SC-779

[redacted]

GPO PRICE $ ____________

CFSTI PRICE(S) $ ____________

Hard copy (HC) $ 3.00

Microfiche (MF) .75

ff 653 July 65

FACILITY FORM 602

N66 23793 (ACCESSION NUMBER)

(THRU)

77 (PAGES)

/ (CODE)

CR 71185 (NASA CR OR TMX OR AD NUMBER)

08 (CATEGORY)

DIGITAL SYSTEMS LABORATORY

ENGINEERING DIVISION CASE INSTITUTE OF TECHNOLOGY

This Research Was Sponsored by

THE NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

NsG-36

A Microcircuit Integrator
for a
Digital Differential Analyzer

Report No. 1-65-36

by

Thomas W. Burtnett

Harry W. Mergler
Professor of Engineering
Principal Investigator
Ns G 36-60

Digital
Systems
Laboratory

1966

ABSTRACT

2393

A brief summary of the advantages and uses of digital differential analyzers is provided. After reviewing the principles of operation, coding, and scaling of digital integrators, the detailed design of an integrator is discussed. The design includes two full adder/subtractor units, two variable length shift registers to provide internal memory, and utilizes a binary method of coding increments. Implemented with integrated circuits, the integrator is demonstrated with an example problem. The experimental results are tabulated and compared.

ACKNOWLEDGMENTS

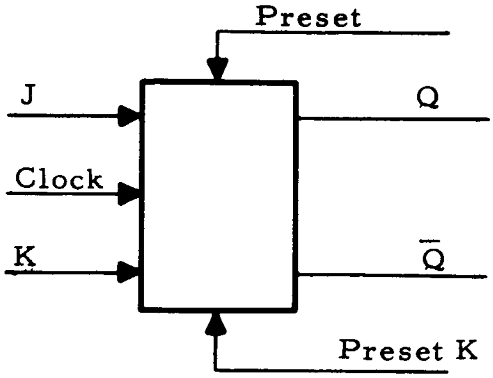
The author wishes to thank Dr. H, W. Mergler for his help in the formulation of this problem and for his guidance at various stages of development. During the implementation of the project, the advice of Dr. Jerome F. Walker, James E. Palmer, and Stuart F. Daniels was helpful and is appreciated.

TABLE OF CONTENTS

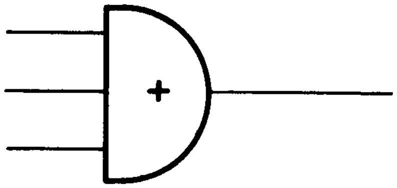
LIST OF FIGURES

LIST OF TABLES

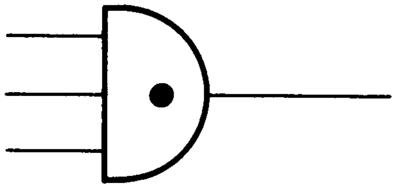
List of Symbols



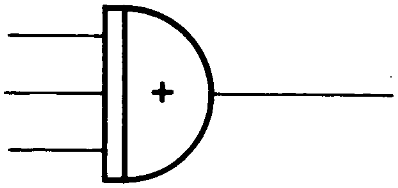
J-K Binary Element



OR Gate



AND Gate



NOR Gate

CHAPTER I
INTRODUCTION

Features of Digital Differential Analyzers

The need for a simple, compact digital computer suitable for solving differential equations led to the development of the Digital Differential Analyzer (DDA). The DDA may be put into the class of analog computers since it is a device consisting of a number of separate functional units, each unit containing all elements required for the particular function. It is similar to analog machines in that the quantity of apparatus needed depends upon the complexity of the problem. The functional units are interconnected directly in accordance with the structure of the problem being solved. However, within the functional unit, the computation is digital.

The basic functional unit in a DDA is the integrator. If only one integrator may operate at one time due to there being only one arithmetic unit, then it is a serial DDA. In a parallel DDA all integrators may operate simultaneously. The parallel DDA is more complex in construction but is naturally faster than the serial DDA. Accuracy of the DDA may be increased at the expense of running time; hence, the potential accuracy greatly exceeds that of other types of analog computers.

An important feature is that the independent variable need not be time. Also it is not necessary to manipulate problems in order to avoid integration with respect to a variable other than the independent variable. These features are possible since all inputs to the integrators are pulse trains which need not be dependent upon time.

The outstanding features of digital integrating devices have been itemized by F. V. Mayorov as follows:

"a) In integration, the computer operates with increments of input quantities and not with the quantities themselve as in the arithmetic machine. This permits considerable increase in computing speed and in switching (or interconnection) of the integrators.

"b) By using integration as a basic operation, operations of multiplication, division, extraction of a root, sine of a number, and logarithm calculation take a time equivalent to that for two or three addition operations. Insertion of various experimental and tabular values into a digital integrating computer can be simpler than in an arithmetic machine . . .

"c) The memory access time in a digital integrating computer is practically zero. The data reaches the machine in a continuous flow, one code after the other, without requiring an access from the memory . . .

"d) Solution of a relatively complex problem in a DDA does not require an internal memory of large capacity . . .

"e) Depending on the required accuracy, the digital integrating computer possesses flexibility in computing speed. A given problem may first be solved with a low accuracy but at a high speed. The same problem is then solved with a higher accuracy but longer solution time."

If the reader is familiar with the mechanical Bush analyzer, G. F. Forbes states that "any problem capable of being solved on the Bush analyzer may be solved more easily on the DDA. Problems too complex for the Bush analyzer can be done on the DDA."

Uses of DDA's

Although the DDA may be used for scientific computation or simulation, it is primarily used for automatic control. The following uses for DDA's were noted by A. V. Shileiko in his book, *Digital Differential Analysers*, originally published in Moscow, 1961:

"Litton Industries have created a digital differential analyzer, now being tested by naval specialists with a view to using it in aircraft.

"A DDA flight-tested in the X-10 rocket occupied .081 m^3, weighed 66 kg, and consumed 100 W, precisely 1/12 of the consumption of its valve-based predecessor. The device performed 93 integrations and enabled both trigonometric and differential equations to be solved.

"Recent advances have led to newer types of digital integrating computers . . . developed by General Electric . . . These compact computers, which are designed for missiles, fighters, and bombers, are also used for control of Polaris missiles in submarines and for inertial navigation of the missiles themselves . . . "

This report first explains the principles of operation of digital integrators used in DDA's. Then the design and implementation of such an integrator is described.

CHAPTER II

PRINCIPLES OF OPERATION

Data Transfer within the Integrator

The integration process $z = \int ydx$ is defined

$$\lim_{\Delta x \to 0} \sum_{j=1}^{n} y_j \Delta x$$

which can be approximated by

$$\sum_{j=1}^{n} y_j \Delta x$$

shown in Figure 1.

This is the familiar concept of summing rectangular strips to obtain the area under a curve, i.e., the integral. If Δx were considered to be a unit distance, the expression reduces to $z = \Sigma y_i$. The digital integrator discussed in this thesis is based on this expression.

Figure 2 shows how this expression could be implemented. For each integration step the change in ordinate values, $\Delta y = y_j - y_{j-1}$, is accumulated in the D register. This is added to y_{j-1} by Σ_1 to produce y_j in the Y register. The Z register

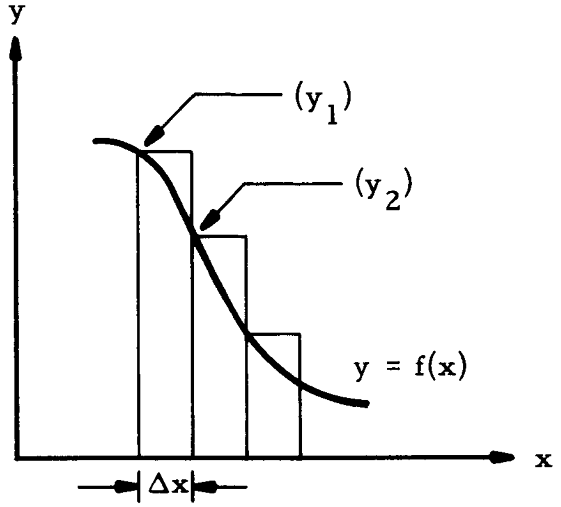


Figure 1 Integration Approximation

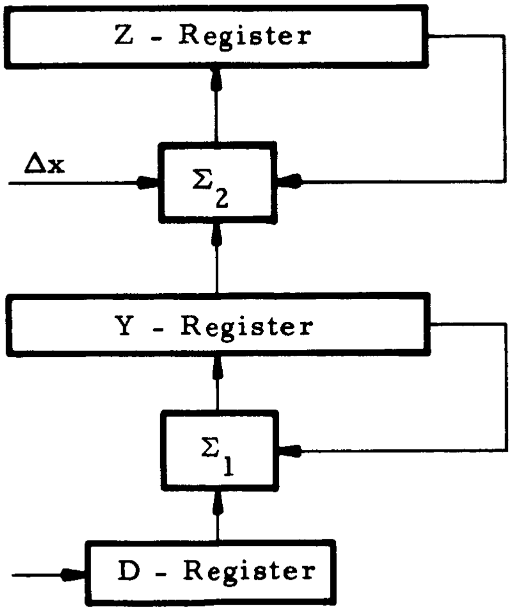


Figure 2 Integrator Block Diagram

already contains

$$\sum_{q=1}^{j-1} y_q \quad .$$

Upon indication of another Δx, the new value of y, y_j, is added to the contents of z by Σ_2 to produce

$$\sum_{q=1}^{j} y_q \quad .$$

Note that the Z register would have to be quite large to contain

$$\sum_{q=1}^{j} y_q$$

for an arbitrary j. Therefore, instead of storing the entire value of z, the Z register is limited to the capacity of the Y register. The overflow from the Z register is interpreted to be the change in the integral, Δz. In Figure 3 the Z register has been renamed the S register since it only contains the less significant part of the integral z.

The operation shown in Figure 3 may be expressed as follows:

$$y = y_0 + \int_{t_0}^{t} \frac{dy}{dt}\, dt$$

$dz = p^{-N} y\, dx$ where N is the number of bits in the Y register.

$$z = z_0 + p^{-N} \int_{x_0}^{x} ydx$$

Thus upon summing the Δz output pulses, a value proportional to $\int ydx$ is obtained.

Coding

There are two popular methods of coding the increments Δx, Δy, Δz. In the ternary method a + 1 corresponds to a positive increment, -1 to a negative increment, and 0 to zero increment. This method requires a + 1 and a - 1 line for each input and output of the integrator. If pulses were used to transmit the increments, a pulse on either line could indicate the respective increment, and the absence of a pulse on both lines could indicate zero increment.

The second method is the binary system of coding incre-

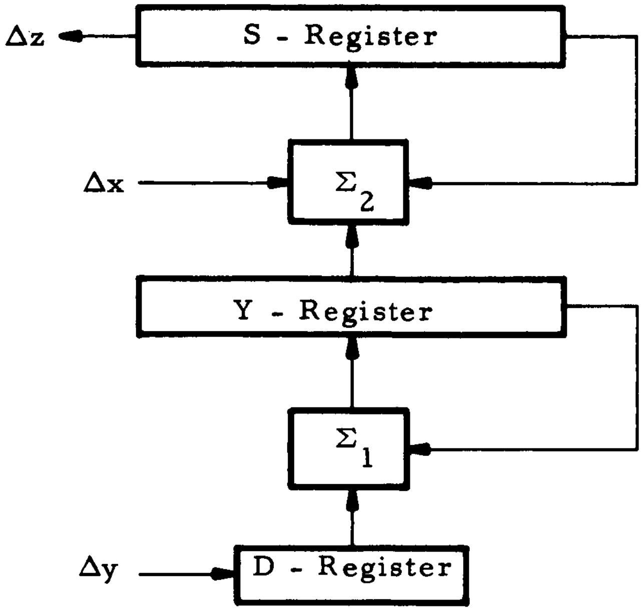


Figure 3 Integrator Block Diagram

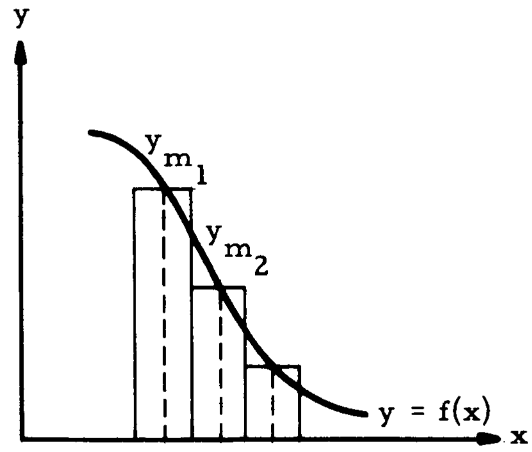


Figure 4 Trapezoidal Approximation

ments. In this system a pulse could be used to indicate a positive increment and the absence of a pulse to represent a negative increment. A zero increment would be transmitted by alternating + 1 and - 1. The circuitry involved in implementing this method is less than for the ternary method. On the other hand, the ternary method is more accurate.

At this point it is appropriate to digress a moment to mention integration using the trapezoidal rule. In this method rather than summing $y_1, y_2, \ldots y_n$, the mean ordinates, $y_m = y_{n-1} + 1/2 \Delta y$, are summed. See Figure 4.

This results in a more accurate approximation of the integral. However, Mayorov points out that in the binary system of coding increments the integration error may amount to a least significant bit in the Y or S register. This error is considerably larger than the error resulting from approximating the integral by rectangles. Hence the trapezoidal method is justified only if the ternary method of coding increments is used.

Returning to the subject of coding, the numbers contained in the Z, Y, and D registers are usually represented in some form of the binary code. One basic form with a few variations will be considered here.

Let the Y register contain numbers of the form $y_{n+2}\ y_{n+1}\ y_n\ y_{n-1} \cdots y_2\ y_1$, where y_i are the binary digits 0 or 1. Bits y_n through y_1 are the information bits, y_{n+1} denotes sign, and y_{n+2} is the overflow bit.

Let $y_{n+1} = 1$ denote a positive number and let $y_{n+1} = 0$ denote a negative number. When $y_{n+1} = 1$ let the information bits contain the true binary form of the number to be represented. For $y_{n+1} = 0$ let the information bits contain the two's complement of the absolute value.

The two's complement, denoted $\tau(N)$, is defined $2^n - N$ where n is the number of bits in N. The y_{n+2} bit will be discussed later.

Let the contents of the S register be similarly defined and be of the form $s_{n+2}\ s_{n+1} \cdots s_2\ s_1$. Examples follow for five information bits where the underlined bit is the sign bit:

$Y = +12$	Y register = $\underline{1}\ 0\ 1\ 1\ 0\ 0$
$S = -7$	S register = $\underline{0}\ 1\ 1\ 0\ 0\ 1$

There are at least two acceptable ways to code the D register. In one the meaning of the sign bit is reversed but the same rules apply to the information bits.

$$\Sigma \Delta y \geq 0 \qquad D = \Sigma \Delta y \text{ (base 2)}$$

$$\Sigma \Delta y < 0 \qquad D = 2^n + \tau(\Sigma \Delta y)$$

For example,

$$\Sigma \Delta y = +12 \qquad D = \underline{0}\ 0\ 1\ 1\ 0\ 0$$

$$\Sigma \Delta y = -5 \qquad D = \underline{1}\ 1\ 1\ 0\ 1\ 1$$

With this definition for the D register, an adder for Σ_1 will be sufficient to produce correct results when $\Sigma \Delta y$ is added to Y, regardless of the sign of either Y or $\Sigma \Delta y$. For example,

<u>Example 1</u>

$$Y \geq 0, \quad \Sigma \Delta y \geq 0$$

$$Y = 2^n + Y; \quad D = \Sigma \Delta y$$

$$D + Y = 2^n + y + \Sigma \Delta y \quad (2^n \text{ indicates positive})$$

Example 2

$$Y < 0, \ \Sigma \Delta y \geq 0, \ |Y| < |\Sigma \Delta y|$$

$$Y = \tau(Y); \ D = \Sigma \Delta y$$

$$\begin{aligned} D + Y &= \tau(Y) + \Sigma \Delta y \\ &= 2^n - Y + \Sigma \Delta y \\ &= 2^n + (\Sigma \Delta y - Y) \end{aligned}$$

$$\Sigma \Delta y - Y > 0 \text{ since } |Y| < |\Sigma \Delta y|$$

$$D + Y > 2^n \qquad (2^n \text{ indicates positive})$$

Example 3

$$\begin{array}{lcl} Y_j & = & +4, \ \Sigma \Delta y = -7 \\ Y_j & = & \underline{1}\,0\,0\,1\,0\,0 \\ D & = & \underline{\underline{1}\,1\,1\,0\,0\,1} \\ Y_{j+1} & & \underline{0}\,1\,1\,1\,0\,1 \quad = \ -3 \end{array}$$

Another method of coding D is to let the information bits contain the true binary form whether the number be + or -. Now Σ_2 must be an adder/subtractor. When d_{n+1} is 1, Σ_1 adds; when d_{n+1} is 0, Σ_1 subtracts. However, the sign bit of D must not enter into the addition or subtraction.

