THE LOGICAL SYNTHESIS OF HYBRID CONTROL SYSTEMS

Thesis Approved:

Karl N. Reid
Thesis Adviser

Paul A. McCollum

L. D. Fitch

Henry R. Sebesta

D. Dunham
Dean of the Graduate College

ACKNOWLEDGMENTS

There are many individuals and factors which have helped make my graduate studies meaningful and enjoyable. One of the factors is the professional atmosphere of the department and personnel at Oklahoma State University.

The main individual to whom I owe very much respect and gratitude is my major adviser, Dr. Karl N. Reid. His encouragement and guidance throughout many projects have been a great inspiration. His advice on this and other writings have been an invaluable aid to me.

I am also fortunate to have had such an excellent committee. The advice and patience of professors H. R. Sebesta, E. C. Fitch, and P. A. McCollum is greatly appreciated.

The work reported here was sponsored by the Center for Systems Science which is supported in part by the National Science Foundation under GU-3160.

There are three types of logic: binary logic, continuous logic, and female logic (better known as female illogic). I am pleased to say that my wife Brenda refrained from using the latter during our stay at Stillwater. For this and her continual support I am forever indebted. Her cooperation in the typing of the manuscript is appreciated.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Control systems basically fall into two groups: digital and proportional systems. Digital control systems have inputs and outputs which are either "ON" or "OFF". The output status is determined by the inputs in a decisional manner. These inputs are referred to as decisional inputs.

The inputs to a totally proportional control system constantly affect the outputs in a proportional manner. These inputs are termed continuous inputs since they are continuous-valued and always affect the outputs regardless of their magnitudes.

A combination of these two extremes is termed hybrid. Hybrid control systems have some combination of decisional and continuous inputs and/or outputs. In a typical system, switching occurs between two or more continuous (proportional) control policies depending upon the decisional state. Continuous variables are gated by decisional variables.

Motivation for Hybrid Logic Synthesis

There are three stages in the process of building a system to perform some control task. These are: (1) the synthesis of the desired control philosophy and the

representative mathematics, (2) the design of a system to implement these ideas or equations, and (3) the analysis and sizing of the physical system.

The synthesis technique which has been developed for digital logic systems provides a systematic procedure for the first two stages mentioned above. This procedure is so systematic that computer programs can carry out most of the procedure. No similar procedure exists for hybrid or proportional control systems. These controllers are designed by engineering intuition which brings together the system requirements, control philosophy, and the designer's prior experience and knowledge. Such design is often a trial and error process.

Since proportional and hybrid control systems represent a broad class of control systems, there is a need for an organized, systematic procedure for the synthesis and design of these systems. This dissertation presents a systematic method for the synthesis of control systems having hybrid rather than merely binary control actuation.

Scope And Results Of Study

This study deals with a procedure for the synthesis of hybrid control systems. The synthesis steps are well defined and mechanized to such a state that most steps could be carried out by the use of a digital computer program. The technique allows the synthesis of the controller for any control system having a control policy or policies

which can be logically described by binary or continuous mathematics. This logical description includes a list of possible conditionals (if-then statements) involving binary or continuous variables. An algebra for the logical variables is adapted to allow equation manipulation and simplification. Finally, implementation procedures are noted to further simplify the designer's task.

The end result of this concept is a unified procedure for the synthesis and implementation of a controller for either binary, hybrid, or proportional control systems.

CHAPTER II

REVIEW OF LITERATURE

The synthesis concept presented here is not reported anywhere in the literature. The concept is a combination of digital logic synthesis and continuous (infinite-valued) logic, both of which are discussed in the literature. A summary of the works cited in the literature pertinent to this study is given below.

Binary Logic To Digital Synthesis

The foundations of formal logic were set in 340 B.C. by the Greek philosopher Aristotle. His work on the laws of logical reasoning was so complete that it has remained essentially unchanged throughout time. Many philosophers tried without success to devise a suitable mathematical representation for formal logical reasoning until 1854 when an English mathematician, George Boole, founded a two-valued algebra which could be used to represent true-false propositions. Boolean algebra has since been well developed for philosophic logic and has become a favorite topic of the mathematician.

Until the present century, Boolean algebra was used

exclusively as a mathematical expression of sentential logic. In 1938 Claude Shannon (1) applied the algebra to the modeling of electric relay contact circuits. With this concept it became possible to convert word statements of desired circuit operation directly into actual circuit diagrams. Using the algebra, complex circuits could be reduced or rearranged to produce simpler circuits. This was a major step towards mechanized synthesis which does not require engineering intuition or design.

This concept was extended from combinational to sequential circuit synthesis involving memory in 1954 by D. A. Huffman (2) and E. F. Moore (3). Huffman and Moore independently developed a method for asynchronous sequential circuit synthesis which is referred to today as the Huffman-Moore model (4,5). The work on logic synthesis has set the stage for the development of such complex switching circuits as the telephone system, general purpose programmable digital computers, special purpose computers, and a host of other digital systems.

Many-Valued Logic To Continuous Synthesis

Digital logic synthesis has proved to be an invaluable tool in modern engineering; but there is another branch in the development of logic which leads to the synthesis concept presented here. After the development of Boolean algebra an intense interest in logic lead to the need of a quantification finer than true or false. Consequently, a

many-valued logic evolved (6,7,8). This logic allowed the logician to work with such quantities as "how much true" or to assign probabilities to an event.

Multi-valued logic has found only limited physical applications (9,10). The use of multi-valued logic promises to reduce hardware significantly compared to the same implementation using binary logic.

Many-valued logic was increased from a three-valued logic to higher-valued logic in an effort to get finer quantification. Finally, an infinite-valued logic emerged (11,12). Infinite-valued logic allows a continuum of truth values between complete truth and complete falsity. The algebra corresponding to infinite-valued logic reduces to Boolean algebra when the truth values are restricted to two values, true or false. Consequently, infinite-valued logic is the most general form of logic algebra.

One of the first reported applications of infinite-valued logic to physical systems was by Schaefer (13). Schaefer used the infinite-valued logic algebra to model electronic circuits with rectifiers. His "rectifier algebra" demonstrated the mathematical modeling and algebraic simplification of rectifier circuits.

Later this logic was used to synthesize arbitrary electronic function generators by Wilkinson (14). This technique allowed the direct synthesis of approximations to multiple input nonlinear functions using diodes. Ginsburg (15,16) has continued with this approach. The work with

the synthesis of functional approximations is an interesting application of continuous or infinite-valued logic; however, it is not the concept presented here.

The literature has indicated how binary logic has evolved into digital logic synthesis and how binary logic has been expanded into infinite-valued logic and continuous functional synthesis. Now, one more step in development allows the mechanized synthesis of hybrid control systems using the concepts of digital logic synthesis and the algebra of continuous logic.

CHAPTER III

THE SYNTHESIS PROCEDURE

The synthesis procedure for hybrid control systems presented here closely follows the procedure for the synthesis of digital logic systems. The procedure consists of the following steps: (1) a description of the physical system to be controlled, (2) a detailed word statement of the desired control philosophy, (3) a definition of logic variables to be used as inputs and outputs of the controller to be synthesized, (4) a complete logical description of the possible input conditions and the desired output states (a state table or primitive flow table), (5) a derivation of basic control equations from the state table or primitive flow table, (6) simplification and manipulation of the basic control equations, (7) derivation of a logic or functional block diagram of the controller, and (8) a physical implementation of the controller. This procedure is applicable to the synthesis of totally proportional control systems, in which case step (4) above is unnecessary. The input-output equations can be written directly.

Steps 1 through 5 are discussed in this chapter and steps 6 through 8 are discussed in the next chapter. These steps are illustrated by one example problem in these chapters. Appendix B gives several problems which further

illustrate the complete synthesis procedure.

Problem Description and Definition

The synthesis procedure begins with the description or knowledge of the forward loop of the physical system to be controlled. This includes such things as the characteristics of the system components, interconnection of components, method of actuation, etc. To illustrate the synthesis technique steps, consider the synthesis of a proportional controller for an automatic-dig backhoe.[1]

Existing Physical System

The physical system to be controlled is illustrated by Figure 1. The three controllable members, the boom, dipstick, and bucket, are actuated by hydraulic cylinders. A hydraulic pump driven by the tractor engine supplies the necessary hydraulic power.

Desired Control Philosophy

The desired control philosophy for this problem is determined from studying the operating technique of skilled operators. As reported by Caywood (17), the best operation

---

[1]This problem was one of the first industrial applications of fluid logic synthesis techniques developed at Oklahoma State University. The solution for the automatic-dig backhoe using digital logic was presented in a paper by Caywood (17). The development here closely parallels the solution given by Caywood except that decisional and continuous variables rather than only decisional variables are used.

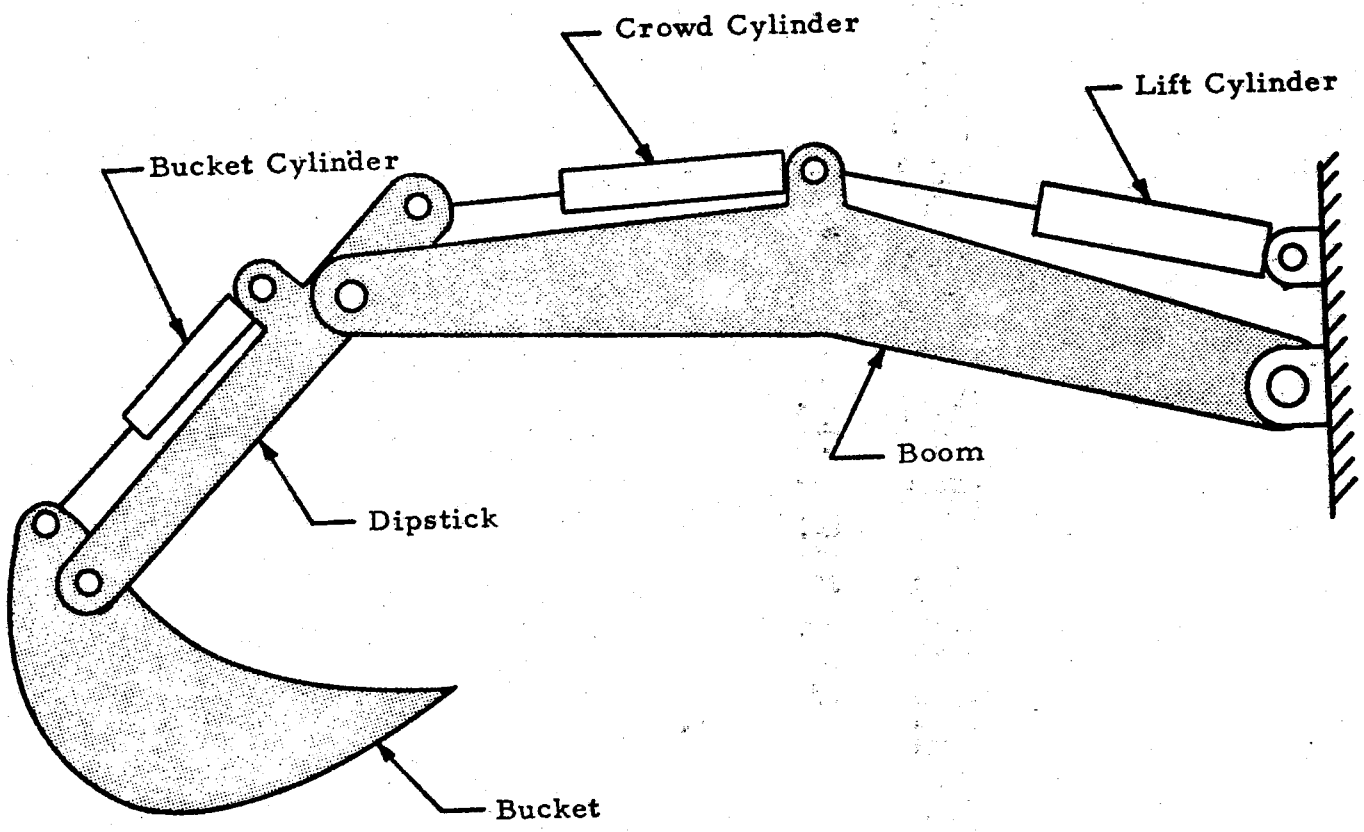


Figure 1. Diagram of Backhoe

of the dig cycle is accomplished by performing the following: Actuate the crowd cylinder to slice off a layer of dirt. While continuing to crowd, curl the bucket to maintain the proper lip angle since an improper lip angle stalls the crowding action. Occasionally, the boom is lifted to maintain a cutting depth which prevents both crowd and bucket cylinders from stalling.

From the above operational description, a word statement of desired control philosophy can be deduced. First of all, the crowd cylinder should be actuated at all times in automatic operation. Second, the bucket should be curled whenever the pressure in the crowd cylinder indicates a stalling condition is being approached (i.e., when the pressure in the extend side of the crowd cylinder is too high). It is desired to curl the bucket at a rate proportional to the overload rather than at a maximum rate. If the crowd cylinder is just barely overloaded, the bucket should be actuated with a slight effort; but, if the crowd cylinder indicates a large overload, the bucket should be actuated with a correspondingly large effort. This proportional control will help reduce the jerk caused by switching the bucket at a maximum effort.

Lastly, the boom should be lifted when the pressures in the crowd and bucket cylinders both indicate the approaching of stalling conditions. Again, the boom should be lifted at a rate proportional to the percent crowd overload in order to provide a smoother control action.

The preceding word statement is valid for the dig cycle of the operation. The bucket must be dumped and repositioned manually before starting on another dig cycle.

Definition of Logic Notation

The logic variables used in the controller are defined by normalized functions of the system variables of the physical system. Thus, the physical variables, the range of the physical variables, thresholds, null points, etc., must be specified before defining logic variables.

System Variables

The system variables and parameters influencing the controller under consideration are given below.

P_b ~ Pressure in extend side of bucket cylinder. $0 \leq P_b \leq P_s$

P_{bo} ~ Bucket cylinder pressure overload threshold.

P_c ~ Pressure in extend side of crowd cylinder. $0 \leq P_c \leq P_s$

P_{co} ~ Crowd cylinder pressure overload threshold.

