

**Programming Manual
for
TRICE**

pb Packard Bell Computer

A SUBSIDIARY OF PACKARD BELL ELECTRONICS
1908 ARMACOST AVENUE • LOS ANGELES 25, CALIFORNIA • GRANITE 8-4247

2nd Edition

30 May 1960

CONTENTS

Section		Page
I	GENERAL DESCRIPTION	1-0
1.1	General	1-1
1.2	Purpose of Equipment	1-1
1.3	Applications	1-1
1.4	Modules	1-2
1.5	Input/Output	1-2
1.6	Size of TRICE System	1-5
1.7	Reference Data	1-5
II	PRINCIPLES OF OPERATION	2-0
2.1	General	2-0
2.2	Input and Output Mathematical Representation	2-0
2.3	Computing Modules	2-0
2.3.1	Integrator	2-0
2.3.1.1	Mathematical Operation	2-3
2.3.2	Constant Multiplier	2-7
2.3.3	Variable Multiplier	2-7
2.3.4	Digital Servo	2-11
2.3.5	Decision Servo	2-11
2.3.6	ΔY Summer	2-11
2.4	Analog-to-Digital Converter	2-12
2.5	Digital-to-Analog Converter	2-12
III	PROGRAMMING PROCEDURE	3-0
3.1	General	3-0
3.2	Analysis of Problem	3-0

CONTENTS (Cont.)

Section		Page
3. 2. 1	Initial Conditions	3-2
3. 2. 2	Limiting Values	3-4
3. 3	Mapping	3-4
3. 3. 1	Approaches to Mapping	3-4
3. 3. 1. 1	Separation of Highest Order Derivative	3-5
3. 3. 1. 2	Integration with Respect to an Independent Variable	3-7
3. 3. 1. 3	Use of Servos	3-12
3. 4	Scaling	3-28
3. 4. 1	Relation of Scaling Factors to Limiting Values of Problem	3-30
3. 4. 2	Primary Scale Powers	3-30
3. 4. 3	Secondary Scale Powers	3-30
3. 4. 4	Output Scale Powers	3-31
3. 5	Coding	3-32
3. 6	Interpretation of Results	3-33
3. 7	Special Programming Considerations	3-34
3. 7. 1	Compensating for Delays	3-35
3. 7. 2	Compensating for Round-Off	3-38
3. 7. 3	Special Programming Considerations, Example Problem	3-39
 IV	 OPERATION	 4-0
4. 1	Initial Control Positions	4-1
4. 2	System Clear Process	4-2
4. 3	Automatic Typewriter Control	4-3
4. 4	Controls	4-4

CONTENTS (Cont.)

Section		Page
4. 4. 1	Power Switches	4-4
4. 4. 2	Power Supply Controls	4-4
4. 4. 3	Input/Output Controls	4-6
4. 4. 3. 1	Analog Inputs	4-6
4. 4. 3. 2	Analog Outputs	4-6
4. 4. 3. 3	Input Switch	4-8
4. 4. 3. 4	Punch Switch	4-9
4. 4. 3. 5	Readout Scale Factor Switches	4-10
4. 4. 3. 6	Backspace Button	4-11
4. 4. 3. 7	Code Delete Button	4-11
4. 4. 3. 8	Start Reader Button	4-12
4. 4. 3. 9	Start Punch Button	4-12
4. 4. 4	Function Switching Switches	4-12
4. 4. 5	Function Switching Set and Reset Buttons	4-12
4. 4. 6	Computation Controls	4-12
4. 4. 6. 1	Address of Modules	4-13
4. 4. 6. 2	Overflow Switch and Overflow Alarms	4-13
4. 4. 6. 3	Address Button	4-14
4. 4. 6. 4	+ Button	4-14
4. 4. 6. 5	- Button	4-14
4. 4. 6. 6	Number Button	4-15
4. 4. 6. 7	Exponent Button	4-15
4. 4. 6. 8	Clear Converter Button	4-15
4. 4. 6. 9	Clear Address Button	4-15
4. 4. 6. 10	Keyboard	4-15
4. 4. 6. 11	M1 Multiverter Reset Button	4-17

CONTENTS (Cont.)

Section		Page
4. 4. 6. 12	Reset All Units Button	4-18
4. 4. 6. 13	Reset Patched Units Button	4-18
4. 4. 6. 14	Reset Single Unit Button	4-18
4. 4. 6. 15	t + /t - Button	4-18
4. 4. 6. 16	Slow Rate + Button	4-19
4. 4. 6. 17	Slow Rate - Button	4-19
4. 4. 6. 18	Compute Button	4-19
4. 4. 6. 19	Single Step Button	4-20
4. 4. 6. 20	Slow Button	4-20
4. 5	Visual Display of Register Contents	4-20
4. 6	Programming the TRICE Patchboard	4-22
4. 6. 1	Division of Patchboard	4-23
4. 6. 2	Existence Code	4-23
4. 6. 3	Sign Code	4-26
4. 6. 4	Convenience Lines	4-26
4. 6. 5	Reset Codes	4-26
4. 6. 6	Overflow Codes	4-27
4. 6. 7	Start Reader	4-27
4. 6. 8	Start Punch	4-27
4. 6. 9	Buffer	4-27
4. 6. 10	Function Switch Code	4-28
4. 6. 11	ΔY Summer Connections	4-28
4. 6. 12	Multiverter Connections	4-28
4. 7	Rules for Patchboard Connections	4-29
4. 7. 1	The A-to-D Output	4-29
4. 7. 2	Input for the D-to-A	4-29
4. 7. 3	Input for the A-to-D Connected to Operate as a D-to-A	4-30
4. 7. 4	Restrictions for Input to Destinations	4-30

CONTENTS (Cont.)

Section		Page
V	DIGITAL INTERPOLATOR	5-0
	APPENDIX	
	Example Problem I: Sine Curve	A-1
	Example Problem II: Damped Sine Wave	A-4
	Example Problem III: Cornu Spiral	A-11
	Example Problem IV: Calculation of Pi	A-15
	Example Problem V: Bessel Function	A-18

LIST OF ILLUSTRATIONS

Figure	Title	Page
1-1	TRICE	1-0
1-2	Variable Multiplier	1-3
1-3	Symbols for TRICE Modules	1-4
2-1	Symbol for Integrator	2-1
2-2	Simple Mechanization of Integrator	2-1
2-3	Graphical Representation of Rectangular Integration	2-3
2-4	Graphical Representation of Trapezoidal Extrapolative Integration	2-4
2-5	Multiplication Using Two Integrators	2-7
2-6	Rectangular Multiplication	2-8
2-7	Trapezoidal Multiplication	2-9
3-1	Map of Third Order Differential Equation	3-6
3-2	Alternate Map of Third Order Differential Equation	3-10

LIST OF ILLUSTRATIONS (Cont.)

Figure	Title	Page
3-3	Map of Simultaneous Differential Equations Using Inte- gration with Respect to Independent Variable	3-11
3-4	Digital Servo Formation of Sum	3-13
3-5	Generation of Absolute Value Using Decision Servo	3-13
3-6	Map of Saw-Tooth Functions Using Servo	3-16
3-7	Map of Clipped Sine Wave	3-17
3-8	Generation of Square Root Using Digital Servo	3-18
3-9	Map of Oscillator Voltage Loop	3-20
3-10	Map of Product	3-21
3-11	Map of Elliptical Spiral	3-22
3-12	Map Using Substitution of Variables	3-23
3-13	Map Using Decision Servo to Avoid Infinite Point	3-24
3-14	Map of Arctangent	3-25
3-15	Map of Sinh x and Cosh x	3-26
3-16	Map of v^n	3-27
3-17	Map of Orbit Problem Employing Special Programming Con- siderations	3-40
4-1	Control Panel	4-0
4-2	Power Supply Controls	4-5
4-3	Input/Output Equipment, Block Diagram	4-7
4-4	Address Block Diagram	4-24
4-5	TRICE Patchboard	4-25
A-1	Map of Sine- Cosine Curve	A-2
A-2	Map of Damped Sine Wave	A-6
A-3	Map of Cornu Spiral	A-12

LIST OF ILLUSTRATIONS (Cont.)

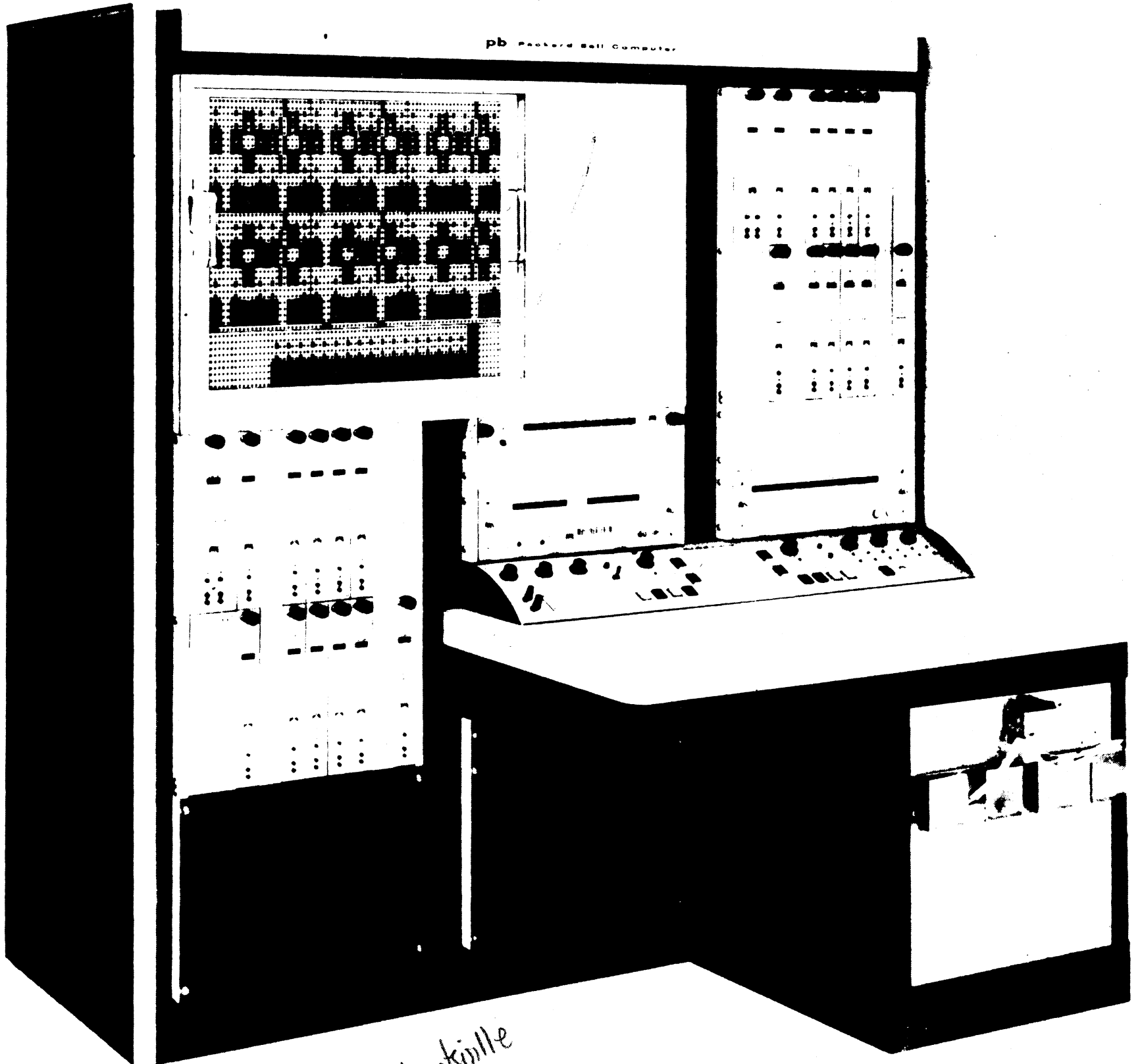
Figure	Title	Page
A-4	Map for Calculation of Pi	A-16
A-5	Map of Bessel Function	A-21

LIST OF TABLES

Table	Title	Page
1-1	Basic TRICE Frame	1-6
1-2	TRICE Modules	1-6
1-3	TRICE Accessories	1-6
3-1	Example Integrands	3-2
3-2	Octal Coding	3-32
4-1	Front Panel Controls, Initial Positions	4-1
4-2	Tape Codes	4-3
4-3	Controls Illuminated by INPUT Mode Selection	4-9
A-1	Accuracy vs. Speed Table for Sine Curve	A-3
A-2	Octal Code for Damped Sine Wave	A-10
A-3	Octal Code for Cornu Spiral	A-14
A-4	Octal Code for Calculation of Pi	A-18
A-5	Octal Code for Bessel Function	A-20

NOTE

The "Programming Manual for TRICE" is written for a comprehensive TRICE system. It will be apparent from the system configuration which portions of the manual apply to a specific system. Furthermore, since the TRICE is expandable, the additional information will apply if a small system is enlarged, or may assist in preparing expansion plans.



*NASA Huntsville
Machine #1*

Figure 1-1. TRICE

SECTION I

GENERAL INFORMATION

1. 1 GENERAL.

This manual describes the TRICE, Transistorized Realtime Incremental Computer Expandable. A typical TRICE system is shown in Figure 1-1. Subjects discussed include a general description, principles of operation, programming, and operating procedures for the TRICE.

1. 2 PURPOSE OF EQUIPMENT.

The TRICE is a completely solid-state computer that is capable of solving analytical problems by the use of digital differential analyzer (DDA) techniques. Problems to be solved by TRICE may involve mathematical terms such as:

- (1) Differentials
- (2) Integrals
- (3) Exponentials
- (4) Logarithms
- (5) Trigonometric functions
- (6) Inverse trigonometric functions
- (7) Discontinuities
- (8) Vector quantities

1. 3 APPLICATIONS.

The possible uses of TRICE in engineering and science are unlimited. A few typical TRICE applications are:

- (1) Control system stability
- (2) Autopilot response simulation
- (3) Pilot plant simulation
- (4) Missile trajectory studies
- (5) Satellite orbit parameter studies
- (6) Impact prediction
- (7) Coordinate transformation of target acquisition
- (8) Stable platform calculations
- (9) Transformation of coordinate systems for stabilizing devices,
e. g. radar
- (10) Satellite orbit prediction
- (11) Airborne guidance control

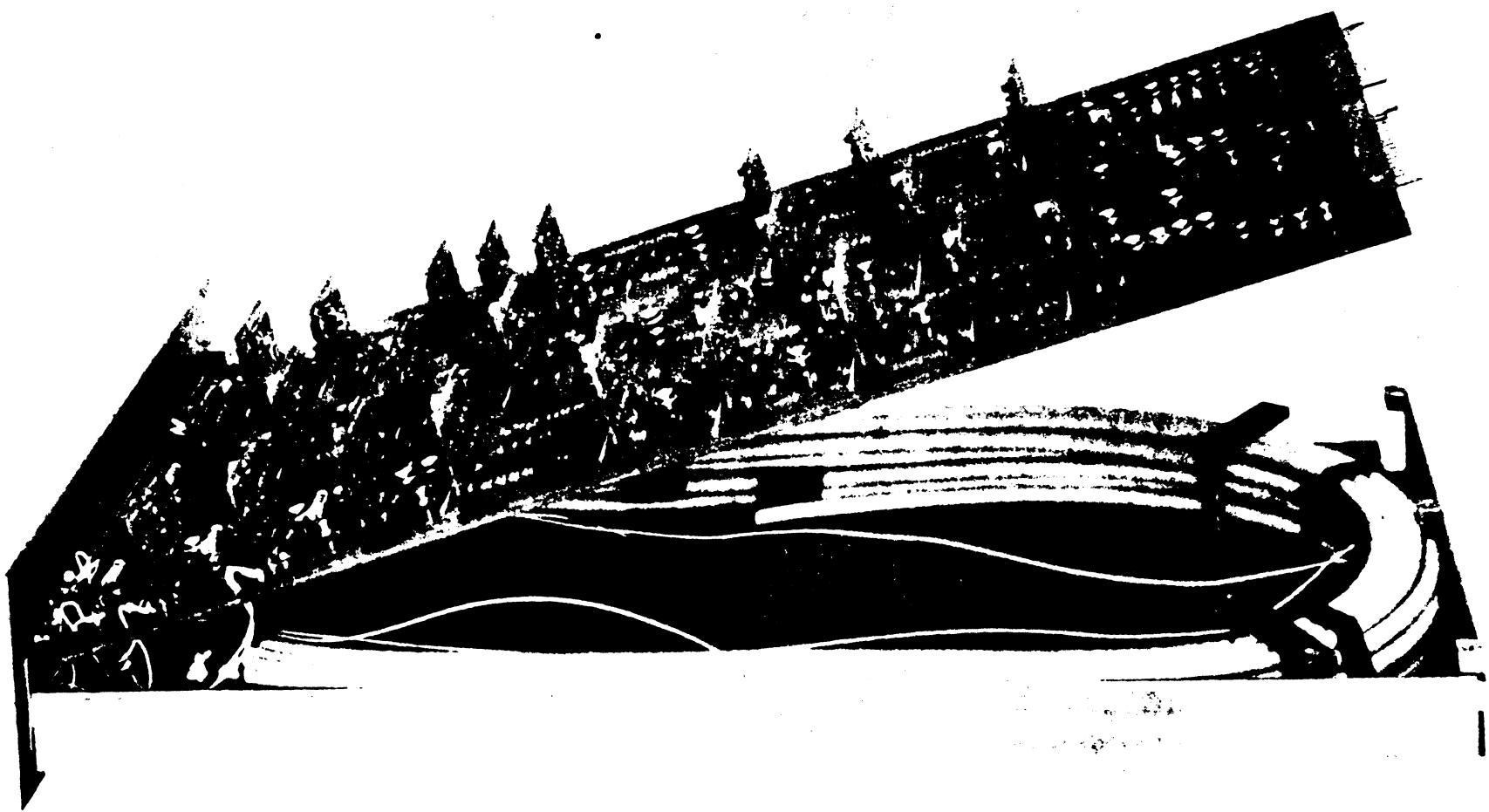
1.4 MODULES.

The TRICE system is composed of independent computing modules constructed on etched boards. These modules operate in parallel; thus, the speed of the system is not affected by the number of computing modules used. Electrical delay lines are used for memory in the various computing modules. Figure 1-2 shows the Variable Multiplier opened for access to its delay lines.

The symbols and the mathematical representations for the inputs and outputs of the TRICE computing modules are shown in Figure 1-3. These modules are interconnected by means of patchcords on a patchboard to solve problems.

1.5 INPUT/OUTPUT.

The input to TRICE may be either manual (keyboard) or automatic (tape). Analog-to-digital and digital-to-analog converters are used to convert analog information to digital form (allowing analog voltages to be an independent variable input) or vice versa (allowing the output to be in analog form).



 HACKARD BELL COMPUTER CORP

Figure 1-2. Variable Multiplier

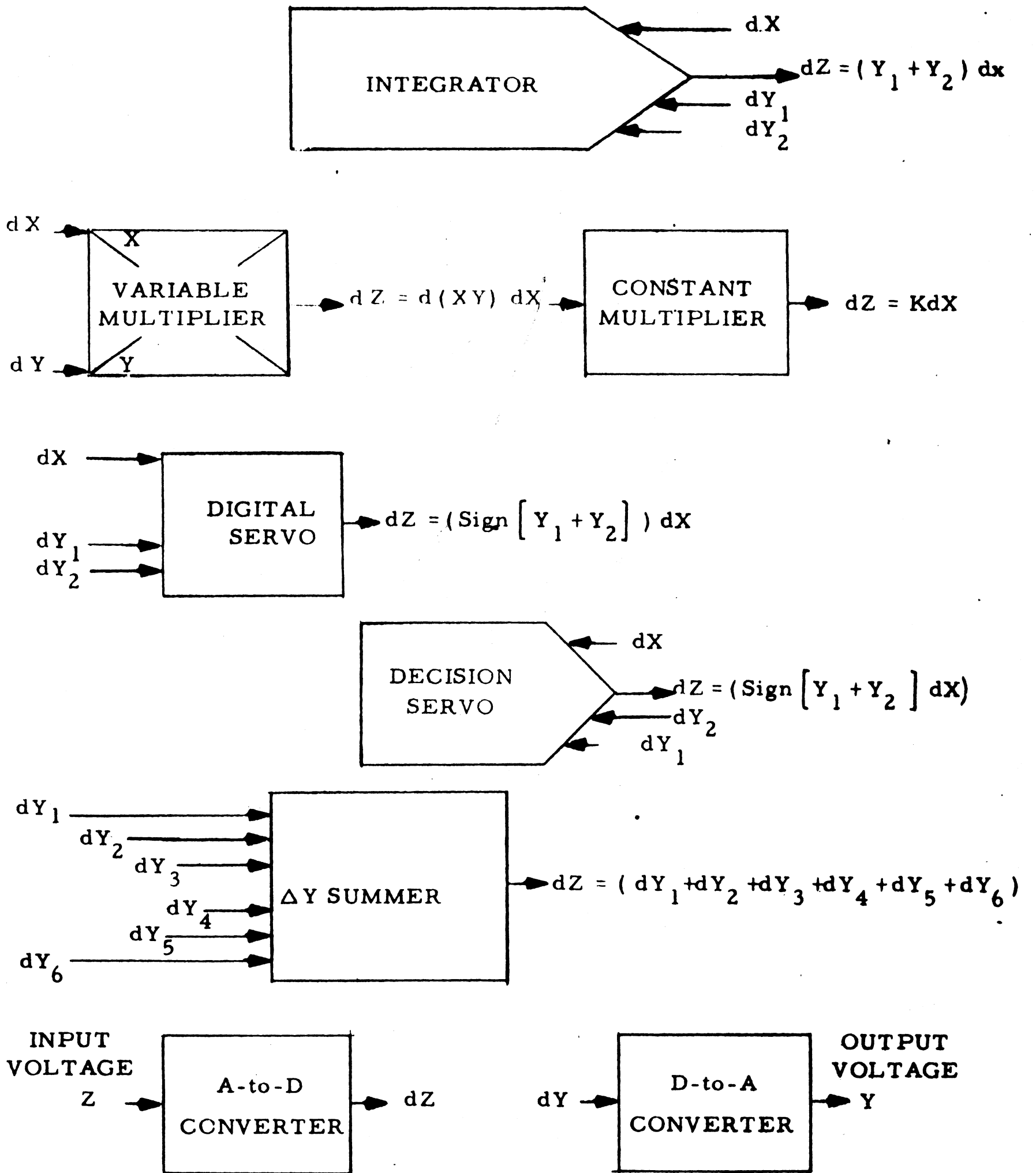


Figure 1-3. Symbols for TRICE Modules

The output may be in digital or analog form; furthermore, the output may be examined in binary or decimal using a plotter, scope, readout registers, or readout tape.

1.6 SIZE OF TRICE SYSTEM.

The size of a TRICE system is determined by the complexity of the problems to be solved. Table 1-1 shows the basic frame required for any TRICE system. The modules of TRICE are listed in Table 1-2. The accessories for TRICE that add versatility to TRICE systems are listed in Table 1-3. A TRICE system is composed of the basic equipment listed in Table 1-1 and the modules and accessories as needed from Table 1-2 and Table 1-3. It should be noted that additional equipment from Table 1-2 or Table 1-3 may be added at any future date.

1.7 REFERENCE DATA.

- (a) SIZE: The modular elements of TRICE are 12.75 inches high and 20 inches deep. With the exception of the Variable Multiplier, which is 3.50 inches wide, all TRICE modules are 1.75 inches wide.
- (b) POWER: The Integrator and ΔY Summer require 8 watts each; the Constant Multiplier and Digital Servo require 6 watts each; the Variable Multiplier requires 10 watts.
- (c) CONSTRUCTION: The modular elements are slide supported at the top. Each element can be pulled out for servicing; by means of extension boards, elements can be operated extended from the cabinet.

Table 1-1. Basic TRICE Frame

Unit
Patchbay
Patchboards
Patchcords
Control Panel
Timing and Control Unit
Readout Buffer
Power Supply *
Racks *

* Additional units necessary for expansion.

Table 1-2. TRICE Modules

Module Name	Module Name
Integrator	Digital Servo
Variable Multiplier	Decision Servo
Constant Multiplier	ΔY Summer

Table 1-3. TRICE Accessories

Input / Output Equipment
Paper Tape Input With High Speed Address System
Binary/Decimal, Decimal/Binary Converter Scaler
Paper Tape Output Unit
Off-Line Flexowriter for Tape Preparation and Printout Unit
A-to-D Converter
D-to-A Converter
Function Switching Logic

SECTION II

PRINCIPLES OF OPERATION

2.1 GENERAL.

The TRICE consists of a set of incremental computer blocks as shown in Figure 1-3. These computing elements may be interconnected on a patchboard to solve problems.

The basic computing element of TRICE is the Integrator. This element is described in detail; other computing elements are similar and are described in general. Analog-to-digital and digital-to-analog converters used with TRICE are also described.

2.2 INPUT AND OUTPUT MATHEMATICAL REPRESENTATION.

It should be recognized that ΔX , ΔY , and ΔZ are more nearly correct representations of the actual mechanization of the inputs and outputs of the TRICE modules than are dX , dY , and dZ ; however, for schematic and programming purposes, the finite Δ increments are replaced by their differential.

2.3 COMPUTING MODULES.

The following paragraphs describe the mathematical operation of each of the computing modules. The Integrator is the basic computing module and thus is described in the greatest detail.

2.3.1 Integrator. -

The Integrator evaluates an integral by summing incremental areas. The sum of these incremental areas approximates the area under the curve. The

accuracy of the integration is dependent upon the size of the increments used. The symbol for an Integrator is shown in Figure 2-1.

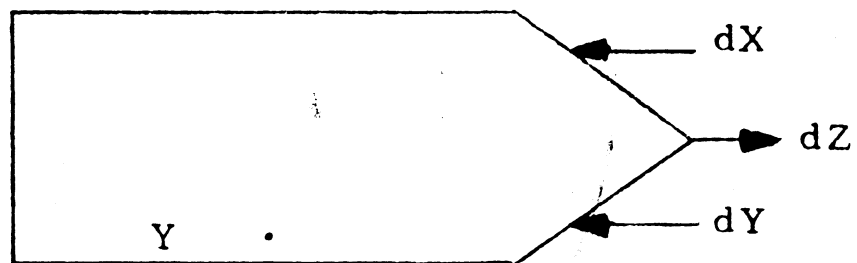


Figure 2-1. Symbol for Integrator

The block diagram for the simplest mechanization of the Integrator is shown in Figure 2-2.

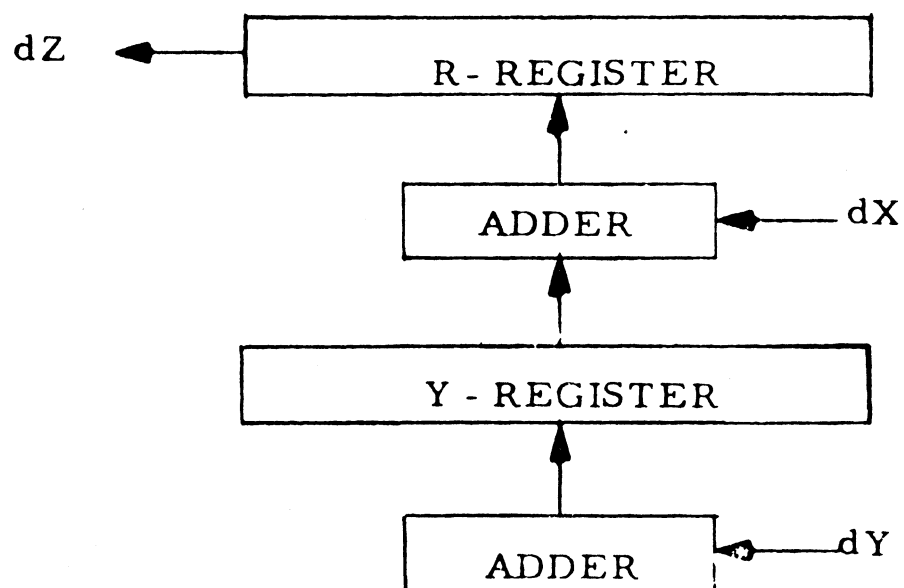


Figure 2-2. Simple Mechanization of Integrator

The value of Y is maintained in the Y -register by the addition of the dY increments (secondary input) to the Y -register. In this manner the value of Y is made to vary in accordance with the desired function. The value Y in the Y -register is added to or subtracted from the value in the R -register each time there is a dX increment input (primary input) to the adder; i. e., Y is added to the R -register when dX is positive, and Y is subtracted from the R -register when dX is negative. The R -register produces a positive dZ output increment when it overflows in a positive-going direction and a negative dZ output increment when it overflows in a negative-going direction. Thus, the sum of the overflows of the R -register is an indication of the summation of the values of $Y dX$ or an approximation to $\int Y dX$.

Graphically, the operation of an Integrator may be represented as shown in Figure 2-3. Thus, the Integrator receives two incremental inputs, dX and dY , and generates an incremental output, dZ . If these incremental outputs are summed, the result is a number Z , where

$$Z \approx \int Y dX$$

over the interval of X values employed.

It should be noted that the independent variable may be any function and need not be time.

The dZ output of one Integrator may serve as the dX or dY input to another Integrator or some other TRICE unit. The dZ output is stored in two lines; one line indicates the sign of the output, and the other line indicates the existence of the output. If the integrand, $f(X)$, exceeds its preassigned full-scale value, an OVERFLOW light turns on and computation may be stopped automatically.

Refinements added to this simple mechanization of the Integrator include an I -register, a trapezoidal correction, and a scaling factor. Provisions are made for two secondary inputs (dY 's). These refinements are explained in the following paragraphs.

