3) to the convolution between filter coefficients and input signal as in eq. (7.7). For the higher dimension N or higher order DPF can be considered in the same manner.

Fig. 7.3 shows the amplitude response of highpass type DPF by showing the order of filter from $2^{nd} - 99^{th}$ order. We can notice that this type of filter has something different from the ordinary filter. For the ordinary filter, when the order is increased, the slope of the filter will be also increased with the same cut-off frequency. In this type of filter, when the order is increased, the slope of adjacent order changes slightly. The distinct change is cut-off frequency. In the case of highpass type DPF,

when higher order the cut-off frequency is increased, if focusing on bandwidth of passband, when higher order the passband will be narrower.

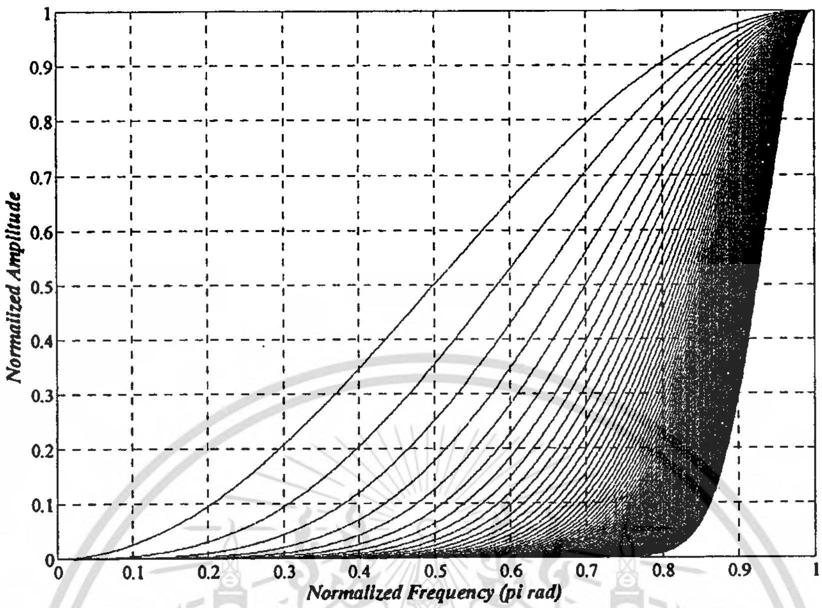


Fig. 7.3 Amplitude Response of Highpass type DPF from 2^{nd} to 99^{th} order

Consider the 2-D DPT with $N=3$, from eq. (7.2) we can express.

$$\begin{bmatrix} X_{0,0} & X_{0,1} & X_{0,2} \\ X_{1,0} & X_{1,1} & X_{1,2} \\ X_{2,0} & X_{2,1} & X_{2,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & -1 & 0 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{1,0} & x_{1,1} & x_{1,2} \\ x_{2,0} & x_{2,1} & x_{2,2} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.8}$$

$$= \underbrace{\begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{0,0}-x_{1,0} & x_{0,1}-x_{1,1} & x_{0,2}-x_{1,2} \\ x_{0,0}-2x_{1,0}+x_{2,0} & x_{0,1}-2x_{1,1}+x_{2,1} & x_{0,2}-2x_{1,2}+x_{2,2} \end{bmatrix}}_{\text{column operation}} \begin{bmatrix} 1 & 1 & 1 \\ 0 & -1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

The first matrix multiplication is comparable to the column operation of 3 vectors of 1-D signal. After that multiply again to the transpose of its Pascal matrix, which is comparable to the row operation of 3 vectors of 1-D that is obtained from column operation. Therefore, the final result is 2-D operation, or 2-D highpass type DPT in eq. (7.9).

$$\begin{bmatrix} X_{0,0} & X_{0,1} & X_{0,2} \\ X_{1,0} & X_{1,1} & X_{1,2} \\ X_{2,0} & X_{2,1} & X_{2,2} \end{bmatrix} = \begin{bmatrix} x_{0,0} & x_{0,0}-x_{0,1} & x_{0,0}-2x_{0,1}+x_{0,2} \\ [x_{0,0}-x_{1,0}] & [x_{0,0}-x_{1,0}]-[x_{0,1}-x_{1,1}] & [x_{0,0}-x_{1,0}]-2[x_{0,1}-x_{1,1}]+[x_{0,2}-x_{1,2}] \\ [x_{0,0}-2x_{1,0}+x_{2,0}] & [x_{0,0}-2x_{1,0}+x_{2,0}]-[x_{0,1}-2x_{1,1}+x_{2,1}] & [x_{0,0}-2x_{1,0}+x_{2,0}]-2[x_{0,1}-2x_{1,1}+x_{2,1}]+[x_{0,2}-2x_{1,2}+x_{2,2}] \end{bmatrix} \tag{7.9}$$

Later, we make the relationship of each input data in the input matrix as

$$\begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{1,0} & x_{1,1} & x_{1,2} \\ x_{2,0} & x_{2,1} & x_{2,2} \end{bmatrix} = \begin{bmatrix} x(m-2,n-2) & x(m-2,n-1) & x(m-2,n) \\ x(m-1,n-2) & x(m-1,n-1) & x(m-1,n) \\ x(m,n-2) & x(m,n-1) & x(m,n) \end{bmatrix} \tag{7.10}$$

Similar to 1-D, since we consider $x_{2,2}$ as the present input $x(m,n)$. So, $X_{2,2}$ should be the present output $y(m,n)$. From eq. (7.9) we would get

$$\begin{aligned} \therefore y(m,n) = {} & x(m,n) - 2x(m,n-1) + x(m,n-2) \\ & - 2x(m-1,n) + 4x(m-1,n-1) + 2x(m-1,n-2) \\ & + x(m-2,n) - 2x(m-2,n-1) + x(m-2,n-2) \end{aligned} \tag{7.11}$$

Form eq. (7.11), we can show the convolution equation that is used for computing the output $y(m,n)$ as follows,

$$y(m,n) = \sum_{k=0}^{2}\sum_{l=0}^{2} h(k,l)x(m-k,n-l) \tag{7.12}$$

where $h(k,l)$ is the impulse response of 2-D digital filter. Also, we can summarize that the impulse response of the highpass type 2-D DPT with $N=3$ in this example is $h^{(HP)}(k,l)$, and call it the highpass type Pascal convolution mask as shown in Fig. 7.4

1	-2	1
-2	4	-2
1	-2	1

Fig. 7.4 Highpass type Pascal Convolution Mask size 3×3

In Fig. 7.5, it shows the amplitude response of highpass Pascal convolution mask

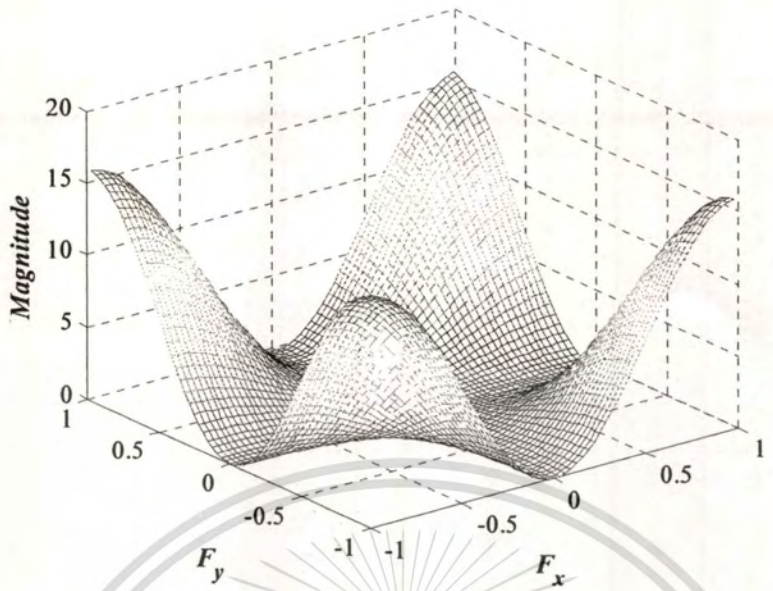


Fig. 7.5 Amplitude Response of Highpass type 2-D DPF size 3×3

At this point, we can see that the operation of Pascal transform matrix is changed from multiplication to convolution in DPF (Discrete Pascal Filter), as from eq. (7.3) to eq. (7.6) in case of 1-D processing, and from eq. (7.8) to eq. (7.11) in case of 2-D processing.

7.2.2 Frequency Response of Lowpass Type DPT

Similar to previous sub-section, only change highpass type Pascal transform matrix $\mathbf{P}^{(HP)}$ to lowpass type Pascal transform matrix $\mathbf{P}^{(LP)}$ both in the case of 1-D and 2-D. The lowpass type 1-D DPT with $N = 3$. We can summarize the 2^{nd} order transfer function as

$$H^{(LP)}(z) = 1 + 2z^{-1} + z^{-2} \tag{7.13}$$

and its frequency response is shown in Fig. 7.6,

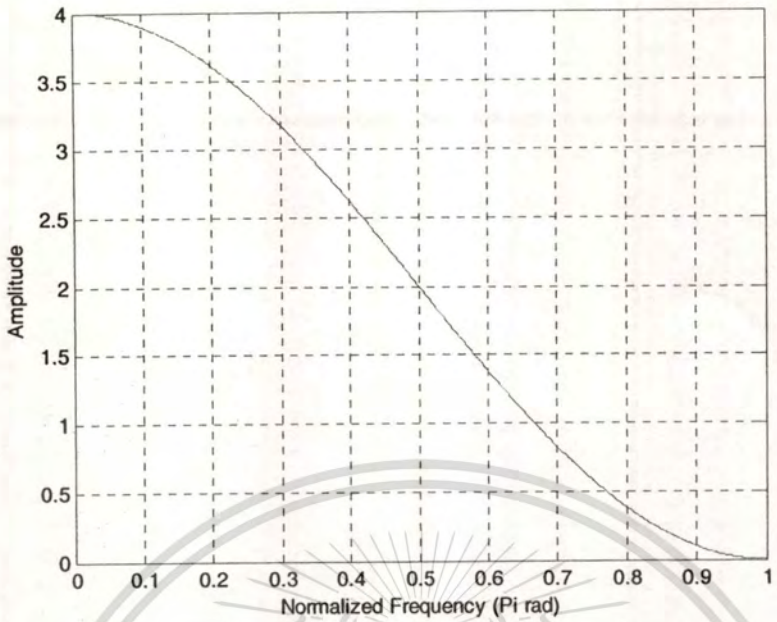


Fig. 7.6 Amplitude Response of Lowpass type DPT at $N = 3$

Fig. 7.7 shows the amplitude response of lowpass type DPF from 2^{nd} order to 99^{th} order. The results are as same as in the case of highpass type, that is higher order will give the narrower passband.

