To the Graduate Council:

I am submitting herewith a thesis written by Elizabeth Barnwell Lowry entitled "Discrete Walsh Transform Technology, Applications to Control Systems." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

J. Milton Bailey

J. Milton Bailey, Major Professor

We have read this thesis
and recommend its acceptance:

R. C Gonzalez

J D. Tillman

Accepted for the Council:

L. E. van Vleck

Vice Chancellor
Graduate Studies and Research

A

DISCRETE WALSH TRANSFORM TECHNOLOGY, APPLICATIONS TO CONTROL SYSTEMS

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Elizabeth Barnwell Lowry

June 1982

ACKNOWLEDGEMENTS

I would like to express my deep appreciation to my major professor, Dr. J. Milton Bailey, for his encouragement and help in obtaining my thesis project and to Dr. Joseph M. Googe for his interest and support throughout my graduate studies. Appreciation is also extended to my committee members for their critical reading of the manuscript and their invaluable comments and to Jessica Daugherty for her skillful preparation of the manuscript.

Most of all, I want to thank my parents for their encouragement and unwavering support throughout my academic career.

This thesis was supported by the Department of the Army under contract DASG60-77-C-0062 of the Ballistic Missile Defense Systems Command.

ABSTRACT

This study was motivated by the Department of the Army's desire to increase the efficiency and speed of response of the autopilot system in interceptors by allowing error correction signals to be transmitted and received in the Walsh domain.

The purpose of this study was to obtain a method for designing and implementing a discrete Walsh transform controller in a linear feedback system and to develop the fast Walsh transform algorithm. In particular, it was desired to obtain a value for the output from a physical system that was equal to the input to the autopilot system as time increased.

The method for analyzing this problem was to simulate, on a digital computer, a second-order linear system within the desired Walsh transform controlled feedback system. The Continuous System Modeling Program (CSMP, version III) and FORTRAN were to be used to perform the simulation.

In conclusion, the desired discrete controller was obtained using an integrator as the controller. A workable algorithm was developed. Several runs of the computer program were made and the results achieved were positive. The output from the physical system approached the value of the input function as time increased.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Overview

This study is an outcome of the Army's interest in developing a highly efficient and fast autopilot system for interceptors. Thus, with the advances in digital sensor and autopilot technology it seems feasible to consider utilizing a generic control system as shown in Figure 1.1. A basic scheme will be designed in this manuscript and simulated on a digital computer. In this design the development of the discrete transform controller is of prime importance. The output, OUT, from the physical system in Figure 1.1 is also of great interest. It is desired for this value, OUT, to approach the value of the input signal to the autopilot system. This result will imply that a workable autopilot system has been developed.

1.2. Discrete Transforms

Discrete orthogonal transforms are finding increasing applications in digital processing due to the rapid advances in digital computers and associated hardware. Fast algorithms, such as the Fast Fourier transform (FFT) for efficient computation of these transforms have been developed that result in reduced computational and storage requirements as well as decreased roundoff error. Research activities in this area have focused on applications that include speech processing, analysis and design of communications systems, image enhancement and pattern recognition [1-10]. Attention has also been focused on the application of orthogonal functions to the solution of differential

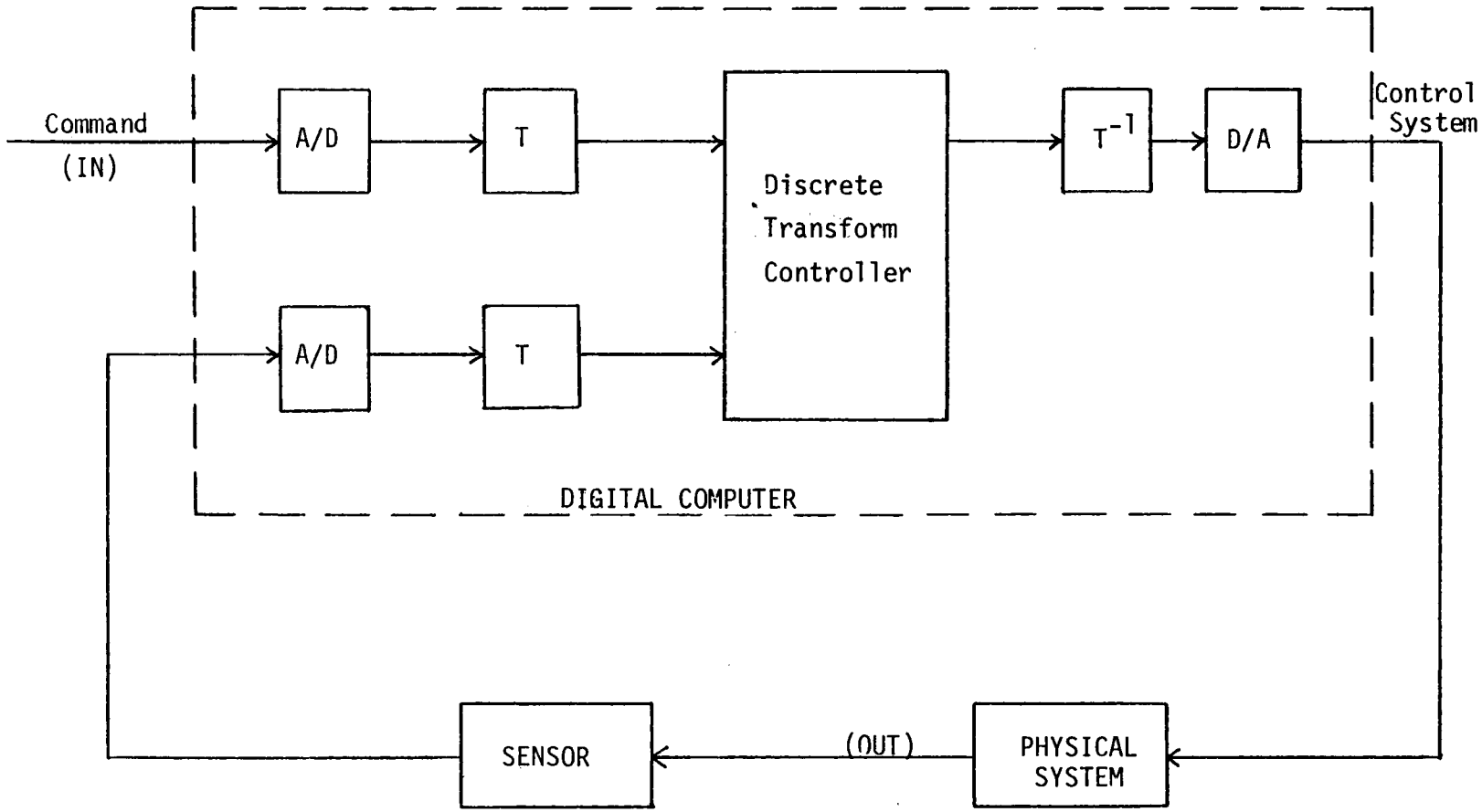

Figure 1.1. The Generic Autopilot System

equations [11-14]. The types of transforms under investigation can be summarized as in Table 1.1 which gives a comparison of the number of operations required to compute the various transforms [1].

Table 1.1

The Number of Arithmetic Operations Required to Compute the Various Transforms

(HT)	Haar	$2(N - 1)$
(CHT)	Complex Haar	$3N - 4$
(DFT)	Discrete Fourier	$N \log_2 N$
(WHT)	Walsh Hadamard	$N \log_2 N$
(ST)	Slant	$N \log_2 N + (2N - 4)$
(DCT)	Discrete cosine	$2N \log_2 (2N)$

In this table the variable N is the number of samples utilized to construct the transform.

Some of the previously mentioned transforms along with the block pulse transform are illustrated in Table 1.2 in both continuous form when applicable and in their discrete form.

Block pulses and Walsh transforms seem particularly suited for the design and analysis of control systems. The three main advantages of these transforms are:

1. These transforms are pulse like in nature and seem suitable for characterizing such digital devices as A/D converters.

2. The ability to efficiently compute these transforms leads to the concept of the discrete transform controller as shown in Figure 1.1. The blocks labeled T represent the algorithms that compute the discrete transform and the blocks labeled T^{-1} denote

Table 1.2

Orthogonal Transforms

Transform	Continuous Form	Discrete Transform Matrix
HARR	$har(0,0,t)\ t \in [0,1]$ $har(r,m,t) = \begin{cases} 2^{r/2}, & \frac{m-1}{2^r} \le t < \frac{m-1/2}{2^r} \\ -2^{r/2}, & \frac{m-1/2}{2^r} \le t < \frac{m}{2^r} \\ 0 \text{ elsewhere for } t \in [0,1] \end{cases}$ where $0 \le r < \log_2 N$ and $1 \le m \le 2^r$.	$\underline{\underline{H}}^*(3) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & \sqrt{2} & \sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & \sqrt{2} & \sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}$ Discrete Haar functions, $N = 8$

Table 1.2 (Cont.)

Transforms	Continuous Form	Discrete Transform Matrix
Block Impulse	$\varphi_i(t) = \begin{cases} 1, & \frac{L-1}{m} \leq t \leq \frac{i}{m} \\ 0 & \text{elsewhere} \end{cases}$	$\phi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$ Discrete Block Pulses, N = 8

Table 1.2 (Cont.)

Transform	Continuous Form	Discrete Transform Matrix
WALSH a) sequence or Walsh ordering $(WHT)_w$	$wal_w(i, t) = cal(s_i, t)$, i = even $wal_w(i, t) = sal(s_i, t)$, i = odd where s_i = $i = 0$ $1/2$ i = even $(i+1)/2$ i = odd	$\underline{\underline{H}}_w(3) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$ ← Wal(0,t), Sal(1,t), Cal(1,t), Sal(1,t), Cal(2,t), Sal(3,t), Cal(3,t), Sal(4,t) Walsh-ordered discrete Walsh functions, N = 8

Table 1.2 (Cont.)

Transform	Countinuous Form	Discrete Transform Matrix
b) dyadic or Paley ordering	$wal_p(i, t) = wal_w[b(i), t]$ where b(i) represents the Gray code-to-binary conversion of i	$\underline{H}_p(3) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$ Paley-ordered discrete Walsh Functions N = 8

Table 1.2 (Cont.)

Transform	Continuous Form	Discrete Transform Matrix
WALSH c) natural or Hadamard ordering $(WHT)_h$ [BIFORE (BT)]	$wal_h(i, t) = wal_w[b\langle i\rangle, t]$ where $\langle i\rangle$ denotes the bit-reversal of i, and $b\langle i\rangle$ represents the Gray code-to-binary conversion of $\langle i\rangle$	$\underline{H}_h(3) =$ $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$ Hadamard-ordered discrete Walsh functions N = 8

Table 1.2 (Cont.)

Transform	Continuous Form	Discrete Transform Matrix
DISCRETE COSINE (DCT)		$L_x(0) = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} X(m)$ $L_x(k) = \sqrt{\frac{2}{N}} \sum_{m=0}^{N-1} X(m)\cos\frac{(2m+1)k}{2N}$ where $k = 1, 2, \ldots, N-1$ and $L_x(k)$ denotes the k-th DCT coefficient

the algorithms that return the signal to the discrete time domain. To design this controller it is necessary to relate the response of a linear system in the discrete transform domain to that in the discrete time domain.

3. Orthogonal functions can be utilized to obtain solutions to differential equations. Such solutions give rise to algebraic methods suitable for system analysis.

This thesis concentrates on the second advantage with emphasis on Walsh transforms. It is shown in the following work that the response of a sampled linear system with an input x_m and output y_m can be represented by the matrix relationship

$$\underline{y} = \underline{\underline{H}}_T \underline{x} + y_o \underline{F} \tag{1.1}$$

The Walsh transform becomes

$$\hat{\underline{y}} = \underline{\underline{T}}\, \underline{\underline{H}}_T \underline{\underline{T}}^{-1}\, \hat{\underline{x}} + y_o \underline{F}_T \tag{1.2}$$

The double underbar indicates a matrix while the single under bar denotes a column matrix.

1.3 Organization

In this thesis the application of Walsh transforms to linear systems theory is discussed and a technique is developed for obtaining $\underline{\underline{H}}_T$ from a given transfer function. Equation 1.2 can be considered as the form of the discrete transform controller shown in Figure 1.1. The application of this technique is illustrated for a second-order system where the transformation matrices T and T^{-1} are implemented by the fast Walsh transform (FWT). The FWT algorithm is developed and the

corresponding subroutinue is presented. A computer program entitled "Continuous System Modeling Program" or "CSMP" for short is used to simulate the desired controller feedback system. Lastly, results of the study are presented and discussed.