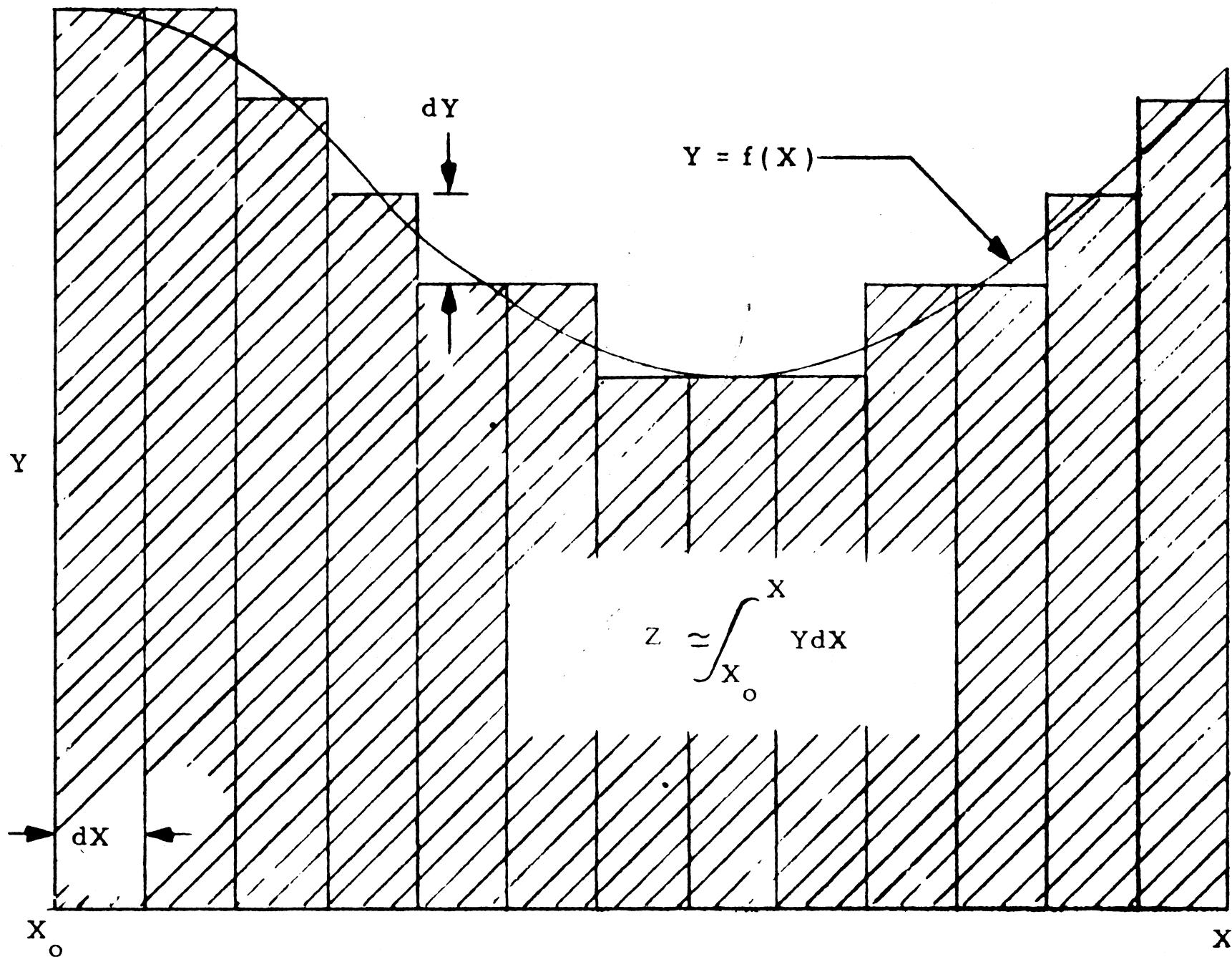


Figure 2-3. Graphical Representation of Rectangular Integration

2. 3. 1. 1 Mathematical Operation. -

Operation is in a trapezoidal extrapolative mode; i. e., the Integrator calculates the area of the $(i+1)$ interval during the i interval (see Figure 2-4). This mode of operation is used because the secondary input, dY , frequently is an Integrator output from a preceding integration interval. Thus, the Y value to be integrated is extrapolated by linear interpolation.

The quantity S_i is added to the R-register during any cycle, i , and is given by

$$S_i = (Y_i + dY_i) dX_i + \left(\frac{1}{2}\right) (dY_i) \left[\text{Sign } (dX_i) \right]$$

where dX_i consists of one binary digit and a sign. The term $\left[\text{Sign } (dX_i) \right]$ is explained as follows:

$\left[\text{Sign } (dX_i) \right]$ equals +1 when dX_i is positive.

$\left[\text{Sign } (dX_i) \right]$ equals -1 when dX_i is negative.

The quantity Y_i is the value of Y at the beginning of the i cycle.

The second term in the equation, $\frac{1}{2} (dY_i) \left[\text{Sign } (dX_i) \right]$, is the trapezoidal correction. The quantity $dY_i (dX_i)$ is the extrapolative correction.

A slightly different, but equivalent, form of the equation for the computation of S_i is

$$S_i = \left[(Y_i + dY_i) |dX_i| + \left(\frac{1}{2}\right) (dY_i) \right] \left[\text{Sign } (dX_i) \right]$$

The above equation is a good representation of how the computation of S_i is mechanized. It should be noted that the primary input, dX , can be a +1, 0, or -1. Thus, the primary input may be 0 on many successive integration cycles. However, in this case

$$S_i = \left(\frac{1}{2}\right) (dY_i) \left[\text{Sign } dX_i \right]$$

where the term $\left[\text{Sign } dX_i \right]$ is the sign of the last non-zero dX_i ; similarly, all TRICE computing modules remember and use the sign of the last non-zero increment. Thus, the effect is that of averaging the value of Y in the region between non-zero inputs.

- (c) If the capacity of the R-register were large enough, its contents would approximate

$$Z_i \approx \int_{x_0}^x Y dX$$

However instead of accumulating the full integral, increments of the integral, dZ , are generated and only the remainder is stored in the R-register.

Hence, the content of the R-register, R_i , at the beginning of any cycle, i , is

$$R_i = \frac{1}{2} + \sum_{K=0}^{K=i-1} (S_k - dZ_k)$$

where R_i is the range between 0 and 1. The R-register is set to an initial value of $1/2$ to minimize the round-off error.

- (d) The incremental output (overflows of the R-register) consists of one binary digit and a sign, i. e., dZ . The output is defined as follows:
- (1) $[\text{Sign}(dZ_i)] = [\text{Sign } S_i]$, i. e., the sign of the output increment is the same as the sign of S_i .
 - (2) $dZ = +1$ if $(R_i + S_i) \geq 1$; i. e., there will be a positive output increment if the remainder in the R-register from the previous cycle plus the value added to the R-register in the present cycle is equal to or greater than 1.
 - (3) $dZ = -1$ if $(R_i + S_i) < 0$; i. e., there will be a negative output increment if the remainder in the R-register from the previous cycle plus the value added to the R-register in the present cycle is less than 0 (in this case S_i is negative).
 - (4) $dZ = 0$ if $0 \leq (R_i + S_i) < 1$; i. e., there is no output increment if the value of the remainder from the previous cycle plus the value added to the R-register in the present cycle is greater than or equal to zero, but less than 1.

Since the Integrator operates in a trapezoidal extrapolative mode, the output for any cycle, i , can be written as

$$dZ_i = \left[(Y_i + dY_i) \left| dX_i \right| + \frac{1}{2} dY_i \right] \left[\text{Sign } dX_i \right]$$

2.3.2 Constant Multiplier. -

The Constant Multiplier is identical to an Integrator except there are no provisions for dY increments (see Figure 1-3). The value of the Y-register is loaded in the same manner as for the other units; however, the value selected for Y is not incremented (it does not change). Thus, Y is treated as a constant in the Constant Multiplier.

2.3.3 Variable Multiplier. -

Multiplication can be performed by the use of two Integrators as shown in Figure 2-5.

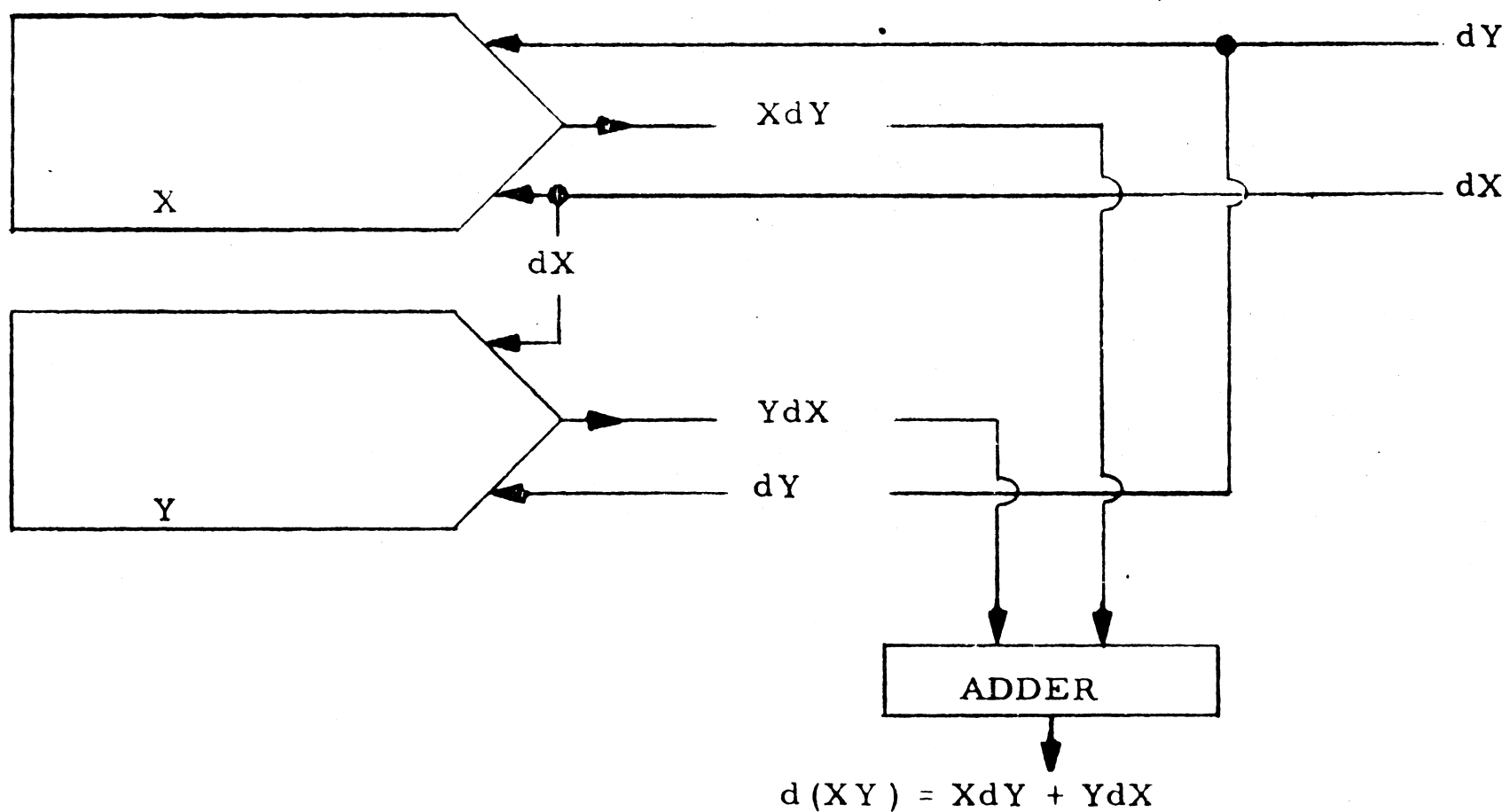


Figure 2-5. Multiplication Using Two Integrators

However, greater circuit economy and computational accuracy are achieved by combining the two Integrators into a single multiplier unit. The Variable Multiplier uses the exact formula

$$d(XY) = (Y+dY)dX + XdY$$

or

$$d(XY) = XdY + YdX + dYdX$$

It should be noted that with rectangular integration the error is $dXdY$.

That is

$$d(XY) = (Y+dY)dX + (X+dX)dY$$

$$d(XY) = YdX + XdY + 2dXdY$$

as shown in Figure 2-6.

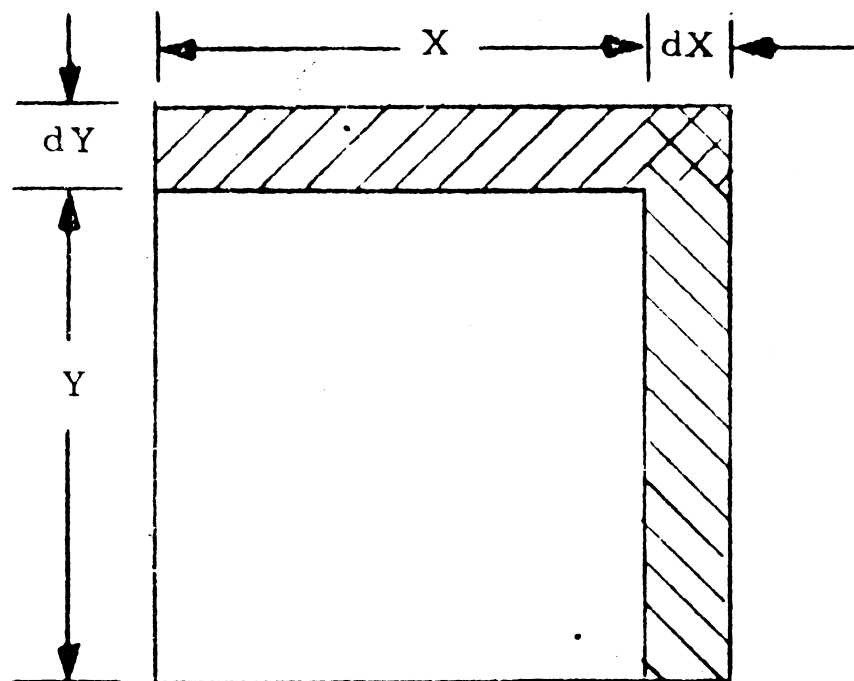


Figure 2-6. Rectangular Multiplication

As illustrated in Figure 2-7, the error in extrapolative trapezoidal integration is $2dXdY$ from

$$d(XY) = \left(Y + \frac{3}{2}dY\right)dX + \left(X + \frac{3}{2}dX\right)dY$$

$$d(XY) = YdX + XdY + 3dXdY$$

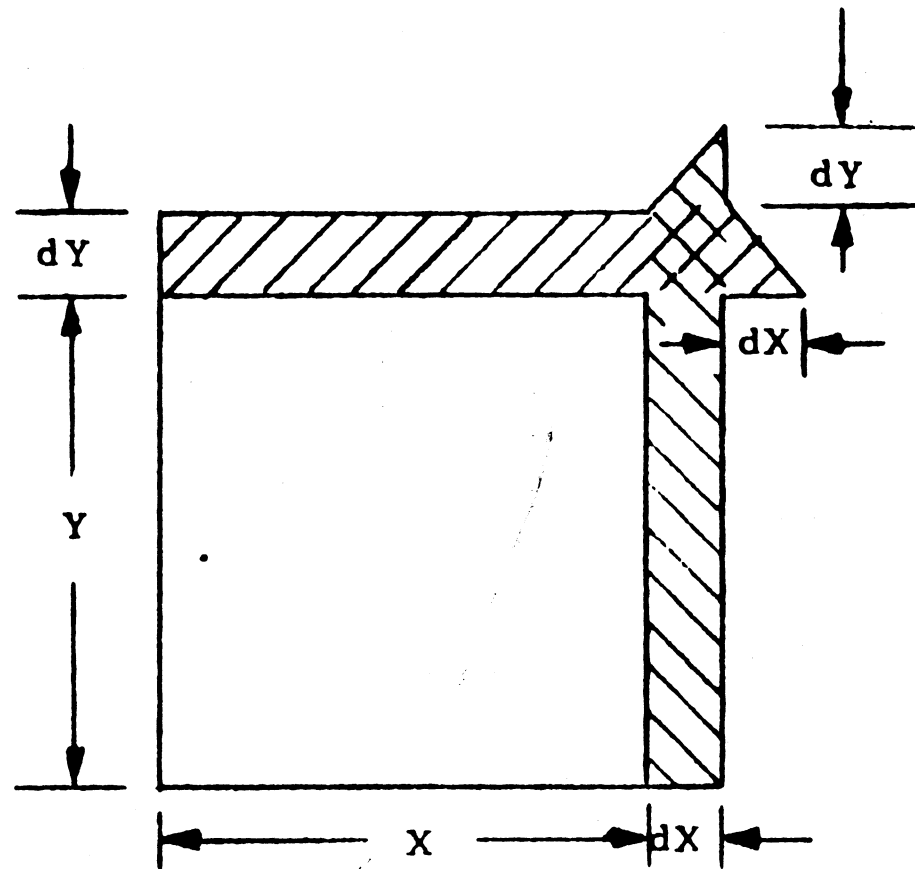


Figure 2-7. Trapezoidal Multiplication

The formula

$$d(XY) = (Y+dY)dX + XdY$$

is implemented by multiplying the old value of X by dY , the new value of Y by dX , and summing both of these quantities into the R-register. In this manner the corner area is added only once. The saving of circuitry over the compound multiplier results mainly from the use of only one R-register, one pair of adders, and one output circuit to accumulate the product and send it out in incremental form. In addition, all overhead functions such as filling, timing, and overflow detection are performed by common circuits shared between the two multiplication factors, X and Y . The Variable Multiplier has the following 5 registers:

- (a) Y-register
- (b) X-register

- (c) Y-initial-condition register
- (d) X-initial-condition register
- (e) R-register

The initial-condition values are stored in the same manner as for the Integrators.

The Variable Multiplier can overflow under each of the following conditions:

- (a) The X-register exceeds the range:

$$-1 \leq X < +1$$

- (b) The Y-register exceeds the range:

$$-1 \leq Y < +1$$

- (c) The quantity $S = YdX + XdY$ exceeds the range:

$$-1 \leq S < +1$$

The first two conditions, (a) and (b), will be satisfied if the scaling is such that

$$\begin{aligned} |X| &< 1 \\ |Y| &< 1 \end{aligned}$$

The last condition (c), will be satisfied if the scaling is such that

$$\begin{aligned} |X| &< \frac{1}{2} \\ |Y| &< \frac{1}{2} \end{aligned}$$

2. 3. 4 Digital Servo. -

The Digital Servo is used as a nulling device in the solution of differential equations or as a decision element in the generation of discontinuous and non-linear functions. The Servo detects any deviation from zero in its Y-register, which is initially set to zero.

The Digital Servo receives two secondary incremental inputs, dY_1 , and dY_2 , and accumulates them in the Y-register. The incremental output, dZ , is

$$\begin{aligned} |\Delta Z| &= 1 \text{ for } |\Delta X| = 1 \text{ when } 0 < |Y| < 1 \\ \text{Sign } \Delta Z &= (\text{Sign } \Delta X) (\text{Sign } Y) \end{aligned}$$

Since the output is directly derived from Y and ΔX , no R-register is required. The Y-register is automatically set to zero by the RESET signal.

2. 3. 5 Decision Servo. -

Discontinuous and non-linear functions, such as square or saw-tooth waves, may be generated using a Decision Servo. The symbolic representation of the Decision Servo is shown in Figure 1-3. The initial condition for the Decision Servo Y-register may or may not be zero. The mathematical operation of the Decision Servo is defined as follows:

$$\begin{aligned} |\Delta Z| &= 1 \text{ for } \left\{ \begin{array}{l} 0 < Y < \frac{1}{2} \\ 0 > Y \geq -\frac{1}{2} \end{array} \right\} \text{ if } |\Delta X| = 1 \\ \text{Sign } \Delta Z &= (\text{Sign } \Delta X) (\text{Sign } Y) \end{aligned}$$

Since the output is derived from Y and ΔX , no R-register is required. The Y-register is automatically reset to the filled initial condition by the RESET signal.

2. 3. 6 ΔY Summer. -

The Integrators and Servos each are capable of receiving only two dY inputs directly. However, as many as six dY inputs can be summed by the ΔY Summer and used as the input to the Integrator or the Servo to which the ΔY Summer is connected.

The partial sums ($dY_1 + dY_2 + dY_3$) and ($dY_4 + dY_5 + dY_6$) are formed during the first phase of operation. These partial sums are in the range from -3 to +3. During the second phase of operation these partial sums are added to produce a final sum in the range from -6 to +6. The shift operation gates the final sum into the Y-register of the unit to which the ΔY Summer is directly connected (an Integrator and a Servo).

2.4 ANALOG-TO-DIGITAL CONVERTER.

The input to the A-to-D converter is an analog voltage representation of a variable, Z. Increments in the form of voltage levels constitute the digital output from the A-to-D converter. If Z increases to the next higher level of voltage, a positive increment is produced (positive output). If Z decreases to the next lower level of voltage, a negative increment is produced (negative output). Thus, the rate at which the output increments occur is dependent upon the slope of the analog input voltage curve. The mathematical symbolic representation of the A-to-D converter is shown in Figure 1-3.

A 14-stage binary counter is used in conjunction with a precision voltage divider to generate a voltage with which to balance the Z input voltage at a summing node. A comparator is triggered by a 100,000-cps signal from the Control unit to compare the error at the summing node with zero volts. When the error exceeds $+(1/2) dZ$ or $-(1/2) dZ$, the counter is pulsed to count down or count up by single dZ increments. These counting pulses constitute the incremental dZ output of the converter.

2.5 DIGITAL-TO-ANALOG CONVERTER.

The D-to-A converter receives information in digital form and converts it to an analog voltage. The mathematical symbolic representation of the D-to-A converter is shown in Figure 1-3. The A-to-D converter can be connected to operate as a D-to-A converter.

SECTION III

PROGRAMMING PROCEDURE

3.1 GENERAL.

This section describes procedures for preparing the mathematical program for TRICE. Section IV describes how to operate TRICE, using these programs. The Appendix contains example problems in which the programming and operating procedures are explained in detail.

The three steps involved in the mathematical programming of TRICE are

- (1) Mapping
- (2) Scaling
- (3) Coding

Many programs may be prepared for any problem. Although straightforward methods for preparing a program are available (such as separating the highest derivative), the efficiency of programming a problem may be increased by the ingenuity of the programmer. Experience in programming will lead to short cuts and the selection of the most efficient programs.

3.2 ANALYSIS OF PROBLEM.

A problem should be analyzed for program simplification before starting a program. Since the TRICE actually solves differential equations, the first step is to make sure that the problem is in the form of a differential equation. The second step in analyzing the problem is to consider the terms with respect to

what each computing module of TRICE can perform, for example:

- (1) Integrators can be used to generate any order of derivative. Assume that the integrand of an Integrator is an n th order derivative; when integrated with respect to the independent variable, the output may be summed to obtain the $(n - 1)$ order derivative (see Example Problem II in the Appendix).
- (2) Digital Servos are used to generate u as a function v , where u is defined implicitly as a function of v by $F(u, v) = 0$. (See Example Problem V in the Appendix.) Digital Servos are also used to increase the number of primary inputs to another unit.
- (3) Constant Multipliers are used to multiply a variable by a constant. (See Example Problem II in the Appendix.)
- (4) Variable Multipliers are used to generate the product of two functions. Example Problem II in the Appendix shows an interesting use made of the Variable Multiplier to vary a constant in a problem.
- (5) Decision Servos are used to generate discontinuities and non-linear functions such as square and saw-tooth waves. Another common use of the Decision Servo is in generating absolute values. The Decision Servo operates like a Digital Servo when the $|Y|$ value is less than $1/2$.
- (6) ΔY Summer are used preceding secondary inputs to a unit when it is desired to increase the number of secondary inputs to that unit. The maximum number of secondary inputs any unit can accept is two; however, the ΔY Summer makes it possible to combine six secondary inputs. Each ΔY Summer is connected directly to an Integrator and a Servo.

The third step in analyzing a problem is to consider any special program advantages that may be obtained or required by a particular mode of operation,

e. g. , input, output, function switching, method of filling, and computation rate.

The last step in analyzing a problem is to consider the problem with respect to initial conditions and limiting values.

3.2.1 Initial Conditions. -

The Integrators, Variable Multipliers, and Decision Servos have initial-condition registers that determine the values for their variable computing registers at the beginning of each computing run. The initial-condition registers hold the initial conditions and insert them into the computing registers each time a RESET signal is received. Hence, it is necessary to determine the initial values for each of these registers by analyzing the problem.

A starting point is given. However, usually the initial conditions for a few of the modules must be calculated. Sometimes a starting point must be assumed. For example suppose the mapping of

$$\frac{d^2 y}{dx^2} + \frac{dy}{dx} - y^2 - \sin y - K = 0$$

leads to the integrands listed in Table 3-1.

Table 3-1. Example Integrands

Integrator	Integrand
1	$\frac{d^2 y}{dx^2}$
2	$\frac{dy}{dx}$
3	y
4	$\cos y$
5	$\sin y$

Assume that the problem requires the second order derivative to be equal to K when $y = 0$. Thus, the values for the initial-condition registers may be chosen as follows:

(1) Integrator 1

Given: Initially $\frac{d^2 y}{dx^2} = K$ and $y = 0$

Hence, the initial-condition register value for Integrator 1 is equal to K .

(2) Integrator 2

Given: Initially $y = 0$ and $\frac{d^2 y}{dx^2} = K$.

The given equation becomes

$$K + \frac{dy}{dx} - K = 0$$

Therefore,

$$\frac{dy}{dx} = 0$$

Hence, the initial-condition register value for Integrator 2 is equal to 0.

(3) Integrator 3

Given: Starting point of $y = 0$

Hence, the initial-condition register value for Integrator 3 is equal to 0.

(4) Integrator 4

Given: Initially $y = 0$; thus, $\sin y = 0$

Hence, the initial-condition register value for Integrator 4 is equal to 0.

(5) Integrator 5

Given: $y = 0$ initially; thus, $\cos y = 1$

Hence, the initial-condition register value for Integrator 5 is equal to 1.

By considering the availability of initial values, difficulties can be avoided in mapping by using those terms whose initial conditions are known or can be easily obtained.

3.2.2 Limiting Values. -

Consideration should also be given to the maximum and minimum values related to a problem. These values may affect the efficiency with which a problem must be mapped. The length of the various computing registers, as determined by scale factors, is also affected by these maximum and minimum values. The values may be physical restrictions in the problem, e. g. , maximum altitude, maximum speed, maximum acceleration and minimum pressure.

3.3 MAPPING.

A map of the solution of a problem is prepared by interconnecting the inputs and outputs of the computing module symbols shown in Figure 1-3.

The method of solution of any mathematical problem may vary among mathematicians; likewise, the map selected for a problem may vary among programmers. Experience and ingenuity of the programmer is a valuable asset

3.3.1 Approaches to Mapping. -

There are several accepted approaches to mapping. Probably the most popular method of mapping is integration with respect to an independent variable, which is explained in paragraph 3.3.1.2. Programmers often develop their own approach to mapping problems. Often the ingenuity of a programmer leads to some unusual uses of the various computing modules. For instance in Example Problem II, the Variable Multiplier 1 is used as a Constant Multiplier with a variable constant.

Thus, the following paragraphs describe a few accepted mapping procedures with example maps for several problems.

NOTE

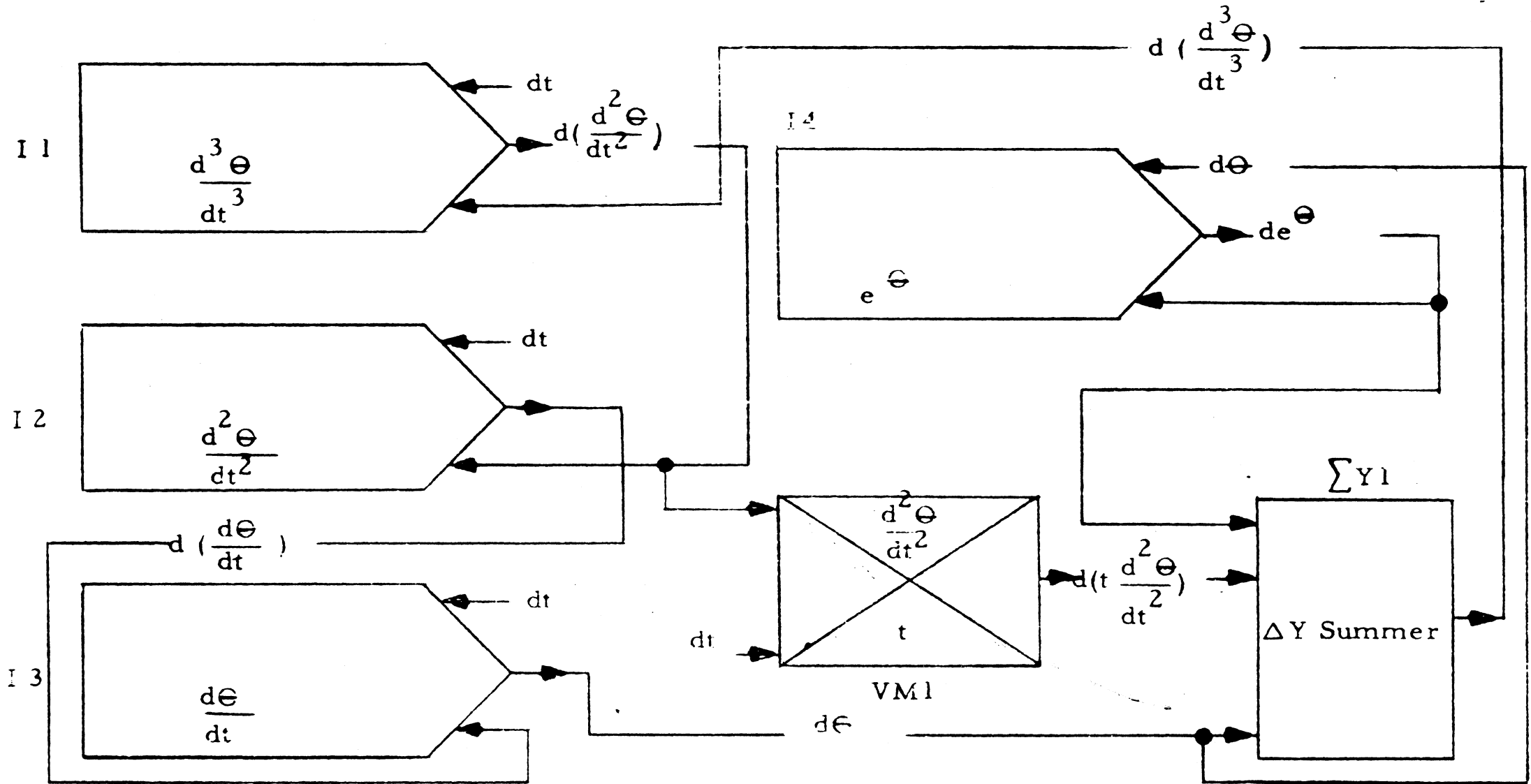
Remember that the output of a computing module may be either positive or negative, depending upon which of the two outputs are used.