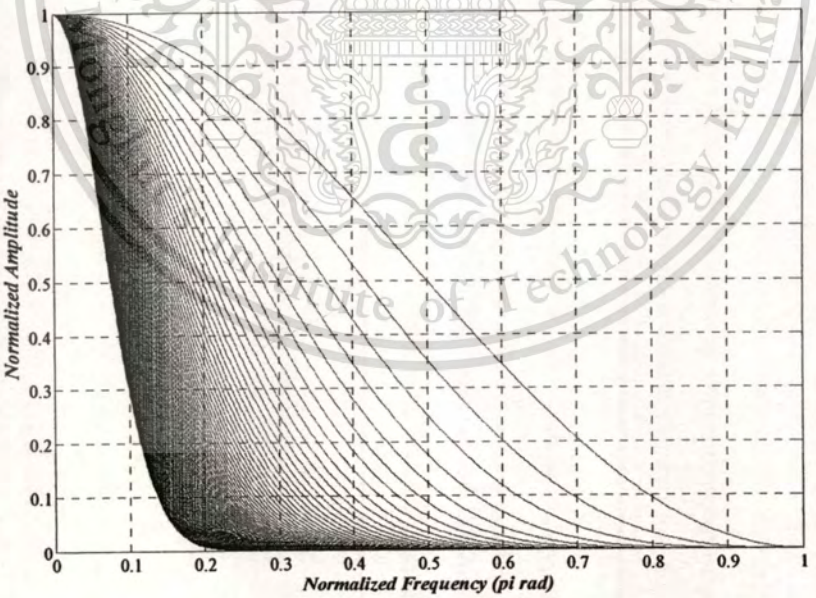


Fig. 7.7 Amplitude Response of Lowpass type DPF from 2^{nd} to 99^{th} order

The impulse response $h^{(LP)}(k,l)$ of the lowpass type 2-D DPF can be summarized as shown in lowpass type Pascal convolution mask size 3×3 in Fig.7.8.

$$\frac{1}{16}\times\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Fig. 7.8 Lowpass type Pascal Convolution Mask size 3×3

The value of (1/16) come from sum of all elements in the mask which are used to scale the coefficients in the mask and is necessary for 2-D LPF (Lowpass Filter). The amplitude response can be shown as in Fig 7.9.

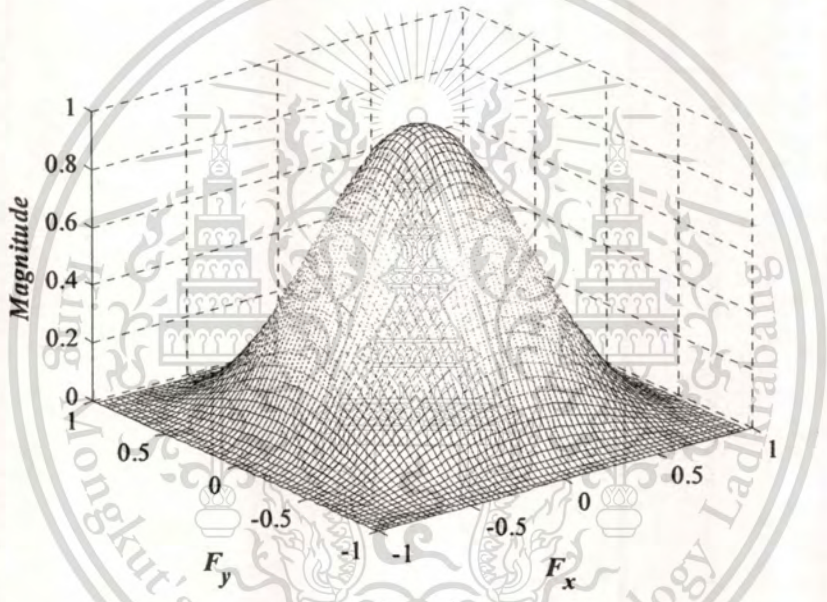


Fig. 7.9 Amplitude Response of Lowpass type 2-D DPF size 3×3

The convolution masks that shown above are filter coefficients which is used to identify the filter characteristic, sometimes may be called Gaussian filter convolution mask as in many digital image processing books. Normally, the convolution mask will be convolved to input image using 2-D difference equation as

$$y(m,n)=\sum_{k=0}^{2}\sum_{l=0}^{2}h(k,l)x(m-k,n-l)$$

where $h(k,l)$ are coefficients in the mask, it means that in case of 3×3 mask size require 9 multipliers and 8 adders/subtractors for operation, and it also include the Gaussian filter

operation. Although the Pascal convolution mask will same as the Gaussian filter convolution mask. But with specific filter structure from the Pascal matrix factorization can give the 2-D DPF and do not need any multipliers for the operation, only 12 adders/subtractors required for filter operation. Therefore, we call this type of filter DPF (the same characteristics as Gaussian filter but specific on filter structure).

7.3 Application of Discrete Pascal Filtering

In the previous section, we found that the hidden frequency characteristic in DPT can help us improve the operation of DPT from multiplication to convolution and we call it DPF. In this section, we will show how DPF filters the signal of both 1-D and 2-D, and both highpass and lowpass filtering and compare their results with the direct DPT proposed in [23-24] in order to show the weakness of such method.

7.3.1 Highpass Filtering Results

The first simulation, we will apply the single sinusoidal frequency 100 Hz to highpass type 1-D DPT and the simulation results can be shown as in Fig. 7.10

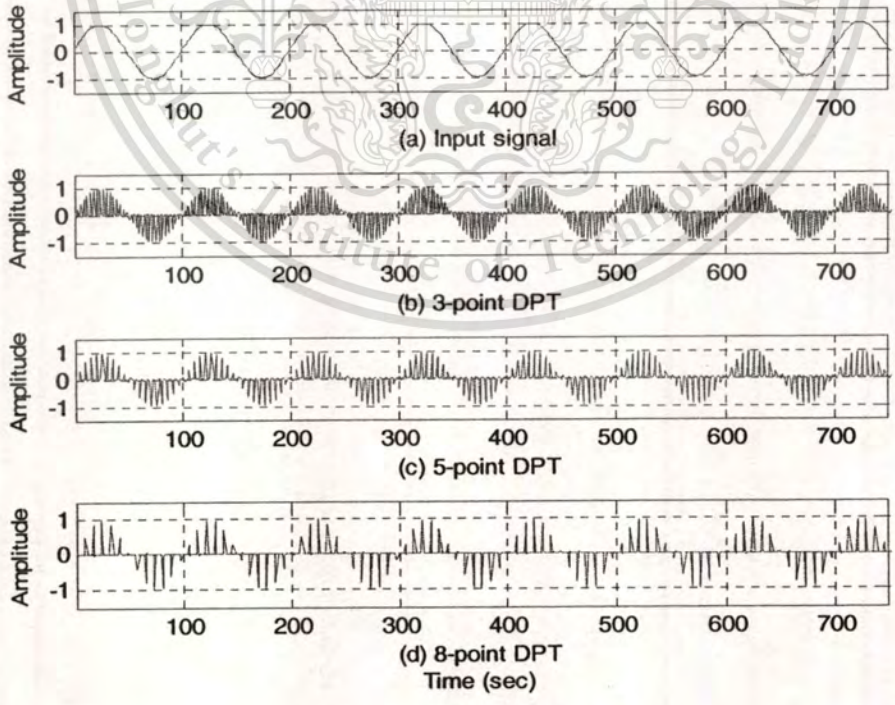


Fig. 7.10 Input signal and Transformed output of Highpass type 1-D DPT

The second simulation, the composite signal that is created from low frequency sinusoidal signal 100 Hz and high frequency sinusoidal signal 1500 Hz, will be applied to the 4^{th} order (if we compare to DPT, it would be 5-point DPT) highpass type 1-D DPF and 5-point highpass type 1-D DPT to make a comparison.