CHAPTER II

WALSH TRANSFORMS

2.1 The Walsh Function

The Walsh functions, $Wal(n, t)$, are a set of functions orthonormal in the interval $(0, 1)$ and having only the value $+1$ or -1. The first four functions are shown in Figure 2.1. In general, if $n = 1, 2, 3, 4, \ldots$, $Wal(2^n-1, t)$ is a square wave with 2^{n-1} periods between 0 and 1. The remaining Walsh functions can be found by use of the product formula:

$$Wal(n, t)\, Wal(m, t) = Wal(m \oplus n, t) \qquad (2.1)$$

The symbol $m \oplus n$ stands for addition modulo 2, and m and n are written as binary numbers and added without carry. For example

$$Wal(3, t)\, Wal(7, t) = Wal(4, t)$$

$$n \oplus m \Rightarrow 11 + 111 = 100 \Rightarrow 4$$

A brief look at the orthonormal property necessitates the following definitions:

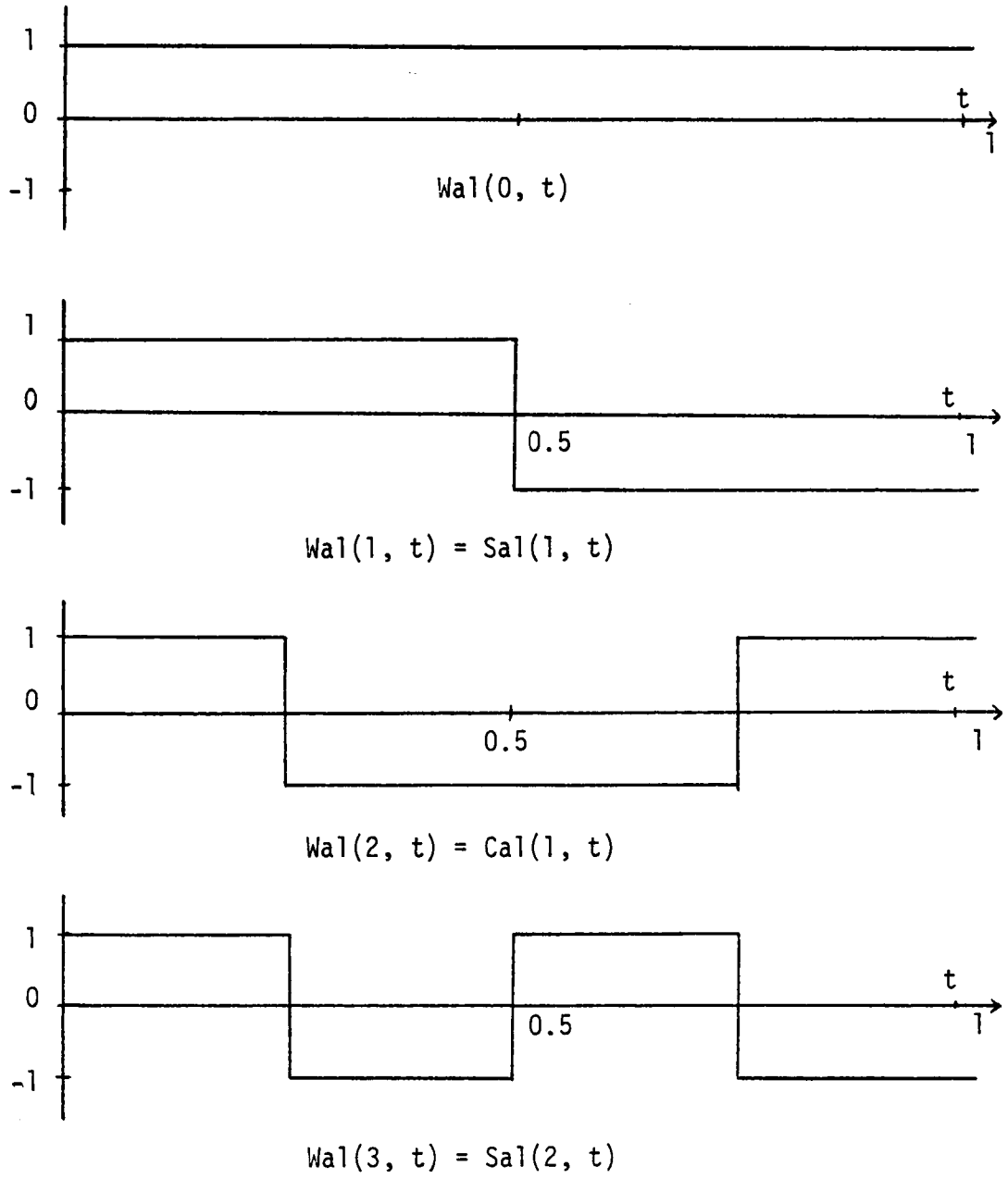

Figure 2.1. The First Four Walsh Functions

$$Wal(0, t) = 1$$

$$\int_0^1 Wal(n, t)dt = 0 \qquad n \neq 0$$

and

$$Wal(m, t)\, Wal(n, t) = Wal(k, t) \ .$$

It follows that

$$\int_0^1 Wal(m, t)\, Wal(n, t)dt = 0 \qquad m \neq n \ . \tag{2.2}$$

Further, since $Wal(n, t)$ is either 0 or 1

$$\int_0^1 Wal(n, t)\, Wal(n, t)dt = \int_0^1 dt = 1 \tag{2.3}$$

and the orthonormal property of the Walsh functions has been demonstrated.

The ordering shown in Figure 2.1 is known as sequency or Walsh order which is the original order for Walsh's function. Sequency is the term used to describe the periodic rate of repetition of the waveform. This rate is independent of the waveform. Sequency is defined as "one half of the average number of zero crossings per unit time interval" [15, p. 13]. As is clearly seen in Figure 2.1 the components are arranged in ascending order of zero crossings, which is

their distinguishing characteristic [15].

2.2 Development of the Walsh Transform

Let $x(t)$ be some function defined in the interval $(0, 1)$. It can be expanded in an infinite series

$$x(t) = \sum_{n=0}^{\infty} a_n \mathrm{Wal}(n, t) \tag{2.4}$$

where

$$a_n = \int_0^1 x(t)\mathrm{Wal}(n, t)dt \tag{2.5}$$

Now let $x(t)$ be sampled at $N = 2^p$ times in $(0, 1)$ so that $N = 2, 4, 8, 16 \ldots$ The times are denoted by $t_0 = 0$, $t_1 = 1/N$, $t_2 = 2/N, \text{---} t_{n-1} = (N-1)/N$ and the respective samples are designated by $x_0, x_1, x_2, \ldots, x_{n-1}$. If the sample is allowed to pass through a zeroth-order hold the resulting signal is a staircase approximation to $x(t)$, which is designated by $x_{SC}(t)$. The case for $N = 4$ is shown in Figure 2.2.

An infinite series is required to represent $x(t)$ by Walsh functions, but the staircase approximation can be expressed exactly by

$$x_{SC}(t) = \sum_{n=0}^{N-1} \hat{x}_n \mathrm{Wal}(n, t) . \tag{2.6}$$

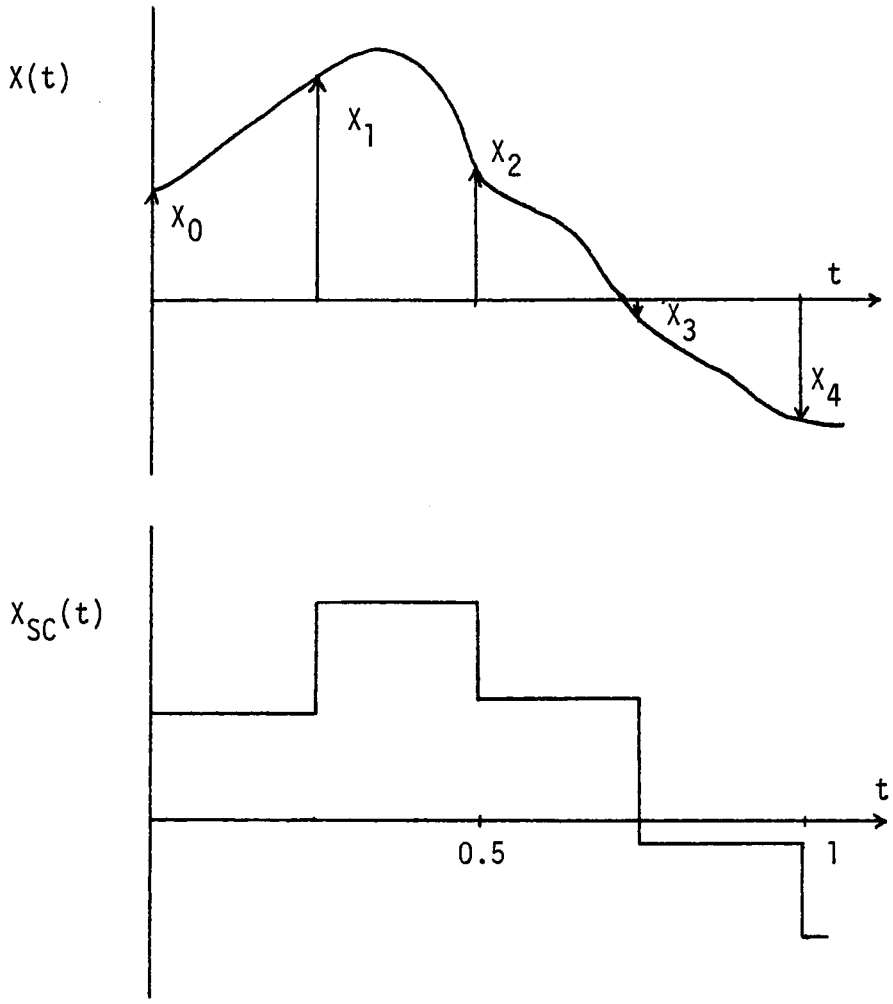

Figure 2.2. Staircase Approximation

The vector $\underline{\hat{X}} = \begin{pmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_n \end{pmatrix}$ is called the Walsh transform of the vector $\underline{X} = \begin{pmatrix} x_0 \\ x_1 \\ x_{n-1} \end{pmatrix}$.

To show this, consider equation 2.6 for $N = 4$.

$$x_{sc}(t) = \hat{x}_0 Wal(0, t) + \hat{x}_1 Wal(1, t) + \hat{x}_2 Wal(2, 5) +$$

$$\hat{x}_3 Wal(3, t)$$

For the interval $0 < t < 1/4$,

$$x_{sc}(t) = x_0, \; Wal(0, t) = 1, \; Wal(1, t) = 1, \; Wal(2, 5) = 1 ,$$

$$Wal(3, t) = 1 ,$$

so

$$x_0 = \hat{x}_0 + \hat{x}_1 + \hat{x}_2 + \hat{x}_3 . \qquad (2.7)$$

In the interval $1/4 < t < 1/2$,

$$x_{sc}(t) = x_1 \ , \ \mathrm{Wal}(0, t) = 1, \ \mathrm{Wal}(1, t) = 1,$$

$$\mathrm{Wal}(2, 5) = -1, \ \mathrm{Wal}(3, t) = -1$$

so

$$x_1 = \hat{x}_0 + \hat{x}_1 - \hat{x}_2 - \hat{x}_3 \ . \tag{2.8}$$

The remaining intervals give

$$x_2 = \hat{x}_0 - \hat{x}_1 - \hat{x}_2 + \hat{x}_3$$

and

$$x_3 = \hat{x}_0 - \hat{x}_1 + \hat{x}_2 - \hat{x}_3 \ .$$

Writing these in matrix form gives

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \end{bmatrix} . \tag{2.9}$$

The square matrix is designated by $\underline{\underline{W}}^{-1}$ and the following equation is obtained

$$\underline{x} = \underline{\underline{W}}^{-1}\hat{\underline{x}} \quad . \tag{2.10}$$

Designating the inverse of $\underline{\underline{W}}^{-1}$ by $\underline{\underline{W}}$, the direct transform is

$$\hat{\underline{x}} = \underline{\underline{W}}\,\underline{x} \quad . \tag{2.11}$$

Noting that

$$\underline{\underline{W}}^{-1}\underline{\underline{W}}^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} = 4I$$

it is seen that

$$\underline{\underline{W}} = \frac{1}{4}\underline{\underline{W}}^{-1} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \quad . \tag{2.12}$$

The situation is very similar to the discrete Fourier transform which exactly represents N samples of the function by a sum of N complex exponentials. The Walsh transform gives an exact representation of the staircase approximation.

Example 2.1

Take as an example a ramp function $f(t)$. The samples are then 0, 1/4, 1/2, 3/4, for $N = 4$, i.e.

$$\underline{x} = \frac{1}{4} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix} .$$

Then the Walsh transform of this input is

$$\hat{\underline{x}} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \left(\frac{1}{4}\right) \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 6 \\ -4 \\ 0 \\ -2 \end{bmatrix} =$$

$$\begin{bmatrix} 3/8 \\ 1/4 \\ 0 \\ -1/8 \end{bmatrix} .$$

The way in which the Walsh functions add up to give the staircase approximation to the ramp is shown in Figure 2.3.

2.3. Application to Linear Systems Theory

2.3.1 The Walsh Domain

Now consider a linear system as shown in Figure 2.4. Given the Walsh transform of $\underline{x}$ the question arises as to the procedure for

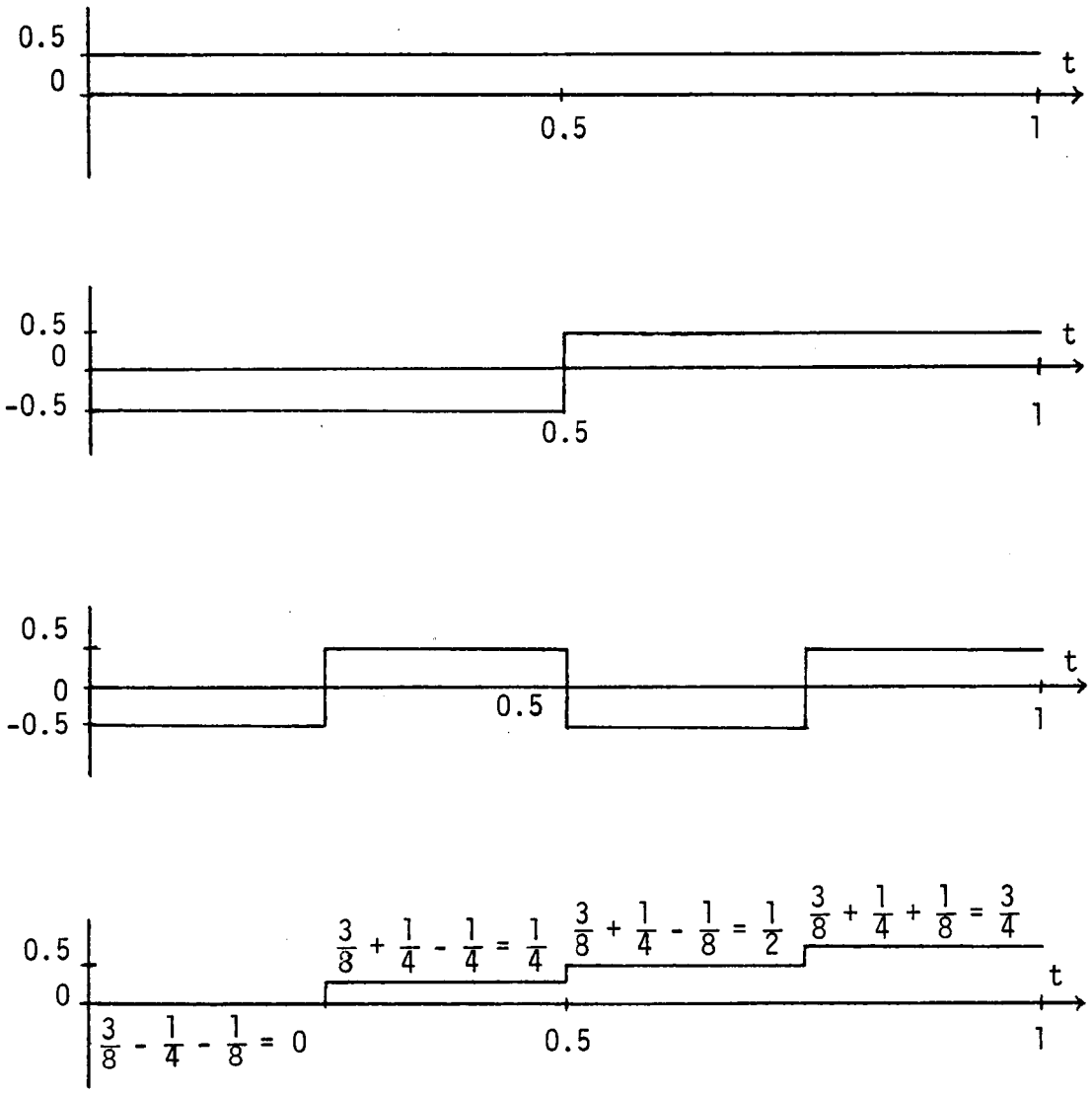

Figure 2.3. Addition of Walsh Functions to Give Sample and Hold Approximation to a Ramp Function.

obtaining the Walsh transform of $\underline{y}$.