3. 3. 1. 1 Separation of Highest Order Derivative.

The most direct approach to mapping a problem is to express the problem as a single n -th order differential equation; then, interconnect the computing modules in such a manner as to generate each of the terms of the equation. This method is only slightly different from the method described in paragraph 3. 3. 1. 2, but results in a more complex map. However, if the highest order derivative of the problem is of importance, which is not usually the case, this method must be used since the method of paragraph 3. 3. 1. 2 will not generate the highest order derivative.

Since the mapping procedure of paragraph 3. 3. 1. 2 is usually more useful and less complex than the map obtained by the method described above, the details of this method are not discussed because this mapping solution should be obvious to the reader after reading paragraph 3. 3. 1. 2.

The problem used in Figure 3-1 is identical to the problem used in Figure 3-2. However, the map of Figure 3-1 is prepared using the separation of the highest order derivative; whereas, the map of Figure 3-2 is obtained after effectively integrating once with respect to the independent variable. Note that the highest order derivative is generated in Figure 3-1, but not in Figure 3-2. Both maps require six modules; however, Figure 3-1 uses a Variable Multiplier to perform the functions of two Integrators. Furthermore, Figure 3-1 requires Integrator I1 to be one of the Integrators that is connected directly to a ΔY Summer.



Given. $\frac{d^3 \Theta}{dt^3} = t \frac{d^2 \Theta}{dt^2} + e^\Theta + \Theta$

Figure 3-1. Map of Third Order Differential Equation

3. 3. 1. 2 Integration with Respect to an Independent Variable.

Express the problem as a single n-th order differential equation, e. g. ,

$$f \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^n \Theta}{dt^n} \right) = 0$$

then separate the highest order derivative, i. e. ,

$$\frac{d^n \Theta}{dt^n} = g \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^{n-1} \Theta}{dt^{n-1}} \right)$$

Multiply the equation by the independent variable differential, dt, i. e. ,

$$\left(\frac{d^n \Theta}{dt^n} \right) dt = g \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^{n-1} \Theta}{dt^{n-1}} \right) dt$$

the result is

$$d \left(\frac{d^{n-1} \Theta}{dt^{n-1}} \right) = g \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^{n-1} \Theta}{dt^{n-1}} \right) dt$$

If the equation

$$\frac{d^n \Theta}{dt^n} = g \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^{n-1} \Theta}{dt^{n-1}} \right)$$

is integrated with respect to the independent variable (t in this case) and then the differential were obtained, the results would be identical, i. e. ,

$$d \left(\frac{d^{n-1} \Theta}{dt^{n-1}} \right) = g \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^{n-1} \Theta}{dt^{n-1}} \right) dt$$

Thus, the effect is to perform the first integration for TRICE. Since this method improves efficiency by reducing the number of modules required and usually the highest order derivative is of no interest in most problems this is the most popular method of mapping problems.

The map is prepared as follows:

- (1) Let the integrand of the first Integrator be

$$\frac{d^{n-1} \Theta}{dt^{n-1}}$$

and integrate with respect to the independent variable, t , which results in

$$d \left(\frac{d^{n-2} \Theta}{dt^{n-2}} \right)$$

as the output of the Integrator.

- (2) Let the integrand of the second Integrator be

$$\frac{d^{n-2} \Theta}{dt^{n-2}}$$

and integrate with respect to the independent variable, t , which produces

$$d \left(\frac{d^{n-3} \Theta}{dt^{n-3}} \right)$$

as the output of the Integrator.

- (3) Continue this process until each term of the equation

$$d \left(\frac{d^{n-1} \Theta}{dt^{n-1}} \right) = g \left(t, \Theta, \frac{d\Theta}{dt}, \dots, \frac{d^{n-1} \Theta}{dt} \right) dt$$

has been generated as outputs from the various Integrators.

- (4) Connect the outputs of the various Integrators in such a manner as to obtain

$$d \left(\frac{d^{n-1} \Theta}{dt^{n-1}} \right)$$

which is the required secondary input to the first Integrator. See Figure 3-2 for the map to the simple problem using this method of mapping. The ΔY Summer is necessary to sum the three terms for the secondary input of I1, which must be one of the Integrators that is connected directly to a ΔY Summer.

- (5) The output of this interconnection is taken from a line that has the desired result on it. For example, if the output of I2 in Figure 3-2 is taken, $d\Theta$ is the output; this could be the input to a D-to-A converter that would sum $d\Theta$ to produce Θ as the output of the D-to-A. This could be used as the vertical axis input to a plotter, and if dt were fed through a D-to-A converter to the horizontal axis of the plotter, a plot of Θ versus t would result. Plots of the other terms may be obtained in the same manner.

Simultaneous differential equations are solved in a similar manner with a group of Integrators being used to generate the derivatives associated with each dependent variable. Figure 3-3 is an example of a simultaneous differential equation mapped using this method. In this problem both secondary inputs to Integrators I1 and I5 are used.

NOTE

The machine variable dt may be the independent variable or the dependent variable in a problem

A labor-saving method of mapping for the experienced programmer is to represent the inputs and outputs of Integrators by equations. An Integrator may be represented by the following equation:

$$d \left[\int \text{Integrand} \right] = \left[\text{Integrand} \right] \left[\text{Primary Input} \right]$$

Given Equation: $\frac{d^3 \Theta}{dt^3} = t \frac{d^2 \Theta}{dt^2} + e^\Theta + \Theta$

$d\ddot{\Theta} = t d\dot{\Theta} + e^\Theta dt + \Theta dt$

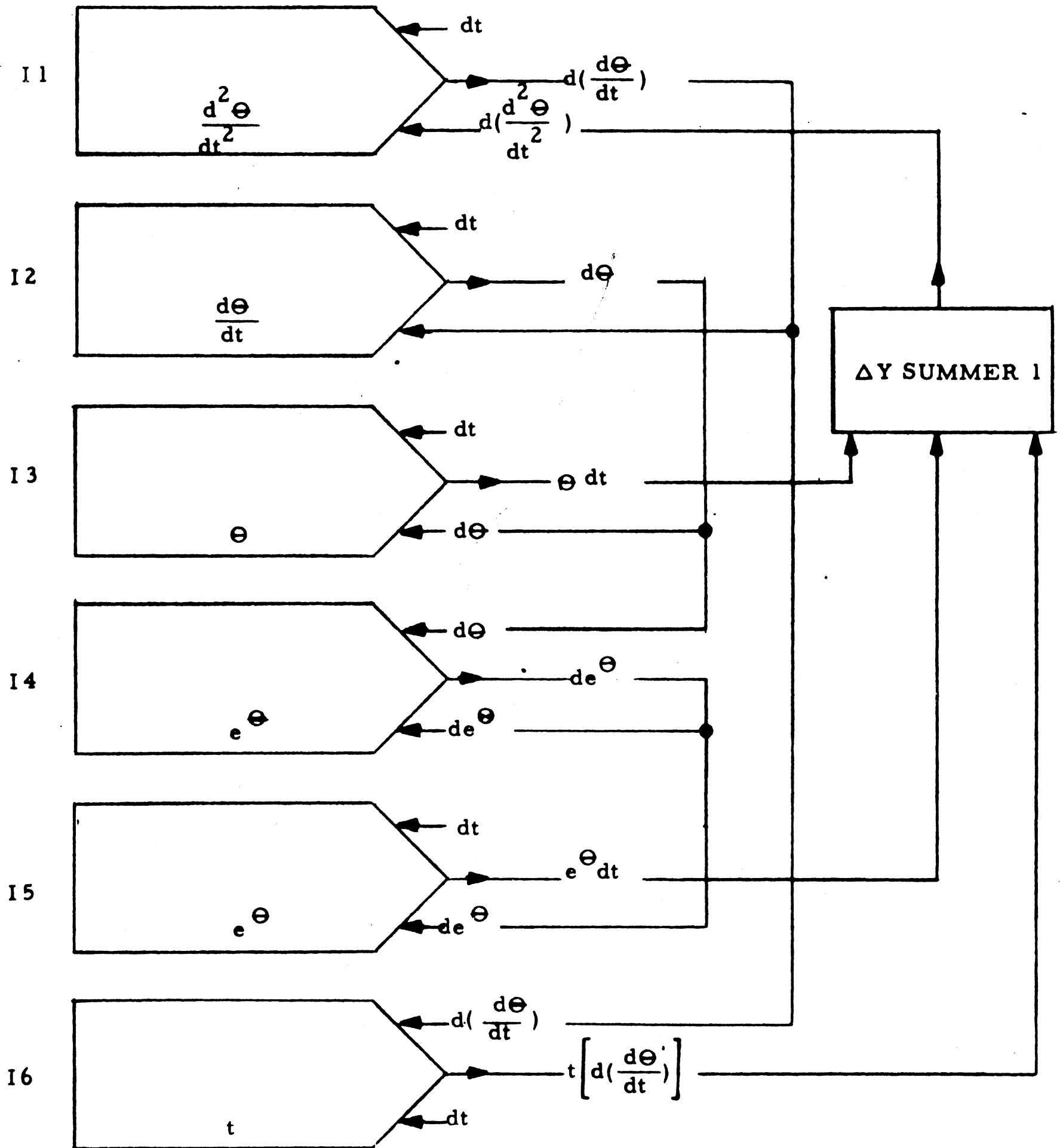
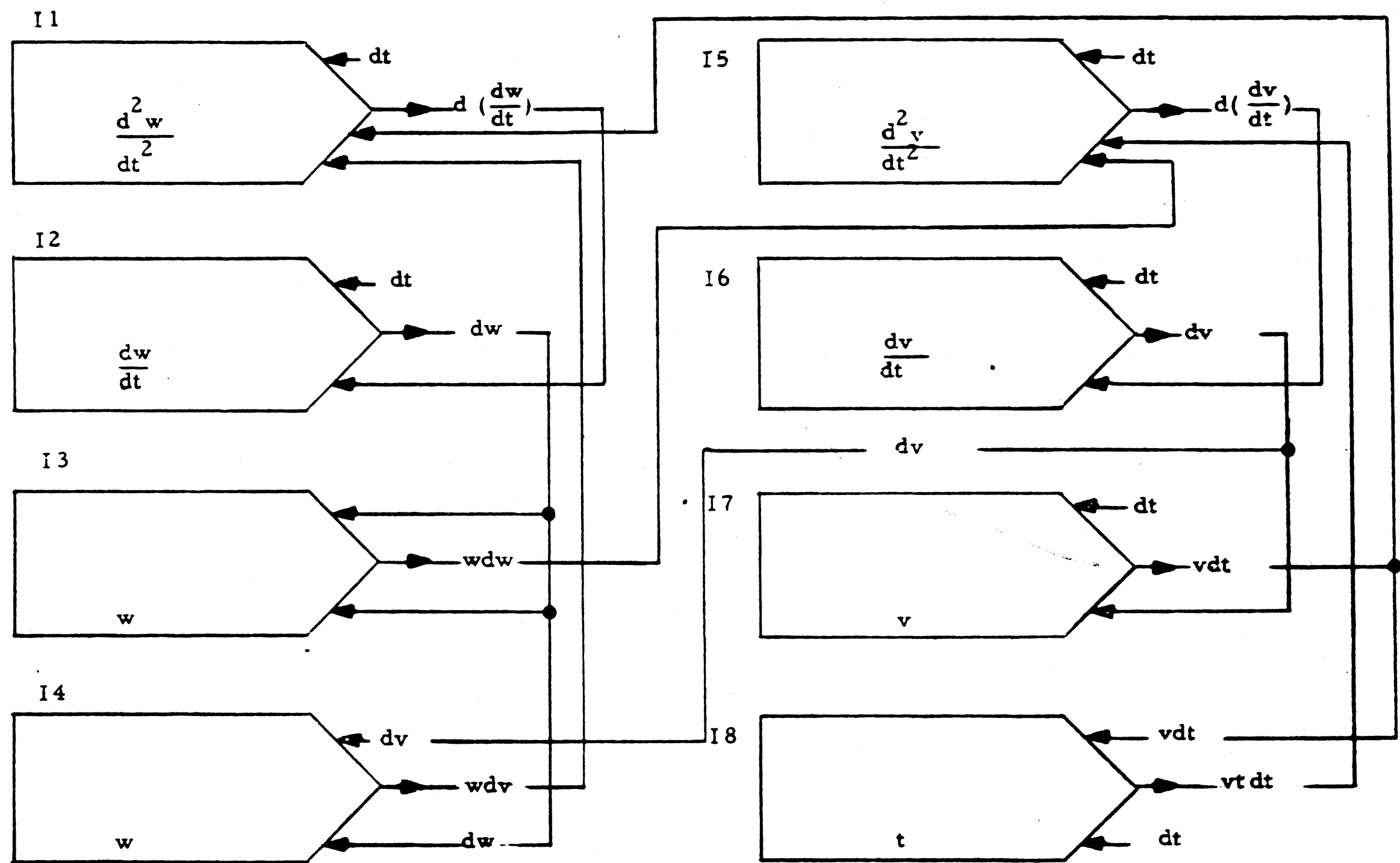


Figure 3-2. Alternate Map of Third Order Differential Equation



Given: $\frac{d^3 w}{dt^3} = w \frac{dv}{dt} + v$; $\frac{d^3 v}{dt^3} = w \frac{dw}{dt} + vt$

Figure 3-3. Map of Simultaneous Differential Equations Using Integration With Respect to Independent Variable

with the secondary input being the derivative of the integrand. The output of the Integrator is

$$d \left[\int \text{Integrand} \right]$$

Using this scheme the map of Figure 3-1 would be as follows:

$$I1 \quad d \left(\frac{d\Theta}{dt} \right) = \left(\frac{d^2\Theta}{dt^2} \right) dt$$

$$I2 \quad d\Theta = \left(\frac{d\Theta}{dt} \right) dt$$

$$I3 \quad \Theta dt = (\Theta) dt$$

$$I4 \quad de^{\Theta} = (e^{\Theta}) d\Theta$$

$$I5 \quad e^{\Theta} dt = (e^{\Theta}) dt$$

$$I6 \quad t \left[d \left(\frac{d\Theta}{dt} \right) \right] = (t) \left[d \left(\frac{d\Theta}{dt} \right) \right]$$

This scheme of course can be extended in the same manner for the other computing modules.

3. 3. 1. 3 Use of Servos.

The use of Servos in mapping a problem is in itself an approach to mapping certain problems. Servos fall into two general classes; one is a Servo that performs the normal functions of a Servo and is referred to as a Digital Servo in this manual; the other is a decision element, which is referred to as a Decision Servo in this manual. The principal differences between these modules can be ascertained from paragraphs 2. 3. 4 and 2. 3. 5.

If the outputs from different computing modules are to be combined as the primary input to a computing module, they must first be made the secondary inputs to a Servo.

For example, if it is desired to make dv the primary input to an Integrator, where dv is defined by the equation

$$dv = dw + du,$$

then the term dv would be obtained by a Digital Servo, indicated as S in Figure 3-4.

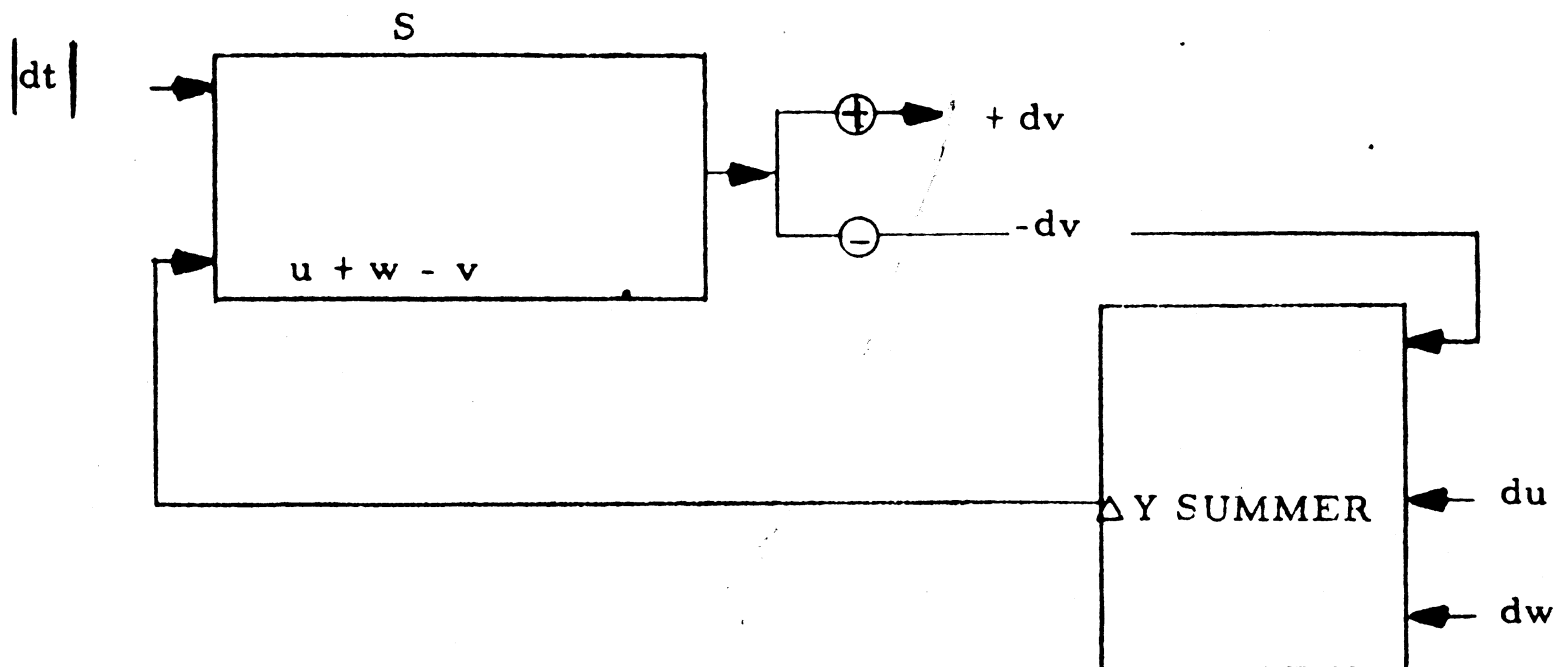


Figure 3-4. Digital Servo Formation of Sum

When it is desired to generate the absolute value of a variable, u , the primary and secondary inputs to a Decision Servo are made the differential du of the variable. Figure 3-5 shows a Decision Servo connected to perform this function.

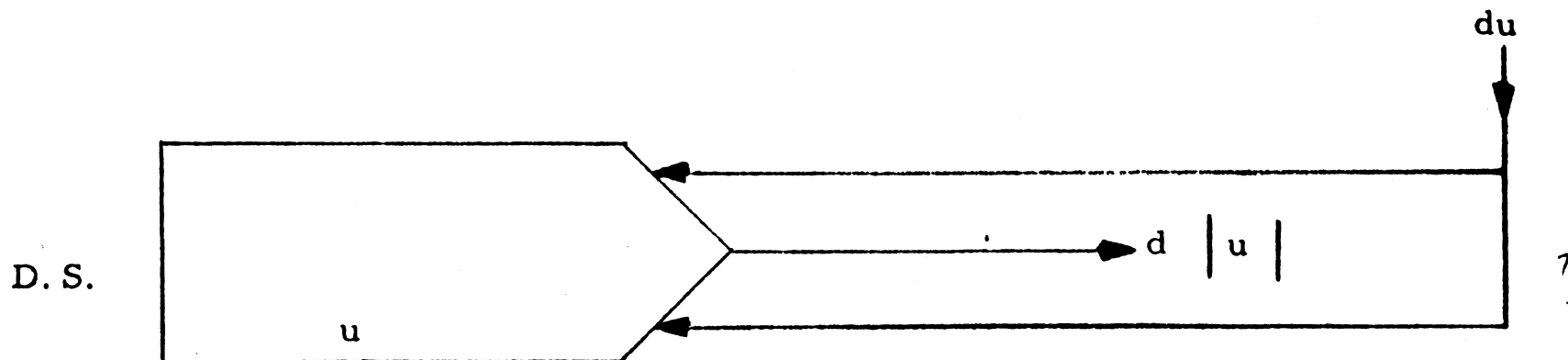


Figure 3-5. Generation of Absolute Value Using Decision Servo

*what happens when
du changes sign & u doesn't!*

The absolute value results from the fact, as indicated in paragraph 2.3.5, that

$$\begin{array}{l} \text{when} \\ \text{and} \end{array} \quad \begin{array}{l} \Delta Z = +1 \\ \Delta X = +1 \\ 0 < Y < +\frac{1}{2} \end{array}$$

$$\begin{array}{l} \text{Also} \\ \text{when} \\ \text{and} \end{array} \quad \begin{array}{l} \Delta Z = +1 \\ \Delta X = -1 \\ 0 > Y \geq -\frac{1}{2} \end{array}$$

$$\begin{array}{l} \text{Furthermore,} \\ \text{when} \\ \text{and} \end{array} \quad \begin{array}{l} \Delta Z = -1 \\ \Delta X = -1 \\ 0 < Y < +\frac{1}{2} \end{array}$$

$$\begin{array}{l} \text{Also} \\ \text{when} \\ \text{and} \end{array} \quad \begin{array}{l} \Delta Z = -1 \\ \Delta X = +1 \\ 0 > Y \geq -\frac{1}{2} \end{array}$$

For example, using the above properties, a set of simultaneous differential equations as defined by

$$\frac{dv}{dx} = -\text{Sign} [w]$$

$$\frac{dw}{dx} = \text{Sign} [v]$$

may be mapped using Decision Servos as shown in Figure 3-6. The D-to-A converters may be connected as shown to obtain the graphs on the analog plotters.

A clipped sine wave, as defined by

$$u(\theta) = b \sin \theta \quad \text{when } b \sin \theta \leq a$$

$$u(\theta) = a \quad \text{when } b \sin \theta \geq a$$

may be generated using the map of Figure 3-7 and the properties indicated in paragraph 2.3.5, i. e. ,

$$\Delta Z = 0 \quad \text{when } +1 > Y \geq +\frac{1}{2}$$

$$\Delta Z = +1 \quad \text{when } +\frac{1}{2} > Y > 0 \quad \text{if } \Delta X = +1$$

$$\Delta Z = -1 \quad \text{when } +\frac{1}{2} > Y > 0 \quad \text{if } \Delta X = -1$$

The normal function of the Digital Servo is to generate u as a function of v , where u is defined implicitly as a function of v by $F(u, v) = 0$.

Paragraph 2.3.4 defines the mathematical operation of a Servo. Figure 3-8 shows a Digital Servo used in the generation of a square root, as defined by

$$F(u, v) = v - u^2 = 0$$

hence

$$u = \sqrt{v}$$

In differential form $dv - 2udu = 0$

The output sign of the Servo must be opposite to the sign of $\frac{\delta F(u, v)}{\delta u}$ for stability.

$$\frac{\delta F(u, v)}{\delta u} = -2u,$$

thus, use (+) sign of the Servo for stability.

When incremental changes in v of Figure 3-8 cause $F(u, v) \neq 0$, the Digital Servo generates incremental changes in u until $F(u, v) = 0$.

Figures 3-9 through 3-16 are maps that have been prepared using a combination of modules and methods of mapping. The Constant Multiplier, which has not been discussed to this point, is used to multiply variables by a constant. The use of the Constant Multiplier in mapping is evident from paragraph 2.3.2. It would be a good exercise for the reader to prepare a map for each of the following problems.

Given: $\frac{dv}{dx} = -\text{Sign}[w]$, $\frac{dw}{dx} = \text{Sign}[v]$

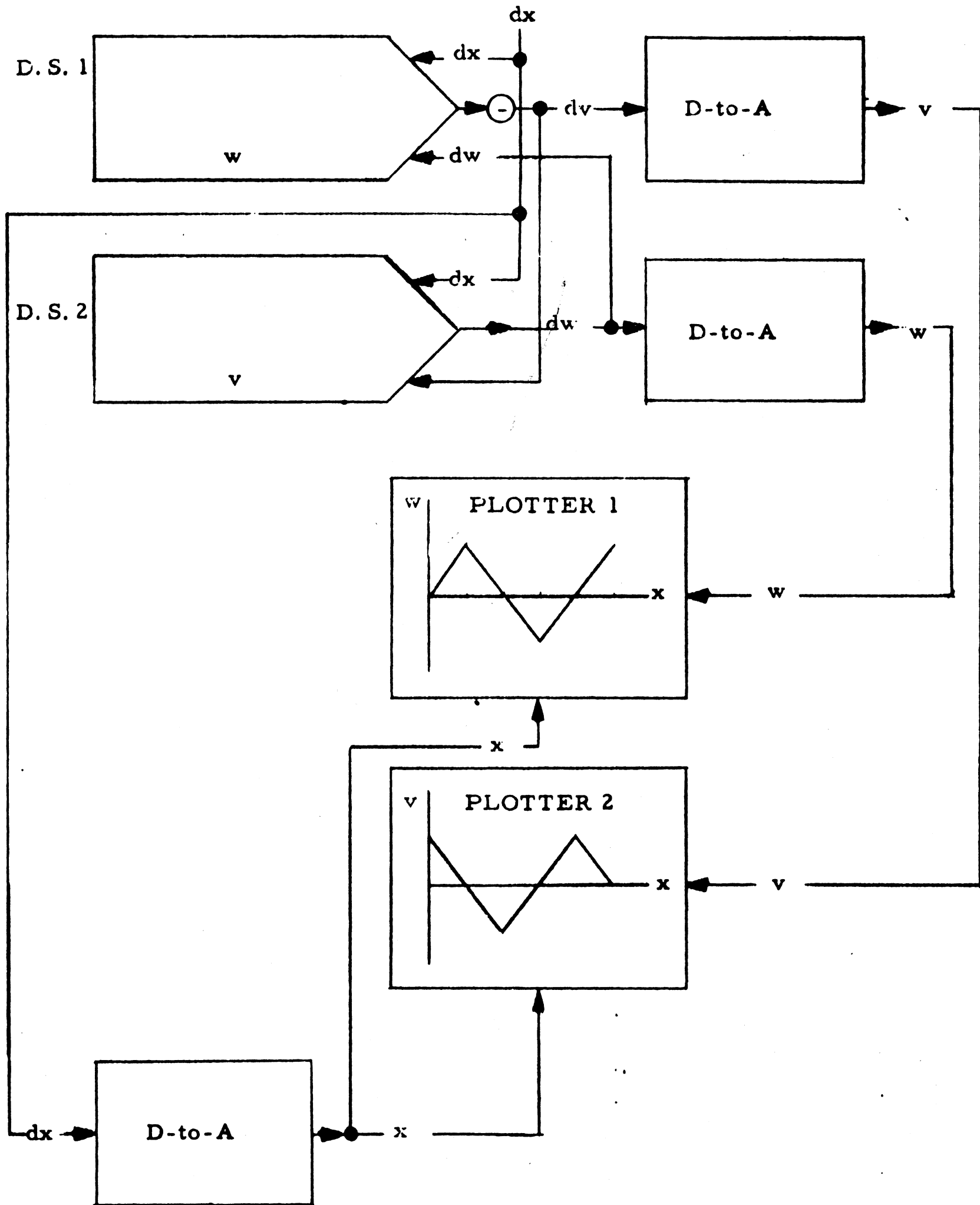


Figure 3-6. Map of Saw-Tooth Functions Using Servos

Given: $u(\theta) = b \sin \theta$ when $b \sin \theta \leq a$
 $u(\theta) = a$ when $b \sin \theta \geq a$

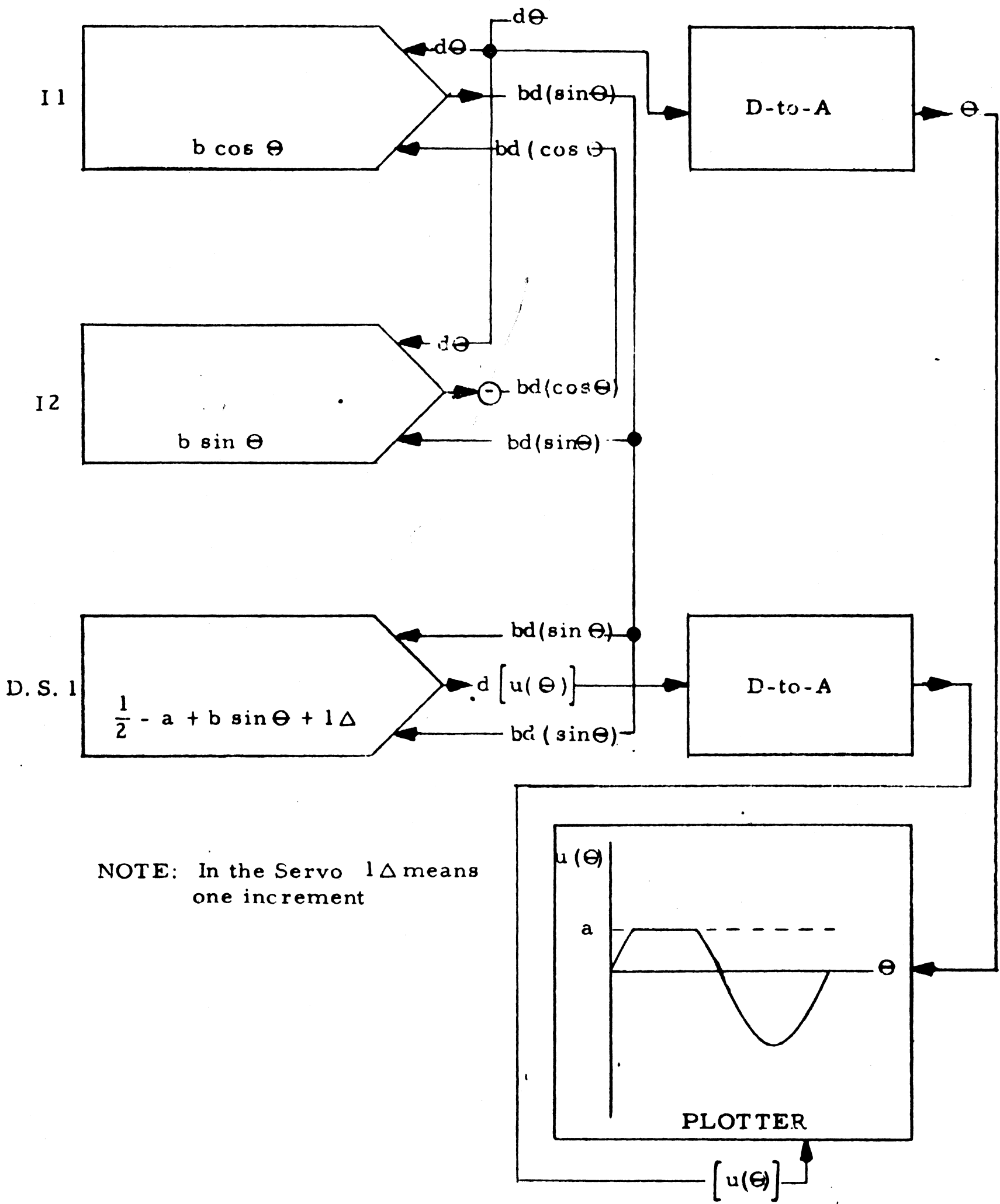


Figure 3-7. Map of Clipped Sine Wave.

