CASE INSTITUTE OF TECHNOLOGY

CLEVELAND, OHIO

ENGINEERING DESIGN CENTER

This Research Was Sponsored by

THE NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

A Digital Controller Employing Truncated Logarithmic Quantization

Report No. EDC 1-64-28

by

John B. Peatman

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

Digital
Systems
Laboratory

August, 1964

ABSTRACT

The function of this thesis is to examine in detail the logic involved in the design of a digital controller. In order to reduce the logic requirements, a unique method of information quantization is utilized. This quantization scheme, called truncated logarithmic quantization leads to reduced information storage requirements. Also, multiplication by a constant can be handled as a natural consequence of using this quantizing scheme, without requiring any additional logic. The adequacy of this quantizing scheme in a control application is first simulated on a general purpose digital computer and then experimentally verified on a two-capacity liquid level system.

Design considerations are discussed thoroughly. A prototype has been constructed in which the design is divided into eight functional blocks, each of which is implemented on a 4" x 6" printed circuit board. To achieve this degree of simplification, several special purpose digital logic elements have been utilized.

author

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

For the sources and the destinations of all signals between the eight functional blocks of the controller, see Figure A. 2. 1 on page 111.

Begin Operate Mode	A signal indicating that the preparation mode has been completed
$c(t)$	System output before quantizing
c_n	System output at the n^{th} sampling instant
$\dot{c}_n$	Derivative of c (t) at the n^{th} sampling instant
$C(s)$	Laplace transform of c (t)
$e(t)$	Set point minus system output, c (t)
E_n	Error at the n^{th} sampling instant
E_n^*	The decoding of $\langle \text{LOG } E_n \rangle$
f	Rate of output pulse generator
$G(s)$	Transfer function of the plant
K	Defined by equation 2-5
K_E	Defined by equation 2-5
$K_{\Delta E}$	Defined by equation 2-5
K_{GA}	Defined by Figure 2. 15

K_I	Defined by equation 2-14
K_p	Defined by equation A. 3-12
K_P	Defined by equation 2-14
Log. Output	Defined by Figure 2. 11
$\langle \text{Log } E_n \rangle$	Truncated logarithmic encoding of E_n
m (t)	Defined by equation 2-13
m_n	Position of the stepping motor after the n^{th} sampling interval; in units of steps
M (s)	Laplace transform of m (t)
Overflow	This signifies that the Logarithmic Pulse Generator has counted 1024 pulses
P	Sampling pulse
r	System reference input; the set point
s	Laplace transform variable
S_i	The i^{th} bit of the Serial Set Point (S_{10} is the most significant bit; it is also the first bit in time)
$SIGN_n$	Sign of the error, E_n, at the n^{th} sampling instant. $SIGN_n$ = -12 volts if the error is negative; that is, if $r - c_n < 0$
t	The time variable
T	Sampling period
T_{dead}	Defined by equation 4-11

t_i	Anticoincident pulses in Output Gating
T_{lag}	Defined by equation 4-11
T_p	Defined by equation A. 3-12
T_s	Negative pulses corresponding to the ten 0's and 1's in Θ_o
Tenth Pulse	A (-) level until the tenth pulse arrives from the transducer, when it becomes a (+) level
Δm_n	Digital controller output at the n^{th} sampling interval
$\Theta_i^{0's}$	Pulses corresponding to the zero's in the serial set point
Θ_o	Serial output from transducer to controller. One's are negative pulses while zero's are positive pulses
$\Theta_o^{0's}$	Negative pulses corresponding to the zero's in Θ_o
$\Theta_o^{1's}$	Negative pulses corresponding to the one's in Θ_o
$(0000 \rightarrow 1111)_n$	A (-)/(+) transition when the E_n Counter counts down from 0000 to 1111
(-) level	-12 volts
(+) level	0 volts
(-)/(+) transition	A transition from -12 volts to 0 volts
$\langle\ \rangle$	Greatest integer in

CHAPTER I

DIGITAL PROCESS CONTROL

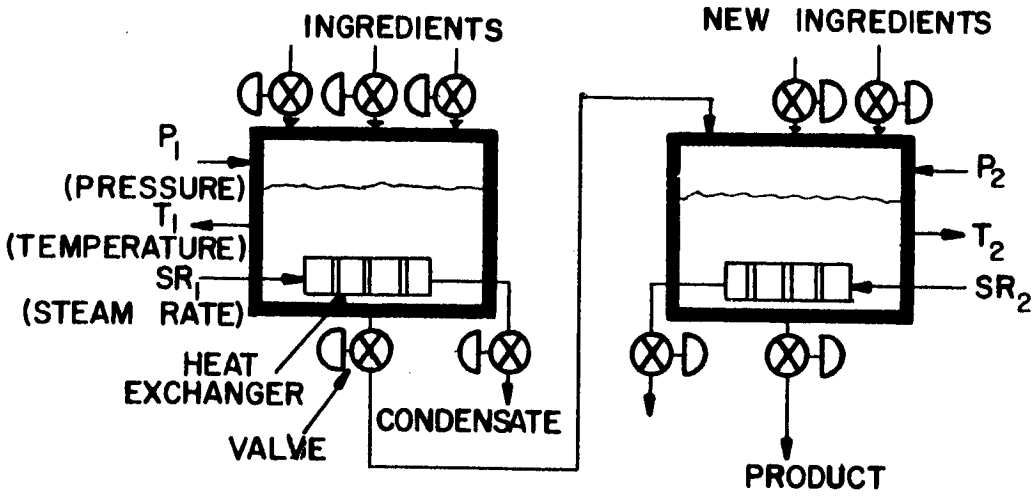
Introduction

There are several justifications for investigating process control problems with a digital approach in mind. To appreciate these it is well to review the approach to process control which has generally been taken in the past, and to note the assumptions, limitations, and also the advantages involved. This is the subject of the next section. Following this the ultimate capability of computer control is discussed. The intention of these two sections is to bracket the capability of the digital controller approach studied in this thesis between the capability inherent in classical process control and that of computer process control. Hence the last section of this chapter will summarize the position taken here.

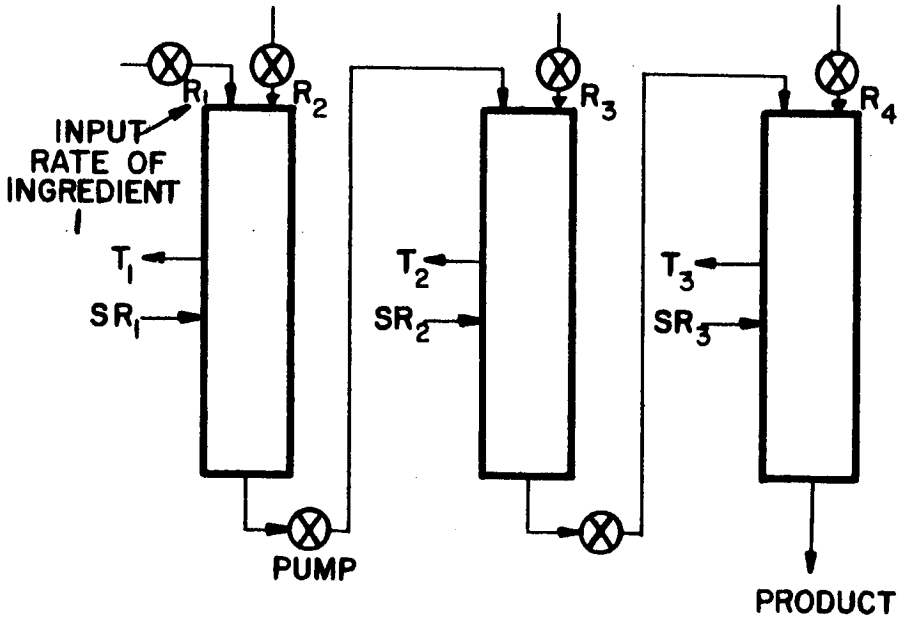
Classical Process Control

In the field of process control, the classical approach used by chemical and instrumentation engineers generally has begun with the derivation of a sequence of reactions which will

lead to the desired product.[1] Then a physical plant is set up in which the individual reactions can be achieved simply by maintaining process variables (temperature, pressure, concentration of ingredients, etc.) at predetermined set points in individual subplants. The sequencing of the individual reactions can be achieved by transporting the process ingredients between several such subplants. In a batch process which requires this sequencing, the ingredients are transferred from one reactor to the next, with each reactor having its own carefully controlled group of set points. A continuous process achieves the same goal by passing the ingredients through a continuous channel in which the set point conditions vary along the channel but not with time. These two approaches are depicted in Figure 1.1. The point to note is that in these classical approaches the chemical engineers take pains to find reactions which can be achieved by maintaining process variables at predetermined set points. The instrumentation engineer's contribution consists of monitoring the process variables and trying to adjust these toward the desired set points by manipulating control variables (flow rate of coolant or fuel, ingredients, etc.). This must be done in spite of



a) A TWO-STAGE BATCH PROCESS



b) A CONTINUOUS PROCESS

FIGURE I-1 BATCH AND CONTINUOUS PROCESSES

process dynamics which can make the cause and effect relationship between the manipulated variables and the process variables a complex function of time. Thus the instrumentation engineer has classically been confronted with the problem of controlling a dynamic process by measuring some variables and using these measurements to control other variables.

Generally the chemical engineer can help the instrumentation engineer to the extent that he can design the process reactions so that each important process variable is mainly dependent on only one manipulated variable. When this decoupling of the variables is achieved sufficiently well, then the overall control problem can be reduced to that of controlling set points in many more or less independent control loops.

Classically the emphasis in the instrumentation industry has been to provide controllers which assume a decoupled system and which have as their main function the control of a single loop. Such a controller must perform its function in the presence of the dynamic characteristics of the reaction. In addition it must operate in spite of changing characteristics of the process as the reaction proceeds, incomplete knowledge

of the reaction kinetics, and the effects of the control going on outside of its loop. In this classical approach these effects can only be labelled as "ignorance of the process" and "disturbances to the loop," and the controller parameters adjusted to yield satisfactory control in spite of these.

It should be apparent that the acceptability of this approach to process control depends heavily on having a chemical engineer who can design a sufficiently decoupled system in the first place.

Before closing this section, it is proper to point out one of the features of the classical approach which will always endear it to the process industries. If one of the control loops should malfunction, there is the possibility of controlling this loop manually, possibly with somewhat inferior control, and thus limping along until the malfunction is repaired. Under such circumstances and with suitable planning, it may be proper to expect no down time for the process as a whole due to instrumentation malfunctions.

Computer Process Control

The implication of the term "computer process control" which is meant to be left with the reader is a system of

instrumentation in which

1. the control problem can be treated as a whole rather than as the net effect of controlling many single loops,
2. the identification of the plant characteristics can be carried out, at least partially, during system operation and this information used to improve control,
3. the computing capability is sufficient to carry out any degree of optimizing control which can be conceived by the instrumentation engineer together with the chemical engineer.

Such a system provides the possibility of lifting the constraint from the chemical engineer for providing a decoupled process. However this will be an advantage only if the chemical engineer can use the extra latitude given him to come up with less expensive reactions or a less expensive plant layout.

The main effect on process performance should be the tighter control which can be achieved. Aside from those improvements resulting from identification of the plant and from

optimizing control, there is also the gain resulting simply because of the integrated point of view seen by a central controller. No longer is it necessary for there to be autonomous loops which cannot distinquish true disturbances from those changes wrought by control in other loops.

As the future unfolds for computer process control, the cost factor will surely come down from today's $1,000/loop at the 50-loop level.[2] However just as today the success of classical process control depends upon the ability of the chemical engineer to design a decoupled system, so tomorrow the success of computer process control will depend on the ability of the instrumentation engineer to build reliable equipment and to organize this equipment so that there can be various types of instrumentation malfunctions without corresponding plant shutdowns. That is, there should be designed into the instrumentation the capability of limping along in spite of temporary instrumentation malfunctions.

The Digital Controller Approach

The view taken here is that there always will be a place for classical process control. Such should be the case for those processes in which a decoupled approach cannot be

appreciably undercut in cost by any other approach. It should also be the case where the size of the process does not warrant the cost of computer control. Finally there will probably always be processes where the simplicity of the solution to the reliability problem justifies the use of the classical approach.

The digital controller approach is an attempt to continue fundamentally in the classical tradition. Consequently any proposed designs for a controller plus transducer for input and actuator for output must compete with existing systems of the basis of cost, performance, and reliability.

To meet this challenge, a digital controller has the inherent advantage of requiring no special tooling; that is, the techniques involved in construction are those of the electronics industry. Today this means the construction of printed circuit boards for component layout, and either hand wiring, machine wiring, or printed circuit wiring for interconnection of these component boards. In the future this will also include the techniques of microminiaturization.

Another feature of the digital approach is the opportunity for providing only one controller to handle all control

problems which differ only in amplitude and time scaling. Thus the input transducers can be monitoring either a small range or a large range of temperature, pressure, flow or other process variable. The speed of the process reaction can be measured in seconds, minutes, or even hours. The controlled variable may have to be controllable to within 10%, 1% or 0.1%. In all these instances the digital controller can provide the necessary time and amplitude scaling with no loss of accuracy and perhaps with no appreciable difference in equipment. To illustrate the simplest example of this, consider a digital controller which samples the information available from the input transducer. If the sampling interval can be easily set equal to one-tenth (or any other fraction) of the dominant time constant of the process, then it makes no difference to the dynamic characteristics of the control loop whether this dominant time constant is ten seconds or three hours. The time scaling of the controller, in order to fit it to the process, consists simply of altering the sampling interval.

One final point of advantage is related to the future and to computer process control. There are processes in which it is advantageous to have a digital computer available to

determine the best product mix in order to maximize profits in the present market. For example in the petroleum industry, one distilling column is used to produce many products. Furthermore the ratio of gasoline to diesel fuel and to crude oil which maximizes profits fluctuates with market prices from day to day. But since these product ratios depend on the set points of the process, it will be useful to have digital controllers which can accept the new set points directly from the digital computer.

In conclusion, three advantages of the digital controller approach are lack of special tooling costs, versatility in operation, and simplicity of use with a supervisory digital computer in a large, multiloop process. Work in this area of single loop direct digital control has been actively pursued by the author's advisor, Professor H. W. Mergler, since October 1962. The purpose of this thesis is to investigate how a specific method of quantizing information can be used to reduce the logic required for a digital controller and to investigate the effect of this method of quantization upon the performance of the controller.

CHAPTER II

TRUNCATED LOGARITHMIC QUANTIZATION

Introduction

The distinquishing characteristics of a single loop digital control system can be explained with the help of Figure 2.1. Here the digital set point is simply a number. The purpose of the control system is to drive the plant or process until its output, as measured by the digital transducer, is equal to the set point. The subtractor generates a number corresponding to the difference between the set point and the output. This number, which represents the error in the output relative to the set point, is designated the system error. The digital controller has available to it this error at the present instant as well as the error at any previous instants which it has taken the trouble to store away. By using this error information the digital controller generates a digital control signal which will tend to drive the plant toward zero error in spite of the dynamic characteristics of the plant and in spite of disturbances acting on the plant.

Figure 2.2 shows a more specific digital control system

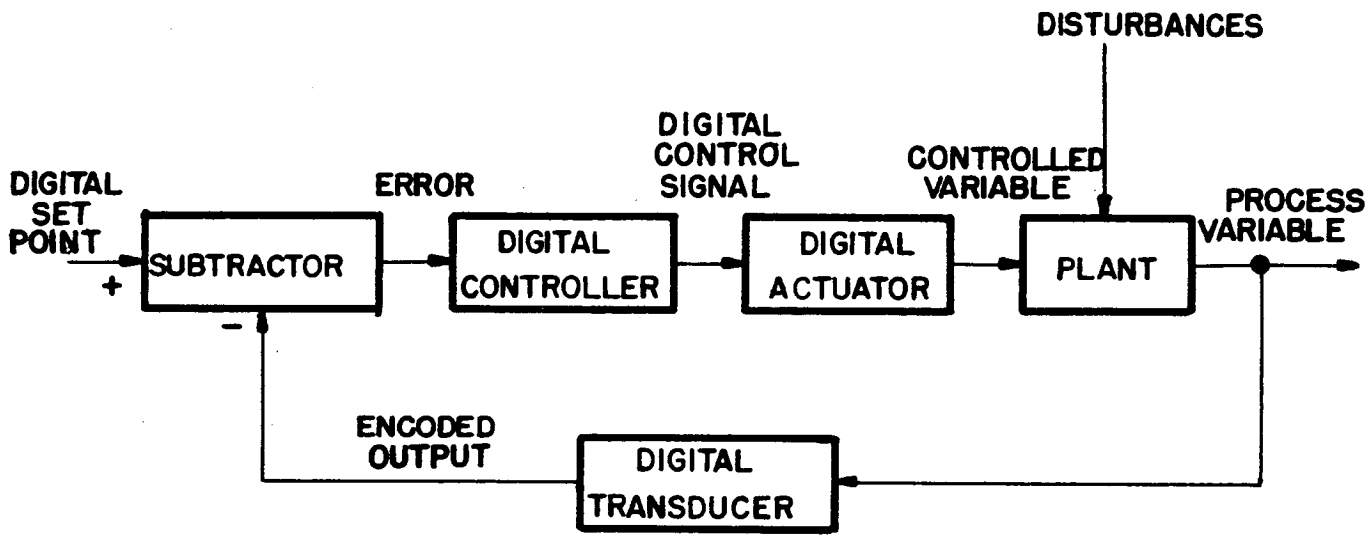


FIGURE 2.1 A DIGITAL CONTROL SYSTEM

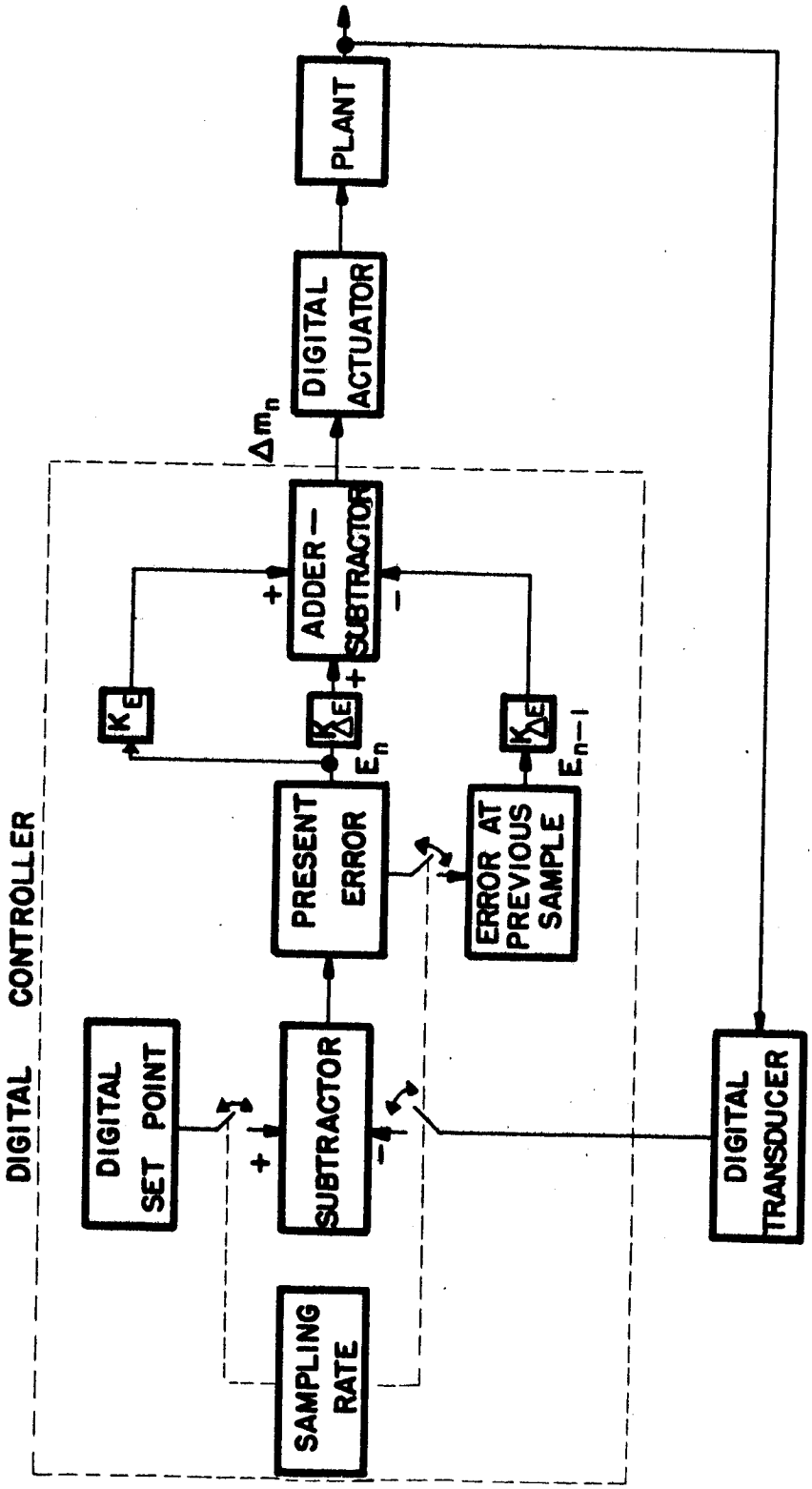


FIGURE 2.2 A SPECIFIC CONTROL SYSTEM

similar to the one studied in this thesis. Note that the information from the digital transducer is sampled periodically. This characteristic is not implied by the system being a digital system, but it is a helpful characteristic for the control of a dynamic plant; that is, a plant in which the process variable does not respond immediately to changes in the controlled variable. The digital controller has control of this sampling process as indicated by the dotted line.

The digital control signal during the n^{th} sampling interval, Δm_n, is a function of E_n, the error at the n^{th} sampling instant, and of E_{n-1}, the error at the $n-1^{th}$ sampling instant. The equation implied by this figure is

$$\Delta m_n = K_E E_n + K_{\Delta E} (E_n - E_{n-1}) \qquad (2-1)$$

Note that to implement this equation, the controller must carry out four functions:

1. subtraction between set point and plant output,
2. storage of the previous error,
3. multiplication by a constant,
4. addition and subtraction of signed numbers.

The quantizing technique presented in the next section is designed to reduce the storage requirements of the controller

and to facilitate multiplication by a constant. In doing this the configuration of the controller is changed to that of Figure 2.3. Here the term $\langle \text{LOG } E_n \rangle$ is used to indicate the truncated logarithmic encoding of the present error. Since this encoding is logarithmic, the desired multiplication of the error, E_n, by a constant, K_E, is indicated here as a summation with LOG K_E, the logarithm of K_E. The decoder now carries out an exact antilog operation, but since $\langle \text{LOG } E_n \rangle$ is not truly the logarithm of E_n, the result of this decoding is not truly $K_E E_n$, and so it is denoted as $K_E E_n^*$.