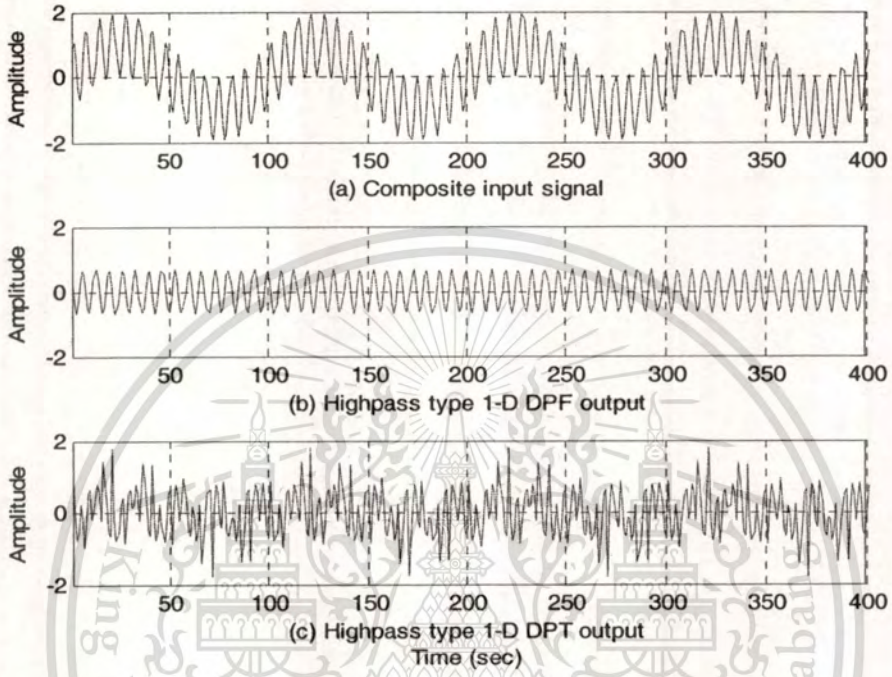


Fig. 7.11 The Comparison Result of Highpass type 1-D DPF and 1-D DPT

From Fig. 7.11, the highpass type 1-D DPF output can ensure that the highpass type 1-D DPF can be applied to highpass filter by the convolution method, but the highpass type 1-D DPT, which uses matrix multiplication, cannot give good or expected results to the transformed output. So, it is classed as a weakness of direct DPT.

The last simulation for this section, highpass type 2-D DPF and highpass type 2-D DPT will be shown. The input image size 256×256 pixels shown in Fig. 7.12 and highpass type Pascal convolution mask size 3×3 , 5×5 and 8×8 will be used for 2-D DPF. The highpass type 2-D DPT will use highpass type Pascal transform matrix with the same size as highpass type Pascal convolution mask to make a comparison with 2-D DPF.

Fig. 7.12 Original image

(a) Pascal Convolution Mask Size 3x3 (b) Pascal Convolution Mask Size 5x5 (c) Pascal Convolution Mask Size 8x8

(d) Pascal Transform Matrix Size 3x3 (e) Pascal Transform Matrix Size 5x5 (f) Pascal Transform Matrix Size 8x8

Fig. 7.13 The Comparison Result of Highpass type 2-D DPF and 2-D DPT

From Fig. 7.13, the highpass type 2-D DPF with the mask size 3×3 gives the best result. Higher mask size gives the poor result because the highpass type Pascal convolution mask does not have a scaling coefficient. Also, when using a higher mask size, it will increase the gain and result in the increase of noise in the output image. Therefore, the highpass type 2-D DPF with the mask size 3×3 can detect the edge of an image better than the 2-D DPT's result.

7.3.2 Lowpass Filtering Results

The first simulation, we apply the sinusoidal frequency 100 Hz to lowpass type 1-D DPT and the simulation results are as shown in Fig. 7.14.

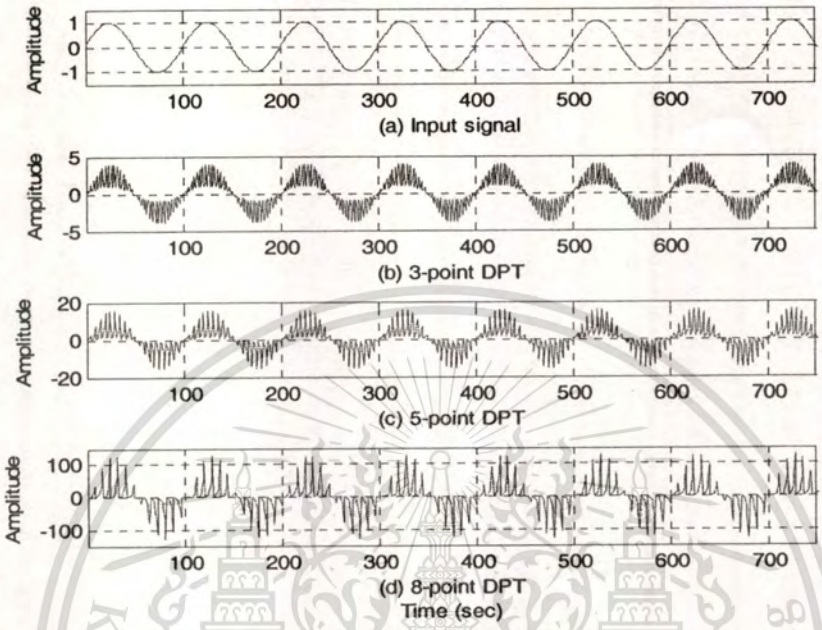


Fig. 7.14 Input signal and Transformed output of Lowpass type 1-D DPT

The second simulation, the composite signal that created from low frequency sinusoidal signal 100 Hz plus high frequency sinusoidal signal 4000 Hz will be applied to the 4^{th} order lowpass type 1-D DPF and 5-point lowpass type 1-D DPT.

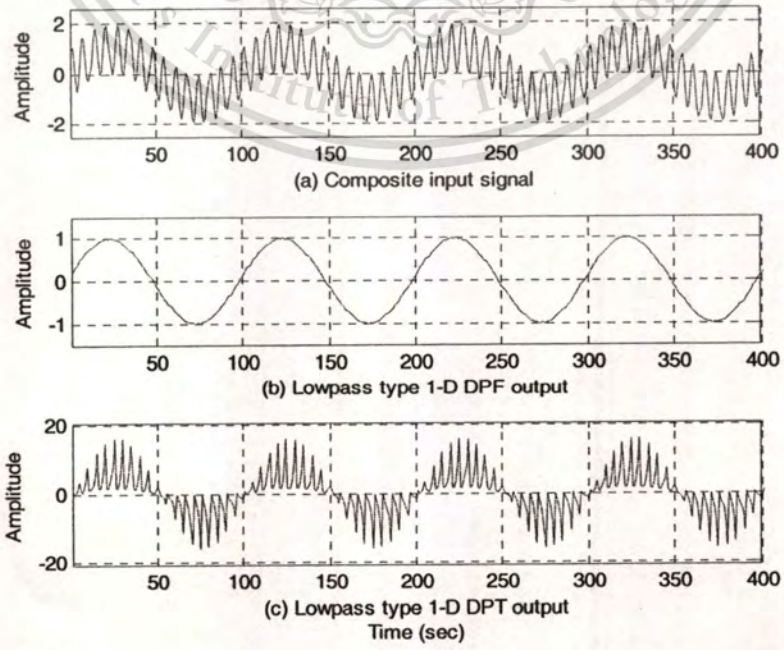
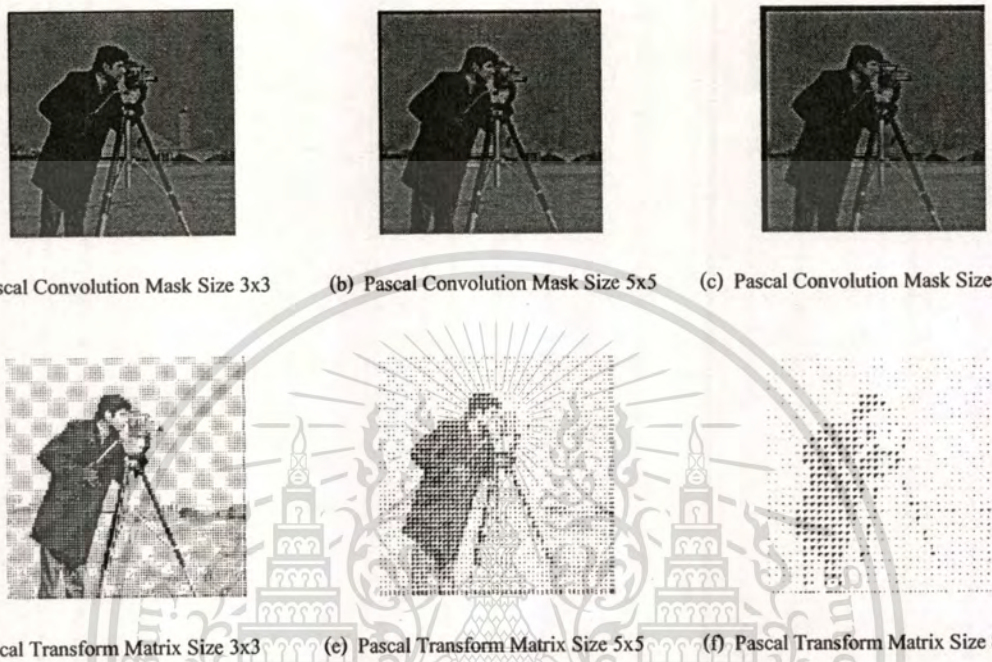


Fig. 7.15 The Comparison Result of Lowpass type 1-D DPF and 1-D DPT

From Fig. 7.15, the lowpass type 1-D DPF output can ensure that the lowpass type 1-D DPF can be applied to lowpass filter, but the lowpass type 1-D DPT cannot give the expected results.

The last simulation, lowpass type 2-D DPF and lowpass type 2-D DPT will be compared in the same manner as in highpass type case.



(a) Pascal Convolution Mask Size 3x3 (b) Pascal Convolution Mask Size 5x5 (c) Pascal Convolution Mask Size 8x8

(d) Pascal Transform Matrix Size 3x3 (e) Pascal Transform Matrix Size 5x5 (f) Pascal Transform Matrix Size 8x8

Fig. 7.16 The Comparison Result of Lowpass type 2-D DPF and 2-D DPT

From Fig. 7.16, the lowpass type 2-D DPF can smoothen an impage, or lowpass filter an image, better than the lowpass type 2-D DPT's result.

7.4 Hardware Implementation of 2-D DPF on FPGA and Display on VGA

In this section, we propose the hardware implementation of 2-D DPF on FPGA and display the image filtering results directly on VGA without using a PC (personal computer) for this system. The input image size 128 x 128 pixels will be stored on EEPROM. FPGA can read the input data, then process by using the highpass type 2-D DPF and send the filtered output to VGA controller unit for display on VGA. The overall system can be shown in Fig. 7.17.