Use can be made of the various modules to operate as control modules to vary initial conditions and change problem parameters. A module used in such a manner makes use of its overflow or an output to actuate the automatic reset.

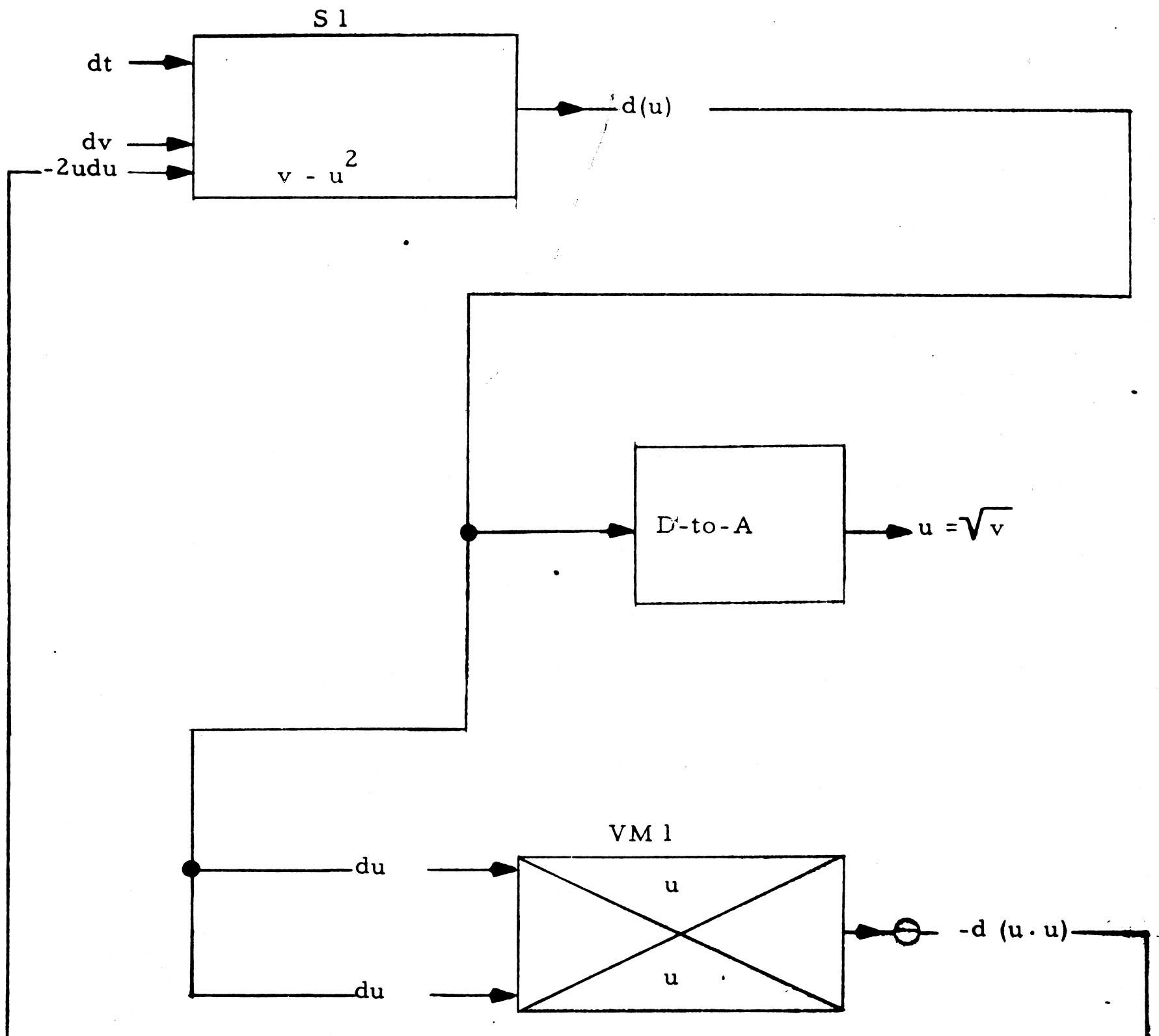


Figure 3-8. Generation of Square Root Using Digital Servo

Problem for Figure 3-9

Given: A voltage loop in an oscillator circuit, that represents the differential equation

$$\frac{d^2 x}{dt^2} + K(x^2 - 1) \frac{dx}{dt} + x = 0$$

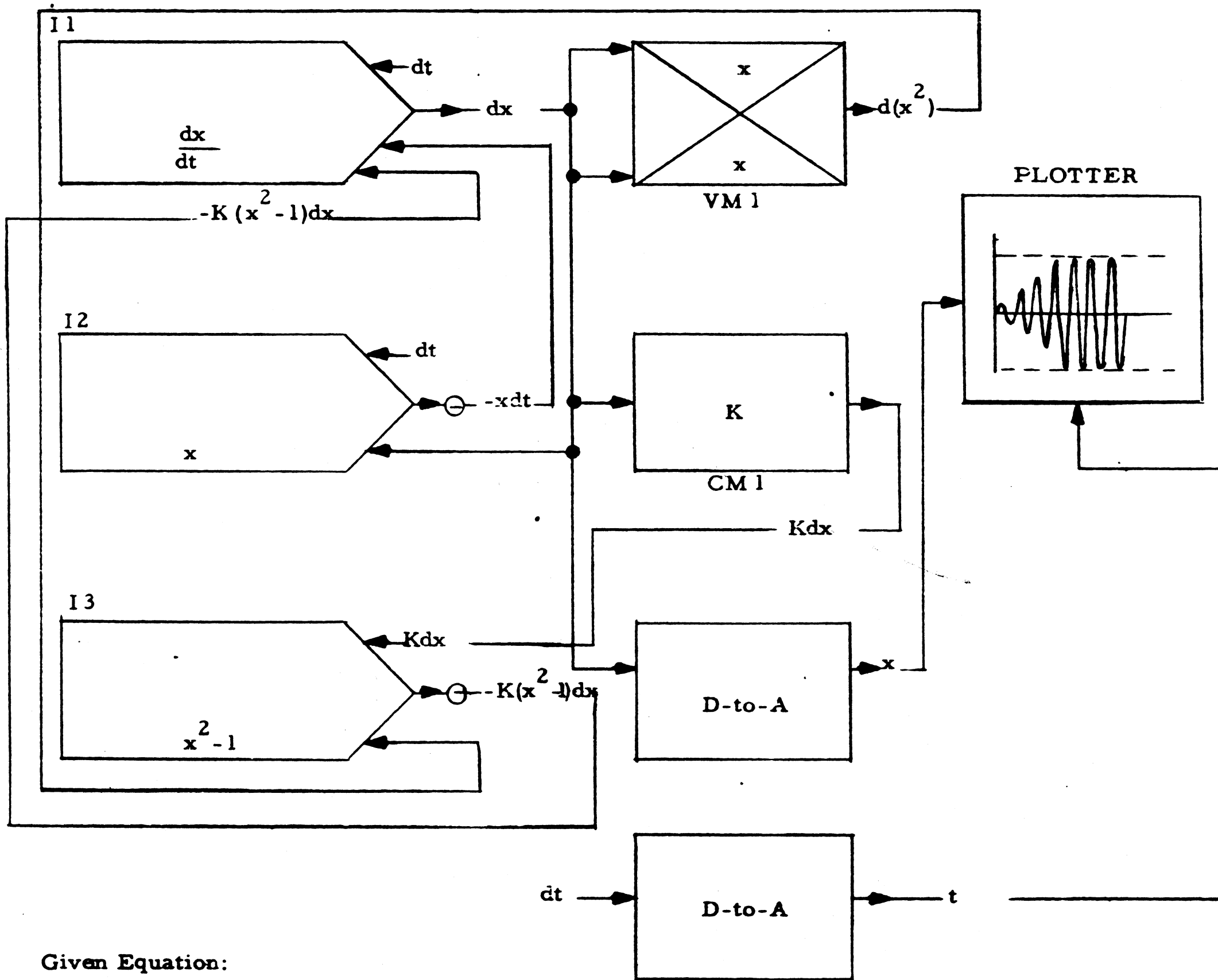
The resistance term

$$K(x^2 - 1)$$

is negative if $x^2 < 1$

and positive if $x^2 > 1$

Thus, the energy in the loop is increased when the resistance becomes more negative; whereas, the energy is decreased when the resistance becomes more positive. A stable oscillator solution is obtained when the energy gains and losses cancel; thus, there is one solution for each value of K. The stable solution may be represented by plotting x versus t . (See Figure 3-9.)



Given Equation:

$$d\left(\frac{dx}{dt}\right) = -K(x^2 - 1)dx - xdt$$

Figure 3-9. Map for Oscillator Voltage Loop

Problem: Generate $Kuvywdx$, where K is a Constant.

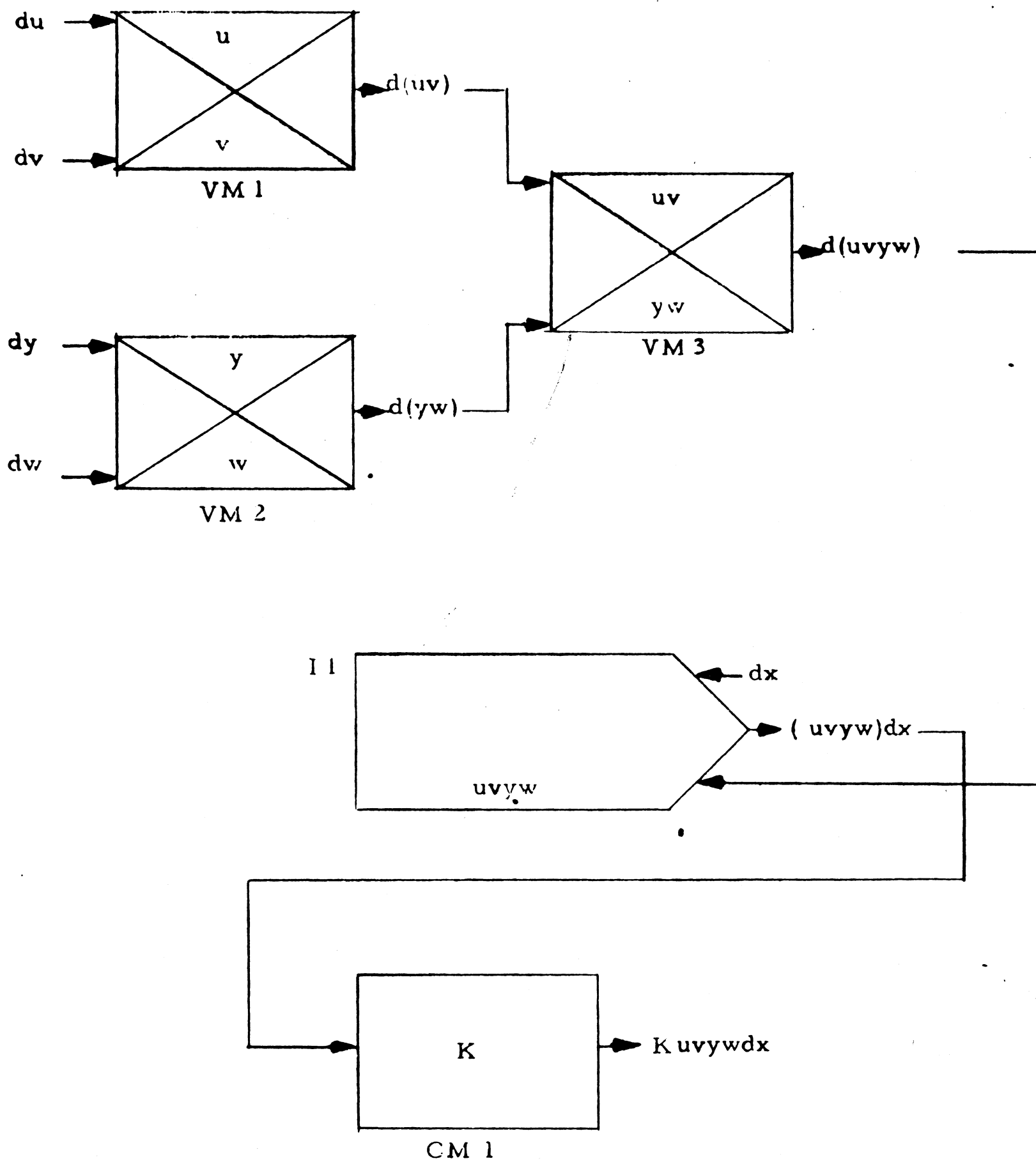


Figure 3-10. Map of Product

Given: Spiral defined by

$$\frac{d^2 x}{dt^2} + c \frac{dx}{dt} + w^2 x = 0$$

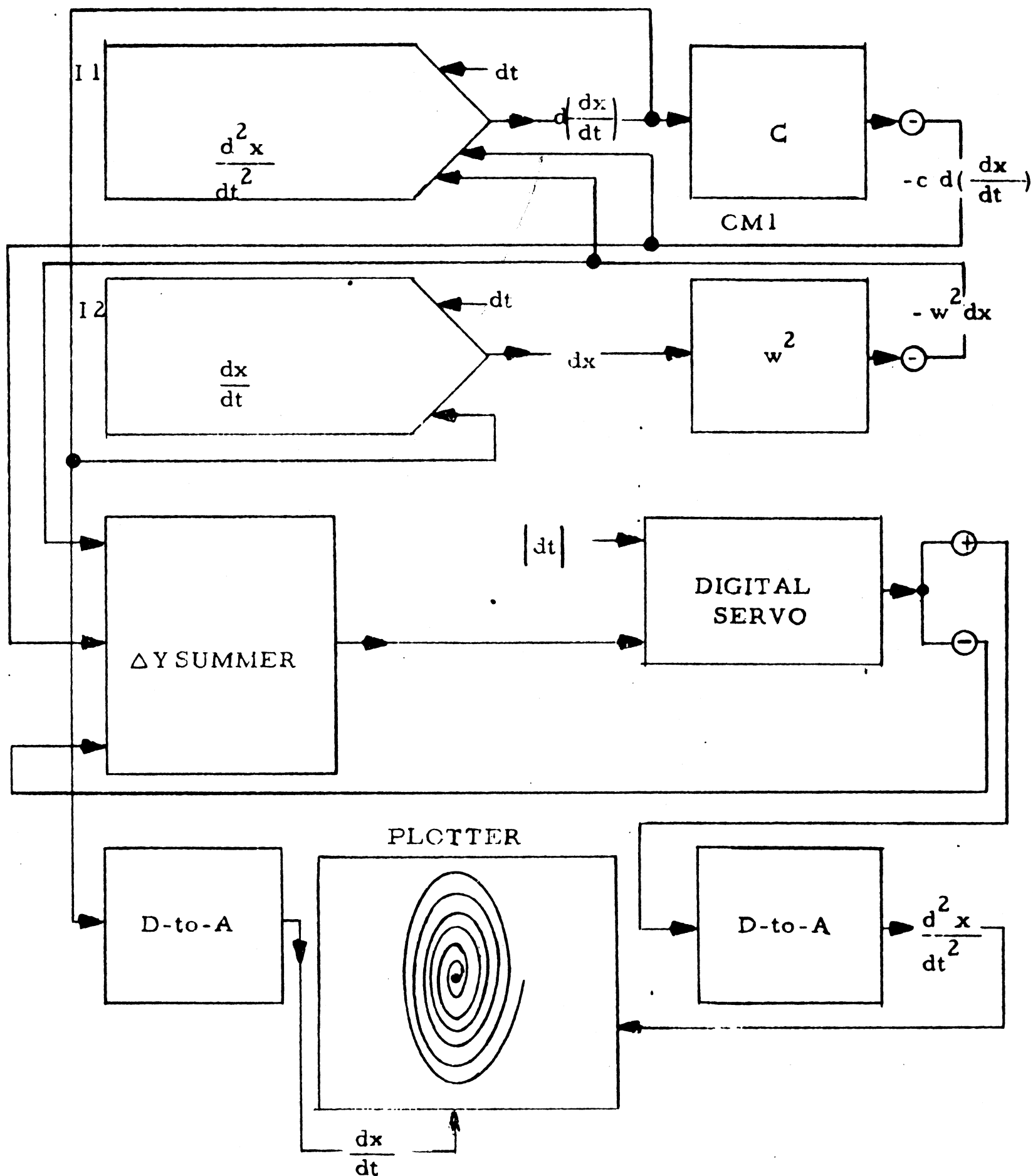


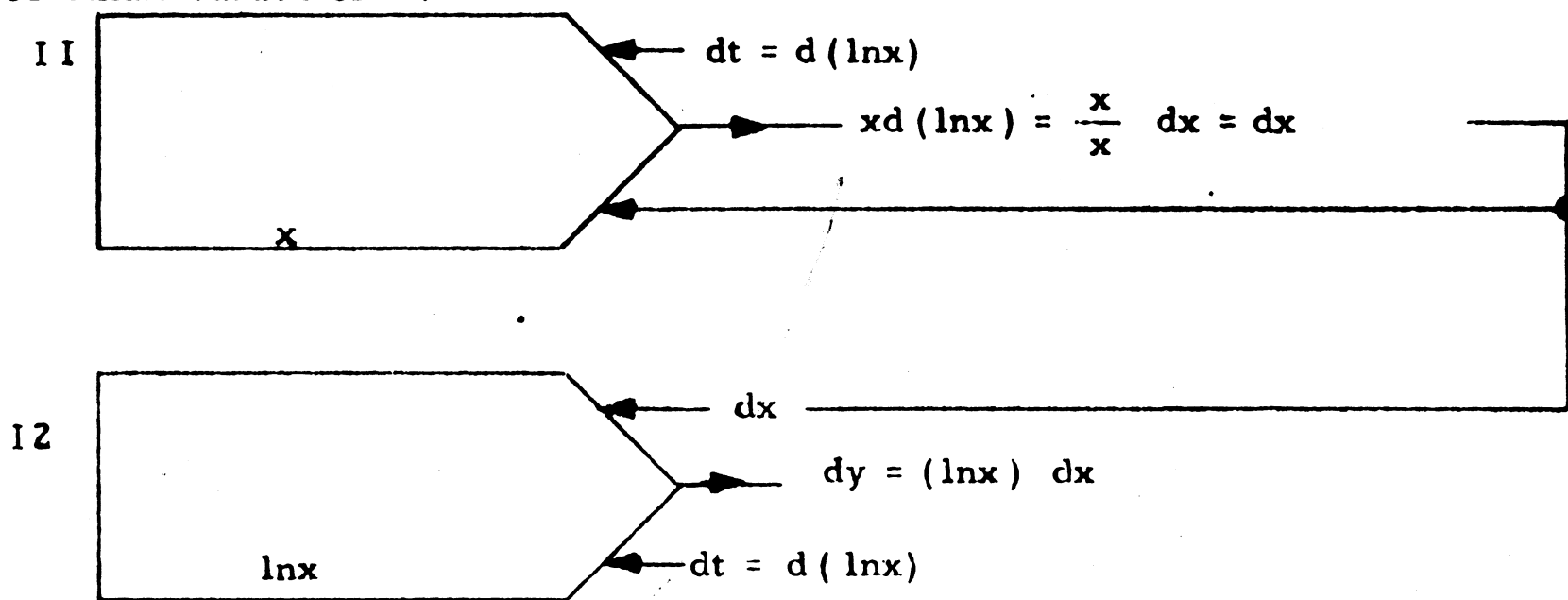
Figure 3-11. Map of Elliptical Spiral

Given: $\frac{dy}{dx} = \ln x$

Solution: $dy = (\ln x) dx$

Let $dt = d(\ln x)$

For small values of x :



NOTE: Integrator 2 uses the machine time, dt , as its secondary input.

For large values of x :

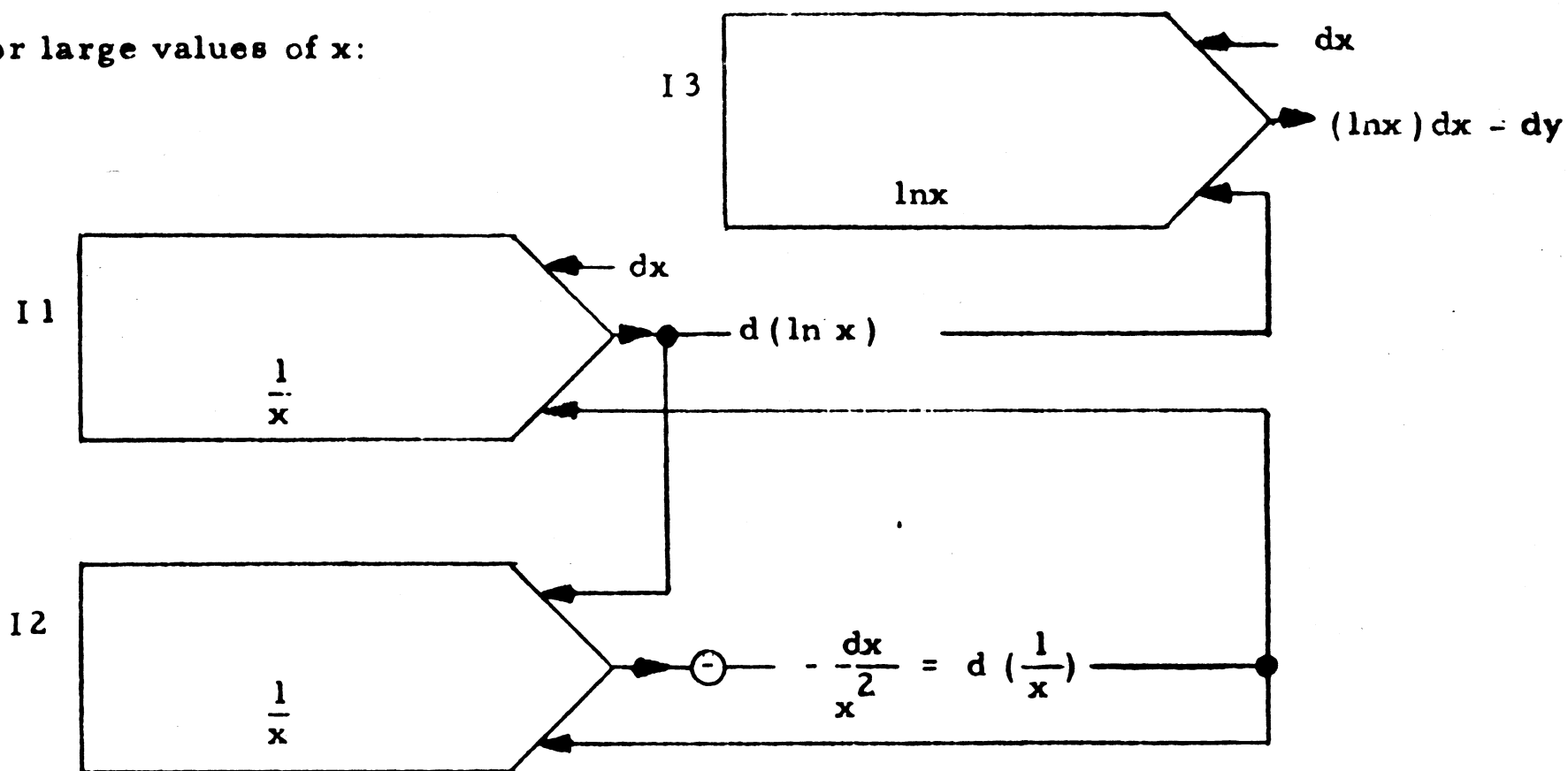
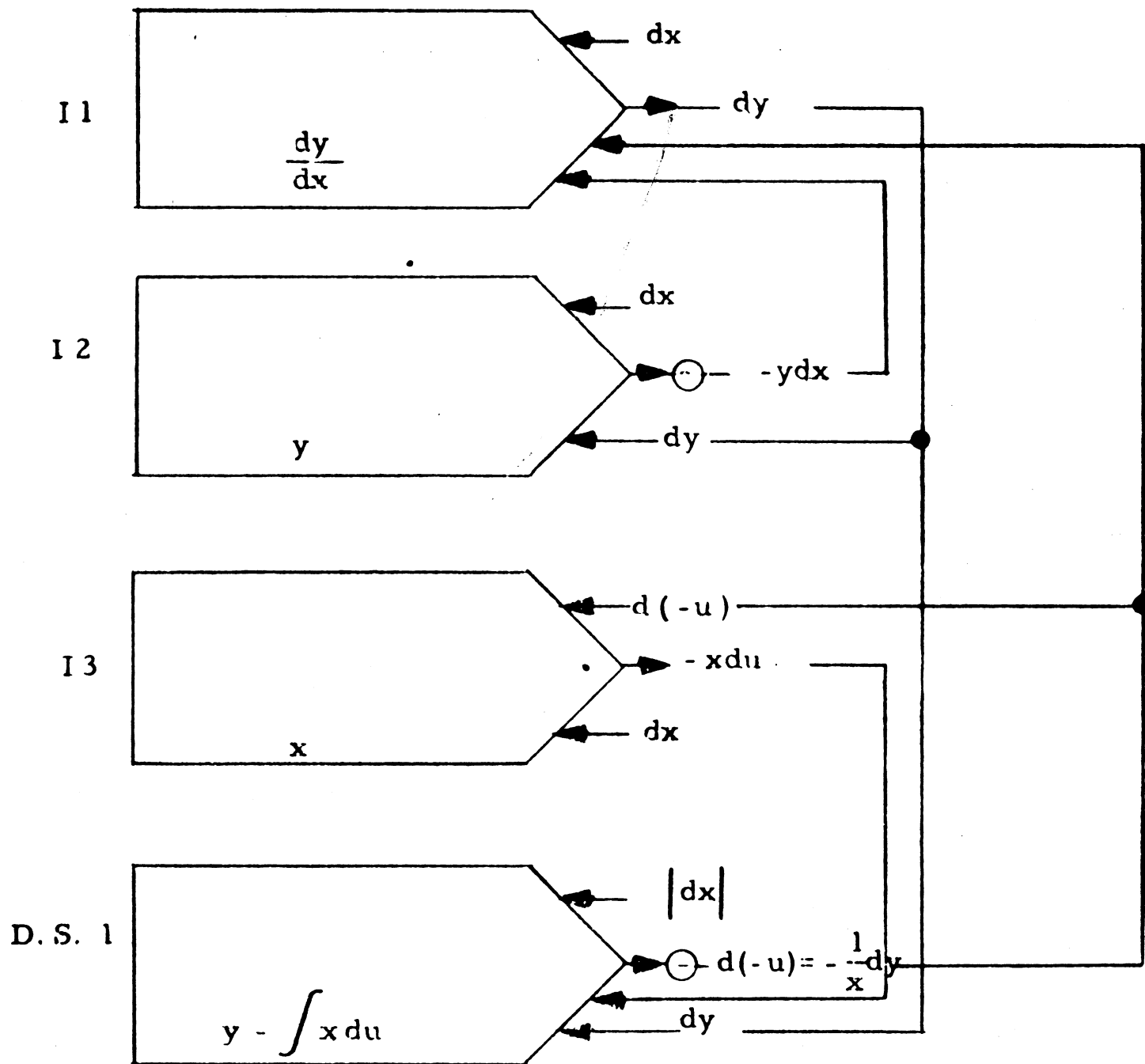


Figure 3-12. Map Using Substitution of Variables.

Given: $\frac{d^2 y}{dx^2} = -\frac{1}{x} \left[\frac{dy}{dx} \right] - y$; let $du = +\frac{1}{x} dy$

Solution: A map using all Integrators for this problem would result in the integrand of an Integrator becoming infinite at $x = 0$. Hence, a Decision Servo is used.



Stability of the Servo Circuit may be proven as described in Example Problem V of Appendix.

Figure 3-13. Map Using Decision Servo to Avoid Infinite Point.