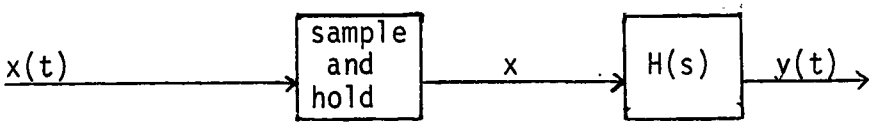

Figure 2.4. A Linear System

In the time domain x and y can be related by

$$\underline{y} = \underline{\underline{H}}_T \underline{x} + y_0 \underline{F}_T \quad . \tag{2.13}$$

Therefore the Walsh transform of $\underline{y}$ is defined as

$$\hat{\underline{y}} = \underline{\underline{W}}\, \underline{y} = \underline{\underline{W}}\, \underline{\underline{H}}_T\, \underline{x} + y_0 \underline{\underline{W}}\, \underline{F}_T \tag{2.14}$$

and since $\underline{x} = \underline{\underline{W}}^{-1}\, \hat{\underline{x}}$,

$$\hat{\underline{y}} = \underline{\underline{W}}\, \underline{\underline{H}}_T\, \underline{\underline{W}}^{-1}\, \hat{\underline{x}} + y_0\, \underline{\underline{W}}\, \underline{F}_T \quad . \tag{2.15}$$

The required version in the Walsh domain is

$$\hat{\underline{y}} = \underline{\underline{H}}_W\, \hat{\underline{x}} + y_0\, \underline{F}_W \tag{2.16}$$

where

$$\underline{\underline{H}}_W = \underline{\underline{W}}\ \underline{\underline{H}}_T\ \underline{\underline{W}}^{-1} \tag{2.17}$$

$$\underline{F}_W = \underline{\underline{W}}\ \underline{F}_T \quad . \tag{2.18}$$

A method must now be developed in order to obtain $\underline{\underline{H}}_T$ from the transfer function $H(s)$ to meet our requirement. How this can be done is illustrated for $H(s) = K/(s + a)$. More complicated cases can be handled with the aid of a partial fraction expansion. For simplicity, $N = 4$ in the remaining work of this chapter so all the matrices are no more than 4×4 .

2.3.2 The Runge-Fox Recurrence Formula

First consider a linear first-order differential equation

$$\frac{dy}{dy} + a(t)\ y = x(t) \quad . \tag{2.19}$$

This equation must hold for any t , including $t_i = h_i$, and thus can be rewritten in the following equivalent forms

$$y_i' + a_i y_i = x_i \tag{2.20}$$

$$y_{i+1}' + a_{i+1}\ y_{i+1} = x_{i+1} \tag{2.21}$$

Adding and dividing by 2,

$$\frac{y'_{i+1} + y'_i}{2} = \frac{x_i}{2} + \frac{x_{i+1}}{2} - \frac{a_i}{2} y_i - \frac{a_{i+1}}{2} y_{i+1} \quad . \tag{2.22}$$

This is the average of y' at $t_i + h$, hence the definition of derivative gives

$$\frac{y'_{i+1} - y'_i}{2} \approx \frac{y_{i+1} - y_i}{h} \quad . \tag{2.23}$$

Substituting equation 2.23 into 2.22 gives

$$\frac{y_{i+1} - y_i}{h} = \frac{x_i}{2} + \frac{x_{i+1}}{2} - \frac{a_i}{2} y_i - \frac{a_{i+1}}{2} y_{i+1} \quad . \tag{2.24}$$

Solving for y_{i+1}:

$$y_{i+1}\left(1 + \frac{ha_{i+1}}{2}\right) = \left(1 - \frac{ha_i}{2}\right) y_i + \frac{h}{2} x_i + \frac{h}{2} x_{i+1} \tag{2.25}$$

and

$$y_{i+1} = \frac{1}{1+\frac{ha_{i+1}}{2}} \left[\left(1 - \frac{ha_i}{2}\right)y_i + \frac{h}{2} x_i + \frac{h}{2} x_{i+1}\right] \quad . \tag{2.26}$$

This is the Runge-Fox recurrence formula.

2.3.3 The $\underline{F}_T$ and $\underline{\underline{H}}_T$ Matrices

The Runge-Fox recurrence formula is now used to obtain

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \underline{\underline{H}}_T \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} + y_0 \underline{F}_T \qquad (2.27)$$

for $H(s) = 1/s+a$.

Here "a" is a constant so that the Runge-Fox formula is, for $h = 1/4$,

$$y_{i+1} = \frac{8-a}{8+a} y_i + \frac{1}{8+a} x_i + \frac{1}{8+a} x_{i+1} \quad .$$

Since the system is linear each of the five inputs y_0, x_0, x_1, x_2 and x_3 can be applied separately.

$$\left.\begin{aligned} y_1 &= \frac{8-a}{8+a} y_0 \\ y_2 &= \frac{8-a}{8+a} y_1 = \left(\frac{8-a}{8+a}\right)^2 y_0 \\ y_3 &= \frac{8-a}{8+a} y_2 = \left(\frac{8-a}{8+a}\right)^3 y_0 \end{aligned}\right\} \qquad x_i = 0$$

From this set of equations it is observed that

$$\underline{F}_T = \begin{bmatrix} 1 \\ \frac{8-a}{8+a} \\ \left(\frac{8-a}{8+a}\right)^2 \\ \left(\frac{8-a}{8+a}\right)^3 \end{bmatrix} . \qquad (2.28)$$

Since y_0 is not a function of any of the x's, the first row of $\underline{H}_T$ is zero. Next consider y_1. By causality, y_1 can depend only on x_0 and x_1. Also, y_2 depends only on x_0, x_1 and x_2 and y_3 on all of the elements above the main diagonal equal to zero. For $x_0 \neq 0$

$$\left.\begin{aligned} y_0 &= 0 \\ y_1 &= \frac{1}{8+a} x_0 \\ y_2 &= \frac{8-a}{8+a} y_0 = \frac{8-a}{(8+a)^2} x_0 \\ y_2 &= \frac{8-a}{8+a} y_2 = \frac{(8-a)^2}{(8+a)^3} x_0 \end{aligned}\right\} \quad \begin{aligned} y_0 &= 0 \\ x_i &= 0, \quad i \neq 0 \end{aligned}$$

These equations form the first column of $\underline{H}_T$.

The second column of $\underline{\underline{H}}_T$ contains the contributions of x_1 to the y's . For only $x_1 \neq 0$

$$\left.\begin{aligned} y_0 &= 0 \\ y_1 &= \frac{1}{8+a} x_1 \\ y_2 &= \frac{8-a}{8+a} y_1 + \frac{1}{8+a} x_1 \\ &= \left[\frac{8-a}{8+a}\left(\frac{1}{8+a}\right) + \frac{1}{8+a}\right] x_1 \\ &= \frac{16}{(8+a)^2} x_1 \\ y_3 &= \frac{8-a}{8+a} y_2 = \frac{16(8-a)}{(8+a)^3} x_1 \end{aligned}\right\} \quad \begin{aligned} y_0 &= 0 \\ x_i &= 0 , \quad i \neq 1 \end{aligned}$$

The third column contains the contributions of x_2 . For only $x_2 \neq 0$

$$\left.\begin{aligned} y_0 &= 0 \\ y_1 &= 0 \\ y_2 &= \frac{1}{8+a} x_2 \\ y_3 &= \frac{8-a}{8+a} y_2 + \frac{1}{8+a} x_2 \\ &= \frac{16}{(8+a)^2} \end{aligned}\right\} \quad \begin{aligned} y_0 &= 0 \\ x_i &= 0 , \quad i \neq 2 \end{aligned}$$

The last column comes from $x_3 \neq 0$.

$$\left.\begin{aligned} y_0 &= 0 \\ y_1 &= 0 \\ y_2 &= 0 \\ y_3 &= \frac{1}{8+a}\, x_3 \end{aligned}\right\} \qquad \begin{aligned} y_0 &= 0 \\ x_i &= 0 \,, \quad i \neq 3 \end{aligned}$$

The complex matrix equation is then

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{8+a} & \frac{1}{8+a} & 0 & 0 \\ \frac{8-a}{(8+a)^2} & \frac{16}{(8+a)^2} & \frac{1}{(8+a)} & 0 \\ \frac{(8-a)^2}{(8+a)^3} & \frac{16(8-a)}{(8+a)^3} & \frac{16}{(8+a)} & \frac{1}{8+a} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$+ y_0 \begin{bmatrix} 1 \\ \frac{8-a}{8+a} \\ (\frac{8-a}{8+a})^2 \\ (\frac{8-a}{8+a})^3 \end{bmatrix} . \qquad (2.29)$$

2.3.4 Extension to Several Periods

The notation found in Table 2.1 will be used to extend the value of $\underline{\underline{H}}_T$ to several periods.

Table 2.1

Notation Used to Extend Value of $\underline{\underline{H}}_T$ to Several Periods

t	x	y
0	x_{10}	$y_{10} = y_0$
0.25	x_{11}	y_{11}
0.50	x_{12}	y_{12}
0.75	x_{13}	y_{13}
1.00	x_{20}	y_{20}
1.25	x_{21}	y_{21}
1.50	x_{22}	y_{22}
1.75	x_{23}	y_{23}
2.00	x_{31}	y_{31}
2.25	x_{32}	y_{32}
----	----	----
$n-1+h_i$	x_{ri}	y_{ni}

The values of x_{10}, x_{11}, x_{12}, x_{13} are available only as components of $\underline{x}_1$, and x_{20}, x_{21}, x_{22}, x_{23} as components of $\underline{x}_2$. It is evident that the first cycle differs from those that follow. In the first cycle, y_0 is specified, and does not depend on x_{10}. In the second, y_{20} depends on y_{13} and on x_{20}, and x_{20} is not available

until $\underline{x}_2$ is given. From the recurrence relation,

$$y_{20} = \frac{8-a}{8+a} y_{13} + \frac{1}{8+a} x_{13} + \frac{1}{8+a} x_{20} \quad .$$

Designating y_{14} as

$$y_{14} = \frac{8-a}{8+a} y_{13} + \frac{1}{8+a} x_{13}$$

allows y_{14} to be computed before $\underline{x}_2$ is given. Then

$$y_{20} = y_{14} + \frac{1}{8+a} x_{20} \quad .$$

y_{14} replaces y_0 of Section 2.3. The first row of $\underline{\underline{H}}_T$ is now

$$\frac{1}{8+a} \quad 0 \quad 0 \quad 0$$

Proceeding as before, with only x_{20} as the input the following equations are obtained

$$y_{20} = \frac{1}{8+a} x_{20}$$

$$y_{21} = \frac{8-a}{8+a} y_{20} + \frac{1}{8+a} x_{20}$$

$$= \frac{16}{(8+a)^2} x_{20}$$

$$y_{22} = \frac{8-a}{8+a} y_{21} = \frac{16(8-a)}{(8+a)^3} x_{20}$$

$$y_{23} = \frac{8-a}{8+a} y_{22} = \frac{16(8-a)^2}{(8+a)^4} x_{20} \quad .$$

These equations give the first column of $\underline{\underline{H}}_T$.

Similarily, the remaining columns of $\underline{\underline{H}}_T$ can be obtained and the complete $\underline{\underline{H}}_T$ is found to be

$$\underline{\underline{H}}_T = \begin{bmatrix} \frac{1}{8+a} & 0 & 0 & 0 \\ \frac{16}{(8+a)^2} & \frac{1}{8+a} & 0 & 0 \\ \frac{16(8-a)}{(8+a)^3} & \frac{16}{(8+a)^2} & \frac{1}{8+a} & 0 \\ \frac{16(8-a)^2}{(8+a)^4} & \frac{16(8-a)}{(8+a)^3} & \frac{16}{(8+a)^2} & \frac{1}{8+a} \end{bmatrix} \qquad (2.30)$$

It is noted that $\underline{F}_T$ given in equation 2.28 is unchanged.

The original $\underline{\underline{H}}_T$ matrix developed in Section 2.3.3 is now designated as $\underline{\underline{H}}_{T1}$. It is different from the $\underline{\underline{H}}_T$ matrix found for each of the succeeding periods in that the next value of y depends only on the previous value and not on the present value. For all the cycles after the first, the matrix is denoted as $\underline{\underline{H}}_T$ and

$$\underline{y}_2 = \underline{\underline{H}}_T \underline{x}_2 + y_{14}\,\underline{F} \quad . \tag{2.31}$$

For $a = 1$,

$$\underline{F} = \begin{bmatrix} 1 \\ 0.7778 \\ 0.6049 \\ 0.4705 \end{bmatrix} \tag{2.32}$$

$$\underline{\underline{H}}_{T1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.1111 & 0.1111 & 0 & 0 \\ 0.0864 & 0.1975 & 0.1111 & 0 \\ 0.0672 & 0.1536 & 0.1975 & 0.1111 \end{bmatrix} \tag{2.33}$$

and

$$\underline{\underline{H}}_T = \begin{bmatrix} 0.1111 & 0 & 0 & 0 \\ 0.1975 & 0.1111 & 0 & 0 \\ 0.1536 & 0.1975 & 0.1111 & 0 \\ 0.1195 & 0.1536 & 0.1975 & 0.1111 \end{bmatrix} \tag{2.34}$$

where

$$y_{14} = 0.7778 y_{13} + 0.1111 x_{13} \quad . \tag{2.35}$$

In general,

$$\underline{y}_{n+1} = \underline{\underline{H}}_T \underline{x}_n + y_{n4} \underline{F} \tag{2.36}$$

and

$$y_{n4} = 0.7778 y_{n3} + 0.1111 x_{n3} \ . \tag{2.37}$$

The flow chart for computing $\underline{y}_n$ is shown in Figure 2.5. Block A represents the algorithm for computing $\underline{\underline{H}}_{T1}$ and block B gives the algorithm for computing each of the cycles after the first.