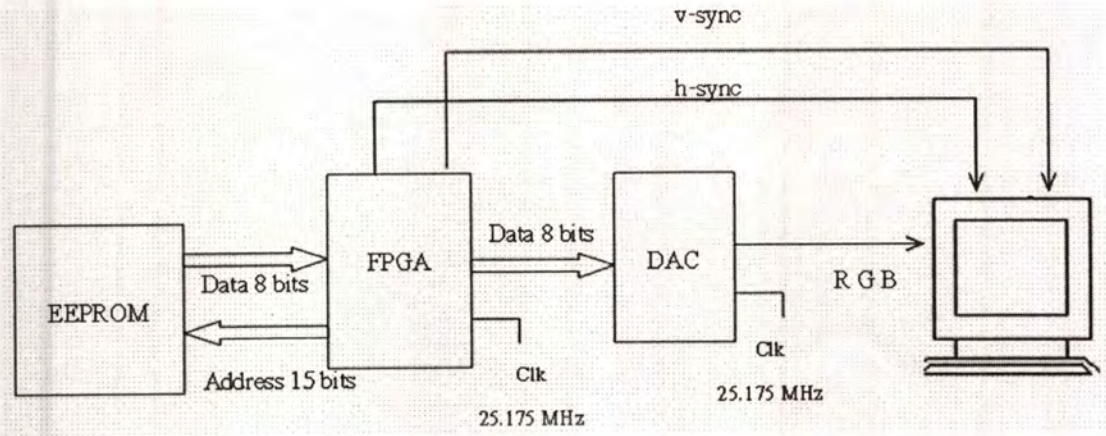


Fig. 7.17 Overall System for 2-D DPF Implementation on FPGA and Display on VGA

In Fig. 7.18 shows the hardware prototype of the experimental test set.

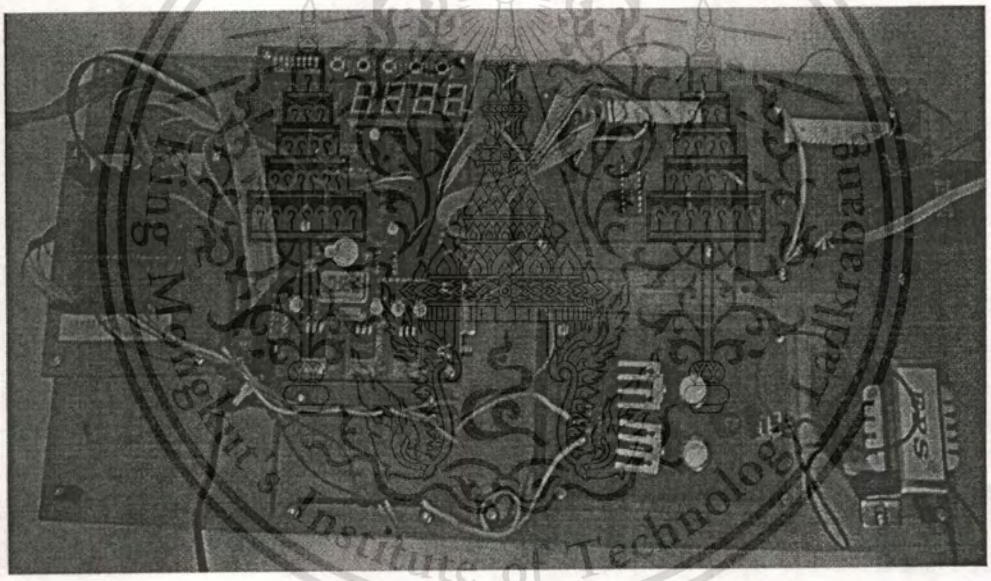


Fig. 7.18 Hardware Prototype for Experiment

Fig. 7.19 shows all circuits which are designed using VHDL and synthesis for mapping on FPGA. The main circuits consist of VGA controller, a counter used for generating the memory address, video_on block used to control the timing of data for display which corresponds to vertical and horizontal synchronization signal, and block_process which is the block of the processing unit.

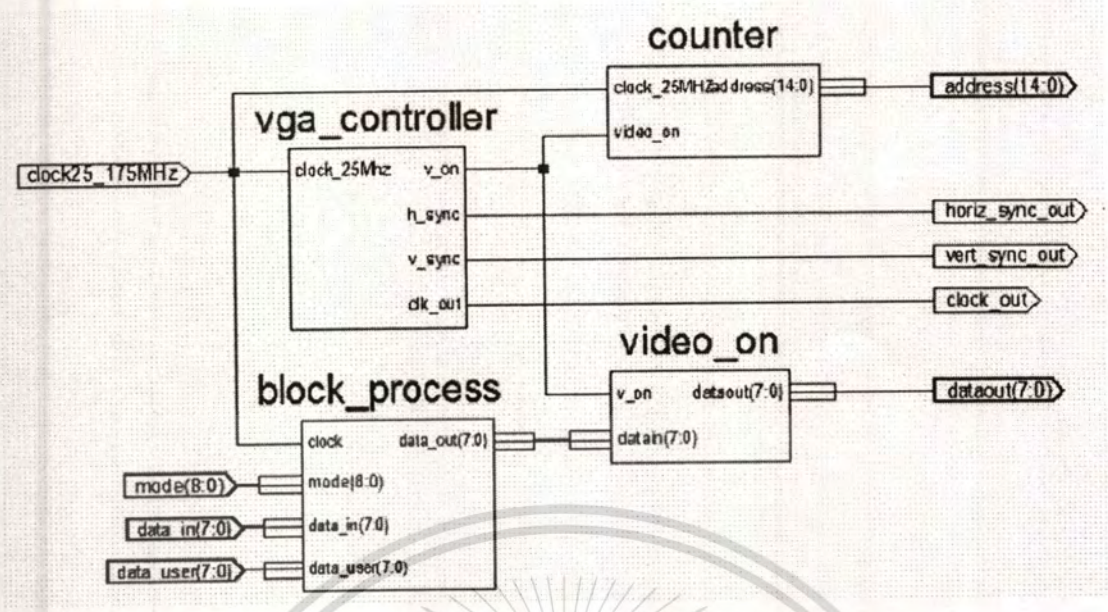


Fig. 7.19 All Circuits inside FPGA

The signals that need to control VGA monitor are vertical (upper trace) and horizontal (lower trace) synchronization signal can be shown in Fig. 7.20 and zoom-in version in Fig. 7.21.

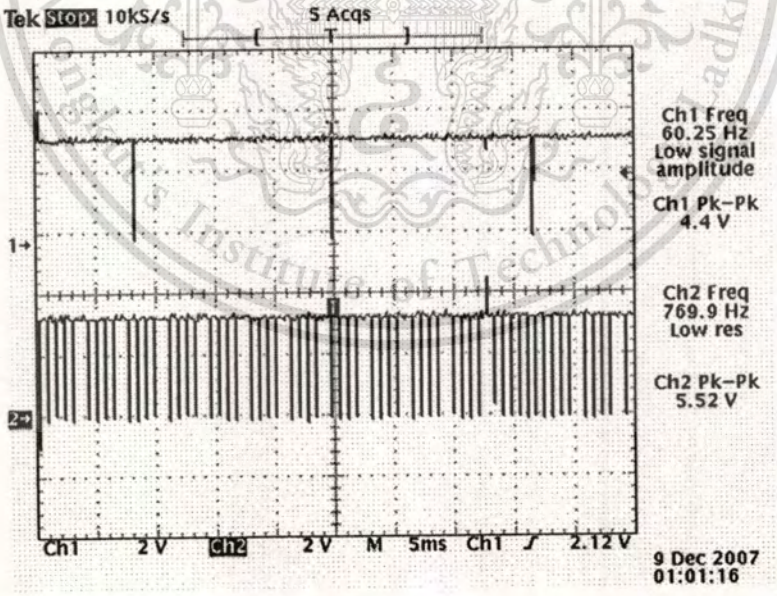


Fig. 7.20 Vertical (upper trace) and Horizontal (lower trace) Synchronization Signal

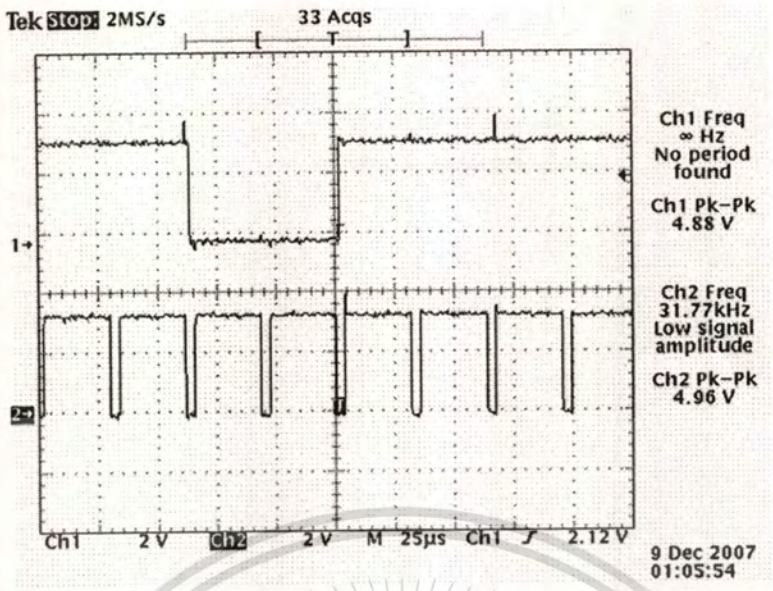


Fig. 7.21 Zoom-in of Vertical (upper trace) and Horizontal (lower trace) Synchronization Signal