Given: $F(u, v) = y(v) \cos u - x(v) \sin u = 0$
 $u = \tan^{-1} \left(\frac{y}{x} \right)$
 $\frac{\delta F(u, v)}{\delta u} \ll 0$ if $|x| + |y| > 0$

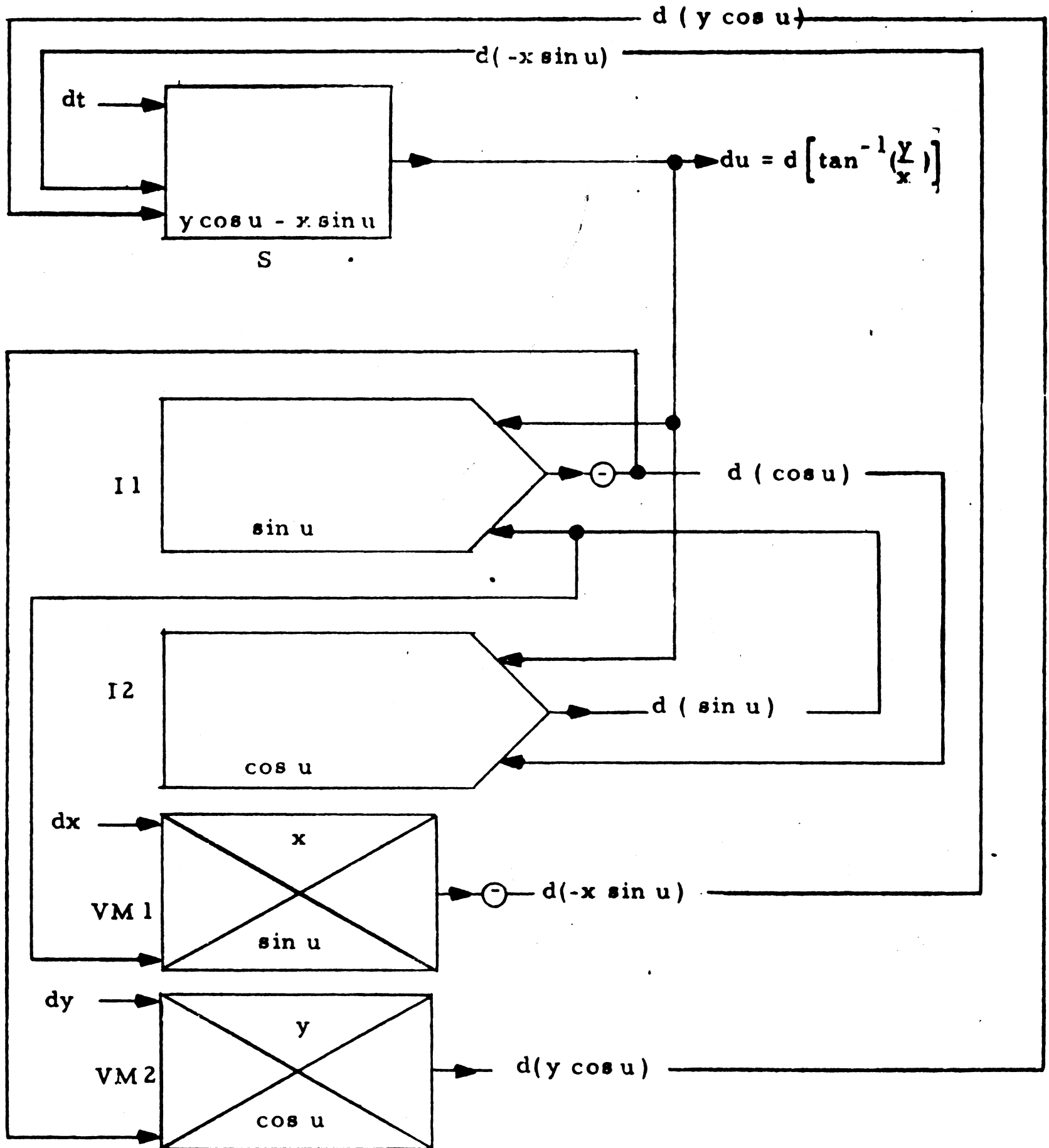


Figure 3-14. Map of Arctangent.

Given: Generate Cosh x and Sinh x.

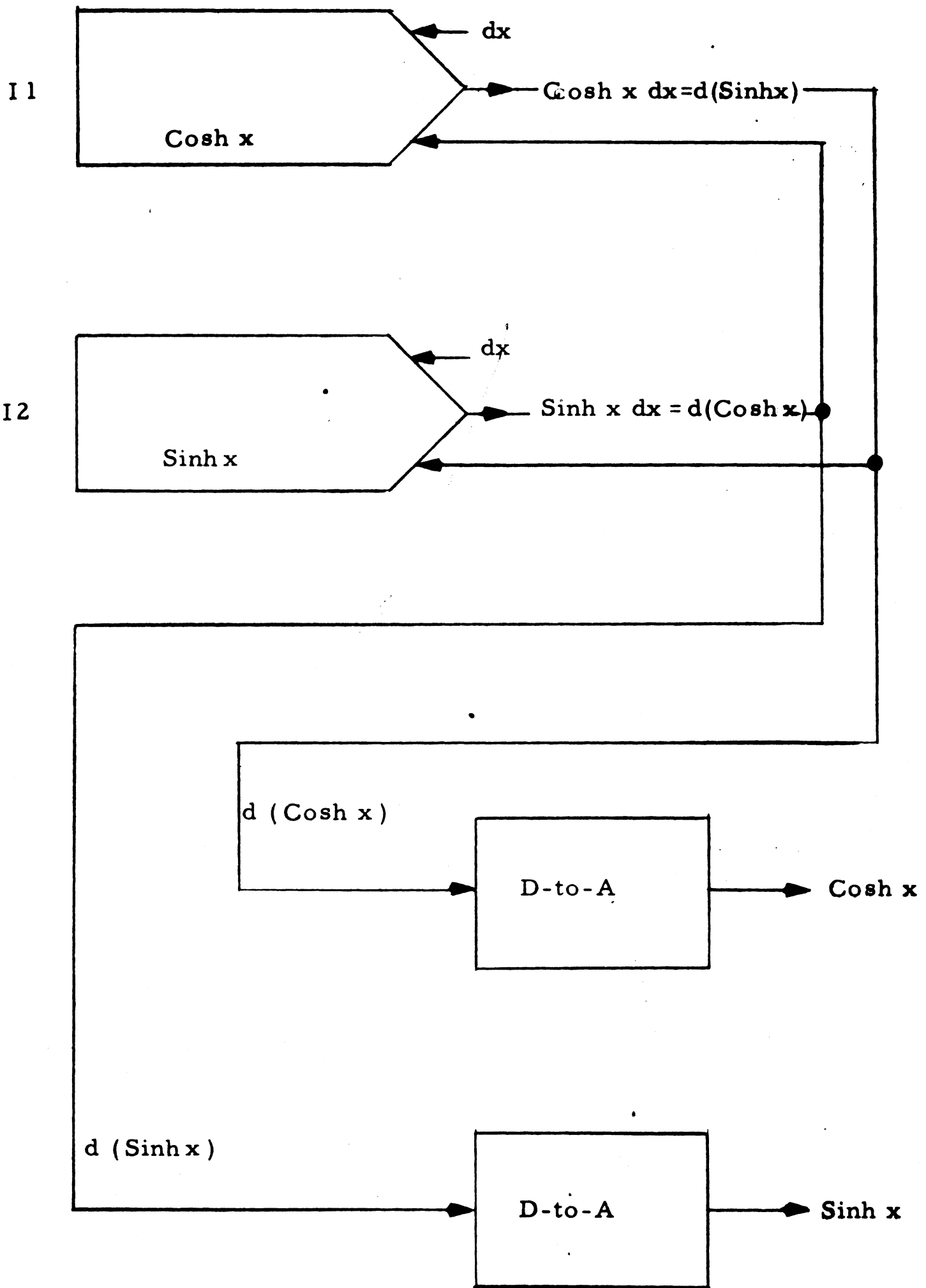


Figure 3-15. Map of Sinh x and Cosh x.

Problem: Generate v^n

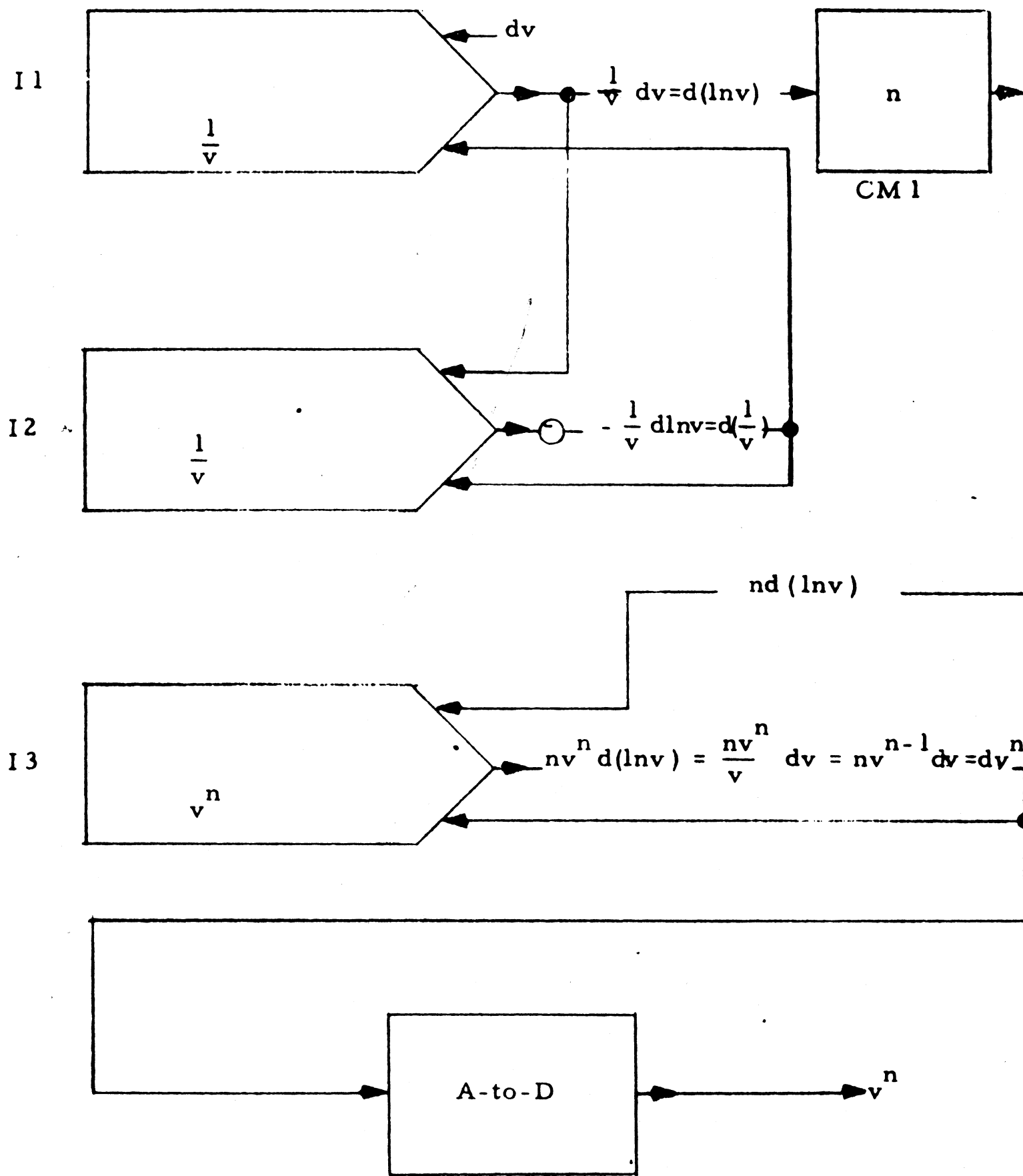


Figure 3-16. Map of v^n

3.4 SCALING.

Scaling is the process that makes the inputs and outputs of the mapped modules compatible. The speed and accuracy of TRICE is determined by the scale factors. Speed can be traded for accuracy as shown in Example Problem I in the Appendix.

The primary inputs, secondary inputs, and outputs of the various computing modules are as follows:

- (a) Integrators have two secondary inputs (when a ΔY Summer is not used), one primary input, one positive and one negative output.
- (b) Constant Multipliers have one primary input, and one positive and one negative output.
- (c) Variable Multipliers have two secondary inputs (which also act as primary inputs), one positive and one negative output.
- (d) Servos (Digital and Decision) have one primary and two secondary inputs (when a ΔY Summer is not used).
- (e) ΔY Summers have six inputs that are summed and serve as the secondary inputs to the Integrator or Servo to which it is permanently connected. Thus, the output of the ΔY Summer is not available on the patchboard. (See Section IV.)

The machine time is 100,000 iterations per second. The maximum output range of the A-to-D converter is $\pm 2^{13}$ increments. The maximum input range of the D-to-A converter is $\pm 2^{13}$ increments. The range of any other input/output equipment used in conjunction with TRICE must be taken into consideration.

Scale powers by definition are the exponents of a selected base and for a binary machine, the base of 2 is used.

3.4.1 Relation of Scaling Factors to Limiting Values of Problem. -

The scale factors for a problem and the initial conditions should be taken from the statement of the problem if possible, and in general these are available in the statement of most problems. Trial and error must be used in selecting these values when they are not stated.

Standard calculus procedures are used to obtain maxima and minima, which constitute the limiting values. However, if such calculations are impractical, or impossible, the values of maxima and minima are estimated or guessed. If the scaling is such that the integrand exceeds the assigned limits, an overflow indication will result and computation may be halted automatically.

The integrand scale power, represented as S_y , is the exponent of the base 2, that, when multiplied by the maximum value of the integrand, will produce an absolute value less than one.

The Variable Multiplier has two variable registers, X and Y; hence it has two integrand scale powers S_x and S_y .

3.4.2 Primary Scale Powers. -

The Integrators, Constant Multipliers, Digital Servos, and Decision Servos have primary inputs. The scale powers associated with these primary inputs are known as the primary scale powers; they are represented by the symbol S_{dx} .

NOTE

Digital and Decision Servos are scaled to a convenient value to allow them to operate properly.

3.4.3 Secondary Scale Powers. -

All computing modules, except the Constant Multipliers have secondary inputs. The scale powers associated with these secondary inputs are known

as secondary scale powers, represented by S_{dy} . The Variable Multiplier has two secondary inputs, S_{dy} and S_{dx} .

The secondary scale powers are determined by the smallest variation of the integrand; i. e., one secondary increment. For example, if it is required that the accuracy of the changes in the integrand be 1/10, the secondary scale power could be +4, because 1/10 is between 1/8, ($\frac{1}{2^3}$), and 1/16 ($\frac{1}{2^4}$).

All secondary factors associated with inputs to a specific module must be equal.

3.4.4 Output Scale Powers. -

All computing modules have outputs; hence, they have output scale factors, which are represented by S_z .

Integrator output scale powers are equal to their integrand scale power plus their primary scale power, i. e.,

$$S_z = S_y + S_{dx}$$

The Constant Multiplier output scale power is equal to the constant's scale power (which is the power to the base 2 that will produce a value smaller than one when multiplied by the constant), plus the primary scale power: i. e.,

$$S_z = S_{dx} + S_k$$

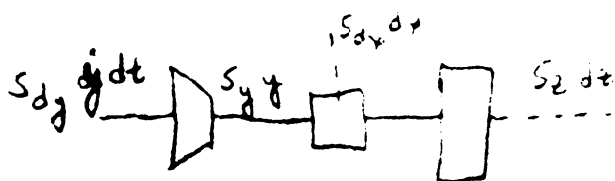
The Variable Multipliers must have their scale powers related as follows:

$$S_z = S_y + S_{dx} = S_x + S_{dy}$$

The length of a register in computing bits (excluding scaling and sign bits) is determined by

Secondary Scale Power - Integrand Scale Power = Register Com-

puting Length, i. e.,



$$(S_{dy} - S_y) = \text{Length of Y-register in bits}$$

For the Variable Multiplier, the lengths of the two variable registers are equal, and the length of each is determined as follows:

$$\text{Length of variable registers in bits} = S_{dx} - S_x = S_{dy} - S_y$$

The various scaling powers are adjusted to obtain compatibility between all units. The Appendix shows the scaling for the Example Problems,

3.5 CODING.

Coding is the process by which the program is made understandable to the TRICE. The map is used to connect the computing modules by means of a patchboard as described in Section IV.

The scaling and initial conditions are filled into the variable registers as described in Section IV. Information may be filled in octal or decimal form.

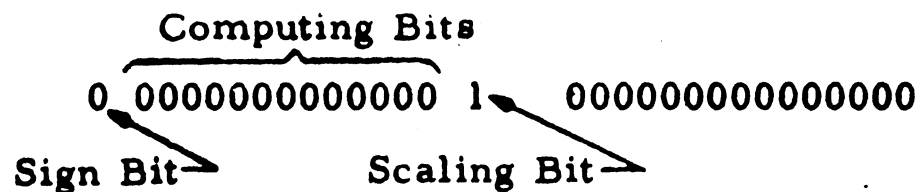
The decimal system is in common use and need not be explained. The octal system is not as common, but just as simple and more exact for use in the TRICE.

The octal code for the fill process is represented by eight keys that correspond to binary digits as indicated in Table 3-2.

Table 3-2. Octal Coding

Key	Binary Digits
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Hence, if it is desired to have a register length of 13 computing positions with an initial value of zero, the desired register should be filled to appear as follows:



This would be accomplished by pushing:

- (a) The zero key four times
- (b) The one key one time
- (c) The zero key five times

The short hand form of writing this is $0_4 1 0_5$.

For another example a register filled with $0_2 1 3 0_2 1 0_3$ would make the register appear as follows:

000 000 001 011 000 000 001 000 000 000

When a decimal code is used for initial conditions and scaling, the TRICE converts the decimal code to an octal code for loading the registers; hence, a truncation error makes the least significant binary bit in the register uncertain.

3.6 INTERPRETATION OF RESULTS.

The method of interpretation of the results obtained from TRICE depends upon the method used to obtain the output. If a tape output is used, the values may be read directly from the tape.

If a plotter or scope is used for the output, the results are known from the scaling of the analog output. The accuracy of the results will depend largely upon the method of obtaining the output.

The single iteration mode of operation is probably the most accurate method of observing the output using a Readout Register; however, this mode of operation is very slow. Usually the output can be obtained by any method to a much higher degree of accuracy than required by the problem.

3.7 SPECIAL PROGRAMMING CONSIDERATIONS.

The previous paragraphs of this section describe straightforward methods of programming. However, for a special class of problems, programming techniques must be used which consider errors that are inherent to the computing modules.

In the straightforward programming methods, the following assumptions were made:

- (a) $dz = ydz$, and
- (b) dx and dz are assumed to be infinitely small increments (differentials).

Errors of the order of an increment (an increment in the R-register) arise in operation due to these assumptions. Hence, if x varies over a range of unity, z may have an error of the order of one increment. This error is very small and will not seriously affect the accuracy of computation as long as the range (in increments) of variation of the independent variable is comparable to the typical register capacity (average register length of modules used). That is, if the typical register length is n bits and the solution (computation) is completed in 2^n iterations or less, the error due to assumption (a) and (b) above, may be neglected. For example, this would be the case for transient solutions when the period of the solution is comparable to the solution time.

However, in those cases where computation proceeds for a longer time (i. e. problems with periodic solutions), the error due to assumptions (a) and (b) above, may accumulate and eventually lead to completely erroneous results. Considerations for reducing these errors will be considered in the following paragraphs.

Two sources which are the major contributors to these errors are:

- (a) Delays which occur between the primary (dx) input to the computing module and its output.
- (b) Round-off.

3.7.1 Compensating for Delays. -

The error due to delays can be reduced by operating in parallel-serial map configuration. The corrective parallel-serial configuration analysis requires the use of the more accurate operation expressions for the computing units as listed below:

- (a) Integrator

$$dz_{(i+1)} = y_{(i)} dx_{(i)} + 1/2 dy_{(i)} \left[\text{sign } dx_{(i)} \right]$$

- (b) Constant Multiplier

$$dz_{(i+1)} = k dx_{(i)}$$

- (c) Variable Multiplier

$$dz_{(i+1)} = y_{(i)} dx_{(i)} + x_{(i-1)} dy_{(i)}$$

- (d) Digital Servo

$$dz_{(i+1)} = \begin{cases} \text{sign } y_{(i)} dx_{(i)} & \text{if } 0 \leq |y_{(i)}| < 1 \\ 0, & \text{otherwise} \end{cases}$$

- (e) Decision Servo

$$dz_{(i+1)} = \begin{cases} \text{sign } y_{(i)} dx_{(i)} & \text{if } -1/2 \leq |y_{(i)}| < 0 \text{ or } 0 \leq |y_{(i)}| < +1/2 \\ 0, & \text{otherwise} \end{cases}$$

NOTE

Increments have subscripts which relate them to the iteration in which they are used; for example, iteration i advances y from $y_{(i-1)}$ to $y_{(i)}$.

Examination of the above operation expressions result in the following conclusions:

- (a) Any computing module has a delay of one iteration (generation) between the primary input and output.
- (b) The Integrator extrapolates y trapezoidally and dy should (for maximum accuracy of extrapolation) be delayed one iteration (generation) with respect to the primary input, dx .
- (c) The Variable Multiplier requires dx and dy to be of the same iteration (generation) for maximum accuracy of the trapezoidal interpolation.

Rules formulated from the above conclusions to achieve maximum operating efficiency are as follows:

- (a) Prepare the map of the problem in a straightforward manner as discussed in the previous paragraphs.
- (b) Number the increments to determine their phase relationship. Start the numbering at the primary input (independent variable) to the Integrator performing the integration of the highest order derivative. Tentatively number all other independent variable inputs (secondary and primary) with the same number. The output of a unit is numbered 1 greater than its primary input.
- (c) Introduce one iteration (phase) delays using Constant Multipliers filled with a constant of 1. Arrange the Constant Multipliers so

that the secondary inputs, dy , to every Integrator which has a dependent variable primary input, has a number greater than its primary input, dx .

- (d) Furthermore, the Constant Multipliers should be arranged to provide each Variable Multiplier with identical numbers for its two inputs.
- (e) Squaring and multiplication should be performed using Variable Multipliers whenever possible.
- (f) All Integrators with independent variable primary inputs have been assigned numbers for their secondary inputs. Hence, the next iteration with an independent variable increment should be assigned a number equal to or greater than any number which has been assigned to any secondary input of the Integrator. For example, if an Integrator which has an independent variable primary input through step (e) above, has the numbers assigned as follows:

dx independent variable increment assigned number 1

dy_1 assigned number 2

dy_2 assigned number 4

Then assign the next independent variable increment, dx , number 4.

- (g) The difference between the numbers assigned to the independent variable in step (a) and in step (f) is the ratio by which this input must be scaled down with respect to the machine independent variable. In the example of step (f), a Constant Multiplier filled with a constant of $1/3$ would be required to generate the problem independent variable from the machine independent variable.

The solution of the problem will be slowed down by this factor;

however, a gain in accuracy may mean the difference between a stable and an unstable solution.

- (h) To improve solution speeds, it may be possible to use different sets of numbers for different Integrators. For example, Integrator 1 may have dx as a primary input with the numbers 1 and 4 assigned to the phase of successive primary increments; at the same time Integrator 2 may also have dx as its primary input, but with the numbers 2 and 5 assigned to the phase of successive primary increments.

NOTE

The necessary delays in this scheme can be generated by Constant Multipliers filled with -1, using the negative outputs.

3.7.2 Compensating for Round-Off. -

A round-off error is due to the finite capacity of the R-register. Initially (start of computation) all R-registers contain a value of $1/2$. Outputs are generated by adding the contents of the Y-register to the R-register until it overflows either in the positive-going or negative-going direction. Therefore, $1/2 K$ increments of the same sign are required at the primary input of a Constant Multiplier containing the constant K to cause an output. If the primary input consists of alternately positive and negative sequences, having fewer increments than $1/2 K$ (i. e. increments of a periodic function of small amplitude), outputs will not be generated and the term thus represented will drop out and the solution will be of a different character (such as damped to undamped). This difficulty can be overcome by using the Constant Multiplier only to generate the existence of the increments of its term; the sign output of the term is taken from the sign of the primary input. Due to the delay between the input and output of the Constant Multiplier, it is necessary that the input increments do not occur more often than every other

iteration so that each output increment will have the correct sign.

3.7.3 Special Programming Considerations, Example Problem. -

As an example of using the special programming procedures described in the above discussion, consider Figure 3-17. Assume that no Variable Multipliers are available for use, thus, necessitating the use of Integrators to perform the squaring process.

Initially, the map of Figure 3-17 is drawn in a straightforward manner. The Constant Multiplier at this time does not appear on the map. Integrators 106 and 108 are being used as Constant Multipliers. The primary input of the Integrator 101 is assigned the number 1. All other dt inputs are also assigned the number "1." This leads to the outputs of Integrators 101, 105, and 106 being assigned the number "2." In turn, the outputs of Integrators 102 and 107 are assigned the number "3," which leads to the outputs of Integrators 103 and 109 being assigned the number "4." Continuing the assignment of numbers the outputs of Integrators 104 and 108 are assigned the number "5." Observing the numbers assigned on the map, it is noted that only Integrator 104 has a secondary input with a number that is not greater than its primary input. Hence, the Constant Multiplier is introduced to produce a delay of "1" iteration. Thus, the secondary input of Integrator 104 has the number "5" assigned to it. Note that a Variable Multiplier should have been used if available. The round-off errors discussed above are not serious in this problem and can be neglected.

Given: $d \left(\frac{dR}{dt} \right) = R d\theta \left(\frac{d\theta}{dt} \right) - \frac{dt}{R^2}$

$d\theta = R_o^2 \left(\frac{d\theta}{dt} \right)_o \left(\frac{dt}{R} \right)$

Also $\left[R_o^2 \left(\frac{d\theta}{dt} \right)_o \right] \left[d \left(\frac{1}{R} \right) \right] = d \left(R \frac{d\theta}{dt} \right)$

* Factor of -2 needed to make this output equal to $d \left(\frac{1}{2R} \right)$ is taken care of in scaling initial condition filled into Integrator 104.

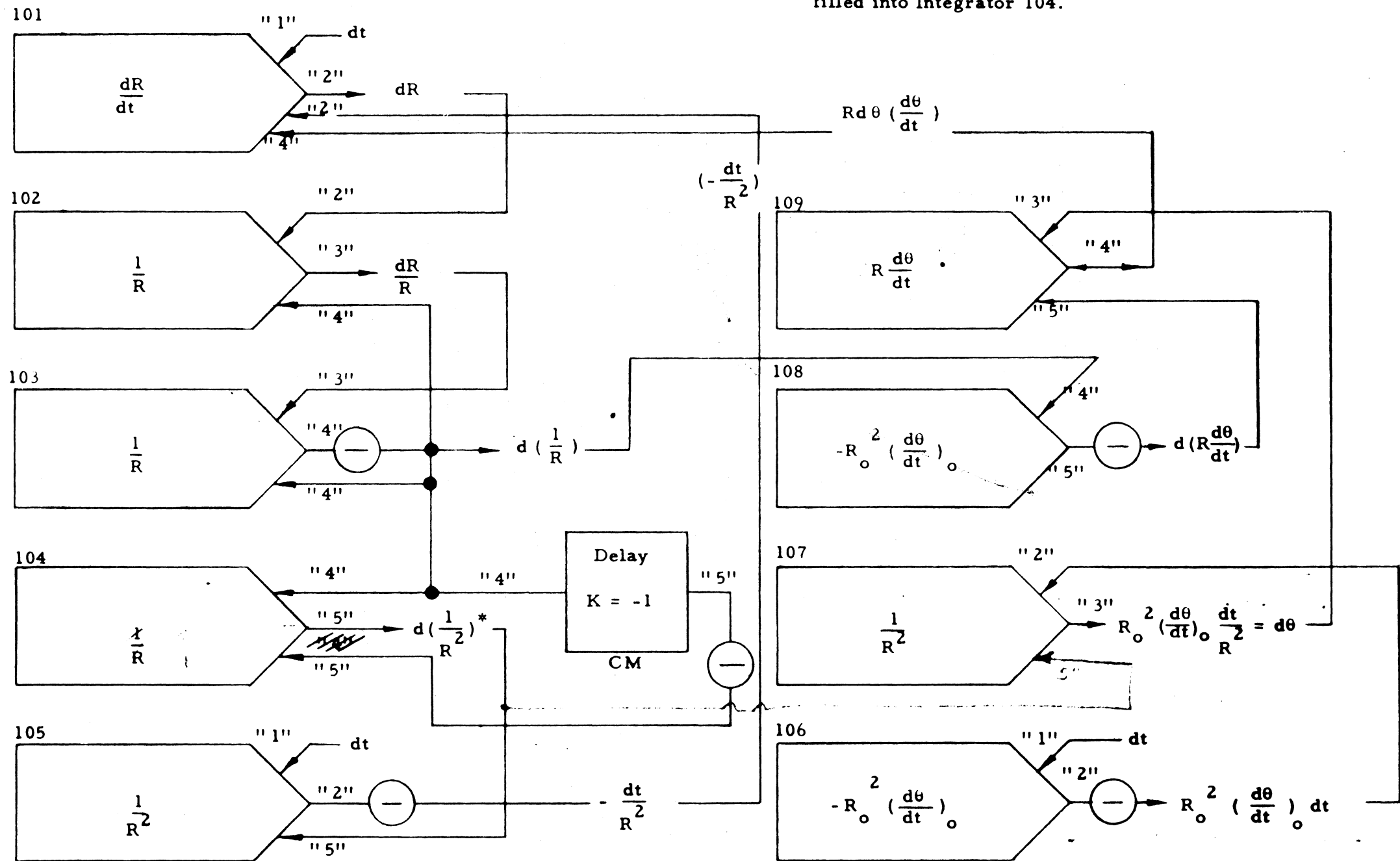


Figure 3-17. Map of Orbit Problem Employing Special Programming Considerations

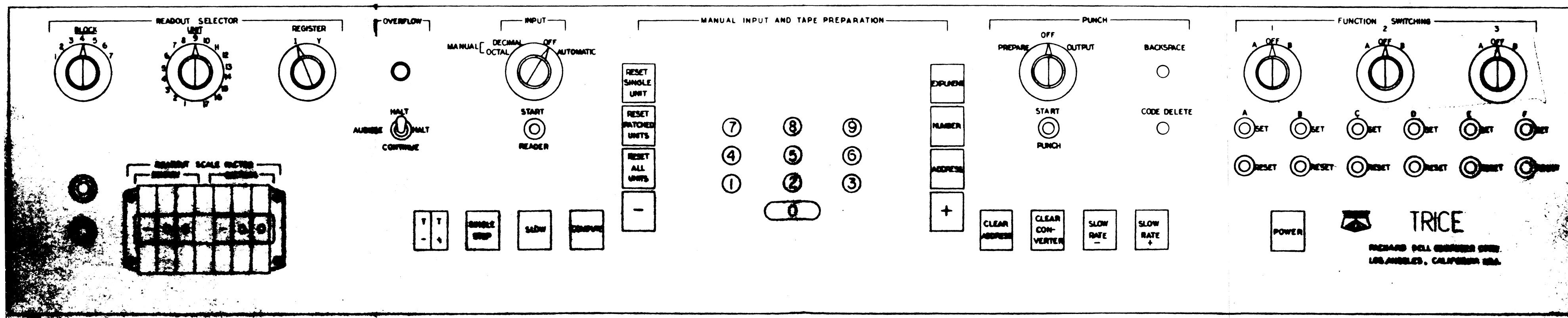
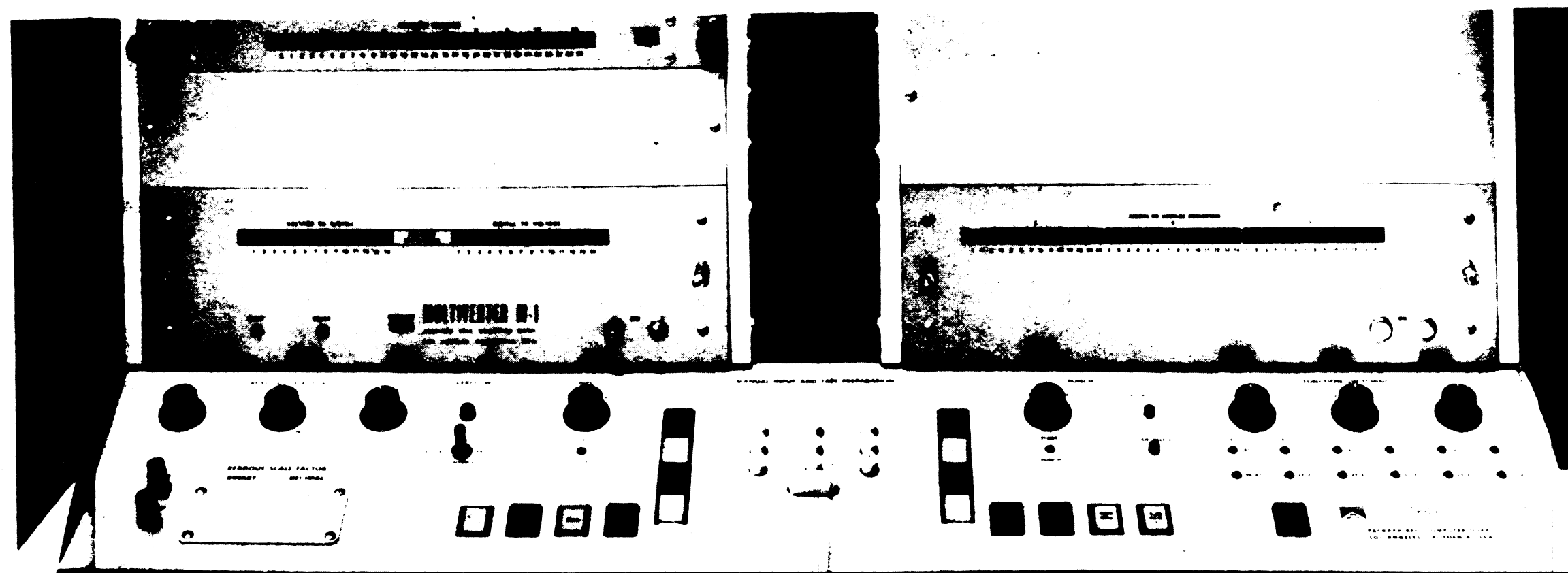


Figure 4-1. Control Panel

SECTION IV

OPERATION

4.1 INITIAL CONTROL POSITIONS.

The front panel controls (see Figure 4-1) should be set to the position indicated in Table 4-1 before applying power.