2.3.5 Examples

To illustrate the matrix method developed above, the following two examples are given.

Example 2.1

Let $x(t)$ be a function defined as

$$x(t) = u(t)$$

having the initial condition

$$y_0 = 0 \ .$$

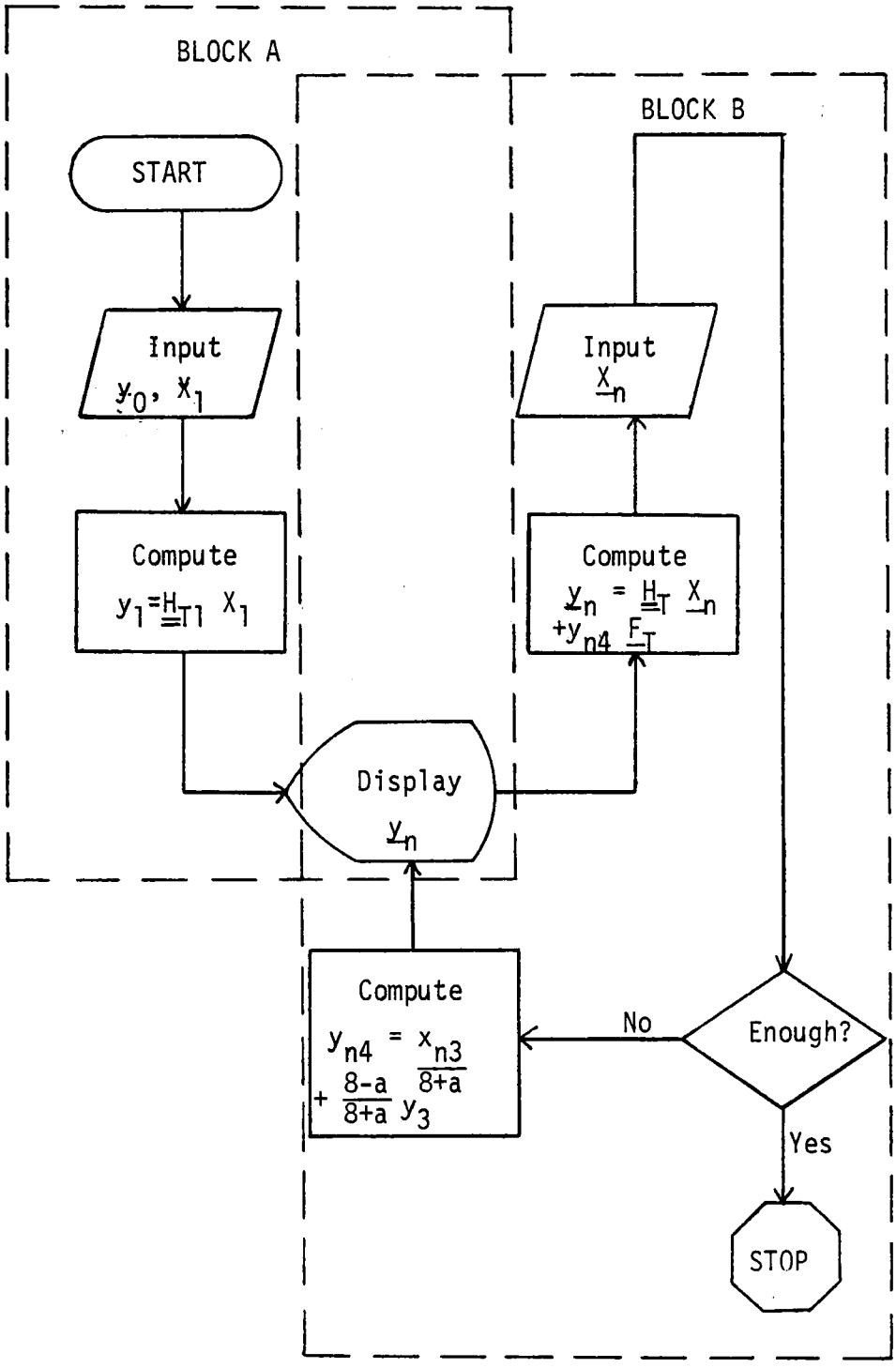

Figure 2.5. Flow Chart to Compute $\underline{y}_n$

The analytic representation of the output is

$$y_{anal} = 1 - e^{-t} \quad .$$

Taking four samples at a time, the unit step input in matrix form is

$$\underline{x}_n = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \text{for all} \quad n \ .$$

Table 2.2 gives the values obtained, using equations 2.32 through 2.37, throughout each cycle represented in the flow chart of Figure 2.5. Only the values for three cycles are shown. The succeeding cycle values can be calculated in a like manner.

Example 2.2

Letting $x(t)$ be a ramp input function

$$x(t) = t$$

with the initial condition

$$y_0 = 0$$

Table 2.2

Computation of y_n for Example 2.1, n = 3

t	n	y_{n4}	$\underline{\underline{H}}_T x$	$y\underline{F}_T$	$\underline{y}$	y_{anal}	% error
0	1	0	0	0	0	0	—
0.25			0.2222	0	0.2222	0.2212	0.45
0.50			0.3950	0	0.3950	0.3935	0.38
0.75		0.5229	0.5294	0	0.5294	0.5276	0.34
1.00	2		0.1111	0.5229	0.6340	0.6321	0.30
1.25			0.3086	0.4067	0.7153	0.7135	0.25
1.50			0.4622	0.3163	0.7785	0.7769	0.21
1.75		0.7549	0.5817	0.2460	0.8277	0.8262	0.18
2.00	3		0.1111	0.7549	0.8660	0.8647	0.15
2.25			0.3086	0.5872	0.8958	0.8946	0.13
2.50			0.4622	0.4569	0.9189	0.9179	0.11
2.75		0.8398	0.5817	0.3552	0.9369	0.9361	0.08

gives the cycle values found in Table 2.3. The analytic solution is

$$y_{anal} = t - 1 + e^{-t} \quad .$$

All the values seen in Table 2.3 have been calculated exactly as those found in Table 2.2.

The author will concentrate on that portion of the flow chart of Figure 2.5 seen in block B. In other words, the matrix of equation 2.30 will be used in determining the desired discrete Walsh transform controller for the given autopilot system.

Table 2.3

Computation of y_n for Example 2.2, n = 2

t	n	y_{n4}	$\underline{\underline{H}}_T x$	$y\underline{F}_T$	$\underline{y}$	y_{anal}	% error
0	1	0	0	0	0	0	——
0.25			0.0278	0	0.0278	0.0288	-3.47
0.50			0.1049	0	0.1049	0.1065	-1.50
0.75		0.2548	0.2205	0	0.2205	0.2224	-0.85
1.00	2		0.1111	0.2548	0.3659	0.3679	-0.54
1.25			0.3364	0.1982	0.5346	0.5365	-0.35
1.50			0.5671	0.1541	0.7212	0.7231	-0.26
1.75		0.8182	0.8022	0.1199	0.9221	0.9238	-0.18

CHAPTER III

THE FAST WALSH-ORDERED WALSH TRANSFORM ALGORITHM

3.1 The Fast Walsh-ordered Walsh Transform

Computation of the discrete Walsh transform (WT) requires N^2 additions and subtractions. This number of calculations is greatly reduced when the WT is computed by means of matrix multiplication. Using matrix multiplication, an algorithm can be found for a set of input samples in $N \log_2 N$ additions and subtractions [15]. This algorithm is known as the fast Walsh transform (FWT). In particular, this thesis will concentrate on the fast Walsh-ordered Walsh Transform $(FWT)_w$. The subscript "w" denotes the sequency or Walsh ordering of the set of Walsh functions.

3.2 The Algorithm

Computation for the $(FWT)_w$ algorithm can be described by means of a signal flow diagram. This diagram consists of a series of nodes representing a variable which is expressed as the sum of two other variables starting from the left of the diagram. Figure 3.1 shows a signal flow graph for a $(FWT)_w$.

The $(FWT)_w$ algorithm begins by reordering the input samples, $X'(m)$, by a method called "bit reversal" [16]. The bit-reversed ordered inputs, $X(m)$ are sent through $\log_2 N$ iterations of additions and subtractions. The appropriate operations are indicated in Figure 3.1. The solid lines on the signal flow graph indicate the calcualted value is to be carried forward to the addition at

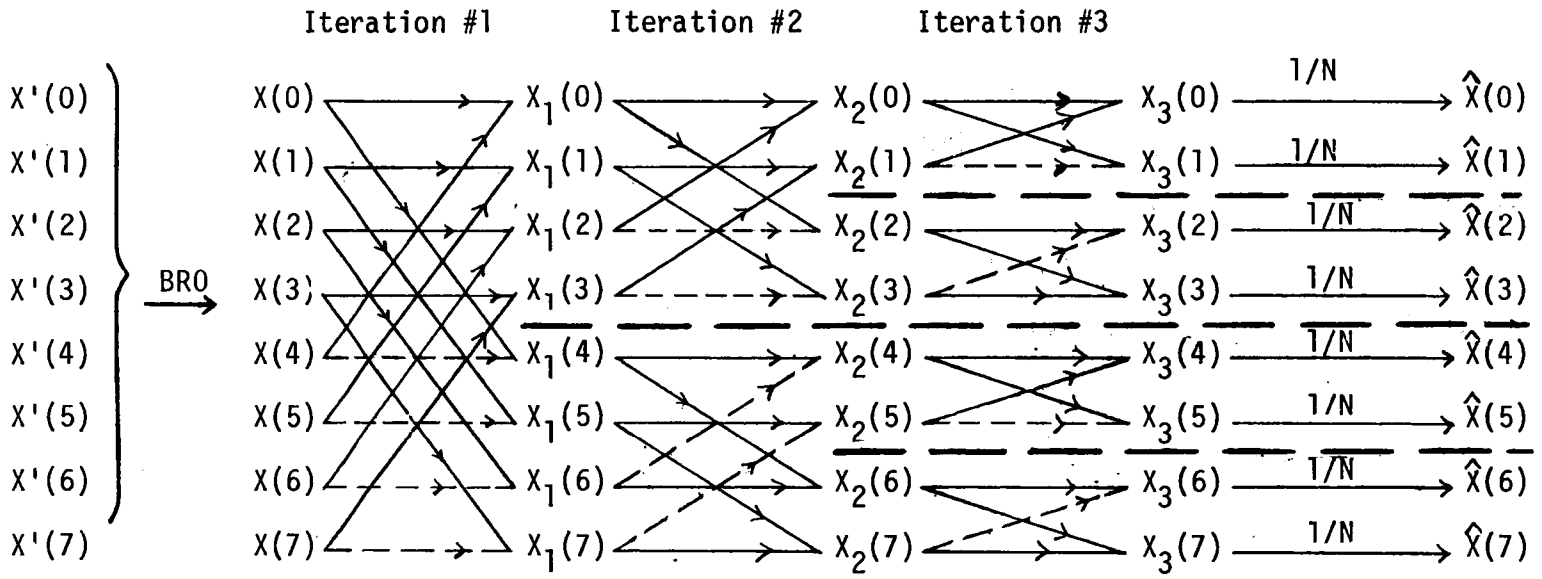

Figure 3.1. $(FWT)_w$ Signal Flow Graph for $N = 8$. The dashed lines denote multiplication by -1 before adding takes place. The heavy broken lines separate blocks.

the next node with the same sign and a dashed line represents multiplication by -1 prior to addition. A final step of multiplying by 1/N is necessary to obtain the desired Walsh transform. The inverse Walsh transform is obtained by the same procedure as above with multiplication by 1/N in the final step eliminated.

To further explain this signal flow diagram method, three terms (bit-reversed order, block, and reveral) are defined, diagrams are given for visual aid, and two examples are included to illustrate the entire approach.

3.2.1 The Bit-Reversal Process

The first step is to reorder the input samples, X'(m), by a method called "bit-reversal". This process reverses the order of the bits in binary representation, putting them in bit-reversed order (BRO) [16, p. 112]. This is exemplified for N = 8 in Table 3.1.

Table 3.1

Illustration of Bit-Reversal

m	m_{binary}	$m_{BRO(binary)}$	m_{BRO}
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

For convenience, the inputs will be put in ascending index order. If the input data sequence is given as $\{X'(m)\} = \{X'(0)\ X'(1)\ \ldots\ X'(N-1)\}$, then the bit reversed sequence is represented by $\{X(m)\} = \{X(0)\ X(1)\ \ldots\ X(N-1)\}$. From Table 3.1 it is observed that the input samples after bit-reversal take on the values as seen in Table 3.2.

Table 3.2

Value of the Input Samples After Bit-Reversal

BRO Samples		Initial Input Samples
X(0)	=	X'(0)
X(1)	=	X'(4)
X(2)	=	X'(2)
X(3)	=	X'(6)
X(4)	=	X'(1)
X(5)	=	X'(5)
X(6)	=	X'(3)
X(7)	=	X'(7)

3.2.2 Definition of a Block

The second step is to define a "block". A designated group of additions/subtractions is called a block. The blocks in Figure 3.1 are separated by heavy broken lines. Iteration #1 has $2^0 = 1$ block; iteration # 2 has $2^1 = 2$ blocks, etc. In general, iteration #n has 2^{n-1} blocks [16]. For $N = 8$, the iterations and blocks are shown in Figure 3.2.