Example 4

$$\begin{array}{rcl} Y_j & = & -12 \\ \Sigma\,\Delta y & = & +3 \end{array} \qquad \begin{array}{rcll} Y_j & = & \underline{0}\,1\,0\,1\,0\,0 & \\ D & = & \underline{+}\,0\,0\,0\,1\,1 & \\ \hline Y_{j+1} & = & \underline{0}\,1\,0\,1\,1\,1 & = -9 \end{array}$$

Example 5

$$\begin{array}{rcl} Y_j & = & +4 \\ \Sigma\,\Delta y & = & -7 \end{array} \qquad \begin{array}{rcll} Y_j & = & \underline{1}\,0\,0\,1\,0\,0 & \\ D & = & \underline{-}\,0\,0\,1\,1\,1 & \\ \hline & & \underline{0}\,1\,1\,1\,0\,1 & = -3 \end{array}$$

During an integration process the Y register must not overflow. If at the end of a cycle the overflow bit, y_{n+2}, contains a 1, the integration must stop and the problem must be rescaled. The S register is expected to overflow. When $s_{n+2} = 1$ then $\Delta Z = +1$; when $s_{n+2} = 0$, $\Delta Z = -1$. The combining of Y and S for each Δx could be accpmplished with an adder for Σ_2. Several examples follow:

Example 6

Let $S_0 = 0$ and $Y = 0$ (constant)

$$\begin{array}{rcl} S & = & \underline{1}\,0\,0\,0\,0\,0 \\ Y & = & \underline{1}\,0\,0\,0\,0\,0 \\ \hline & & 1\,\underline{0}\,0\,0\,0\,0\,0\,, \qquad \Delta Z = +1 \end{array}$$

The bit to the left of the underlined bit is the overflow bit. Since it is a 1, $\Delta Z = +1$. Continuing with $Y = 0$,

$$\begin{array}{rr} S = & \underline{0}00000 \\ & \underline{1}00000 \\ \hline & 0\underline{1}00000, \end{array} \qquad \Delta Z = -1$$

$$\begin{array}{rr} S = & \underline{1}00000 \\ & \underline{1}00000 \\ \hline & 1\underline{0}00000, \end{array} \qquad \Delta Z = +1.$$

Hence the output alternates + 1, - 1, + 1, - 1 . . . which is a net change of zero for the integral Z.

Example 7

Let $S_0 = 0$ and $Y = Y_{\max}$

$$\begin{array}{rr} S = & \underline{1}00000 \\ Y = & \underline{1}11111 \\ \hline & 1\underline{0}11111, \end{array} \qquad \Delta Z = +1$$

$$\begin{array}{rr} S = & \underline{0}11111 \\ & \underline{1}11111 \\ \hline & 1\underline{0}11110, \end{array} \qquad \Delta Z = +1$$

$$\begin{array}{rr} S = & \underline{0}\,1\,1\,1\,1\,0 \\ & \underline{1}\,1\,1\,1\,1\,1 \\ \hline & 1\,\underline{0}\,1\,1\,1\,0\,1, \end{array} \qquad \Delta Z = +1$$

Thus the maximum Y causes the maximum positive rate of change of Z. $\Delta Z = +1, +1, +1 \ldots$

Example 8

Let $S_0 = 0$ and $Y = 1/2\, Y_{min}$

$$\begin{array}{rcl} Y_{min} & = & \underline{0}\,0\,0\,0\,0 = -16 \\ 1/2\, Y_{min} & = & \underline{0}\,1\,0\,0\,0 = -\ 8 \end{array}$$

$$\begin{array}{rcll} S_0 & = & \underline{1}\,0\,0\,0\,0 & \\ Y & = & \underline{0}\,1\,0\,0\,0 & \\ \hline & & \underline{1}\,1\,0\,0\,0, & \Delta Z = -1 \\ & & \underline{0}\,1\,0\,0\,0 & \\ \hline & & \underline{0}\,0\,0\,0\,0, & \Delta Z = +1 \\ & & \underline{0}\,1\,0\,0\,0 & \\ \hline & & \underline{0}\,1\,0\,0\,0, & \Delta Z = -1 \\ & & \underline{0}\,1\,0\,0\,0 & \\ \hline S_4 = S_0 & = & \underline{1}\,0\,0\,0\,0, & \Delta Z = -1 \end{array}$$

This is half of the maximum negative rate of change. All three of the above examples have been for Δx always being +1. When $\Delta x = -1$, the operation $S - y$ is to occur.

Mayorov suggests that when $\Delta x = -1$, the one's complement of Y should be added to S. However, this writer feels that this introduced unnecessary error.

To illustrate,

<u>Example 9</u>

$$\begin{array}{lll@{\qquad\qquad}lll}
\Delta x & = & -1 \text{ (constant)} & Y & = & \underline{1}\,0\,0\,0 \\
S_0 & = & \underline{1}\,0\,0\,0 & \bar{Y} & = & \underline{0}\,1\,1\,1
\end{array}$$

$$\begin{array}{llll@{\qquad}ll}
S_0 = & \underline{1}\,0\,0\,0 & & & \underline{1}\,0\,1\,1 & \\
 & \underline{0}\,1\,1\,1 & & & \underline{0}\,1\,1\,1 & \\
\cline{2-2}\cline{5-5}
 & \underline{1}\,1\,1\,1 & \Delta z = 0 & & \underline{0}\,0\,1\,0 & \Delta z = 1 \\
 & \underline{0}\,1\,1\,1 & & & \underline{0}\,1\,1\,1 & \\
\cline{2-2}\cline{5-5}
 & \underline{0}\,1\,1\,0 & \Delta z = 1 & & \underline{1}\,0\,0\,1 & \Delta z = 0 \\
 & \underline{0}\,1\,1\,1 & & & \underline{0}\,1\,1\,1 & \\
\cline{2-2}\cline{5-5}
 & \underline{1}\,1\,0\,1 & \Delta z = 0 & & \underline{0}\,0\,0\,0 & \Delta z = 1 \\
 & \underline{0}\,1\,1\,1 & & & \underline{0}\,1\,1\,1 & \\
\cline{2-2}\cline{5-5}
 & \underline{0}\,1\,0\,0 & \Delta z = 1 & & \underline{0}\,1\,1\,1 & \Delta z = 0 \\
 & \underline{0}\,1\,1\,1 & & & \underline{0}\,1\,1\,1 & \\
\cline{2-2}\cline{5-5}
 & \underline{1}\,0\,1\,1 & \Delta z = 0 & & \underline{1}\,1\,1\,0 & \Delta z = 0
\end{array}$$

The rate of change of Z should be zero; hence, Δz should alternate between + 1 and - 1. However, after 10 iterations $\Sigma \Delta z = -2$. The reason for this discrepancy is that $\underline{0}\ 1\ 1\ 1$ is by previous definition equal to - 1. This only points up the fact that the one's complement differs from the two's complement by one in the least significant bit. For N information bits, the error produced for $\Delta x = -1$ (constant) is given by,

$$\epsilon = 2^{-N} M \text{ where M is the number of iterations.}$$

If the two's complement of Y were added to S when $\Delta x = -1$, the result would be exact. Note that $\tau(\underline{1}\ 0\ 0\ 0) = \underline{1}\ 0\ 0\ 0$. It has already been shown that successive additions of $\underline{1}\ 0\ 0\ 0$ produce alternating + 1 and - 1 for Δz.

Another approach which will insure an exact result when $\Delta x = -1$ is to actually perform the subtraction S - Y. If an adder/subtractor were used for Σ_2, then there would never be any need to take the one's complement or two's complement of Y. It has been defined that $\Delta z = +1$ when there is an overflow. But if a subtraction were to occur when $\Delta x = -1$, what is the meaning of an "overflow" in the S register? In subtraction, rather than an outgoing carry, there is sometimes an outgoing borrow. In the example below, the outgoing borrow will be noted.

Example 10

Let $S_0 = 0$, $Y = 8$, $\Delta x = -1, -1, -1, -1$

$$\begin{array}{rcll}
S_0 & = & \underline{1}\;0\;0\;0\;0 & \\
Y & = & \underline{\underline{1}}\;1\;0\;0\;0 & \\
\cline{3-3}
S_1 = (S_0 - Y) & = & \underline{1}\;1\;0\;0\;0 & B_0 = 1 \\
 & - & \underline{\underline{1}}\;1\;0\;0\;0 & \\
\cline{3-3}
S_2 & = & \underline{0}\;0\;0\;0\;0 & B_0 = 0 \\
 & - & \underline{\underline{1}}\;1\;0\;0\;0 & \\
\cline{3-3}
S_3 & = & \underline{0}\;1\;0\;0\;0 & B_0 = 1 \\
 & - & \underline{\underline{1}}\;1\;0\;0\;0 & \\
\cline{3-3}
S_4 = S_0 & = & \underline{1}\;0\;0\;0\;0 & B_0 = 1
\end{array}$$

Recall example 8 where $S_0 = 0$, $Y = -8$, and $\Delta x = +1, +1, +1, +1$. Note that examples 8 and 10 are actually the same integration. In example 8, the output sequence for Δz was $-1, +1, -1, -1$. The similarity to the sequence of outgoing borrows in example 10 is obvious. It seems that when subtracting, if $B_0 = 1$ then $\Delta z = -1$ and if $B_0 = 0$ then $\Delta z = +1$. This deduction is true and is proved below:

To Prove : The outgoing borrow, B_0, produced by subtracting N from M is the complement of the outgoing carry, C_0, produced by adding the two's complement of N to M.

Proof: Let M and N be binary numbers of equal length.

Let n be the number of bits in M and N, including sign bit.

The two's complement of N, $\tau(N)$, equals $2^n - N$, where n is the number of bits in N.

There are only two possible cases for M - N; either $N > M$ or $M \geq N$.

Case 1 $N > M$

The subtraction M - N will result in an outgoing borrow.

$B_0 = 1$

$$M + \tau(N) = M + 2^n - N$$

$$= 2^n + (M - N)$$

$$= 2^n - (N - M)$$

$$N > M \rightarrow (N - M) > 0$$

Hence, $2^n - (N - M) < 2^n$

All numbers less than 2^n can be contained in n bits. An overflow would be indicated by a 1 in the n + 1 bit. Hence there is no overflow. $\underline{C_0 = 0}$

Case 2 $M \geq N$

The subtraction M - N will not produce an outgoing borrow. $\underline{B_0 = 0}$

$$m + \tau(N) = M + 2^n - N$$

$$= 2^n + (M - N)$$

$$M \geq N \rightarrow (M - N) \geq 0$$

Hence, $2^n + (M - N) \geq 2^n$

All numbers greater than or equal to 2^n cannot be contained in less than n + 1 bits. Since M has n bits, $M < 2^n$

$$N_{min} = 0$$

Hence, $(M - N) < 2^n$

$$2^n + (M - N) < 2^{n+1}$$

$$([\ 2^n + (M - N)]_{max} = 2^{n+1} - 1)$$

Since $2^{n+1} > 2^n + (M - N) \geq 2^n$ there must be a 1 in the n + 1 bit. Therefore, $\underline{\underline{C_0 = 1}}$. Q. E. D.

Adders

It has been stated earlier that the Y register must not overflow during an integration process. However, if it is permitted to overflow, the integrator can be used as an adder. Figure 5 shows the schematic for an integrator which is wired as an adder.

Initially, the Y register is set to its maximum positive value, + max, and Δx is always + 1. First disregard u, v, and w. The first Δx pulse will produce $\Delta z = + 1$. Since Δz is fed back into the dy input, it will cause the y register to change from + max to the maximum negative value, - max. The next Δx pulse will produce $\Delta z = - 1$, which will return the Y register to + max. Hence, for no inputs u, v, and w, the output alternates between + 1 and - 1. If u, v, and w would produce $\Sigma \Delta y = + 3$, then the next three Δz outputs would be - 1 ———— neglecting future inputs. In this way the

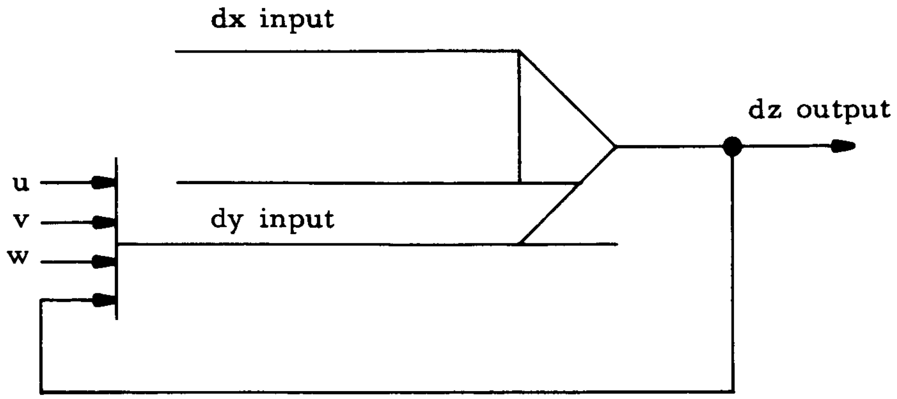


Figure 5 Schematic of an Adder

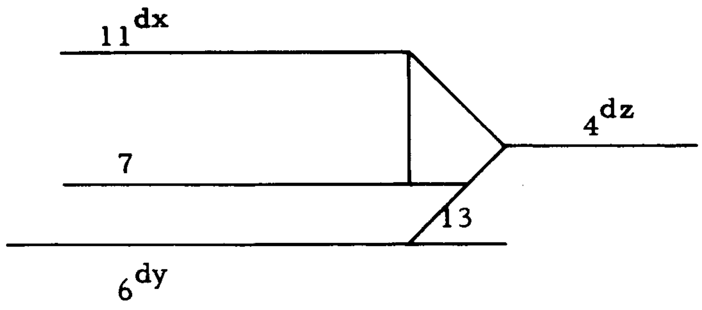


Figure 6 Schematic for Example 11

integrator adds but with sign change and time delay.

$$-\frac{dz}{dt} = \frac{du}{dt} + \frac{dv}{dt} + \frac{dw}{dt} .$$

Integrator Modes

The assumed mode of the integrators in most discussions is the interpolative mode. This is when upon receiving Δx and Δy, the present contents of the Y register are added to S. At the end of the addition cycle,

$$S_{i+1} = S_i + Y_i$$

$$Y_{i+1} = Y_i + \Delta y$$

The extrapolative mode differs in that the new value of y is added to S.

Then, $$S_{i+1} = S_i + Y_{i+1}$$

$$Y_{i+1} = Y_i + \Delta y.$$

The choice of mode of an integrator is dependent upon its position in the schematic of the solution. The interpolative mode gives greater accuracy when the dy inputs are integrator outputs which have occurred in the same or a previous integration period. When the dy inputs are the outputs from an integration occurring at a future time, the mode should be extrapolative. In an actual DDA the mode of each integrator would be set manually.

Scale Factors

G. F. Forbes has proposed a method of scaling to aid in solving problems on a DDA. The following is an example taken from his book, Digital Differential Analyzers:

Example 11

Given: The scale of input dx is 2048 pulses per unit. The scale of input dy is 64 pulses per unit. The maximum absolute value of y is to be less than 88 units throughout the integration.

Determine the scale of the output, dz, and the integrator length, i.e., the number of bits necessary for the Y register so that it does not overflow.

In the schematic representation of the integrator, Figure 6, the scale is indicated by writing the power of two rather than the number itself.

The scale of dy is 6, $(2^6 = 64)$. The number of binary bits necessary to contain 88 units is 7, $(2^7 = 128)$. Hence the scale of y is 7. The input dy is added to the sixth bit to the right of the binary point and there are seven bits to the left of the binary point. Thus the integrand length is 13, $(7 + 6 = 13)$. Usually the integrand length is referred to as the integrator length since both the Y and S registers are the same length.

If y were unity, one unit of dx input (2^{11} pulses) would produce $2^{11}/2^7 = 2^4$ output dz pulses. These pulses must represent one unit of z, thus the scale of dz is 4.

In general, the scale of y plus the scale of dy is equal to the integrator length. The scale of dx minus the scale of y is equal to the scale of dz. When the output of an integrator is used as the input to another or the same integrator, the scales must be the same.

These conclusions may be derived mathematically by considering the basic equation, dz = ydx.

$$dz = p^{-N} ydx \text{ for N bit registers}$$

Inserting scale factors

$$S_z dz = p^{-N} S_y \, y \, S_x dx$$

Since the integrator must produce dz = ydx,

$$S_z \, dz = p^{-N} S_y \, S_x \, dz$$

The scale factors are usually chosen to be whole number powers of the base of the notation in use.

Redefining,

$$S_z = p^{S_z}; \; S_y = p^{S_y}; \; S_x = p^{S_x}$$

$$p^{S_z} \, dz = p^{-N} \, p^{S_y} \, p^{S_x} \, dz$$

$$p^{S_z} = p^{-N} \, p^{S_y} \, p^{S_x}$$

$$S_z = - N + S_y + S_x \qquad (1)$$

$p^{S_y} \; p^m \leq p^N$ where p^m is the maximum value of y.

$$S_y + m \leq N \qquad (2)$$

When $m = N - S_y$

$$S_z = S_x - m \qquad (3)$$

Forbes discusses the generation of trigonometric functions, algebraic functions, inverse trigonometric function, normalization, differentiation, solution of simultaneous equations, complex functions, complex polynomials, and many other uses of a DDA.

For many problems, the following procedure suggested by E. L. Braun may be used:

1. Reduce to a set of differential equations.

2. Isolate the highest derivative of each dependent variable by putting it on the left side of the equation and all other terms on the right.

3. Assume the highest derivative is known, and by integrating it, generate all lower derivatives required in the problem.

4. Combine the variables on the right side of the

equation and use this sum as the source of the assumed highest derivative.

Example 12

$$a\ddot{x} + b\dot{x} + cx = 0$$

$$\ddot{x} = - b/a\ \dot{x} - c/a\ x$$

This is shown in Figure 7.