Table 4-1. Front Panel Controls, Initial Positions

Switch	Position
READOUT SELECTOR BLOCK	Disregard
READOUT SELECTOR UNIT	Disregard
READOUT SELECTOR REGISTER	Disregard
READOUT SCALE FACTOR BINARY	- 0 0
READOUT SCALE FACTOR DECIMAL	- 0 0
OVERFLOW	AUDIBLE HALT
t + / t -	Disregard
INPUT	OFF
SINGLE STEP	Disregard
SLOW	Disregard
COMPUTE	Disregard
RESET SINGLE UNIT	Disregard
RESET PATCHED UNIT	Disregard
RESET ALL UNITS	Disregard
-	Disregard
EXPONENT	Disregard
NUMBER	Disregard
ADDRESS	Disregard

Table 4-1. Front Panel Controls, Initial Positions (Cont.)

Switch	Position
+	Disregard
Keys 0 through 9	Disregard
PUNCH	OFF
CLEAR ADDRESS	Disregard
CLEAR CONVERTER	Disregard
SLOW RATE -	Disregard
SLOW RATE +	Disregard
FUNCTION SWITCHING 1	OFF
FUNCTION SWITCHING 2	OFF
FUNCTION SWITCHING 3	OFF
POWER	Not Depressed (Light Out)

4.2 SYSTEM CLEAR PROCESS.

After applying power to the system, it is necessary to clear the input devices and set them to a state in which they can accept information. This procedure is as follows:

- (a) Set INPUT switch to a MANUAL position (DECIMAL or OCTAL).
- (b) Actuate ADDRESS button.
- (c) Actuate CLEAR ADDRESS button.
- (d) Actuate CLEAR CONVERTER button.

The system is now cleared and ready to accept information.

4.3 AUTOMATIC TYPEWRITER CONTROL.

The functions of some of the panel controls may be made automatic by punched codes on tape. The automatic typewriter keys and tape codes for these functions are shown in Table 4-2.

Table 4-2. Tape Codes

Control Function	Automatic Typewriter Key	Tape Code							
		Flexowriter Channel Designation							
		7	6	5	4	.	3	2	1
		Packard Bell Channel Designation							
		U	V	W	.	X	Y	Z	
0	0	1	0	0	.	0	0	0	
1	1	0	0	0	.	0	0	1	
2	2	0	0	0	.	0	1	0	
3	3	1	0	0	.	0	1	1	
4	4	0	0	0	.	1	0	0	
5	5	1	0	0	.	1	0	1	
6	6	1	0	0	.	1	1	0	
7	7	0	0	0	.	1	1	1	
8	8	0	0	1	.	0	0	0	
9	9	1	0	1	.	0	0	1	
+	Space	0	1	0	.	0	0	0	
ADDRESS	a	1	1	0	.	0	0	1	
NUMBER	n	1	1	0	.	0	1	0	
EXPONENT	x	0	1	0	.	0	1	1	
RESET ALL UNITS	r	1	1	0	.	1	0	0	
CLEAR ADDRESS	.	0	1	0	.	1	0	1	
RESET SINGLE UNIT	s	0	1	0	.	1	1	0	
RESET PATCHED UNITS	p	1	1	1	.	0	0	0	
-	-	0	1	1	.	0	0	1	
TAB	(From TRICE)	1	1	1	.	1	1	0	
BLANK		0	0	0	.	0	0	0	
CODE DELETE	(From TRICE)	1	1	1	.	1	1	1	

Unused

4.4 CONTROLS.

The controls of TRICE consist of rotary switches, digit switches, knobs, buttons, pin jacks, and coax connectors. The controls for the TRICE are shown in Figure 4-1. The following paragraphs describe these controls and connectors; however, the START button on the M1 should be disregarded.

4.4.1 Power Switches. -

The POWER button on the control panel supplies AC power to the Multi-verter and TRICE power supply by means of a terminal strip inside the cabinet. Pushing the POWER button turns power ON or OFF. The power is ON when the button is illuminated. (See Figure 4-1.)

The M1 Multiverter and the MC 72 cases containing input and output logic each have their own power switch on their front panel. (See M1 Manual for description of the M1 Multiverter.)

4.4.2 Power Supply Controls. -

The TRICE power supply generates DC voltages of +50, +8, -8, and -14 volts. The M1 indicator lights require +50 volts, whereas the TRICE elements (not including the M1) require +8, -8, and -14 volts. The voltages can be checked at the output terminals on the power supply.

The +8, -8, and -14 volts can be adjusted over a small range by means of the three knobs on the front panel of the power supply (see Figure 4-2). These knobs are located at the rear of the cabinet. The center knob is the adjustment for the minus 8-volt level. The left knob is the adjustment for the plus 8-volt level and the right knob is the adjustment for the minus 14-volt level.

The Multiverter and the Input and Output units contain their own power supplies. See the M1 Manual for information about the M1 power supplies.

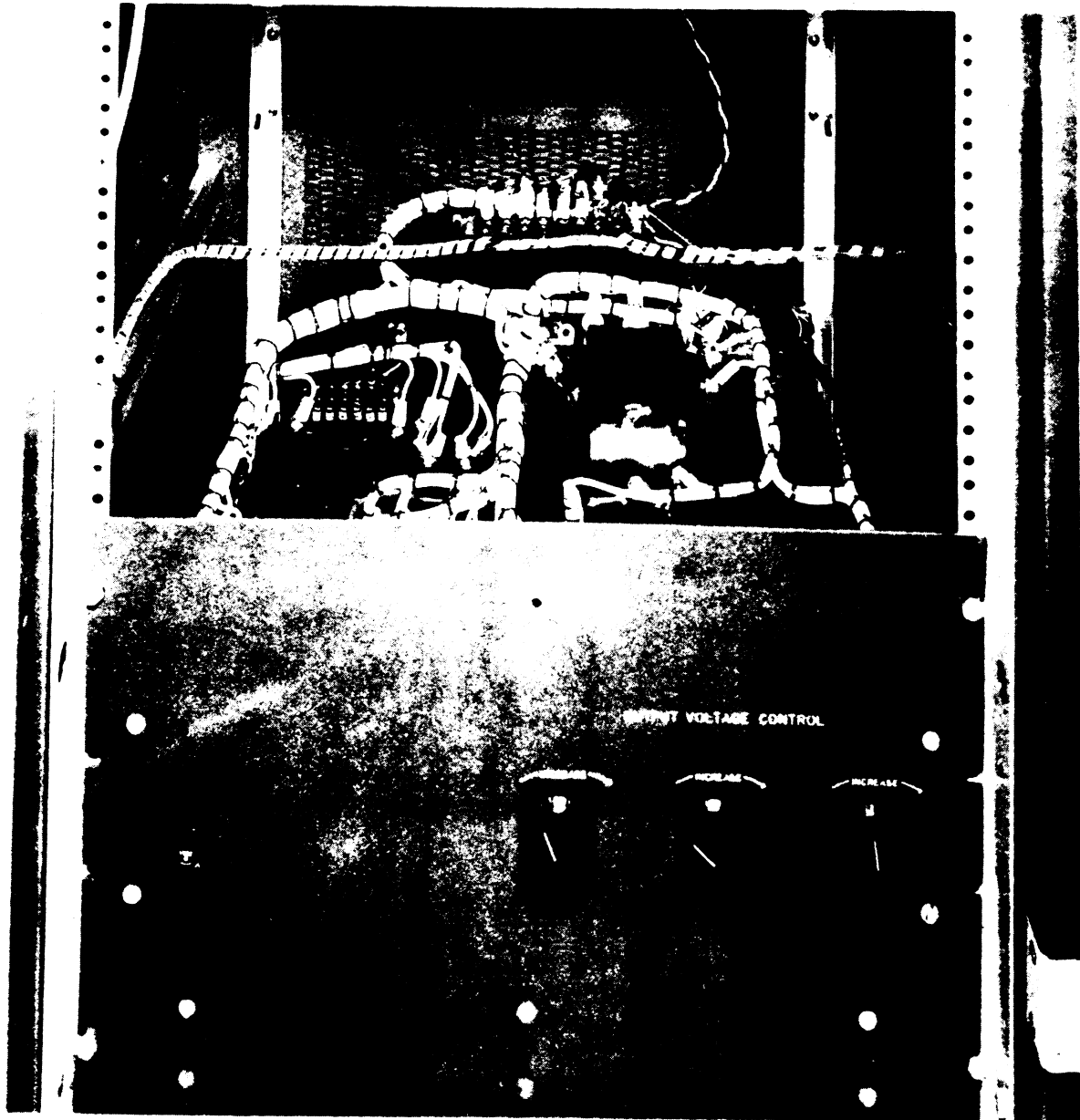


Figure 4-2. Power Supply Controls

4. 4. 3 Input/Output Controls. -

TRICE input/output equipment consists of the following units:

- (a) Control panel keyboard and switches.
- (b) Punched paper tape reader.
- (c) Tape reader logic and address unit.
- (d) Decimal-to-binary and binary-to-decimal converter scaler with decimal display.
- (e) Buffer register with binary display.
- (f) Punch out logic unit.
- (g) Paper tape perforator.

A block diagram of the interconnections of these units is shown in Figure 4-3. Several modes of operation of this equipment are possible using the controls on the Control Panel.

4. 4. 3. 1 Analog Inputs. -

Designated points on the patchboard are the connections for analog inputs to the TRICE. The full-scale input voltage range is from -100 to +100 volts (see the M 1 Manual). If the TRICE system has provisions for analog inputs, the patchboard reference designations are shown on the TRICE system schematics. (See the M 1 Manual.)

4. 4. 3. 2 Analog Outputs. -

If the TRICE system has provisions for analog outputs, the patchboard reference designations for these outputs are shown on the TRICE system schematics. The analog output range is from -6.67 volts to +6.67 volts, with an output impedance of 3333 ohms.

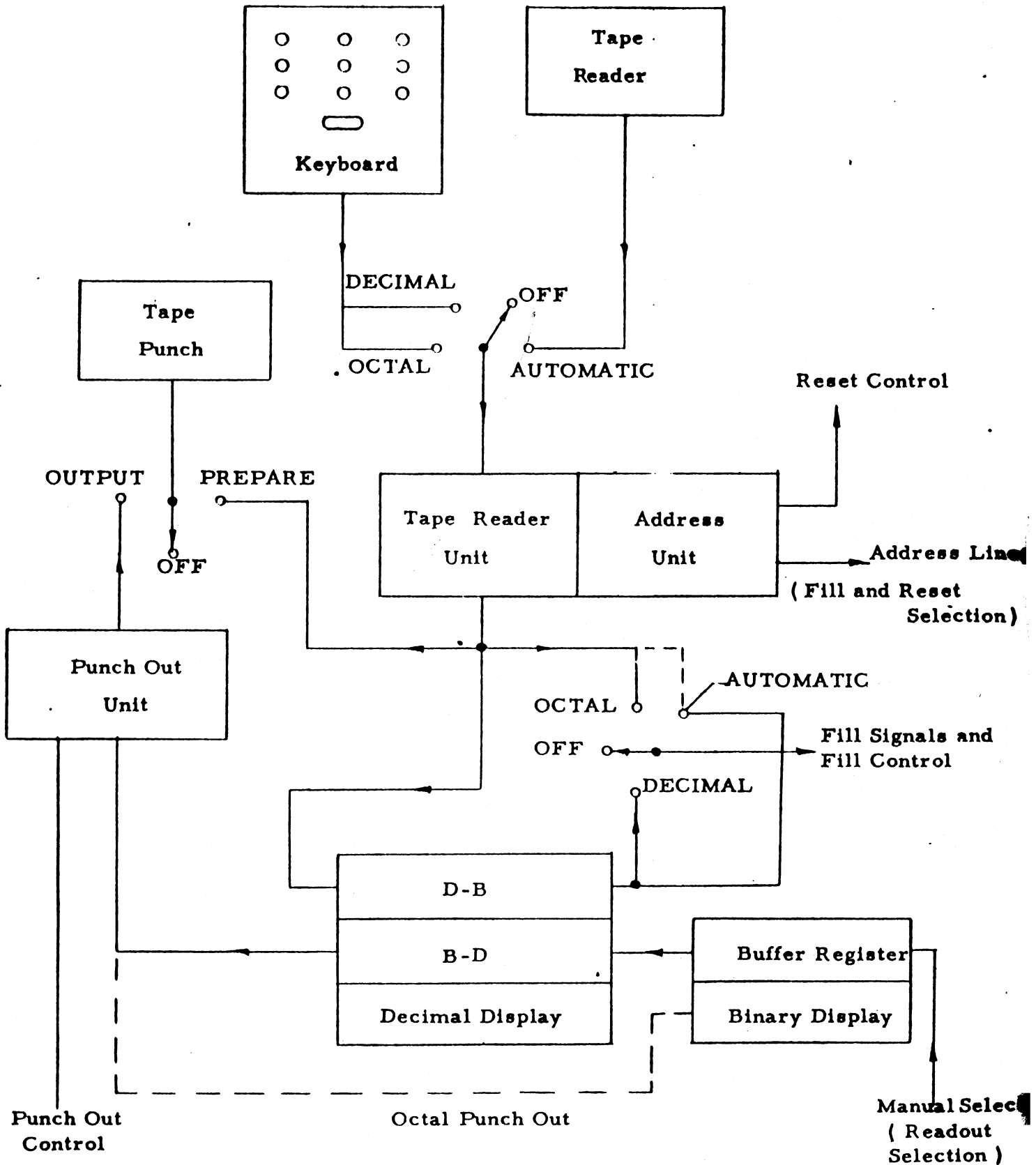


Figure 4-3. Input/Output Equipment, Block Diagram

Normally, the DA 3 converters are used to obtain analog outputs; however, when an M1 Multiverter is not being used to accept analog input information, it may be connected to operate as a D-to-A converter. (See the M1 and DA 3 Manuals .)

NOTE

An analog input to the TRICE must not be used when the M1's are connected as D-to-A converters.

4.4.3.3 INPUT Switch. -

The INPUT switch has four positions as follows:

- (a) In the OFF position, both the keyboard and the tape reader are disconnected.
- (b) In the OCTAL position, information is inserted in octal form via the manual keyboard. The octal mode of operation provides an exact value including the least significant bit.
- (c) In the DECIMAL position, information is inserted in decimal form via the manual keyboard. The conversion of information from decimal form to octal form required for machine operation incurs truncation and round-off errors which make the least significant bit uncertain.
- (d) In the AUTOMATIC position, information is inserted via the tape reader at sixty characters per second.

The controls normally used in a mode of operation are illuminated when the mode is selected by the INPUT switch (See Table 4-3).

Table 4-3. Controls Illuminated by INPUT Mode Selection.

INPUT Switch Position				Control Buttons Illuminated
OFF	OCTAL	DECIMAL	AUTOMATIC	
	x	x		0 through 9
		x		+, -
	x	x		RESET SINGLE UNIT
	x	x		RESET PATCHED UNIT
	x	x		RESET ALL UNITS
	x	x		ADDRESS
	x	x		NUMBER
		x		EXPONENT
	x	x		CLEAR ADDRESS
		x		CLEAR CONVERTER

4.4.3.4 PUNCH Switch. -

The PUNCH switch has three positions as follows:

- (a) In the OFF position the tape punch is disconnected.
- (b) In the PREPARE position, the preparation of tapes is accomplished. In this mode, information is normally supplied from the manual keyboard; however, tapes read on the reader may also be duplicated.
- (c) In the OUTPUT position, a command from a computing module, as selected on the patchboard, causes the contents of a preselected register to be punched on the tape. This information is first presented to the converter, where it is converted from binary-to-decimal and the proper scale factors (which have been set on the READOUT SCALE FACTOR switches) are applied. The decimal information is then presented to the punch unit.

4. 4. 3. 5 READOUT SCALE FACTOR Switches. -

Each initial condition value will have a binary scale factor as described in paragraph 3. 4 of this manual. The binary scale factor for an initial condition value is denoted by the notation 2^s , where s is the binary exponent.

In addition to the binary scale factor, a base ten factor must also be applied. This factor, 10^d , is obtained by representing the initial value as a decimal fraction times ten raised to the appropriate power. For example, if a variable, w , has an initial value of 8765.0000 in decimal, then w may be represented as

$$.87650000 (10^4)$$

or, as a safety conversion measure, it may be represented as

$$.08765000 (10^5)$$

The values of s and d are used by the code converter as described in the following paragraphs.

The code converter is a two-way device capable of converting from decimal-to-binary and from binary-to-decimal.

In the decimal-to-binary mode, the converter accepts a decimal fraction and applies a factor $(2^s) (10^d)$ to it. The values of s and d are entered from the keyboard or tape. The resultant decimal fraction is converted to a binary fraction. This fraction is truncated after r bits, where r is the desired number of bits in the register. The value of r may be entered from the keyboard or the tape. A scaling bit is placed in the position following the least significant of these r bits. The binary information is then shifted out of the converter, three bits at a time, to form octal digits which are loaded into the selected module. The sign bit is placed at its proper position in the most significant octal digit. Negative numbers are loaded in inverted (one's complement) form.

In the binary-to-decimal mode, the converter receives a binary fraction from the readout register and converts it to a decimal fraction. Negative numbers are inverted. A factor of $(2^{-s})(10^{-d})$ is applied to the decimal fraction. The values of s and d are set up on the appropriate READOUT SCALE FACTOR switch positions. The resulting decimal value is displayed on the decimal readout register and may also be punched out on tape through the output unit.

The value of s for the output variable is set up in the BINARY positions of the READOUT SCALE FACTOR switch. The value of d for the output variable is set up in the DECIMAL positions of the READOUT SCALE FACTOR switch. The values of s and d that are set up in the READOUT SCALE FACTOR positions for the output variable are identical to the values of s and d used when loading the initial condition values (via keyboard or tape) for that variable. If w (example above) is selected as output variable, +04 is set up in the DECIMAL positions of the READOUT SCALE FACTOR switch.

4.4.3.6 BACKSPACE Button. -

When the PUNCH switch is in the PREPARE position and the BACKSPACE button is pressed, the tape moves back one space. The BACKSPACE button is used in conjunction with the CODE DELETE button for effectively erasing incorrect information from the tape.

4.4.3.7 CODE DELETE Button. -

If incorrect information is found to exist on the tape, it may be effectively erased by using the BACKSPACE button to backspace to the spot on the tape that has incorrect information. Pushing the CODE DELETE button then causes all tracks of tape to be punched (this is treated by the tape reader the same way as absence of information).

4. 4. 3. 8 START READER Button. -

When the INPUT switch is set to AUTOMATIC and the START READER button is pushed, the tape reader controls the automatic filling of modules as indicated by the code on the tape.

4. 4. 3. 9 START PUNCH Button. -

When the PUNCH switch is in the OUTPUT position and the START PUNCH button is pushed, the selected output will be punched out on the paper tape.

4. 4. 4 FUNCTION SWITCHING Switches. -

The three FUNCTION SWITCHING knobs provide a means of switching rapidly from one program to another by changing the positions of these knobs.

Each of the three FUNCTION SWITCHING knobs is related to three rows (6 holes per row) on the patchboard. When a switch is in position A, the top row of holes is connected to the center row of holes, i. e. , the first top row hole is connected to the first center row hole, the second top row hole is connected to the second center row hole, etc. When a switch is in position B, the center row and the bottom row of holes are connected. When a switch is in the OFF position, the rows of holes are not connected.

4. 4. 5 FUNCTION SWITCHING SET and RESET Buttons. -

The FUNCTION SWITCHING SET and RESET buttons are intended to be used with auxiliary control logic.

4. 4. 6 Computation Controls. -

The following paragraphs describe controls used to

- (a) Enter data for computation
- (b) Prescribe the mode of computation
- (c) Methods of indicating overflows

4.4.6.1 Address of Modules. -

Each computing module has an address number. The ΔY Summers do not have an address number. The address number is composed of one decimal digit (block number) followed by two decimal digits (unit number). The assignment of block and unit numbers is shown in Figure 4-4, page 4-24. Thus, the address number consists of the block number joined with the unit number.

For example, the address of the Constant Multiplier in the upper right hand corner of Block No. 1 is 115. The address of the Integrator adjacent to the Variable Multiplier in Block No. 1 is 101. The Variable Multiplier has two addresses; the lower numbered address is for the X-register and the higher numbered address is for the Y-register. (Note that Servos and Integrators are interchangeable.)

4.4.6.2 OVERFLOW Switch and OVERFLOW Alarms. -

When the capacity of a variable computing register is exceeded (by adding to many increments to it), an overflow is indicated by a light on the front panel of the module and by the OVERFLOW light on the control panel. The Integrators, Variable Multipliers, and Servos have variable computing register; hence, they have OVERFLOW lights.

If the OVERFLOW switch is in the HALT position, computation will stop one iteration after an overflow occurs, and the OVERFLOW light on the control panel will illuminate. If the OVERFLOW switch is in the AUDIBLE HALT position, computation will stop one iteration after an overflow occurs, and the OVERFLOW light on the control panel illuminates and the alarm buzzer sounds. However, when the OVERFLOW switch is in the CONTINUE position, computation will continue regardless of overflows. In this case, the OVERFLOW light on the control panel illuminates when an overflow occurs, and the individual module OVERFLOW lights will illuminate. Overflows are reset by pressing the appropriate reset button.

The slot marked CLOCK on the control unit front panel provides access to the clock bias pot, which is the adjustment for clock width. Moreover, as the Constant Multiplier and the ΔY Summer cannot overflow, their slots should be disregarded.

4. 4. 6. 3 ADDRESS Button (Typewriter "a" Key). -

The address of the various modules is explained in paragraph 4. 4. 6. 1. For manual fill or manual tape preparation, the address of a module is selected by pushing the ADDRESS button and then pushing the three keys of the keyboard that define the address of desired module (most significant digit, i. e. , block digit, first).

If a number of units in the same block and tens scope are to be filled with identical contents, push the block digit, then the common tens digit, and then each of the unit digits.

This button also clears any previously selected addresses, and also resets the parity flip-flop.

4. 4. 6. 4 + Button (Typewriter Space Bar). -

The + button must be pushed before entering positive decimal numbers or positive exponents. This button must also be pressed after filling the last exponent digit, when in the decimal fill mode of operation, before entering the two-digit decimal number that corresponds to the desired register length.

4. 4. 6. 5 - Button (Typewriter " - " Key). -

The - button must be pushed before entering negative decimal numbers or negative exponents.

4. 4. 6. 6 NUMBER Button (Typewriter " n" Key). -

After the desired module has been addressed, the NUMBER button must be pressed before entering the numerical information, including sign, into the modules (or for preparing tape) via the keyboard.

4. 4. 6. 7 EXPONENT Button (Typewriter " x" Key). -

In decimal mode of operation, the EXPONENT button must be pressed before entering the scaling exponents. Each exponent consists of a sign and two decimal digits. The first exponent is the exponent of the base two, determined when scaling the problem, as described in paragraph 3. 4. The second exponent entered is the power to base ten, which is explained in paragraph 4. 4. 3. 5.

4. 4. 6. 8 CLEAR CONVERTER Button. -

The CLEAR CONVERTER button provides a means, in case of an error during the filling of a number or the entering of exponents, to re-start the fill process of the addressed register at the point of pushing the NUMBER button. (Note that the address of the module is retained when this button is pressed.)

4. 4. 6. 9 CLEAR ADDRESS Button. (Typewriter Period Key). -

The CLEAR ADDRESS button provides a means of clearing the addresses held by the reader unit. This provides a means to start over at the point of pushing the ADDRESS button. The CLEAR ADDRESS button must be used in the case of a wrong module addressed, before any number is entered.

If a tape is being prepared, this button puts a stop code on the tape. This stop code is for the TRICE, not for the automatic typewriter.

4. 4. 6. 10 Keyboard. -

The manual keyboard is used for addressing and filling (including scaling).

A module is addressed as follows (INPUT switch in one of the MANUAL positions):

- (a) Push ADDRESS button.
- (b) Push the key that has the module's block number.
- (c) Push the key that corresponds to the module's first digit.
(Modules have two digits in their address; the first numbered module is 01.)
- (d) Push the key that corresponds to the module's last digit.

Initial conditions are inserted into any TRICE initial condition register by the fill process. Furthermore, the fill process is the only process by which a register may be cleared of its previous contents. After the desired register has been selected by address, the register is filled by pushing the NUMBER button, then the appropriate keys (one at a time) of the keyboard.

When the INPUT switch is in the OCTAL position, the first key pressed inserts three binary bits into the addressed register's least significant bit positions. The binary bits correspond to the keys as follows:

KEY	BINARY BITS
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

The second key pressed shifts the first three binary bits three bits toward the most significant end of the register and inserts the binary bits represented by the depressed key into the vacated three least significant bit positions. This process is continued until the keys have been pressed ten times, at which time the register is filled. Hence, the first three binary bits inserted will be in the three most significant bit positions (last three bit positions on the right side of the scope). The last binary "one" inserted is the scaling bit, which determines the length of the register. The least significant bit is the first bit to the left of the scaling bit. The fill process can be observed on a scope or readout register as described in paragraph 4.5.

When the INPUT switch is in the DECIMAL position, push the + or - button, after pushing NUMBER button, to indicate the sign of the number. The first key pressed should then be the most significant decimal digit of the number. The keys are then pressed in order (descending significance of the decimal digits) until a total of eight decimal digits have been entered (zeros are used for later digits if the decimal number is less than eight digits long). The binary scaling exponent is entered after the decimal number, by pushing the EXPONENT button, the pressing the + or - button to indicate sign of the exponent, and then complete the operation by pressing the two decimal digits on the keyboard (most significant first) that correspond to the scaling exponent. The second exponent (tens exponent) is entered in a similar manner. The next step is to push the + button, then the most significant digit of the desired register length, then the last digit of the register length.

It should be noted that tapes may be prepared if the PUNCH switch is in the PREPARE position

4.4.6. 11.M.1 Multiverter RESET Button. -

The RESET button of the M 1 Multiverter may be used to clear its counter. (See the M 1 Manual for more detailed information.)