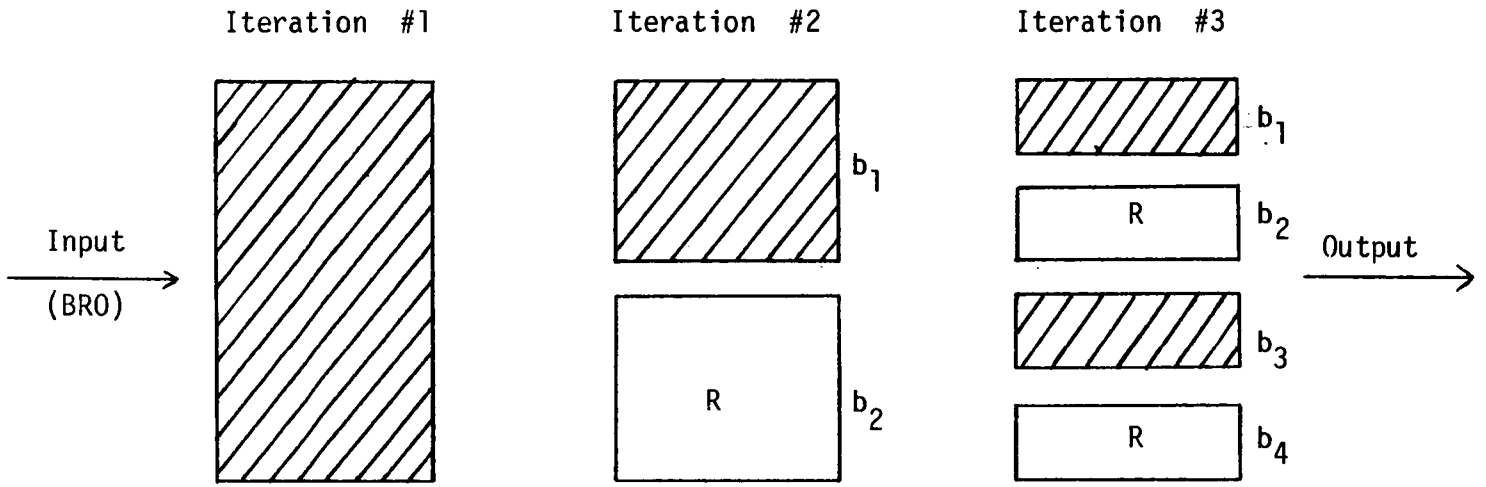

Figure 3.2. Block and Iteration Positions in the $(FWT)_w$ Flow Graph for $N = 8$. BRO means bit-reversed order, $b_1, b_2, \ldots, b_{2^{n-1}}$ designates the number of blocks in the n-th iteration, R denotes a block with reversal.

The number of iterations, n, is determined from the following formula:

$$n = \log_2 N \ .$$

3.2.3 Definition of Reversal

The third step is to define a "reversal". In the following discussion it can be seen that a reversal causes the addtions in the appropriate block of an iteration of a signal flow graph to be replaced by subtractions, and subtractions replaced by additions. Without reversal the values of the next iteration can be determined using the following equations [16]:

$$X_{n+1}(m) = X_n(m) + X_n(m + N/2^n) \qquad \text{additions} \qquad (3.2)$$

and

$$X_{n+1}(m + N/2) = X_n(m) - X_n(m + N/2^n) \qquad \text{subtractions} \qquad (3.3)$$

With a reversal we have:

$$X_{n+1}(m) = X_n(m) - X_n(m + N/2^n) \qquad \text{subtractions} \qquad (3.4)$$

and

$$X_{n+1}(m + N/2) = X_n(m) + X_n(m + N/2^n) \qquad \text{additions} \qquad (3.5)$$

For example, a block with reversal and the same block without reversal is shown in Figure 3.3.

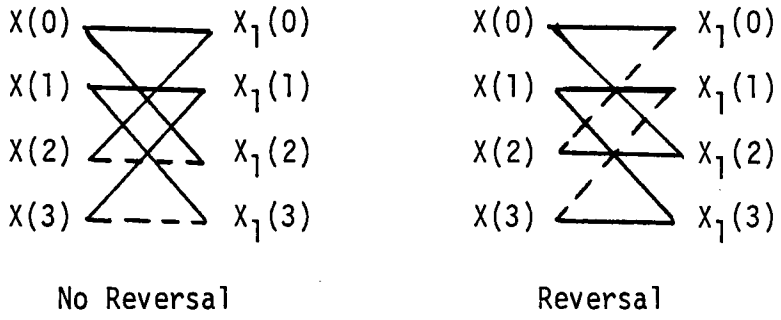

Figure 3.3. The Effect of a Reversal

A couple of simple rules are used to determine where reversals should occur on the signal flow graph. They are listed below [16, p. 114].

Rule 1. Iteration #1 contains no reversals.

Rule 2. If b_ℓ , $\ell = 1, 2, \ldots, 2^{n-1}$ denote the blocks in the n-th iteration, then the blocks with reversals are those whose index is a multiple of 2 . In other words, every other block experiences a reversal starting with block b_2 (see Figure 3.2).

3.2.4 Examples

The entire approach of the $(FWT)_W$ algorithm is now illustrated with the following two examples.

Example 3.1

Use the $(FWT)_W$ to find the Walsh transform of $\{X'(m)\} = \{1\ 1\ 2\ 1\ 3\ 2\ 1\ 2\}$.

Solution: If $\{X(m)\} = \{X(0)\ X(1)\ X(2)\ X(3)\ X(4)\ X(5)\ X(6)\ X(7)\}$ denotes the sequence obtained by the bit-reversal of $\{X'(m)\}$, then we have:

$X(0) = X'(0) = 1$ $\qquad$ $X(4) = X'(1) = 1$

$X(1) = X'(4) = 3$ $\qquad$ $X(5) = X'(5) = 2$

$X(2) = X'(2) = 2$ $\qquad$ $X(6) = X'(3) = 1$

$X(3) = X'(6) = 1$ $\qquad$ $X(7) = X'(7) = 2$

Substituting for $\{X'(m)\}$, $\{X(m)\}$ and $N = 8$ in Figure 3.1 we obtain Table 3.3. Therefore $\{\hat{X}(m)\} = \{\frac{13}{8}\ \frac{-3}{8}\ \frac{-3}{8}\ \frac{1}{8}\ \frac{1}{8}\ \frac{-3}{8}\ \frac{1}{8}\ \frac{1}{8}\}$. These values of $\hat{X}(n)$ are the same as those obtained by using the matrix multiplication developed in Chapter II.

Since the researcher is presently working with the case for $N = 4$, the following example is given.

Example 3.2

Use the $(FWT)_W$ to find the Walsh transform of $\{X'(m)\} = \{1\ 3\ 2\ 1\}$.

Table 3.3

$(FWT)_W$ Signal Flow Graph Values for Example 3.1

$X'(0) = 1$	$X(0) = 1$	$X_1(0) = 2$	$X_2(0) = 5$	$X_3(0) = 13$	$\xrightarrow{1/8}$	$\hat{X}(0)$
$X'(1) = 1$	$X(1) = 3$	$X_1(1) = 5$	$X_2(1) = 8$	$X_3(1) = -3$	$\xrightarrow{1/8}$	$\hat{X}(1)$
$X'(2) = 2$	$X(2) = 2$	$X_1(2) = 3$	$X_2(2) = -1$	$X_3(2) = -3$	$\xrightarrow{1/8}$	$\hat{X}(2)$
$X'(3) = 1$	$X(3) = 1$	$X_1(3) = 3$	$X_2(3) = 2$	$X_3(3) = 1$	$\xrightarrow{1/8}$	$\hat{X}(3)$
$X'(4) = 3$	$X(4) = 1$	$X_1(4) = 0$	$X_2(4) = -1$	$X_3(4) = 1$	$\xrightarrow{1/8}$	$\hat{X}(4)$
$X'(5) = 2$	$X(5) = 2$	$X_1(5) = 1$	$X_2(5) = 2$	$X_3(5) = -3$	$\xrightarrow{1/8}$	$\hat{X}(5)$
$X'(6) = 1$	$X(6) = 1$	$X_1(6) = 1$	$X_2(6) = 1$	$X_3(6) = 1$	$\xrightarrow{1/8}$	$\hat{X}(6)$
$X'(7) = 2$	$X(7) = 2$	$X_1(7) = -1$	$X_2(7) = 0$	$X_3(7) = 1$	$\xrightarrow{1/8}$	$\hat{X}(7)$

Solution: If $\{X(m)\} = \{X(0)\ X(1)\ X(2)\ X(3)\}$ denotes the sequence obtained by the bit-reversal of $\{X'(m)\}$, then we obtain:

$$X(0) = X'(0) = 1$$
$$X(1) = X'(2) = 2$$
$$X(2) = X'(1) = 3$$
$$X(3) = X'(3) = 1$$

Substituting $\{X'(m)\}$ and $\{X(m)\}$ in Figure 3.4, we obtain:

$$X'(0) = 1 \quad X(0) = 1 \quad X_1(0) = 4 \quad X_2(0) = 7 \xrightarrow{1/4} \hat{X}(0)$$
$$X'(1) = 3 \quad X(1) = 2 \quad X_1(1) = 3 \quad X_2(1) = 1 \xrightarrow{1/4} \hat{X}(1)$$
$$X'(2) = 2 \quad X(2) = 3 \quad X_1(2) = -2 \quad X_2(2) = -3 \xrightarrow{1/4} \hat{X}(2)$$
$$X'(3) = 1 \quad X(3) = 1 \quad X_1(3) = 1 \quad X_2(3) = -1 \xrightarrow{1/4} \hat{X}(3)$$

Thus $\{\hat{X}(n)\} = \{\frac{7}{4}\ \frac{1}{4}\ \frac{-3}{4}\ \frac{-1}{4}\}$. These same values of $\hat{X}(n)$ can be obtained by using the matrix multiplication presented in Chapter II.

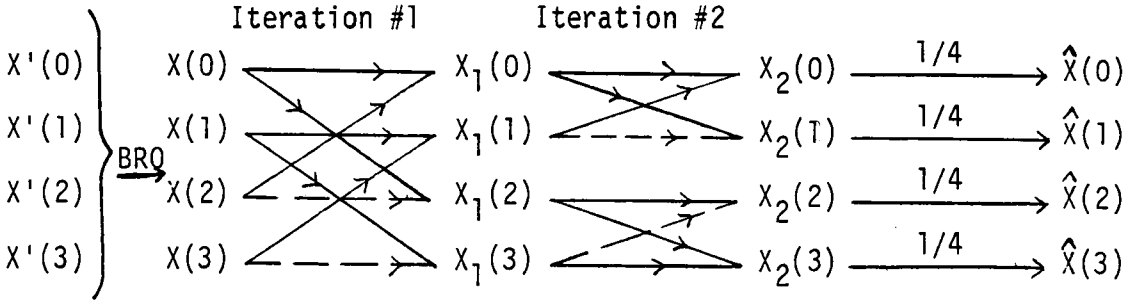

Figure 3.4. $(FWT)_w$ Signal Flow Graph, $N = 4$.

3.3 The $(FWT)_W$ Subroutine

A listing of a subroutine for computing the Walsh-ordered Walsh transform using the fast algorithms developed in Section 3.2 is presented in Appendix A. This program was adapted from that listed in Reference 16, pages 142-43. The subroutine is written in FORTRAN and will be inserted within a larger program implementing CSMP in order to simulate the desired second-order controlled feedback system.

CHAPTER IV

COMPUTER MODELING USING CSMP

4.1 The CSMP Code

The CSMP program is used to solve a system of ordinary differential equations or analog block diagrams like those encountered in systems theory. The advantages of the CSMP digitial simulation over the use of an analog machine are that [17, p. 2]:

1. it is easier to program,
2. it gives more accurate results,
3. it has more potential in handling nonlinear and time-variant problems, and
4. the user needs not be concerned with amplitude and time-scaling.

4.2 The Discrete Transform Controller

Now that $\underline{\underline{H}}_T$ has been obtained as shown in Section 2.3.4, the discrete transform controller of Figure 1.1 on page 2 is known. For the second-order control system to be implemented in this study the desired discrete transform controller is the Walsh transform of an integrator. For integration, "a" is equal to zero in equation 2.30. The elements of $\underline{\underline{H}}_T$ then become

$$\underline{\underline{H}}_T = \begin{bmatrix} 0.125 & 0 & 0 & 0 \\ 0.25 & 0.125 & 0 & 0 \\ 0.25 & 0.25 & 0.125 & 0 \\ 0.25 & 0.25 & 0.25 & 0.125 \end{bmatrix} \qquad (4.1)$$

and the discrete transform controller takes the form

$$\underline{\underline{WH_TW^{-1}}} = \begin{bmatrix} 0.5 & 0.25 & 0 & 0.125 \\ -0.25 & 0 & 0.125 & 0 \\ 0 & -0.125 & 0 & 0 \\ -0.125 & 0 & 0 & 0 \end{bmatrix} \quad (4.2)$$

4.3 The Control Feedback System

The complete CSMP program developed is a simulation of the control system shown in Figure 4.1. The W block performs the $(FWT)_W$ and the W^{-1} block performs the inverse $(FWT)_W$. The discrete transform controller is designated by the WH_TW^{-1} block and is simulated using matrix multiplication.

The physical system of Figure 1.1 is shown in Figure 4.1 as a second-order system represented by the transfer function

$$H(s) = \frac{K1}{(s+1)\ (0.25s + 1)}$$

where K1 is a gain constant. Higher order systems can be implemented with only minor changes in the CSMP simulation found in this manuscript.