Definition of Truncated Logarithmic Quantization

Truncated logarithmic quantization is most easily defined by a table comparing typical inputs and outputs to a truncated logarithmic encoder as shown in Figure 2.4a. This table is shown in Figure 2.4b. The binary numbers which are expressed with just zeros and ones can be converted to the more familiar decimal form shown by using the weighting equation:

$$E_n \text{ decimal} = 2^5 A_6 + 2^4 A_5 + 2^3 A_4 + 2^2 A_3 + 2^1 A_2 + 2^0 A_1 \qquad (2\text{-}2)$$

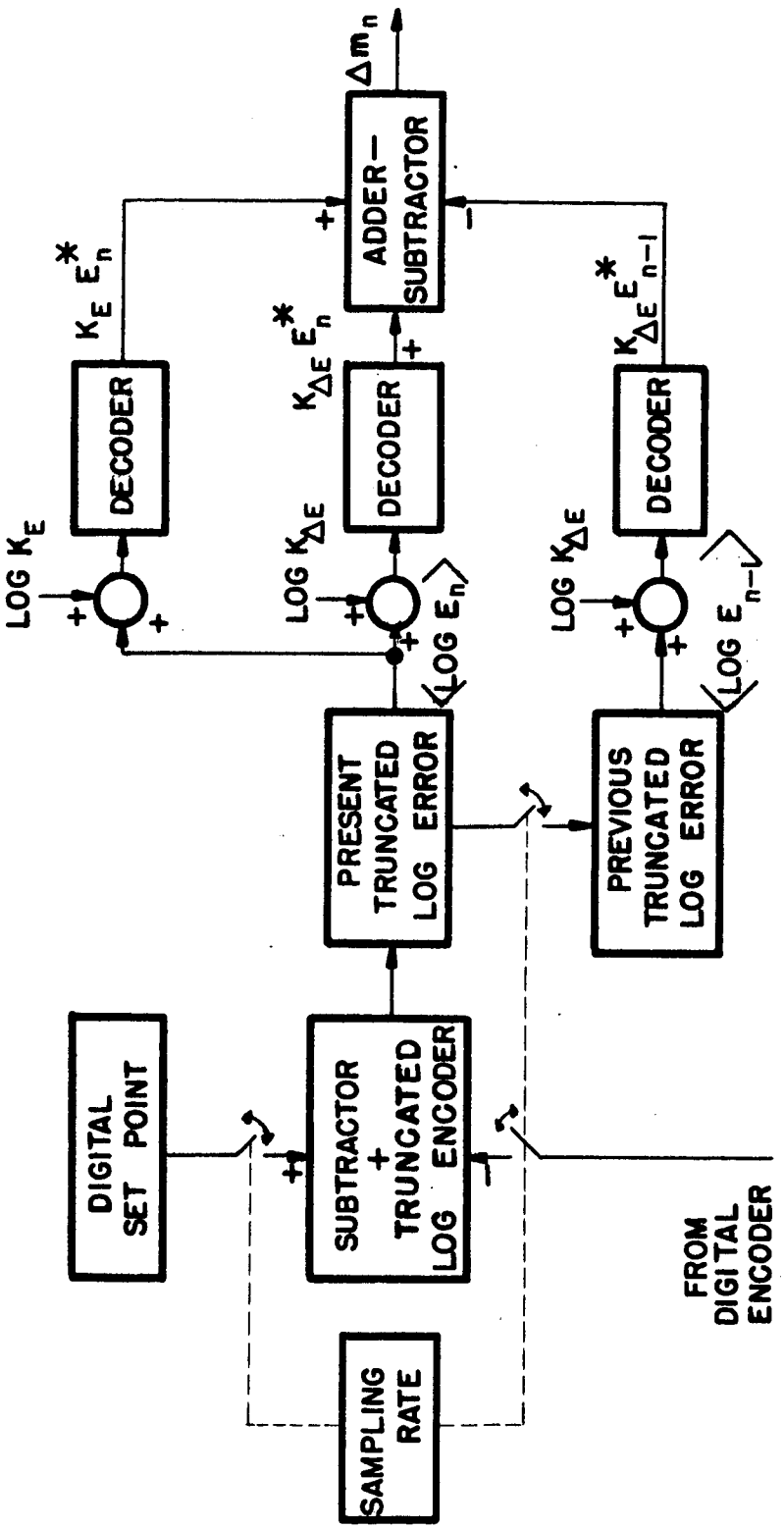
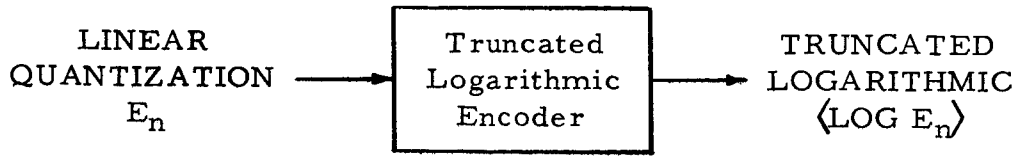


FIGURE 2.3 A CONTROLLER EMPLOYING TRUNCATED LOGARITHMIC QUANTIZATION



a) A Truncated Logarithmic Encoder

Binary	E_n Decimal	$\langle LOG\ E_n \rangle$ Decimal	Binary
000000	0	0	000
000001	1	1	001
000010	2	2	010
000011	3	2	010
000100	4	3	011
000101	5	3	011
000110	6	3	011
000111	7	3	011
001000	8	4	100
001111	15	4	100
010000	16	5	101
011111	31	5	101
100000	32	6	110
-000111	-7	-3	-011
-001000	-8	-4	-100

b) Typical Input-Output Pairs

FIGURE 2.4 TRUNCATED LOGARITHMIC QUANTIZATION

The powers of two in this equation weight the different A_i's where A_i is the i^{th} bit of the binary number (A_i is a zero or a one). Thus for example,

$$000110 \text{ binary} = 32 \times 0 + 16 \times 0 + 8 \times 0 + 4 \times 1$$
$$+ 2 \times 1 + 1 \times 0 = 6 \text{ decimal}$$

The key to the truncated logarithmic encoding scheme lies in noting that $\langle \text{LOG } E_n \rangle$ is just a number equal to the position of the most significant "one" in the binary representation of E_n. For example, the most significant "one" in the binary number 000110 lies in the third position from the right. Hence it is encoded as binary 011 (which is equal to decimal three).

Note that this encoding scheme is not one-to-one; e.g., there are three other numbers besides 000110 which are encoded as 011. This will result in a savings in the storage requirements since, for example, any fifteen bit number plus sign can be quantized into a four bit number plus sign. Of course, information is lost in the process, but it is the contention of the author that for many control purposes the amount of information remaining is sufficient to afford good control. This will be investigated more thoroughly toward

the end of the chapter.

The name "truncated logarithmic quantization" is derived from the equation for this encoding scheme:

$$\langle \text{LOG } E_n \rangle = \begin{cases} 0 & \text{If } E_n = 0 \\ (\text{SIGN } E_n) \cdot \langle 1 + \text{LOG}_2 |E_n| \rangle & \text{If } E_n \neq 0 \end{cases} \quad (2\text{-}3)$$

where $\text{LOG}_2 |E_n|$ is the logarithm to the base 2 of $|E_n|$. This will in general be an irrational number if E_n is not a power of two. The brackets, $\langle\ \rangle$, are meant to indicate truncation, or cutting off, at the decimal point, to leave the greatest included integer. For example, $\langle 3.9965 \rangle = 3$.

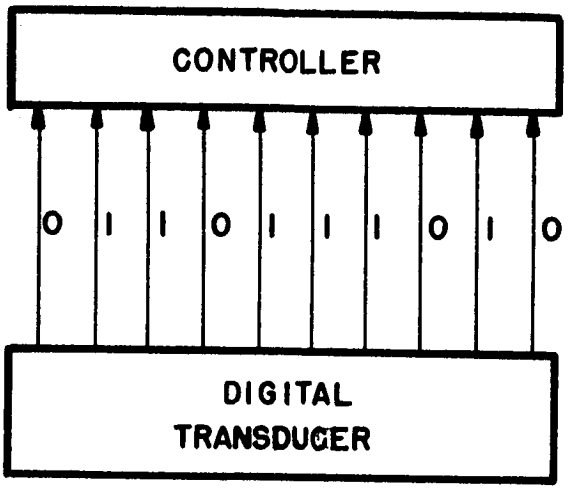
Encoding Algorithm

One way in which the truncated logarithmic quantization of the error can be formed has already been indicated. First the error can be formed using linear quantization and then it is only necessary to count over to the most significant "one" in this error. However in the implementation it might be convenient to be able to avoid having to form, and store, the error in its linearly quantized form.

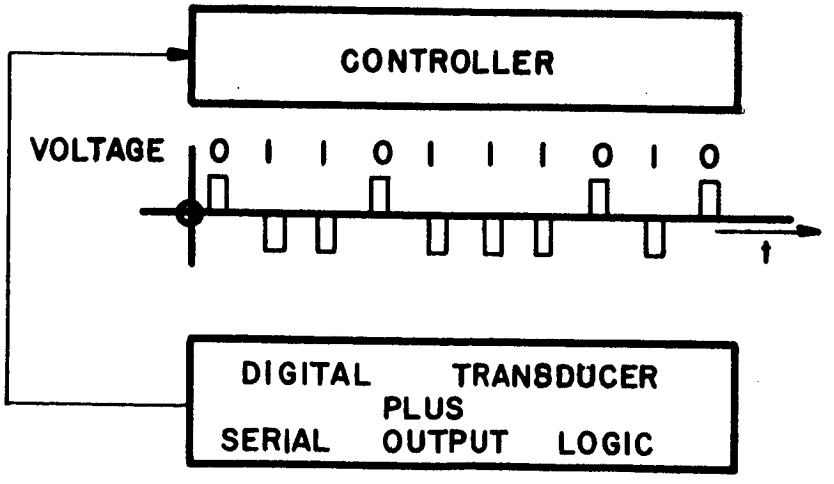
One instance where this would be convenient occurs in many process control applications where it is desirable to

have the controller located in a control room which might be quite remote from the digital transducer. If the digital information consists of a ten bit binary number, then it could be transferred to the controller from the digital transducer in parallel as in Figure 2.5a. This has the merit of simplicity; however it also requires stringing ten wires (plus a ground wire) between the transducer and the controller. An alternative approach is shown in Figure 2.5b in which only one wire is needed to transmit the same information; however it requires extra logic associated with the transducer to achieve the required time-sharing of the wire. With this serial transmission, only one bit of information will be available at a time in the controller for forming the truncated logarithmic error. This information could be stored to form again the complete number. On the other hand, this storage logic can be saved if the truncated logarithmic error can be formed "on the fly" during the serial reception of the information.

To illustrate the ideas needed for a serial encoder consider the examples of binary subtraction given in Figure 2.6. The problem is to determine where the most significant "one" of the difference will be if the minuend and the



(a) PARALLEL TRANSMISSION



(b) SERIAL TRANSMISSION

FIGURE 2.5 DIGITAL TRANMISSION OF INFORMATION

a)

```
        ↑   ↑
  0011 1 1 0110
- 0011 0 1 0011
-------------------
  0000 1 0 0011
```

b)

```
        ↑   ↓
  0011 1 1 0010
- 0011 0 1 0111
-------------------
  0000 0 1 1011
```

c)

```
      ↑↓↓ ↓   ↑
  0100 0 1 1010
- 0011 1 1 0110
-------------------
  0000 1 0 0100
```

d)

```
      ↑↓↓ ↓   ↓
  0100 0 1 0010
- 0011 1 1 1110
-------------------
  0000 0 1 0100
```

FIGURE 2.6 EXAMPLES OF BINARY SUBTRACTION

subtrahend are provided one bit at a time. Since the most significant "one" of the difference is desired, it is appropriate to consider the numbers as serial information available one bit at a time, most significant bit first. In Figure 2.6a, the first bits match and hence the first bit of the difference is zero. Similarly the second, third and fourth bits match, yielding zeros in the difference. The fifth bits are mismatched and note that the most significant bit of the difference occurs in this bit. Now look at Figure 2.6b. This is almost the same example as Figure 2.6a except that the eighth bits are mismatched in the opposite sense. Note that here the most significant "one" of the difference occurs in the sixth bit. Next look at Figure 2.6c and 2.6d and note that a mismatch followed immediately by mismatches of the opposite sense shift the most significant bit of the difference to the right. Again the exact position of the most significant "one" of the difference depends upon later mismatch conditions.

The boxes enclosing two bit positions in each example are intended to focus attention upon the two bit positions in which the most significant "one" of the difference must occur due to the first mismatch and to any immediately adjacent

mismatches of the opposite sense. To determine in which of these two positions the most significant "one" of the difference occurs, it is convenient to consider the subtraction process as the net result of two separate subtractions. Thus in each of the examples of Figure 2.6 consider the subtraction of the six bit numbers formed by omitting the four bits to the right of the boxes. In every case the difference is binary 10. Now the only way the last four bits, which were momentarily omitted, can affect this result is to "borrow one," in which case the difference will be reduced by one to binary 01. But the criterion for "borrowing one" is whether the sign of the subtraction of the last four bits is different from the sign of the subtraction of the first six bits. This in turn depends upon the sense of the most significant mismatch in the last four bits. The arrows of Figure 2.6 are used to indicate the sense of the important mismatches in each subtraction.

The complete serial encoding process is summarized in Figure 2.7. This figure describes the process in flow diagram form.

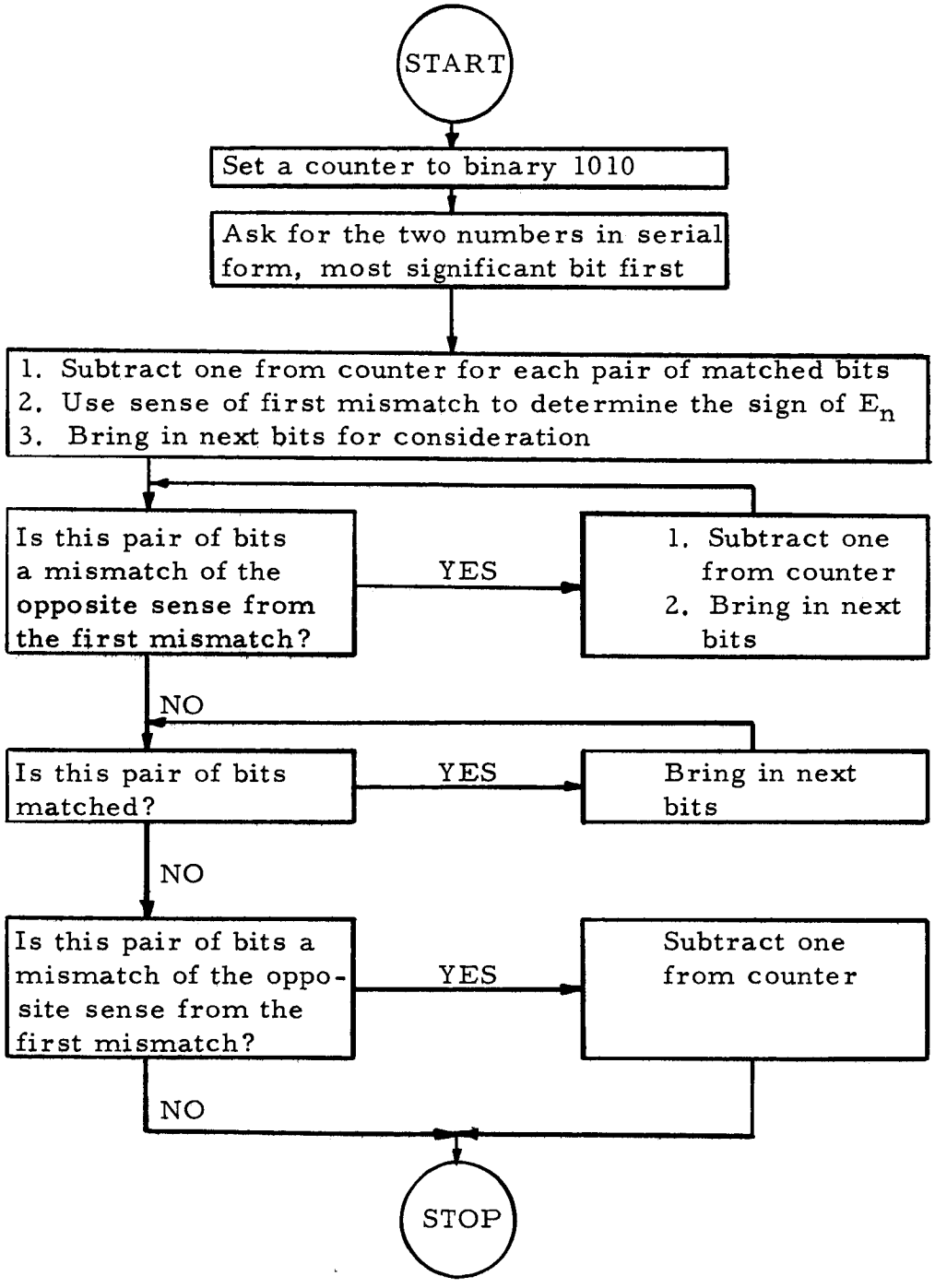


FIGURE 2.7 ENCODING ALGORITHM FOR TEN BIT NUMBERS

Decoding and Overall Operation

As was pointed out in Figure 2.3, it is necessary to decode numbers expressed in truncated logarithmic form in order to achieve quasi-linear operation. If the inverse of equation 2-3 is used, the results will be that of Figure 2.8. If this decoding operation is very involved, then the savings in hardware achieved through the merits of the encoding scheme could all be lost because of the decoding. Fortunately a method is possible in which one unit, a "Logarithmic Pulse Generator," can be utilized to carry out all three decoding operations as well as the required multiplications by constants. The purpose of this section is to describe the operation of the Logarithmic Pulse Generator and to indicate how it can be used to generate the required controller output, Δm_n.

The fundamental operation of the Logarithmic Pulse Generator is shown in Figure 2.9. Note that the input is a string of pulses which can be turned on at some time after the generator has been reset. To make use of this generator for decoding, consider the configuration of Figure 2.10. Assume that it is desired to decode $\langle \text{LOG } E_n \rangle = 4$. A proper decoding, as seen from the chart of Figure 2.8, is 8. Before switch A

$\langle \text{LOG } E_n \rangle$	E_n^*
0	0
1	1
2	2
3	4
4	8
5	16
-4	-8
-5	-16

FIGURE 2.8 DECODING EXAMPLES

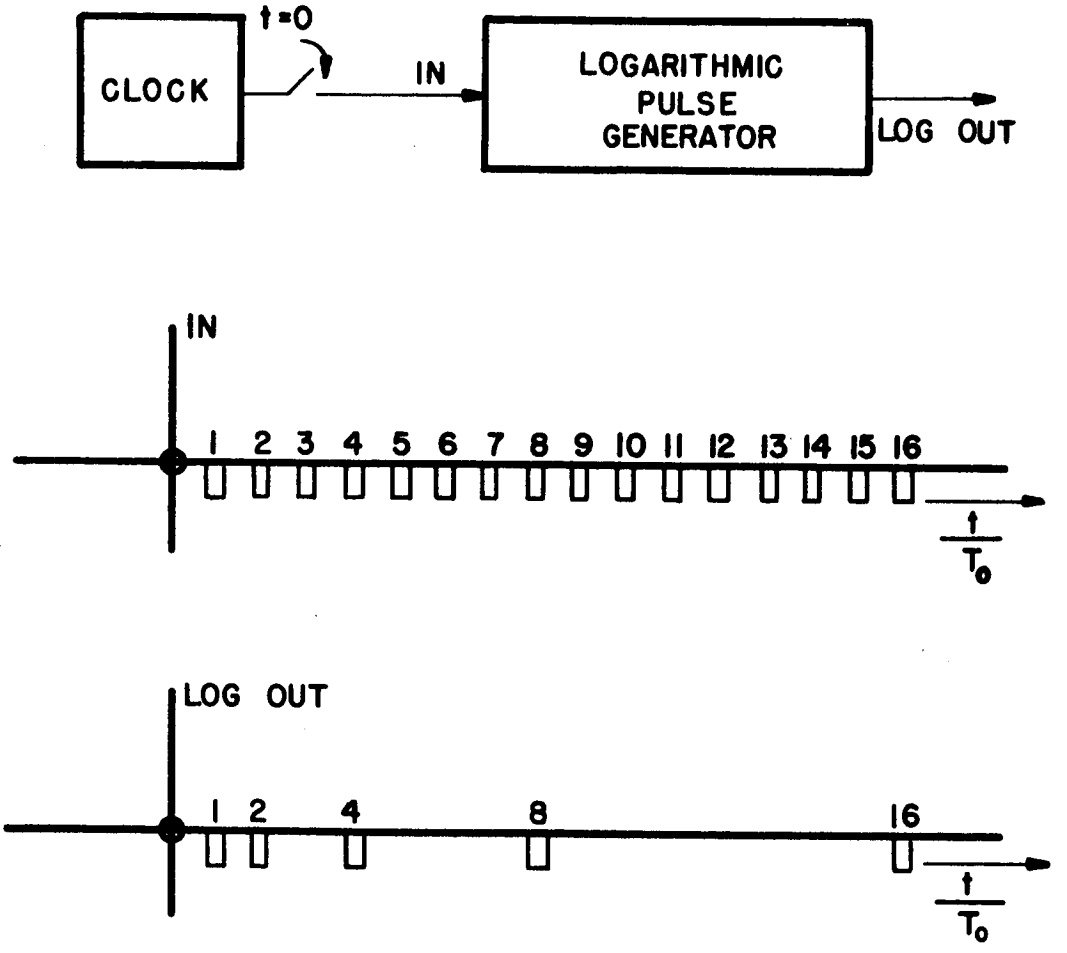


FIGURE 2.9 BASIC LOGARITHMIC PULSE GENERATOR

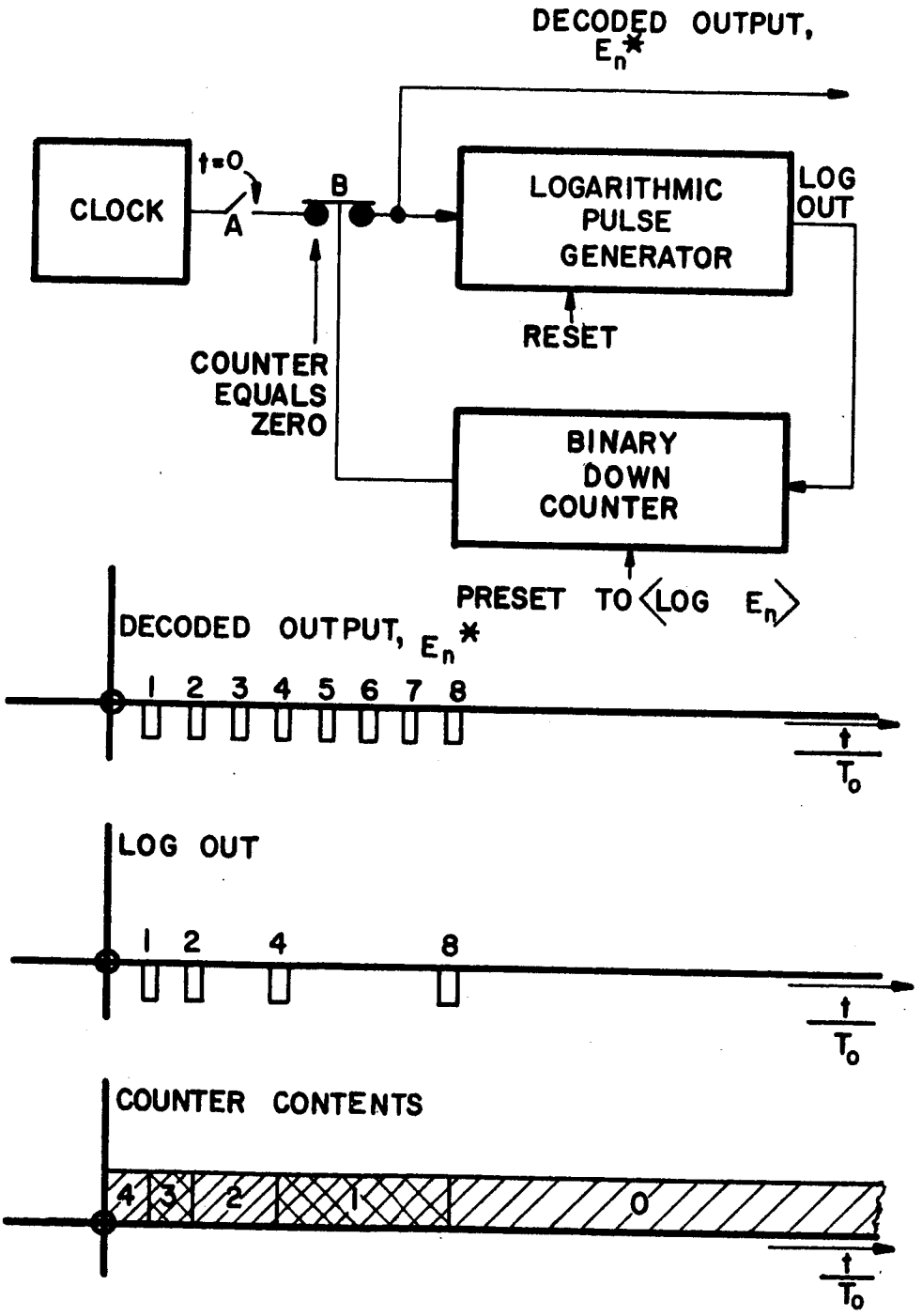


FIGURE 2.10 BASIC DECODING OPERATION

is closed at t = 0, the generator is reset and the down counter (a counter which counts down to zero) is preset to the desired $\langle \text{LOG } E_n \rangle = 4$. When switch A is closed at t = 0, the pulses which go into the Logarithmic Pulse Generator are also used as the DECODED OUTPUT, in pulse form. The LOG. OUTPUT pulses count the counter down until the contents of the counter is zero. This opens switch B, which in turn cuts off any more pulses from the clock to the DECODED OUTPUT or to the Logarithmic Pulse Generator.