4. 4. 6. 12 RESET ALL UNITS Button (Typewriter " r " Key). -

When the RESET ALL UNITS button is pressed

- (a) The initial values are inserted into all the variable registers of the Integrators. (The initial condition registers are left unchanged.)
- (b) The R-register of each Integrator and each Constant Multiplier is filled to one-half of its capacity.
- (c) Each Digital Servo Y-register is reset to "zero." Each Decision Servo Y-register is reset to its initial value.
- (d) Both of the converter registers are set to the middle of their range (M l register indicators read 00000000000000, which corresponds to 0 volts).
- (e) The control panel and the module overflow lights are extinguished (buzzer turned OFF if it is ON).

4. 4. 6. 13 RESET PATCHED UNITS Button (Typewriter " p" Key). -

When the RESET PATCHED UNITS button is pressed, the result is as described in paragraph 4. 4. 6. 12, except that only the units patched for reset on the patchboard are effected. (The Multiverter is not reset.)

4. 4. 6. 14 RESET SINGLE UNIT Button (Typewriter "s" Key). -

When the RESET SINGLE UNIT button is depressed, the results are as described in paragraph 4. 4. 6. 12 for the unit whose address has been selected.

4. 4. 6. 15 t + /t - Button . -

When the t + /t - button is pressed, the sign of the positive machine time (accessible at the patchboard as a patched +t) is changed to the sign opposite that which it had before. Pressing the button a second time, again

changes the sign of the positive machine time. The button has two lights in it. When the right light (blue) is illuminated (normal position), the increments generated for positive machine time (+t on patchboard) are positive; when the left light (red) is illuminated, the increments generated for positive machine time (+ t on patchboard) are negative.

When the t + /t - button is pressed, the sign of the negative machine time (accessible at the patchboard as a patched -t) is changed to the sign opposite that which it had before. Pressing the button a second time, again changes the sign of the negative machine time. When the left light is illuminated, the increments generated for negative machine time (-t on patchboard) are positive; when the right light is illuminated, the increments generated for negative machine time (-t on patchboard) are negative.

NOTE

Never push this button during computation.

4.4.6.16 SLOW RATE + Button. -

When the COMPUTE switch is depressed and the SLOW RATE + button is depressed, positive increments at the rate of 100 iterations per second are available at the SLOW Z output holes on the patchboard.

4.4.6.17 SLOW RATE - Button . -

When the COMPUTE switch is depressed and the SLOW RATE - button is depressed, negative increments at 100 iterations per second are available at SLOW Z output holes on the patchboard.

4.4.6.18 COMPUTE Button. -

When the COMPUTE button is depressed, increments will be generated or accepted and computation will be performed. The machine time (dt) rate is

100,000 iterations per second in this mode of operation. The COMPUTE button will stay latched (light ON) until the SLOW or SINGLE STEP button is pressed, at which time it returns to its normal position. If the SLOW button is latched (light ON), the COMPUTE button can not be latched until the SLOW button has been released by pressing the SINGLE STEP button.

4.4.6.19 SINGLE STEP Button. -

When the SINGLE STEP button is depressed, a single iteration of computation is initiated. The button will return to its normal position when released.

This button may be pressed and released to stop computation when the COMPUTE or SLOW button has been depressed.

4.4.6.20 SLOW Button. -

When the SLOW button is depressed, increments will be generated or accepted and computation will be performed. The machine time (dt) rate is 100 iterations per second in this mode of operation. The SLOW button remains latched (light ON) until the SINGLE STEP button is actuated. Depressing the SLOW button will also make the COMPUTE button return to its normal position if it is depressed.

4.5 VISUAL DISPLAY OF REGISTER CONTENTS.

The contents of the various registers can be displayed on a scope by connecting the scope ground to the ground pin jack and the scope input to the desired register pin jack on the module containing the register. (When the contents of a register are displayed on a scope, the least significant digit of the register is on the left side of the scope display.) The scope sync should be connected to the P₃₀ pin jack of the Control unit. (Use low capacitance probes for the scope connections.)

The SYNC pin jack provides access to one of eight possible signals from the Control unit. These signals are clearly identified on the circuit board.

The CLOCK pin jack on the front of the Control unit provides access to the clock output for scope display or sync purposes. The "1" pin jack on the front of the Control unit provides access to the G_1 signal for sync purposes during checkout procedures.

The SHIFT pin jack on the front of the ΔY Summer provides access to the shift signal S_h ; whereas, the $\sum \Delta Y$ pin jack provides access to the output of the ΔY Summer, Y_a .

The two pin jacks to the left of the READOUT SCALE FACTOR switches facilitate scope display of the module selected by the READOUT SELECTOR switches.

When the readout register and code converter are not being used for punching out information, they may be used as visual monitors of the registers throughout the computer. With the PUNCH switch in the OUTPUT position, the newest information must be maintained in the registers for immediate punching, which requires constant shifting-in of new information. Thus, the old information is not held long enough for the eye to register.

When the readout register is not being used as output device, the old information is retained for about twenty times the time required for shifting in new information. This appears visually as though the display register is static, although, when computation is proceeding, the value held by the display register will track the value of the variable display. In addition to the binary display, a decimal display by means of the code converter is available, provided that the code converter is not being used for input or output (i. e., the INPUT switch is set to the OFF or AUTOMATIC position and the PUNCH switch is in the OFF or PREPARE position). In order to display the variables in the same units as the initial condition values, the decimal display will utilize the READOUT SCALE

FACTOR switches on the control panel.

Selection of the monitored register is performed by the manually operated **READOUT SELECTOR** rotary switches.

The three **READOUT SELECTOR** switches are as follows:

- (a) **BLOCK** switch.
- (b) **UNIT** switch.
- (c) **REGISTER** switch.

The modules are selected by address number, as explained in paragraph 4.4.6.1. The **BLOCK** switch is set to the block number of the desired module. The **UNIT** switch is set to the unit number of the desired module. The **I-Y** switch is set to **I** for initial condition registers (**I-register**) and to **Y** for the integrand registers (**Y-register**, also **X-register** for **VM**). The contents of the selected registers are viewed on the **READOUT REGISTER** (binary and/or decimal types) or on a scope.

4.6 PROGRAMMING THE TRICE PATCHBOARD.

All interconnections between computing elements are made on the **TRICE** patchboard by means of patchcords (see Figure 4-5). These connections provide means to transfer increments from one module to another. Increments appear as voltage levels on two lines, the existence line and sign line. Logic "true" corresponds to -7 volts, and logic "false" corresponds to 0 volts. A positive increment appears as a "1" (logic "true," -7 volts) on the existence line and a "1" on the sign line. A negative increment appears as a "1" on the existence line and a "0" (logic "false," 0 volts) on the sign line. The non-existence of an increment is indicated by logic "false" (0 volts) on the existence line; however, the sign line is not significant.

The designation code of the patchboard is described in the following paragraphs.

4.6.1 Division of Patchboard. -

The patchboard is laid out in blocks that corresponds to blocks of computing modules (see Figures 4-4 and 4-5).

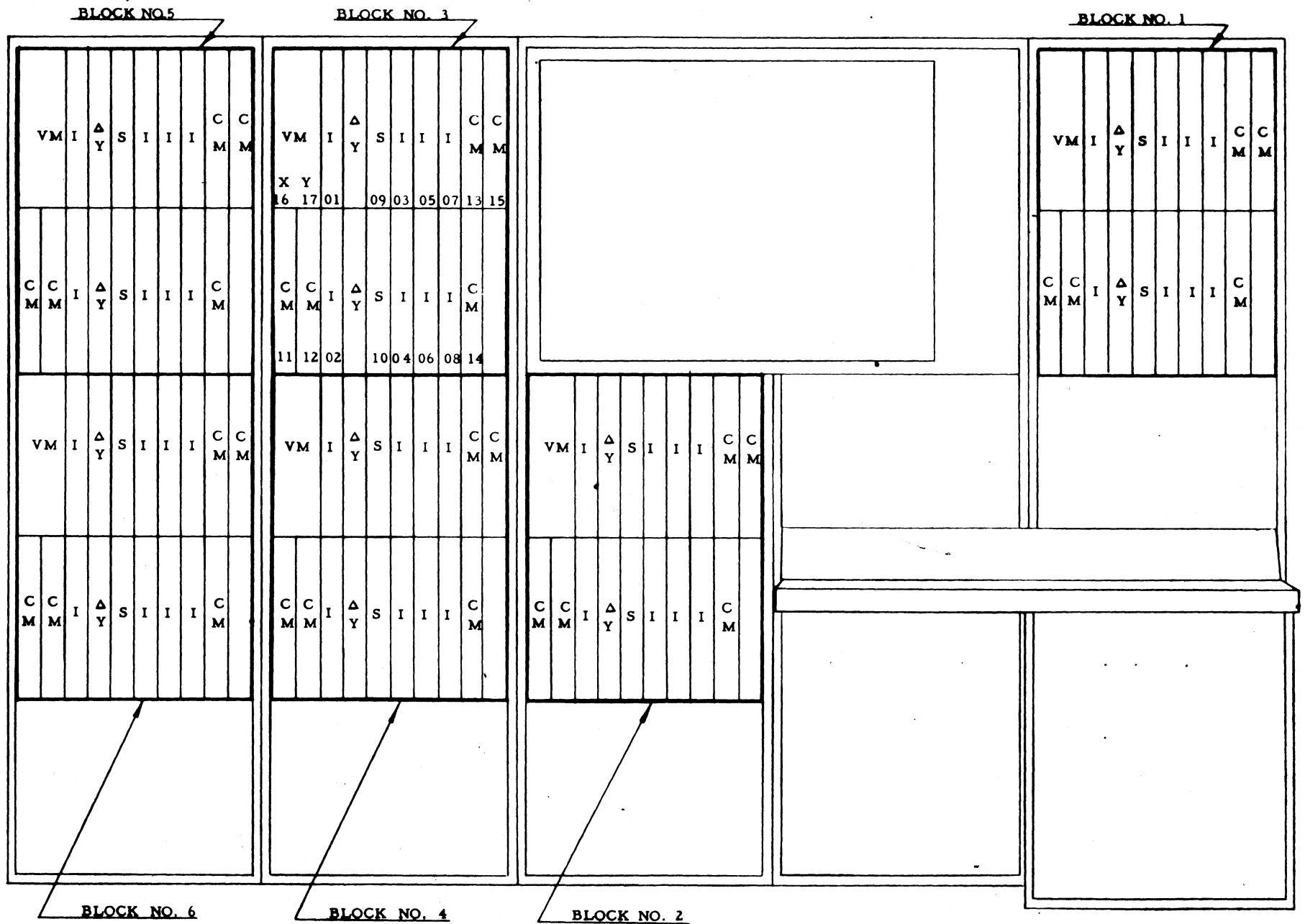
The patchboard code is explained as follows:

- (a) The heaviest lines on patchboard separate the major blocks of information.
- (b) The medium weight lines indicate the boundary of plug holes for a module.
- (c) The lightest lines indicate the bounds of code designations.

4.6.2 Existence Codes. -

The codes used to represent the existence lines are explained below. The existence of an increment is indicated when the line is -7 volts.

- (a) The t code designation is used to specify the existence of machine time.
- (b) The Y_1 and Y_2 code designations are used to specify the existence input for secondary inputs. These codes are used for the Servos and Integrators, as they have two secondary inputs. The ΔY Summer has inputs designated as $Y_1, Y_2, Y_3, Y_4, Y_5,$ and Y_6 .
- (c) The Y code designation is used to specify the existence input for the Y secondary input of the Variable Multiplier.
- (d) The X code designation for the Variable Multiplier is used to designate the existence input for the X secondary input.
- (e) The X code designation (for units other than the Variable Multiplier) is used to specify the existence of the primary input of the module.
- (f) The Z code designation is used to specify the existence of a module's output.



Section IV

Operation

Figure 4-4. Address Block Diagram

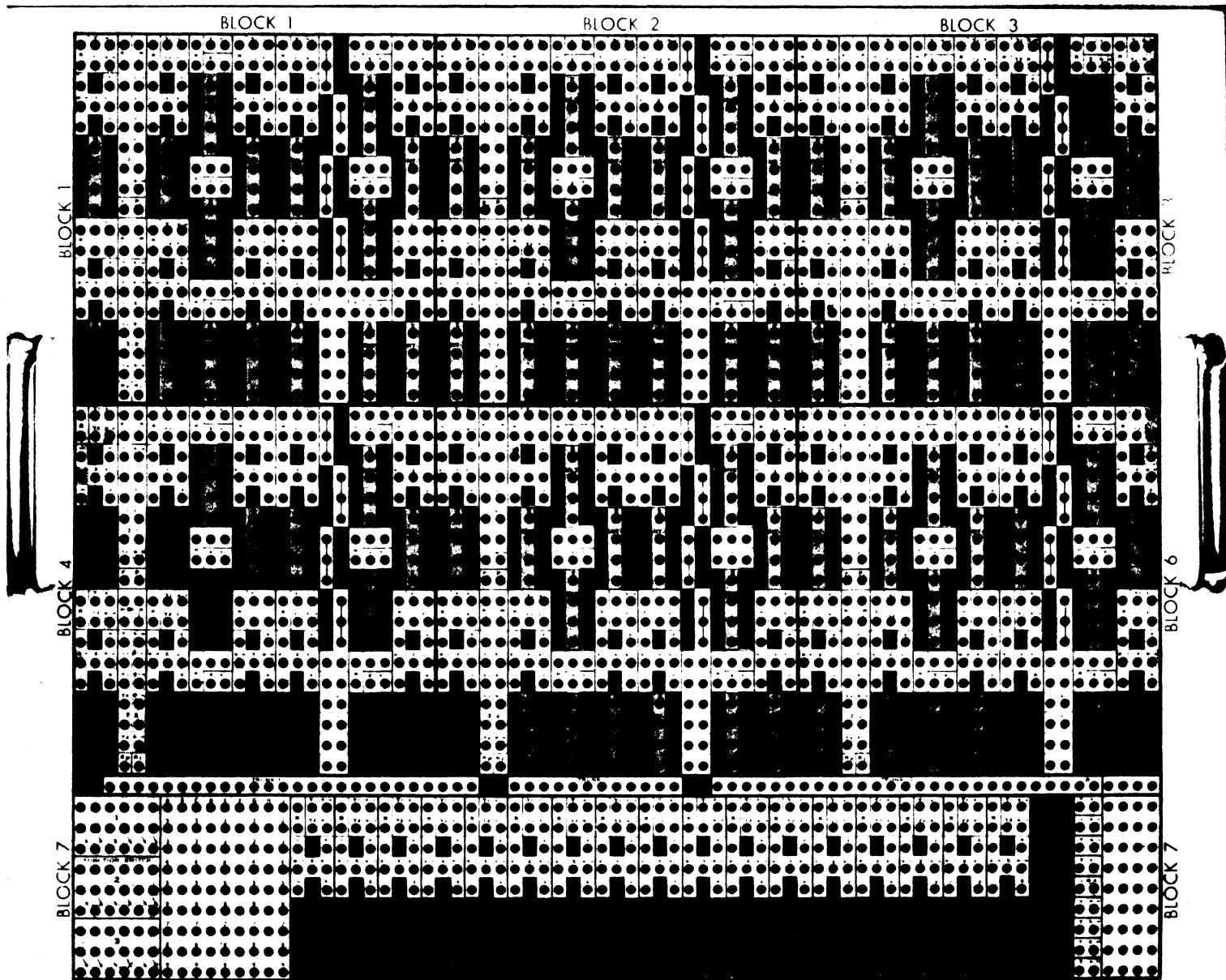


Figure 4-5. TRICE Patchboard

4.6.3 Sign Code. -

The codes used to represent the sign lines are explained below. The sign is positive when the line is - 7 volts and negative when it is 0 volts.

- (a) The + - code designation is used to specify the sign input that is teamed with existence, as indicated to the right or left.
- (b) The - code designation is used to specify the negative sign output that is teamed with existence as indicated to the right.
- (c) The + code designation is used to specify the positive output that is teamed with existence as indicated to the left.

4.6.4 Convenience Lines. -

The codes described below are used to designate lines that are included as a convenience to facilitate patchboard connection.

- (a) The J-1 through J-11 plug holes of the individual blocks are internally connected to corresponding holes on the other blocks, i. e., the right J-1 holes of blocks 1, 2, 3, 4, 5, and 6 are all jumpered together, etc.
- (b) The plug holes above the J-6 plug holes are jumpered on the back side of the patchboard in groups of three, as indicated by jumper lines on patchboard.
- (c) TRUNK 1, TRUNK 2, and TRUNK 3 designations are intended to serve as a means to connect additional TRICE systems.

4.6.5 Reset Codes. -

The codes described below are used in conjunction with reset operations.

- (a) The R code designation is used to specify a module's RESET plug

holes. If two R plug holes of a module are jumpered by a plug, this module will be reset by the reset-patched-units signal.

- (b) The R_t code designation is used to specify the total reset control. When the signal patched into the R_t hole goes "true," a total reset occurs, thereby resetting all units.
- (c) The R_p code designation is used to specify the patched reset control. When the signal patched into the R_p hole goes "true," the modules patched for reset, as described in (a) above, are reset to their initial conditions.

4.6.6 Overflow Codes. -

The O_f code designation is used to specify the overflow signal. An overflow of any unit, causes an overflow signal, which is made available at the O_f plug hole. This signal may be used for control functions such as automatic reset (R_p , R_t), fill (F_c) and punchout (P_c).

4.6.7 Start Reader. -

The F_c code designation is used to specify the start reader signal, which is initiated by a signal patched to this plug hole. The INPUT switch must be in AUTOMATIC position.

4.6.8 Start Punch. -

The P_c code designation is used to specify the start punch signal, which is initiated by the signal patched to this plug hole and causes the punching of the contents of the selected register on tape. The PUNCH switch must be in the OUTPUT position.

4.6.9 Buffer. -

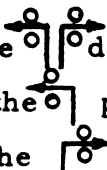


The B code designation is used to specify the existence of the buffer's

outputs. The buffer is used when one computing module's output provides inputs to many computing units. The B_{in} code designation is used to specify the existence input to the buffer.

4.6.10 Function Switch Code. -

The three sets of function switch plug holes are connected such that the middle row of holes is connected to the bottom row (6 plug holes per row) when the applicable FUNCTION switch is in the B position. The center row is connected to the top row when the FUNCTION switch is in the A position. The rows are not connected when the switch is in the OFF position. It should be noted that the holes in a row are not jumpered together.

4.6.11 ΔY Summer Connections. -

The  designations are used to select the output of a ΔY Summer. Jumpering the  pair of holes connect the adjacent Integrator; whereas, jumpering the  pair of holes connect the adjacent Servo.

4.6.12 Multiverter Connections. -

In Block No. 7, lower right hand corner, the code designations are as follows:

- (a) The Y designation indicates the input existence line for the D-to-A converter.
- (b) The + - designation indicates the sign line for the D-to-A converter.
- (c) The E out indicates an output of the D-to-A.
- (d) The REF indicates the output of a precision -20 volts reference voltage.
- (e) The designations of P_1 in, P_2 in, P_3 in, P_4 in, are intended to be the inputs to four pots; whereas, P_1 out, P_2 out, P_3 out, and P_4 out, are intended as the four pots' outputs.

- (f) The designations A_1 in, A_1 out, A_2 in, A_2 out, A_3 in, A_3 out, A_4 in, A_4 out, refer to plug holes which are intended to be used for operational amplifier inputs and outputs (used to match the TRICE output to an analog system).

NOTE

The panel to the right of the patchboard is provided for mounting components such as pots and operational amplifiers. The pots may be used to adjust the output voltages of the D-to-A converters to values suitable for inputs to a device such as an analog plotter. The operational amplifiers in this case are used to match the TRICE output impedances to the input impedances of the analog plotter.

The plug holes, plotter Y and X, refer to the provision for a plotter's input connections.

4.7 RULES FOR PATCHBOARD CONNECTIONS.

The rules for connecting the patchboard should be observed as indicated in the following paragraphs.

4.7.1 The A-to-D Output. -

The A-to-D outputs can be used as dY or dX for all TRICE computing elements. The maximum register length can be 24 bits.

4.7.2 Input for the D-to-A. -

The D-to-A input can come from any source (dt, I, CM, Servo, VM, A-to-D).

4.7.3 Input for the A-to-D Connected to Operate as a D-to-A. -

When an M 1 is connected to operate as a D-to-A, the input can come from any source (dt, I, CM, VM, Servo).

4.7.4 Restrictions for Input to Destinations. -

An increment in every iteration is designated by dt. For + dt, connect the existence to t and the sign of the input to +. For - dt, connect the existence of the input to t and the sign of the input to -.

Because of timing differences in generating outputs, all sources cannot be used as inputs to all destinations.

The dZ outputs from I, CM, VM, S, or dt may be used as dX or dY inputs to any or all units. This allows the I, CM, VM, and Servo registers to have a maximum length of 26 bits.

The dZ output from the A-to-D converter may be used as dY or dX inputs to any or all units. This allows the I and Servo registers to have a maximum length of 24 bits.

If a ΔY Summer is used in connection with an Integrator or Servo the register length of that unit is restricted to 23 bits.

NOTE

The length of a computing register is the number of significant binary bits excluding sign and scaling bit.

As mentioned previously, though, source terminals must not be connected to other source terminals.

The slow rate (SLOW Z) can not be used as an input to a D-to-A converter.

SECTION V

DIGITAL INTERPOLATOR

(Arbitrary Function Table)

This Section describes the Digital Interpolator (an arbitrary function table). The Digital Interpolator is a unit which provides a means to generate arbitrary functions, i. e., functions defined by numerical points on the curve. The unit interpolates between the numerical defining points to generate a smooth curve of the arbitrary function. This Section will be provided at a later date.

APPENDIX

**This Appendix contains TRICE
programs of Sample Problems**

Example Problem 1: Sine Curve

Problem: Solve $y = A \sin t$

Solution: This problem must be written in differential form. Thus,

$$dy = A \cos t dt$$

Map:

- (1) Assume $A \cos t$ is the integrand of Integrator 1.
- (2) Make dt the primary input; thus, $A \cos t dt$ is the output of Integrator 1.

$$A \cos t dt = d (A \sin t)$$

- (3) For the secondary input, $d (A \cos t)$ is needed for Integrator 1.
- (4) Assume $d (A \cos t)$ is the output of Integrator 2.

$$d (A \cos t) = -A \sin t dt$$

- (5) Hence, dt is the primary input to Integrator 2.
- (6) Furthermore, the integrand of Integrator 2 is $(- A \sin t)$.
- (7) The required secondary input to Integrator 2 is $d (-A \sin t)$, which is $- A \cos t dt$.
- (8) It is noted that the needed secondary input for Integrator 1 is now available from Integrator 2.
- (9) It is also noted that the needed secondary input for Integrator 2 is available in the output of Integrator 1, except that the sign is wrong.
- (10) However, if the output of Integrator 1 is connected to the secondary input of Integrator 2, the contents of Integrator 2 becomes $+ A \sin t$ and the output of Integrator 2 will remain $- A \sin t dt$

Appendix

if the signal is taken from the negative output. The minus sign at the output of Integrator 2 in Figure A-1 indicates a reversal of sign.

- (11) The output from the TRICE is taken from Integrator 1; i. e.,

$$dy = A \cos t dt$$

Initial Conditions:

- (1) From $A \cos t$, we find at $t = 0$ that the Y-register of Integrator 1 will be A. The contents of Integrator 2, which is $A \sin t$, must initially be zero. The map is shown in Figure A-1.

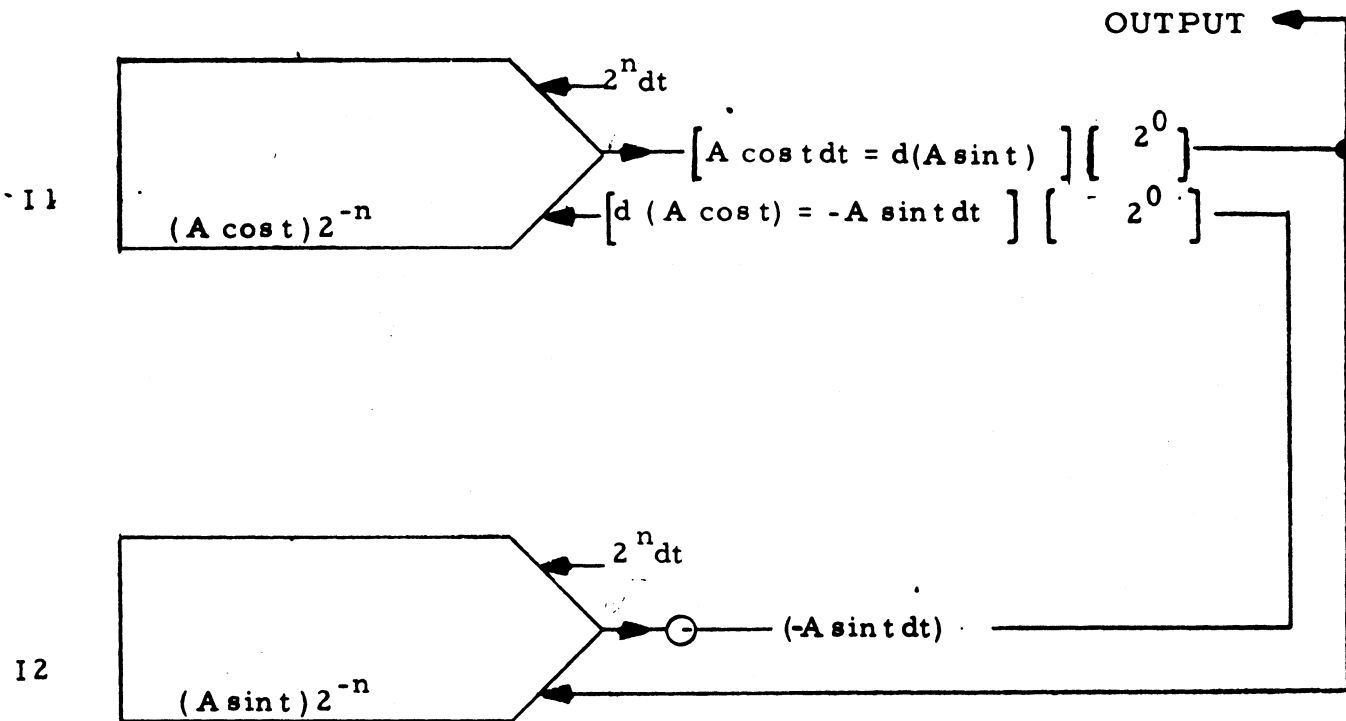


Figure A-1. Map of Sine - Cosine Curve

Appendix

Scaling: Since the Y-register of the two Integrators contain $A \cos t$ and $A \sin t$, which have the same limit A ; therefore, the length of the registers must be the same. Thus, in this case, scaling only requires selecting a register length. By increasing the register length, accuracy may be gained.

Table A-1 shows the characteristics associated with various register lengths. By shortening the register length, speed may be gained.

One increment of t = one increment of machine time = 10^{-5} seconds.

$2^n dt$ means there are 2^n increments in a unit of t . One unit of t = $2^n 10^{-5}$ seconds.

Table A-1. Accuracy vs. Speed Table for Sine Curve

n (Register Significant Binary Bits)	A (Amplitude Increments)	2^n (Unit of t in Increments)	Period in Increments	Period in Milli Seconds	Frequency in Cycles/Second
1	1	2	12.6	0.126	7940
2	3	4	25.1	0.251	3980
3	7	8	50.3	0.503	1980
4	15	16	101	1.01	990
5	31	32	201	2.01	498
6	63	64	402	4.02	248
7	127	128	805	8.05	124
8	255	256	1610	16.1	62.1
9	511	512	3220	32.2	31.0
10	1023	1024	6450	64.5	15.5
11	2047	2048	12900	129	7.9
12	4095	4096	25700	257	3.89
13	8191	8192	51400	514	1.95
14	16383	16384	103000	1030	0.97

Appendix

Programming Patchboard:

Connect the patchboard as described in Section IV.

Coding:

After the desired scaling has been selected, the initial conditions and the scaling bit are filled as described in Section IV.

Output:

The output of the TRICE may be used to display the curve on a scope, plotter, or other output equipment.

Example Problem II. Damped Sine Wave.

Given: A damped wave that is defined by

$$\frac{d^2 \Theta}{dt^2} + 2K \frac{d\Theta}{dt} + w^2 \Theta = 0$$

For $K \ll w$ then $\Theta \approx A \sin wt$.

Solve this problem by using TRICE.

Solution: Separate the highest derivative

$$\frac{d^2 \Theta}{dt^2} = -2K \frac{d\Theta}{dt} - w^2 \Theta$$

Map: The map is shown in Figure A-2.

Scaling: (1) Derive the first and second order derivatives from

$$\Theta \approx A \sin wt$$

that is,

$$\frac{d\Theta}{dt} \approx A w \cos wt,$$

and
$$\frac{d^2 \Theta}{dt^2} \simeq -A w^2 \sin wt$$

- (2) Assume dt scale factor to be 2^{17} , because this value is close to the real-time of the machine.
- (3) Make the amplitude of Θ equal to 2^{13} , which allows the full range of the D-to-A to be used provided that $d\Theta$ is scaled for 2^0 .
- (4) Thus, for Integrator 2

$$\begin{aligned} S_z &= 0 \\ S_{dx} &= +17 \\ S_z &= S_{dx} + S_y \end{aligned}$$

Hence,
$$S_y = -17$$

which indicates the maximum allowable amplitude of $\frac{d\Theta}{dt}$ is 2^{+17} .