From Figure 4.1 it is seen that the output from the physical system, OUT, and the input signal, IN, are transformed into their Walsh counterparts (WALOUT and WALIN, respectively) prior to summing. This summation gives the desired error signal, E ,

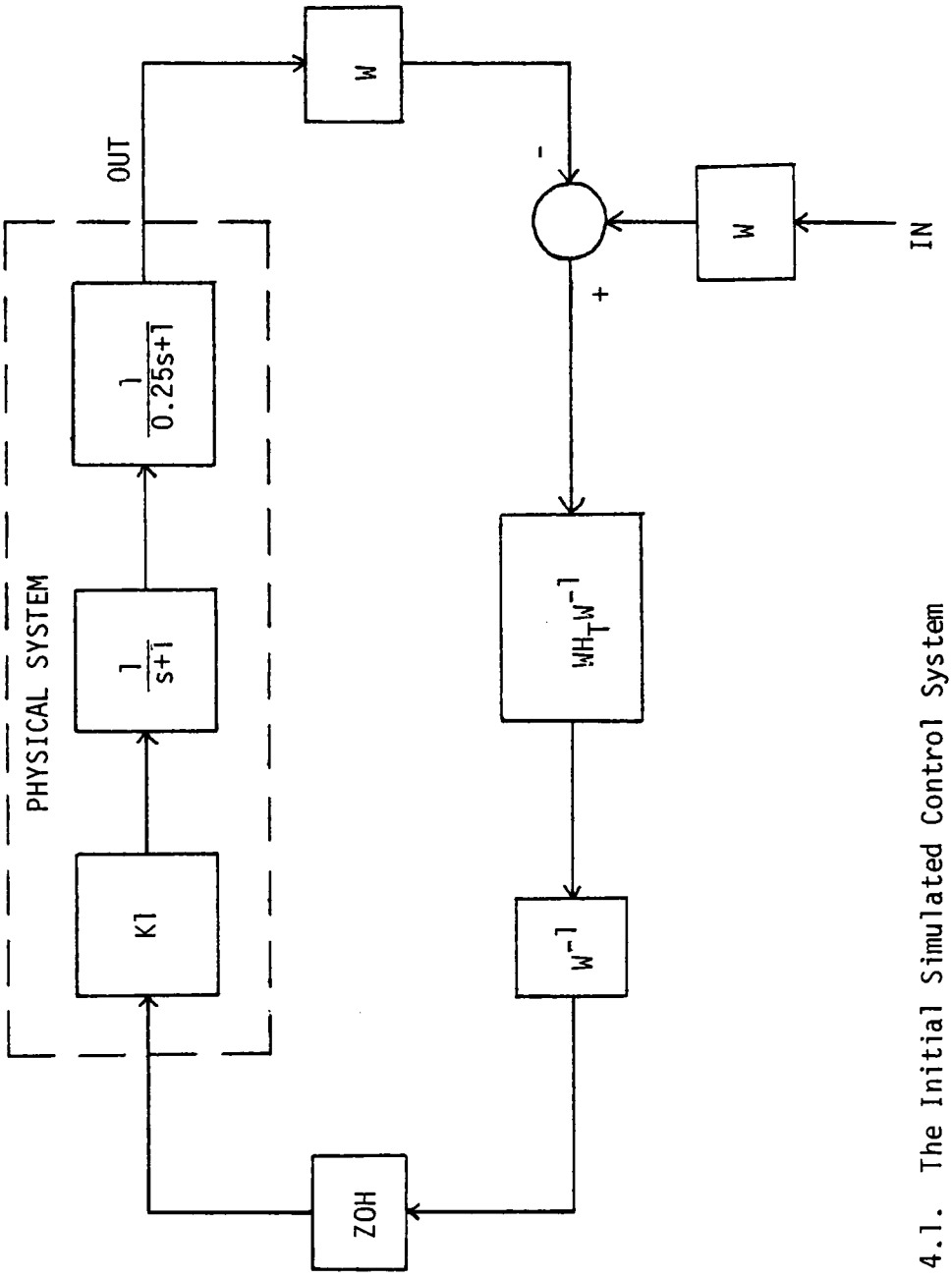

Figure 4.1. The Initial Simulated Control System

E = WALIN - WALOUT .

In an effort to reduce the amount of computer time required, a more efficient model was devised. This model is shown in Figure 4.2. The following identity of Walsh transforms allowed for the more efficient model:

Let $\hat{f}$ and $\hat{g}$ denote the Walsh transforms of some functions f and g, respectively. Then

$$\hat{f} + \hat{g} = \widehat{(f+g)} \quad . \tag{4.3}$$

The "D/A" block of the generic autopilot system in Chapter I is accomplished by means of a staircase approximation to the discrete output of the T^{-1} block and is termed a "zeroth order hold". This step is represented by the ZOH block in Figures 4.1 and 4.2.

4.4 The Simulation Program

The controller feedback system as seen in Section 4.3 is simulated with the aid of CSMP III, a more recent version of CSMP, and FORTRAN. Two program listings are found in the appendices. Appendix B is a listing for the simulation of the system shown in Figure 4.1. Appendix C is a listing for the simulation of the system seen in Figure 4.2.

4.5 Results

The feedback control system was initially simulated using a step function as the input. Several runs were made using various values

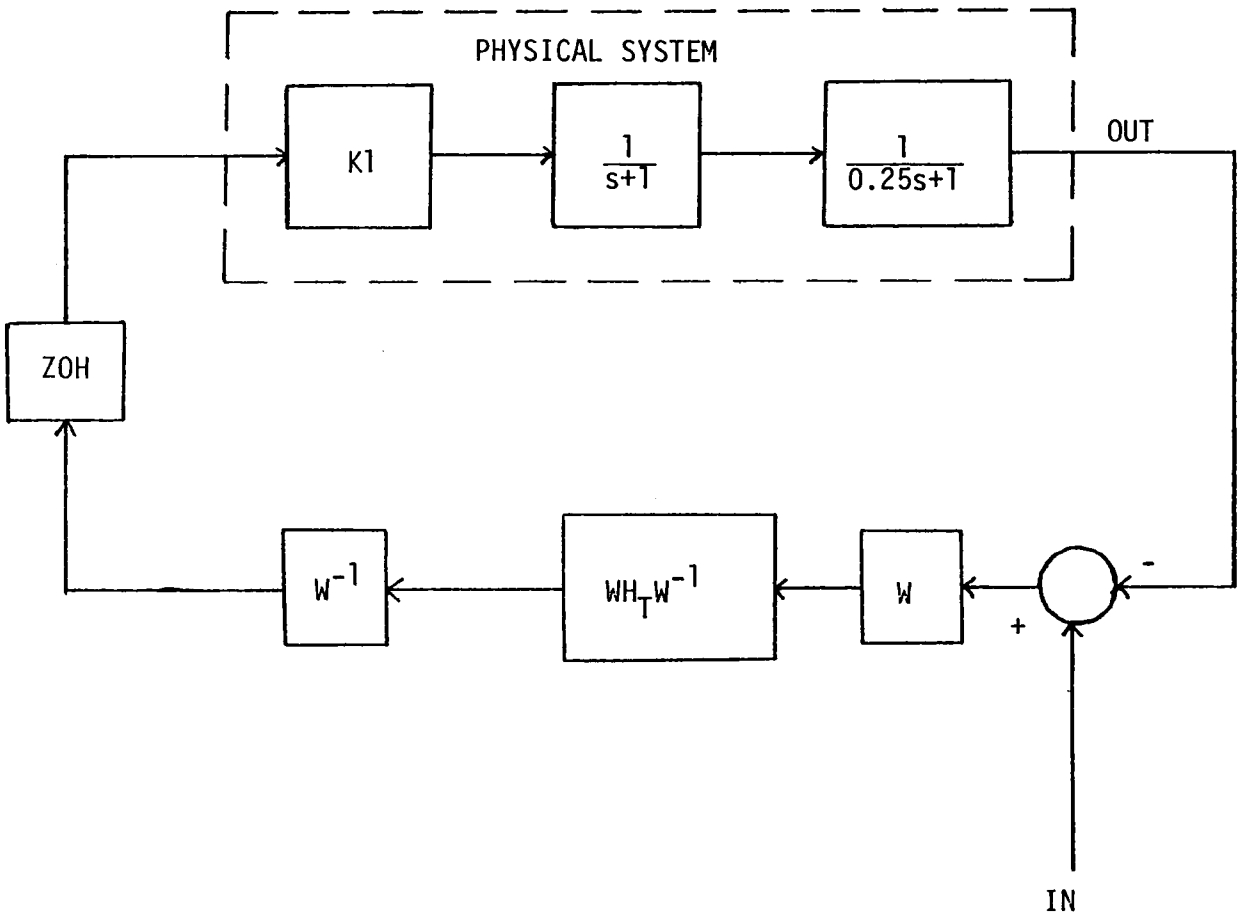

Figure 4.2. The Reduced Simulated Control System.

for the gain constant, K1 , of the transfer function. It was anticipated and desired that the physical system give an output curve, OUT, that approached the value of the step input. The system performed as was expected and the OUT curve for low gain can be observed in Figure 4.3.A.

As the gain constant is increased the system should reach a point of instability, i.e. the OUT waveform should begin to oscillate as the gain constant is increased. Figure 4.3.A-F shows that the system responded as expected as the gain was increased.

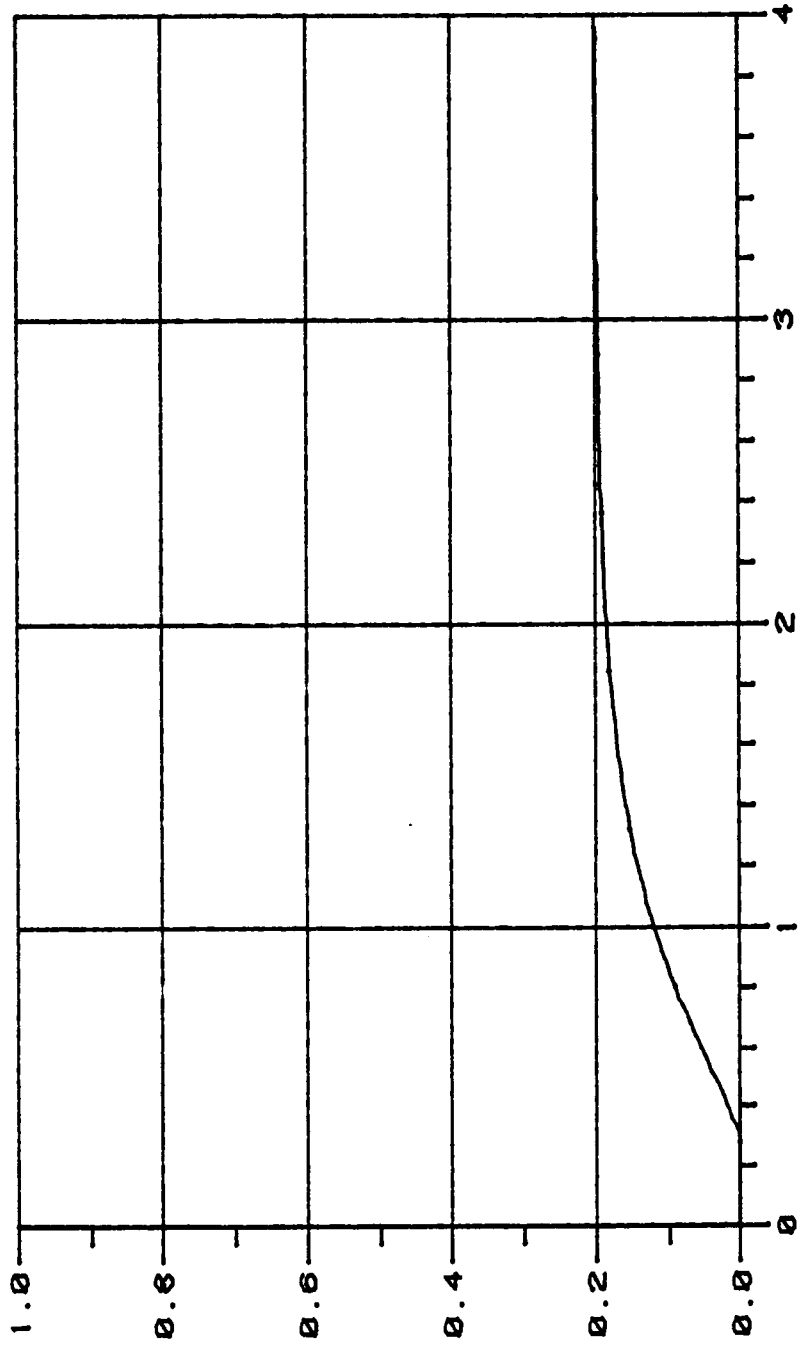

Figure 4.3. System Response Due to Step Input for Increasing Values of Gain, K1

A. For Low Gain, K1 = 2

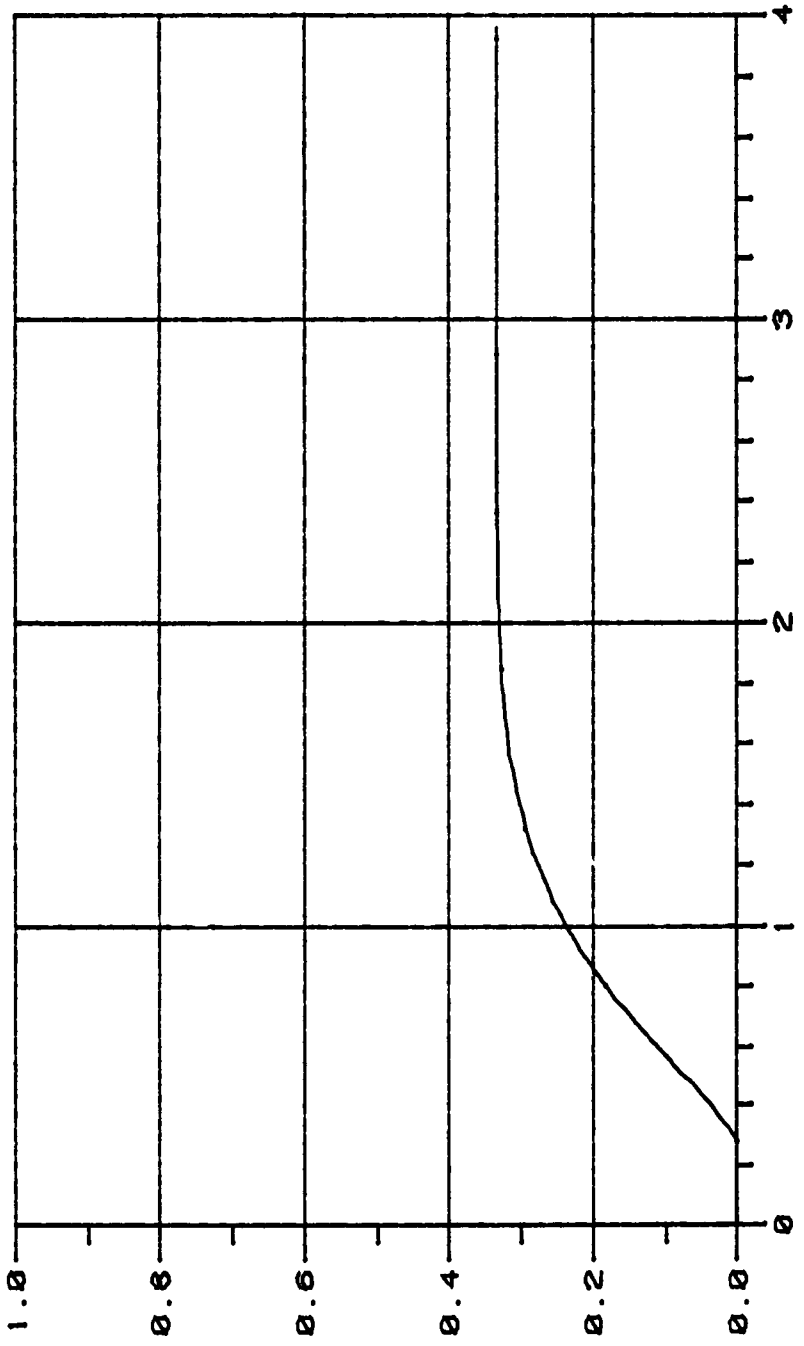

Figure 4.3 (Continued)