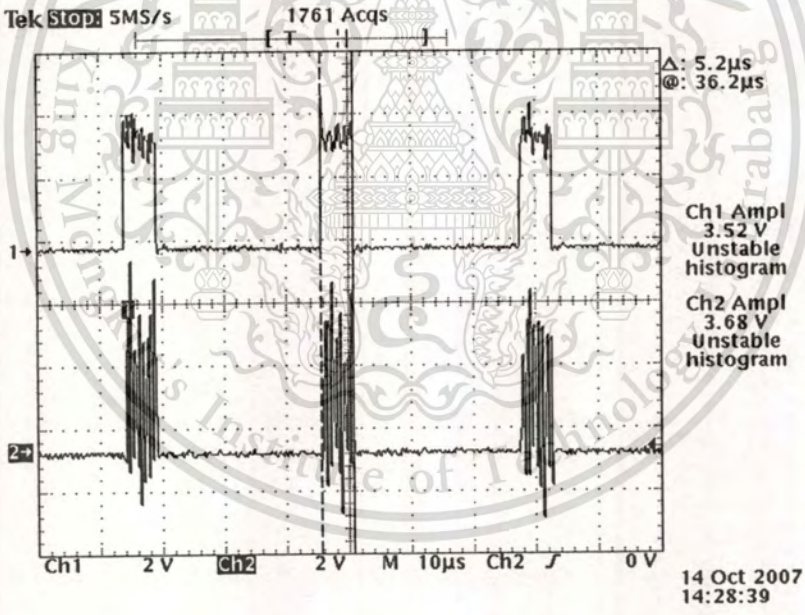


Fig. 7.22 Video-On Signal (upper trace) versus One bit of Address line of Memory (lower trace)

From Fig. 7.22 shows the timing period that FPGA will access the data form memory which is controlled by the video-on signal and in Fig 7.23 shows the video-on (upper trace), output from memory sent to FPGA (middle trace) and output from FPGA that sends the digital to analog converter to VGA (lower trace). Therefore, the video-on is the important signal used to control the timing of data for overall processing.

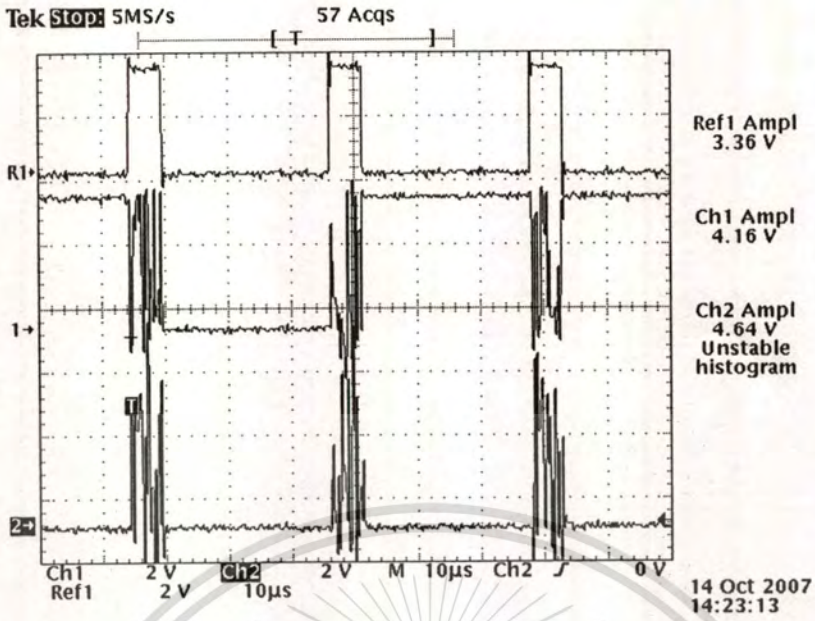


Fig. 7.23 Video-On Signal (upper trace), One bit of Memory Output (middle trace) and One bit of D/A Input (lower trace)

The highpass type 2-D DPF will be implemented on FPGA with Pascal convolution mask size 3x3. Then, as the input image size is 128x128 pixels, the delay line used to generate the rowth index can be replaced by 128 delay element (z^{-128}). The structure of 2-D DPF on FPGA can be shown in Fig. 7.24

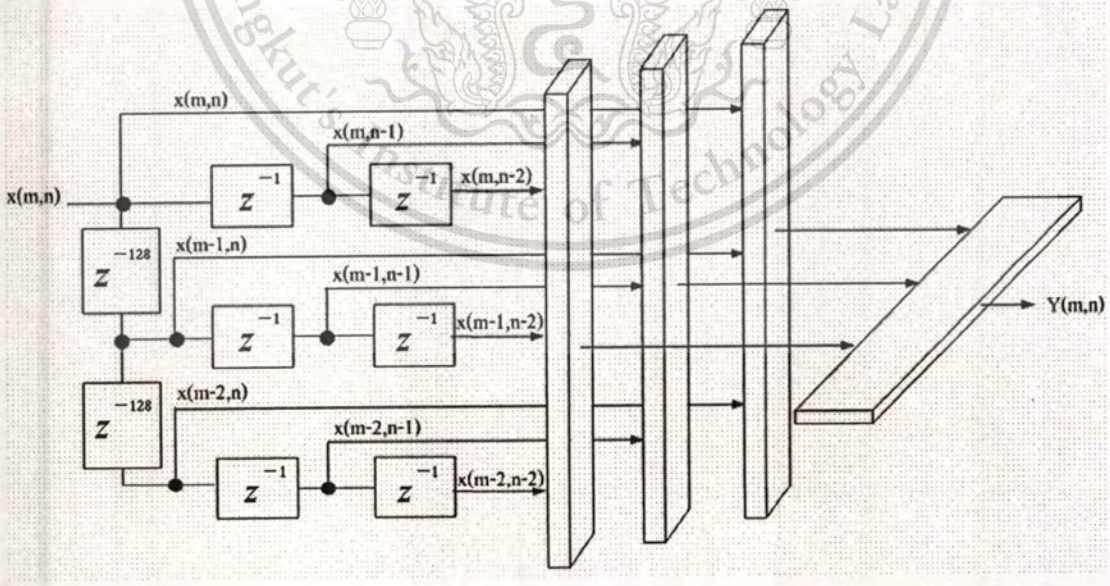
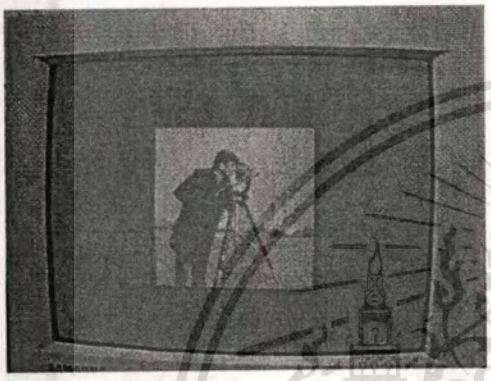


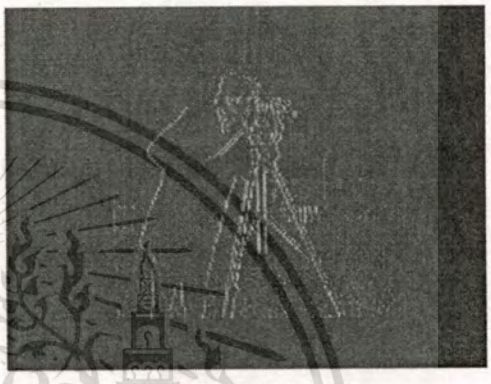
Fig. 7.24 Structure of 2-D DPF on FPGA

For the experimental results from the hardware prototype are shown in Fig. 7.25-7.29

(a) Original Image



(b) Original Image on VGA



(c) Filtered Image on VGA

Fig. 7.25 The 2-D DPF Experimental Result1 from Hardware Prototype

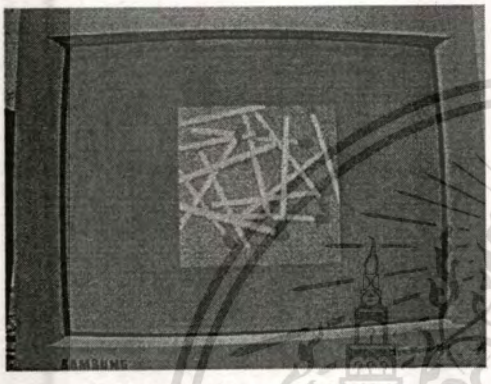
(a) Original Image

(b) Original Image on VGA

(c) Filtered Image on VGA

Fig. 7.26 The 2-D DPF Experimental Result2 from Hardware Prototype

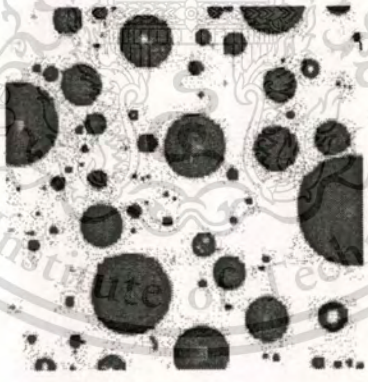
(a) Original Image



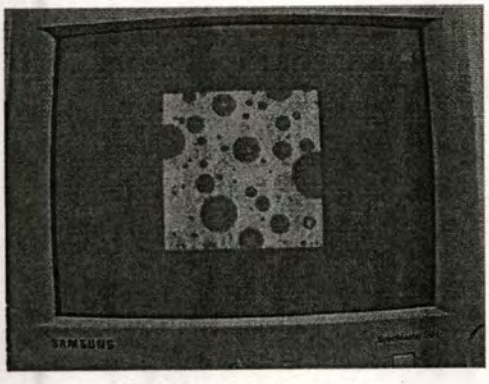
(b) Original Image on VGA

(c) Filtered Image on VGA

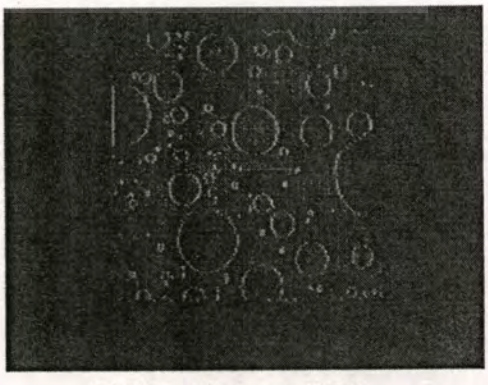
Fig. 7.27 The 2-D DPF Experimental Result3 from Hardware Prototype