The schematic for this problem is shown in Figure 8.

Appropriate scaling can be done when the approximate range of each integrand and the desired accuracy are known.

Computer Program

Before beginning a hardware design of a digital integrator, it was felt that a computer simulation would be helpful in understanding and evaluating the simpler, yet confusing, method of binary transfer of increments. The following flow chart was used in writing this program:

IL = integrator length

S_x = scale of x

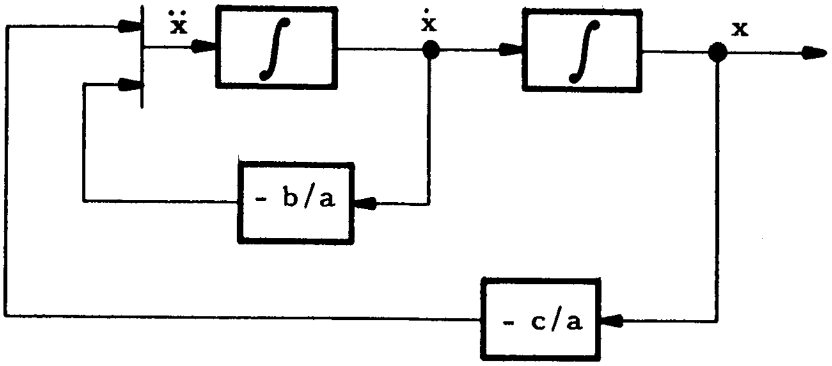


Figure 7 Block Solution for Example 12

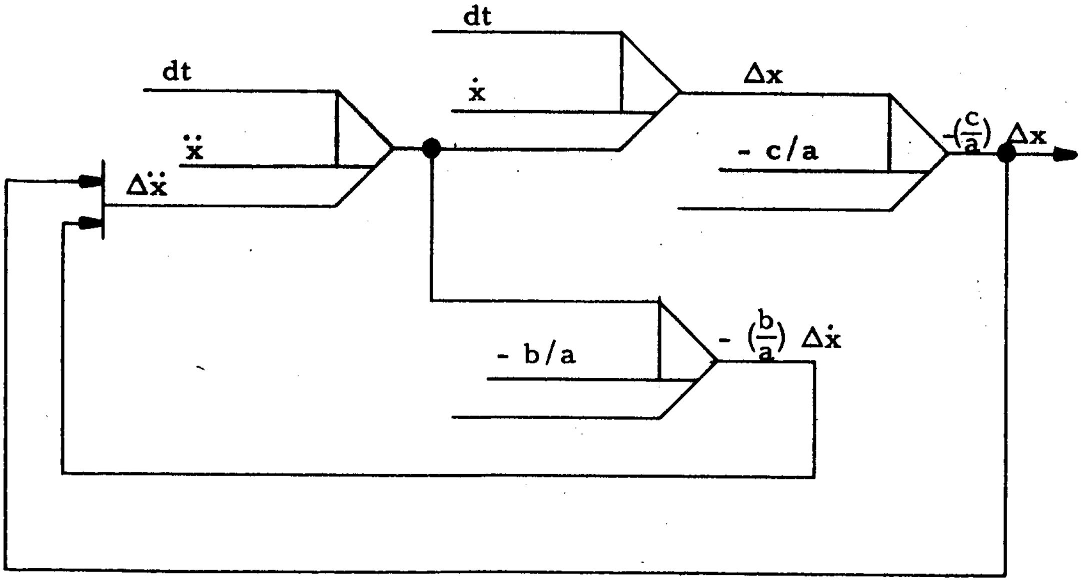
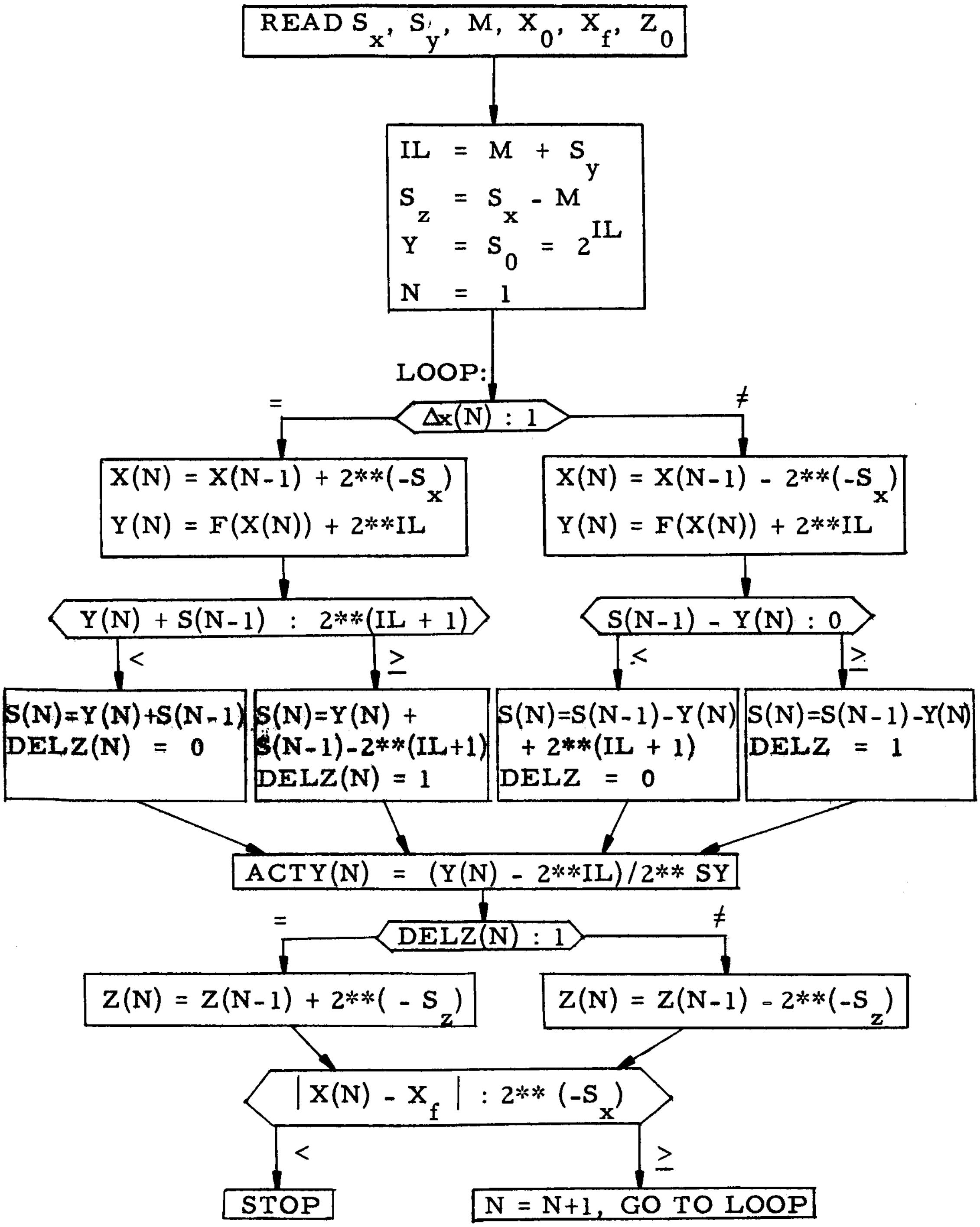


Figure 8 Schematic for Example 12

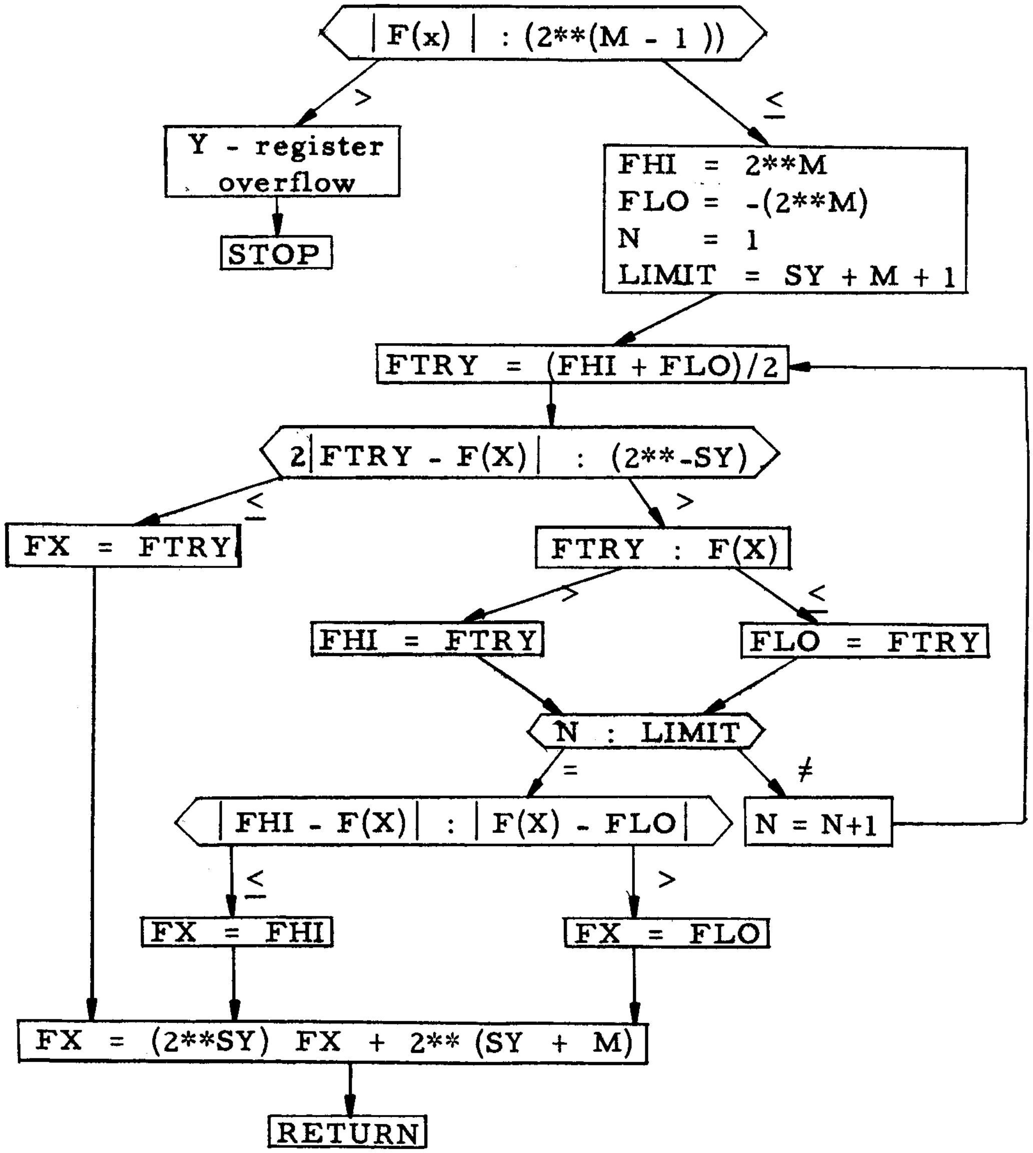
M = maximum value of y

ACTY = actual y value

F(x) = a separate subroutine which provides the value of the dependent variable when given a particular value, x, of the independent variable.


READ S_x, S_y, M, X_0, X_f, Z_0
IL = M + S_y
S_z = S_x - M
Y = S_0 = 2^{IL}
N = 1
LOOP:
Δx(N) : 1
=
≠
X(N) = X(N-1) + 2**(-S_x)
Y(N) = F(X(N)) + 2**IL
X(N) = X(N-1) - 2**(-S_x)
Y(N) = F(X(N)) + 2**IL
Y(N) + S(N-1) : 2**(IL + 1)
<
≥
S(N-1) - Y(N) : 0
<
≥
S(N)=Y(N)+S(N-1)
DELZ(N) = 0
S(N)=Y(N) +
S(N-1)-2**(IL+1)
DELZ(N) = 1
S(N)=S(N-1)-Y(N)
+ 2**(IL + 1)
DELZ = 0
S(N)=S(N-1)-Y(N)
DELZ = 1
ACTY(N) = (Y(N) - 2**IL)/2** SY
DELZ(N) : 1
=
≠
Z(N) = Z(N-1) + 2**(- S_z)
Z(N) = Z(N-1) - 2**(-S_z)
\|X(N) - X_f\| : 2** (-S_x)
<
≥
STOP
N = N+1, GO TO LOOP

Prior to writing the subroutine, F(x), the following flow chart was prepared:



```
      ALGOL       MARCH 21,1965           INTERFACE   FEBRUARY 15,1965       PASS2   DECEMBER 23, 1964
              1       REAL
BLOCK 1     LEVEL 1                                                                              B0
              2       PROCEDURE F(X,M,SY,FX,FAKE,NOGO)$
BLOCK 2     LEVEL 2
              3       REAL X, FX, FAKE$
              4       INTEGER M, SY, NOGO$  BEGIN
              5       INTEGER LIMIT, N               $                                           B1
              6       REAL FHI, FLO, FTRY          $
              7       FX = 8.0 -      2.0*X*X $
              8       FAKE = FX $
              9       IF ABS(FX) GTR 2**M-1 THEN NOGO = 1 ELSE NOGO = 0 $
             10       FHI = 2**M $ FLO = -(2**M) $
             11       LIMIT = SY + M+ 1 $
             12       N = 1$
             13       CHESS..FTRY = (FHI+FLO)/2   $
             14       IF 2*ABS(FTRY-FX) LEQ 2.0**-SY THEN
             15       FX = FTRY ELSE BEGIN
             16       IF FTRY GTR FX THEN FHI= FTRY ELSE FLO = FTRY $                            B2
             17       IF N LSS LIMIT THEN BEGIN N = N+1 $ GO TO CHESS$ END $                     B3     F*
             18       IF ABS(FHI-FX ) LEQ ABS (FX-FLO) THEN FX = FHI ELSE FX = FLO$ END $        E2
             19       FX = (2.0**SY) * FX + 2**(SY+M) $ END$                                     E1
  END BLOCK 2
             20
             21
             22       START..
             23       BEGIN
             24       INTEGER L $ READ(L) $ BEGIN
BLOCK 3     LEVEL 2                                                                              B4
             25       INTEGER SX, SY, M            ,N ,IL,SZ  ,NOGO ,LAST              $
BLOCK 4     LEVEL 3                                                                              B5
             26       FORMAT FDELX(A1, 72I1) ,SCALES(E2,X5.        ,                             Q
             27                      ,SCALE OF DX =', I3, X3,'SCALE OF DELTA Y =', I3,X3,        Q
             28       'MAXIMUM ACTUAL Y IS', D9.3, A1,                                           Q
             29       'INTEGRATOR LENGTH =',I3,X3, 'SCALE OF DELTA Z (OUTPUT) =',I3,X3,          Q
             30       'UPPER LIMIT OF INTEGRATION, X =' , D9.3, A1),
             31       HEAD(X8,'DELTA', X8,'ACTUAL',X14,'ACTUAL',X13,'INTEGRATOR',X10,            Q
             32       'INTEGRATOR',X4,'DELTA', X8,'ACTUAL', A3,                                  Q
             33       ' N ',X7,'X',X,3,'X',X19,'Y', X20,'Y',                                     Q
             34       X19,'S', X12, 'Z', X13,'Z', A1),
             35       NIT('INITIAL VALUES', R17.7, 3R20.7, X8, R20.7, A1) ,
             36       OUT(I4, I7, 4R20.7, I8, R20.7, A1) ,
             37       MESS('THE VALUE OF Y EXCEEDED CAPACITY OF Y REGISTER', A1)$
             38       REAL XF   ,G                                        $
             39       REAL ARRAY X,Y, S,Z(0..L) $  REAL ARRAY ACTY(0..L) $
             40       INTEGER ARRAY DELX, DELZ(1..L) $
             41       READ (  SX, SY, M, X(0), XF, Z(0), LAST)        $
```

SIMULATION PROGRAM

```
   42       FOR N = (1,1,L, DO DELX(N) = 1 $
   43       DELX(2) = DELX(4) = DELX(5) = 0.00$
   44       IL = M+SY $ SZ = SX-M$
   45       ACTY(0) = 0.0 $
   46       G = 2.0**M - 1.0 $
   47       Y(0) = S(0) = 2**IL $
   48       WRITE(SCALES, SX, SY, G, IL, SZ, XF) $
   49       WRITE(HEAD) $ WRITE( NIT, X(0),ACTY(0), Y(0) , S(0), Z(0) ) $
   50       N=1$
   51       LOOP.. IF DELX(N) EQL 1 THEN BEGIN
   52       X(N) = X(N-1) + 2.0**-SX $                                          B6
   53       F(X(N), M, SY,Y(N), ACTY(N), NOGO) $
   54       BEGIN IF Y(N) + S(N-1) LSS 2**(IL+1) THEN                           B7
   55       BEGIN S(N) = Y(N) + S(N-1) $ DELZ(N) = 0$END                        B8    E8
   56       ELSE BEGIN S(N) = Y(N) + S(N-1) -2**(IL+1) $ DELZ(N) = 1$ END$      B9    E9
   57       END$ END  ELSE BEGIN                                                E7    E6
   58       X(N) = X(N-1) - 2.0**-SX $                                          B10
   59       F(X(N), M, SY,Y(N), ACTY(N), NOGO) $
   60       BEGIN IF S(N-1)  - Y(N) LSS 0 THEN                                  B11
   61       BEGIN S(N) = S(N-1) - Y(N) + 2**(IL+1) $                            B12
   62       DELZ(N) = 0$ END                                                    E12
   63       ELSE BEGIN S(N) = S(N-1) - Y(N) $ DELZ(N) = 1$ END$                 B13   E13
   64       END$ END$                                                           E11   E10
   65       IF NOGO EQL 1 THEN BEGIN                                            B14   E14
   66       WRITE (MESS) $ X(N)  = XF $ END $
   67       IF DELZ(N) EQL 1 THEN
   68       Z(N) = Z(N-1) + 2.0**-SZ ELSE
   69       Z(N) = Z(N-1) - 2.0 ** -SZ $
   70       IF N LEQ 40 THEN
   71       WRITE(OUT,N, DELX(N), X(N), ACTY(N) , Y(N), S(N), DELZ(N), Z(N) ) $
   72       IF 2*ABS(X(N) - XF) GEQ 2.0**-SX THEN
   73       BEGIN N = N + 1$ GO TO LOOP $ END $                                 B15   E15
   74       WRITE(OUT,N, DELX(N), X(N), ACTY(N) , Y(N), S(N), DELZ(N), Z(N) ) $
   75       IF LAST NEQ 1 THEN GO TO START $
   76       END$                                                                E5
 END BLOCK 4
   77       END$                                                                E4
 END BLOCK 3
   78             FINISH                                                        F0
 END BLOCK 1
COMPILATION COMPLETED
```