B. Gain = 4

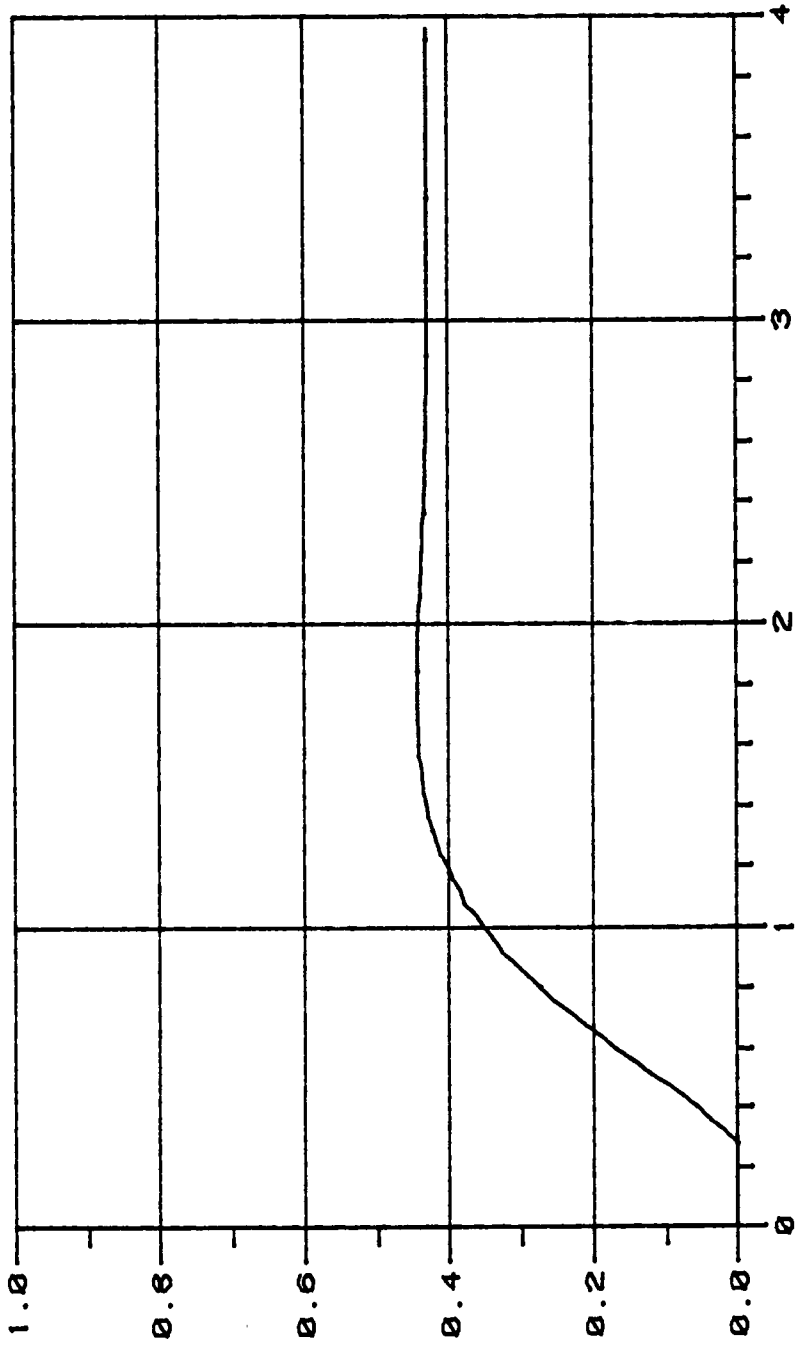

Figure 4.3 (Continued)

C. Gain = 6

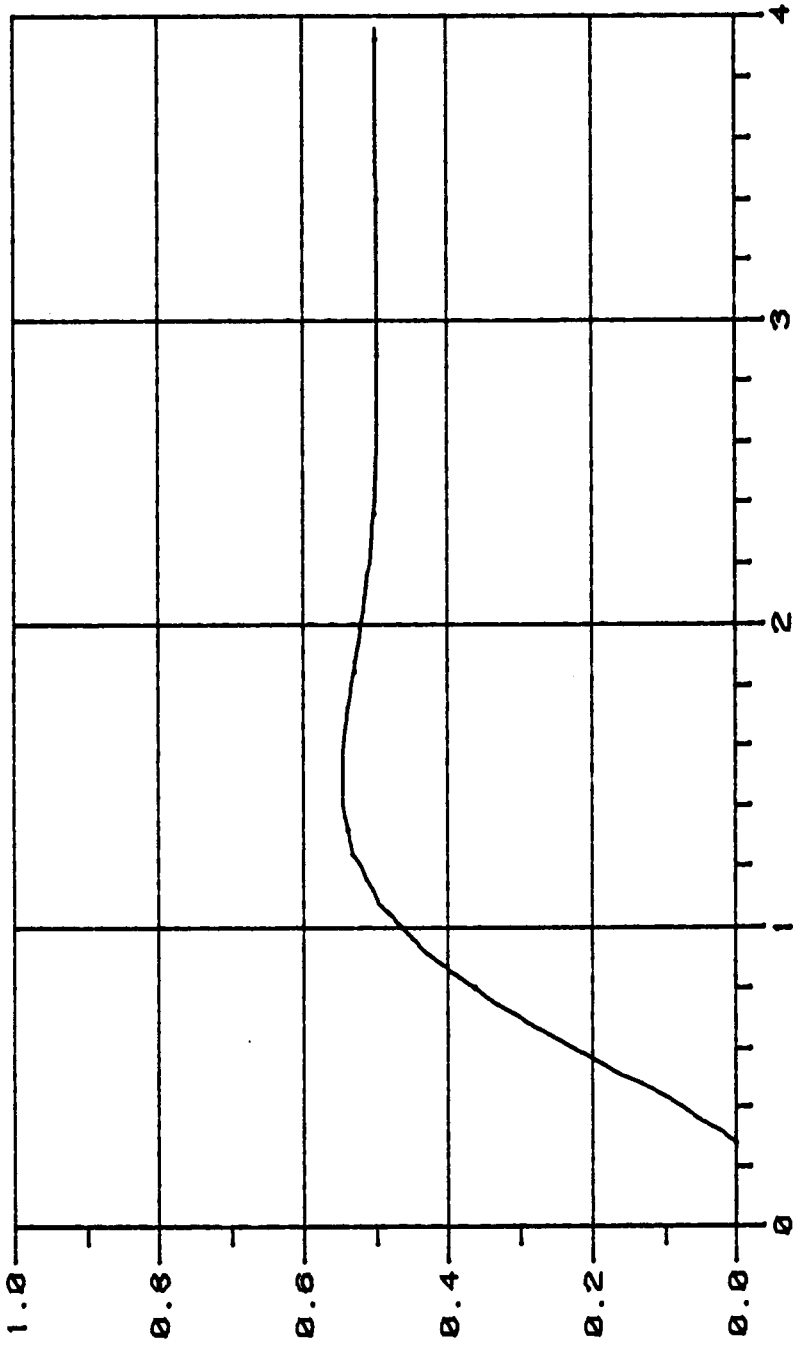

Figure 4.3 (Continued)

D. Gain = 8

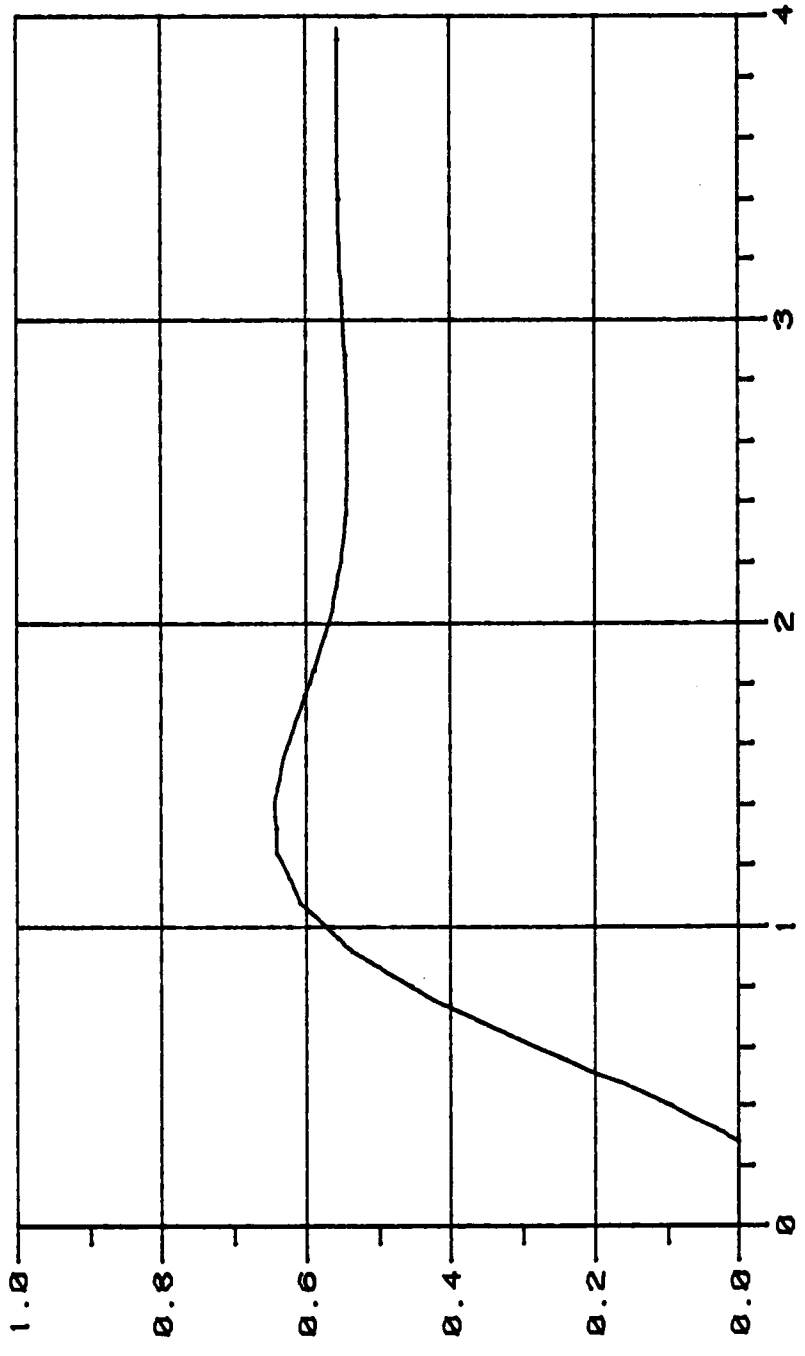

Figure 4.3 (Continued)

E. Gain = 10

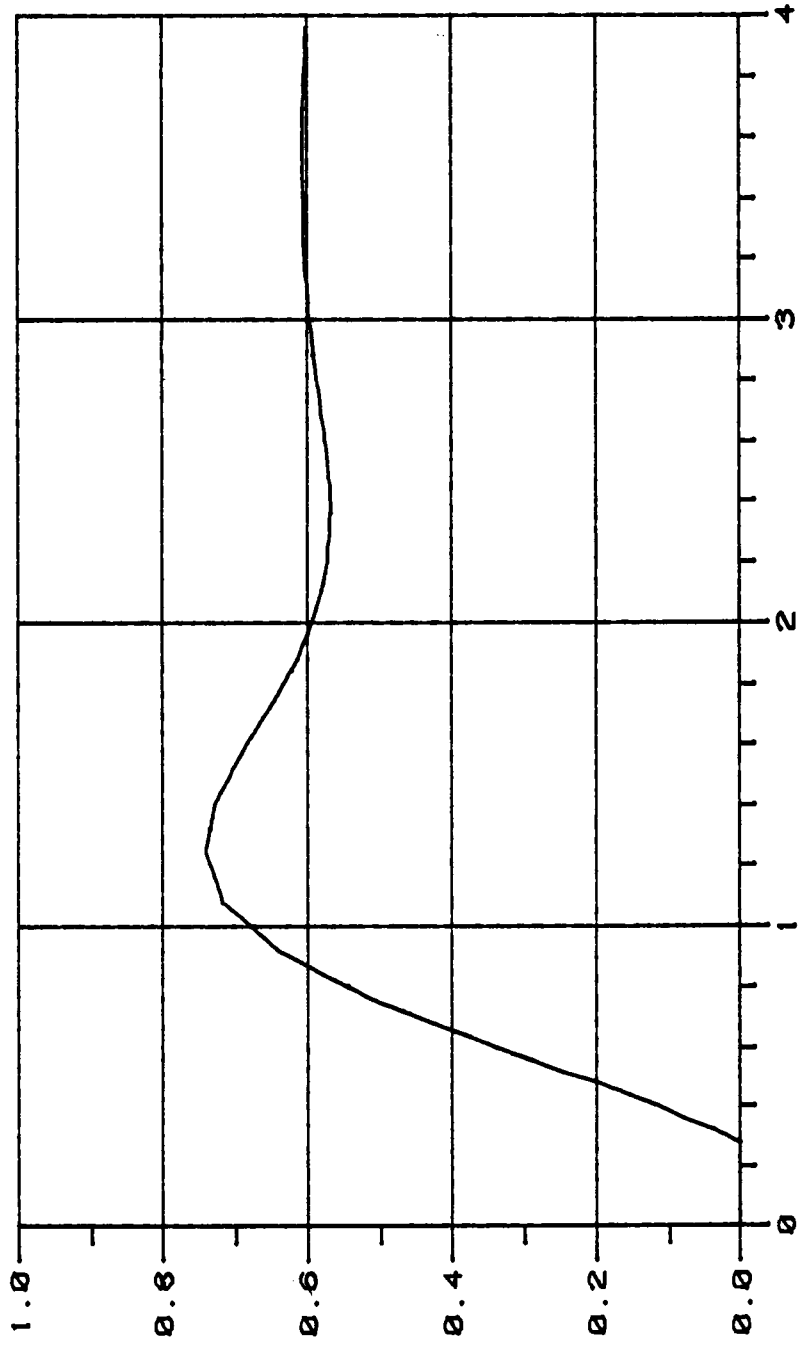

Figure 4.3 (Continued)

F. For High Gain, K1 = 12

CHAPTER V

CONCLUSIONS

Many of the conclusions in regard to the ability for the autopilot system to perform were reached in the preceeding chapter. In summation, it was determined that the discrete transform controller did properly control the output from the second-order physical system. The desired controller was implemented as an integrator.