P_s ~ Maximum hydraulic supply pressure.

Logic Variables

The logic variables are defined from the physical variables. For this problem there are both decisional and continuous variables. The decisional variables determine the control policy, and continuous inputs describe the "how" of the policy. The overload conditions in the crowd

and bucket cylinders determine the control policy. Thus, the two binary decision variables X_1 and X_2 are defined as follows:

$$X_1 \sim P_b - P_{bo} \qquad X_1 = 0 \Rightarrow P_b \leq P_{bo}$$
$$X_1 = 1 \Rightarrow P_b > P_{bo}$$
$$X_2 \sim P_c - P_{co} \qquad X_2 = 0 \Rightarrow P_c \leq P_{co}$$
$$X_2 = 1 \Rightarrow P_c > P_{co}$$

The quantities $(P_b - P_{bo})$ and $(P_c - P_{co})$ are termed the arguments of X_1 and X_2 respectively. These variables are indicators of the overload condition. If there is an overload, $X_i=1$; if not, $X_i=0$.

The problem statement indicates the necessity for proportional or continuous-valued control. The required continuous variable is the percent overload in the crowd cylinder, α_2, defined below.[2]

$\alpha_2 \sim$ Percent crowd overload

$$\alpha_2 = \frac{P_c - P_{co}}{P_s}$$

The control outputs to be synthesized are the actuation signals for the crowd, bucket, and lift cylinders. These equations should specify when (i.e., under what conditions)

---

[2] A Greek letter is used to indicate a continuous variable whose value is any real number α such that $-B \leq \alpha \leq B$. The continuous logic variables are usually normalized such that $B=1$; however, in general $0<B<\infty$. A Roman letter is used to indicate a binary variable whose value is restricted to zero or one. $X_i=0$ if the argument is less than or equal to zero, and $X_i=1$ if the argument is positive. The subscripts for binary and continuous variables are the same whenever they refer to the same physical variable. Since X_2 is related to crowd pressure in this problem, α_2 rather than α_i is used to denote a continuous function of P_c.

and how (i.e., the rate) the cylinder should be actuated. The crowd cylinder is actuated in a binary fashion and the bucket and lift cylinders are actuated proportionally. Thus, the outputs to be synthesized are:

Z_1 ~ Extend crowd cylinder

ζ_2 ~ Extend bucket cylinder

ζ_3 ~ Retract lift cylinder

The logic equation for Z_1 will be binary; whereas, the equations for ζ_2 and ζ_3 will be continuous. The resulting equations will be a function of inputs X_1, X_2, and α_2.

State Table Representation

Having defined the decisional inputs, every possible combination of the decision inputs must be considered and the desired output state specified. For a combinational problem, the state table is the most convenient representation. Sequential problems requiring memory are best described using the primitive flow table. If it is not known whether the problem requires memory or not, the primitive flow table can still be used for combinational problems.

A state table consists of two major columns, one containing every possible combination of the decisional inputs, and the second containing the desired output state for each input state. This state table should be thought of as a matrix whose entries indicate the input-output relationship. The matrix concept is very significant since it has lead to two thesis topics. The author first conceived the use of a

matrix to represent the input-output relation of known combinational equations [Z] = [M][X]. Then it was realized that this relationship (matrix) could be synthesized for either combinational or sequential problems. Since the input and output vectors are known, the entries in the matrix are the only unknown. By a definite procedure the entries can be determined. This concept lead to the author's master's thesis on the "state matrix method" for digital system synthesis (18).

Combinational problems have only zeros or ones as the entries in the matrix. Since the entries in the matrix are multiplied by each term in the input vector, a zero entry indicates that the output is not a function of a certain input (or input state). A one entry indicates the output is related to that input state. Sequential problems have memory variables as well as zeros and ones as entries. This states that the output is related to the inputs modified by memory states. Thus, variables as well as identity elements ("o" or "1") can be entered into the matrix. If this is true, then could additional binary inputs or even proportional variables be entered to further modify or describe the given input states? Of course, almost any type of variable can be entered. Memory variables, additional inputs, continuous inputs, or even dynamic operators can be entered. The entry of the Laplace operator $\frac{1}{s}$ could be used to indicate that the output is the integral of the input. Other forms of operators could be used to obtain the desired

response (e.g., $\frac{s}{s+1}$, etc.). This type of synthesis would be termed dynamic logic. In general, the decisional state tells when to give an output, the output tells what output to give, and the entries in the matrix or state table tell how to give the output (zero = no output; one = full output; α = proportional output; etc.). The matrix notation will not be used in this writing.

This problem is combinational and has two decisional inputs and three outputs. The state table is given below.

Inputs	Outputs		
$X_1 X_2$	Z_1	ζ_2	ζ_3
00	1	0	0
01	1	α_2	0
11	1	α_2	α_2
10	1	0	0

The first decisional state is the condition $X_1=0$ and $X_2=0$ which implies the bucket and crowd cylinders are not overloaded. For this decisional state the desired control policy is to crowd at full rate ($Z_1=1$) but not to actuate the bucket or lift cylinders ($\zeta_2=0$, $\zeta_3=0$). With this output state, the backhoe will continue to slice off a layer of dirt.

The second decisional state ($X_1=0$, $X_2=1$) indicates an overload in the crowd cylinder. As stated in the desired control philosophy, if this condition exists the control policy is to continue to crowd ($Z_1=1$) while curling the bucket with an effort proportional to the percent crowd overload ($\zeta_2=\alpha_2$). The boom is not lifted ($\zeta_3=0$) under

these conditions. Accordingly, this output state is noted in the second row. With this output state, the bucket will be curled until the lip angle is correct so that the overload is relieved or state three is encountered.

State three indicates that both bucket and crowd cylinders are overloaded. Under this condition the desired output state is to crowd, curl the bucket proportionally, and lift the boom with an effort proportional to the percent crowd overload. This output state ($Z_1=1$, $\zeta_2=\alpha_2$, $\zeta_3=\alpha_2$) is entered in row three. Actuation of both bucket and boom cylinders will relieve the overloading condition.

The last state indicates that the bucket cylinder alone is overloaded. As is often the case, this state is not totally specified by the word statement but is encountered in the logical specification. Under this input condition the crowd only should be actuated. This will either relieve the overload or cause a transition back to state three.

This completes the state table. This table now contains the complete information required to implement the controller. The next step is the derivation of mathematical equations from this table which contain the same information represented by the table.

Derivation of Basic Control Equations

The basic control equations are formed by combining together the states in the table for each output. To obtain the equation for an output, each decisional state is first

multiplied by the corresponding entry in the column under that output and then the resulting terms are combined together into a single equation. This is equivalent to multiplying a matrix by the input vector. This equation then contains a term for each decision state in the table. This expanded form of the output equation is termed the basic control equation.

The general rule for the method of combination of the terms is to algebraically ADD together terms involving continuous variables, and to logically OR together terms involving variables which are exclusively binary.[3]

As an example of this equation derivation, consider the state table of the previous section which is repeated below for convenience.

Inputs	Outputs		
$X_1 X_2$	Z_1	ζ_2	ζ_3
00	1	0	0
01	1	α_2	0
11	1	α_2	α_2
10	1	0	0

[3]The algebra used here is a combination of the ordinary algebraic operators of "ADD" and "MULTIPLY" and the logical operators of "OR" and "AND" for continuous variables. The operators are denoted by:

$+$ ~ Algebraic ADD:	$\alpha_1 + \alpha_2$	= SUM
Juxtaposition ~ MULTIPLY:	$\alpha_1 \alpha_2$	= PRODUCT
$\oplus$ ~ Logical OR:	$\alpha_1 \oplus \alpha_2$	= MAX(α_1, α_2)
$\circ$ ~ Logical AND:	$\alpha_1 \circ \alpha_2$	= MIN(α_1, α_2)
$\overline{\alpha}$ ~ Logical COMPLEMENT:	$\overline{\alpha}_1$	= $-\alpha_1$

This algebra and its properties are stated explicitly in Appendix A. The above definitions are sufficient for an understanding of the concepts presented in this chapter.

For the first output equation, Z_1, each decisional state is multiplied by the corresponding entry under Z_1. By substituting the logic notation for "00", the first state becomes $\overline{X}_1 \circ \overline{X}_2$ multiplied by "1", $(\overline{X}_1 \circ \overline{X}_2)1$. All of the entries for Z_1 are binary; thus, each term should be combined using the logical OR.

$$Z_1 = (\overline{X}_1 \cdot \overline{X}_2)1 \oplus (\overline{X}_1 \cdot X_2)1 \oplus (X_1 \cdot X_2)1 \oplus (X_1 \circ \overline{X}_2)1$$

Using the identity property that $(X)1=X$, the above equation reduces to

$$Z_1 = \overline{X}_1 \cdot \overline{X}_2 \oplus \overline{X}_1 \circ X_2 \oplus X_1 \circ X_2 \oplus X_1 \circ \overline{X}_2 .$$

This is the basic control equation. It can obviously be simplified but the simplification is discussed in the next chapter.

The equation for ζ_2 is obtained by multiplying each decision state by its entry and combining the terms to yield:

$$\zeta_2 = (\overline{X}_1 \cdot \overline{X}_2)0 + (\overline{X}_1 \circ X_2)\alpha_2 + (X_1 \circ X_2)\alpha_2 \oplus (X_1 \circ \overline{X}_2)0$$

Of course, X multiplied by "0" is zero so that two of these terms automatically vanish:

$$\zeta_2 = \overline{X}_1 \cdot X_2 \alpha_2 + X_1 \circ X_2 \alpha_2$$

This is the basic control equation for ζ_2.

The last control equation has only one non-trivial term since there are three zero entries. Thus, the basic control equation for ζ_3 is

$$\zeta_3 = X_1 \cdot X_2 \alpha_2 .$$

At this point, the basic control equations have been derived which specify the complete control philosophy.

Usually the equations can be simplified or put into better form before attempting to implement a physical controller from these equations. The simplification and implementation of control equations is the subject of the next chapter.

CHAPTER IV

IMPLEMENTATION OF EQUATIONS

This chapter discusses the conversion of the basic control equations described in the previous chapter into physical systems. Before discussing physical implementations, it is appropriate to present simplification and manipulation methods for the basic control equations.

Equation Simplification

There are two basic methods to simplify or manipulate equations. The use of the algebraic properties is most often the best method. Graphical techniques such as the Karnaugh maps prove to be quite helpful especially when there are optional terms which can be used.

Algebraic Methods

The basic concepts of the algebra are introduced in the previous chapter. The reader should refer to Appendix A for a more complete discussion of this algebraic system. A few simple examples of equation simplification are presented here; further examples are presented in the problems of Appendix B.

As the first example of simplification, consider the

binary reduction of the equation for Z_1 from the backhoe problem in the previous chapter. The basic control equation is given by:

$$Z_1 = \overline{X}_1 \cdot \overline{X}_2 \oplus \overline{X}_1 \cdot X_2 \oplus X_1 \cdot X_2 \oplus X_1 \cdot \overline{X}_2$$

By property P.8 of Appendix A,

$$Z_1 = \overline{X}_1 \cdot (\overline{X}_2 \oplus X_2) \oplus X_1 \cdot (X_2 \oplus \overline{X}_2).$$

By P.14:

$$Z_1 = \overline{X}_1 \cdot 1 \oplus X_1 \cdot 1$$

By P.13:

$$Z_1 = \overline{X}_1 \oplus X_1$$

Again using P.14, the final result is

$$Z_1 = 1.$$

The above equation states that Z_1 is independent of the inputs X_1 and X_2, thus being ON at every state for automatic operation.

The equation for ζ_2 represents the use of continuous variables rather than exclusively binary variables. The basic control equation for ζ_2 is

$$\zeta_2 = \overline{X}_1 \cdot X_2 \alpha_2 + X_1 \cdot X_2 \alpha_2.$$

This equation involves two hybrid terms. The simplification of this equation requires two basic properties of hybrid operations. First, the $X_2 \alpha_2$ term can be "factored out" by a property discussed in the "Notes on the Algebra" subsection of Appendix A. This property would be a distributive property except that multiplication rather than AND is used to combine the binary term with the basically continuous term as follows:

$$\zeta_2 = (\overline{X}_1 + X_1) X_2 \alpha_2 .$$

The second property required is that the summation of mutually exclusive binary terms is equivalent to the logical "OR" of the terms. Thus, $\overline{X}_1 + X_1 = \overline{X}_1 \oplus X_1 = 1$. Using this property and the identity property yields the final simplified equation,

$$\zeta_2 = X_2 \alpha_2 .$$

This equation merely states that the bucket cylinder should be extended with an effort proportional to α_2 only when α_2 is positive.

The basic control equation for ζ_3 involves only one hybrid term and requires no simplification; however, note that

$$\zeta_3 = X_1 \zeta_2 = X_1 \cdot X_2 \alpha_2 .$$

Karnaugh Maps

Karnaugh maps provide a graphic visualization of equations. Karnaugh maps are most useful for lengthy complex equations or when optional terms are present.

The "cells" of the map represent distinct decisional input states and the entries in the cell indicate the state modifier or how the output is effected. Like the state table or matrix, the entries can be "0" or "1" as in the familiar binary Karnaugh maps, or they can be any other variable. The maps used here may be termed hybrid Karnaugh maps since the entries are often continuous variables.

The equation represented by a Karnaugh map can be derived by grouping together the largest possible "sub cubes" comprised of cells with the same entry. The sub cube of decision states is then multiplied by its common entry. Sub cubes with the "0" entry are multiplied by "0" and thus drop out of the final equation. Sub cubes with the "1" entry represent totally binary terms. Sub cubes having continuous variable entries result in hybrid terms.

As an example, consider the hybrid Karnaugh map for a three decisional input equation with a continuous input as shown below.

X_1 \ X_2X_3	00	01	11	10
0	1	1	0	0
1	1	1	α_2	α_2

$$\zeta_1 = \overline{X}_2 + X_1 \cdot X_2 \alpha_2$$

This map contains two non-trivial sub cubes. The larger sub cube is binary which results in the term $\overline{X}_2$. The continuous sub cube represents the binary state $X_1 \cdot X_2$ modified by α_2, thus resulting in the hybrid term $X_1 \cdot X_2 \alpha_2$ in the equation shown for ζ_1. The rules for combining the terms (sub cubes) from a hybrid Karnaugh map are the same as for the state tables discussed in the previous chapter.