At this point it might be questioned how the DECODED OUTPUT in pulse form can be used to produce Δm_n. For this purpose a stepping motor forms an ideal digital actuator. It gives an output rotation of its shaft which is proportional to the number of input pulses it receives. In addition, with very little supplementary logic, the stepping motor can accept pulses at an UP input which will step it in one direction, or it can accept pulses at a DOWN input which will step it in the opposite direction. Making use of this, the stepping motor can be used as an exact adder-subtractor; that is, if it receives 2018 UP pulses interspersed with 2015 DOWN pulses, then it will step UP a net of exactly 3 steps.

To get the scale factors, K_E and $K_{\Delta E}$, the Logarithmic Pulse Generator is provided with two other features. The first feature scales the input pulse rate to provide submultiples, $\frac{f}{R}$, of the input pulse rate as shown in Figure 2.11. This provides a simple way for multiplying E_n^* by an integral multiple of one-half. The other feature gives a way to multiply E_n^* by integral multiples of two. Consider the effect of inhibiting the first LOG. OUTPUT pulse on the operation of the system shown in Figure 2.10. Now the counter will not be counted down to zero until the sixteenth input pulse. Thus the number of output pulses has been multiplied by two. By inhibiting the first two LOG. OUTPUT pulses the DECODED OUTPUT will be multiplied by four. Figure 2.11 shows schematically an INHIBITING SWITCH which can inhibit from zero up to the first six LOG. OUTPUT pulses. Thus it can provide a multiplier of from unity to sixty-four.

These ideas are all tied together in Figure 2.12, which shows schematically the complete generation of the output. The block labelled ANTICOINCIDENCE insures that the stepping motor will not be asked to deal with two pulses arriving simultaneously. For simplicity, the output gating to account

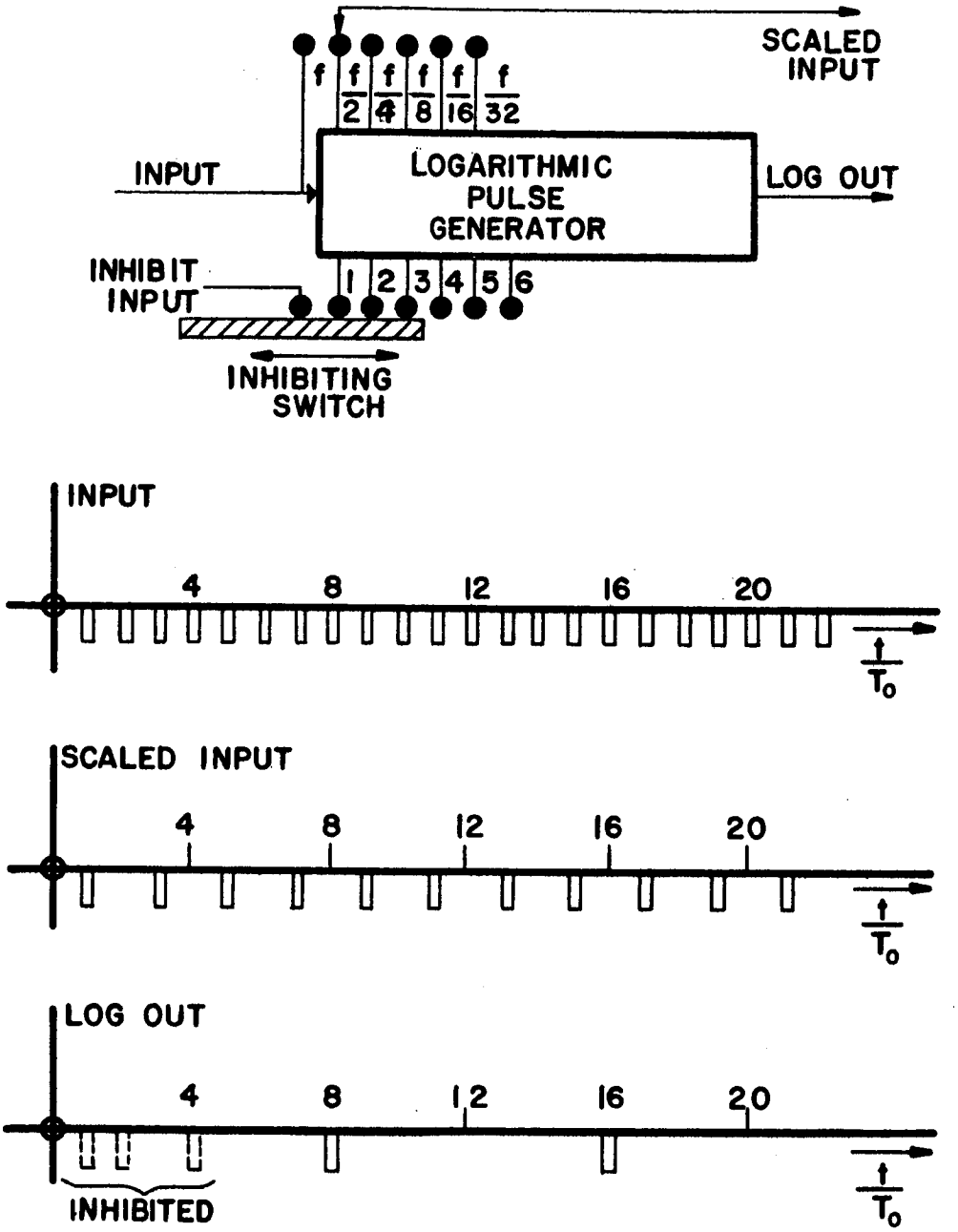


FIGURE 2.11 ACTUAL LOGARITHMIC PULSE GENERATOR

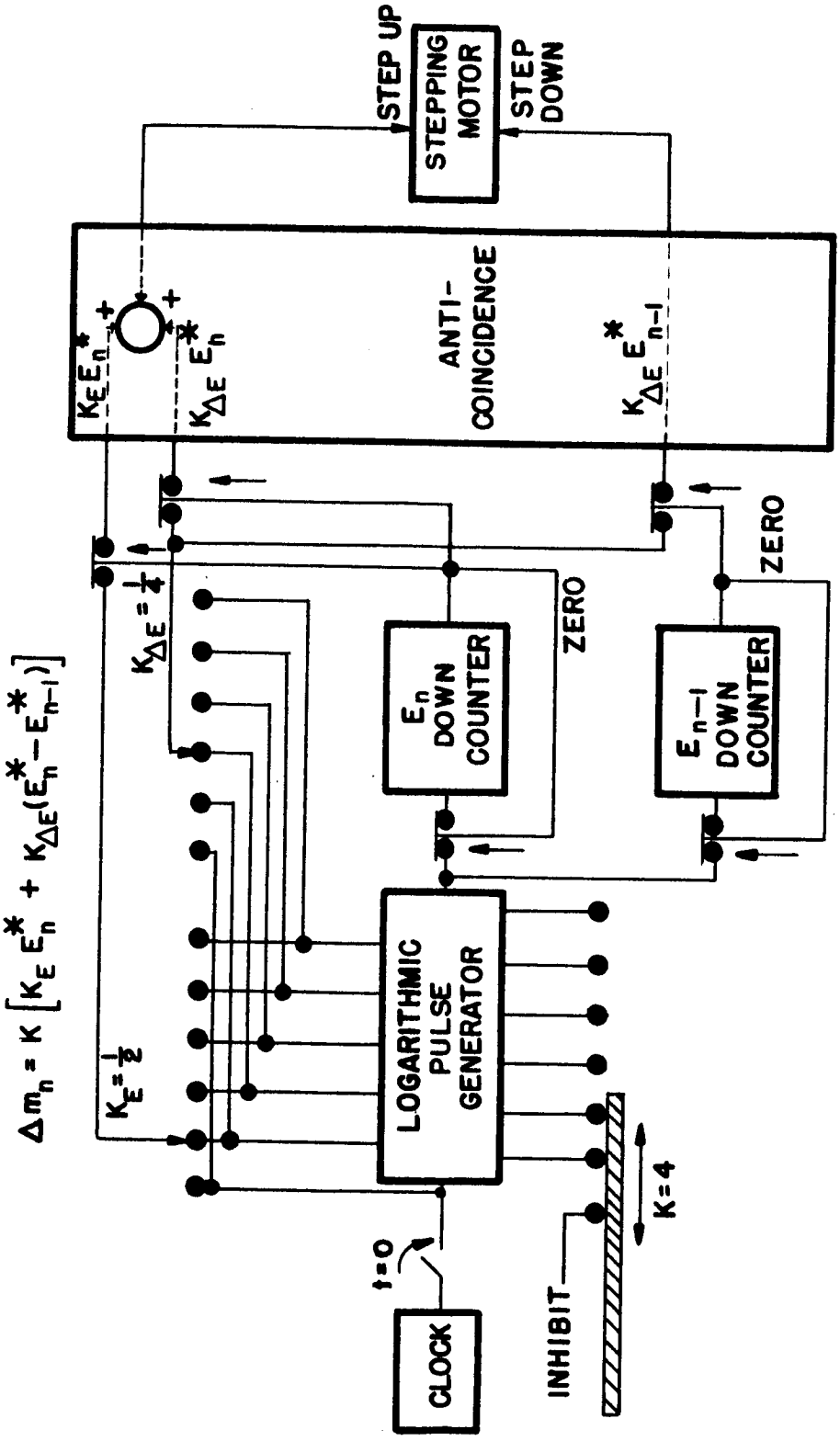


FIGURE 2.12 OUTPUT GENERATION

for the signs of E_n and E_{n-1} is not shown.

Performance in a Control System

It is not enough that truncated logarithmic quantization should lead to a somewhat simpler implementation as compared with that needed for another controller employing linear quantization. It is also necessary that the resulting controller be capable of providing good control. To investigate the performance of this controller, this section is devoted to the derivation of the transfer function of an approximately equivalent linear continuous controller. Much of the discussion here is included to indicate those conditions which are assumed in the approximating process. The section following this will then be devoted to a digital computer simulation study of the control of a specific plant.

For the purpose of defining the dynamic characteristics of the controller, it is convenient to lump the stepping motor (which acts as the controller's adder-subtractor) with the controller and to consider each step of the motor as one quanta of output, m_n. Then the output position after the change wrought by the pulses coming during the n^{th} sampling interval is given by the difference equation:

$$m_n = m_{n-1} + \Delta m_n \tag{2-4}$$

where

$$\Delta m_n = K\left[K_E E_n^* + K_{\Delta E}(E_n^* - E_{n-1}^*)\right] \tag{2-5}$$

In this equation K results from inhibiting LOG. OUTPUT pulses from the Logarithmic Pulse Generator as in Figure 2.12, while K_E and $K_{\Delta E}$ are implemented with the scaling switches shown in that same figure.

The above equations will now be manipulated to yield a difference equation form which approximates a differential equation. If T is defined as the sampling interval, then from equation 2-4

$$\frac{m_n - m_{n-1}}{T} = \frac{\Delta m_n}{T} \tag{2-6}$$

Substituting equation 2-5 into this yields

$$\frac{m_n - m_{n-1}}{T} = \frac{KK_E}{T} E_n^* + KK_{\Delta E} \frac{E_n^* - E_{n-1}^*}{T} \tag{2-7}$$

Now it is necessary to digress because the ultimate goal here is to find the transfer function of an approximately equivalent linear continuous controller. Before this can be done, some approximate way must be found for relating E_n^* to $e(nT)$ and for relating $\frac{E_n^* - E_{n-1}^*}{T}$ to $\frac{d}{dt} e(nT)$ where $e(t)$ is a continuous function of time and represents the difference

between the set point and the actual output before quantizing and sampling. Figure 2.13 illustrates the effects of quantization and decoding on e(nT) to give E_n^*. The question which must be answered is, what is the slope of the straight line through the origin which yields the "best" fit to this stepped curve? The criterion of "best" in this instance is difficult to define since what is really desired is similar performance from a linear controller and from one with this quantizing effect. The choice shown in Figure 2.13 is defined by

$$\sqrt{2}\, E_n^* \approx e(nT) \tag{2-8}$$

or

$$E_n^* \approx \frac{1}{\sqrt{2}}\, e(nT) = .707\, e(nT) \tag{2-9}$$

This choice has been made rather arbitrarily on the basis that when E_n^* is large, then the true e(nT) will lie in the approximate range

$$\frac{1}{\sqrt{2}}\,(\sqrt{2}\, E_n^*) \lesssim e(nT) \lesssim \sqrt{2}\,(\sqrt{2}\, E_n^*) \tag{2-10}$$

That is, the maximum error in the approximation of 2-8 is about 41%. While the choice of the proper slope to use has been somewhat arbitrary, note that this slope certainly lies within the bounds of

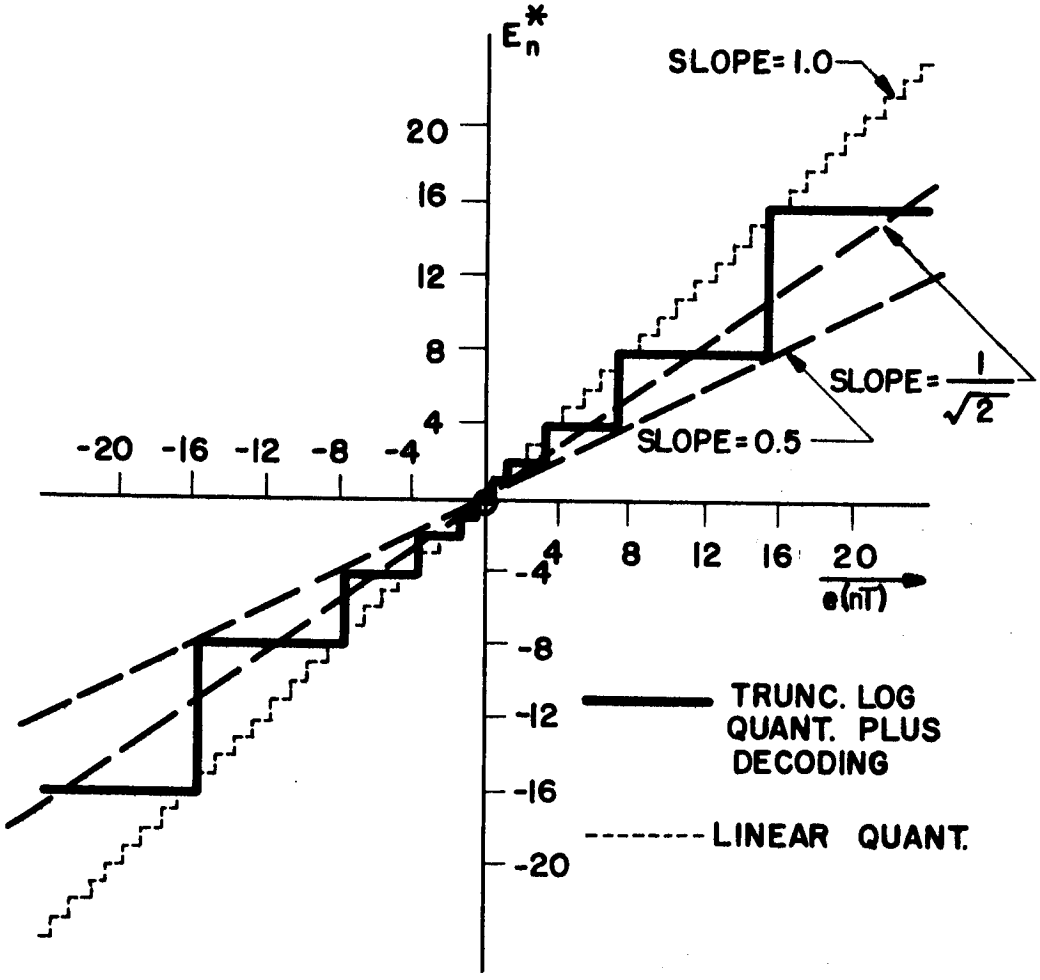


FIGURE 2.13 RELATION BETWEEN E_n^* AND $e(nT)$

$$0.500 < 0.707 < 1.000 \qquad (2\text{-}11)$$

dictated by the straight lines in Figure 2.13.

The next problem requiring attention is to relate $\frac{E_n^* - E_{n-1}^*}{T}$ to $\frac{d}{dt}e(nT)$. Here the problem is somewhat more involved because there is not only the scaling problem of the previous paragraph but also a problem requiring time averaging to achieve an approximate equivalence. For example, Figure 2.14 shows an arbitrary e(t) versus time together with the corresponding $\frac{de}{dt}$ and also $\frac{E_n^* - E_{n-1}^*}{T}$. The discontinuous nature of this latter signal means that it can never be equivalent to the continuous $\frac{de}{dt}$ when looked on only from instant to instant. However in a control environment this instant to instant equivalence is not necessarily required providing there exists an average equivalence. Consequently if the averaging effect is permitted to take place without distortion due to the intermittent bursts of information which characterize this signal, then the problem reduces to the scaling problem of the previous paragraph.

The scaling problem for this difference factor can be resolved in much the same way as it was previously for the proportional factor. This leads to the approximate

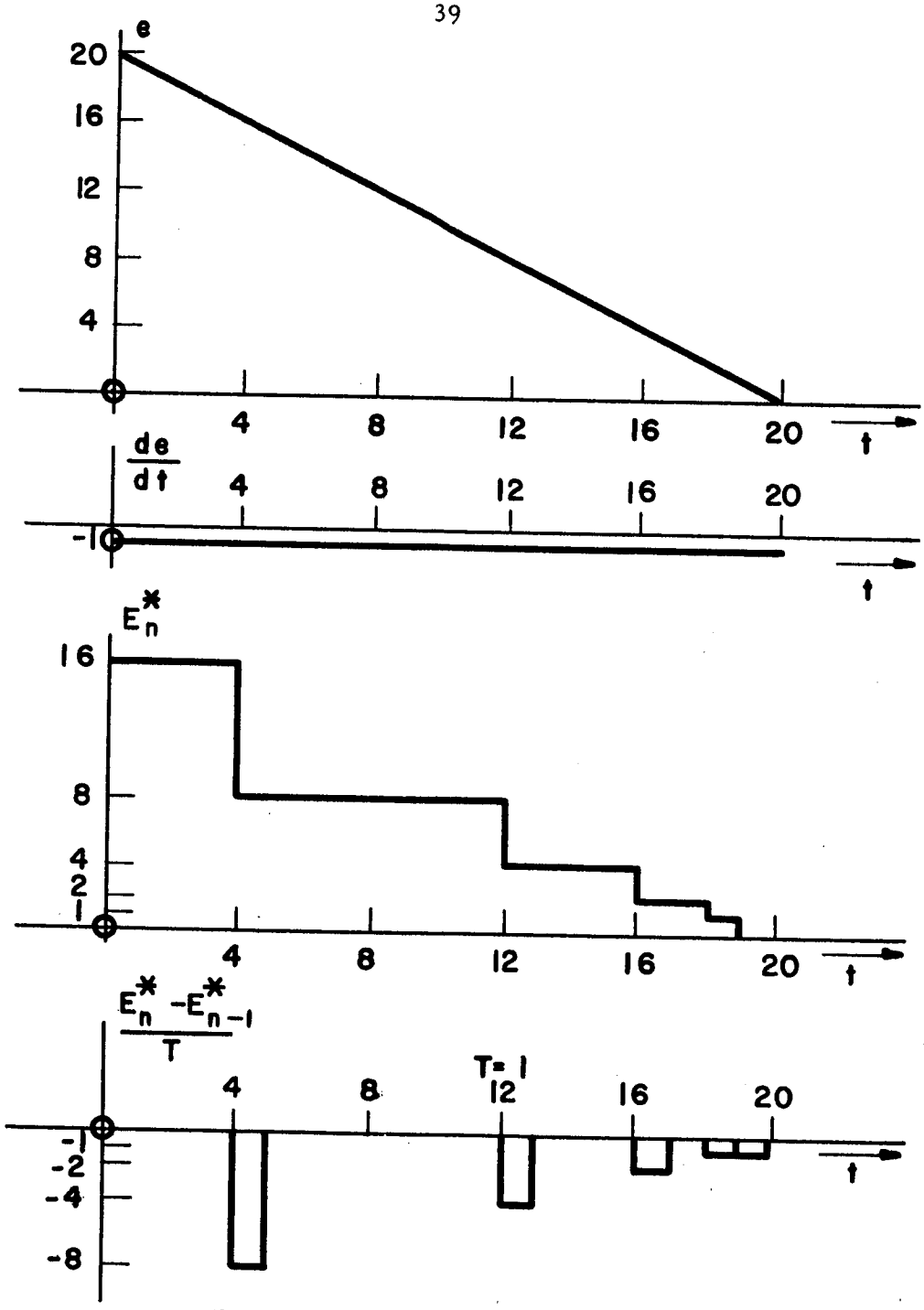


FIGURE 2.14 COMPARISON OF $\frac{E_n^* - E_{n-1}^*}{T}$ WITH $\frac{de}{dt}$

relationship:

$$\text{Average}\left[\frac{E_n^* - E_{n-1}^*}{T}\right] \approx \text{Average}\left[\frac{1}{\sqrt{2}} \cdot \frac{de}{dt}\right] \quad (2\text{-}12)$$

Two important qualifications on this result are:

1. the sampling interval, T, must be sufficiently small relative to the dominant time constant of the plant;
2. the output must not saturate.