SIMULATION PROGRAM, CONTINUED

```
     SCALE OF DX = 10   SCALE OF DELTA Y =  6   MAXIMUM ACTUAL Y IS   15.000
INTEGRATOR LENGTH = 10   SCALE OF DELTA Z (OUTPUT) =  6   UPPER LIMIT OF INTEGRATION, X =    1.000


          DELTA        ACTUAL             ACTUAL            INTEGRATOR          INTEGRATOR      DELTA        ACTUAL
  N         X            X                  Y                   Y                   S             Z            Z
INITIAL  VALUES    0.000000, 00       0.000000, 00        1.024000, 03        1.024000, 03               0.000000, 00
   1        1      9.765625,-04       7.999998, 00        1.536000, 03        5.120000, 02        1      1.562500,-02
   2        0     -0.000000, 00       8.000000, 00        1.536000, 03        1.024000, 03        0     -0.000000, 00
   3        1      9.765625,-04       7.999998, 00        1.536000, 03        5.120000, 02        1      1.562500,-02
   4        0     -0.000000, 00       8.000000, 00        1.536000, 03        1.024000, 03        0     -0.000000, 00
   5        0     -9.765625,-04       7.999998, 00        1.536000, 03        1.536000, 03        0     -1.562500,-02
   6        1      0.000000, 00       8.000000, 00        1.536000, 03        1.024000, 03        1      0.000000, 00
   7        1      9.765625,-04       7.999998, 00        1.536000, 03        5.120000, 02        1      1.562500,-02
   8        1      1.953125,-03       7.999992, 00        1.536000, 03        0.000000, 00        1      3.125000,-02
   9        1      2.929687,-03       7.999982, 00        1.536000, 03        1.536000, 03        0      1.562500,-02
  10        1      3.906250,-03       7.999969, 00        1.536000, 03        1.024000, 03        1      3.125000,-02
  11        1      4.882812,-03       7.999952, 00        1.536000, 03        5.120000, 02        1      4.687500,-02
  12        1      5.859375,-03       7.999931, 00        1.536000, 03        0.000000, 00        1      6.250000,-02
  13        1      6.835937,-03       7.999906, 00        1.536000, 03        1.536000, 03        0      4.687500,-02
  14        1      7.812500,-03       7.999877, 00        1.536000, 03        1.024000, 03        1      6.250000,-02
  15        1      8.789062,-03       7.999845, 00        1.536000, 03        5.120000, 02        1      7.812500,-02
  16        1      9.765625,-03       7.999809, 00        1.536000, 03        0.000000, 00        1      9.375000,-02
  17        1      1.074218,-02       7.999769, 00        1.536000, 03        1.536000, 03        0      7.812500,-02
  18        1      1.171875,-02       7.999725, 00        1.536000, 03        1.024000, 03        1      9.375000,-02
  19        1      1.269531,-02       7.999677, 00        1.536000, 03        5.120000, 02        1      1.093750,-01
  20        1      1.367187,-02       7.999626, 00        1.536000, 03        0.000000, 00        1      1.250000,-01
  21        1      1.464843,-02       7.999570, 00        1.536000, 03        1.536000, 03        0      1.093750,-01
  22        1      1.562500,-02       7.999511, 00        1.536000, 03        1.024000, 03        1      1.250000,-01
  23        1      1.660156,-02       7.999448, 00        1.536000, 03        5.120000, 02        1      1.406250,-01
  24        1      1.757812,-02       7.999382, 00        1.536000, 03        0.000000, 00        1      1.562500,-01
  25        1      1.855468,-02       7.999311, 00        1.536000, 03        1.536000, 03        0      1.406250,-01
  26        1      1.953125,-02       7.999237, 00        1.536000, 03        1.024000, 03        1      1.562500,-01
  27        1      2.050781,-02       7.999158, 00        1.536000, 03        5.120000, 02        1      1.718750,-01
  28        1      2.148437,-02       7.999076, 00        1.536000, 03        0.000000, 00        1      1.875000,-01
  29        1      2.246093,-02       7.998991, 00        1.536000, 03        1.536000, 03        0      1.718750,-01
  30        1      2.343750,-02       7.998901, 00        1.536000, 03        1.024000, 03        1      1.875000,-01
  31        1      2.441406,-02       7.998807, 00        1.536000, 03        5.120000, 02        1      2.031250,-01
  32        1      2.539062,-02       7.998710, 00        1.536000, 03        0.000000, 00        1      2.187500,-01
  33        1      2.636718,-02       7.998609, 00        1.536000, 03        1.536000, 03        0      2.031250,-01
  34        1      2.734375,-02       7.998504, 00        1.536000, 03        1.024000, 03        1      2.187500,-01
  35        1      2.832031,-02       7.998395, 00        1.536000, 03        5.120000, 02        1      2.343750,-01
  36        1      2.929687,-02       7.998283, 00        1.536000, 03        0.000000, 00        1      2.500000,-01
  37        1      3.027343,-02       7.998167, 00        1.536000, 03        1.536000, 03        0      2.343750,-01
  38        1      3.125000,-02       7.998046, 00        1.536000, 03        1.024000, 03        1      2.500000,-01
  39        1      3.222656,-02       7.997922, 00        1.536000, 03        5.120000, 02        1      2.656250,-01
  40        1      3.320312,-02       7.997795, 00        1.536000, 03        0.000000, 00        1      2.812500,-01
1030        1      1.000000, 00       6.000000, 00        1.408000, 03        2.830000, 02        1      7.343750, 00
```

RESULTS OF FIRST RUN

```
     SCALE OF DX = 11   SCALE OF DELTA Y =  6   MAXIMUM ACTUAL Y IS   15.000
INTEGRATOR LENGTH = 10   SCALE OF DELTA Z (OUTPUT) =  7   UPPER LIMIT OF INTEGRATION, X =    1.000

            DELTA        ACTUAL              ACTUAL              INTEGRATOR         INTEGRATOR      DELTA        ACTUAL
   N          X            X                   Y                     Y                  S             Z            Z
INITIAL  VALUES       0.000000, 00        0.000000, 00        1.024000, 03        1.024000, 03                 0.000000, 00
    1         1        4.882812,-04        7.999999, 00        1.536000, 03        5.120000, 02       1        7.812500,-03
    2         0       -0.000000, 00        8.000000, 00        1.536000, 03        1.024000, 03       0       -0.000000, 00
    3         1        4.882812,-04        7.999999, 00        1.536000, 03        5.120000, 02       1        7.812500,-03
    4         0       -0.000000, 00        8.000000, 00        1.536000, 03        1.024000, 03       0       -0.000000, 00
    5         0       -4.882812,-04        7.999999, 00        1.536000, 03        1.536000, 03       0       -7.812500,-03
    6         1        0.000000, 00        8.000000, 00        1.536000, 03        1.024000, 03       1        0.000000, 00
    7         1        4.882812,-04        7.999999, 00        1.536000, 03        5.120000, 02       1        7.812500,-03
    8         1        9.765625,-04        7.999998, 00        1.536000, 03        0.000000, 00       1        1.562500,-02
    9         1        1.464843,-03        7.999995, 00        1.536000, 03        1.536000, 03       0        7.812500,-03
   10         1        1.953125,-03        7.999992, 00        1.536000, 03        1.024000, 03       1        1.562500,-02
   11         1        2.441406,-03        7.999988, 00        1.536000, 03        5.120000, 02       1        2.343750,-02
   12         1        2.929687,-03        7.999982, 00        1.536000, 03        0.000000, 00       1        3.125000,-02
   13         1        3.417968,-03        7.999976, 00        1.536000, 03        1.536000, 03       0        2.343750,-02
   14         1        3.906250,-03        7.999969, 00        1.536000, 03        1.024000, 03       1        3.125000,-02
   15         1        4.394531,-03        7.999961, 00        1.536000, 03        5.120000, 02       1        3.906250,-02
   16         1        4.882812,-03        7.999952, 00        1.536000, 03        0.000000, 00       1        4.687500,-02
   17         1        5.371093,-03        7.999942, 00        1.536000, 03        1.536000, 03       0        3.906250,-02
   18         1        5.859375,-03        7.999931, 00        1.536000, 03        1.024000, 03       1        4.687500,-02
   19         1        6.347656,-03        7.999919, 00        1.536000, 03        5.120000, 02       1        5.468750,-02
   20         1        6.835937,-03        7.999906, 00        1.536000, 03        0.000000, 00       1        6.250000,-02
   21         1        7.324218,-03        7.999892, 00        1.536000, 03        1.536000, 03       0        5.468750,-02
   22         1        7.812500,-03        7.999877, 00        1.536000, 03        1.024000, 03       1        6.250000,-02
   23         1        8.300781,-03        7.999862, 00        1.536000, 03        5.120000, 02       1        7.031250,-02
   24         1        8.789062,-03        7.999845, 00        1.536000, 03        0.000000, 00       1        7.812500,-02
   25         1        9.277343,-03        7.999827, 00        1.536000, 03        1.536000, 03       0        7.031250,-02
   26         1        9.765625,-03        7.999809, 00        1.536000, 03        1.024000, 03       1        7.812500,-02
   27         1        1.025390,-02        7.999789, 00        1.536000, 03        5.120000, 02       1        8.593750,-02
   28         1        1.074218,-02        7.999769, 00        1.536000, 03        0.000000, 00       1        9.375000,-02
   29         1        1.123046,-02        7.999747, 00        1.536000, 03        1.536000, 03       0        8.593750,-02
   30         1        1.171875,-02        7.999725, 00        1.536000, 03        1.024000, 03       1        9.375000,-02
   31         1        1.220703,-02        7.999701, 00        1.536000, 03        5.120000, 02       1        1.015625,-01
   32         1        1.269531,-02        7.999677, 00        1.536000, 03        0.000000, 00       1        1.093750,-01
   33         1        1.318359,-02        7.999652, 00        1.536000, 03        1.536000, 03       0        1.015625,-01
   34         1        1.367187,-02        7.999626, 00        1.536000, 03        1.024000, 03       1        1.093750,-01
   35         1        1.416015,-02        7.999599, 00        1.536000, 03        5.120000, 02       1        1.171875,-01
   36         1        1.464843,-02        7.999570, 00        1.536000, 03        0.000000, 00       1        1.250000,-01
   37         1        1.513671,-02        7.999541, 00        1.536000, 03        1.536000, 03       0        1.171875,-01
   38         1        1.562500,-02        7.999511, 00        1.536000, 03        1.024000, 03       1        1.250000,-01
   39         1        1.611328,-02        7.999480, 00        1.536000, 03        5.120000, 02       1        1.328125,-01
   40         1        1.660156,-02        7.999448, 00        1.536000, 03        0.000000, 00       1        1.406250,-01
 2054         1        1.000000, 00        6.000000, 00        1.408000, 03        1.656000, 03       0        7.328125, 00
```

RESULTS OF SECOND RUN

The two simulated integrations included in this report approximate the area under the parabola $y = 8 - 2x^2$ from $x = 0$ to $x= 1$. This is shown in Figure 9.

The schematic for the first run is shown in Figure 10.

The program was given the scales of dx and dy and the maximum value of y. The integrator length and scale of dz are determined by

$$IL = S_y + S_{dy}$$

$$S_{dz} = S_{dx} - S_y .$$

It will be noted that in the program the binary method of increments is used for dx and dz but not for dy. However, the value of y used in the calculations is limited in accuracy by the number of bits in the Y register. Note in the results that for the first 40 iterations, the actual y value changes from 8.0 to 7.99779. This change is too small to change the value in the Y register which is significant to 6 binary places ($S_{dy} = 6$). The printed value of 1536 for the Y register includes the sign bit and ignores binary point. It may be decoded as follows:

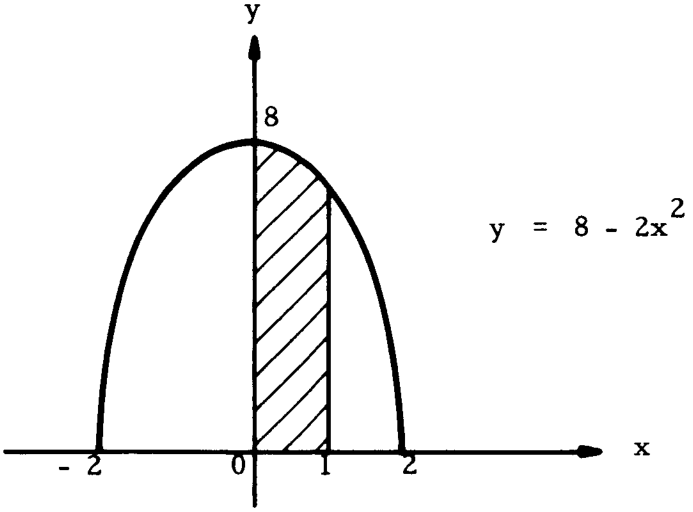


Figure 9 Simulated Integration

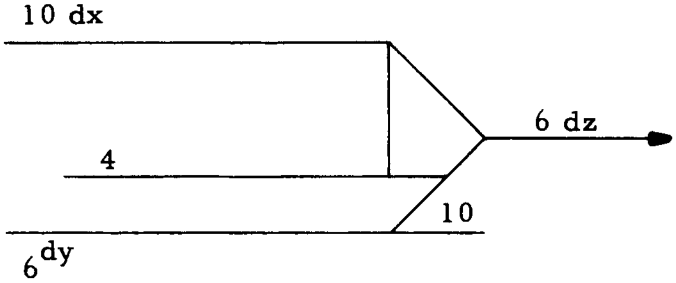


Figure 10 Schematic for First Computer Run

$$1536 - 2^{10}\text{ (sign bit)} = 1536 - 1024$$
$$= 512$$

$$(512)_{10} = (1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0)_2$$

Inserting the binary point six places from the right ($S_{dy}=6$),

$(1\ 0\ 0\ 0\ .\ 0\ 0\ 0\ 0\ 0\ 0)_2 = 8.0$, the value of y.

Note that the Y register does gradually change since on the last iteration it is 1408 when y has decreased to 6.

For each step if $\Delta x = 1$, the Y register is added to the S register and if $\Delta x = 0$, then Y is subtracted from S. If there is an overflow after addition or no outgoing borrow after subtraction, then $\Delta z = 1$; otherwise $\Delta z = 0$. A summation of all previous Δz outputs is the value of the integral, recorded in the last column. Note that when $\Delta z = 0$, an increment of Z ($.015625 = 2^{-6}$, since $S_{dz} = 6$) is subtracted from the summation.

The three zeros for Δx (cycles 2, 4, and 5) may be interpreted as increments in the negative direction on the x axis. The value of z for this initial period oscillates then goes negative after cycle 5. This may be explained since at this point $\Sigma \Delta x = -1$, hence the actual value of x is to the *left* of the y axis. The area *should* be negative at this point.

Since $S_{dx} = 10$, the value of $x = 1$ should occur after $2^{10} = 1024$ cycles. Because three of the Δx increments were negative, there must be three extra positive Δx increments or a total of 6 extra Δx.

$1024 + 6 = 1030$, the printed number of iterations.

(The three negative Δx increments were inserted into the program for illustrative purposes only.)

The final value of z is compared with the actual value to determine the error.

$$\begin{array}{rl} 7.343750 & = \text{calculated area} \\ \underline{-\ 7.333333} & = \text{actual area} \\ \epsilon = .010417 & \end{array}$$

In the second approximation of the area, the only change was the scale of dx from 10 to 11. Now rather than summing 1024 strips between 0 and 1, the integral will be the summation of 2048 strips. Hence, the running time is doubled.

$$
\begin{array}{rrl}
 & 7.333333 & = \text{actual area} \\
 & \underline{-\ 7.328125} & = \text{calculated area} \\
\epsilon = & .005208 &
\end{array}
$$

This illustrates the inverse relationship between running time and error. By doubling the running time, the error was halved.

The results of this program provided the incentive to design an integrator with the binary method of increments.

CHAPTER III

DESIGN AND IMPLEMENTATION

Integrator Design

This chapter deals with the design of an integrator which could be used as the basic unit in a parallel DDA. The increments Δx, Δy, and Δz are coded in the binary system. A serial adder/ subtractor is used to combine Y with the contents of the S register; thus, the inaccuracies involved with one's complement addition (see example 9) are avoided. Although no greater accuracy will result, an adder/subtractor is used to combine $\Sigma \Delta y$ and Y to simplify circuit design.