As a further example, the equation for ζ_2 in the backhoe problem can be derived from the map shown below.

X_1X_2	00	01	11	10
0	α_2	α_2		0

$$\zeta_2 = X_2 \alpha_2$$

Further examples are given in Appendix B.

System Block Diagrams

As an intermediate step from mathematical equation to physical implementation, the functional block diagram for the controller equations can be derived. This section is intended to introduce the nomenclature for typical blocks used in a block diagram.

A block diagram gives the connection scheme for the variables using basic functional blocks. The functional blocks represent the fundamental operations or functions. A listing of the functional blocks used here is given in Figure 2. These blocks are a combination of binary logic blocks and the standard blocks for the ordinary functions.

A functional block diagram is constructed by replacing a functional block for every operator or function in the equation. For example, the block diagram of Figure 3 represents the equations for the backhoe controller. Further examples are given in Appendix B.

Physical Implementation

The last step in the synthesis procedure is the physical implementation of the control equations. The control equations can be implemented using electronic, electrical, fluidic, fluid power, or mechanical hardware. The selection of which hardware type should be used is based upon the availability of power, method of final actuation,

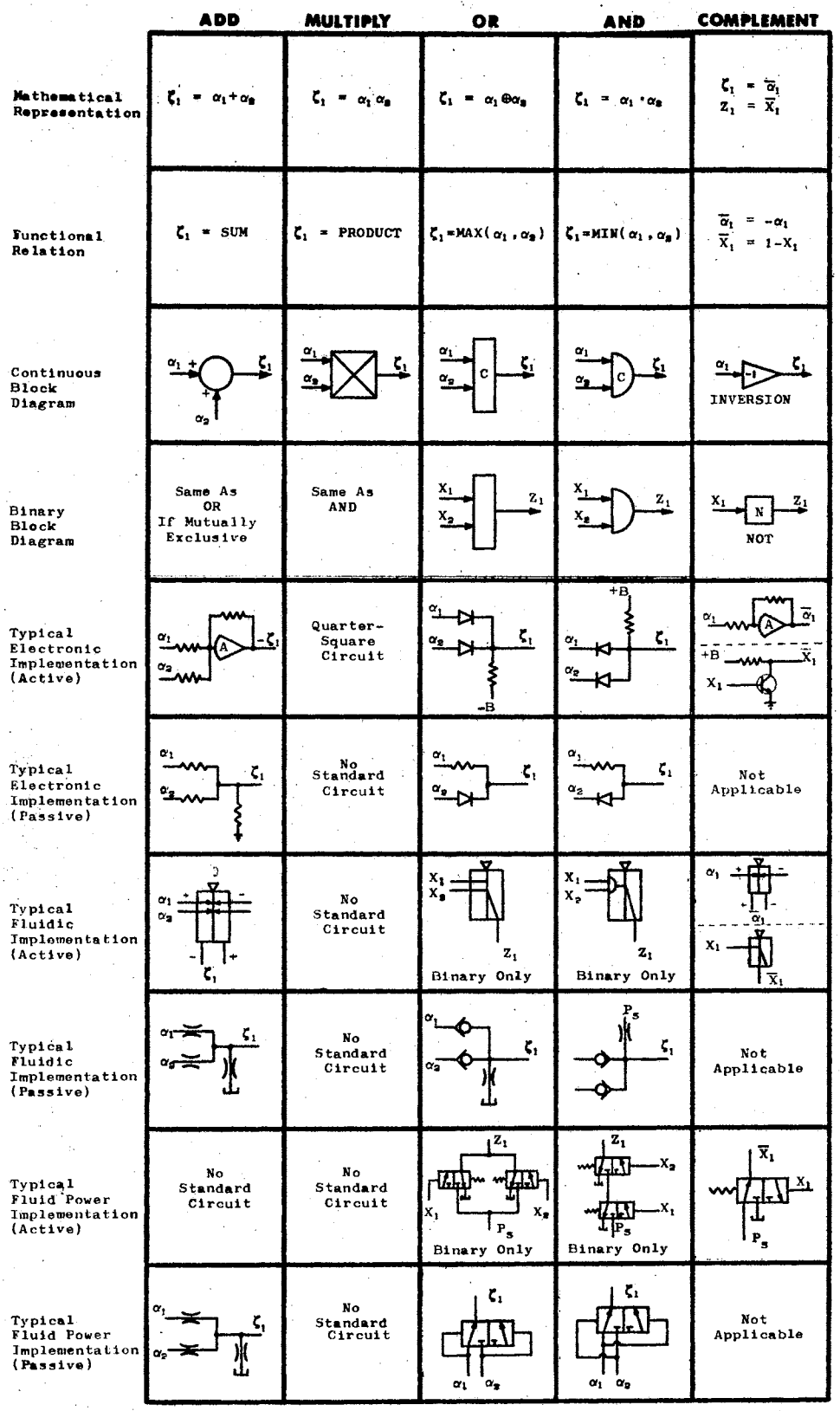


Figure 2. Functional Blocks and Implementations

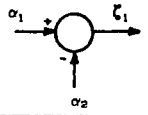
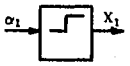
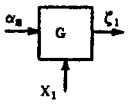
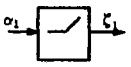
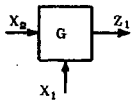
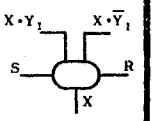
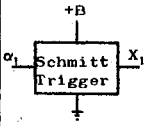
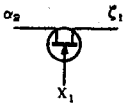
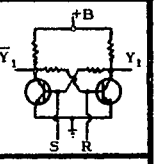
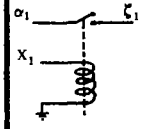
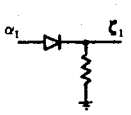
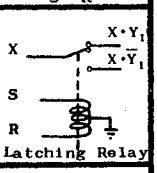
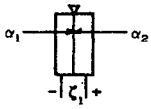
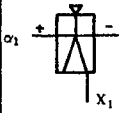
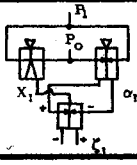
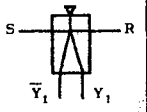
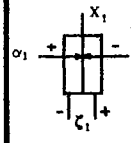
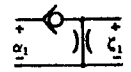
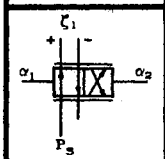
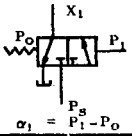
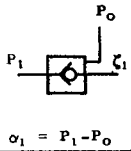
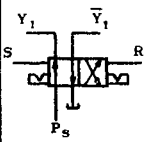
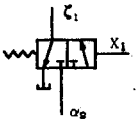
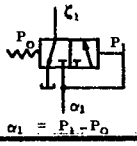
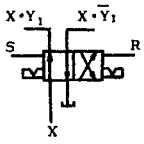
	COMPARATOR	DIGITIZER	GATE	RECTIFIER	MEMORY
Mathematical Representation	$\zeta_1 = \alpha_1 - \alpha_2$	$X_1 \sim \alpha_1$	$\zeta_1 = X_1(\alpha_2)$	$\zeta_1 = X_1 \alpha_1$	$Y_1 = S \oplus \bar{R} \cdot Y_1$ $(S \cdot R = 0)$
Functional Relation	ζ_1 = DIFFERENCE	$X_1 = 0$, $\alpha_1 \leq 0$ $X_1 = 1$, $\alpha_1 > 0$	$\zeta_1 = 0$, $X_1 = 0$ $\zeta_1 = \alpha_2$, $X_1 = 1$	$\zeta_1 = 0$, $\alpha_1 \leq 0$ $\zeta_1 = \alpha_1$, $\alpha_1 > 0$	Retains Previous State
Continuous Block Diagram					Not Applicable
Binary Block Diagram	Not Applicable	Not Applicable		Not Applicable	 Passive
Typical Electronic Implementation (Active)				No Standard Circuit	
Typical Electronic Implementation (Passive)	No Standard Circuit	Not Applicable			 Latching Relay
Typical Fluidic Implementation (Active)			No Standard Circuit		
Typical Fluidic Implementation (Passive)	No Standard Circuit	Not Applicable			No Standard Circuit
Typical Fluid Power Implementation (Active)		 $\alpha_1 = P_1 - P_o$	No Standard Circuit	 $\alpha_1 = P_1 - P_o$	
Typical Fluid Power Implementation (Passive)	No Standard Circuit	Not Applicable		 $\alpha_1 = P_1 - P_o$	

Figure 2. (Continued)

$\zeta_1 = \text{Auto}$

$\zeta_2 = X_2 \alpha_2$

$\zeta_3 = X_1 X_2 \alpha_2$

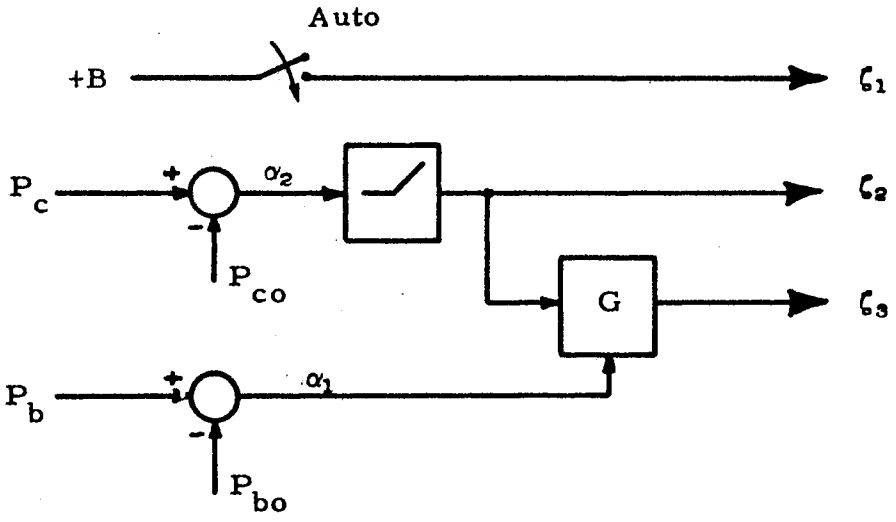


Figure 3. Block Diagram for Backhoe Controller

existance of signals, simplicity of hardware, etc. For example, if the signals exist as fluid signals and a power cylinder is to be used for the final actuation, then it might be preferable to use fluidic or fluid power components in the controller.

Figure 2 shows typical implementation configurations for electronic, fluidic, or fluid power hardware. Active and passive schemes are shown. Passive devices usually reduce hardware and power requirements.

The functions mentioned in Figure 2 represent only the very basic function types. Many physical devices have multi-function capabilities. For example, the rectifier function represents the functions of a digitizer and a gate. A bistable fluidic amplifier performs the functions of a comparator and a digitizer. There are many other devices which have combined functional properties which are too numerous to mention here. Examples of multi-function devices are given in Appendix B.

The basic method for implementation is to substitute a physical element for each block in the controller block diagram. When possible, multi-function devices can be used to reduce hardware. Most functions have various implementation configurations within a hardware type so that the designer has a choice of how to implement the function.

Since hydraulics is used throughout the rest of the system, a fluid power controller would seem to be a natural choice for implementation hardware. Figure 4 gives the

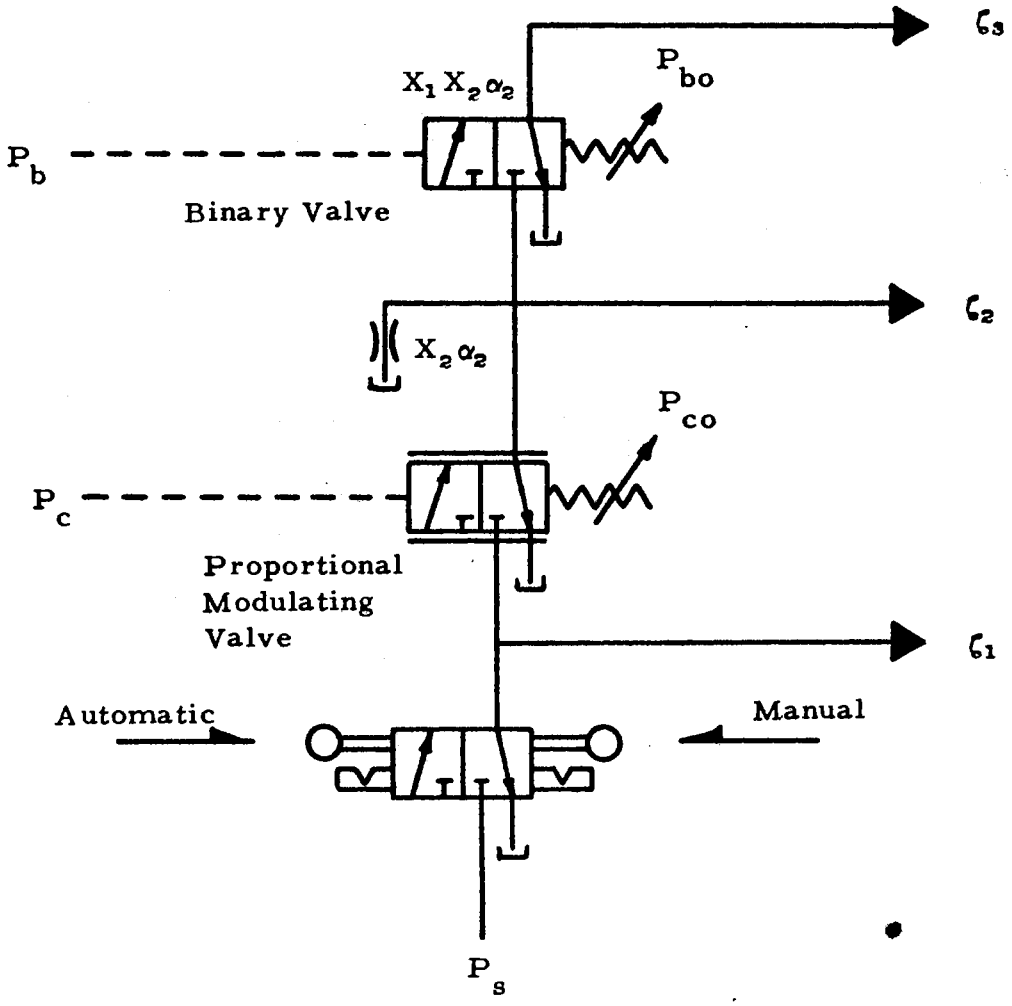


Figure 4. Fluid Power Implementation of Controller

fluid power implementation of the controller. This circuit is identical to the binary control circuit derived by Caywood (17) except that a proportional modulating valve rather than a binary valve is used to produce $X_2\alpha_2$. A fluidic implementation for this controller is shown by Figure 5. Figure 6 is an electronic implementation.