- (5) Determine the value of w as follows:

Since
$$\frac{d\Theta}{dt} \simeq A w \cos wt$$

and
$$\left[\frac{d\Theta}{dt} \right]_{\max} \simeq 2^{17},$$

therefore
$$Aw \simeq 2^{17}.$$

Given
$$\Theta \simeq A \sin wt$$

and from (3)
$$\Theta_{\max} = 2^{13},$$

thus,
$$A = 2^{13}.$$

Hence,
$$w \simeq \frac{2^{17}}{2^{13}}$$

or
$$w \simeq 2^4$$

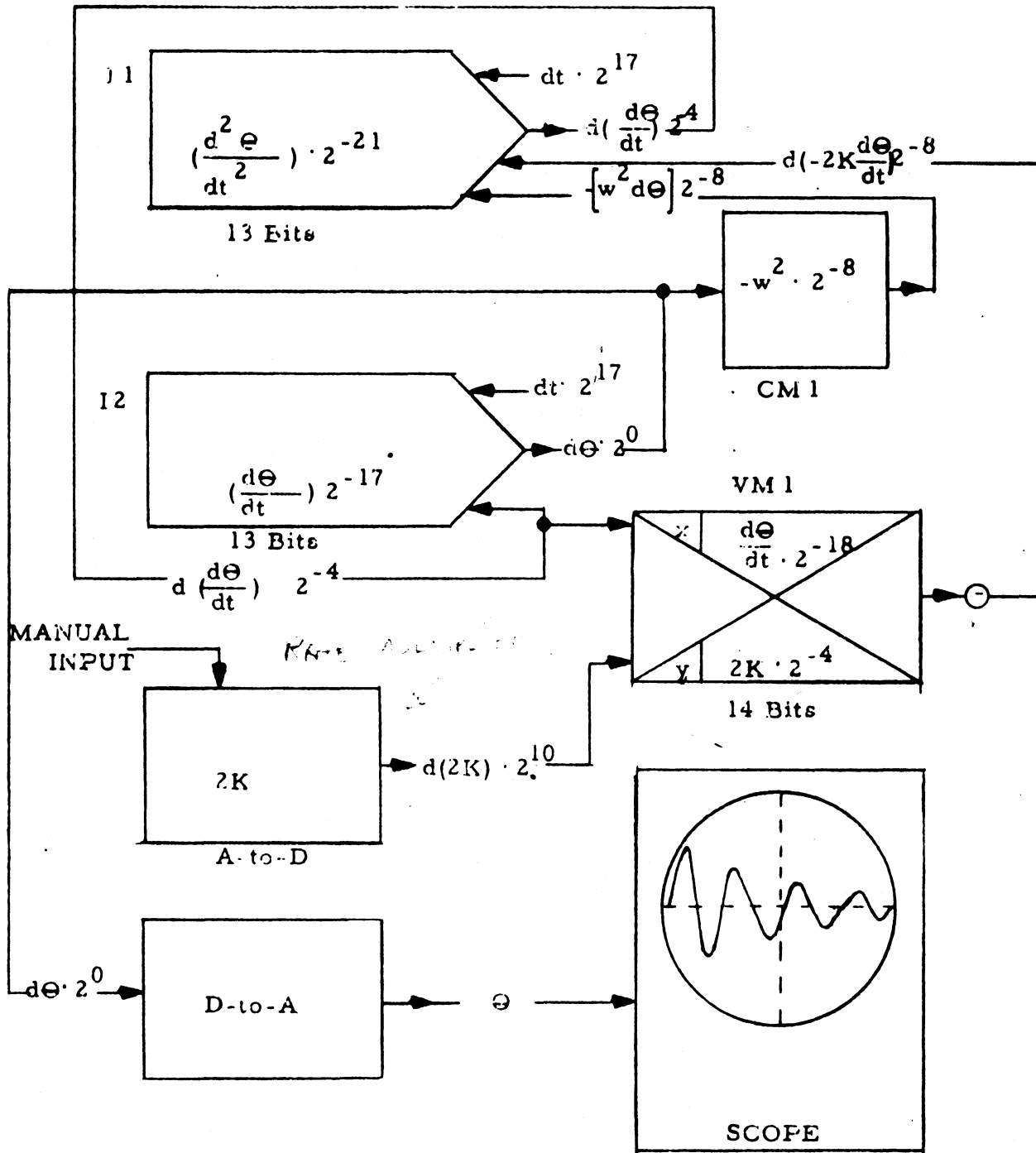


Figure A-2. Map of Damped Sine Wave

- (6) Determine the maximum value of $\frac{d^2 \Theta}{dt^2}$ as follows:

Since,
$$\frac{d^2 \Theta}{dt^2} \cong -Aw^2 \sin wt$$

then
$$\left[\frac{d^2 \Theta}{dt^2} \right]_{\max} \cong \left| -Aw^2 \right| \cong 2^{13} (2^4)^2 \cong 2^{21}.$$

- (7) Determine the scale factors of Integrator 1 as follows:

$$S_y = -21, S_{dx} = +17$$

thus,
$$S_z = S_y + S_{dx} = -21 + 17 = -4$$

- (8) The output of Integrator 1 is the secondary input of Integrator 2; therefore, for Integrator 2

$$S_{dy} = -4$$

- (9) For the Variable Multiplier, scaling $|X| < \frac{1}{2}$ and $|Y| < \frac{1}{2}$ assures the prevention of overflows (as indicated in paragraph 2.3.3).

Since,
$$\left[\frac{d\Theta}{dt} \right]_{\max} \cong 2^{17}$$

$$S_x = -18 \text{ for the Variable Multiplier.}$$

Therefore, VM Register Length = $-4 - (-18) = 14$ Bits.

- (10) For the Constant Multiplier, the primary input is the output of Integrator 2. Thus, S_{dx} is 0 for the Constant Multiplier.

Since,
$$S_z = S_k + S_{dx}$$

Appendix

and $w = 2^4$

and $w^2 = 2^8$

which requires that

$$S_k \leq -8$$

$$S_z = -8 + 0 = -8$$

(11) For Variable Multiplier

$$S_z = S_z \text{ of CM}$$

$$S_z = S_x + S_{dy}$$

$$S_{dy} = -8 - (-18) = 10$$

$$S_y = S_z - S_{dx} = -8 - (-4)$$

$$S_y = -4$$

This limits $2K$ to 2^3 .

(12) Length of Registers:

I1 Register Length = $S_{dy} - S_y = (-8) - (-21) = 13$ Bits

I2 Register Length = $S_{dy} - S_y = (-4) - (-17) = 13$ Bits

VM Register Length = $S_{dy} - S_y = S_{dx} - S_x = 14$ Bits

CM Register must be long enough to carry all bits of w^2 .

Initial Conditions:

- (1) Let computation start at $t = 0$. Hence the initial conditions are derived as follows:

Choose $K = 0$
 and $\Theta = A \sin \omega t = 0$

$$\frac{d\Theta}{dt} = A \omega \cos \omega t$$

Choose $\frac{d\Theta}{dt} = 2^{15}$

Scaling is chosen to vary parameters.

$$\frac{d^2\Theta}{dt^2} = -A \omega^2 \sin \omega t = 0$$

Thus, $\frac{d^2\Theta}{dt^2} = 0$
 $\omega = 2^4$

- (2) The value of $2K$ can be varied from $2K = 0$ to $2K = \pm 2^3$ to obtain a solution to the problem for each value. Other solutions can also be obtained by using different values of ω^2 .

Fill Process:

- (1) The octal code for filling the registers is given in Table A-2.

* NOTE

The register length here is 13 computing bits; thus, the binary value of the least significant bit is found as follows:

$$\text{Value of the least significant bit} = \frac{1}{2^S dy}$$

Hence, $\text{L. S. B.} = \frac{1}{2^{-4}} = 2^4$

Appendix

Thus, the register appears as follows:

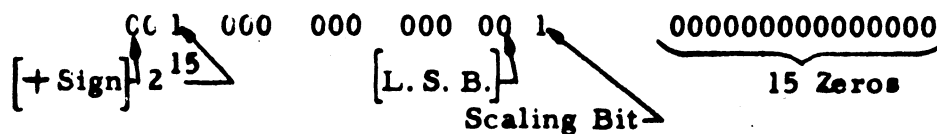


Table A-2. Octal Code for Damped Sine Wave

Module	Register	Number of Binary Computing Bits	Initial Value (Binary)	Octal Code
I1	Y	13	0	0 ₄ 10 ₅
* I2	Y	13	2 ⁻²	1 0 ₃ 1 0 ₅
VM1	X	14	2 ⁻³	0 4 0 ₃ 4 0 ₄
VM1	Y	14	0	0 ₅ 4 0 ₄
CM1	** Y	3	-2 ⁰	4 2 0 ₈

** Register holds 2-complement of $-w^2$.

Patchboard Connections:

Using the map, program the patchboard (for a more detailed discussion of programming the patchboard, see Section IV).

Output:

In this case the output is viewed on a scope.

Example Problem III. Cornu Spiral

Problem: Generate a cornu spiral as defined by

$$X(t) = \int_0^t \cos t^2 dt$$

$$Y(t) = \int_0^t \sin t^2 dt$$

Solution: Write equations in differential form; i. e.,

$$d[X(t)] = \cos t^2 dt$$

$$d[Y(t)] = \sin t^2 dt$$

Map: The map is shown in Figure A-3.

Initial Conditions:

$$\begin{aligned} \text{Start at } t &= 0 \\ \cos t^2 &= 1 \quad X = 0 \\ \sin t^2 &= 0 \quad Y = 0 \end{aligned}$$

Thus, the initial values are set for all modules.

- Scaling:**
- (1) S_y of each Integrator is -1, because the maximum value of their integrands is 1; due to symmetry, the Integrator would overflow if 0 were used because the register's range is $-1 \leq Y < 1$.
 - (2) This problem will produce an overflow from the Variable Multiplier when its register length is exceeded. Hence, the number of turns in the end of the spiral will be determined by the length of the Variable Multiplier registers.

Arbitrarily set the length of the Variable Multiplier to 18 bits. Set output scale factor of Integrator 1 at 2^{+13} in order to use the full range of the D-to-A. Hence, primary scale factor is 2^{+14}

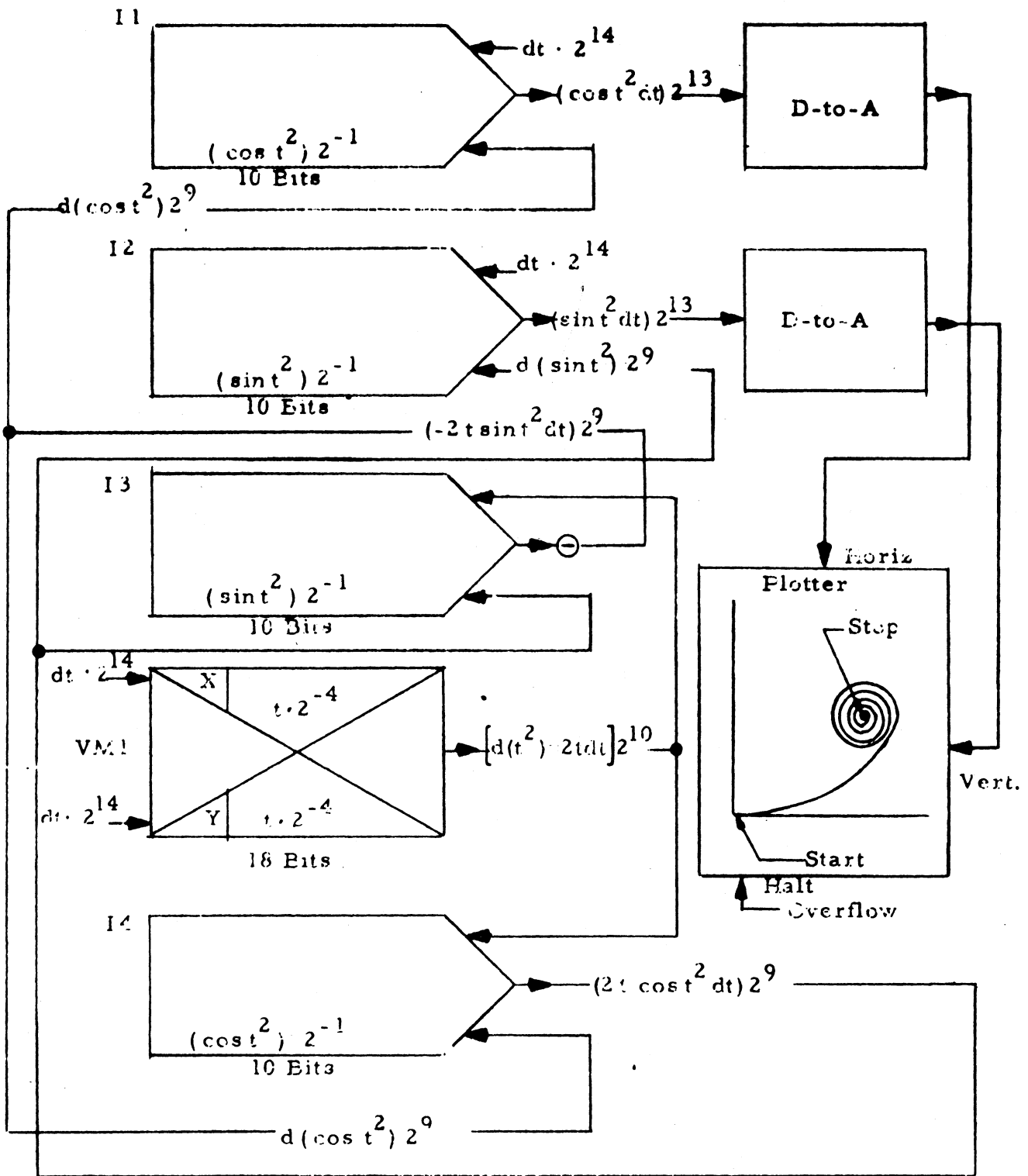


Figure A-3. Map of Cornu Spiral

Time varies from $t = 0$ to $t = 2^3$.

The scale factors for the Variable Multiplier are thereby fixed at

$$S_{dx} = S_{dy} = +14$$

$$S_x = S_y = -4$$

However, in this case, S_x and S_y are the values that cause the integrands of their respective registers to be 1 just before the registers overflow.

$$S_z = 10$$

(3) Scale factors for Integrator 1 are found to be

$$S_{dx} = +14$$

$$S_{dy} = S_z \text{ of Integrator 3} = 10 + (-1) = 9$$

$$S_y = -1$$

$$S_z = 14 + (-1) = 13$$

Register length of Integrator 1 = $+9 - (-1) = 10$ bits

(4) Scale factors of Integrator 2 are

$$S_{dx} = 14$$

$$S_z = 14 - 1 = 13$$

$$S_{dy} = S_z \text{ of Integrator 4} = 10 + (-1) = 9$$

Register length of Integrator 2 = $9 - (-1) = 10$

(5) Scale factors of Integrator 3 are

$$S_{dx} = 10$$

$$S_{dy} = 9$$

Appendix

$$S_y = -1$$

$$S_z = 9$$

Register length of Integrator 3 = 10 bits

(6) Scale factors and register length for Integrator 4 are the same as for Integrator 3.

Patchboard Connections:

Using the map, program the patchboard (see Section IV).

Fill Process:

The initial values and scaling bits are filled as described in Section IV. The octal code for the fill process is given in Table A-3.

Table A-3. Octal Code for Cornu Spiral

Module	Register	Number of Binary Computing Bits	Initial Value (Decimal)	Octal Code
I1	Y	10	1/2	2 0 ₂ 1 0 ₆
I2	Y	10	0	0 ₃ 1 0 ₆
I3	Y	10	0	0 ₃ 1 0 ₆
I4	Y	10	1/2	2 0 ₂ 1 0 ₆
VM1	X	18	0	0 ₆ 2 0 ₃
VM1	Y	18	0	0 ₆ 2 0 ₃

Output: In this problem, an analog plotter is used to plot the curve. The overflow caused by VM1 is used to halt operation of the plotter.

NOTE

A Servo or an Integrator can be used in the problem to cause the time to vary from $-t$ to $+t$, and then from $+t$ to $-t$; thus, the curve retraces itself. In this mode repeatability can be observed. This mode of operation is excellent for observing the output with a scope.

Example Problem IV. Calculation of Pi.

Problem: Calculate π .

Solution: If a sine function is started at $x = 0$ and x is made to vary up to π , the value of π can be obtained from a register that sums x .

Map: The map for this problem is shown in Figure A-4. There will be an output of the Servo for each dt input until its Y-register reaches a value of 0; at which time there will be no output. Thus, $dt = dx$ until the Y-register of the Servo reaches zero. (See paragraph 2.3.4 for an explanation of the Servo's theory of operation.)

One increment is put into the Servo's Y-register to allow computation to start (see paragraph 2.3.4). The secondary input to the Servo is $d(\sin x)$; hence, when x reaches π the value of the Servo Y-register is only one increment away from the value 0; hence, the summed outputs of the Servo should be π .

Appendix

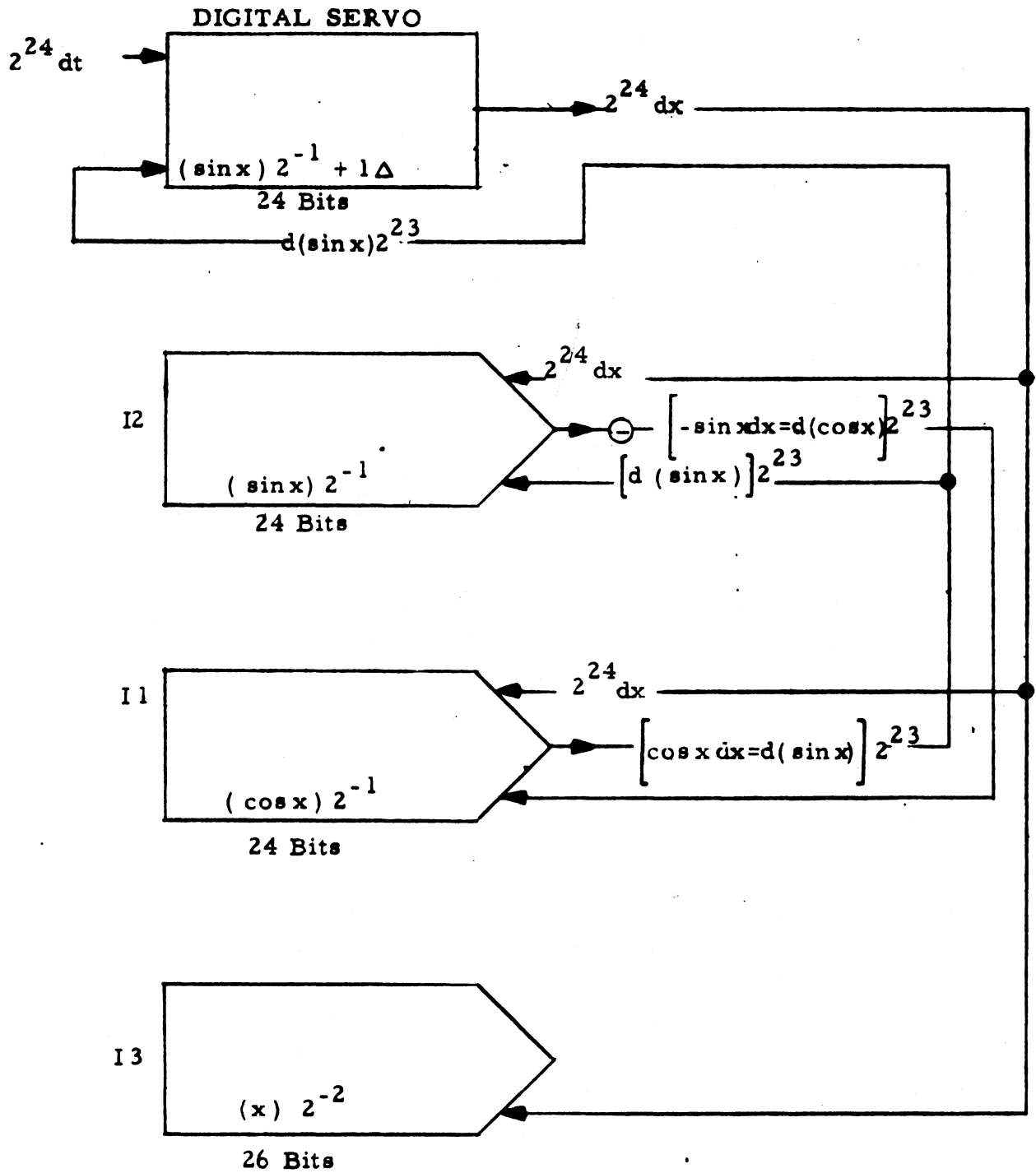


Figure A-4. Map for Calculation of π .

Output: In this case the results may be viewed in Integrator 3 by means of a readout register (see Section IV for operation of readout register).

Scaling: Arbitrarily select 2^{24} for the value for dt . The integrand scale factor of the Servo may be represented as 2^{-1} . Thus, $dx = 2^{24}$ and scaling requires each computing register to have a length of 24 bits. The register length of Integrator 3 must be 26 bits long, since x scale factor is 2^{-2} , because the maximum value is π .

Patchboard Connections:

Using the map, connect the patchboard as described in Section IV.

Initial Conditions:

The initial value of x is zero.

Hence, initially

$$\sin x = 0$$

$$\cos x = 1$$

Fill Process:

The initial values and scaling bits are filled as described in Section IV. The octal code for the fill process is given in Table A-4.

Results: The value of π in the octal code is $31_2 0375_2 242$. The last "one" in the register is the scaling bit; the readout register's value is the exact value to at least 25 binary places.

Table A-4. Octal Code for Calculation of π .

Module	Register	Number of Binary Computing Bits	Initial Value (Decimal)	Octal Code
I1	Y	24	1/2	20 ₇ 20
I2	Y	24	0	0 ₈ 20
S	Y	24	1 Increment	0 ₈ 60
I3	Y	26	0	0 ₉ 4

Example Problem V. Bessel Function

Problem: Generate the Bessel Function as described by

$$\frac{d^2 y}{dx^2} = -\frac{1}{x} \frac{dy}{dx} - y$$

Maximum value of $y = 1$ at $x = 0$

and $0 \leq x < 4$

$$\frac{dy}{dx} < 1$$

Solution: The result of multiplying the equation by dx is

$$d\left(\frac{dy}{dx}\right) = -\frac{1}{x} dy - y dx$$

Map: The map is shown in Figure A-5. The Servo is used because when $x = 0$, $\frac{1}{x}$ becomes infinite.

Scaling:

- (1) Maximum value of $x = 4$. Hence, the value selected as a scaling power for the Integrator 3 integrand is -2 .
- (2) Select $dx \cdot 2^{+11}$, giving a register length of 13 bits for Integrator 3.
- (3) The Servo is initially set to zero; the first input increment, dx to Integrator 2 causes a ydx output increment. After several iterations a dy increment from Integrator 1 is fed into the Y-register of the Servo, resulting in a -1 increment being in the Servo Y-register. However, since the Servo's Y-register contains this increment, there will be an output, du_1 , from the Servo due to the dy . Actually $du = \frac{1}{x} dy$. Integrator 3 now contains a small positive number due to the dx 's and primary input $d(-u_1)$ causes the positive output of Integrator 3 eventually to emit $-xdu$, which is equal to a feedback dy . The feedback $-xdu$ is a positive increment which is used as a secondary input to the Servo which causes the value of the Y-register to become 0 before the second dy input occurs.

The scale power for the output of the Servo is 10. Since both secondary inputs must have identical scale factors,

$$S_y = 8 \text{ for Servo.}$$

The Constant Multiplier is used to make the secondary scale factor compatible for the Servo.

- (4) The register lengths are found to be

I 1	Register Length = $10 - 0 = 10$ Bits
I 2	Register Length = $11 - (-1) = 12$ Bits
I 3	Register Length = $11 - (-2) = 13$ Bits
Servo	Register Length taken as 13 Bits.

Appendix

Patchboard Connections:

Use the map to connect the TRICE patchboard as described in Section IV.

Fill Process:

The octal code for the fill process is given in Table A-5.

Output:

In this case a plotter is used. A table may be taken from the plotter.

Table A-5. Octal Code for Bessel Function

Module	Register	Register Length in Binary Computing Bits	Initial Value	Octal Code
I1	Y	10	0	$0_3 1 0_6$
I2	Y	12	1/2	$2 0_3 2 0_5$
I3	Y	13	0	$0_4 1 0_5$
Servo	Y	13	0	$0_4 1 0_5$

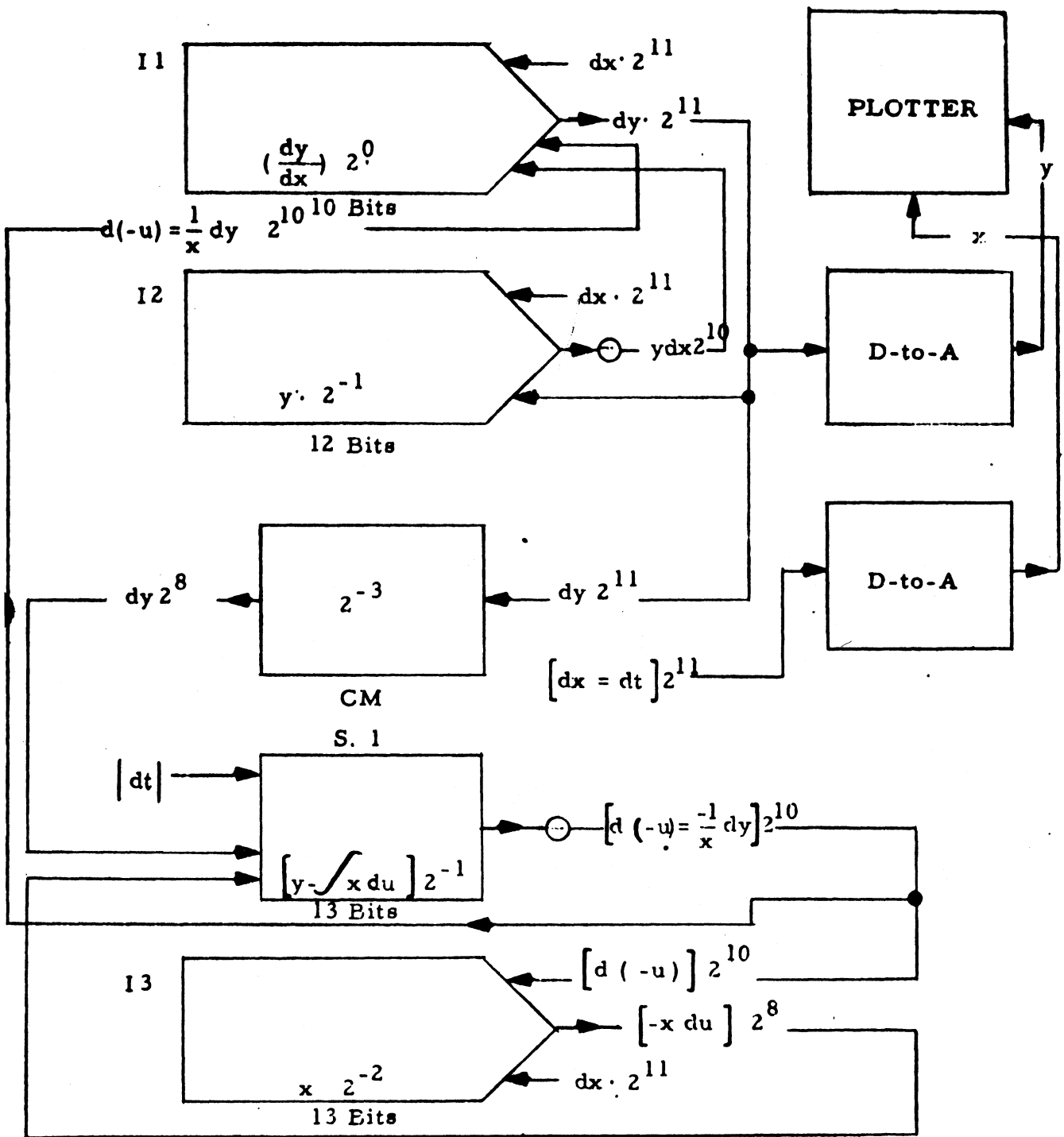


Figure A-5. Map of Bessel Function

* CM shouldn't have to have a 1 word delay/note for $CM = -1$ then

?? P 3.7.2 & p. 336 item (b) is no attenuation

Is there a 2 step algorithm to avoid delays?

Time scaling & delays to be used when dx and z paths are cascaded.

Decimal readout? how? "B/D, D/B Converter scales"

Accessories: Paper tape input
 " " output
 B/D, D/B Converter scale

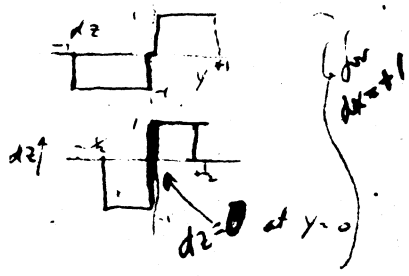
Flexowriter for tape properties
 ADC
 DAC
 Function switching logic

overflow light on int. chassis?

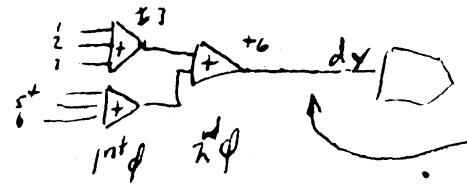
all modules remember sign of last non-zero increment (dx)

In mult: $S = XdY + YdX$ can overflow; $Y + X < 29$ bits?

dig. servo = $dz = (\text{sign } Y) dx$ for $\{ Y \neq 0, \neq 1$
 decision servo $dz = -(\text{sign } Y) dx$ for $-\frac{1}{2} \leq Y < \frac{1}{2}$
 $Y \neq 0$
 Some except for upper limits on Y



DY Summar



permanent connection to Y integrator or Servo
 DY Summar output not available at P.P.
 (select INT or Servo at P.P.)

* Combine 2 R registers w/ one Y register for Ydx and Ydt
 2 R registers w/ one R reg for $dz = (Y_1 + Y_2) dt$ or $= Y_1 dx + Y_2 dt$

in VM $dx + dy$ inputs must have same scale factor

$dx + dy$ requires dig. Servo + DY Summar to drive dx input or DAC

(du) requires a decision servo (not clear what happens when du changes sign)?

Multiplic (VM) required for squaring $du^2 = du du$?? (see 3-18)

DAC required to get t as voltage

DAC has integrating register - one dy input - can't read from Y register

for dt int. max register length = 27 + sign + scale bit = 29
 for dx int. " " " " = 26 + " + " " = 28

all Sdy to a module must be equal !!

delay compensation - CM on every input!