The first requirement is really equivalent to the condition needed to reconstruct a continuous curve from sampled values; it is necessary that the curve be fairly predictable between samples. The most important factor affecting the second requirement is whether or not the stepping motor is driven to its limits so that it can step no farther, resulting in lost pulses. If this happens then the averaging effect deteriorates in proportion to the amount of this "saturation."

Finally an approximately equivalent linear continuous model for the controller plus stepping motor can be obtained by substituting equations 2-9 and 2-12 into equation 2-7 and replacing $\frac{m_n - m_{n-1}}{T}$ with $\frac{dm}{dt}$. This yields

$$\frac{dm}{dt} = \frac{KK_E}{T}\frac{1}{\sqrt{2}}e + KK_{\Delta E}\frac{1}{\sqrt{2}}\frac{de}{dt} \quad (2\text{-}13)$$

The corresponding transfer function can be written

$$\frac{M(s)}{E(s)} = K_P + \frac{K_I}{S} \tag{2-14}$$

which is the form of a proportional-plus-integral (P-I) controller. The coefficients are given by

$$K_P = \frac{K K_{\Delta E}}{\sqrt{2}} \tag{2-15}$$

and

$$K_I = \frac{K K_E}{\sqrt{2}\ T} \tag{2-16}$$

where, again, T is the sampling interval, and K, K_E, and $K_{\Delta E}$ are defined by equation 2-5.

Digital Computer Simulation

The purpose of this simulation study is to compare the operation of a controller employing truncated logarithmic quantization with that of a controller using the more natural linear quantization. To make this comparison meaningful, the two systems are operated with identical sampling rate, plant characteristics, and controller settings except for an overall gain factor of $\frac{1}{\sqrt{2}}$ in the controller with linear quantization to provide the equivalence approximation of the previous section.

The plant which is simulated has a transfer function of

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

This particular form is taken in anticipation of the experimental studies performed on the implemented controller. Except for time and magnitude normalization this transfer function is the same as that for the linearized experimental plant. To use this normalized plant in the simulation studies, one approach consists of using the equivalent pair of difference equations relating the output, c_{n+1}, and its derivative, $\dot{c}_{n+1}$, at the $n+1^{th}$ sampling instant with these values plus the input, m_n, at the n^{th} sampling instant. Thus beginning with

$$G(s) = \frac{C(s)}{M(s)} = \frac{1}{(s+1)^2} \qquad (2\text{-}18)$$

and considering m_n to be a step of m_n quanta, then

$$s^2 C(s) - s c_n - \dot{c}_n + 2s C(s) - 2c_n + C(s) = \frac{m_n}{s} \qquad (2\text{-}19)$$

or $$C(s) = \frac{s C_n + \dot{c}_n + 2 c_n + \frac{m_n}{s}}{(s+1)^2}$$

$$= \frac{s}{(s+1)^2} c_n + \frac{1}{(s+1)^2} (\dot{c}_n + 2 c_n) + \frac{m_n}{s(s+1)^2}$$

$$C(s) = \left[\frac{1}{(s+1)} - \frac{1}{(s+1)^2}\right] c_n + \frac{1}{(s+1)^2}(\dot{c}_n + 2c_n) +$$

$$\left[\frac{1}{s} + \frac{-1}{(s+1)} + \frac{-1}{(s+1)^2}\right] m_n \qquad (2\text{-}20)$$

The inverse Laplace Transform of this expression, $c(t-nT)$, can now be found, its derivative taken to give $\dot{c}(t-nT)$ and both of these evaluated at $t = (n+1)T$, giving c_{n+1} and $\dot{c}_{n+1}$. The results are

$$c_{n+1} = (1+T)e^{-T}c_n + Te^{-T}\dot{c}_n + \left[1 - (1+T)e^{-T}\right] m_n \qquad (2\text{-}21)$$

and $$\dot{c}_{n+1} = -Te^{-T}c_n + (1-T)e^{-T}\dot{c}_n + Te^{-T}m_n \qquad (2\text{-}22)$$

The block diagrams of the two systems are depicted on the one generalized diagram of Figure 2.15. For both systems

$$e_n = r - c_n \qquad (2\text{-}23)$$

$$m_n = K_e e_n^* + K_{\Delta e}(e_n^* - e_{n-1}^*) + m_{n-1} \qquad (2\text{-}24)$$

In addition, for the linearly quantized system, e_n^* is related to e_n as in Figure 2.13. Also

$$K_{GA} = \frac{1}{\sqrt{2}} \qquad (2\text{-}25)$$

For the truncated logarithmic quantized system, e_n^* is related to e_n as in Figure 2.13. Also

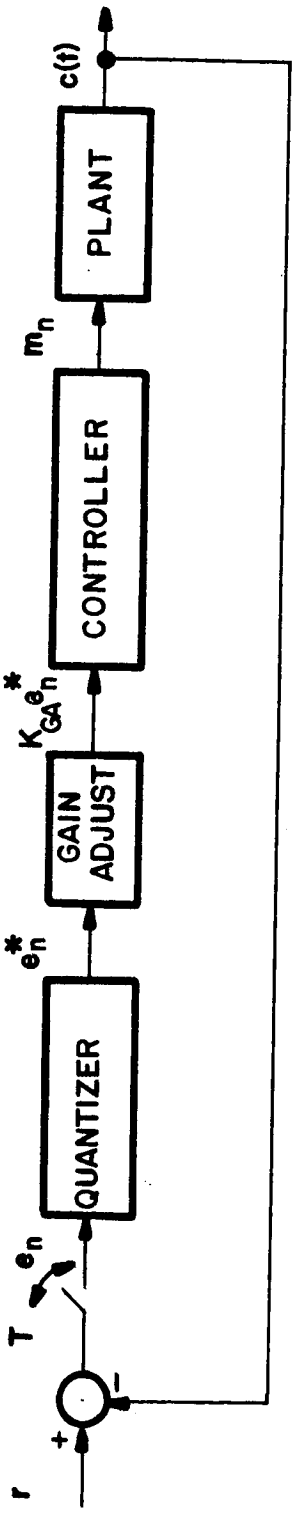


FIGURE 2.15 SIMULATED SYSTEM

$$K_{GA} = 1 \qquad (2\text{-}26)$$

The results of this comparison are shown in Figures 2.16 and 2.17 for two different magnitudes of step input and for controller settings of $T = 1/8$, $K_e = 1/4$, $K_{\Delta e} = 2$. Note that the responses are not identical but that the character of the two are similar. The important point is that good control is possible using truncated logarithmic quantization.

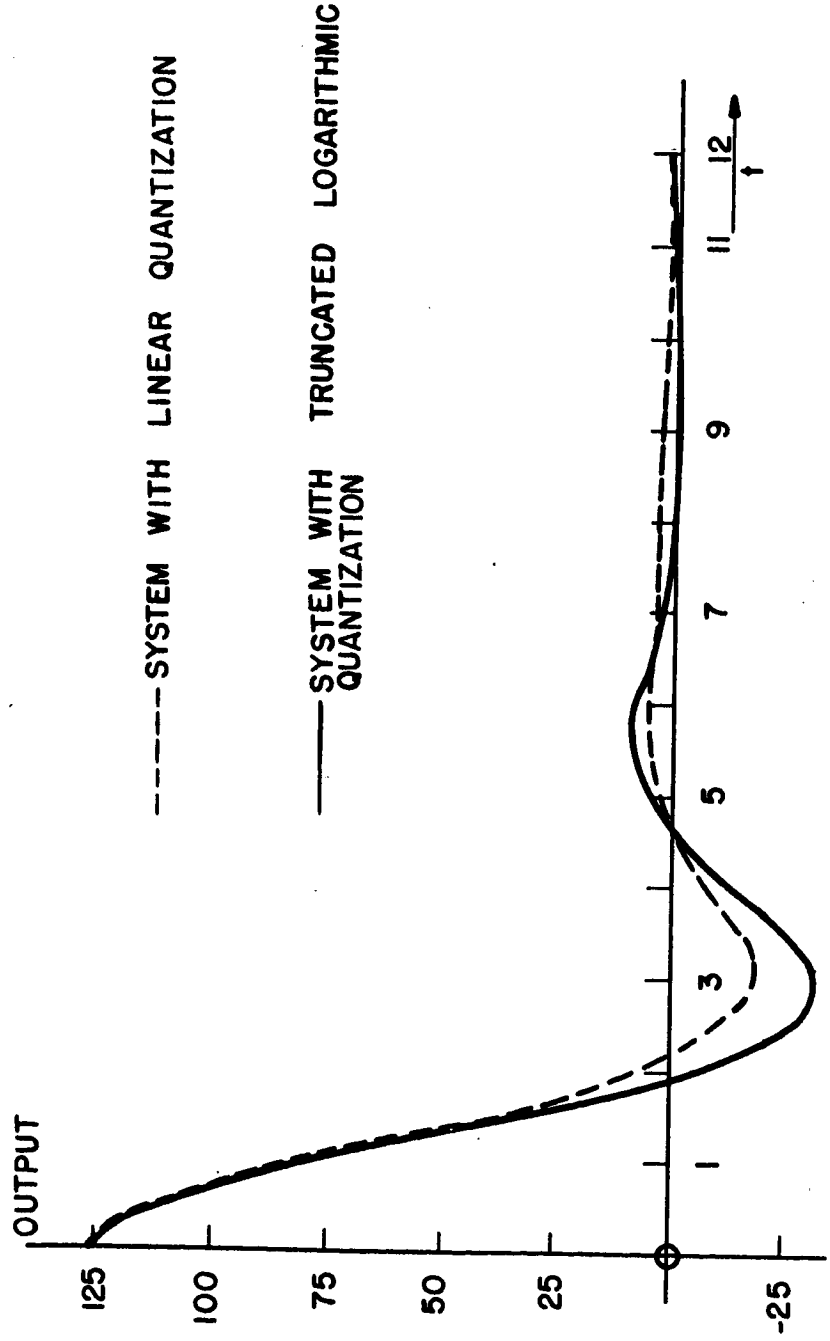


FIGURE 2.16 SIMULATED RESPONSES TO A 125 QUANTA STEP

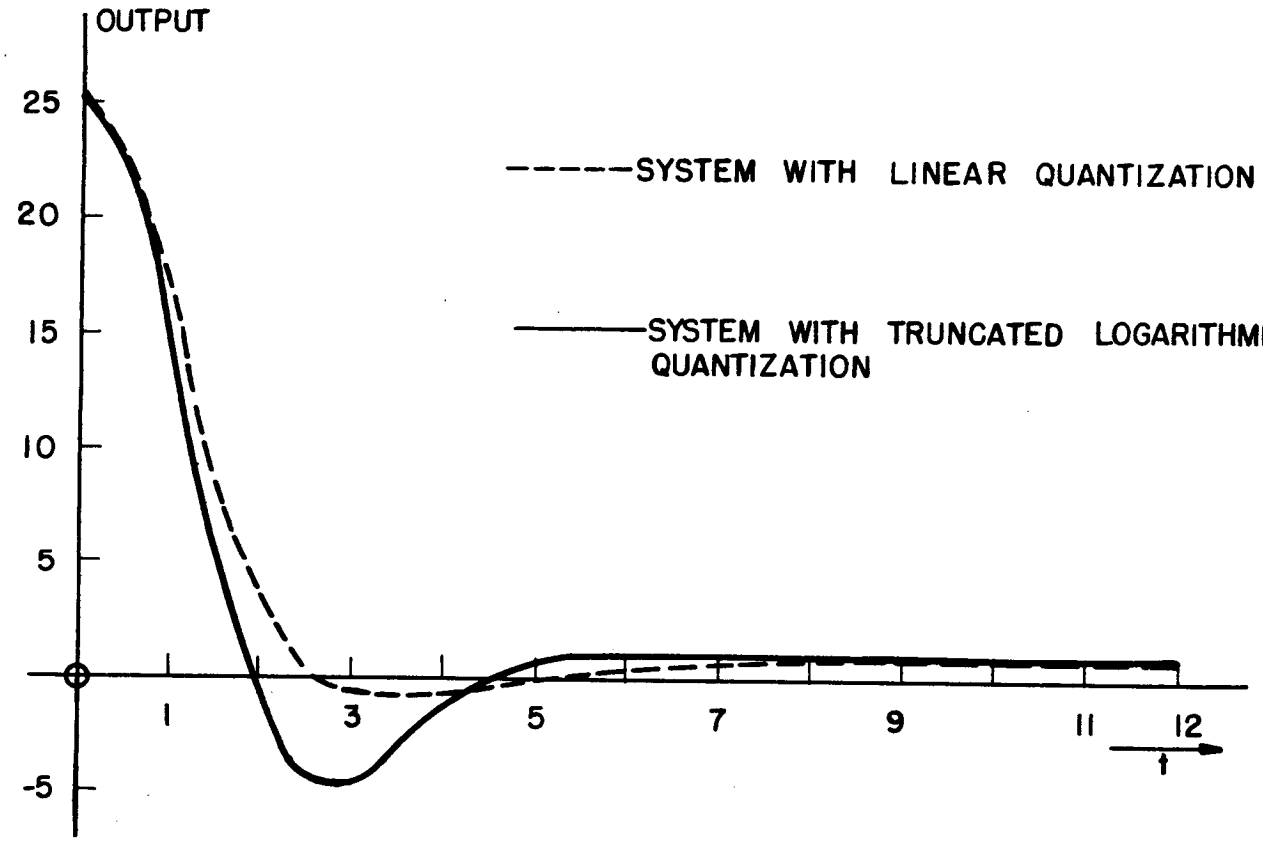


FIGURE 2.17 SIMULATED RESPONSES TO A 25 QUANTA STEP

CHAPTER III

DIGITAL CONTROLLER DESIGN

Introduction

One of the fundamental objectives of this study has been to design and construct a prototype of the digital controller. The digital logic elements (flip-flops, AND gates, etc.) available for design purposes have not been constrained to include only a few fundamental components. Rather, any special purpose logic element has been utilized provided only that it would yield a significant reduction in the total logic required. On the other hand, a mild constraint to the design was incurred by the decision to break the total controller function up into a number of functional blocks, and then to realize each of these functional blocks in one special purpose printed circuit board.

This chapter begins with the input and output constraints imposed upon the controller design. Following this, the philosophy of operation of the entire controller is discussed. This operation is then broken down so that it can be carried out by eight functional blocks, and each of these blocks is

examined in detail. Finally the chapter is closed with photographs of the implemented controller together with close-ups of typical printed circuit boards used for the functional blocks.

Input and Output Constraints

The design of the controller has been constrained to accept the output from the digital transducer in serial binary form, most significant bit first, as shown in Figure 2.5b. Furthermore the sampling rate is assumed to be controlled by an adjustable clock in the controller which sends a pulse to the digital transducer. This pulse then initiates the serial transmission from transducer to controller.

The controller output is assumed to be in a form which is compatible with the small amount of logic required for driving a stepping motor. The form used consists of Step Up pulses to step the motor in one direction and Step Down pulses to step it in the opposite direction.

Controller Operation and Philosophy

The design requirements of the controller have been dictated to a large degree by the requirements of the logic employed for generating the controller output. Thus once the

decoding scheme of Figure 2.12 was decided upon, this dictated the need for two down counters for holding the present and previous errors. This in turn led to the need for storage of the present error, $\langle \text{LOG } E_n \rangle$, which is needed during the next sampling interval to form $\langle \text{LOG } E_{n-1} \rangle$. The controlling logic of Figure 2.12 is needed. It, in effect, opens and closes the various switches shown there. Provision must be made for obtaining anticoincidence among the three pulse trains which go to the stepping motor. In addition $\langle \text{LOG } E_n \rangle$ must be formed from the set point information and the digital transducer information.

The main factor which led to this scheme of output generation was the reduction in logic required when compared with alternative approaches. However there are two other advantageous features which result from this approach.

1. The controller parameters, K, K_E, and $K_{\Delta E}$ are all digital in nature. This means that use of the controller requires absolutely no calibration.
2. The three output pulse trains which go to the stepping motor are interspersed with each other. The alternative of transmitting these

pulse trains to the stepping motor one after another can produce erroneous results during the subtraction of two of the pulse trains if the stepping motor runs up against either end of its range of travel. If this happens, pulses will be lost, yielding an incorrect output. By interspersing the three pulse trains, the stepping motor must move, during each sampling interval, more steadily toward the final position.

Operation of the controller is divided into two modes. The preparation mode operation is depicted, in abbreviated form, in Figure 3.1. Operation begins with the generation of the sampling pulse, P, which initiates the transmission of the transducer output to the Input Block. In the Input Block the form of this information is changed and ten timing pulses, T_s, are derived from it. The Serial Set Point is a functional block which distributes these ten timing pulses, one to each of ten wires. By means of the set point switches shown in Figure 3.1, this sequence of ten pulses is converted into a serial representation of the switch positions. This set point information is then compared with the transducer output information in the

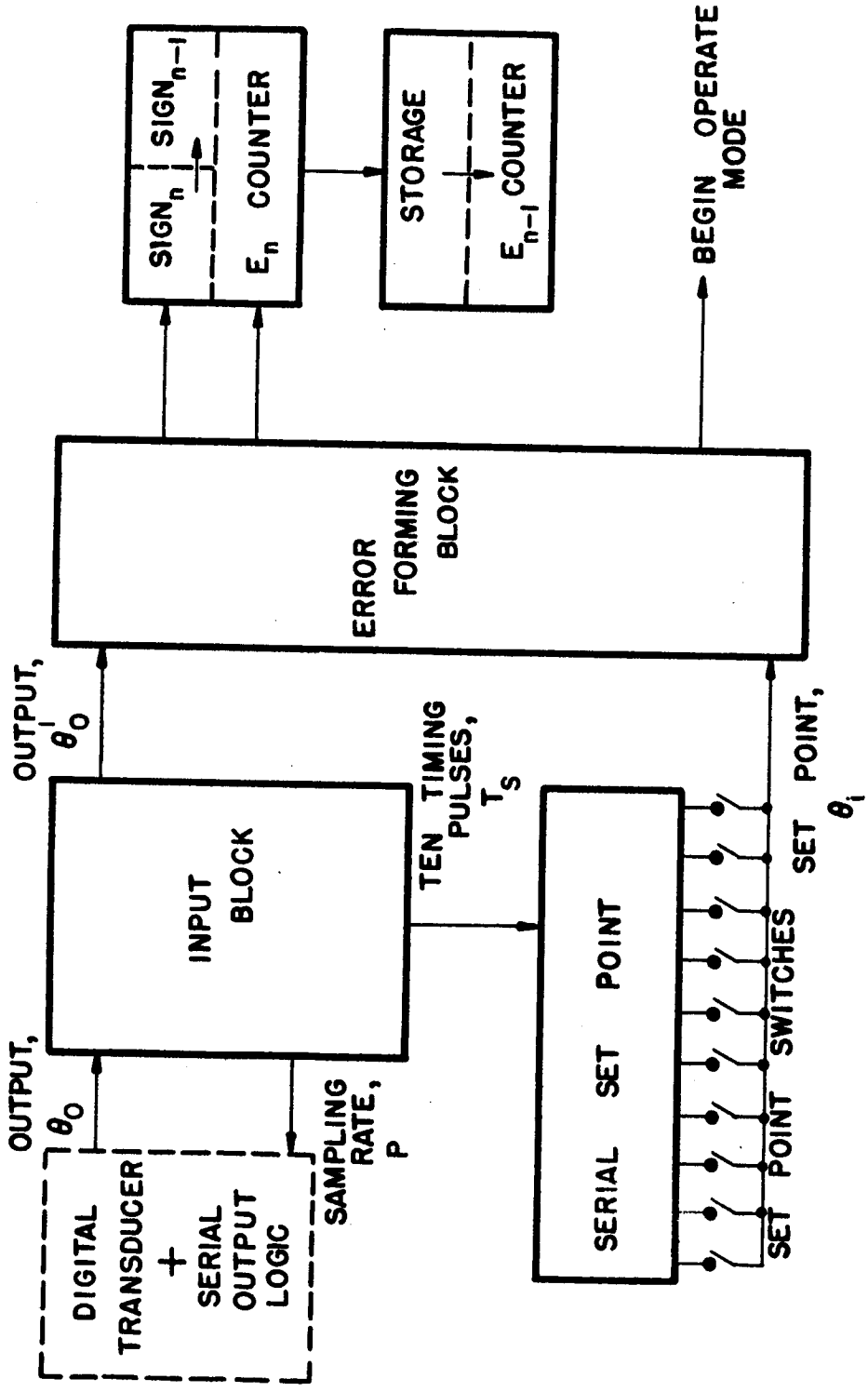


FIGURE 3.1 LOGIC FOR THE PREPARATION MODE

Error Forming Circuit. The resulting error ends up in the E_n Counter and its sign ends up in the $SIGN_n$ memory element. Meanwhile the previous error, which has been kept in Storage, is transferred to the E_{n-1} Counter. The sign of the previous error has also been transferred to the $SIGN_{n-1}$ memory element. When the present error is finally formed, two things happen. First this present error is stored in Storage as well as in the E_n Counter. Second, the Error Forming Circuit emits a signal, "Begin Operate Mode."