Specification of the implementing circuit completes the synthesis procedure. One point which should be mentioned is the relation between physical variables and signal variables. The logic or signal variables are scaled representations of the physical variables. The logic and computation is done with the signal variables. These signal variables must then be rescaled back up to actuation levels. This is considered to be part of the analysis step in the controller development.

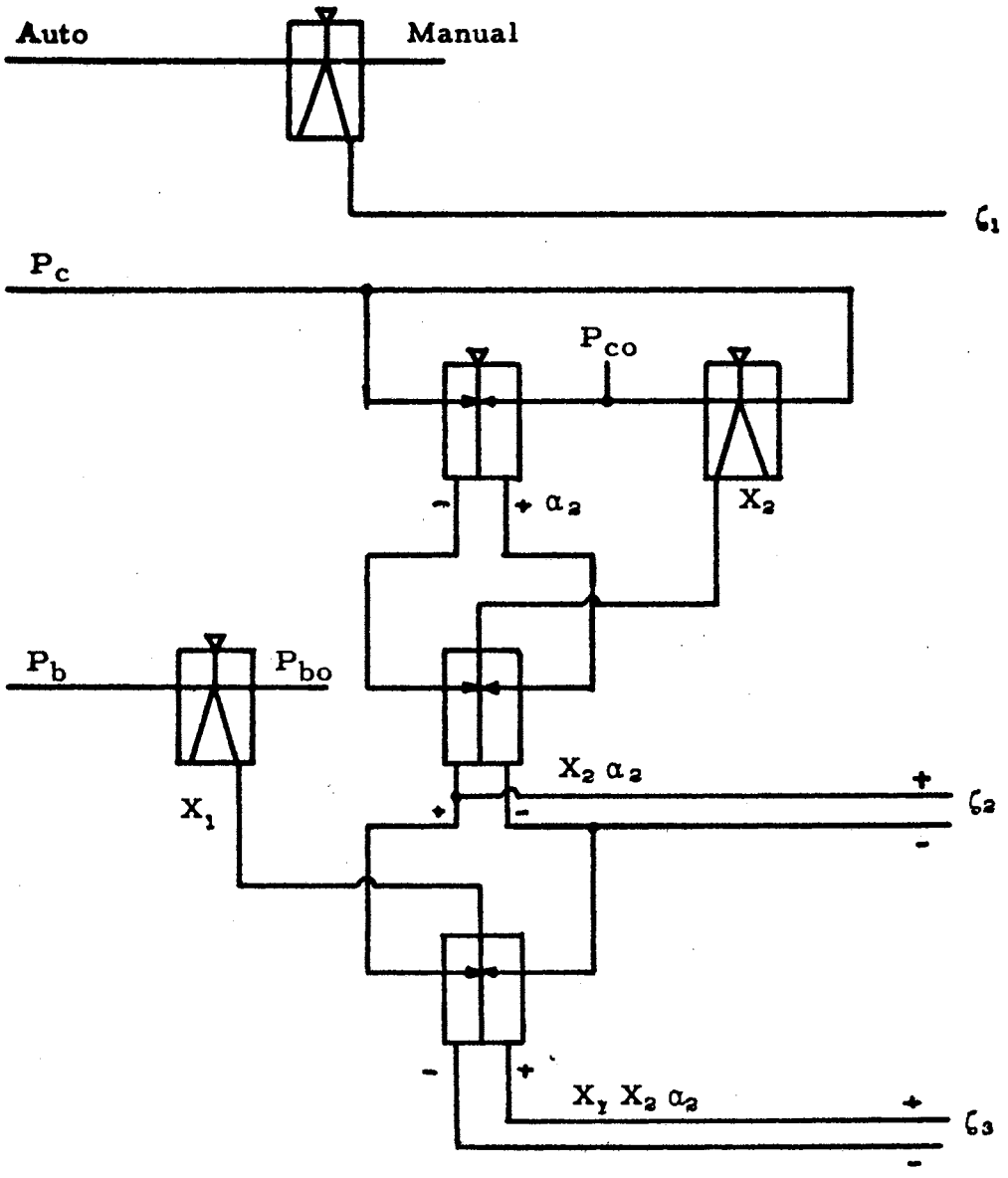


Figure 5. Fluidic Implementation of Controller

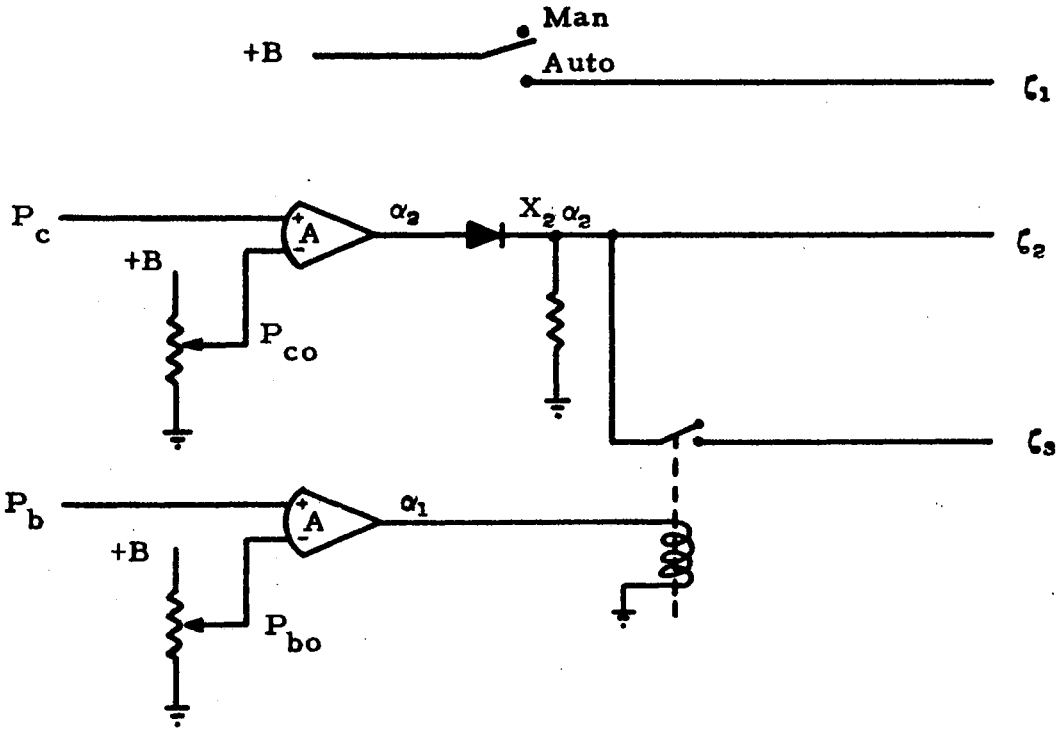


Figure 6. Electronic Implementation of Controller

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

The synthesis procedure is illustrated in the preceeding chapters by means of a simple but meaningful example. A designer likely would have little difficulty in synthesizing the controller in this example by intuition alone. However, as system complexity increases, the procedure presented in this dissertation allows an organized approach to synthesis, whereas an intuitive approach would be exceedingly difficult.

The procedure presented is euqally applicable for use in the systematic synthesis of control systems having totally decisional variables (digital), totally continuous (proportional), or both decisional and continuous (hybrid) variables.

Other principle features of the procedure are:

1. The trial and error design process normally associated with hybrid or proportional control systems is eliminated.
2. A priori knowledge of the circuit configuration is not required.
3. Implementation is allowed with any available type

of hardware having the required functional characteristics.

4. A majority of the steps in the procedure can be mechanized in a digital computer program.

Recommendations for Further Study

The synthesis procedure presented here has very definite steps to follow. Most of these steps are mechanized and could be performed by a digital computer program. Programs for digital synthesis have been written which accept information from a primitive flow table and derive sequential logic equations representing that table (18). A similar program should be written which accepts continuous inputs as state modifiers. A better form for a program would be to have more elementary input information. A sequence of basic if-then conditionals could be used as input information and the program itself could set up a primitive flow table. The input conditionals could contain the state modifier.

The listing of physical implementing devices is not complete. Numerous multi-function devices are not mentioned. Efforts should be made to compile a more complete listing of the functional capabilities of existing devices. This list should include mechanical logic devices. The impedance characteristics of physical devices should be considered. The logical difference between differential signals and single line signals often used in fluid circuits

should be properly noted.

Reference is made to dynamic logic in Chapter III. Dynamic logic involves the entry of dynamic operators in the matrix or state table to describe the desired dynamic response. This concept should be studied to determine the possible merits of such a technique. The forward loop dynamics must be considered while synthesizing a controller loop. The relationship between optimal control theory and the proposed dynamic logic should be established.

A SELECTED BIBLIOGRAPHY

(1) Shannon, C. E. "A Symbolic Analysis of Relay and Switching Circuits," Trans. AIEE, Vol. 57 (1938).

(2) Huffman, D. A. "The Synthesis of Sequential Switching Circuits," Journal of the Franklin Institute, Vol. 257, No. 3 (1954).

(3) Moore, E. F., ed. Sequential Machines: Selected Papers. Reading, Massachusetts: Addison-Wesley, 1964.

(4) Marcus, M. P. Switching Circuits for Engineers. London: Prentice-Hall International, 1968.

(5) Fitch, E. C. Fluid Logic. (unpub. manuscript, Stillwater, Oklahoma: School of Mechanical and Aerospace Engineering, 1966).

(6) Ackermann, R. J. An Introduction to Many-Valued Logics. New York: Dover Publications, 1967.

(7) Rosser, J. B. and Turquette, A. R. Many-Valued Logics. Amsterdam: North-Holland Publishing Co., 1952.

(8) Zinov'en, A. A. Philosophical Problems of Many-Valued Logic. Dordrecht, Holland: D. Reidel Publishing Co., 1963.

(9) Givone, D. D. and Snelsire, R. W. "The Design of Multiple-Valued Logic Systems." New York: State University of New York, AD 671 677, 1968.

(10) Johnson, D. L. "The Application of Four-Valued Boolean Algebra to Switching Circuits." (Ph. D. thesis, Stillwater; Oklahoma State University, 1957).

(11) Lukasiewicz, J. and Tarski, A. "Untersuchungen Uber Den Aussagenkalkul." Comptes Rendus Des Séances de la Société des Sciences et des Letters de Varsovie. Warsaw, Poland, Classe III, Vol. 23 (1930).

(12) McNaughton, R. "A Theorem About Infinite-Valued Sentential Logic." Journal of Symbolic Logic, Vol. 16, No. 1 (1951).

(13) Schaefer, D. H. "A Rectifier Algebra." Trans. of the AIEE, Vol. 73, pt. I (1955).

(14) Wilkinson, R. H. "A Method of Generating Functions of Several Variables Using Analog Diode Logic." IEEE Trans. on Electronic Computers, Vol. 12 (1963).

(15) Ginzburg, S. A. "Logical Method of Synthesizing Function Generators." Proceedings of the International Federation of Automatic Control, Vol. 4 (1961).

(16) Ginzburg, S. A. "Continuous Logic and Its Application." Automation and Remote Control, Vol. 28, pt. 1 (1967).

(17) Caywood, J. A. "Digital Controls Help Develop Hydraulics for Backhoe." Hydraulics and Pneumatics, October, 1969.

(18) Woods, R. L. "The State Matrix Method for the Synthesis of Digital Logic Systems." (M.S. thesis, Stillwater: Oklahoma State University, 1970).

(19) Reid, K. N., Woods, R. L., and Gunda, R. Application of Fluidics In the Control of Military Vehicle Propulsion Systems. Final Report to TACOM, U. S. Army Tank Automotive Command, Stillwater, Oklahoma: School of Mechanical and Aerospace Engineering, to be published 1971.

(20) Cole, J. H. "Synthesis of Optimum Complex Fluid Logic Sequential Circuits." (Ph. D. thesis, Stillwater: Oklahoma State University, 1968).

APPENDIX A

THE ALGEBRA OF HYBRID LOGIC

APPENDIX A

THE ALGEBRA OF HYBRID LOGIC

The algebra used here involves four binary operators and two unary operators with variables having more than one set of limits; consequently, it is necessary to formally establish the rules and properties for this algebra. This appendix gives the formal formulation of this algebraic system. Simplified concepts are presented in the text so as to lessen the reader's problem of understanding the synthesis concepts.

Statement Of The Algebra

An abstract algebra consists of elements, relations, operators, definitions, and postulates and theorems (refered to here as properties). Each of the above items is presented in detail below.

Elements

The set of real numbers α such that $-B \leq \alpha \leq B$. The variable α will usually be normalized such that $B=1$; but in general $B<\infty$.

Relations

Equality: $=$
Greater than: $>$ Greater than or equal to: $\geq$
Less than: $<$ Less than or equal to: $\leq$

Operations

$\oplus \sim$ Logical "OR" $\alpha_1 \oplus \alpha_2 = MAX(\alpha_1, \alpha_2)$
$\cdot \sim$ Logical "AND" $\alpha_1 \cdot \alpha_2 = MIN(\alpha_1, \alpha_2)$
$+ \sim$ Addition $\alpha_1 + \alpha_2 =$ Sum of α_1 and α_2
Juxtaposition $\sim$ Multiplication $\alpha_1 \alpha_2 =$ Product of α_1 and α_2

Definitions

D.1 Greek letters indicate continuous variables whose value is any real number.