The integrator is capable of operating in either the interpolative or extrapolative mode. For versatility of interconnection, the integrator length may be varied from 4 to 10. Also, the Y register may be permitted to overflow, thus making possible other uses such as an adder to combine pulse trains. The Δz output is compatible with both Δx and Δy inputs.

The block diagram is shown in Figure 11. This diagram when expanded to include timing and more interconnections is shown in Figure 12.

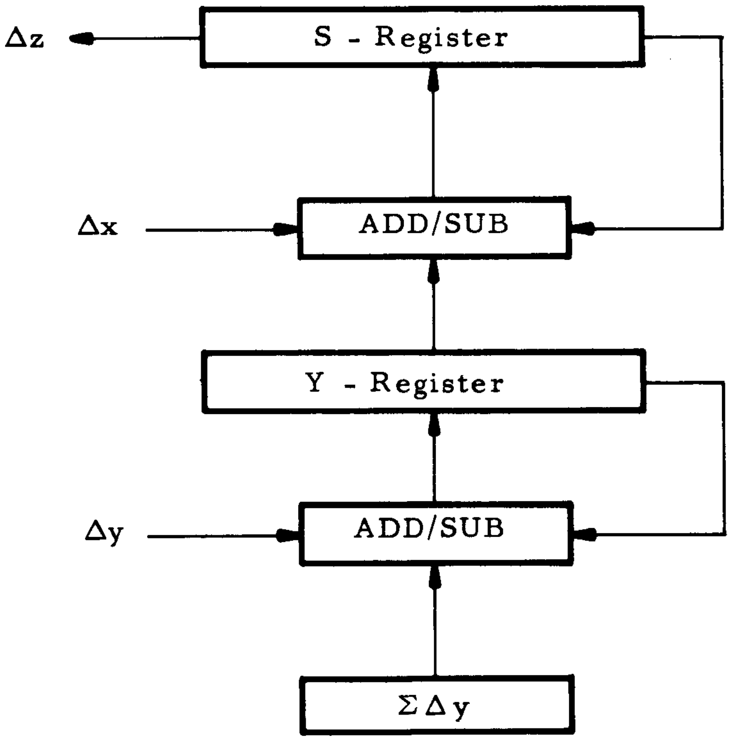


Figure 11 Integrator Block Diagram

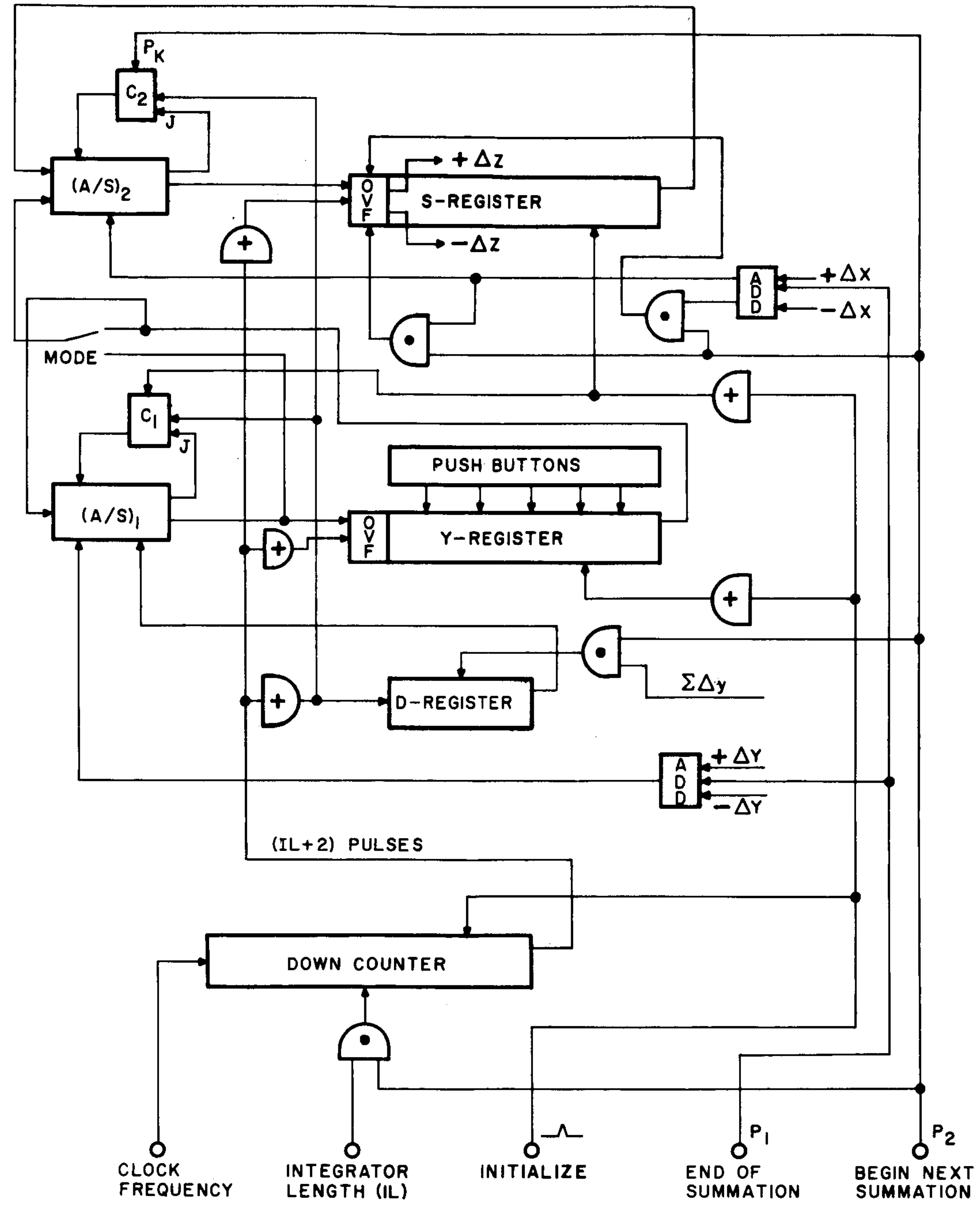


FIGURE 12 DETAILED BLOCK DIAGRAM

Operation

Before beginning an integration process, the initialize pulse resets the Y and S registers to zero and resets C_1 to zero. If the initial value for Y is other than zero, it is read into the Y register via parallel inputs. During the start pulse, P_2, the integrator length is set into the down counter, $\Sigma \Delta y$ is read into the D register, the S register overflow bit is set or reset, and C_2 is reset. After P_2 the down counter emits IL + 2 pulses at the clock frequency to completely shift the registers through the adder/subtractors. The end of summation pulse, P_1, sets or resets the ADD flip-flop, depending upon the signs of Δx and Δy. The cycle is then repeated.

The mode switch, also in Figure 12, permits choice of the old value of y which is in the register or the latest value of y as it comes out of the adder/subtractor.

The detailed logic design is in Appendix I. These circuits were divided to fit on 5 printed circuit boards approximately 4 3/4" x 5" as follows:

2 boards: two 12 bit shift registers.

1 board : two full adder/subtractors.

2 boards: one 3 bit shift register,
one 4 bit backward counter, and
assorted gates and flip-flops for
timing or control.

The complete integrator and the associated control unit are shown in Figure 13. Other views of the integrator and two of the circuit boards are shown in Figures 14 through 17. The lights are also on circuit boards for easy insertion. They display the contents of the S and Y registers. The upper rotary switch has seven positions for setting the integrator length, i.e., the length of registers S and Y. The lower two position rotary switch is set depending upon whether there is one or more than one dy input. When there is one dy input, the D register is set to 001; when there is more than one dy input, the switch permits $\Sigma \Delta y$ to be parallel shifted into the D register by an adder unit which it is assumed is part of the DDA. The push buttons across the front panel are for presetting an initial condition other than zero into the Y register. The toggle switch determines the mode — interpolative or extrapolative.