The logic for the operate mode is depicted in abbreviated form in Figure 3.2. The points to be noted here are the functional blocks and the important signal lines which are involved and the close relationship between Figure 3.2 and Figure 2.12. The Stop Logic block performs the function of the switches shown in Figure 2.12. The Output Gating block carries out the anticoincidence function and also manipulates the step sign, depending on the signs of E_n and E_{n-1}. One point of discrepancy between Figure 3.2 and 2.12 is the output information of the two counters. In Figure 2.12 the counters counted down to zero and stopped. Here the counters are counted through zero to 1111 and stopped there because the

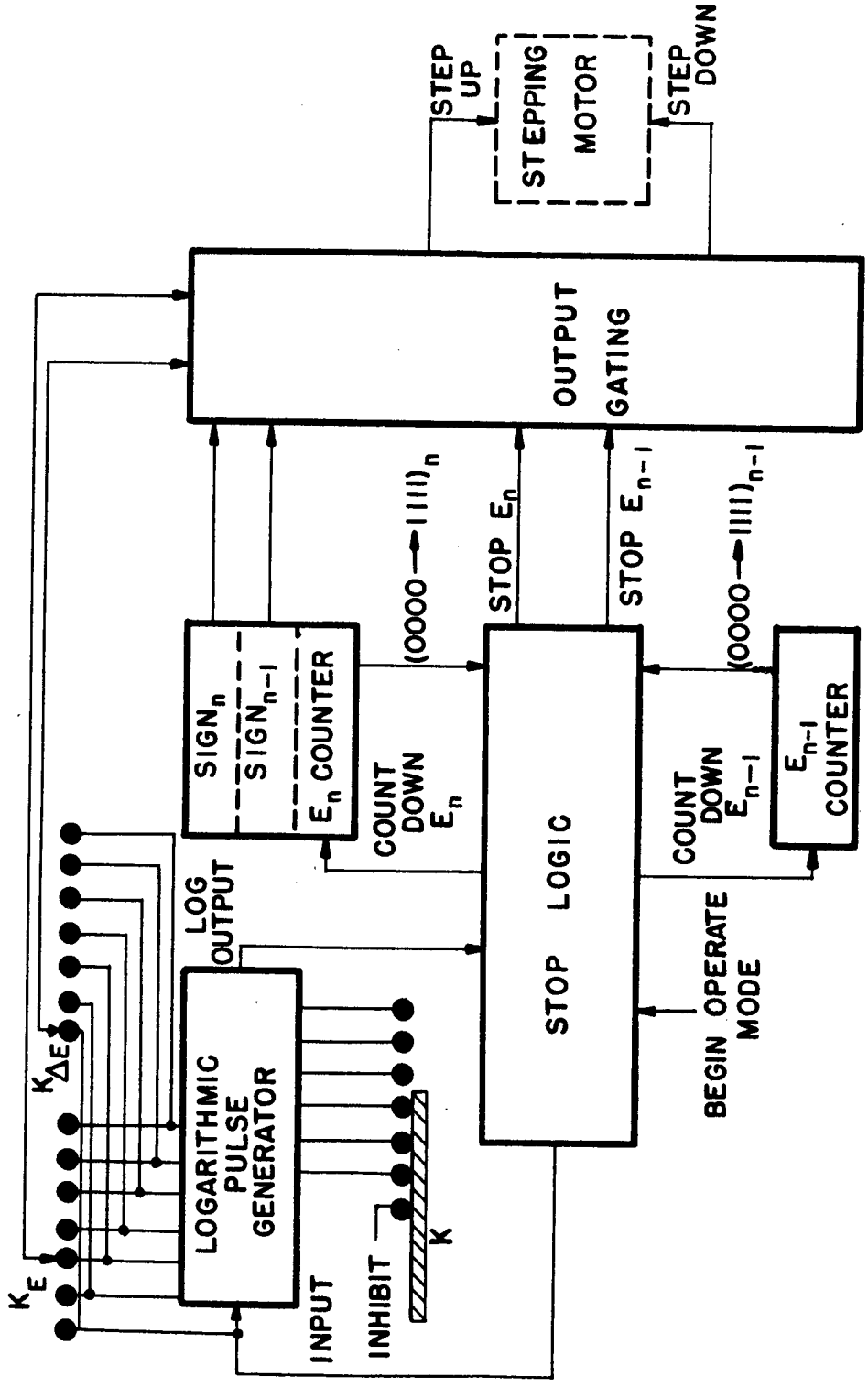


FIGURE 3.2 LOGIC FOR THE OPERATE MODE

resulting logic is simpler. This necessitates counting each counter down one step before starting the output generation.

Logic Element Descriptions

In this section the logic elements and signals of Figure 3.3 and 3.4 will be discussed. Because some of the logic elements are sensitive to voltage levels, as in Figures 3.3a and 3.3b, these will be designated as (+) and (-) levels respectively. Other of the logic elements are sensitive to a voltage transition from -12 volts to 0 volts, as in Figure 3.3c and 3.3d. This will be designated a (-)/(+) transition.

A purely level logic element is shown in Figure 3.4a. This can be considered an AND gate, as designated by the dot inside the symbol. To be interpreted in this fashion, it is necessary to take note of the reference polarities on the inputs and the output of the symbol. These indicate that the output, C, will be a (-) level if both inputs A AND B are (+) levels (otherwise C will be a (+) level).

Figure 3.4b indicates a pulse generator with two steering gate inputs. The output, H, will be a negative pulse any time certain input conditions are met. One way to obtain an input pulse is to trigger the D-E steering gate input with

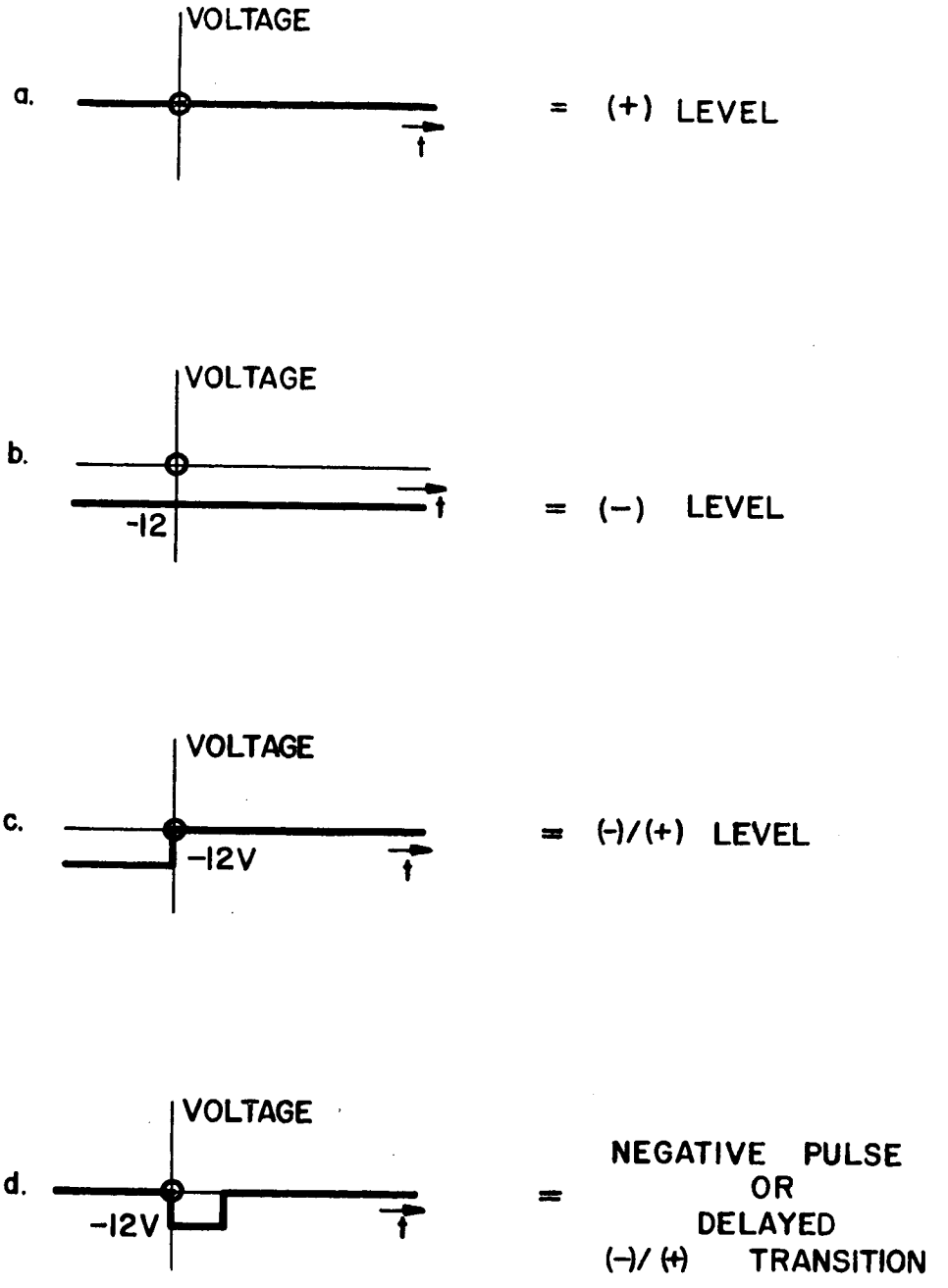


FIGURE 3.3 SIGNALS AND THEIR DESIGNATIONS

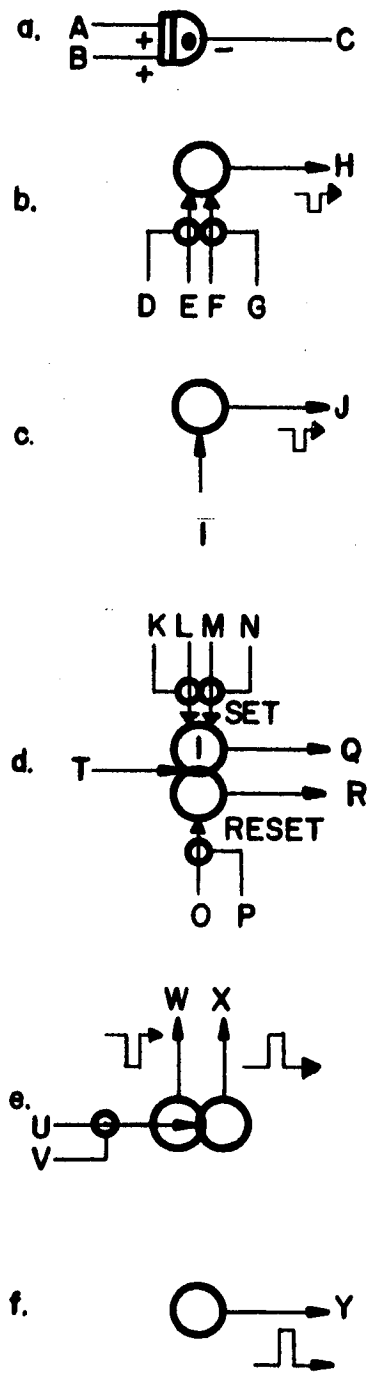


FIGURE 3.4 LOGIC ELEMENTS

a (+) level on E and a (-)/(+) transition on D. Thus for this input the pulse generator is really an AND gate for a (+) level and a (-)/(+) transition, where the output is a negative pulse instead of a level as in the AND gate of Figure 3.4a. Another way to obtain an output pulse is to trigger the F-G steering gate input with a (+) level on F and a (-)/(+) transition on G. Thus the pulse generator behaves like an OR gate in that either the D-E steering gate input OR the F-G steering gate input can be used to trigger a negative pulse output. In general a pulse generator can have as many of these steering gate inputs as desired. On occasion a pulse generator will be shown with only a single arrow on the input, as in Figure 3.4c. The input here is the (-)/(+) transition, and it is implied that the level input, which is not shown, is always connected to 0 volts.

The next logic element, shown in Figure 3.4d, is a flip-flop. This element has two outputs, Q and R, which are levels that are always opposites of each other. That is, if Q is a (+) level, then R must necessarily be a (-) level. The Set and Reset inputs to a flip-flop are steering gates and thus they are shown exactly the same as for the pulse generator. The difference between the two devices lies in the output behavior.

If a flip-flop is set (by triggering on the K-L input, for example) then the Q output will go to the (-) level and remain there. Any further Set inputs will not affect the output. However when the flip-flop receives a Reset input, then the Q output will return to the (+) level. Again if the flip-flop is already reset, then further Reset inputs will not affect the output. Finally there is a special Trigger input, T. Any time a (-)/(+) transition enters the flip-flop at T, the output will change. That is, it will go from the Set condition to the Reset condition or vice-versa.

The one-shot of Figure 3.4e is a logic element which behaves, for the W output, exactly like a pulse generator. However it has two features which give it some versatility. First and foremost the output pulse width can be made very much longer than the output pulse width of the pulse generator. Secondly not only is the normal W output available, but also its inverse or complement, a positive pulse, is available at output X.

The final element, shown in Figure 3.4f, is a free running pulse generator. Its output is a steady stream of positive pulses (-12 volts to 0 volts and then back to -12 volts)

with constant pulse width but adjustable pulse rate. This type of "clock" element has been used in the design rather than a free running multivibrator (a device with a circuit configuration similar to a one shot or a flip-flop) because of the vast range in pulse rate which can be achieved while using only one potentiometer.

With the description of these logic elements completed, the detailed logic in each functional block will be described.

Input Block

This functional block, shown in Figure 3.5, performs two distinct functions. One is to generate the pulse, P, which is used to initiate the flow of serial information from the digital transducer. Three other outputs, P', P'', and P''', are generated to reset various other functional blocks. Because some of the resetting cannot be done prematurely, there is an inhibit input, denoted "Reset 1," which comes from the Stop Logic. Until this is a (+) level, the outputs P and P' are inhibited. In normal operation this inhibit is never called upon to function. However if it is attempted to set the sampling rate faster than the controller output can be generated, then this inhibit signal prevents what would otherwise be faulty operation.

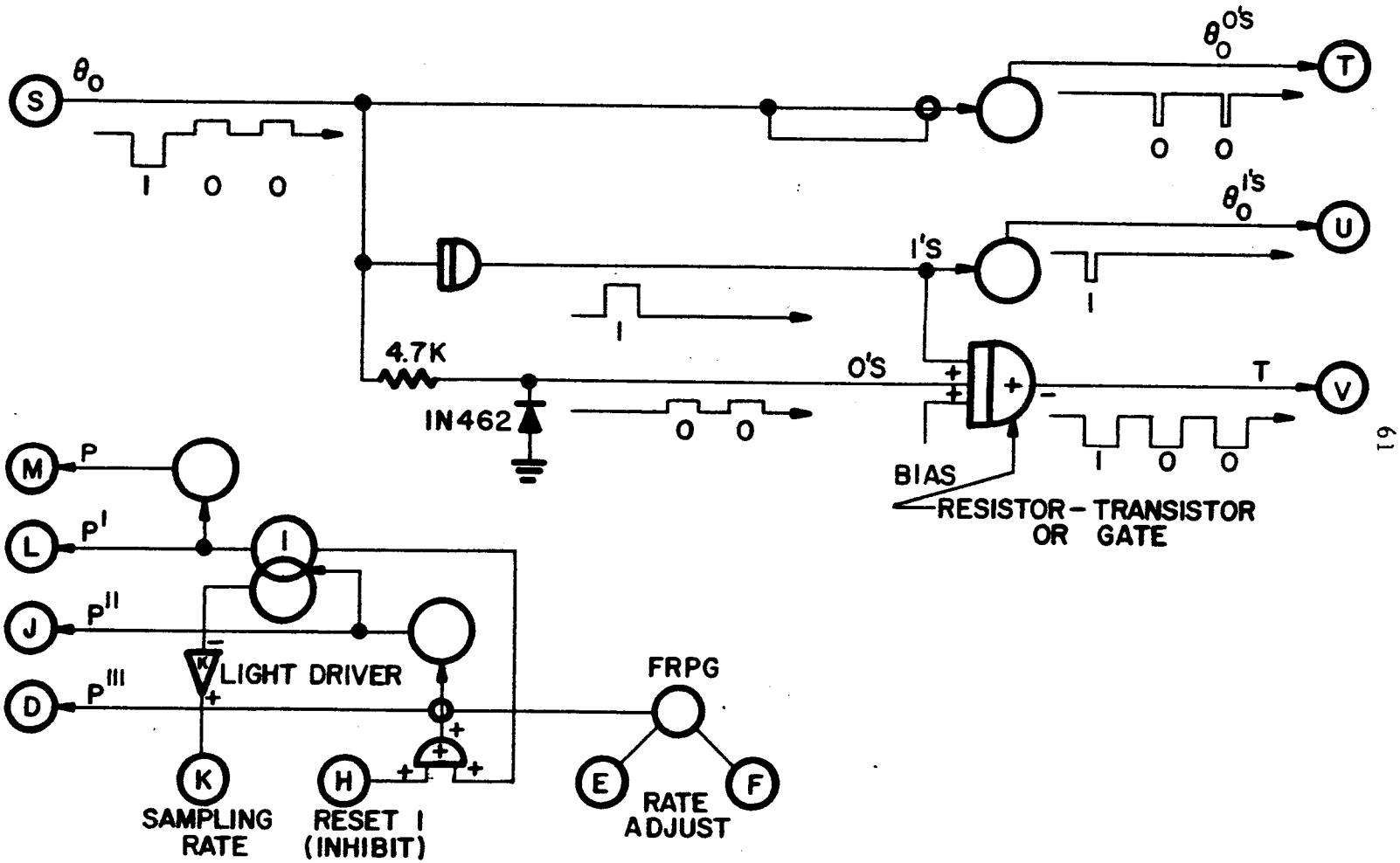


FIGURE 3.5 INPUT BLOCK

The function of the upper half of Figure 3.5 is described by the four waveforms on the input and the three outputs. Three types of information are picked out of the input waveform, which consists of either negative or positive pulses about 0 volts. The top pulse generator is used in an interesting manner to pick out the positive input pulses. The level logic element below this picks out the negative input pulses and feeds the result into a pulse generator in order to provide negative output pulses. In both of these cases the outputs are short duration negative pulses triggered from the leading edge of the input pulses. The resistor-transistor OR gate is used to generate a wide output pulse whenever there is either a 0 or a 1 input pulse. It must operate in spite of the difference in pulse amplitudes and in spite of the difference in the levels involved.

Incidently the letters encircled on the edges of Figure 3.5 denote terminals on the printed circuit board connector into which this board is plugged.

Serial Set Point

Figure 3.6 shows this functional block. The five flip-flops together with the two AND gates shown in symbolic form

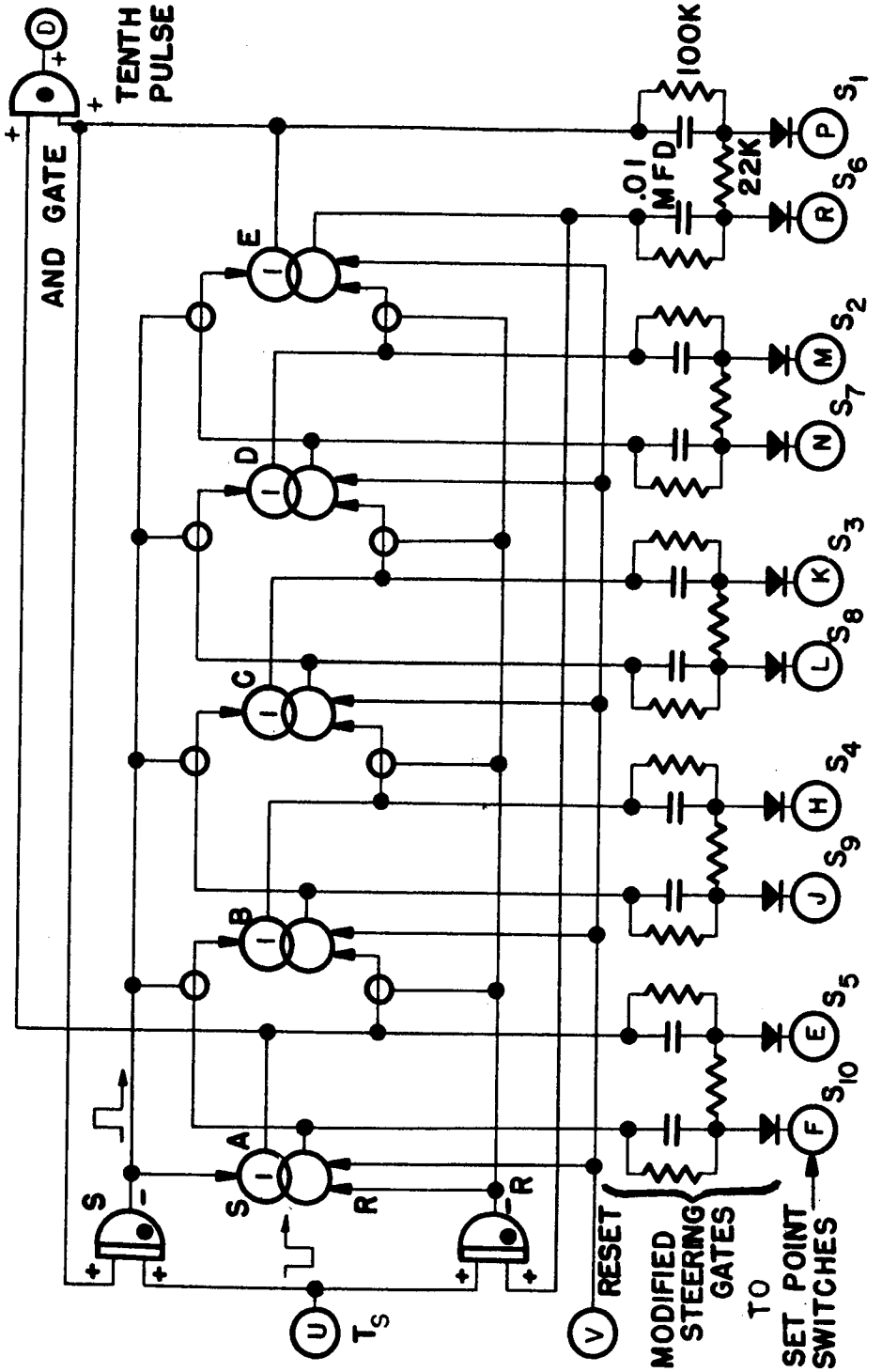


FIGURE 3.6 SERIAL SET POINT

synthesize a counter which counts in the five bit Lippel code shown in Figure 3.7. The ten outputs of these five flip-flops are used to generate ten sequential pulses from -6 volts to +6 volts and back to -6 volts. The resistor-diode AND gate gives an output which rises to 0 volts upon reception of the tenth input pulse.

Error Forming Circuit

This functional block implements the flow diagram of Figure 2.7. In addition it must generate an output to set the flip-flop corresponding to the sign of E_n and an output, "Begin Operate Mode."

Since the sequencing of events in this circuit is sufficiently complex to warrant it, attention will first be given to a transition map. This is a formal representation of a sequential process which has a finite number of inputs, outputs, and states. These terms can best be defined with reference to the specific transition map for this circuit, shown in Figure 3.8. As drawn, this is a five state machine with each state represented by a circle enclosing a number. The circuit begins in state 0 and remains in this state as long as each bit of the serial set point is the same as the corresponding bit of the

	A	B	C	D	E
Reset	0	0	0	0	0
First Input Pulse	1	0	0	0	0
Second Input Pulse	1	1	0	0	0
Third Input Pulse	1	1	1	0	0
Fourth Input Pulse	1	1	1	1	0
Fifth Input Pulse	1	1	1	1	1
Sixth Input Pulse	0	1	1	1	1
Seventh Input Pulse	0	0	1	1	1
Eighth Input Pulse	0	0	0	1	1
Ninth Input Pulse	0	0	0	0	1
Tenth Input Pulse	0	0	0	0	0

FIGURE 3.7 LIPPEL CODE COUNTER SEQUENCE

STATES

0	Start and until first difference
1	First difference
2	Opposite difference immediately
3	Match after state 1 or state 2
4	Stop

INPUTS

A	Match
B	+ difference
C	- difference
D	Same difference
E	Opposite difference

OUTPUTS

S	Set sign (If $\Theta_i - \Theta_o < 0$)
X	Count down one unit

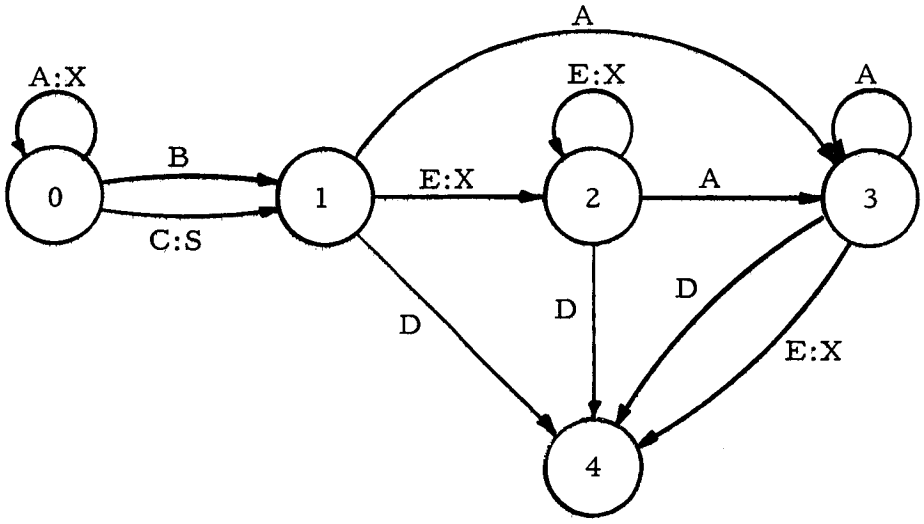


FIGURE 3.8 INITIAL TRANSITION MAP FOR THE ERROR FORMING CIRCUIT

transducer output. This sequence of events is indicated by the little loop, labelled A:X, above state 0. The A indicates that the two bits are the same, or matched, while the X indicates that the binary counter mentioned in Figure 2.7 should be counted down one unit. When finally the two bits are mismatched, operation shifts to state 1. If the bit from the transducer output, Θ_o, is a one while the bit from the set point, Θ_i, is a zero, then this represents a negative error which in turn is represented by the setting of the $SIGN_n$ flip-flop. This set of conditions is denoted by C:S on the transition map. Note that the circuit is sequential in that this first mismatch is the only time when use is made of any information to set the $SIGN_n$ flip-flop.