An efficient method for determining the discrete transform controller was developed and it has been shown that Walsh transforms can successfully be used in control system applications. What can be performed in the Walsh domain can be done in other domains, but the preference for the discrete Walsh transform came about by the corresponding matrix containing only the integers ± 1 .

Other aspects that merit investigation are

1. developing a simulation program exclusively in FORTRAN,
2. building a mechanical system to connect directly to the digital computer and testing out the feedback control system,
3. comparing the results obtained in this study with those found using other transform methods to determine the computational efficiency of the various methods, and
4. instead of using an integrator for the discrete transform controller allow "a" to take on various values and compare the results.

LIST OF REFERENCES

LIST OF REFERENCES

1. Rao, K.R., K. Revuluri, M.A. Narasimham and N. Ahmed. "Complex HAAR Transform", IEEE Transactions on Acoustics Speech and Signal Processing, pp. 102-104, Feb. 1976.

2. Proc. 1970-1973 Symp. Appl. of Walsh Functions. Available from Nat. Tech. Inform. Service, U.S. Dept. Commerce, Springfield, VA 22151. Order number: 1970: AD-707 431, 1971: AD-727 000, 1972: Ad-744 650, 1973: AD-763 000. 1974. Proceedings may be obtained directly from IEEE Service Center, 445 Hoes Lane, Piscataway, NJ 08854, Order number: 74CH0861-5EMC.

3. Ahmed, N., T. Natarajan, and K.R. Rao. "Discrete cosine transform", IEEE Trans. Comput. (Corresp.), vol. C-23, pp. 90-93, Jan. 1974.

4. Rao, K.R., M.A. Narasimham, and K. Revuluri. "Image data processing by Hadamard Haar transform," IEEE Trans. Comput., vol. C-24, pp. 888-896, Sept. 1975.

5. Campanella, S.J. and G.S. Robinson. "A comparison of orthogonal transformations for digital speech processing," IEEE Trans. Commun. Technol., vol. COM-19, pp. 1045-1050, Dec. 1971.

6. Fino, B.J. "Relations between Haar and Walsh/Hadamard transform," Proc. IEEE (Lett.), vol. 60, pp. 647-648, May 1972.

7. Fino, B.J. and V.R. Algazi. "Slant Haar transform," Proc. IEEE (Lett.), vol. 62, pp. 653-654, May 1974.

8. Shore, J.E., "On the applications of Haar functions," IEEE Trans. Commun., vol. COM-21, pp. 209-216, Mar. 1973.

9. Rao, K.R. and N. Ahmed. "Modified complex BIFORE transform," Proc. IEEE (Lett.), vol. 60, pp. 1010-1012, Aug. 1972.

10. Robinson, G.S. "Logical convolution and discrete Walsh and power spectra," IEEE Trans. Audio Electroacoust., vol. AU-20, pp. 271-270, Oct. 1972.

11. Chew, C.F. and C.H. Hsiao. "A State Space Approach to Walsh Series Solution of Linear Systems," INT. J. SYSTEMS SCI., vol. 6, No. 9, pp. 833-858, 1975.

12. Shieh, L.S., C.K.Y. Evng and B.C.M. Cinnis. "Solution of State-SPACE Equations via Block-Pulse Functions," INT. J. CONTROL, vol. 28, No. 3, pp. 383-392, 1978.

13. Hung, J.C., C.C. Liu and P.Y. Chow. "An Algebraic Method for Systems Parameter Identification." Proceedings of the Fourteenth Asilomar Conference on Circuits, Systems and Computers, 17-19 November 1980.

14. Hung, J.C. and R.C. Liu. "A Comparison of Several Algebraic Methods for Control System Applications," Proceedings of the IEEE 1981 Southeastcon, 6-8, April 1981.

15. Beauchamp, K.G. Walsh Functions and Their Applications. New York: Academic Press, 1975.

16. Ahmed, N. and K.R. Rao. Orthogonal Transform for Digital Signal Processing. New York: Springer-Verlag, 1975.

17. Speckhart, Frank H. and Walter L. Green. A Guide to Using CSMP-The Continuous System Modeling Program. New Jersey: Prentice-Hall, Inc., 1976.

APPENDICES

APPENDIX A

PROGRAM LISTING FOR $(FWT)_W$ SUBROUTINE

```
C  THIS ROUTINE CALCULATES THE FAST WALSH TRANSFORMS FOR ANY GIVEN
C  NUMBER WHICH IS A POWER OF TWO

C  II = 0  WALSH-ORDERED WT
C  II = 1  INVERSE WALSH-ORDERED WT

C  N=  NUMBER OF SAMPLES

       REAL PR(10), X(32), Y(32), Z(32), E(32)
       INTEGER  B,B2,D,F,P


C  START BIT REVERSE

       DO 30 I = 1,N
          B=  I-1
          L=  1
10        D=  B/2
          PR(L)= 1
          IF (B.EQ.(D*2)) PR(L) = 0
          IF (D.EQ.0)  GOTO 20
          B=D
          L= L + 1
          GOTO 10
20        CONTINUE
          P = 1
          F = N
          DO 25  J=1,L
             F= F/2
25        P= P + F* PR(J)
30     Y(P)= X(I)

       DO 35  I=1,N
35     X(I)= Y(I)
40     CONTINUE


C  CALCULATE NUMBER OF ITERATIONS, K

       K= 0
       IR= N
41     IR=IR/2
       IF (IR.EQ.0)  GOTO 42
       K= K+1
       GOTO 41
42     CONTINUE
```

```
C  START LOOP FOR (LOG TO BASE TWO OF N) ITERATIONS

         DO 70 M=1,K


C  CALCULATE NUMBER OF PARTITIONS

         IF (M.EQ.1) NP=1
         IF (M.NE.1) NP= NP*2
         M1 = N/NP
         M2 = M1/2

C  START LOOP FOR THE NUMBER OF PARTITIONS

         A= 1.
         DO 60 MP=1,NP
            B= (MP-1)* M1

C  START A LOOP THROUGH THIS PARTITION

         DO 50 MP2=1,M2
            M3= M2+ MP2+ B
            B2= B+ MP2
            Y(B2)= X(B2)+ A* X(M3)
            Y(M3)= X(B2)- A* X(M3)
50       CONTINUE
         A= -A
60       CONTINUE

C  SWITCH  Y(I)  TO X(I)

65       DO 65 I=1,N
         X(I)= Y(I)
70       CONTINUE
         IF (II.EQ.1) RETURN

C  DIVIDE RESULTS BY FLOAT  (N)

         R= 1./N
         DO 80 I=1,N
80       X(I)= X(I)* R

         RETURN
         END
```

APPENDIX B

CSMP III PROGRAM LISTING FOR INITIAL CONTROL SYSTEM

```
INITIAL
   PARAM  K1=1, T1=1, T2=0.25, N=4, IJ=0
   INPUT= 1.0*STEP(0.0)
      DIMENSION  PR(10),X(32),Y(32),Z(32),E(32),HT(4,4),HTE(32),S(32)
      DIMENSION WALIN(32), WALOUT(32)
   FIXED  B,B2,D,F,I,II,IR,J,K,L,M,M2,M3,MP,MP2,N,NP,P,IJ
NOSORT
      READ(5,999) ((HT(J,I),I=1,4),J=1,4)
999   FORMAT(4F10.0)

DYNAMIC

NOSORT
   XI= INPUT
   IF (KEEP.NE.1)  GO TO 20
   IJ= MOD(IJ,4) + 1
   X(IJ)= XI
   IF(IJ.NE.4)  GOTO 20
   II= 0
   CALL WT(N,X,II)

   DO 2  I=1,N
   WALIN(I)= X(I)
2  E(I)= WALIN(I)- WALOUT(I)
   IF (KEEP.EQ.1)  WRITE(6,95)  (WALIN(I), I=1,N)
95 FORMAT(1X, 'FWT OF INPUT= ', 8F12.4)
   IF (KEEP.EQ.1)  WRITE(6,100)  (E(I),  I=1,N)
100 FORMAT(1X,'ERROR DUE TO STEP INPUT  E= ', 8F12.4)

   DO 3  J=1,N
      S1= 0
      DO 4  I=1,N
         S(I)= HT(J,I)* E(I)
         S1= S1+ S(I)
4     CONTINUE
3  HTE(J)= S1

   DO 5  I=1,N
5  X(I)= HTE(I)
   II= 1
   CALL   WT(N,X,II)
   IF  (KEEP.EQ.1)  WRITE(6,105)  (X(I),  I=1,N)
105  FORMAT(1X,'INVWAL OF DESIRED SIGNAL= ', 8F12.4)
```

```
    DO 6   I=1,N
       E1= X(I)

       E(I)= Q

SORT

       R1= K1* E(I)
       R2= REALPL(0.0,T1,R1)
       OUT= REALPL(0.0,T2,R2)

NOSORT
       X(I)= OUT
6  CONTINUE
    II= 0
    CALL WT(N,X,II)

    DO 7  I=1,N
7   WALOUT(I)= X(I)
    IF (KEEP.EQ.1)  WRITE(6,110)  (WALOUT(I),  I=1,N)
110  FORMAT(1X,'FWT OF SYSTEM OUTPUT= ', 8F12.4)

20   CONTINUE

TERMINAL

TIMER FINTIM=4.0, OUTDEL=0.04, DELT=0.04, DELMIN=8.D-7

OUTPUT Q,OUT, INPUT
LABEL STEP INPUT EFFECT ON SYS RESPONSE(OUT), ERROR(Q), STEP
INPUT(INPUT)
END
INPUT
    0.5       0.25     0.0       0.125
   -0.25      0.0      0.125     0.0
    0.0      -0.125    0.0       0.0
   -0.125     0.0      0.0       0.0
ENDINPUT
STOP
```

APPENDIX C

CSMP III PROGRAM LISTING FOR REDUCED CONTROL SYSTEM

```
INITIAL
   PARAM  K1=1, T1=1, T2=0.25, N=4, IJ= 0
   INPUT= 1.0*STEP (0.0)
      DIMENSION  PR(10),X(32),Y(32),Z(32),E(32),HT(4.4),HTE(32),S(32)
      DIMENSION  IN(32), OUT(32)
   FIXED B,B2,D,F,I,II,IR,J,K,L,M,M2,M3,MP,MP2,N,NP,P,IJ
NOSORT
      READ(5,999) ((HT(J,I),I=1,4),J=1,4)
999   FORMAT(4F10.0)

DYNAMIC

NOSORT
   XI= INPUT
   IF (KEEP.NE.1)  GO TO 20
   IJ= MOD(IJ,4) + 1
   IN(IJ)= XI
   IF (IJ.NE.4)  GOTO 20

   DO 2  I=1,N
   E(I)= WALIN(I)- WALOUT(I)
2  X(I) = E(I)

   IF (KEEP.EQ.1)  WRITE(6,100) (E(I), I=1,N)
100 FORMAT(1X,"ERROR DUE TO STEP INPUT E= ', 8F12.4)
   II = 0
   CALL WT(N,X,II)


   DO 3 J=1,N
      S1=0
      DO 4 I=1,N
         E(I)=X(I)
         S(I)= HT(J,I)* E(I)
         S1= S1+ S(I)
4     CONTINUE
3  HTE(J)= S1

   DO 5 I=1,N
5  X(I)= HTE(I)
   II= 1
   CALL WT(N,X,II)
   IF (KEEP.EQ.1) WRITE(6,105) (X(I), I=1,N)
105 FORMAT(IX,'INVWAL OF DESIRED SIGNAL= ', 8F12.4)
```

```
   DO 6  I=1,N
      E1= X(I)

SORT
      A1= IMPULS(0.0,0.04)
      Q= ZHOLD(A1,E1)

NOSORT
      E(I)= Q

SORT
      R1= K1* E(I)
      R2= REALPL(0.0,T1,R1)
      OUTPUT= REALPL(0.0,T2,R2)

NOSORT
      OUT(I)= OUT
6  CONTINUE
20  CONTINUE

TERMINAL

TIMER FINTIM=4.0,   OUTDEL=0.04, DELT=0.04, DELMIN=8.D-7
OUTPUT Q, OUT, INPUT
LABEL STEP INPUT EFFECT ON SYS RESPONSE(OUT), ERROR(Q), STEP INPUT(IN-
PUT)
END
INPUT
    0.5       0.25      0.0       0.125
   -0.25      0.0       0.125     0.0
    0.0      -0.125     0.0       0.0
   -0.125     0.0       0.0       0.0
ENDINPUT
STOP
```

VITA

Elizabeth Barnwell Lowry was born in Greenville, South Carolina, on March 11, 1955. She is the daughter of Dr. and Mrs. F. Houston Lowry, Jr., of Madisonville, Tennessee. She attended elementary school in Madisonville and graduated from Madisonville High School in June 1973. She received a four-year Valedictorian Scholarship to Maryville College, Maryville, Tennessee. She graduated in May 1977 with a Bachelor of Arts degree in Mathematics and a certification in Secondary Education. Following graduation she began her employment with the Knoxville City School System, Knoxville, Tennessee. In June 1979, she enrolled in The University of Tennessee, Knoxville Summer School. In the fall of 1979 she accepted a teaching assistantship and continued study toward a Master's degree. The Master of Science degree in Electrical Engineering was received in June 1982.

The author was a member of the Society of Industrial and Applied Mathematics (SIAM) and a member of the Institute of Electrical and Electronics Engineers, Inc. (IEEE).