(a) Original Image



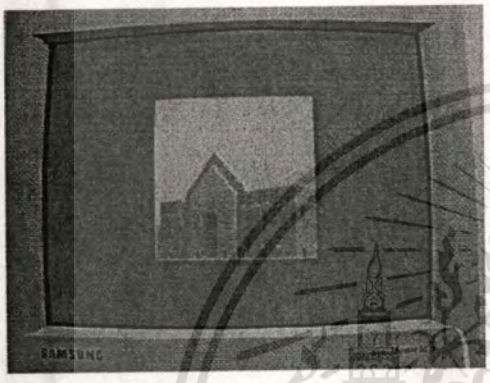
(b) Original Image on VGA



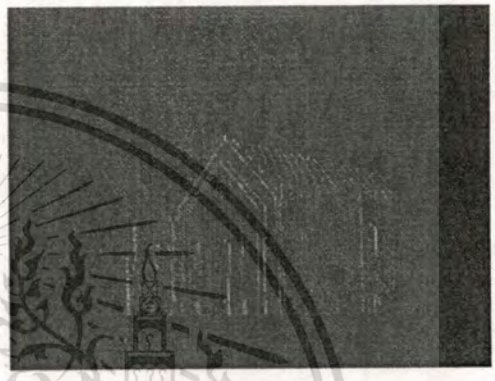
(c) Filtered Image on VGA

Fig. 7.28 The 2-D DPF Experimental Result4 from Hardware Prototype

(a) Original Image



(b) Original Image on VGA



(c) Filtered Image on VGA

Fig. 7.29 The 2-D DPF Experimental Result5 from Hardware Prototype

7.5 Hardware-Software Co-Design for Digital Image Filtering Using 2-D DPF

This section proposes a hardware-software (HW-SW) co-design system for digital image processing operation. The targets of this system are mobile or embedded systems with image processing capability. We use SCE-MI (Standard Co-Emulation Modeling Interface) to communicate the system software to the DPF hardware. The SCE-MI was designed only for verification with emulators and/or simulators, but the nature is an abstraction layer of communications and we though that it is usable to develop the embedded applications on a PC. The hardware part connects to the software part on a PC via SCE-MI which hide underlying PCI bus, device driver and FPGA.

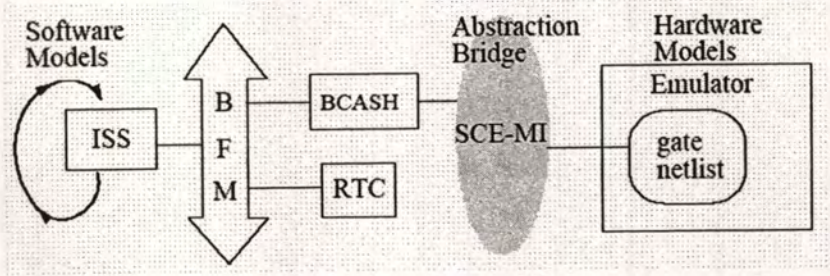


Fig. 7.30 Using the SCE-MI as An Abstraction Bridge

As seen in Fig. 7.30, SCE-MI is an abstraction bridge between hardware models and software models. The communications in SCE-MI is message oriented, and we can use high level abstraction message between hardware and software as shown in Fig. 7.31.

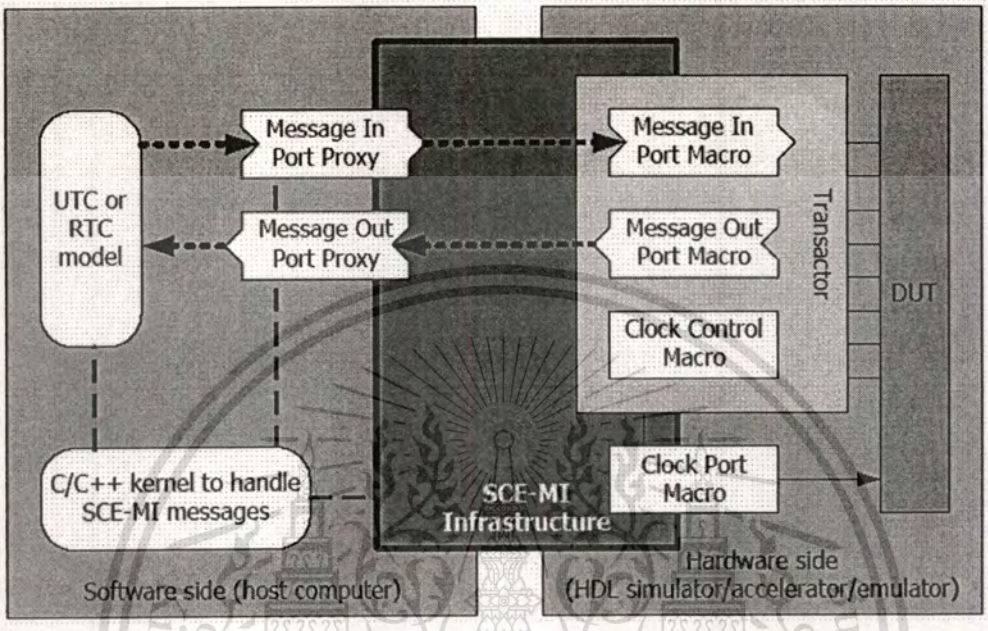


Fig. 7.31 High-Level view of Run-Time components

The 2-D DPF will be the hardware part on FPGA and the working cooperate with the software part which is written in C++ using SCE-MI. Fig. 7.32 is the proposed HW-SW co-design system, through the targets of this system are mobile or embedded systems with image processing capability, we can develop the software part as a PC application using SCE-MI API.

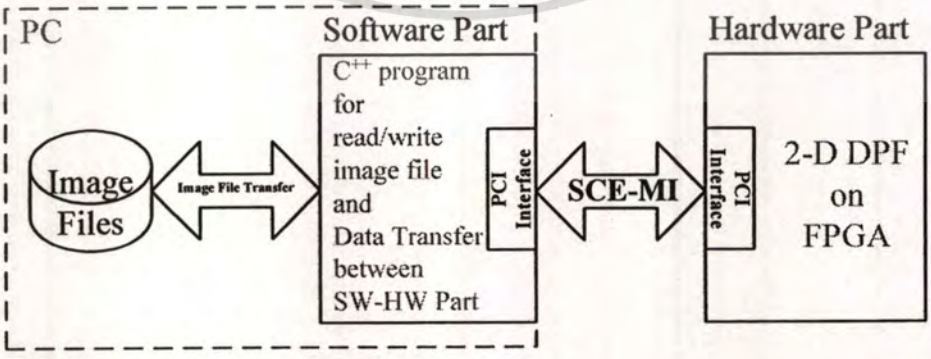
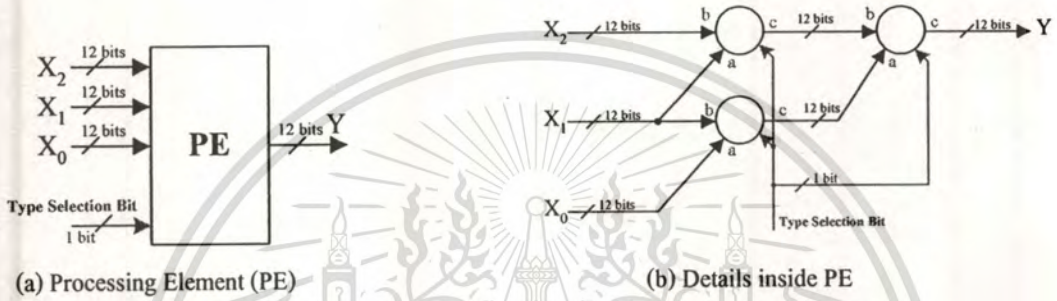


Fig. 7.32 The Proposed HW-SW Co-Design System

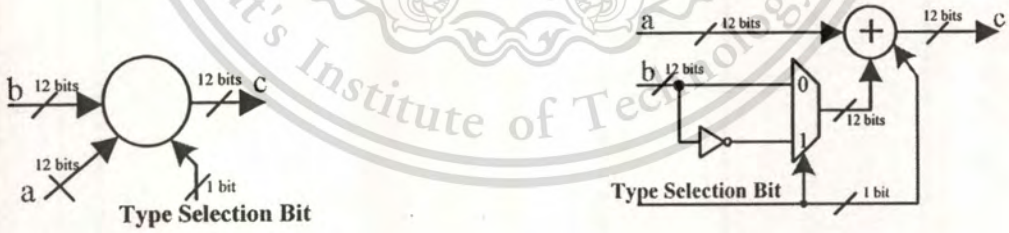
Fig. 7.32 shows the experimental development system structure. The 2-D DPF architecture is designed with SFL (Structured Functional description Language) and implemented into FPGA with automatically generated transactor and PCI interface core module. Whole FPGA logic is compiled and mapped on Altera Stratix FPGA (EP1S10F780C7ES) using the QuatusII software. The 2-D DPF for this section can operate both highpass and lowpass type operations depending on the selection from type selection bit, and consists of four 1-D DPF's. We call each 1-D DPF a Processing Element (PE).