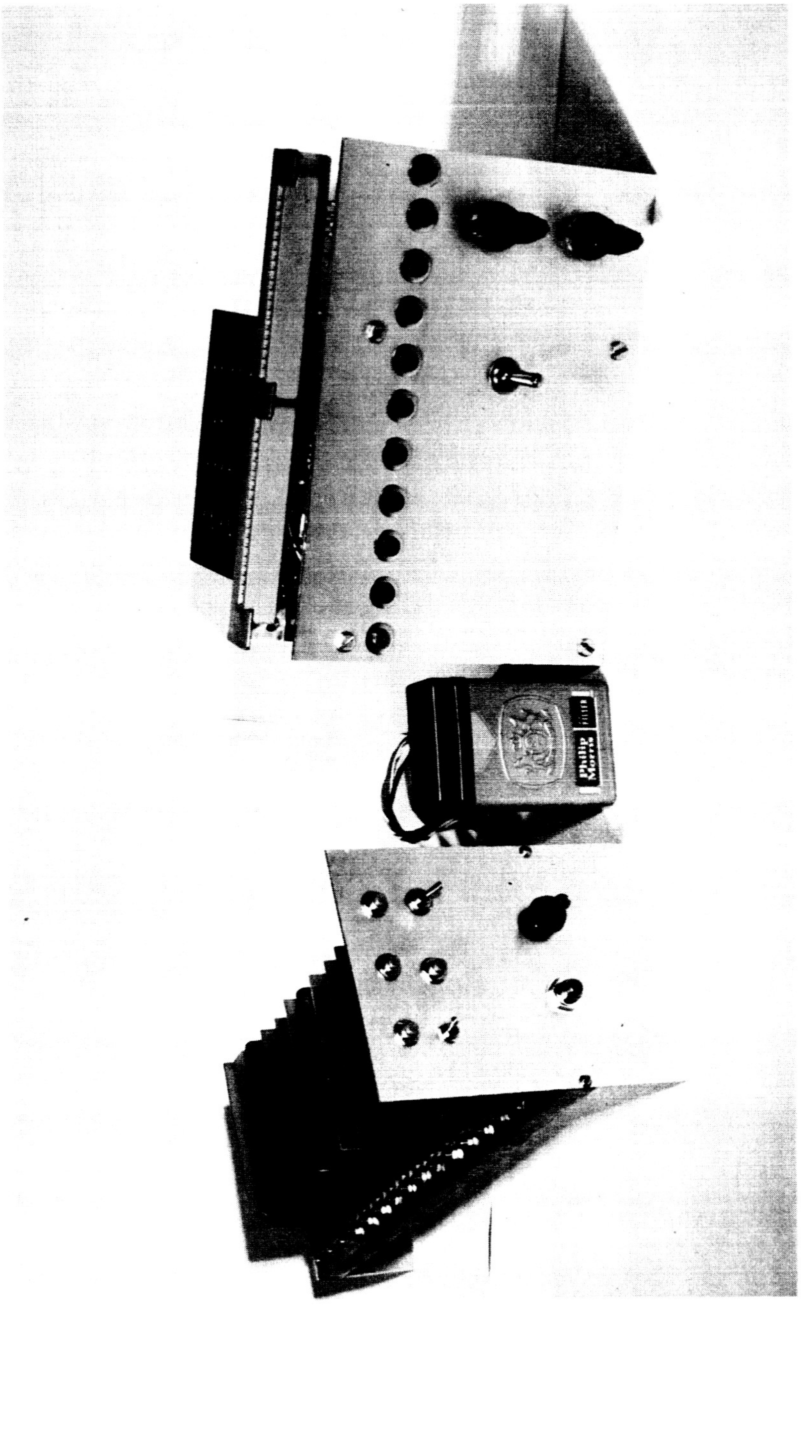


Figure 13 Integrator with Control

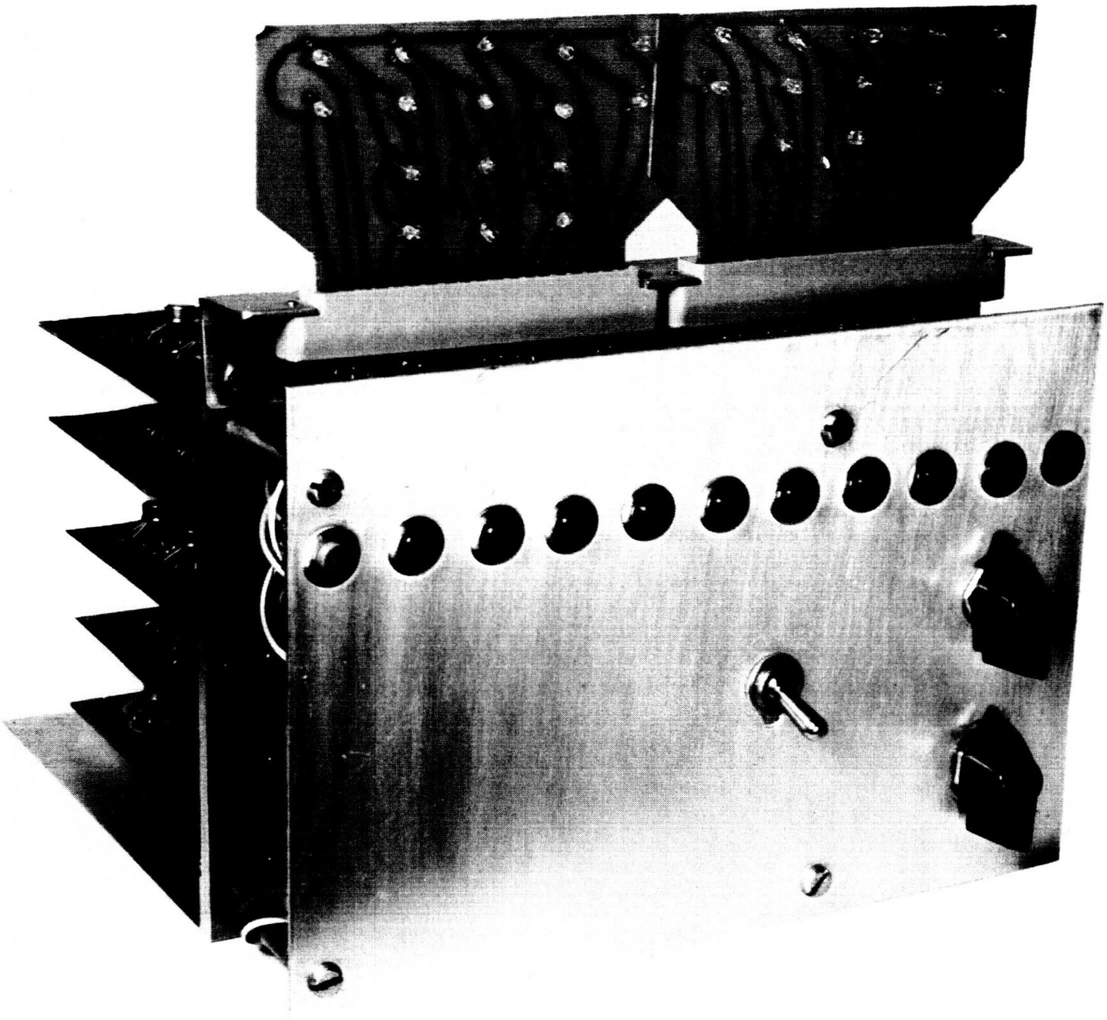


Figure 14 Integrator - Front View

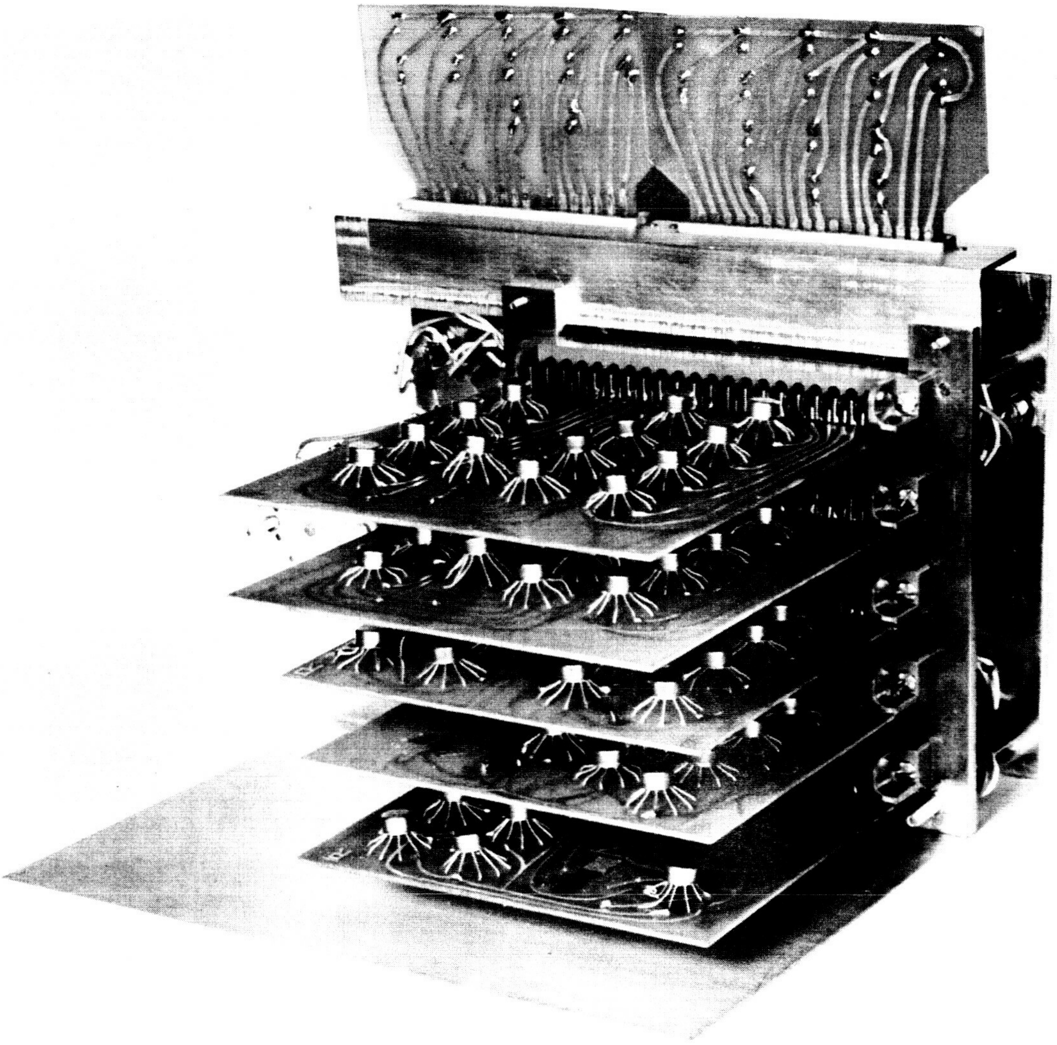


Figure 15 Integrator - Rear View

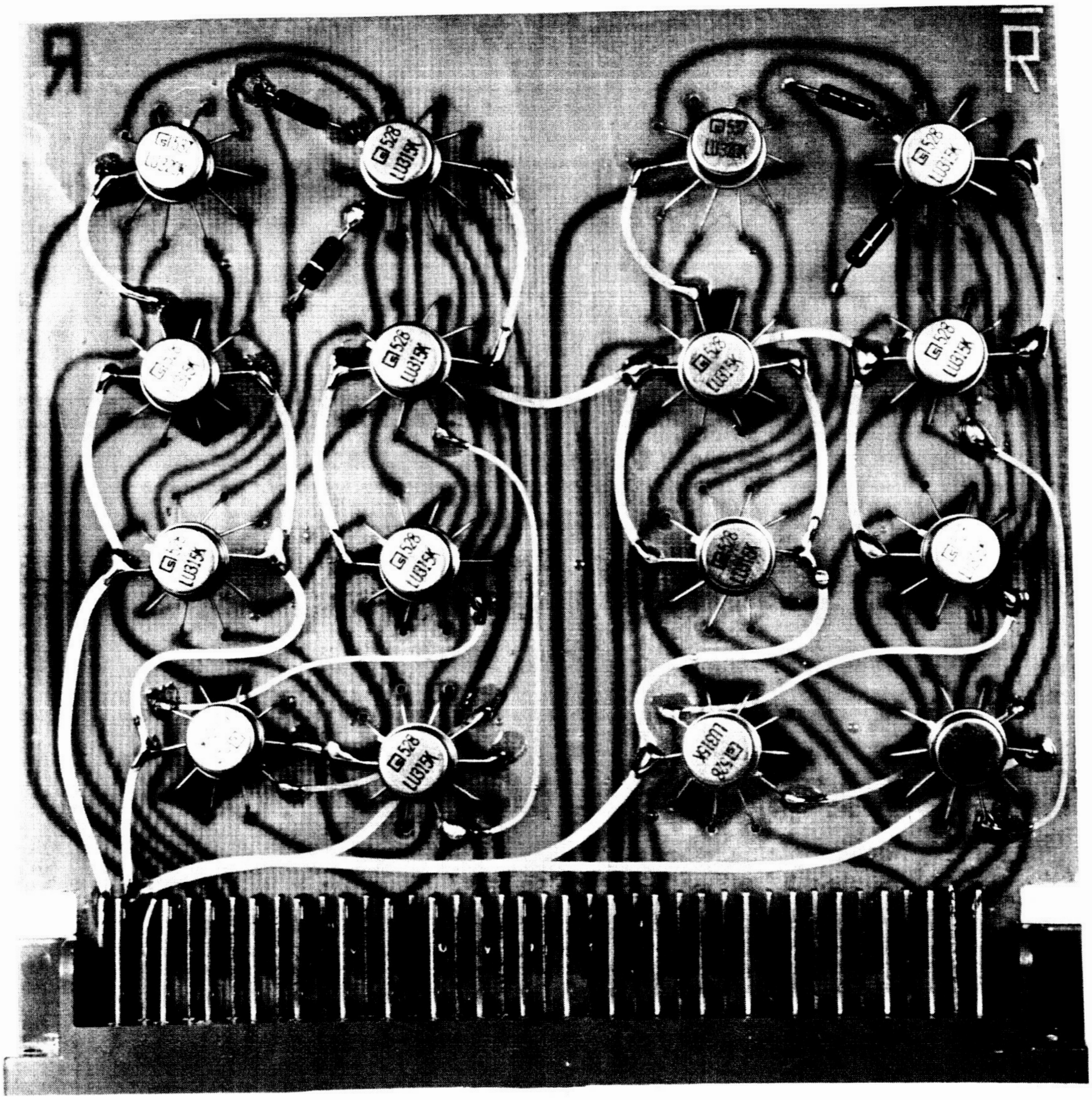


Figure 16 Circuit Board Containing
Two Full Adder/Subtractors

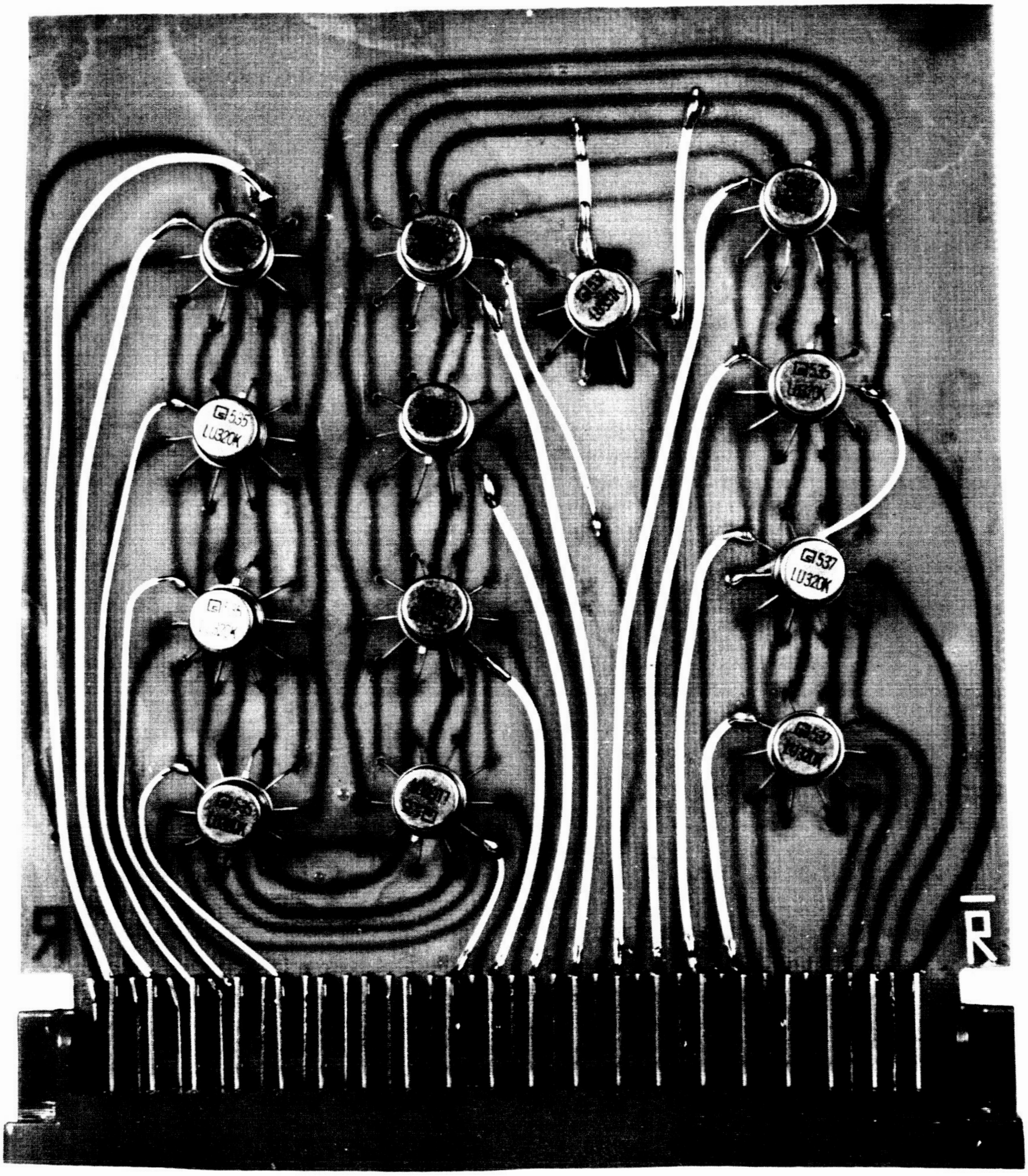


Figure 17 Circuit Board Containing a 12 bit Shift Register.

CHAPTER IV

COMMENTS AND RECOMMENDATIONS

The use of integrated circuits for this project presented no major implementation problems. No noise problems or capacitance problems were encountered. Five printed circuit boards were used to contain the circuits but the integrator presented in this report could easily be placed on four boards at an average of 15 cans/board. A greater packing density could be obtained using multilayer boards. The previous pictures of the boards show that the spacing could be decreased if interconnections were made on more than two planes.

Another space-saving approach would be to use integrated circuit modules. The integrator is basically two shift registers and two adder/subtractors. Hence, it is conceivable that the bulk of the integrator could be contained in four cans.

The potential accuracy of the integrator is limited by the length of the shift registers. Thus, if greater accuracy than the results in Appendix IV is desired, the integrator registers must be greater than 12 bits.

An easily overlooked feature of the Signetics integrated circuits is that they can supply 2 mA at the "1" level and sink 12 mA at the "0" level. Therefore, if a lamp is to indicate the state of a binary element, it should be connected between a positive voltage and either output of the element. Excluding other loading, the lamp may operate with up to 12 mA without overloading the element. The "0" voltage is less than .6 volts while the supply voltage is 4.5 volts. Hence the ideal lamp should require a voltage of 4.5 - .6 = 3.9 volts. A 3 volt lamp was used in this project and diodes provided the necessary voltage drop between the lamp and the supply voltage.

In the interpolative mode the integrator may be operated with a clock frequency up to 1.6 Mc. However, in the extrapolative mode the maximum rate is 850 Kc. This is due to the fact that in the extrapolative mode the two adder/ subtractors operate in series rather than in parallel. This situation could be remedied by designing one synchronous arithmetic unit which would implement the following equations:

Mode	Outputs
	(letters refer to registers)
Interpolative	$0_1 = Y \pm D$
	$0_2 = S \pm Y$
Extrapolative	$0_1 = Y \pm D$
	$0_2 = S \pm (Y \pm D)$

With this arithmetic unit the maximum frequency should not be mode dependent.

APPENDIX I

LOGIC DESIGN

Before considering the detailed logic, it is necessary to mention some details about the micrologic used for implementation.

The input of an AND gate is a standard sink load. The input of an OR gate is a standard source load. The flip flop J and K inputs represent 1/2 sink load each. The P_J and P_K inputs are one source load each, and the clock input is 3/4 sink load. The NORS, ORS, and flip flop outputs have a maximum fan-out capability of 12 source loads plus 5 sink loads. The fan-out of the AND gate is limited to 10 source loads. These rules are adhered to in the circuit to follow.

All logic is positive-tru e, i.e., positive voltages correspond to logical 1 or true, while negative voltages correspond to logical 0, or false. The elements may be used for functions other than their name by defining other than positive-true on their inputs and outputs. See Table 1.

The configuration for setting or resetting the S register overflow bit was designed by noting that the carry produced by

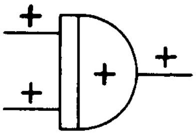
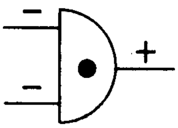
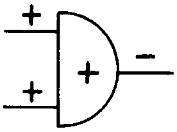
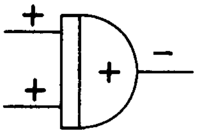
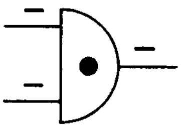
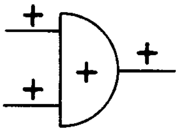
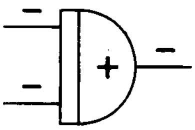
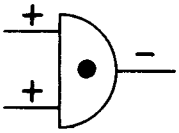
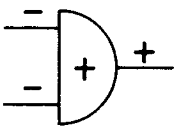
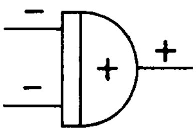
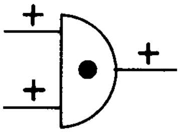
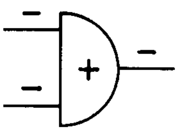
Logical	Circuit Element		
Function	NOR	AND	OR
NOR			
OR			
NAND			
AND			

Table 1 Logic Elements

adding the two's complement is the complement of the borrow produced by subtraction. This was proved earlier in this paper. When subtracting, an outgoing borrow will produce a one in the overflow bit; no outgoing borrow, a zero. But in both cases the opposite is necessary for the proper Δz output. Hence, by setting the overflow bit to a one prior to the summation (when $\Delta x = -1$), then the final state of the overflow bit is complemented. If $\Delta x = + 1$, the overflow bit is preset to zero since addition will occur.

If the output of the integrator is to be minus the integral, the $+ \Delta z$ and $- \Delta z$ output lines are interchanged. Then the complement of the output pulse train is produced.

Adder/Subtractor

The adder/subtractor must be a "full" adder/subtractor; i.e., it must accept a borrow or carry input and produce a borrow or carry output when appropriate. In Table 2 the truth table is shown.

A	B	C_i/B_i	ADD	S/D	C_o/B_o
0	0	0	1	0	0
0	0	1	1	1	0
0	1	0	1	1	0
0	1	1	1	0	1
1	0	0	1	1	0
1	0	1	1	0	1
1	1	0	1	0	1
1	1	1	1	1	1
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	0	0	1
1	0	0	0	1	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	1	1

Table 2 Truth table for A + B, A - B.

The maps for S/D and C_0/B_0 are in Tables 3 and 4.

A	B	0^0	0^1	1^1	1^0	C^{ADD}
0	0	0	0	1	1	
0	1	1	1	0	0	
1	1	0	0	1	1	
1	0	1	1	0	0	

Table 3:

Map for S/D

A	B	0^0	0^1	1^1	1^0	C^{ADD}
0	0	0	0	0	1	
0	1	1	0	1	1	
1	1	0	1	1	1	
1	0	0	0	1	0	

Table 4:

Map for C_0/B_0

$$S/D = A \oplus B \oplus C$$

$$S/D = A\bar{B}\bar{C} + ABC + \bar{A}B\bar{C} + \bar{A}\bar{B}C$$

$$C_0/B_0 = \bar{A}\,C\,\overline{ADD} + \bar{A}\,B\,\overline{ADD} + BC$$
$$+ \ AC(ADD) + AB(ADD)$$

$$C_0/B_0 = \bar{A}\,\overline{ADD}\,(B + C) + A(ADD)\,(B + C) + BC$$

Other equations are possible, but since the micrologic gates have three inputs, these equations were chosen to minimize gates. One four input gate is necessary. This is implemented by using two diodes on one of the gate legs as shown in Figure 18.