The operation of the circuit continues until state 4 is reached, at which time the encoding of the error is complete.

To synthesize a circuit which will perform the function mapped here there are several steps to be considered. So far the map has served the useful purpose of defining the operation of the circuit explicitly. The first question to be asked is whether a simpler transition map will do the same job. A formal technique presented by Huffman, Mealy, and Moore[3] solves

this problem and, as it turns out, the reduced map of Figure 3.9 will perform the same function. The justification for this is that in all cases the outputs are the same in the two maps regardless of the sequence of the inputs.

The next step of the synthesis is called the "assignment problem" and at the present time there is no general solution to it, although Hartmanis[4] and others[5] have presented techniques for obtaining good assignments under certain restricted conditions. The problem begins with the recognition that to synthesize a sequential circuit having from 2^N states up to 2^{N+1}-1 states requires at least N binary memory elements, or flip-flops. This is true because each state can be represented by one set of flip-flop conditions. For example, the circuit of Figure 3.9 requires at least two flip-flops, A and B. An arbitrary assignment might have state 0 be represented by both flip-flops being reset, state 4 by both being set, state 1, 2 by A being set and B being reset, and state 3 by A being reset and B being set. The idea behind the state assignment problem is to select that assignment which leads to the cheapest implementation including the input gating for the flip-flops (in order to change state) and the output gating for the

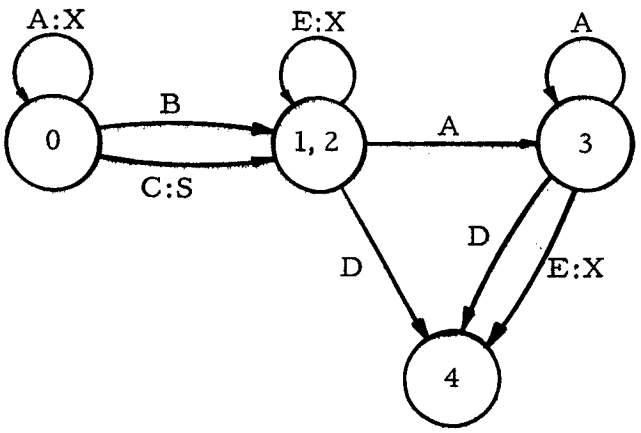


FIGURE 3.9 REDUCED TRANSITION MAP FOR THE ERROR FORMING CIRCUIT

circuit (in order to generate the required outputs; e.g., Set Sign and Count Down). This circuit also has the Begin Operate Mode output which has not been needed for consideration until now.

The final solution to the state assignment problem was made quite intuitively and involves the use of one more flip-flop than the minimum number needed. However it was felt that the resulting simplification in the input and output gating would more than make up for this. This solution is shown in Figure 3.10 and the three state-determining flip-flops are labelled FIRST DIFF. (which is not set in state 0 but which is set thereafter), MATCH AFTER DIFF. (which is set in state 3 and may or may not be in state 4), and STOP (which is set in state 4). Note that the Count Down output is given in two parts. This is done because these are not the only count down inputs to the E_n Down Counter and so a savings in logic will result if all of these inputs are ORed together all at once in one pulse generator (in the Stop Logic functional block).

The remaining gating in the Error Forming Circuit is used to form the different inputs indicated in Figure 3.8.

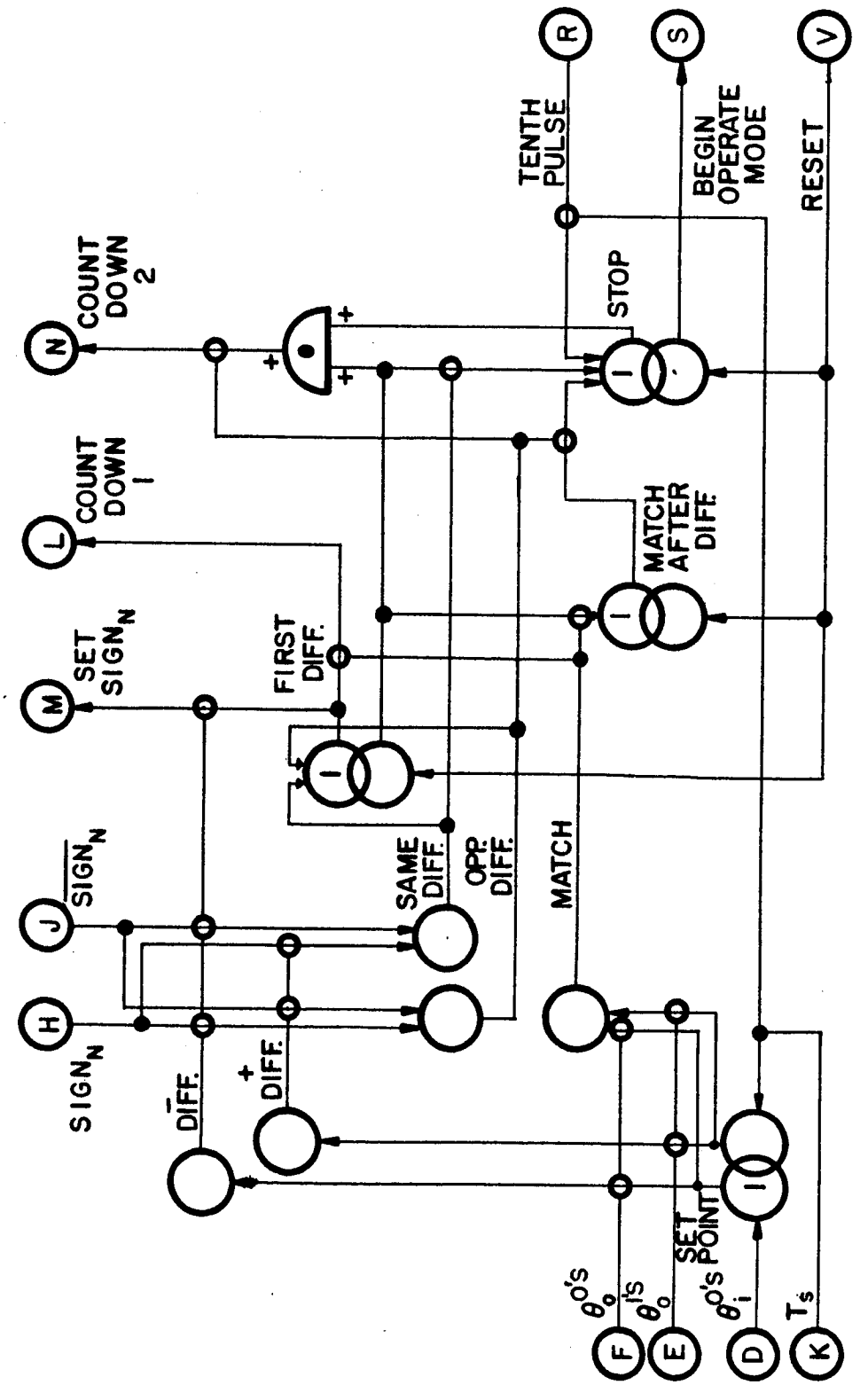


FIGURE 3.10 ERROR FORMING CIRCUIT

Data Storage and the Reset Timing

The storage and manipulation of the error data takes place in the two functional blocks shown in Figure 3.11, the E_n Counter Plus Signs, and Figure 3.12, the Storage Plus E_{n-1} Counter. In each figure the four bit binary down counter can be identified by the trigger inputs to the flip-flops. The complex part to these circuits is the sequencing of operations used to reset the counters and to perform the transfer of data. For the transfer operation the "one" sides of the E_n flip-flops in Figure 3.11 are connected to the level inputs of the reset steering gates of the four bit Storage register in Figure 3.12. The timing diagram for resetting is shown in Figure 3.13, and the numbers 1-4, indicating the sequential order of events, are also used in Figures 3.11 and 3.12 as numbers within boxes in order to help simplify the identification of the sequencing.

Logarithmic Pulse Generator

This functional unit implements the operation indicated in simplified form in Figure 2.9 and in more detail in Figure 2.11. The circuit is shown in symbolic form in Figure 3.14. Some of the symbols are familiar (i.e., the flip-flops, the

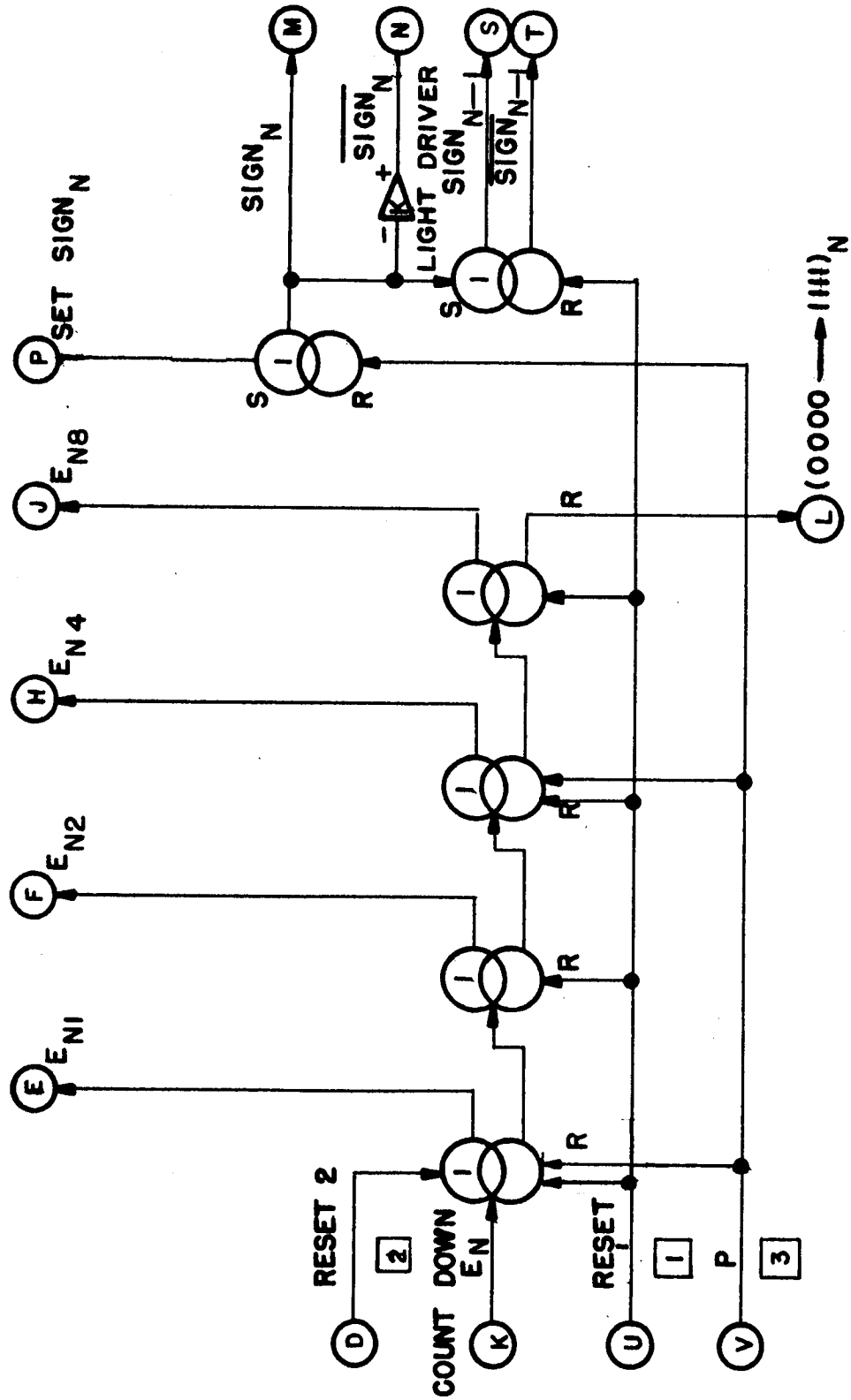


FIGURE 3-11 E_N COUNTER + SIGNS

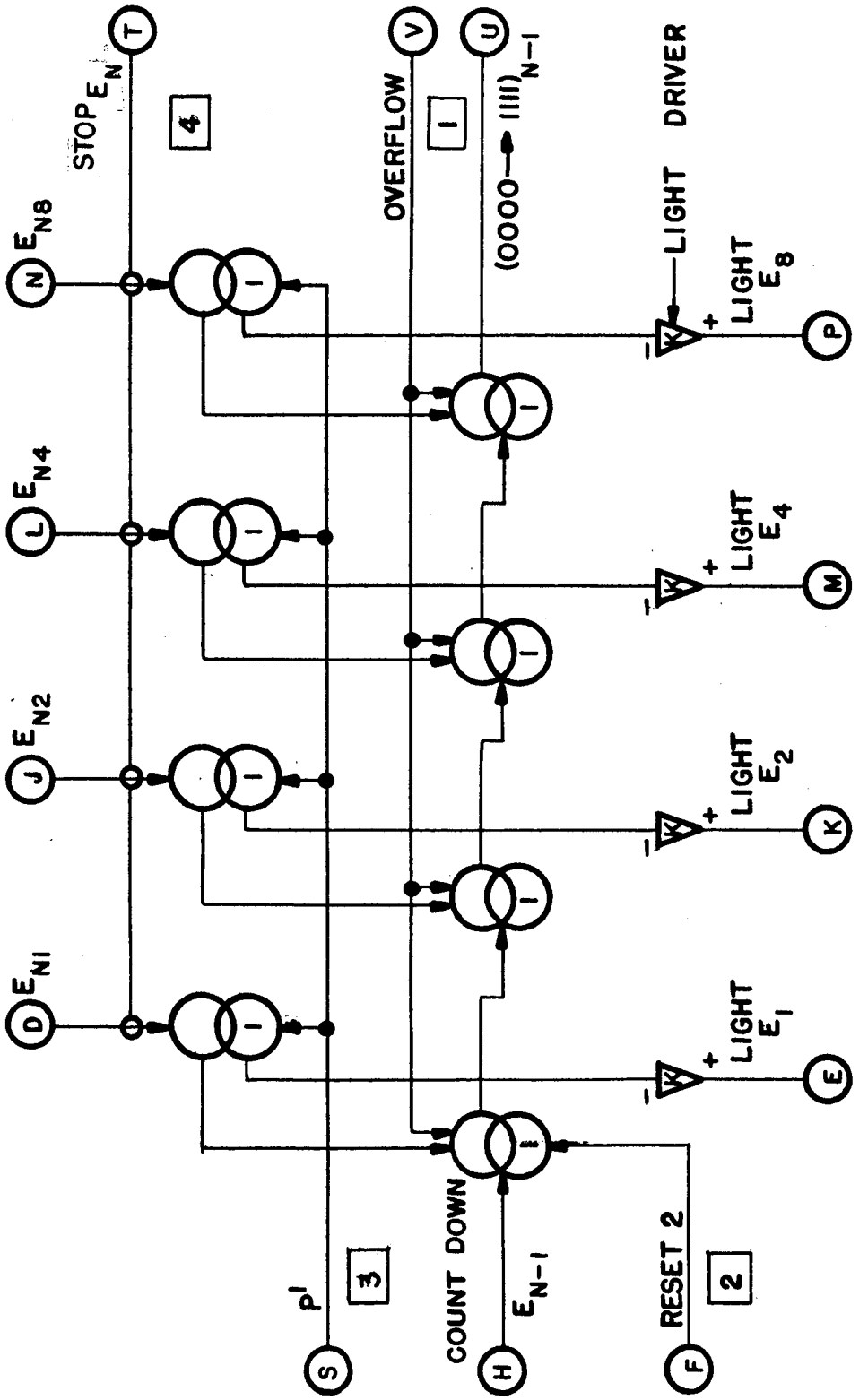


FIGURE 3.12 STORAGE+ E_{N-1} COUNTER

SEQUENTIAL ORDER	SIGNAL	COMMENTS
1	Reset 1 or Overflow	Generated in Stop Logic at the completion of the Operate Mode; Resets E_n Counter, E_{n-1} Counter and $SIGN_{n-1}$ to zero
2	Reset 2	Generated in Stop Logic from Reset 1; Sets E_n Counter and E_{n-1} Counter to all ones.
3	P'' or P' or P	Generated in Input Block at the sampling instant; Resets E_n Counter to binary 1010; Resets $SIGN_n$, and in so doing transfers its contents to $SIGN_{n-1}$; Sets Storage register to 1111 and in so doing transfers its contents to E_{n-1} Counter; Resets Serial Set Point and Error Forming Circuit; Begins Preparation Mode.
4	Stop E_n	Generated in Stop Logic from the Begin Operate Mode signal; Transfers the information just formed in the E_n Counter to the Storage register (using non-destructive readout).

FIGURE 3.13 TIMING SEQUENCE

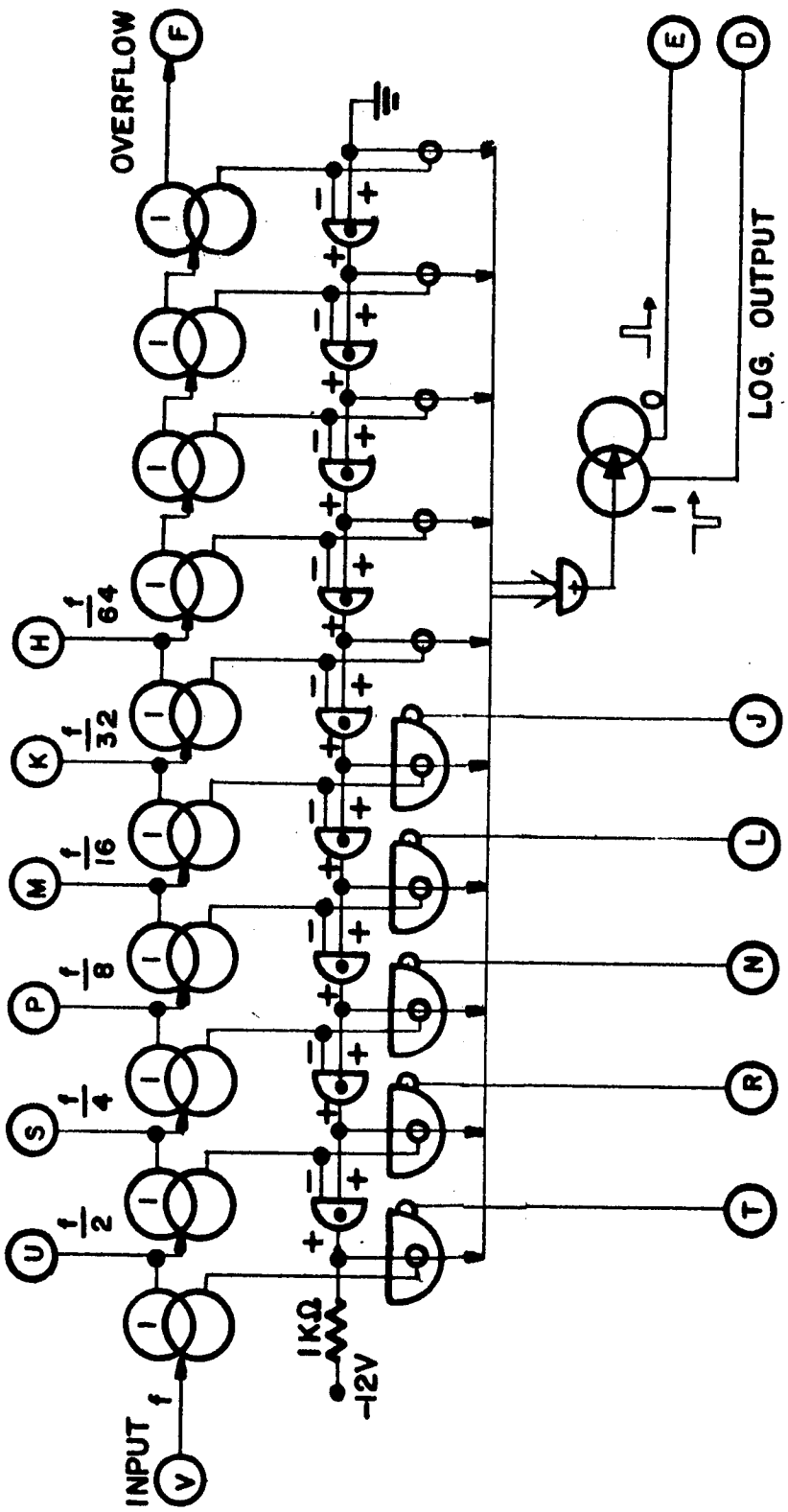
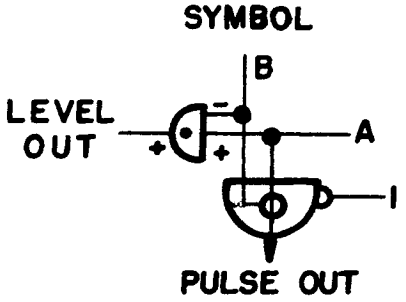


FIGURE 3.14 LOGARITHMIC PULSE GENERATOR

one-shot, and the steering gates) while others are not.

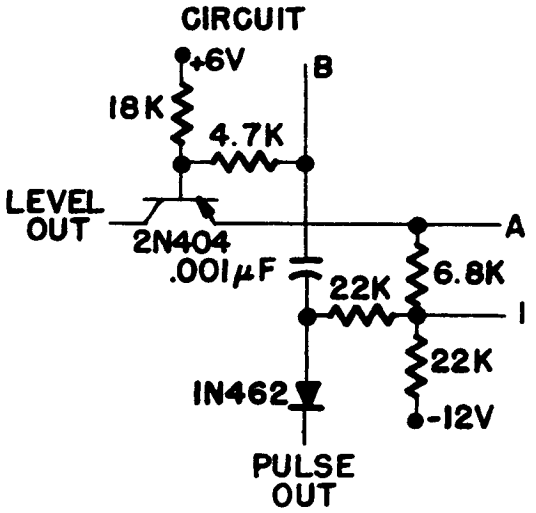
In the entire controller this circuit is the most important example of an instance where the use of special purpose digital circuitry could greatly reduce the amount of circuitry required. The AND gates and also the steering gates with the extra inhibit inputs of Figure 3.14 are described in more detail in Figure 3.15. For each output a truth table is given which indicates the behavior at that output as a function of all possible combinations of inputs. The Level Out truth table simply indicates that this device does indeed act as an AND gate if the reference polarities are assumed as shown on the inputs and output of the symbol. The Pulse Out truth table indicates that this circuit actually does behave as a steering gate unless the inhibit input, I, is at -12 volts, in which case the output is inhibited.