D.2 Roman letters indicate binary variables whose value is restricted to 0 (zero) and 1 (one). The subscript of a binary variable always corresponds to the subscript of a continuous variable and is defined as follows:

$$X_i = 1 \Rightarrow \alpha_i > 0$$
$$= 0 \Rightarrow \alpha_i \leq 0$$

D.3 A and B are the lower and upper limits of the variable α such that $A \leq \alpha \leq B$. These limits are symmetrical in the case of continuous variables and are 0 and 1 for binary variables.

Properties of the Algebra

Arithmetic Operations

P.1 Associative
a. $(\alpha_1 + \alpha_2) + \alpha_3 = \alpha_1 + (\alpha_2 + \alpha_3)$
b. $(\alpha_1 \alpha_2) \alpha_3 = \alpha_1 (\alpha_2 \alpha_3)$

P.2 Commutative
a. $\alpha_1 + \alpha_2 = \alpha_2 + \alpha_1$
b. $\alpha_1 \alpha_2 = \alpha_2 \alpha_1$

P.3 Distributive (Jux. Over +)
a. $\alpha_1 (\alpha_2 + \alpha_3) = \alpha_1 \alpha_2 + \alpha_1 \alpha_3$

P.4 Identity Elements
a. Additive identity is 0. Such that $\alpha+0=\alpha$
b. Multiplicative identity is 1, Such that $\alpha 1=\alpha$
The identity elements are unique.

P.5 Inverse Operation
a. Additive inverse for α is $-\alpha$.
Such that $\alpha+(-\alpha)=\alpha-\alpha=0$
b. Multiplicative inverse for α is $\frac{1}{\alpha}$.
Such that $\alpha(\frac{1}{\alpha})=\frac{\alpha}{\alpha}=1$

Logical Operations

P.6 Associative
a. $(\alpha_1 \oplus \alpha_2) \oplus \alpha_3 = \alpha_1 \oplus (\alpha_2 \oplus \alpha_3)$
b. $(\alpha_1 \cdot \alpha_2) \cdot \alpha_3 = \alpha_1 \cdot (\alpha_2 \cdot \alpha_3)$

P.7 Commutative
a. $\alpha_1 \oplus \alpha_2 = \alpha_2 \oplus \alpha_1$
b. $\alpha_1 \cdot \alpha_2 = \alpha_2 \cdot \alpha_1$

P.8 Distributive ($\cdot$ over $\oplus$ and $\oplus$ over $\cdot$)
a. $\alpha_1 \cdot (\alpha_2 \oplus \alpha_3) = (\alpha_1 \cdot \alpha_2) \oplus (\alpha_1 \cdot \alpha_3)$
b. $\alpha_1 \oplus (\alpha_2 \cdot \alpha_3) = (\alpha_1 \oplus \alpha_2) \cdot (\alpha_1 \oplus \alpha_3)$

P.9 Idempotent
a. $\alpha_1 \oplus \alpha_1 = \alpha_1$
b. $\alpha_1 \cdot \alpha_1 = \alpha_1$

P.10 Complement
General Definition
a. $\overline{\overline{\alpha}}_1 = A+B-\alpha_1$
b. $\overline{\overline{\alpha}}_1 = \alpha_1$
Continuous Variables (INVERSION)
c. $\overline{\alpha}_1 = -\alpha_1$
Binary Variables (NOT)
d. $\overline{X}_1 = 1-X_1$

P.11 Absorption
a. $\alpha_1 \oplus (\alpha_1 \cdot \alpha_2) = \alpha_1$
b. $\alpha_1 \cdot (\alpha_1 \oplus \alpha_2) = \alpha_1$

P.12 De Morgan
a. $\overline{\alpha_1 \oplus \alpha_2} = \overline{\alpha}_1 \cdot \overline{\alpha}_2$
b. $\overline{\alpha_1 \cdot \alpha_2} = \overline{\alpha}_1 \oplus \overline{\alpha}_2$

P.13 Identity Elements
General
a. $\alpha_1 \oplus A = \alpha_1$
b. $\alpha_1 \cdot B = \alpha_1$
c. $\alpha_1 \oplus B = B$
d. $\alpha_1 \cdot A = A$

Binary Variables

e. $X_1 \oplus 0 = X_1$

f. $X_1 \cdot 1 = X_1$

g. $X_1 \oplus 1 = 1$

h. $X_1 \cdot 0 = 0$

P.14 Inclusion

Continuous Variables

a. $\alpha_1 \oplus \overline{\alpha}_1 = |\alpha_1|$

b. $\alpha_1 \cdot \overline{\alpha}_1 = -|\alpha_1|$

Binary Variables

c. $X_1 \oplus \underline{X}_1 = 1$

d. $X_1 \cdot X_1 = 0$

Combined Operations

P.15 Distributivity (+ over $\oplus$ and + over $\cdot$)

a. $\alpha_1 + (\alpha_2 \oplus \alpha_3) = (\alpha_1 + \alpha_2) \oplus (\alpha_1 + \alpha_3)$

b. $\alpha_1 + (\alpha_2 \cdot \alpha_3) = (\alpha_1 + \alpha_2) \cdot (\alpha_1 + \alpha_3)$

P.16 Distributivity (Jux. over $\oplus$ and Jux. over $\cdot$)

a. $\alpha_1 (\alpha_2 \oplus \alpha_3) = X_1 (\alpha_1 \alpha_2 \oplus \alpha_1 \alpha_3) + \underline{X}_1 (\alpha_1 \alpha_2 \cdot \alpha_1 \alpha_3)$

b. $\alpha_1 (\alpha_2 \cdot \alpha_3) = X_1 (\alpha_1 \alpha_2 \cdot \alpha_1 \alpha_3) + \overline{X}_1 (\alpha_1 \alpha_2 \oplus \alpha_1 \alpha_3)$

P.17 Complement

a. $\overline{\alpha_1 + \alpha_2} = \overline{\alpha}_1 + \overline{\alpha}_2$

b. $\overline{\alpha_1 \alpha_2} = \overline{\alpha}_1 \alpha_2 = \alpha_1 \overline{\alpha}_2$

Hybrid Operations

P.18 Hybrid Combination

a. $X_1 \cdot X_2 \alpha_3 = (X_1 \cdot X_2) \alpha_3 \neq X_1 \cdot (X_2 \alpha_3)$

P.19 Symmetrical Addition

a. $X_1 \alpha_1 + \overline{X}_1 \alpha_1 = \alpha_1$

P.20 Sign Function

a. $\underline{X}_1 - \overline{X}_1 = \text{SIGN}(\alpha_1)$ Where: $\text{SIGN}(\alpha) = 1 \Rightarrow \alpha > 0$

b. $X_1 - X_1 = \text{SIGN}(-\alpha_1)$ $= -1 \Rightarrow \alpha \leq 0$

P.21 Complement

a. $\underline{X}_1 \alpha_2 = X_1 \overline{\alpha}_2$ Note: α_2 could be α_1

P.22 Symmetrical Complement

a. $\widetilde{X}_1 \alpha_1 = \overline{X}_1 \alpha_1 = \alpha_1 - X_1 \alpha_1$

Notes On The Algebra

This section is presented in order to give further details on the terms and concepts presented in the statement of the algebra.

Operators

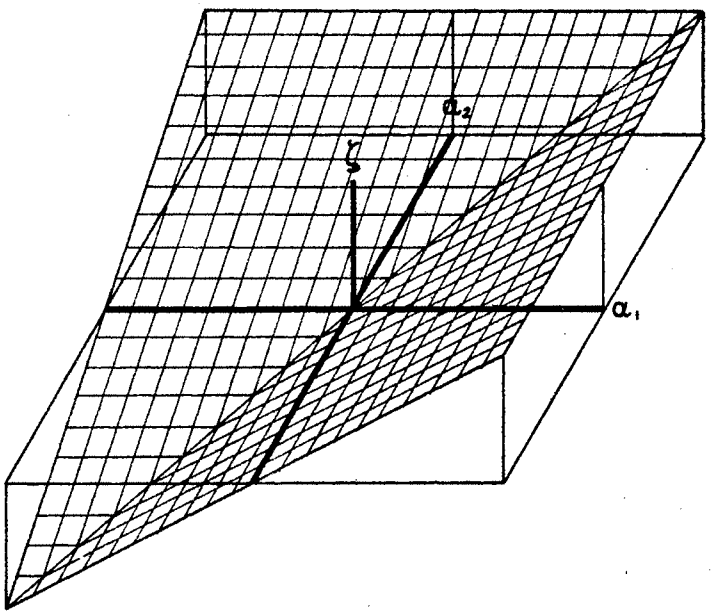
This algebra is a result of the combination of the two ordinary operators, ADD and MULTIPLY, and two logical operators OR and AND. The basic operational definitions for continuous logic are:

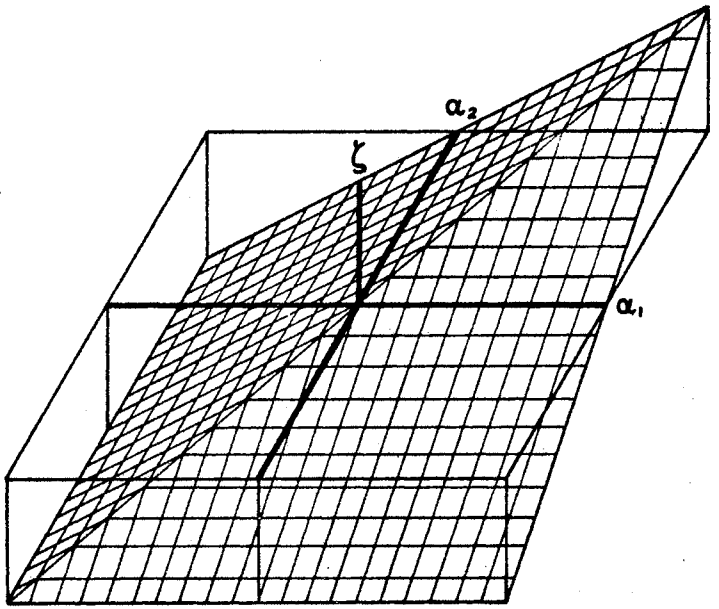
Logical OR: $\alpha_1 \oplus \alpha_2 = MAX(\alpha_1, \alpha_2)$
Logical AND: $\alpha_1 \cdot \alpha_2 = MIN(\alpha_1, \alpha_2)$
Complement: $\overline{\alpha}_1 = -\alpha_1$

The logical OR is the operator which takes the greater of the truth values and the logical AND takes the lesser truth value (or greater falsity). The complement represents the opposite truth value. Since the limits of truth values are $-B \leq \alpha \leq B$, the neutral truth value is at zero. Thus, the complement is the negative of the truth value.

When working with the logical operators it is quite often helpful to refer to graphical representation of the functions. This is especially true when working with hybrid equations. Figure 7 is a graphical representation of the two logical operators. Notice that these two surfaces are the combination of two planes, $\zeta_1 = \alpha_1$ and $\zeta_1 = \alpha_2$. The functions are the maximum and minimum of the two planes.

Figure 8 illustrates the ordinary functions of ADD and MULTIPLY. These functions should be quite familiar





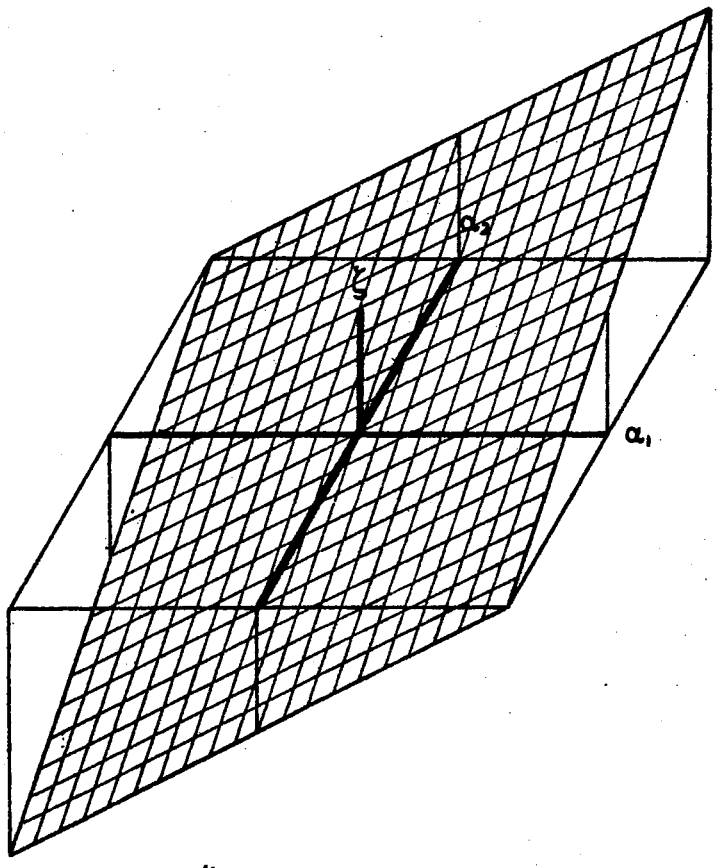
$\zeta = \alpha_1 \oplus \alpha_2$

$= MAX(\alpha_1, \alpha_2)$

$\zeta = \alpha_1 \cdot \alpha_2$

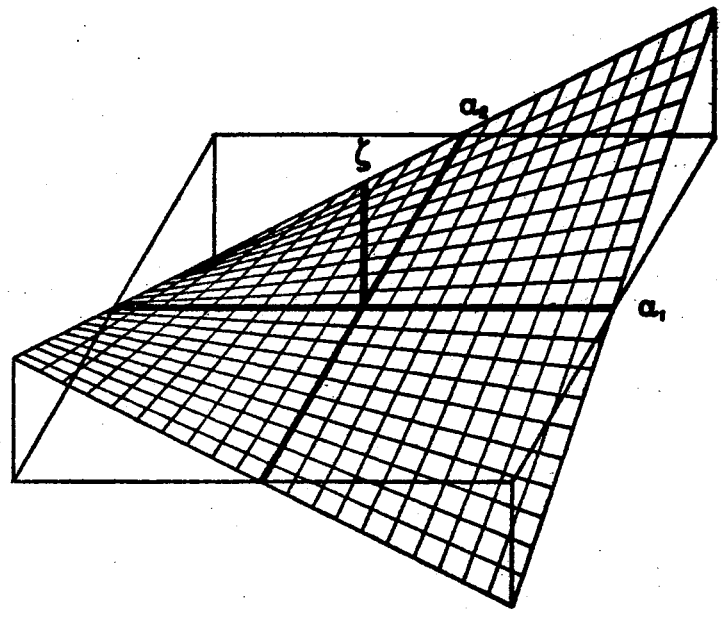
$= MIN(\alpha_1, \alpha_2)$

Figure 7. Logical Operators OR and AND



$\zeta = \alpha_1 + \alpha_2$

= SUM



$\zeta = \alpha_1 \alpha_2$

= PRODUCT

Figure 8. Algebraic Operators ADD and MULTIPLY

looking. Notice the similiarity of OR and ADD, and AND and multiply in the first quadrant, especially along the boundaries (i.e., $\alpha_i=0$).