The shift register in Figure 19 contains all necessary input and output lines for either register Y or register S. This

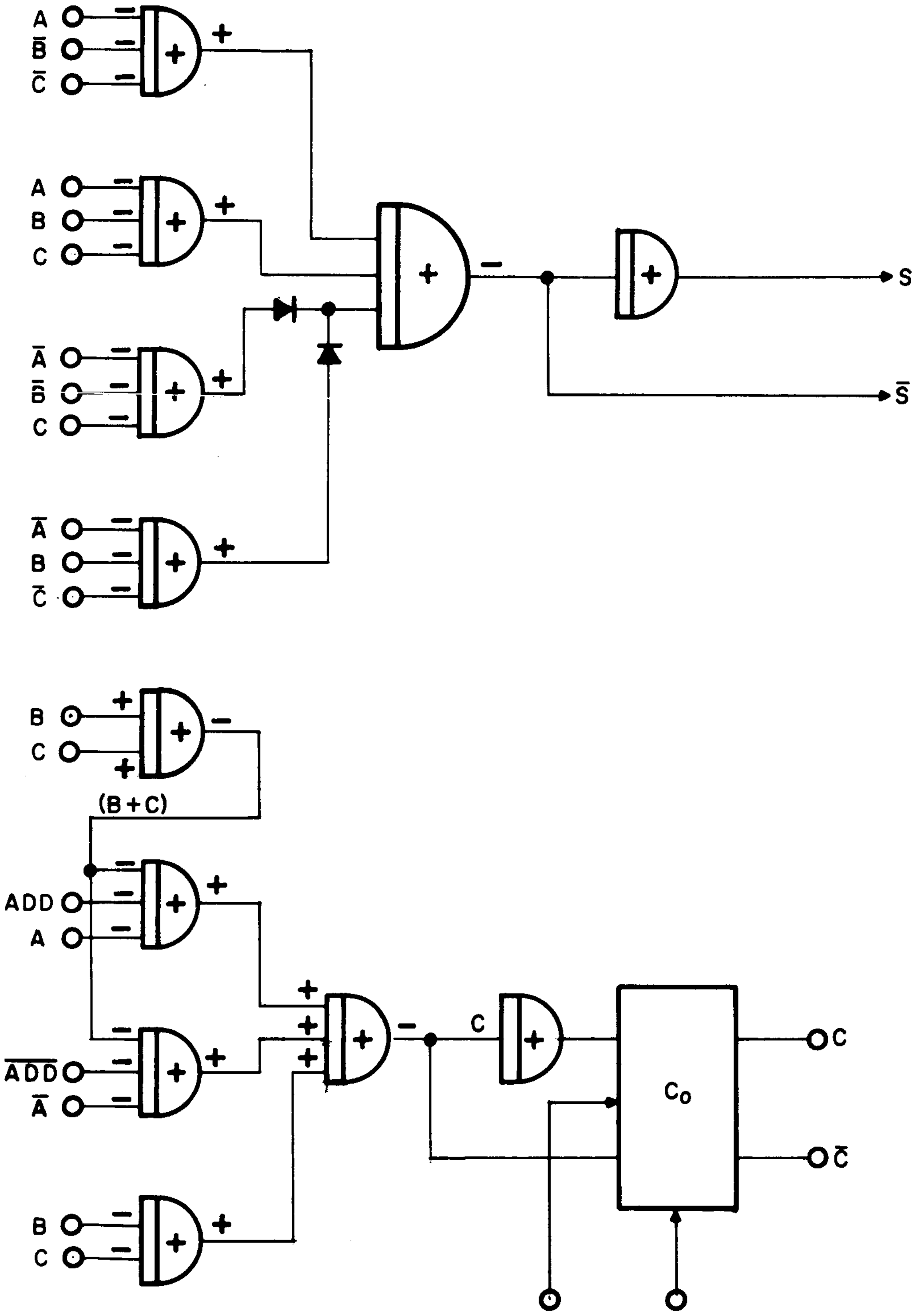


Figure 18 Full Adder/Subtractor

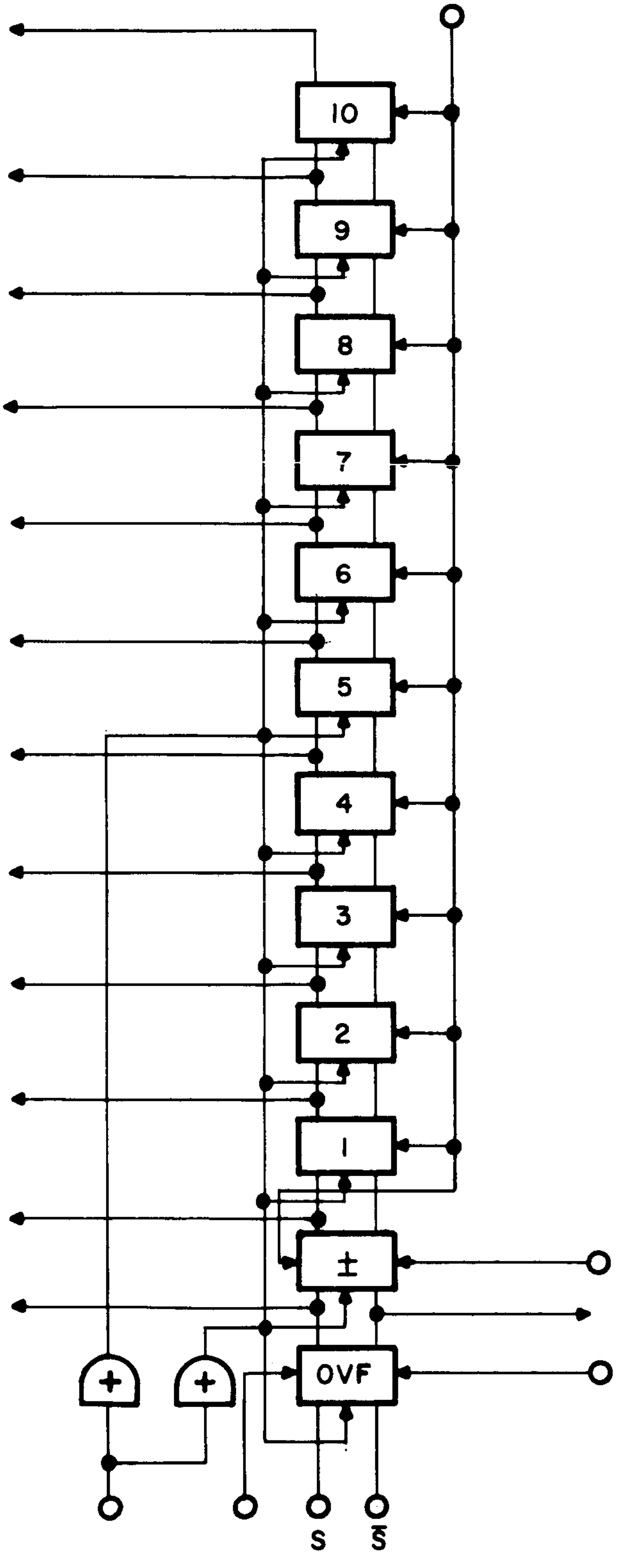


Figure 19 Shift Register

permits one circuit board design to meet the needs of both registers.

The down counter is shown in Figure 20. The integrator length is set on a rotary switch before starting the process. For each cycle the input levels will be such that IL + 1 is read into the counter. It will then emit IL + 2 shift pulses to the rest of the integrator. The counter stops at 1 1 1 1.

Figure 21 shows the D register. Any number representing $\Sigma\,\Delta y$ from 0 to 7 may be set into this register. The sign of $\Sigma\,\Delta y$ is recorded by the ADD flip flop (shown in Figure 12) whose inputs are Δy and $-\Delta y$. It is assumed that a separate unit which combines several dy inputs will feed this device when more than one dy input is desired.

The control unit in Figure 22 was built in addition to the integrator to test its operation. A control similar to this would be adequate for up to 12 integrators. The unit provides to the integrator a basic frequency and the pulses P_1 and P_2 at the proper times. After the start pulse the eleven bit backward counter permits 2^{10} P_2 pulses to go to the integrator. The pulse generator produces from 400 Kc to 1.8 Mc operating frequency when a 15K potentiometer is used for R.

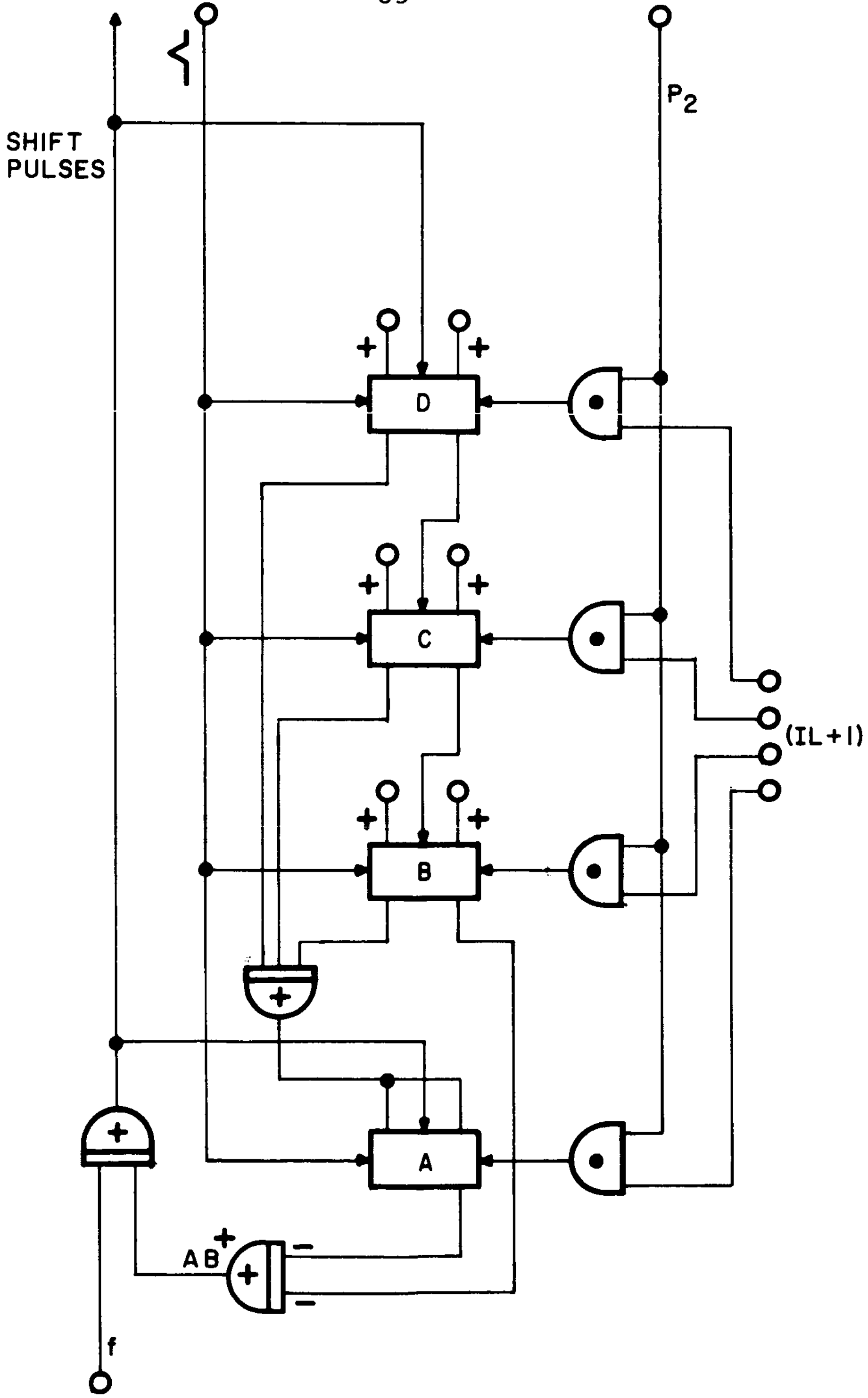


Figure 20 Down Counter

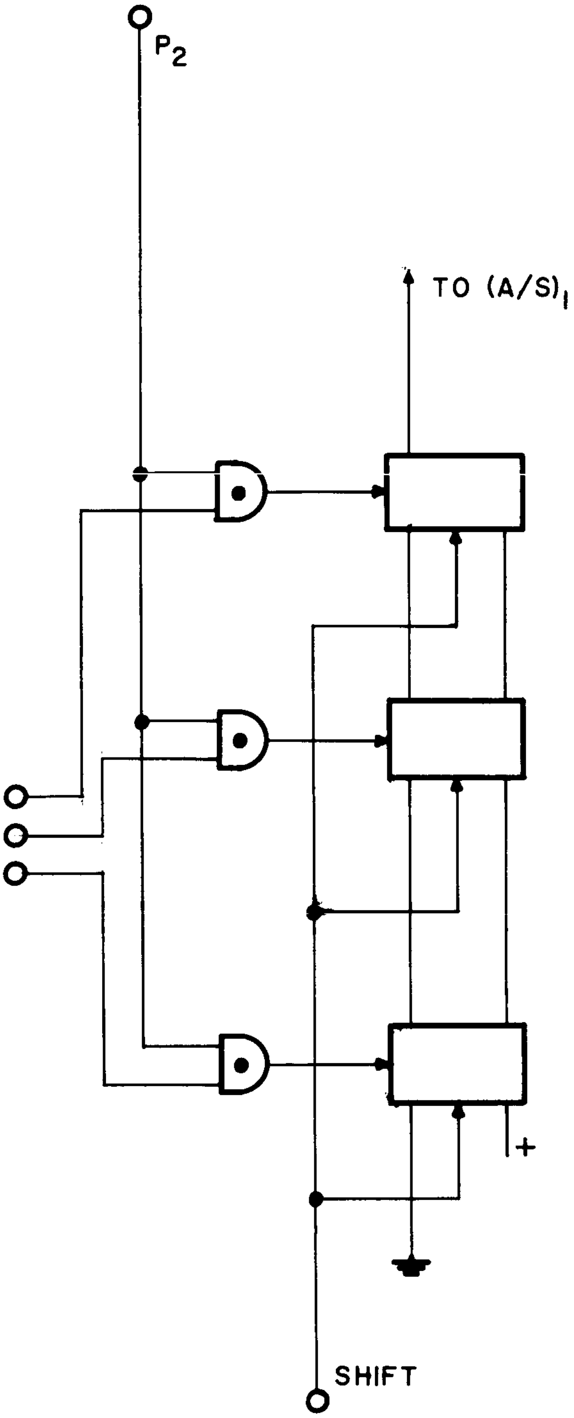


Figure 21 D Register

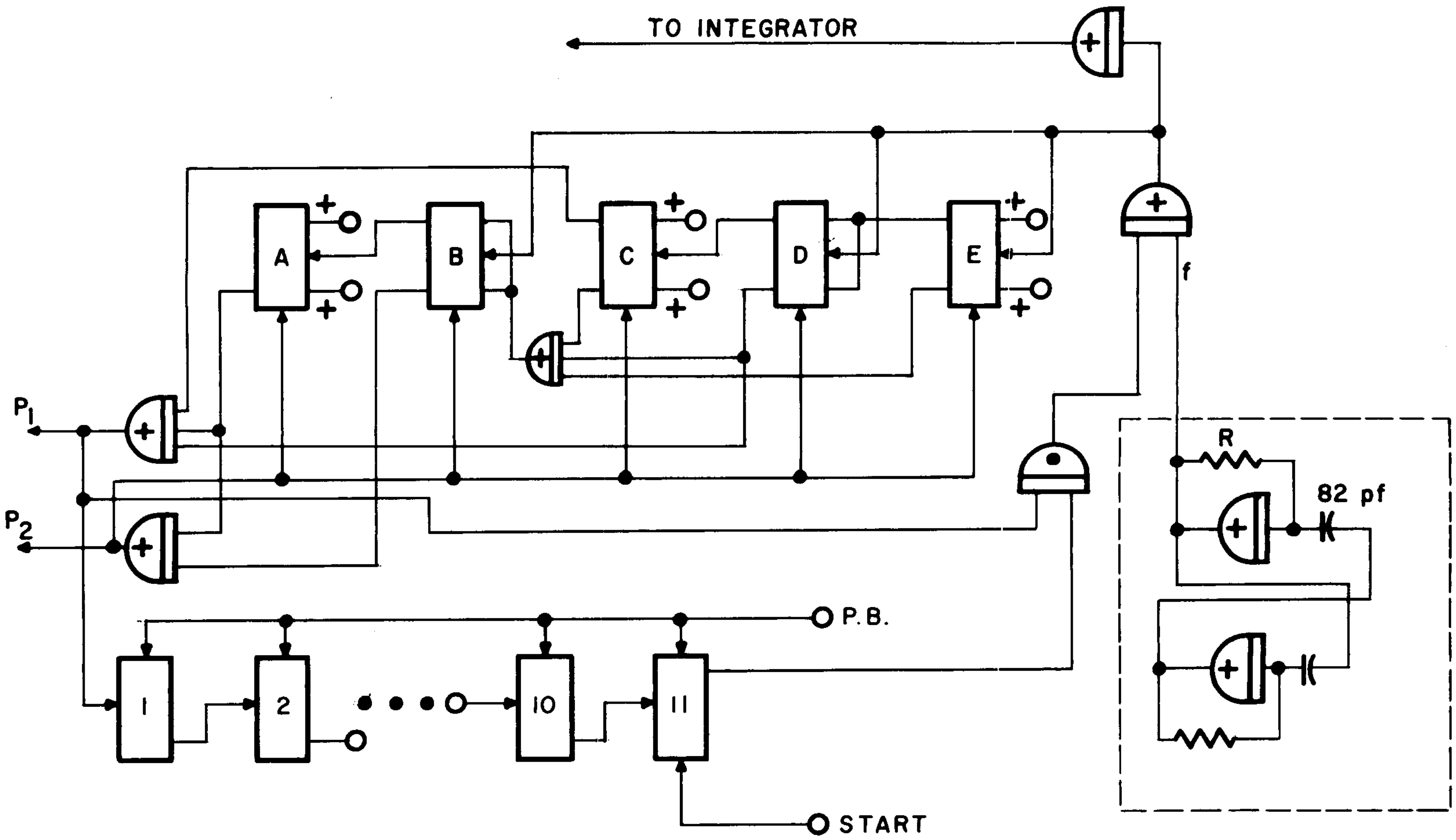


Figure 22 Control Unit

APPENDIX II
INTEGRATED CIRCUITS

Table 5 itemizes the amount of logic necessary to implement the integrator and control unit.

INTEGRATOR

QUANTITY		ITEM
35		J-K binary elements
	16	Dual NOR gates (2 gates/can)
	4	Dual OR gates
	5	Dual AND gates
	25	gate cans
35		binary element cans
TOTAL 60		TO-5 cans for integrator.