(a) Processing Element (PE) (b) Details inside PE

Fig. 7.33 Processing Element for 1-D DPF

Fig. 7.33 shows the details inside each PEs block which is network of 2's comp unit (two's complement adder/subtractor unit), and the type selection bit will determine the mode of addition or subtraction. The details of 2's comp unit can be shown in Fig. 7.34.



(a) Symbol of 2's comp unit (b) Circuits inside 2's comp unit

Fig. 7.34 Two's complement adder/subtractor unit (2's comp unit) for PE

For experimental setup, we divide the experiment into 2 cases as HW-SW Co-Design1 and HW-SW Co-Design2 as follows,

<u>HW-SW Co-Design1:</u>

The baseline structure of 2-D DPF is shown in Fig. 7.35. The basic operation can be described as followings, each pixel data of three lines will send from SCE-MI to 2-D DPF

circuits. The group of unit delay will prepare 9 points of data for processing. PE_1, PE_2, and PE_3 will process the data from rows 1,2,3, respectively (This operation is equivalent to independent row operation). After that, the outputs from these PEs will be the inputs of PE_4 to produce the result. The mode of operation for each PEs can be selected by type selection bit which is '0' for lowpass type operation and '1' for highpass type operation.

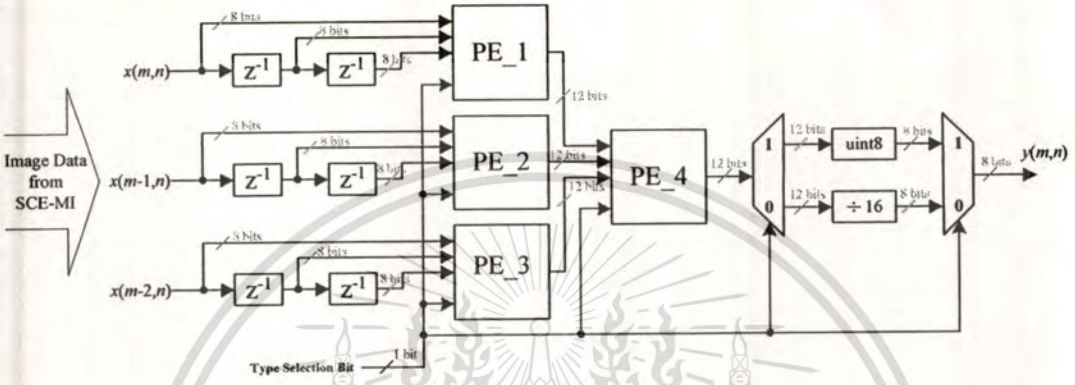


Fig. 7.35 The Proposed 2-D DPF structure for HW part on FPGA

If the type selection bit is '0', the result from PE_4 will go to the division by 16 circuit (practical design by shift right 4 bits) before passing through to be the output image. This output image is a lowpass output which corresponds to the operation by a lowpass type Pascal convolution mask size 3x3 for normal image filtering. For type selection bit is '1', the result from PE_4 will go to uint8 (stand for, unsigned integer 8 bits) circuit because the result from the highpass type operation may give the negative value result but our image representation can show only the positive value between 0-255 in 8 bits/pixel system. Therefore, uint8 circuit will set all negative value to 0, and all positive value over 255 will be set to 255.

HW-SW Co-Design2:

In 2-D DPF image processing, each conjunct pixels will be reduced in the conjunct processing elements. Therefore, we can use transferred data from PC in parallel processing. In Fig. 7.36 we will show another 2-D DPF design which utilize 6 of the basic 2-D DPF units. With this structure, up to 6 pixels will be calculated in parallel.

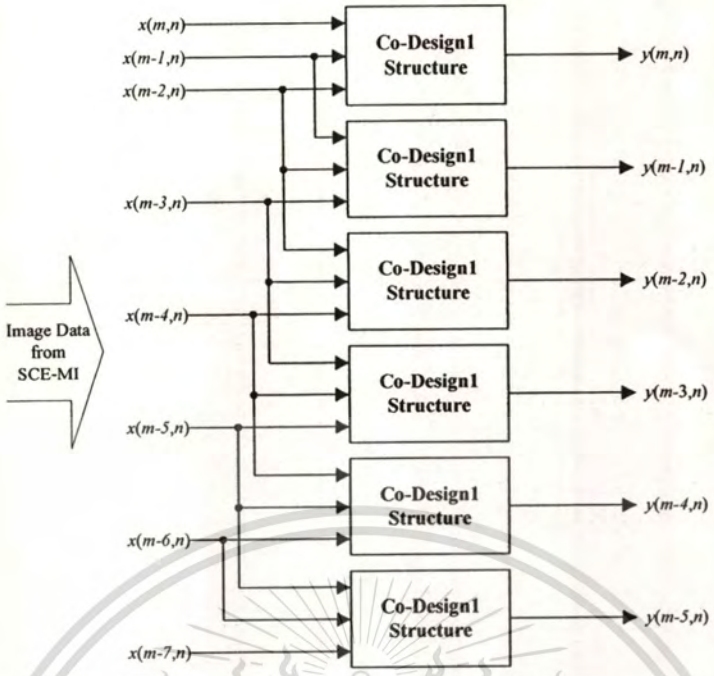


Fig. 7.36 The Proposed 2-D DPF Co-Design2 structure for HW part on FPGA

For experimental results, the proposed system works and we obtained output images as shown in. Fig. 7.37-7.41

(a) Input Image (b) Highpass Output (c) Lowpass Output

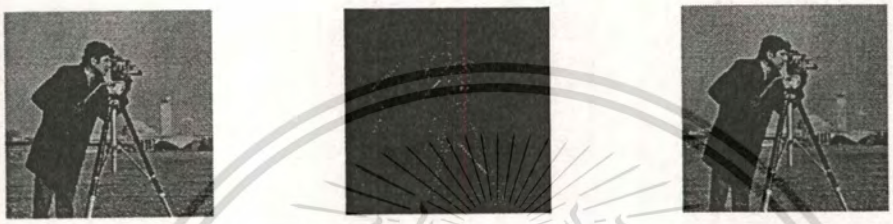
Fig. 7.37 Input Image size 64×64 pixels

(a) Input Image (b) Highpass Output (c) Lowpass Output

Fig. 7.38 Input Image size 128×128 pixels

(a) Input Image (b) Highpass Output (c) Lowpass Output

Fig. 7.39 Input Image size 256×256 pixels



(a) Input Image (b) Highpass Output (c) Lowpass Output

Fig. 7.40 Input Image size 512×512 pixels

(a) Input Image (b) Highpass Output (c) Lowpass Output

Fig. 7.41 Input Image size 1024×1024 pixels

The logic synthesis results are shown in Table 7.1.

Table 7.1 Logic Synthesis Results for Altera's Stratix FPGA

	Used Logic Cells	Maximum Frequency (MHz)
HW-SW Co-Design1	402	390.02
HW-SW Co-Design2	1,248	390.02
PCI Interface for SCE-MI	984	27.47
Overall circuits Co-Design1	1,386	27.47
Overall circuits Co-Design2	2,232	27.47

In Table 7.2, we show the execution time purely on a PC (Intel Celeron 2.53 GHz CPU), estimation execution time for 200 MHz ARM processor, our experiment system1 (simple 2-D DPF) and our experiment system2 (6 pixels in parallel).

Table 7.2 Execution Time of 2-D DPF Image Processing

	Processing Speed (ms)			
Image Size (pixels)	**PC Pure SW**	**ARM Est. SW**	**HW-SW Co-Design1**	**HW-SW Co-Design2**
64×64	0.18	9.26	13.7	3.2
128×128	0.71	36.9	54	13
256×256	2.8	147	216	49
512×512	11	634	861	194
1024×1024	41	2,371	3,435	779

As shown in Table 7.2, the PCI bus overhead is heavy and FPGA emulation does not speed up the processing. But it will be useful when implemented on some embedded systems which utilize an ARM processor inside.

7.6 Conclusion

The investigation of frequency characteristic or filtering characteristic that is hidden in the DPT was proposed, both highpass type and lowpass type DPT and both 1-D and 2-D signal are considered. The results from frequency characteristic investigation can be used to upgrade the ordinary DPT to DPF. The DPF can give good results in sense of signal filtering and can clarify the weakness of ordinary or direct DPT. Consequently, both 1-D and 2-D and both highpass type and lowpass type DPF can be applied to signal filtering applications.

Chapter 8

Conclusions

Conclusions of overall Pascal matrices research in this thesis can be summarized as follows, in chapter 2, a one-to-one mapping method for transforming the *s*-domain IIR transfer functions to the *z*-domain ones or vice versa using the first-order *s-z* transformations have been proposed. The original prototype IIR filter can be completely restored from the transformed one. Based on the one-to-one mapping, various inverses of the generalized Pascal matrices have been proven. Also, recurrence formulas have been derived for recursively computing the generalized Pascal matrices from their boundary elements, which simplifies the generations of the generalized Pascal matrices. Chapter 3, a unified s-z transformation model was used to derive and prove a more unified Pascal matrix. Both lowpass-to-lowpass and lowpass-to highpass transformations are included in this single model. Chapter 4 proposed the method for s-z transformation using Pascal matrix operations cooperate with the frequency transformation, this method is very efficient in the case of high order filter design and simpler than substituting the relationship between s-z into the transfer function directly. The new method for designing a biquad digital filter based on modified Pascal matrix and its filter structure realization has been proposed in chapter 5, it is very easy since it needs only one matrix equation to compute the filter coefficients, which correspond to the new proposed filter structure. Moreover, computing the denominator is the same as an ordinary 2^{nd} direct form II structure, and the number of delay units used is the same as a single output filter. With only some modification in feed-forward path, this structure can give 5 outputs by an extension circuit compared to normal 2^{nd} order IIR filter that can give only 1 output. In chapter 6 the DPT hardware realization of both 1-D and 2-D and both highpass type and lowpass type was proposed. The hardware realization using Pascal matrix factorization or decomposition into binary (1,0-1) matrices can give an efficient structure based on a butterfly unit for DPT. The obtained DPT flow graph can be used to transform without multiplications, only additions are needed. In the last chapter, the investigation of frequency characteristic or filtering characteristic that is hidden in the DPT was proposed, both highpass type and lowpass type DPT and both 1-D and 2-D signal were considered. The results from frequency characteristic investigation can be used to upgrade the ordinary DPT to DPF. The DPF can give good results in sense of signal filtering and can clarify the weakness of ordinary or direct DPT.