Figure 9 illustrates α versus α and $\overline{\alpha}$ versus α. Single variable graphs such as these are quite helpful when working with hybrid functions and are introduced here for illustration.

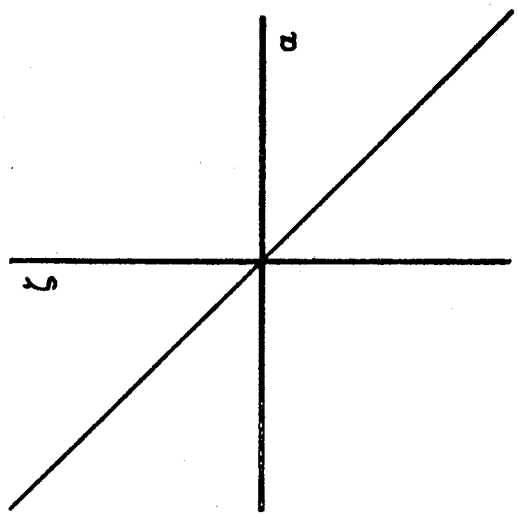
Arithmetic Properties

Properties P.1 through P.5 are the ordinary arithmetic properties. Subtraction and division are treated as inverses rather than separate operators.

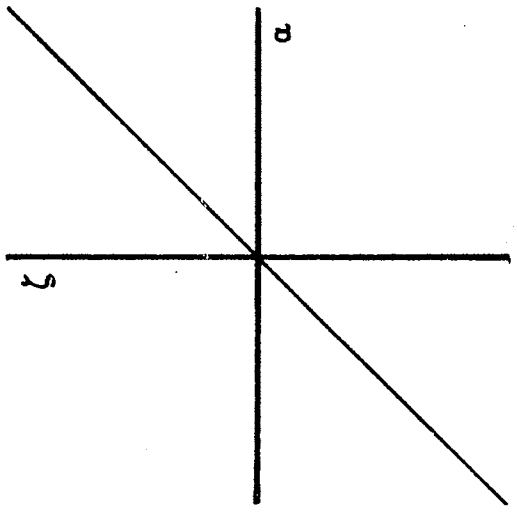
One additional property which should be mentioned here is that the addition of mutually exclusive binary terms is the same as "OR" (i.e., $X_1X_2+X_1\overline{X}_2 = X_1X_2 \oplus X_1\overline{X}_2 = X_1$, etc.).

Logical Properties

The properties P.6 through P.14 are a combination and complete listing of the properties stated in the literature for continuous logic. Most of the properties are stated using continuous variables. These properties are valid also for binary variables unless otherwise noted. The reason for differing properties is due to the different logical limits for continuous ($-B\leq\alpha\leq B$) and binary ($X=0$ or 1) variables.



$\zeta = \bar{a}$



$\zeta = a$

Figure 9. Single Variable COMPLEMENT

Combined Properties

The combined operator properties are not as useful as the others, but are noted since they do involve the two groups of operators. See P.15 through P.17.

Hybrid Properties

The hybrid properties stated here do not appear in the literature since the concept of a combined logic algebra using symmetrical continuous and binary variables is not reported. The basic form of a hybrid term is a pure binary term multiplying a continuous term as shown in P.18.

The binary term cannot be regrouped arbitrarily with the continuous term. The hybrid term should be thought of as being a continuous term gated by a binary term. The gating is noted by multiplication rather than "AND".

The concept of symmetry is used repeatedly and its meaning should be noted. The most basic example of symmetrical functions is shown in Figure 10.

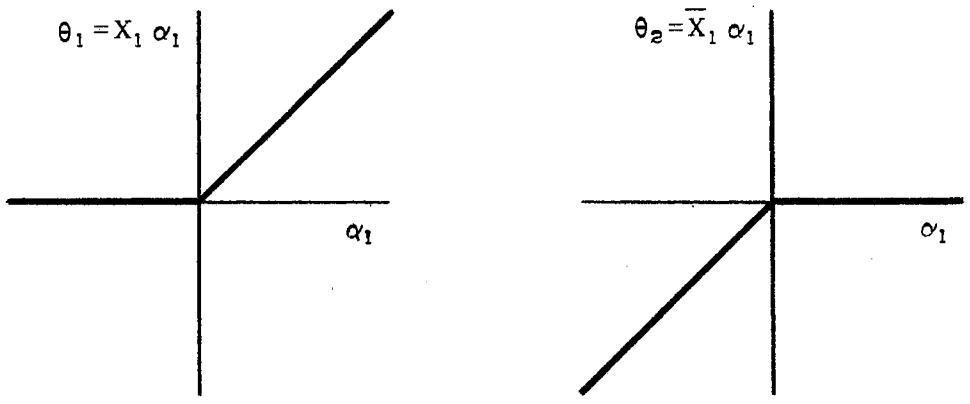


Figure 10. Symmetrical Functions

As shown by Figure 10 the functions $X_1 \alpha_1$ and $\overline{X}_1 \alpha_1$ are symmetrical since the function for positive arguments is the same as the function for negative arguments. For example, in the functions of Figure 10 $\theta_1(X_1 \alpha_1) = \Phi(\alpha_1)$ and $\theta_2(\overline{X}_1 \alpha_1) = \Phi(\alpha_1)$ where $\Phi(\alpha_1) = \alpha_1$. As another example consider

$$\theta_1 = X_1 \alpha_1 + X_2 \alpha_2$$

$$\theta_2 = \overline{X}_1 \alpha_1 + \overline{X}_2 \alpha_2$$

then $\Phi = \alpha_1 + \alpha_2$.

In general, symmetrical functions have the property that adding the two functions for positive and negative arguments yields the function valid for both positive and negative arguments (i.e., $\theta_1 + \theta_2 = \Phi$, recall from above that $X_1 \alpha_1 + \overline{X}_1 \alpha_1 = \alpha_1$).

The symmetrical complement is defined to give the symmetrical portion of a function. The general definition of the symmetrical complement (tilda bar) yields:

$$\widetilde{\theta_1} = \theta_2 = \Phi - \theta_1 .$$

As an example of the general symmetrical complement, consider the equation for ζ_1^+ in solution two of the controller for the hydrostatic transmission in Appendix B. This equation describes the control policy for positive errors. The symmetrical control policy for negative errors, ζ_1^-, can by found by the following:

$$\zeta_1^- = \widetilde{\zeta_1^+} = \zeta_1 - \zeta_1^+$$

where, $\zeta_1^+ = \overline{X}_1 \cdot X_2 \alpha_2 + X_1 \cdot X_3 \alpha_3$

$$\zeta_1 = \overline{X}_1 \alpha_2 + X_1 \alpha_3 .$$

Thus, $\zeta_1^- = (\bar{X}_1 \alpha_2 + X_1 \alpha_3) - (\bar{X}_1 \cdot X_2 \alpha_2 + X_1 \cdot X_3 \alpha_3)$

$= \bar{X}_1 (\alpha_2 - X_2 \alpha_2) + X_1 (\alpha_3 - X_3 \alpha_3)$

$= \bar{X}_1 \cdot \bar{X}_2 \alpha_2 + X_1 \cdot \bar{X}_3 \alpha_3 .$

The last equation above is the same as the one given in Appendix B.

APPENDIX B

EXAMPLE SYNTHESIS PROBLEMS

APPENDIX B

EXAMPLE SYNTHESIS PROBLEMS

This appendix discusses the problem description and complete solution of various synthesis problems. It is hoped that these problems exemplify the various types of problems that can be solved using this technique.

Single Variable Search Equations

This problem is applicable to a computer program in which a search for the maximum value of a function is to be made. A single variable function is considered here; however, a multivariable function could also be considered.

Problem Statement

The maximum value of a unimodal function of a single variable is to be found in a general purpose computer program. This is done by an iterative process of searching. From a known value of the independent variable, a step is taken and the corresponding value of the function is determined. The new function value is then compared to the previous one to determine whether there was an increase or decrease in the function value. An increase in the function value indicates the step of the independent

variable was in the correct direction. If this is true, further steps in this direction should lead to the peak. A decrease in the function value indicates the peak has been passed and the increment was taken in the wrong direction.

As an additional convergence feature, the rate of the functional variations (i.e., the second difference of the function) is considered. For example, a positive first difference in the function and a negative second indicates the function is beginning to peak and small proportional steps should be taken; whereas, a positive first and a positive second difference indicates that the curve is increasing rapidly as opposed to converging and large steps should be taken. Similar reasoning is applied to the negative first difference. Both directions of the independent variable search must be considered.

The problem is to derive the equations which determine the magnitude and direction of the step increment in the independent variable according to the above philosophy.

Problem Variables

The problem variables include:

$J(S)$ = The function to be maximized.

S = Independent variable.

ΔS_m = Maximum increment allowed in S.

Logic Variables

The logic variables are defined by the problem variables. Both decisional and continuous variables are necessary. The variables which determine the decisions to be made are:

$X_1 \sim \Delta J = J_n - J_{n-1}$

$X_2 \sim \Delta^2 J = \Delta J_n - \Delta J_{n-1}$

$X_3 \sim \Delta S = S_n - S_{n-1}$

$X_i = 1$ if the argument is positive and $X_i = 0$ if the argument is zero or negative. The continuous variable is normalized and defined in the following manner:

$$\alpha_1 = \frac{\Delta J}{\Delta J_{max}}$$

The equation to be synthesized is the magnitude and direction, ζ_1, of the increment.

$\zeta_1 > 0 \Rightarrow$ Increase S

$\zeta_1 < 0 \Rightarrow$ Decrease S

such that $S_{n+1} = S_n + \Delta S_m \zeta_1$.

Solution

The possible conditions with the decision variables are illustrated by Figure 11.

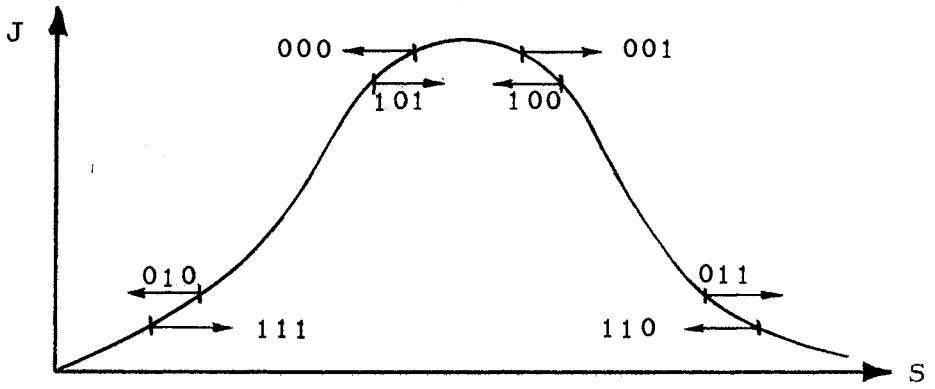


Figure 11. Function to be Maximized

Stated explicitly, the increment should be chosen as follows:

1. If $\Delta J>0$ and $\Delta^2 J>0$ for $\Delta S>0$, a large step should be taken. This is the "111" condition shown in the graph.

2. Progressing in this direction will hopefully lead to the condition $\Delta J>0$ and $\Delta^2 J<0$ for $\Delta S>0$ as indicated by the "101" shown. In this case the peak is almost reached and smaller steps proportional to ΔJ should be taken (ie. if ΔJ is small, the increment should be small, etc.).

3. If the peak is overshot for some reason, the direction should be reversed and appropriate size steps should be taken. (ie. Proportional steps in the opposite direction for "001" and large steps for "011").

4. Similiar reasoning applies for $\Delta S<0$. (ie. Continue moving to the left proportionally for "100" ect.).

There are eight possible conditions as discussed above. These are shown in the state table below.

$X_1 X_2 X_3$	ζ_1	(Direction)
111	1	→
101	α_1	→
001	α_1	↩
011	-1	↩
110	-1	←
100	$-\alpha_1$	←
000	$-\alpha_1$	↪
010	1	↪

The output equation for the magnitude and direction of the step is derived by combining all of the above conditionals.

$$\zeta_1 = X_1 \cdot X_2 \cdot X_3 + X_1 \cdot \overline{X}_2 \cdot X_3 \alpha_1 + \overline{X}_1 \cdot \overline{X}_2 \cdot X_3 \alpha_1 - \overline{X}_1 \cdot X_2 \cdot X_3$$
$$-X_1 \cdot X_2 \cdot \overline{X}_3 - X_1 \cdot \overline{X}_2 \cdot \overline{X}_3 \alpha_1 - \overline{X}_1 \cdot \overline{X}_2 \cdot \overline{X}_3 \alpha_1 + \overline{X}_1 \cdot X_2 \cdot \overline{X}_3$$

By property P.10 of Appendix A, $-\alpha_1 = \overline{\alpha}_1$. Using P.7 and P.8 to collect terms on X_3:

$$\zeta_1 = X_3 [\, X_1 \cdot X_2 + X_1 \cdot \overline{X}_2 \alpha_1 + \overline{X}_1 \cdot \overline{X}_2 \alpha_1 - \overline{X}_1 \cdot X_2 \,]$$
$$+\overline{X}_3 [-X_1 \cdot X_2 + X_1 \cdot \overline{X}_2 \overline{\alpha}_1 + \overline{X}_1 \cdot \overline{X}_2 \overline{\alpha}_1 + \overline{X}_1 \cdot X_2 \,].$$

By P.7 and P.8:

$$\zeta_1 = X_3 [X_2 (X_1 - \overline{X}_1) + \overline{X}_2 \cdot (X_1 + \overline{X}_1) \alpha_1 \,]$$
$$+\overline{X}_3 [X_2 (\overline{X}_1 - X_1) + \overline{X}_2 \cdot (X_1 + \overline{X}_1) \overline{\alpha}_1 \,].$$

Using P.14 and P.20:

$$\zeta_1 = X_3 [X_2 \mathrm{SIGN}(\alpha_1) + \overline{X}_2 \alpha_1 \,]$$
$$+\overline{X}_3 [X_2 \mathrm{SIGN}(\overline{\alpha}_1) + \overline{X}_2 \overline{\alpha}_1 \,].$$

Note in the above that multiplication replaces "AND" whenever regrouping requires that continuous or non-binary variables be combined with a binary term.