CONTROL UNIT

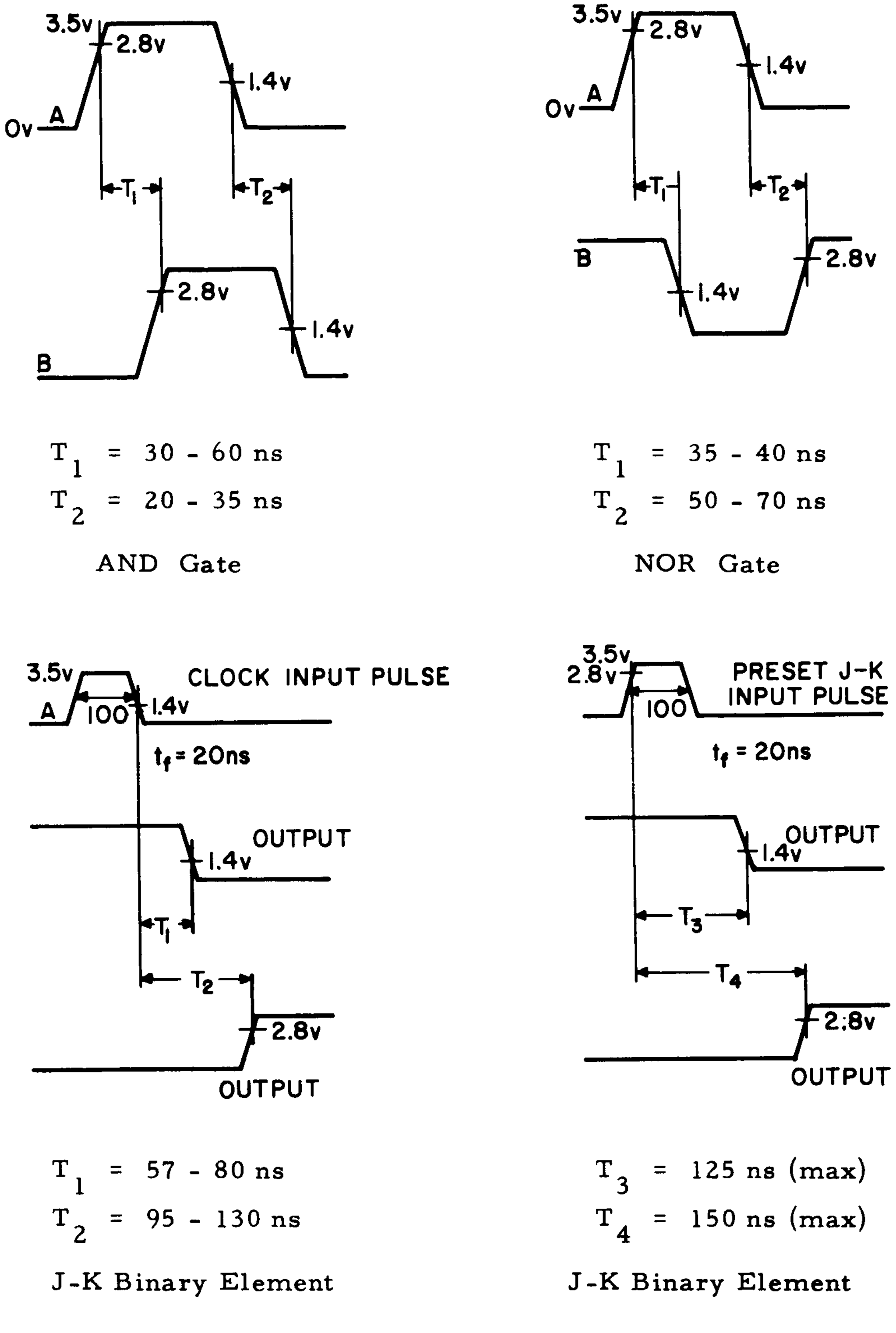
QUANTITY		ITEM
16		J-K binary elements
	4	Dual NOR gates
	1	Dual AND gate
	5	gate cans
16		binary element cans
TOTAL 21		TO-5 cans for control
TOTAL 81		TO-5 cans for project.

Table 5 Quantity of Logic Elements Used

APPENDIX III

TIMING

The delays associated with the logic elements are noted in Figure 23. The timing diagram for critical voltage levels is Figure 24. All measurements are in nanoseconds.



T_1 = 30 - 60 ns

T_2 = 20 - 35 ns

AND Gate

T_1 = 35 - 40 ns

T_2 = 50 - 70 ns

NOR Gate

T_1 = 57 - 80 ns

T_2 = 95 - 130 ns

J-K Binary Element

T_3 = 125 ns (max)

T_4 = 150 ns (max)

J-K Binary Element

Figure 23 Timing Delays

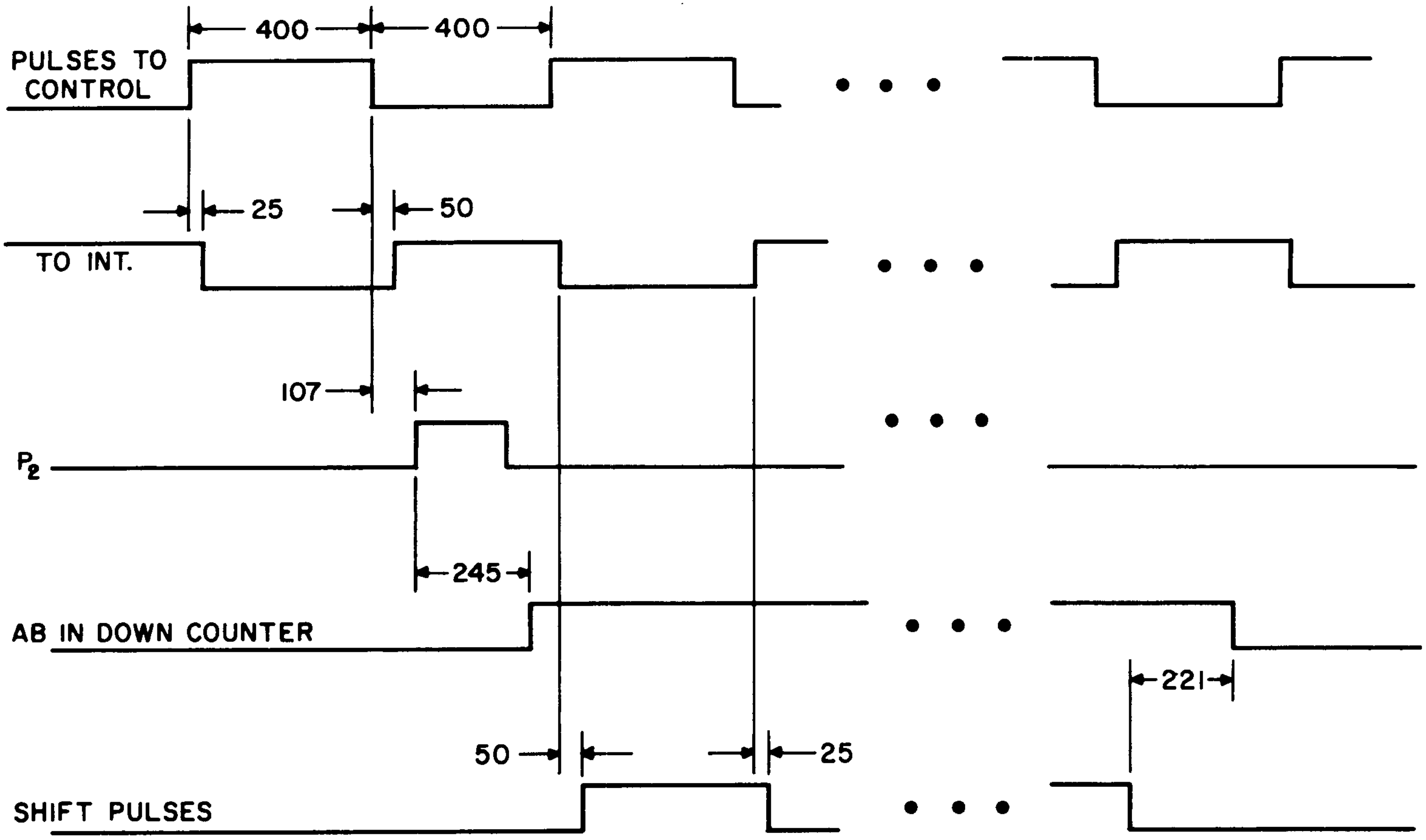


Figure 24 Timing Diagram

APPENDIX IV

EXPERIMENTAL RESULTS

Problem: Solve the differential equation $\dot{y} = y$. Figure 25 shows the schematic of the solution.

The solution to this equation is $y = e^x$. In the results to follow, the initial conditions $(x_0, y_0) = (0, 1)$ were used.

Case 1

$$S_x = 10, \quad 0 < x \leq 1, \quad M = 2$$

$$S_z = S_x - M = 8$$

$$S_y = S_z = 8$$

$$IL = S_y + M = 10,$$

These conditions utilize the integrator to its greatest accuracy for calculating the value of e. The integration process will, in effect, divide the inveral $0 < x \leq 1$ into $2^{10} = 1024\ \Delta x$ increments and sum the value of y at each Δx. The mode is interpolative for best results. Since in the generation of an exponential the early values of y greatly influence its later values, the value of y produced by the integrator is dependent upon the initial Δy.

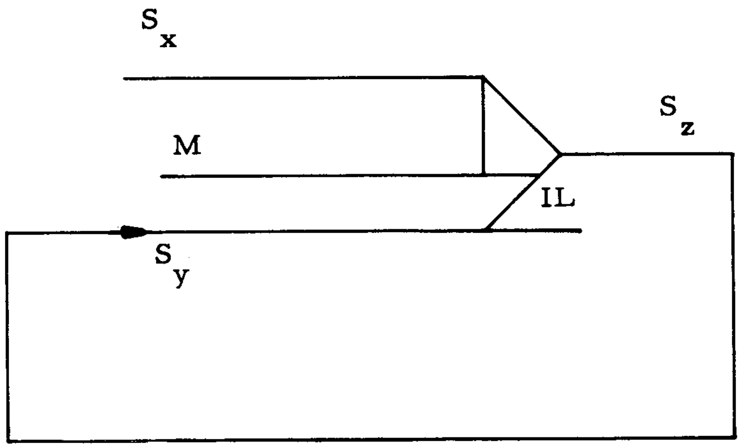


Figure 25 Schematic for $y = \dot{y}$

The initial Δy is known only if the sign of the slope $\frac{dy}{dx}$ is known at x_0. For the exponential, $(\frac{dy}{dx})_0 = +$. Therefore, a better result should be obtained if $(\Delta y)_0 = +1$. The result in Table 6 when $(\Delta y)_0 = +1$ is that

$$\begin{array}{rl} e = & 10.10111000 \\ & 2.00000 \\ & .50000 \\ & .12500 \\ & .06250 \\ & .03125 \\ \hline = & 2.71875 \end{array}$$

The actual value of e is 2.71828 . . . The calculated value is in error by + 0.00047. The least significant bit in the calculated value represents an increment of $2^{-8} = .0039...$ Hence this result is the closest possible value to e which is expressible in 10 information bits.

Cases 2 - 9

The maximum y is varied from $M = 3$ through $M = 10$ although the integration is only to $x = 1$ ($y = e$).

Cases 10 - 18

Same as cases 1 - 9 except that the mode is extrapolative.

Case 19

$$S_x = 9, \quad 0 < x \leq 1, \quad M = 2$$

$$S_z = S_x - M = 7$$

$$S_y = S_z = 7$$

$$IL = S_y + M = 9$$

Here the integrator length was reduced to 9 since the number of Δx between 0 and 1 was reduced from 1024 to 512. The mode is interpolative.

Cases 20 - 22

M varies from 3 through 5.

Cases 23 - 26

Same as cases 19 - 22 except that the mode is extrapolative.

Case 27

$$S_x = 10, \quad 0 < x \leq 2, \quad M = 3, \text{ Interpolative}$$

$$S_z = S_x - M = 7$$

$S_y = S_z = 7$

$IL = S_y + M = 10$

This is the same as Case 1 except that x is permitted to go to 2. Hence, the result is the value of e^2. ($e^2 = 7.3891$.)

$$\begin{array}{lrl}
\text{For } (\Delta y)_0 = 1,\ e^2 = & 111.0110110 & \\
 & 7.000000 & \\
 & .250000 & \\
 & .125000 & \quad 7.4219 \\
 & .031250 & \quad \underline{7.3891} \\
 & \underline{.015625} & \text{error}=0.0328 \\
 & 7.421875 &
\end{array}$$

$$\begin{array}{lrl}
\text{For } (\Delta y)_0 = 0,\ e^2 = & 111.0100110 & \\
 & 7.000000 & \\
 & .250000 & \quad 7.3891 \\
 & .031250 & \quad \underline{7.2969} \\
 & \underline{.015625} & \text{error}=-0.0922 \\
 & 7.296875 &
\end{array}$$

Cases 28, 29

Same as case 27 except $M = 4, 5$.

Case 30

$M = 5$, $0 < x \leq 3$, $S_x = 10$, Interpolative

$$\begin{array}{rr} e^3 = & 10100.10100 \\ & 20.000 \\ & .500 \\ & \underline{.125} \end{array}$$

Calculated e^3 = 20.625

Correct e^3 = 20.086

Cases 31 - 34

Same as cases 27 - 30 except that the mode is extrapolative.

Cases 35 - 37

$S_x = 9, \quad 0 < x \leq 2, \quad M = 3, 4, 5$

Interpolative Mode

Cases 38 - 40

Same as cases 35 - 37 except that the mode is extrapolative.

Case 41

$S_x = 10 \quad 1 < x \leq 2, \quad M = 3,$ Interpolative.

Case 27 produced the value of e^2 starting from the initial conditions $(x, y) = (0, 1)$. Here the inserted initial conditions are $(x, y) = (1, e)$.

$$
\begin{array}{lcrl}
e^2 & = & 111.0110010 & \text{for } (\Delta y)_0 = 1 \\
 & & 7.000000 & \\
 & & .250000 & \\
 & & .125000 & \\
 & & \underline{.015625} & \\
\text{Calculated } e^2 & = & 7.390625 & \\
\text{Correct } e^2 & = & \underline{7.3891} & \\
\text{error} & = & .0015 &
\end{array}
$$

This result is the closest possible value to e^2 which is expressible in 10 information bits.

TABLE 6

EXPERIMENTAL RESULTS

Case No.	$(\Delta y)_0 = 0$	$(\Delta y)_0 = 1$
1	10.10110100	10.10111000
2	010.1011000	010.1011110
3	0010.101010	0010.110000
4	00010.10100	00010.11010
5	000010.1000	000010.1110
6	0000010.100	0000011.000
7	00000010.00	00000011.10
8	000000001.0	000000100.0
9	0000000001.	0000000101.
10	10.10110100	10.10111010
11	010.1011000	010.1011110
12	0010.101100	0010.110000
13	00010.10100	00010.11010
14	000010.1000	000010.1110
15	0000010.100	0000011.000
16	00000010.00	00000011.10
17	000000001.0	000000100.0
18	0000000001.	0000000101.
19	10.1011000	10.1011100
20	010.101010	010.110000

21	0010.10100	0010.11010
22	00010.1000	00010.1110
23	10.1011000	10.1011110
24	010.101010	010.110000
25	0010.10100	0010.11010
26	00010.1000	00010.1110
27	111.0100110	111.0110110
28	0111.010000	0111.011110
29	00111.00100	00111.10010
30	10011.01100	10100.10100
31	111.0101000	111.0111000
32	0111.010000	0111.100000
33	00111.00100	00111.10100
34	10011.01110	10100.10110
35	111.001110	111.011100
36	0111.01000	0111.10010
37	00110.1110	00111.1100
38	111.010000	111.011110
39	0111.00100	0111.10010
40	00110.1110	00111.1100
41	111.0101100	111.0110010

If Δx is made - 1 rather than + 1, the limits of integration may be reversed. Therefore, for each of the above cases when Δx is changed from + 1 to - 1 after the forward integration, the reverse integration will return the integrator to its initial condition, (x_0 , y_0).

When Δx is set at - 1 and the initial condition is $(x_0, y_0) = (0, 1)$, then the following results are produced.

$$\begin{aligned}
e^{-1} &= 0.010111100 \\
e^{-2} &= 0.001000110 \\
e^{-3} &= 0.000011010 \\
e^{-4} &= 0.000001010 \\
e^{-5} &= 0.000000100 \\
e^{-6} &= 0.000000010 \\
e^{-7} &= 0.000000000
\end{aligned}$$

$$\begin{array}{lcr}
 & e^{-1} = & 0.010111100 \\
 & & .2500000 \\
 & & .0625000 \\
 & & .0312500 \\
 & & .0156250 \\
 & & \underline{.0078125} \\
\text{Calculated } e^{-1} & = & .3671865 \\
\text{Correct } \quad e^{-1} & = & .3678794 \\
 & & \underline{.3671875} \\
\text{error} & = & .0006919
\end{array}$$

The least significant bit is $2^{-9} = 0.001953 \ldots$ Hence this result is the closest possible value to e^{-1} which is expressible in 10 information bits.

The above values of e^{x} for $0 > x \geq -7$ were produced regardless of mode or $(\Delta y)_0$. The indifference to $(\Delta y)_0$ is because in the negative direction the later values of e^{x} do not depend so heavily on the early values. The mode has no effect since in the interval $-7 \leq x < 0$, the difference between y_i and y_{i+1} is insignificant to 9 binary places.

BIBLIOGRAPHY

1. Bellomy, F., "Digital Differential Analyzer", Electro-Technology, April, 1965.

2. Braun, Edward L., Digital Computer Design, Academic Press, New York, 1963.

3. Forbes, George F., Digital Differential Analyzers, George F. Gorbes, 10117 Bartee Avenue, Pacoima, California, 1957.

4. Mayorov, F. V., Electronic Digital Integrating Computers, Iliffe Books Ltd., London, 1964.

5. Shileiko, A. V., Digital Differential Analysers, The Macmillan Company, New York, 1964.