Figure 3.14 consists of a ten bit binary up counter which counts through 1024 input pulses before the Overflow output sees a (-)/(+) transition. By picking off the outputs of the first few counter stages, the scaling required to implement K_E and $K_{\Delta E}$ is achieved. The Log. Output picks out the (+)/(-) transition in any flip-flop when all flip-flops to the right of this



TRUTH TABLES

A	B	LEVEL OUT
0V	0V	-12V
0V	-12V	0 V
-12V	0 V	-12V
-12V	-12V	-12V



I	A	B	PULSE OUT ?
FLOATING	0 V	-12V→0V	YES
FLOATING	ANY	OTHER COMBINATION	NO
-12V	ANY	COMBINATION	NO

FIGURE 3.15 CIRCUITRY ASSOCIATED WITH THE LOGARITHMIC PULSE GENERATOR

flip-flop are in the zero state (i.e., when the lines coming out of the "1" sides of these flip-flops are all at 0 volts). The AND gates diagnose when this latter condition is met and use this information as the level input on the appropriate steering gate. That this behavior will yield the desired operation can be seen from the counter contents and the output required as shown in Figure 3.16 and comparing this with the desired output of Figure 2.11.

To implement a K Multiplier Switch, the Inhibit Input of Figure 2.11 can be wired using any of three approaches. The first is to use a special purpose shorting switch which performs the function shown in Figure 2.11, tying successive steering gate inhibit inputs to -12 volts. If such a switch is difficult to obtain then a second approach is to use an ordinary six pole, seven position switch and to wire it up to perform this function. A third solution, and the one used in the controller, makes use of a single pole, seven position switch plus diodes mounted between the successive positions, as in Figure 3.17. A second function uses this same switch to turn off the controller output pulses, and thus to provide a K = 0 gain position.

Input Pulse	Counter Contents (Least significant bit on left)	Log. Output
0	0000000000	
	↓ ----------	1st Pulse
1	1000000000	
	↓ ----------	2nd Pulse
2	0100000000	
3	1100000000	
	↓ ----------	3rd Pulse
4	0010000000	
5	1010000000	
6	0110000000	
7	1110000000	
	↓ ----------	4th Pulse
8	0001000000	

FIGURE 3.16 CORRELATION OF COUNTER CONTENTS AND LOG. OUTPUT

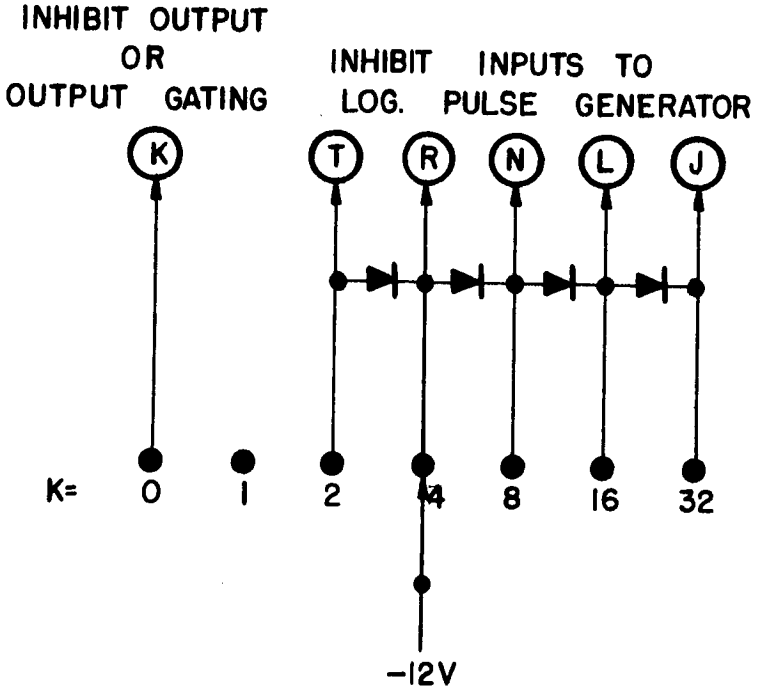


FIGURE 3.17 WIRING FOR K MULTIPLIER SWITCH

Output Gating

The Output Gating logic of Figure 3.18 can be considered in several parts. First, the "heart" of the operate mode circuitry is the free running pulse generator (FRPG) shown. This provides the input to a Gray code counter, which in turn generates four anticoincident pulse trains, t_0, t_1, t_2, and t_3. The t_0 pulses go to the Logarithmic Pulse Generator via the Stop Logic, and after appropriate gating some of these pulses return as K_E and $K_{\Delta E}$ pulses. Then if (+) levels are present on Stop E_n and Stop E_{n-1}, the three anticoincidence flip-flops are set. This gating corresponds to the "ZERO" switches of Figure 2.12. Thus the setting of these flip-flops occurs at approximately the same instant as when the t_0 pulses occur. To space these pulses apart so that the interval between the Step Up and the Step Down pulses going to the stepping motor is not too short, the three anticoincidence flip-flops are reset, in sequence, by t_1, t_2, and t_3 and the (-)/(+) transitions so produced are gated by $SIGN_n$ and $SIGN_{n-1}$ into the Step Up and the Step Down pulse generators.

The Inhibit Output signal comes from the K Multiplier switch and is -12 volts with K = 0. This prevents the three

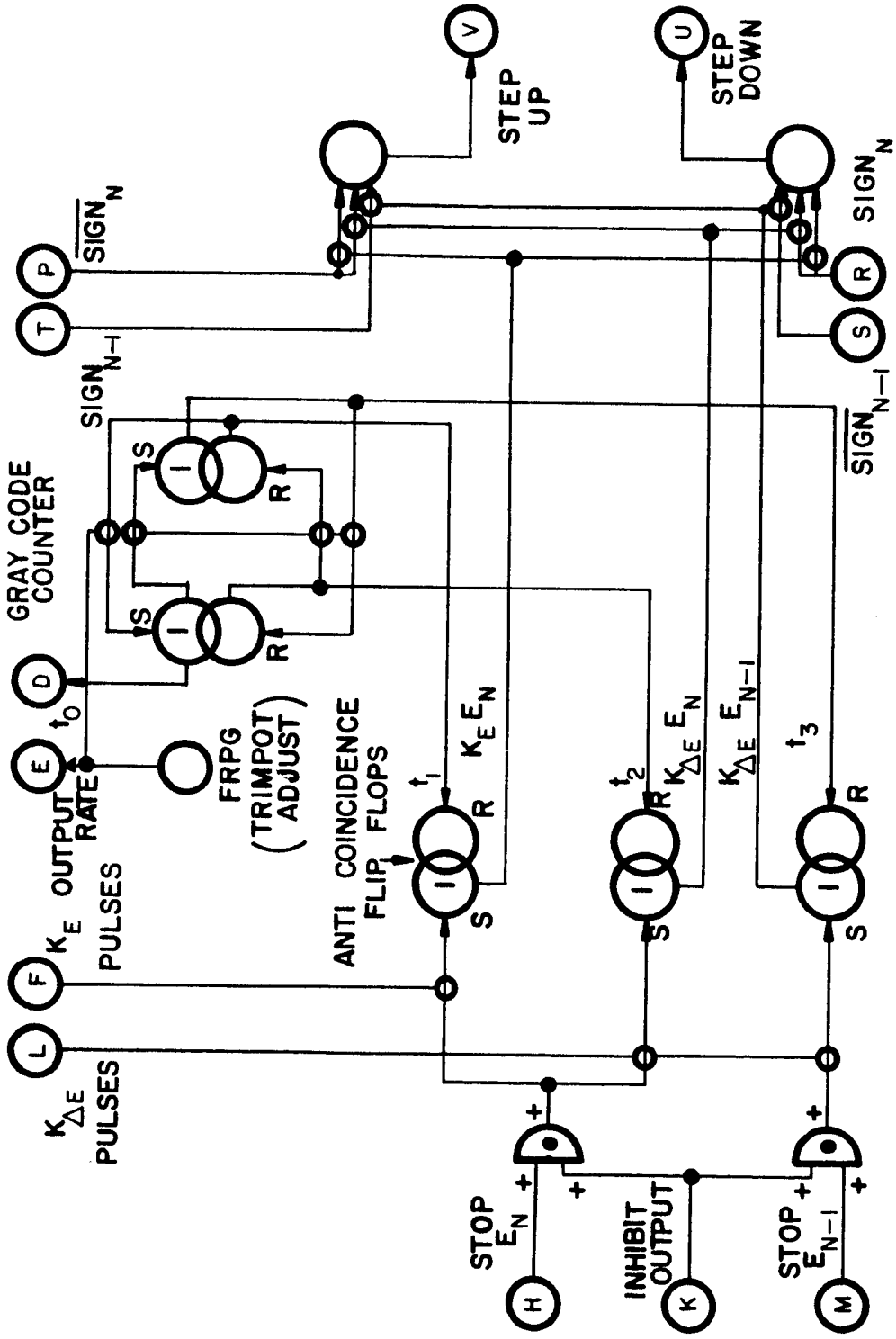


FIGURE 3.18 OUTPUT GATING

anticoincidence flip-flops from being set and thus inhibits the output.

Stop Logic

In this functional block, shown in Figure 3.19, the three flip-flops, Stop, Stop E_n and Stop E_{n-1}, perform the function of the switches of Figure 2.12. The Stop flip-flop together with the Log. Input pulse generator serves to turn on and off the t_0 pulse train to the Logarithmic Pulse Generator. This pulse train is started by the "Begin Operate Mode" signal. It is stopped by the Overflow signal from the Logarithmic Pulse Generator which indicates that the ten bit counter has counted through its complete cycle of 1024 counts to all zeros.

The Stop E_n and Stop E_{n-1} flip-flops operate in conjunction with the steering gates on the three anticoincidence flip-flops in the Output Gating circuit either to pass or to inhibit the K_E and the $K\Delta_E$ pulse trains. Recall that these pulse trains are allowed to pass after the start of the operate mode and until the E_n counter or the E_{n-1} counter is counted down through zero to all ones. To make the result correct, an extra count has to be subtracted quickly at the very beginning of the operate mode. This is accomplished by using the (-)/(+)

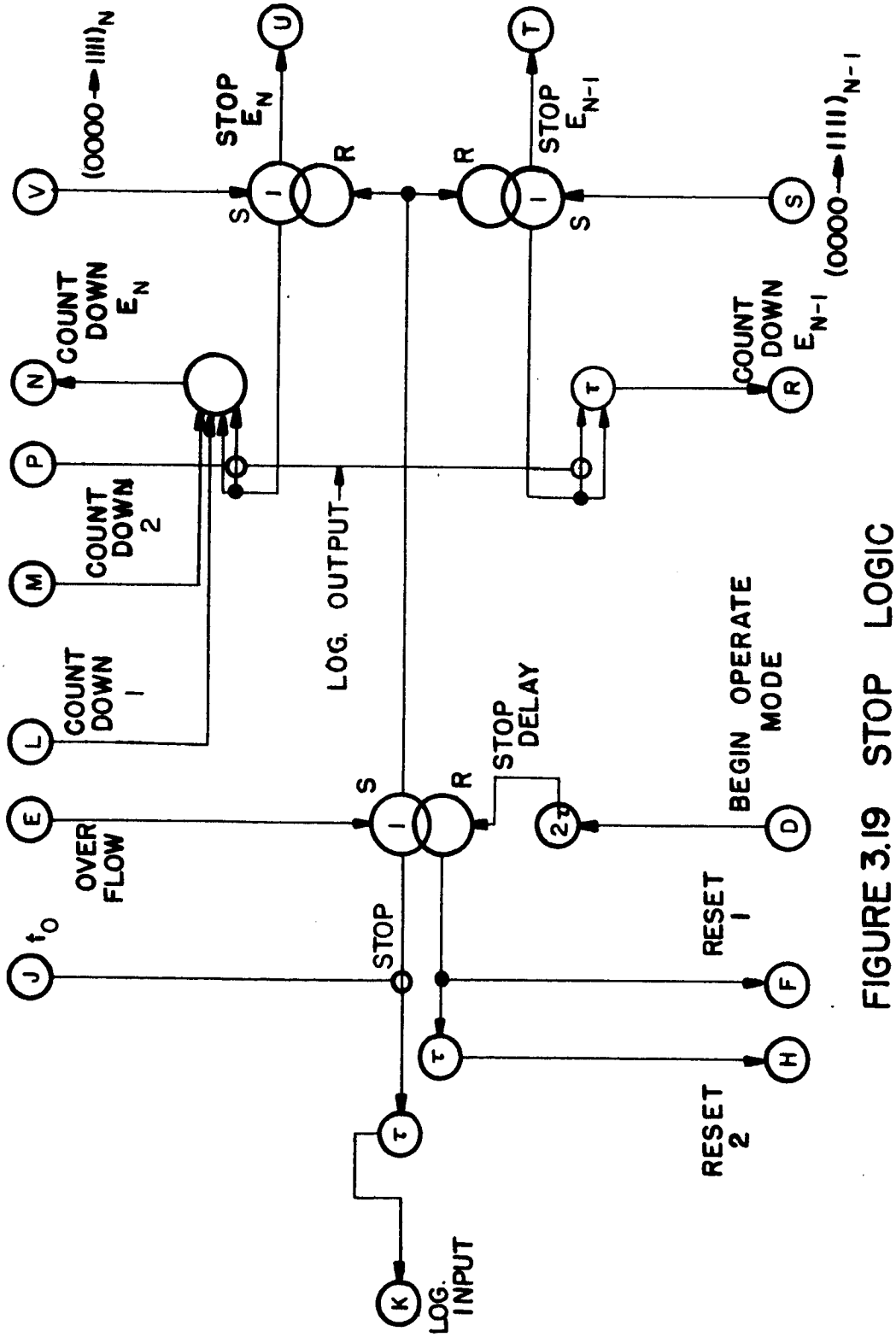


FIGURE 3.19 STOP LOGIC

transition which results when the Stop E_n or Stop E_{n-1} flip-flops are reset at the beginning of the operate mode. Thus the Count Down E_n pulse generator has four inputs: two inputs come from the Error Forming Circuit and are used during the preparation mode; a third input forms this single pulse at the beginning of the operate mode; the fourth input is the Log. Output from the Logarithmic Pulse Generator which is gated through by the Stop E_n level until the E_n Counter has counted through zero.

The Stop Delay pulse generator allows a delay between the last count down pulse during the preparation mode and the single pulse mentioned above which comes at the very beginning of the operate mode.

Physical Realization

One of the goals of the implementation was to create a device which not only works as planned but which also has a finished appearance. This was one of the chief reasons for developing the circuitry in eight functional blocks, each on a separate printed circuit board. Figure 3.20 is a photograph of the finished controller, illustrating the construction as a plug-in module.

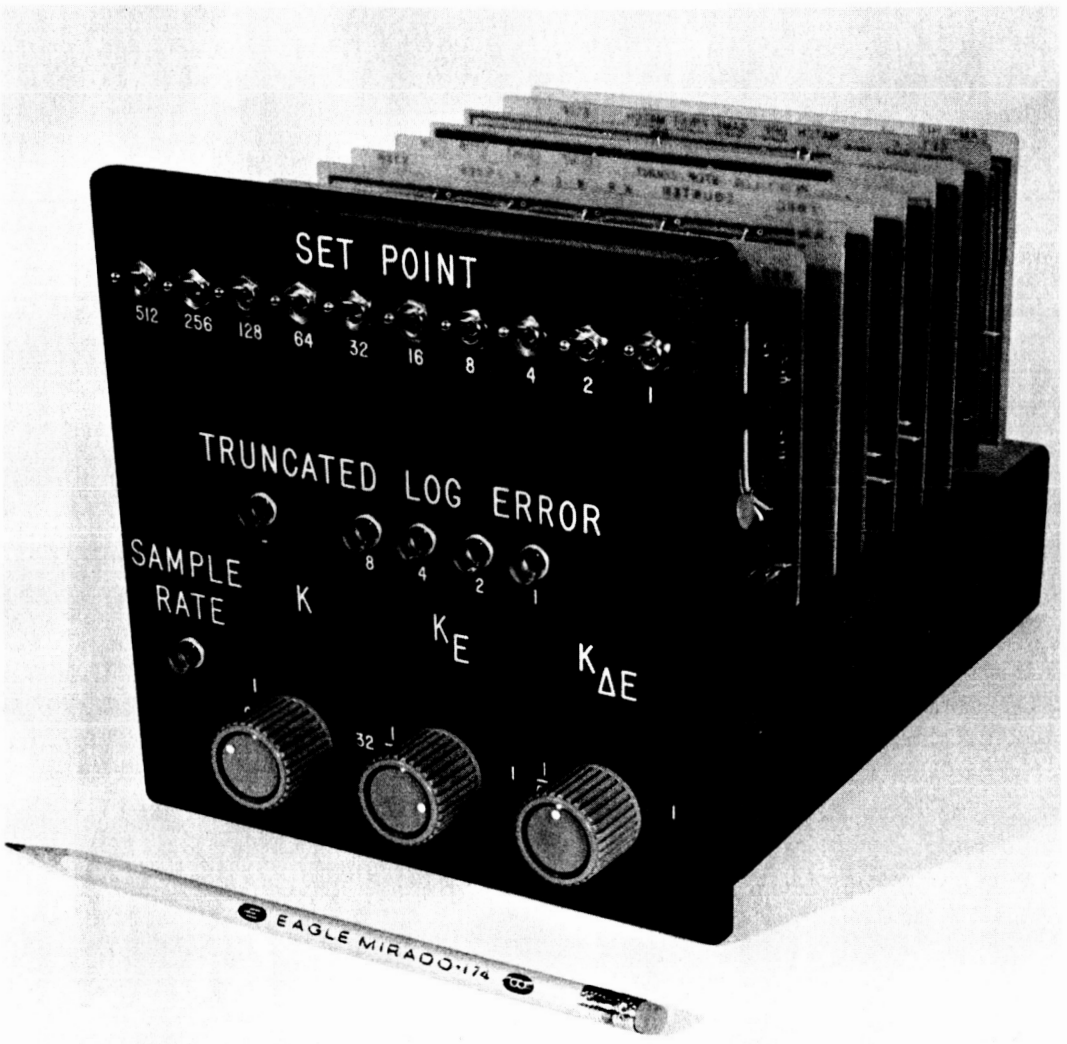


FIGURE 3.20 THE DIGITAL CONTROLLER

Of the eight printed circuit boards constructed, the most densely packed, and therefore the one which determined the 4" x 6" size of the boards, is the Logarithmic Pulse Generator shown in Figure 3.21. The last board constructed, and thus perhaps the one with the most finished appearance, is the Error Forming Circuit of Figure 3.22.

The modular construction utilized in placing components such as flip-flops and pulse generators on the boards has the advantage of simplifying the problems of servicing. It also tends to lead to a board layout in which some areas are very densely packed with components while other areas are quite bare. This may be considered a disadvantage if a more random arrangement of elements leads to smaller boards. Note for this controller that since the most dense board, the Logarithmic Pulse Generator, is also very iterative, it was probably most compactly constructed on a modular basis.