The last equation is the desired equation for the step size and direction.

Controller For Hydrostatic Transmission

This problem is worked in two ways. In the second method additional decision variables are introduced. Algebraic simplification reduces the equation such that the additional decision variables are eliminated from the equations. A technique such as this can be used to help solve totally proportional synthesis problems.

Problem Statement

A load is driven by an engine using a hydrostatic transmission. The hydrostatic transmission has a speed ratio continuously adjustable from zero to infinity. The infinitely variable transmission provides a proper match between engine and load. Using the adjustable speed ratio, the engine can be forced to operate at various points for the same load. From laboratory tests, it has been determined that the engine operates most desirably and efficiently along one certain curve. This optimum operation curve can be approximated by the two straight lines shown in Figure 12. This figure is a plot of engine manifold vacuum versus engine speed showing constant throttle contours. The dashed lines show the desired engine operation.

The problem is to synthesize a controller which will automatically adjust the speed ratio of the transmission to whatever it must be to force engine operation along this optimum curve. Reference 19 gives a complete development and continuous solution to this problem.

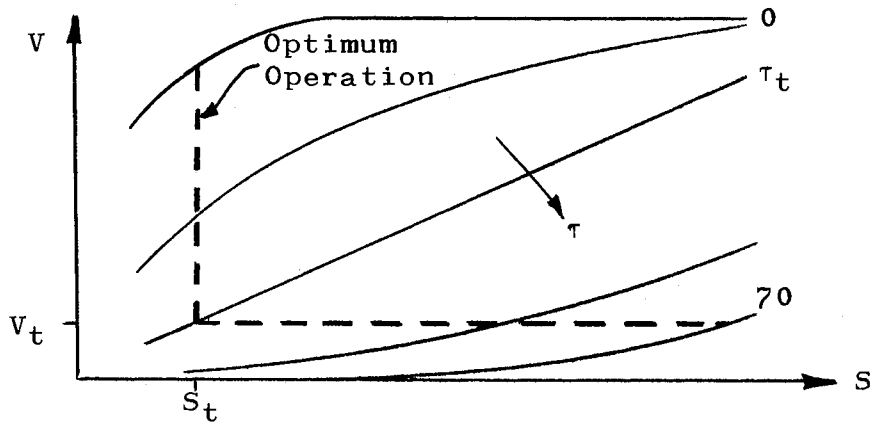


Figure 12. Vacuum-Speed Map for Typical I.C. Engine

This is accomplished by monitoring the throttle position to determine whether the engine should be controlled along a constant speed or a constant vacuum. As shown by Figure 12, the engine is controlled along a line of constant speed, S_t, for $0 \leq \tau \leq \tau_t$. For $\tau > \tau_t$ the engine is controlled at a constant vacuum, V_t. In the constant speed range, an overspeed error requires an increase in speed ratio to load the engine down. The speed ratio should be increased at a rate proportional to the percent overspeed and decreased proportional to the underspeed. In the constant vacuum range, the speed ratio is adjusted at a rate proportional to the vacuum error (i.e., increase for overvacuum, decrease for undervacuum).

Physical Variables

There are four variables of interest to the controller. These are:

τ ~ Throttle position (degrees) $0 \leq \tau \leq 70$

S ~ Engine speed (RPM) $0 \leq S \leq 3500$

V ~ Manifold vacuum (in. Hg) $0 \leq V \leq 15$

R_s ~ Speed ratio $0 \leq R_s \leq \infty$

The following three constants or thresholds must be defined.

τ_t ~ The value of throttle position at which the control policy switches between constant speed and constant vacuum.

S_t ~ The optimum engine speed for the constant speed portion of operation.

V_t ~ The optimum manifold vacuum for the constant vacuum portion of operation.

Logic Variables (Solution One)

This problem solution has only one decision variable, the throttle position.

$X_1 \sim \tau - \tau_t$ $\qquad X_1 = 0 \Rightarrow \tau \leq \tau_t$

$\qquad\qquad\qquad\qquad\quad = 1 \Rightarrow \tau > \tau_t$

The continuous inputs are defined as follows:

$$\alpha_2 = \frac{S - S_t}{3500}$$

$$\alpha_3 = \frac{V - V_t}{15}$$

The control output to be synthesized is the rate (and direction) at which the speed ratio is changed.

$$\zeta_1 = \frac{d}{dt} R_s$$

$\zeta_1 > 0 \Rightarrow$ Increase speed ratio

$\zeta_1 < 0 \Rightarrow$ Decrease speed ratio

Solution One

For this solution a one decisional input state table

is formed as shown below.

X_1	ζ_1
0	α_2
1	α_3

From this table the output equation can be derived.

$$\zeta_1 = \overline{X}_1 \alpha_2 + X_1 \alpha_3$$

These two terms are mutually exclusive; when X_1 is OFF the control will be α_2, and when X_1 is ON the control will be α_3. Notice that ζ_1 can be either positive or negative according to the sign of α_2 or α_3.

Solution Two

For the second solution, the problem will be further decomposed by considering the two continuous inputs as decisional variables. This is done to illustrate that even if it is not known beforehand that the control policy is symmetrical, the problem can still be solved. The answer reduces to the equation given in solution one.

The three decision variables are:

$X_1 \sim \alpha_1$

$X_2 \sim \alpha_2$

$X_3 \sim \alpha_3$

Where the continuous variables are defined as:

$$\alpha_1 = \frac{\tau - \tau_t}{70}$$

$$\alpha_2 = \frac{S - S_t}{3500}$$

$$\alpha_3 = \frac{V - V_t}{15}$$

As an example of this decomposition consider the case where $\tau \leq \tau_t(X_1=0)$. If there is an overspeed the speed ratio should be increased. If there is an underspeed the speed ratio should be decreased. Thus, the over or underspeed condition is treated as a decision input. The problem is further decomposed by synthesizing two outputs, one to increase the speed ratio and one to decrease the speed ratio. Accordingly the outputs are defined below.[1]

$\zeta_1^+ > 0 \Rightarrow$ Increase speed ratio (ζ_1^- must be zero)

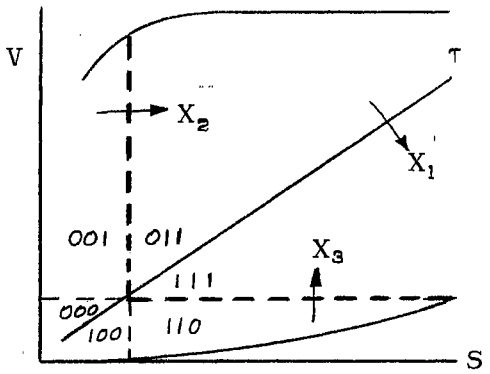
$\zeta_1^- < 0 \Rightarrow$ Decrease speed ratio (ζ_1^+ must be zero)

Using the simplified graph shown below a state table for the possible conditions can be derived. Three decision states are inaccessable. Accordingly the outputs for these states are optional.

---

[1]The notation ζ_1^+ and ζ_1^-, although cumbersome, is carefully chosen to have significance. The reason for this notation goes back to the notation often used in fluid logic. Reports on fluid logic (5, 18, 20) use Z_1 for the extend signal and $\overline{Z}_1$ for the retract signal. This is highly misleading for some types of sequential circuits in which the cylinder remains in the last position actuated. In this type of circuit the extend and retract signals may both be "OFF". Thus $\overline{Z}_1$, is not the complement of Z_1. Since the extend and retract signals are somewhat independent, the notation Z_1 and Z_2 is often used for extend and retract of cylinder 1. But then the subscripts do not always correspond to the cylinder number. For this reason, the notation Z_1^e and Z_1^r or better, Z_1^+ and Z_1^- could be used. This is the motivation behind using ζ_1^+ for the "increase speed ratio" signal and ζ_1^- for the "decrease speed ratio" signal. This allows the subscript to correspond to one output variable and the superscript to represent the sign of the variable.

$X_1 X_2 X_3$	ζ_1^+	ζ_1^-
000	0	α_2
001	0	α_2
011	α_2	0
010	-	-
110	0	α_3
111	α_3	0
101	-	-
100	0	α_3



The optional terms in the state table can best be treated by the use of Karnaugh maps. Substituting continuous variables for the optional terms in the ζ_1^+ map yields:

X_1 \ X_2X_3	00	01	11	10
0	0	0	α_2	α_2
1	0	α_3	α_3	0

$$\zeta_1^+ = \overline{X}_1 \cdot X_2 \alpha_2 + X_1 \cdot X_3 \alpha_3$$

Substituting zeros for the optional terms in the ζ_1^- map yields:

X_1 \ X_2X_3	00	01	11	10
0	α_2	α_2	0	0
1	α_3	0	0	α_3

$$\zeta_1^- = \overline{X}_1 \cdot \overline{X}_2 \alpha_2 + X_1 \cdot \overline{X}_3 \alpha_3$$

A close examination of these two equations reveals the similiarity of terms. The existence of the terms $X_2 \alpha_2$ and $X_3 \alpha_3$ insures that ζ_1^+ is always positive. The terms $\overline{X}_2 \alpha_2$ and $\overline{X}_3 \alpha_3$ make ζ_1^- always negative. These factors might lead one to suspect that ζ_1^+ and ζ_1^- are symmetrical. This can be tested by adding the two together. Of course in this case, it is known that the two are symmetrical. Thus:

$$\zeta_1 = \zeta_1^+ + \zeta_1^-$$
$$= (\overline{X}_1 \cdot X_2 \alpha_2 + X_1 \cdot X_3 \alpha_3) + (\overline{X}_1 \cdot \overline{X}_2 \alpha_2 + X_1 \cdot \overline{X}_3 \alpha_3)$$

By P.7 and P.8:

$$\zeta_1 = \overline{X}_1 (X_2 \alpha_2 + \overline{X}_2 \alpha_2) + X_1 (X_3 \alpha_3 + \overline{X}_3 \alpha_3)$$

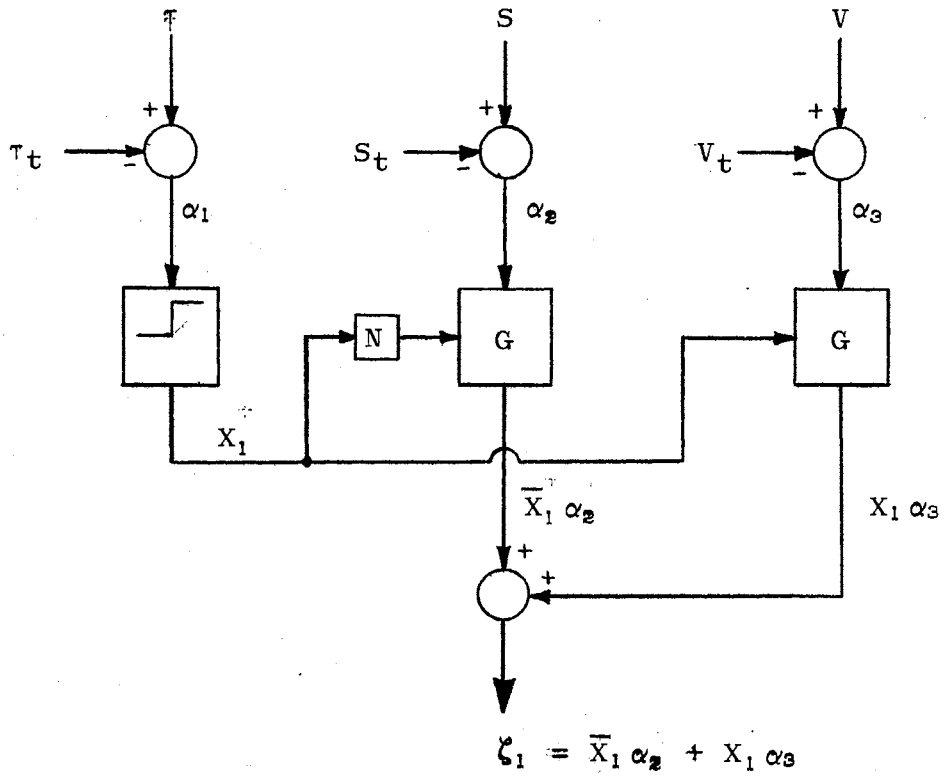
Using P.19 yields:

$$\zeta_1 = \overline{X}_1 \alpha_2 + X_1 \alpha_3$$

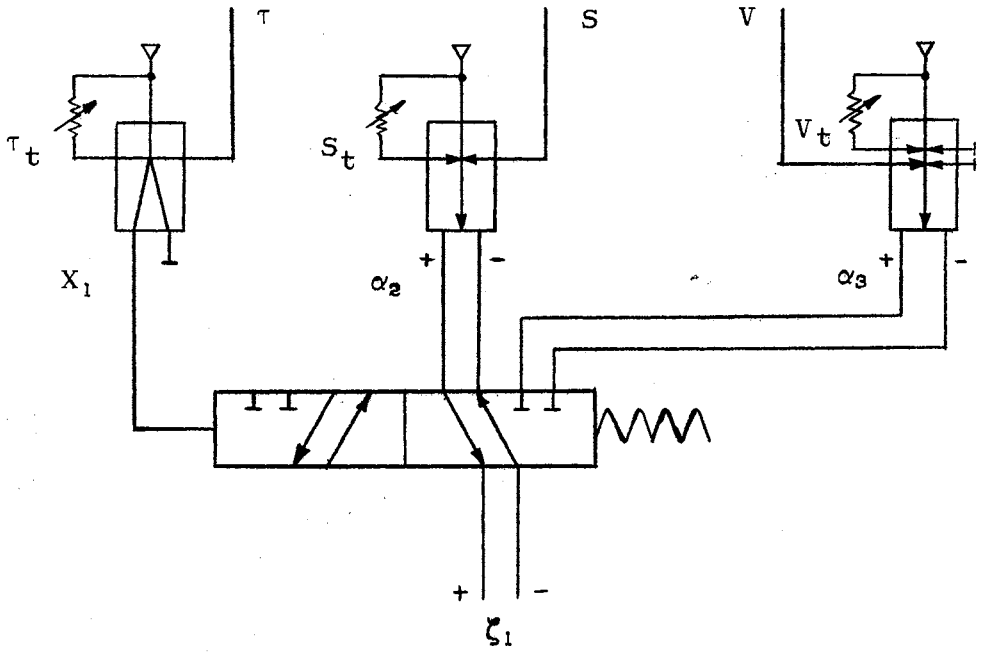
This is the same equation derived in the first solution. Notice that the decision variables X_2 and X_3 have been eliminated from the final equation.