References

[1] S. K. Mitra, *Digital Signal Processing: A Computer-Based Approach*, 3rd Edition. New York: McGraw-Hill, 2006.

[2] H. M. Power, "The mechanics of the bilinear transformation," *IEEE Trans. Educ.* (Short Papers), vol. 10, pp. 114-116, June 1967.

[3] E. I. Jury, "Remarks on "The mechanics of bilinear transformation"," *IEEE Trans. Audio and Electroacoust.*, vol. 21, pp. 380-382, Aug. 1973.

[4] E. I. Jury and O. W. C. Chan, "Combinatorial rules for some useful transformations," *IEEE Trans. Circuit Theory*, vol. 20, No. 5, pp. 476-480, Sept. 1973.

[5] M.-S. Lee and C. Chang, "Switched-capacitor filters using the LDI and bilinear transformations ," *IEEE Trans. Circuits Syst.*, vol. 28, No. 4, pp. 265-270, Apr. 1981.

[6] V. Biolkova and D. Biolek, "LDI matrix and its utilization for the circuit analysis and design," *Proc. Radioelektronika'99*, pp. 38-41, Brno, Czech Republic, 1999.

[7] Y. Fukui, N. Yabuki, and A. Kosaka, "New s-z transformation and its switched capacitor realization," *Proc. IEEE ISCAS'88*, pp. 1999-2002, 1988.

[8] B. Psenicka, F. Garcia-Ugalde, and A. Herrera-Camacho, "The bilinear z transform by Pascal matrix and its application in the design of digital filter," *IEEE Trans. Signal Process. Lett.*, vol. 9, No. 11, pp. 368-370, Nov. 2002.

[9] B. Psenicka and F. Garcia-Ugalde, "z transform from lowpass to bandpass by Pascal matrix," *IEEE Trans. Signal Process. Lett.*, vol. 11, No. 2, pp. 282-284, Feb. 2004.

[10] J. Konopacki, "The frequency transformation by matrix operation and its application in IIR filters design," *IEEE Signal Process. Lett.*, vol. 12, No. 1, pp. 5-8, Jan. 2005.

[11] S. Chivapreecha, S. Sriyapong, S. Junnapiya, and K. Dejhan, "Bilinear s-z with frequency transformation using Pascal matrix operation," *Proc. IEEE ISCIT'05*, pp. 739-742, Beijing, China, Oct. 2005.

[12] S. Chivapreecha and K. Dejhan, "Pascal matrix operation for bilinear s-z with frequency transformation," *Proc. ITC-CSCC'06*, vol. III, pp. 129-132, Chiang Mai, Thailand, July 2006.

[13] V. Biolkova and D. Biolek, "Generalized Pascal matrix of first order s-z transforms," *Proc. IEEE ICECS'99*, pp. 929-931, Pafos, Cyprus, 1999.

[14] D. Biolek and V. Biolkova, "Algorithmic s-z transformations for continuous-time to discrete-time filter conversion," *Proc. IEEE ISCAS'01*, vol. I, pp. 588-590, Sydney, Australia, May 2001.

[15] K. Ichige, N. Otsuka, and R. Ishii, "An automatic design procedure of IIR digital filters from an analog low-pass filter," *Signal Processing*, vol. 57, No. 3, pp. 223-231, 1997.

[16] T.-B. Deng, S. Chivapreecha, and K. Dejhan, "Generalized Pascal matrices, inverses, computations and properties using one-to-one rational polynomial s-z transformations," accepted to publish in *IEEE Trans. Circuits Syst. I: Regular Papers.*

[17] Z.-S. Kang, "An analytical Pascal matrix transform for s-to-z domain transfer functions," *IEEE Trans. Signal Process. Lett.*, vol. 13, No. 10, pp. 597-600, Oct. 2006.

[18] E. C. Ifeachor and B. W. Jervis, *Digital Signal Processing : A Practical Approach*, Addison-Wesley Publishers Ltd., 1993.

[19] M. E. Van Valkenburg, *Analog Filter Design*, Holt-Saunders International Editions : New York, 1982.

[20] C. Pradabpet, S. Yimman, W. Hinjit, S. Chivapreecha and K. Dejhan, "Design and implementation of biquad digital filter," *Proc. The 9th Asia-Pacific Conference on Communications (APCC 2003)*, vol. 3, pp. 1138-1142, September 2003.

[21] S. Chivapreecha, S. Yimman, C. Pradabpet and K. Dejhan, "FPGA implementation of multi-functional digital filter based on non-recursive scheme," *Proc. The International Conference on Robotics, Vision, Information and Signal Processing (ROVISP 2005)*, pp. 20-24, July 2005.

[22] W. Mongkhonmalee, S. Chivapreecha, S. Tooprakai, and K. Dejhan, "Biquad digital filter design using Pascal matrix," *Proc. International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2007)*, July 2007.

[23] M. F. Aburdene and T. J. Goodman, "The Discrete Pascal Transform and Its Applications," *IEEE Signal Processing Letters,* vol. 12, No. 7, pp. 493-495, July 2005.

[24] T. J. Goodman and M. F. Aburden, "A hardware implementation of the discrete Pascal transform for image processing," *Proc. of SPIE-IT&T Electronic Imaging,* SPIE vol. 6064, pp.60640 H-1 – 6064 H-8, 2006.

[25] S. Chivapreecha, U. Nithirochananont, and K. Dejhan, "Invertigation of frequency characteristic in discrete Pascal transform and Its applications," *Proc. The 4th International Colloquium on Signal Processing and Its Application (CSPA 2008)*, Kuala Lumpur, Malaysia, March 2008.

[26] A. N. Skodras, "Fast Discrete Pascal Transform," *Electronics Letters*, vol. 42, No. 23, November 2006.

[27] A. Edelman and G. Strang, "Pascal Matrices," *Am. Math Mon.*, pp. 189- 197, 2004.

Related Publications

1. Tian-Bo Deng, ***Sorawat Chivapreecha***, and Kobchai Dejhan, "Generalized Pascal Matrices, Inverses, Computations and Properties Using One-to-One Rational Polynomial s-z Transformations," ***Accepted for Publication in IEEE Trans. on Circuits and Systems I (CAS-I) : Fundamental Theory and Applications.***
2. ***S. Chivapreecha***, A. Jaruvarakul, and K. Dejhan, "Modified Pascal Matrix for Biquad Digital Filter Design and Its Filter Structure Realization," *Proc. The 4th International Colloquium on Signal Processing and Its Application (CSPA 2008)*, Kuala Lumpur, Malaysia, March 7-9, 2008.
3. Tian-Bo Deng, ***Sorawat Chivapreecha***, and Kobchai Dejhan, "Generalized Pascal Matrices and Inverses Using One-to-One Rational Polynomial s-z Trnasformations," *Proc. The 23rd International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2008)*, Shimonoseki, Yamagishi, Japan, July 6-9, 2008.
4. Tian-Bo Deng, ***Sorawat Chivapreecha***, and Kobchai Dejhan, "Recurrent Formula and Property of Generalized Pascal Matrix," *Proc. The 23rd International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2008)*, Shimonoseki, Yamagishi, Japan, July 6-9, 2008.
5. ***S. Chivapreecha,*** A. Jaruvarakul, N. Jaruvarakul and K. Dejhan, "Adaptive Equalization Architecture Using Distributed Arithmetic for Partial Response Channels," *Proc. 2006 IEEE Tenth International Symposium on Consumer Electronics (ISCE2006)*, St. Petersburg, Russia, June 28- July 1, 2006.
6. ***S. Chivapreecha,*** A. Jaruvarakul and K. Dejhan, "Multi-Functional Digital Filter Based on Non-Recursive Scheme," *Proc. 8th International Conference and Exhibition on Digital Signal Processing and Its Applications (DSPA-2006)*, Moscow, Russia, March 29-31, 2006.
7. ***S. Chivapreecha***, S. Sriyapong, S. Junnapiya and K. Dejhan, "Bilinear s-z with frequency transformation using Pascal matrix operation," *Proc. 2005 International Symposium on Communications and Information Technology (ISCIT 2005)*, Beijing, China, October 12-14, 2005.

Author Biography

Sorawat Chivapreecha was born on 12 September 1977 in Nakhon Pathom. He graduated high school from Suankularb Wittayalai School (OSK 113), Bangkok, and received the B. Eng. degree in Telecommunication Engineering from the Suranaree University of Technology (SUT), Nakhon Ratchasima in 1998 and M. Eng. degree in Electrical Engineering from the King Mongkut's Institute of Technology Ladkrabang (KMITL), Bangkok in 2001. Since 2002, he has been a member of Department of Telecommunication Engineering at Faculty of Engineering, KMITL. During October and November 2005, he was a Thai researcher who received the grant from GISTDA (Geo-Informatics and Space Technology Development Agency), Ministry of Science and Technology for training specialists on "Earth Remote Sensing Space Systems" specialized course at the NPO Mashinostroyenia, Reutov, Moscow, Russian Federation. During April to June 2008, he was a visiting research scholar at Department of Embedded Technology, School of Information Technology and Science, the Tokai University, Japan. His research interests include digital filter design and implementation, VLSI for signal processing, digital system design with FPGA applications and image processing. He is also a member of IEEE and IEICE.