Implementation

Manifold vacuum exists as a fluidic signal and engine speed can be fluidically sensed easily. Hydraulic power for actuation is available in the hydrostatic transmission. Thus, it appears that a fluidic implementation would be most practical. A fluidic implementation for this controller is given by Figure 13.



BLOCK DIAGRAM USING BASIC FUNCTIONS



FLUIDIC IMPLEMENTATION

Figure 13. Hydrostatic Transmission Controller

Proportional Rate Milling Machine

This problem introduces the use of a primitive flow table for a sequential synthesis problem using continuous variables.

Problem Statement

The machine to be controlled is a vertical milling machine having a table which can traverse in either direction. Under normal operation the automatic controller should drive the table back and fourth at a constant rate with automatic reversal of crossfeed direction. If the bit experiences an overload, it is desired that the crossfeed rate be reduced so that the bit can cut slower to reduce the load. The feed rate should be decreased in proportion to the overload.

Physical Variables

The physical system and variables are illustrated by Figure 14.

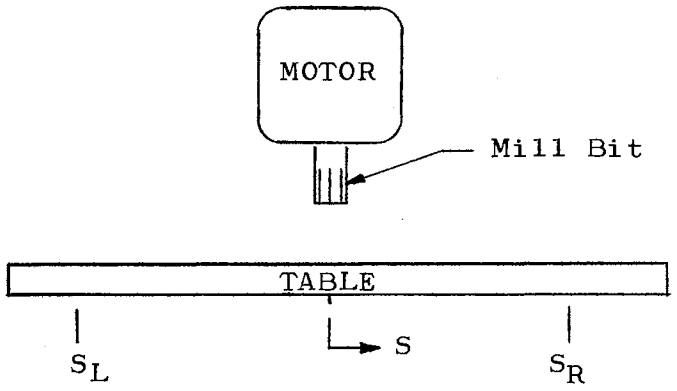


Figure 14. Milling Machine

The loading on the mill bit can be determined from either the torque or speed of the motor driving it. The torque is used in this problem to monitor the loading condition. The output characteristics of the motor are shown below. A torque greater than T_{mo} indicates an overload.

T_m ~ Motor torque

S_m ~ Motor speed

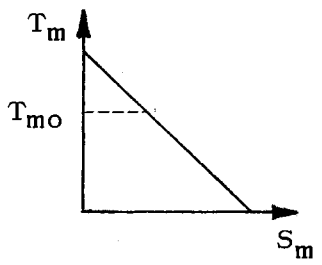
T_{mo} ~ Motor torque overload threshold

T_{max} ~ Maximum motor torque

S ~ Table position

S_L ~ Left traverse limit

S_R ~ Right traverse limit



Logic Variables

The variables which influence the operational policy of the machine are the overload indicator and the traverse limits.

$X_1 \sim T_m - T_{mo}$ $\qquad$ $X_1 = 1 \Rightarrow T_m > T_{mo}$

$X_2 \sim S_L - S$ $\qquad$ $X_2 = 1 \Rightarrow S < S_L$

$X_3 \sim S - S_R$ $\qquad$ $X_2 = 1 \Rightarrow S > S_R$

The only continuous variable of interest is the overload signal.

$$\alpha_1 = \frac{T_m - T_{mo}}{T_{max}}$$

The rate of crossfeed for an overload is reduced by the amount α_1. It is therefore convient to define the modified rate β_1 as follows:

$$\beta_1 = 1 - \alpha_1$$

The control output to be synthesized is the direction and magnitude of the crossfeed. This is broken down into two outputs as follows:

$\zeta_1^+ > 0 \Rightarrow$ Feed to the right

$\zeta_1^- > 0 \Rightarrow$ Feed to the left

Problem Solution

In this problem the sequence of events is of importance. Hence, a primitive flow table must be used. The primitive flow table for the possible sequences of events is shown below.

$X_1 X_2 X_3$ 000	001	011	010	110	111	101	100	ζ_1^-	ζ_1^+
(1)	2		-				5	0	1
3	(2)	-				6		1	0
(3)	-		4				7	1	0
1		-	(4)	8				0	1
1				-		6	(5)	0	β_1
	2				-	(6)	7	β_1	0
3				8		-	(7)	β_1	0
			4	(8)	-		5	0	β_1

As this flow table shows, memory is required. Memory can be assigned by one of many techniques. For this problem the classical or Huffman-Moore model produces the least complex equations. In the classical technique the rows of the primitive flow table must be merged together to form the merged flow table. A row can be merged with another

row if all of the stable state and transition path numbers coincide or are optional. This merged flow table reduces to two rows and is given below. The required memory is indicated to the left of this table.

	$X_1X_2X_3$ 000	001	011	010	110	111	101	100
$\overline{Y}_1$	(1)	2	-	(4)	(8)	-	6	(5)
Y_1	(3)	(2)	-	4	8	-	(6)	(7)

The switching conditions can be determined from the Karnaugh map shown below.

Y_1	$X_1X_2X_3$ 000	001	011	010	110	111	101	100
0	0	1	‡	0	0	‡	1	0
1	1	1	Θ	0	0	Θ	1	1

Set = X_3

Reset = X_2

The output equations can best be derived from a Karnaugh map since there are some optional terms. The ζ_1^- map and equation are shown below.

Y_1	$X_1X_2X_3$ 000	001	011	010	110	111	101	100
0	0	-	-	0	0	-	-	0
1	1	1	‡	‡	β_1	β_1	β_1	β_1

$$\zeta_1^- = Y_1(\overline{X}_1 + X_1\beta_1)$$

The ζ_1^+ map and equation are derived below.

Y_1	$X_1X_2X_3$ 000	001	011	010	110	111	101	100
0	1	1	1	1	β_1	β_1	β_1	β_1
1	0	0	-	-	-	-	0	0

$$\zeta_1^+ = \overline{Y}_1(\overline{X}_1 + X_1\beta_1)$$

Substituting back for β_1 yields:

$$\zeta_1^+ = \overline{Y}_1(\overline{X}_1 + X_1(1-\alpha_1))$$

$$= \overline{Y}_1(\overline{X}_1 + X_1 - X_1\alpha_1).$$

It can be seen that $X_1+\overline{X}_1 = 1$. Thus the equations reduce to the following.

$$\zeta_1^+ = \overline{Y}_1(1 - X_1\alpha_1)$$

$$\zeta_1^- = Y_1(1 - X_1\alpha_1)$$

Where Y_1 Set = X_3 ; Reset = X_2

A close examination reveals that these equations describe the desired control policies.

These symmetrical equations can be combined to form a single equation in which the sign controls the direction rather than having two equations to indicate direction. This can be done by combining the + direction and the - direction equations as follows.

$$\zeta_1 = \zeta_1^+ - \zeta_1^-$$

$$\zeta_1 = \overline{Y}_1(1-X_1\alpha_1) - Y_1(1-X_1\alpha_1)$$

By P.3:

$$\zeta_1 = (\overline{Y}_1 - Y_1)(1-X_1\alpha_1)$$

By P.20:

$$\zeta_1 = \text{SIGN}(\overline{Y}_1)(1-X_1\alpha_1)$$

Implementation

The fluid power implementation is given by Figure 15. The function $1-X_1\alpha_1$ is similiar to a "continuous NOT" and is performed by a single valve. The memory element merely reverses the direction of crossfeed.

Previous implementations have shown only the controller implementation. Figure 15 illustrates the complete system in which start up provisions are shown. A binary detent valve is used to select manual or automatic operation and a "right" and "left" direction override selectors are added for direction control. The left and right traverse limit sensors are also shown. Note that torque and pressure drop across a hydraulic motor are directly proportional.

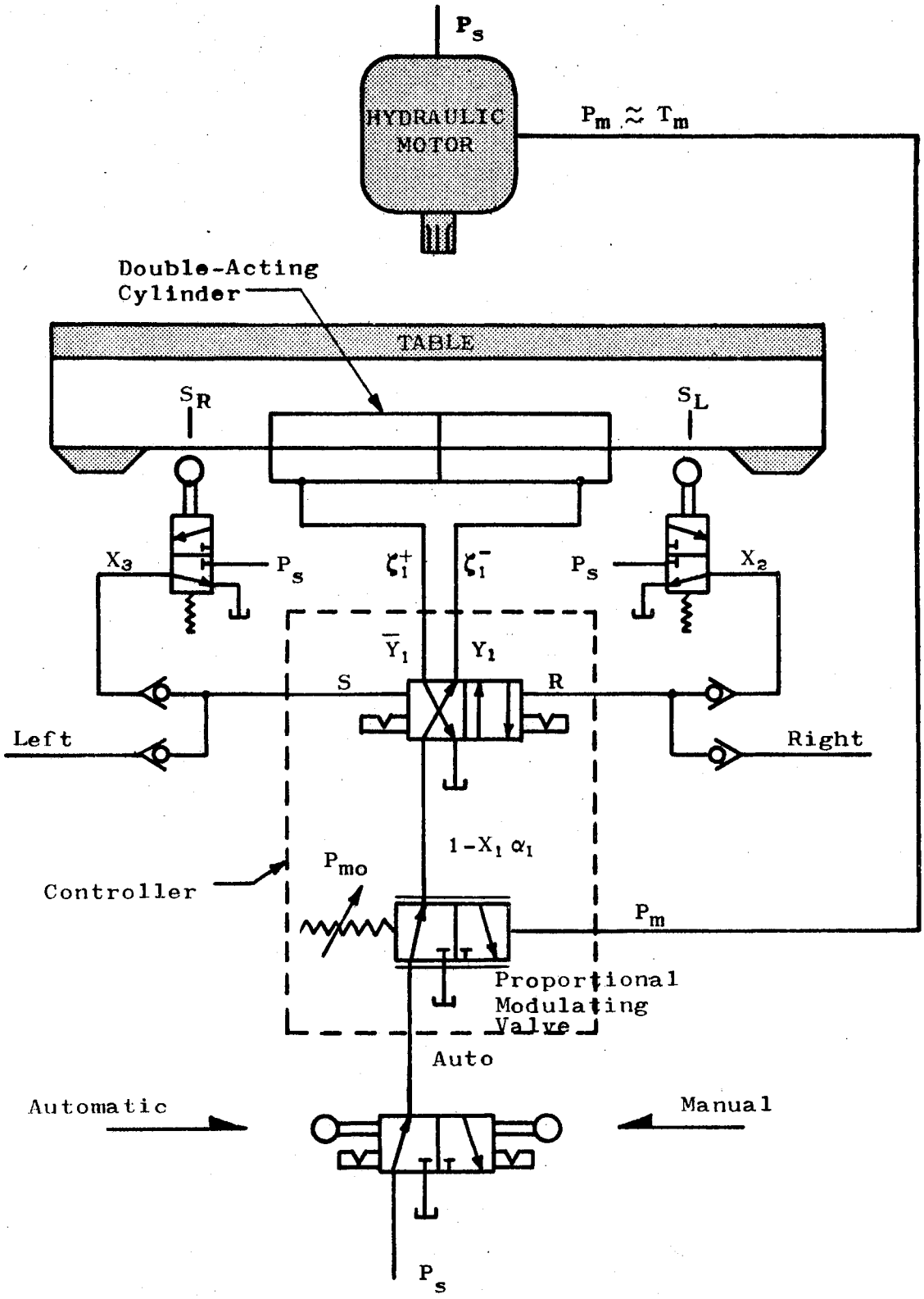


Figure 15. Hydraulic Implementation Showing Complete Milling Machine System

VITA

Robert Loren Woods

Candidate for the Degree of

Doctor of Philosophy

Thesis: THE LOGICAL SYNTHESIS OF HYBRID CONTROL SYSTEMS

Major Field: Mechanical Engineering

Biographical:

Personal Data: Born in Frederick, Oklahoma, April 14, 1945, the son of Mr. and Mrs. Aldon Woods.

Education: Graduated from Frederick High School, Frederick, Oklahoma, in May, 1963; attended Oklahoma State University from 1963 to 1965; received the Bachelor of Science degree from Southern Methodist University in 1967, with a major in Mechanical Engineering; received the Master of Science degree from Oklahoma State University in May, 1970; completed requirements for the Doctor of Philosophy degree at Oklahoma State University in July, 1971.

Professional Experience: Technical analyst, Nuclear Research Services, Dallas, Texas, 1967; graduate teaching assistant, Southern Methodist University, 1968; graduate research assistant, Oklahoma State University, 1968-71.

Professional Organizations: Member Pi Tau Sigma; member ASME.