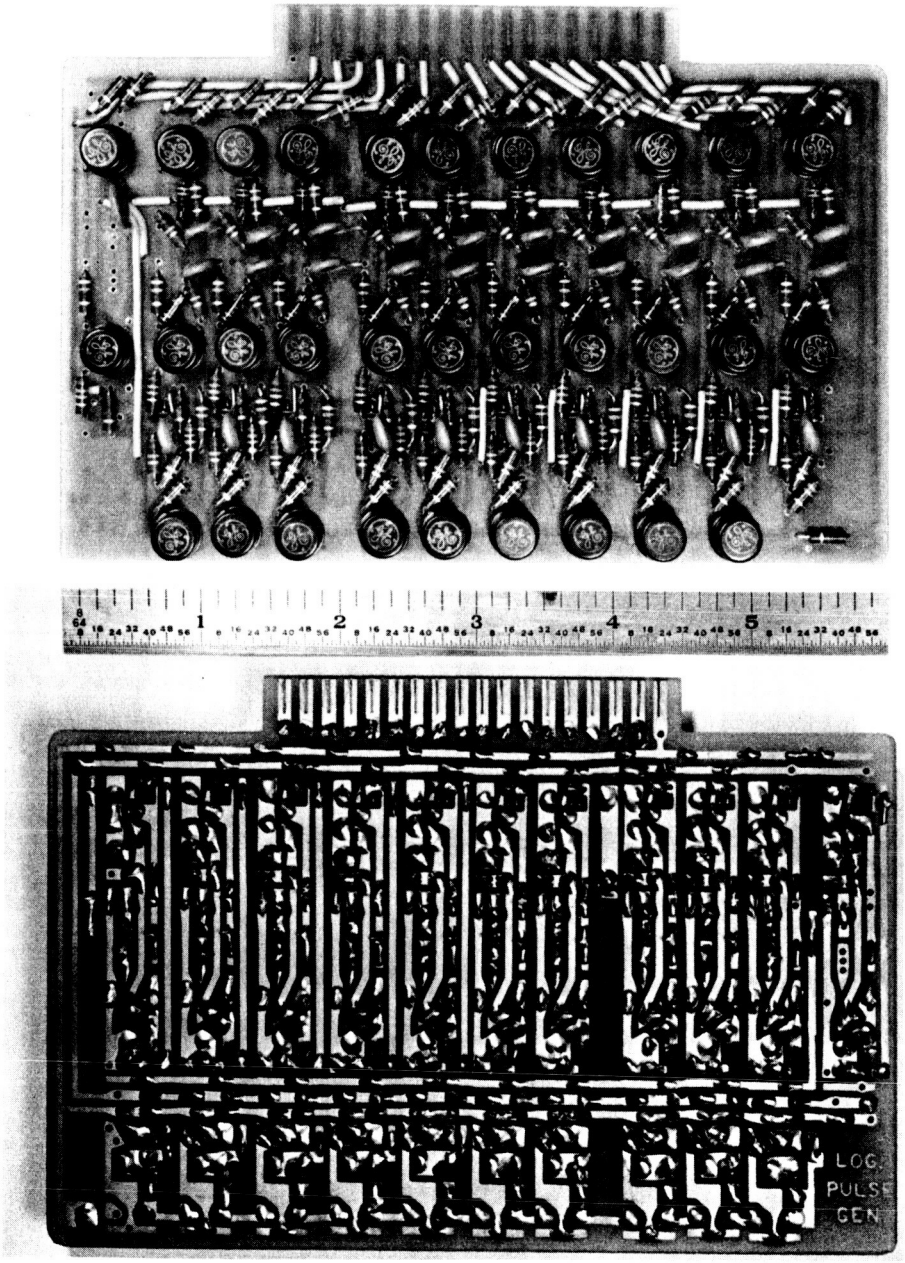


FIGURE 3.21 LOGARITHMIC PULSE GENERATOR CARD

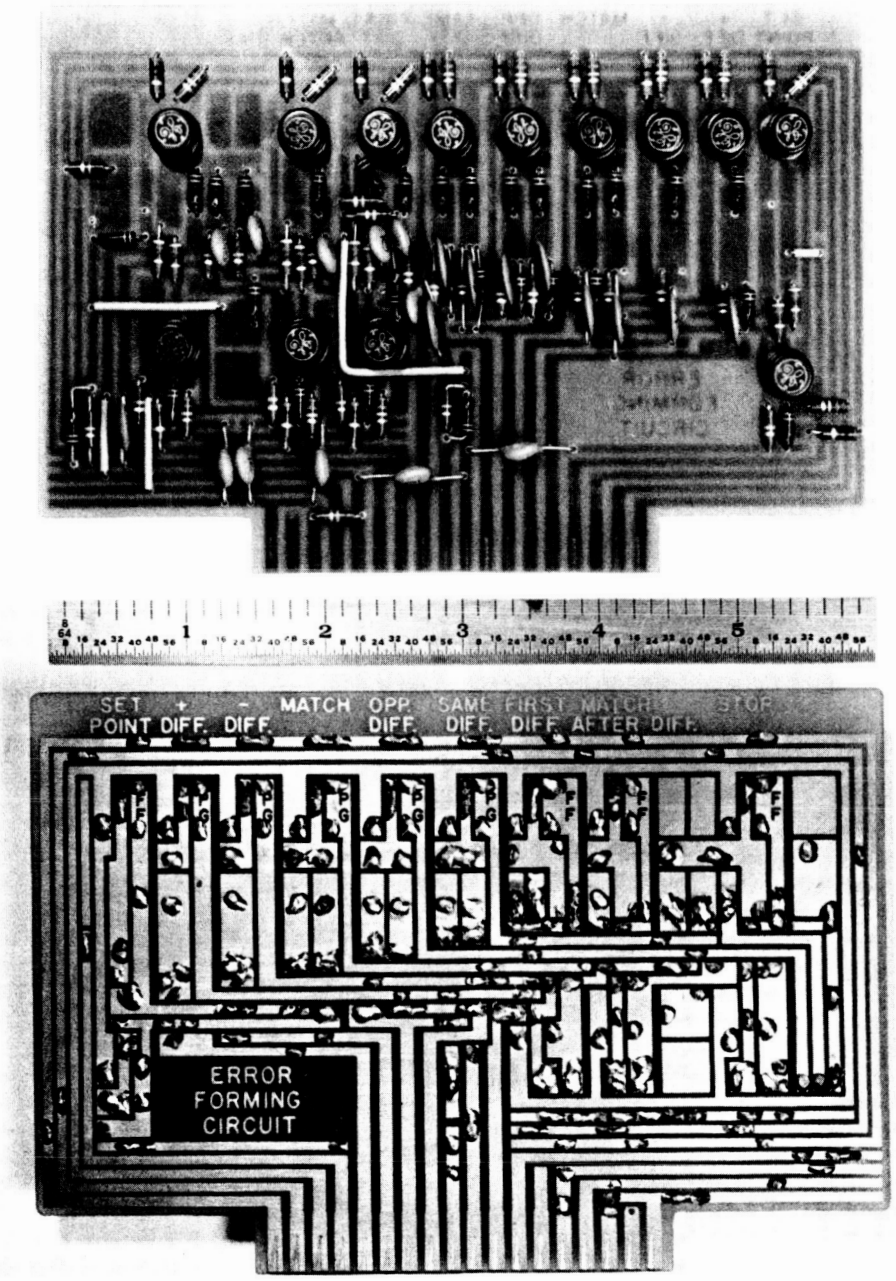


FIGURE 3.22 ERROR FORMING CIRCUIT CARD

CHAPTER IV

EXPERIMENTAL RESULTS

Introduction

In this chapter the digital controller is applied to the control of a two-capacity liquid level plant. After a description of the plant is presented, consideration is given to the adjusting of the four controller parameters, K, K_E, $K_{\Delta E}$, and T. Then the closed loop performance of the system is investigated.

Plant Characteristics

The two-capacity liquid level plant is shown in Figure 4.1. The water level in the lower barrel is monitored with a float which drives the digital encoder. The flow rate into the upper barrel is adjusted by the pulses from the digital controller which step the stepping motor and thus also the valve connected to the water supply. This plant is a fair approximation to many industrial processes, and as such it provides a valid testing ground for the controller.

The plant dynamics are investigated in Appendix III

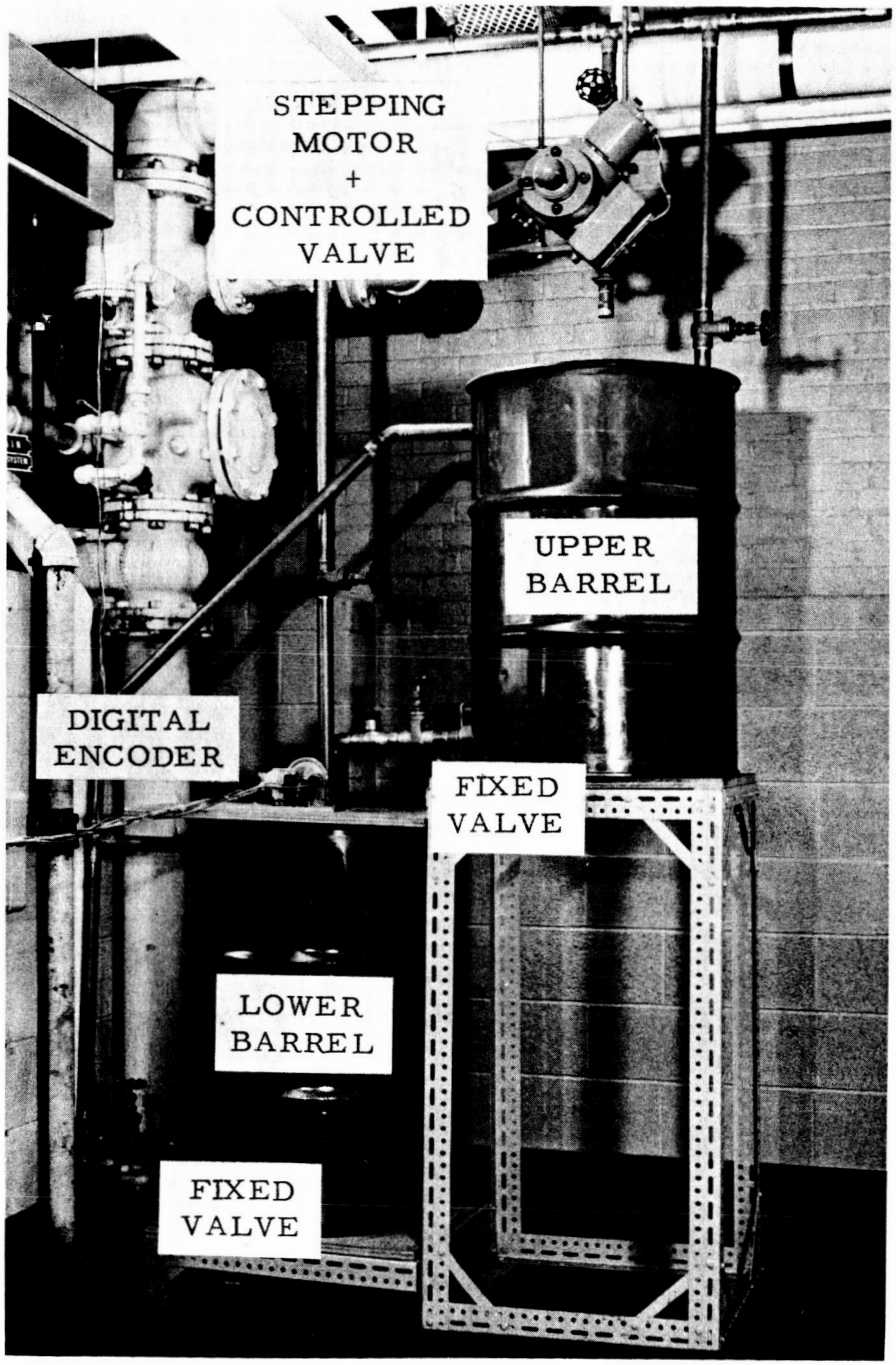


FIGURE 4.1 THE EXPERIMENTAL PLANT

where a linearized model is derived. For convenience the transfer function is repeated here:

$$\frac{\Theta_o(s)}{M(s)} = \frac{K_p}{(1 + T_p s)^2} \quad \frac{\text{output quanta}}{\text{step}} \tag{4-1}$$

where $K_p = 0.72$ (4-2)

and $T_p = 428$ seconds (4-3)

Adjusting the Controller Parameters

The controller has been designed to implement equation 2-5, repeated here:

$$\Delta m_n = K \left[K_E E_n^* + K_{\Delta E} (E_n^* - E_{n-1}^*) \right] \tag{4-4}$$

In the implementation of this equation, the parameters can take on the following values

$$K = 0, 1, 2, 4, 8, 16, 32 \tag{4-5}$$

$$K_E,\ K_{\Delta E} = 1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64 \tag{4-6}$$

The sampling rate pulse generator is designed to provide sampling rates in the range

$$3 \text{ seconds} < T < 240 \text{ seconds} \tag{4-7}$$

However the lower limit on the sampling rate is contingent upon the speed of output generation as follows. The rate of the output rate pulse generator, f pulses/second, cannot be greater than the maximum rate at which the stepping motor

can step reliably, or else the stepping motor will not be a reliable adder-subtractor. Now during the operate mode every fourth pulse from this generator goes to the Logarithmic Pulse Generator, and the operate mode lasts until 1024 pulses have been counted there. Consequently the sampling interval must be greater than

$$\frac{1024 \times 4 \text{ pulses}}{f \ \frac{\text{pulses}}{\text{second}}} = \frac{4096}{f} \text{ seconds} \qquad (4\text{-}8)$$

In addition the preparation mode may last as long as it takes to transmit the ten serial bits of information from the digital transducer to the controller, perhaps one second.

In order to adjust these parameters, consider the photograph of the controller, Figure 3.20. Note that K, K_E, $K_{\Delta E}$, and T are all adjustable on the front panel. The adjustment for T is made by turning the recessed potentiometer until the time interval between two lightings of the SAMPLE RATE light is equal to T.

The output rate, which must be tailored to the stepping motor, can be set once and for all on a specific process. Accordingly its adjustment is made by turning a Trimpot mounted on the Output Gating functional block and monitoring the output rate pulse generator by connecting a scope to the

terminals located on top of the chassis.

In setting the controller parameters, note that there is some redundancy between K, K_E, and $K_{\Delta E}$. For example, the settings K = 8, K_E = 1 and $K_{\Delta E}$ = 1/4 yield the same controller equation 4-4 as the settings K = 16, K_E = 1/2, and $K_{\Delta E}$ = 1/8. For small inputs or disturbances the above sets of settings are truly equivalent. However for large inputs the controller cannot generate more than $1024\,K_E$ output pulses corresponding to the term $K K_E E_n^*$, or $1024\,K_{\Delta E}$ output pulses corresponding to either of the terms $K K_{\Delta E} E_n^*$ or $K K_{\Delta E} E_{n-1}^*$. Consequently, given values of $K K_E$ and $K K_{\Delta E}$, it is advisable to eliminate the redundancy by making K_E and $K_{\Delta E}$ as large as possible within the range of equation 4-6 and then setting K accordingly.

The view taken here for obtaining a set of controller settings, given a specific plant, is to use any quick rule of thumb to get some settings that give fairly good control and then to iterate toward better control by checking the actual response to specific settings. One rule of thumb, presented by Grinten,[6] gives values for the parameters of a proportional plus integral linear continuous controller. By using these

together with the approximate equivalence relations for this controller, equation 2-14 through 2-16, the following relations are derived

$$K K_E = \frac{\sqrt{2}\ T}{K_p T_{dead}} \tag{4-9}$$

$$K K_{\Delta E} = \frac{\sqrt{2}\ T_{lag}}{K_p T_{dead}} \tag{4-10}$$

where T is the sampling period and where K_p, T_{lag}, and T_{dead} are defined from an approximation to the unit step response of the plant of the form:

$$q(t) = K_p \left[1 - e^{-\frac{t - T_{dead}}{T_{lag}}} \right] u(t - T_{dead}) \tag{4-11}$$

Figure 4.2 shows the approximation by this model to the plant step response shown in the Appendix Figure A.3.2.

The sampling period, T, should be chosen to be small relative to $T_{dead} + T_{lag}$ in order to approximate continuous control.

Closed Loop Performance

Using the values for K_p, T_{dead}, and T_{lag} found from Figure 4.2, the following equations are obtained by substitution into equations 4-9 and 4-10:

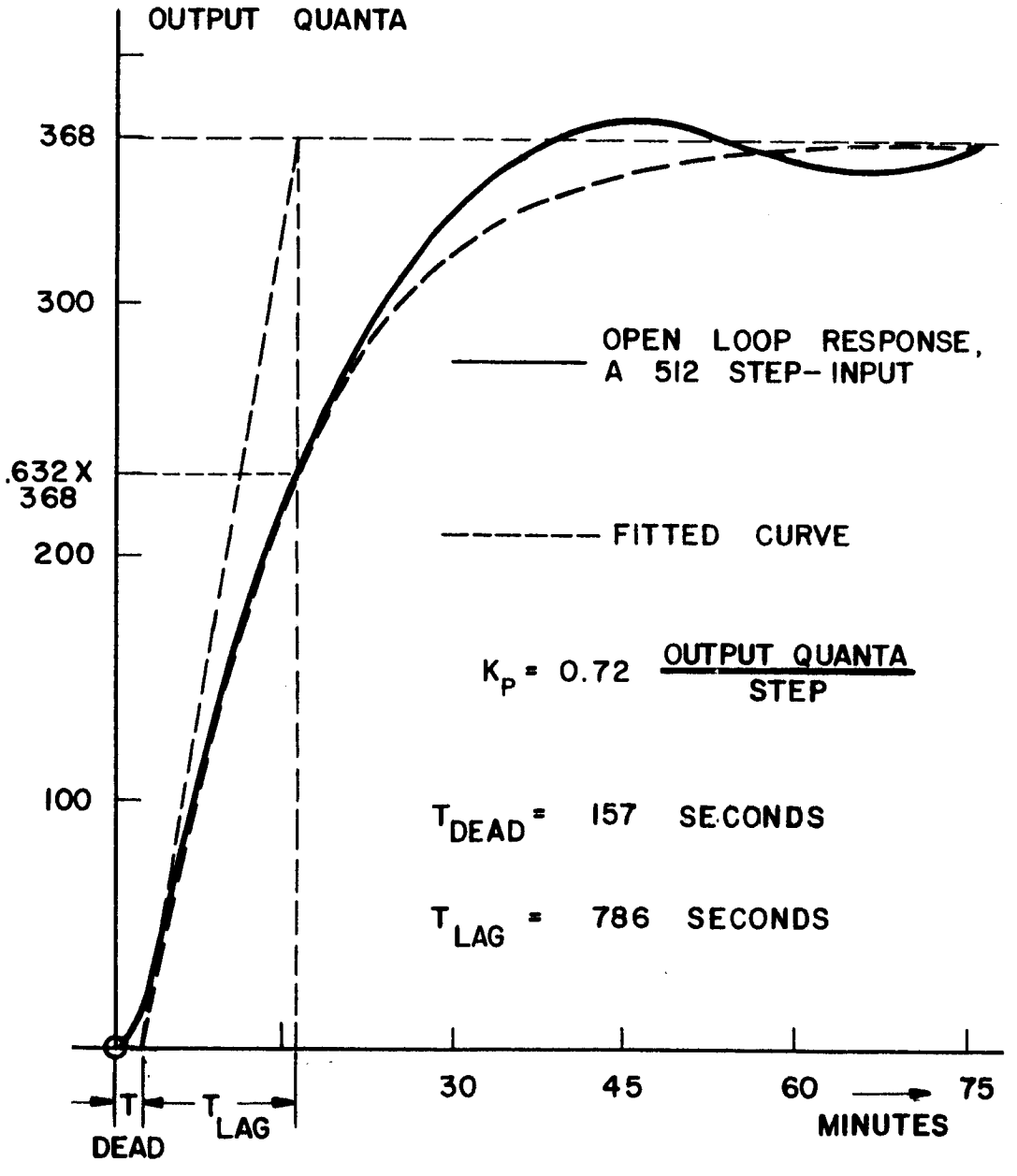


FIGURE 4.2 APPROXIMATION TO THE PLANT STEP RESPONSE

$$K K_E = \frac{T}{80} \tag{4-12}$$

$$K K_{\Delta E} = 9.8 \tag{4-13}$$

The controller parameters can be made to fit equation 4-12 exactly, but equation 4-13 can only be approximated. This is all right since the purpose of the "rule of thumb" is only to obtain a set of approximate controller settings. Consequently a resulting set is

$$T = 80 \text{ seconds} \tag{4-14a}$$

$$K = 8 \tag{4-14b}$$

$$K_E = \frac{1}{8} \tag{4-14c}$$

$$K_{\Delta E} = 1 \tag{4-14d}$$

The response of the system to a 125 quanta step under these conditions is illustrated by the solid curve, a, in Figure 4.3. In order to appreciate the effect of each controller parameter on the system response, each of the other curves (b, c, d, and e) illustrate the response which results when one of the parameters of equations 4-14 is changed by a factor of two.

Ideal test conditions for this comparison would have the plant open loop disturbance response with no variation at all. Then the desired cause and effect relationship would

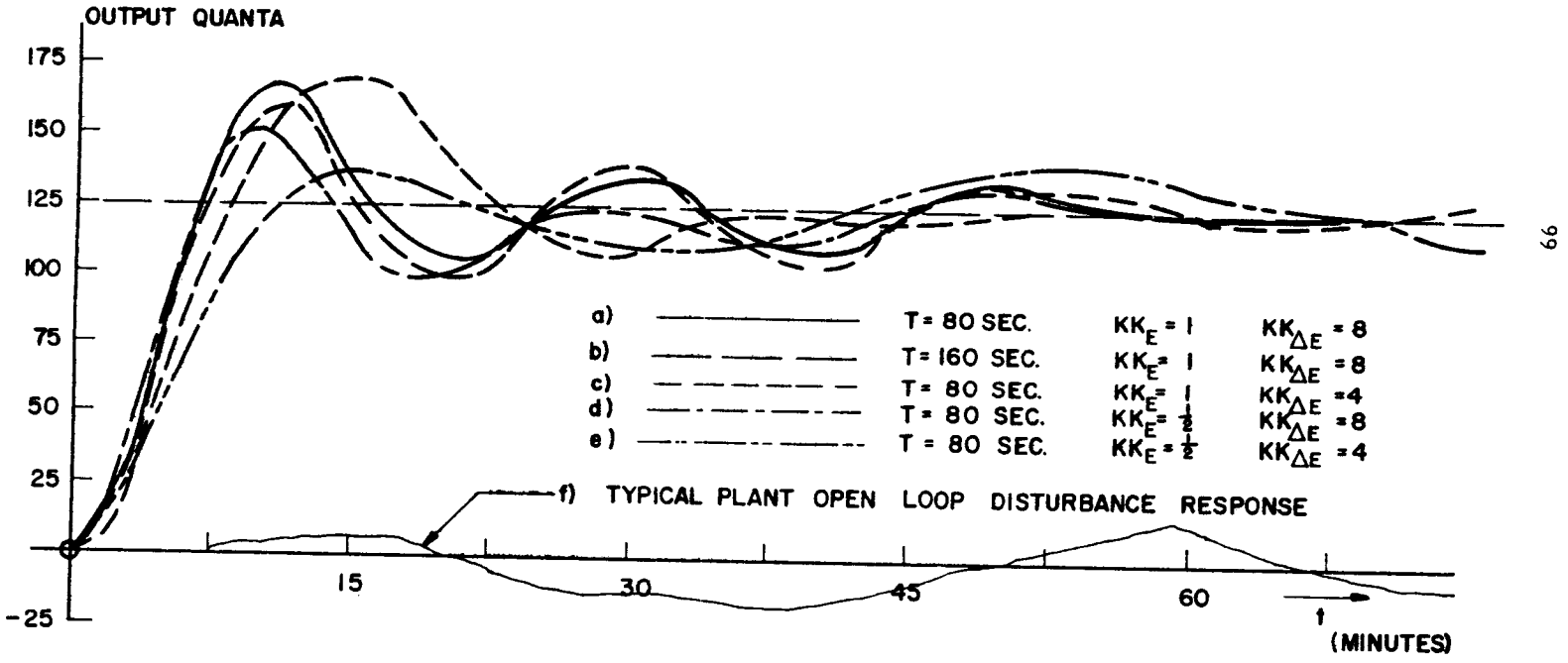


FIGURE 4.3 EXPERIMENTAL RESPONSE CURVES

lead to the exact step response of the system. However the actual open loop disturbance response is illustrated in Figure 4.3f. This indicates that the other curves are seeing not only the 125 quanta step input but also variations having an RMS value of about 5 to 10 output quanta due to disturbances in the pressure of the water supply. As long as the open loop disturbance response is slowly varying relative to the response time of the closed loop system, little of this is reflected in the closed loop response curves.

Figure 4.3c illustrates that halving the effective proportional gain of the controller (refer to equations 2-14 and 2-15) gives a slower, looser response. On the other hand Figure 4.3d illustrates that halving the effective integral gain (refer to equations 2-14 and 2-16) leads to a slightly faster, tighter response. Finally by decreasing the loop gain, as in Figure 4.3e, the response becomes relatively sluggish.

Figure 4.3b illustrates that doubling the sampling period to 160 seconds tends to give a response with the effective integral gain halved, similar to Figure 4.3d. However this response is somewhat looser than Figure 4.3d because the sampling interval is larger relative to the $T_{dead} + T_{lag}$

of 943 seconds found in Figure 4.2.

Summary

This thesis has been directed toward the logical design of a digital controller. Although it has been considered in use with a specific plant and compared to a conventional analog controller, no claim is made comparing its performance in general with that of an analog controller. The objective here was completely one of simplifying the logic involved in the controller, and the use of truncated logarithmic quantization has been justified on the basis of a simulation study and heuristic reasoning alone. A thorough analytical study of the performance of a control system utilizing truncated logarithmic quantization is beyond the scope of this thesis. However the controller appears capable of providing good control in those applications where a linear, proportional plus integral controller would also provide good control.

The design of the digital controller has been described and the complete operation divided into eight functional blocks, each of which is implemented on a separate 4" x 6" printed circuit board. To indicate the amount of logic used in another way, the total number of transistors used is 128.

APPENDIX I

PROGRAM FOR SIMULATION STUDY

In this appendix the digital computer program used in the simulation studies will be presented and explained briefly. The program is written using the BALGOL compiler. This is the version of the international computer language, ALGOL, which is used on the Burroughs 220 computer. Figure A.1.1, a copy of the program, is most easily understood by referring to Figure A.1.2, the corresponding flow diagram.

To start, there are four things the computer must do before going into the double iteration loop represented by the two diamonds. Data must be read in, values of the variables must be initialized, the plant parameters must be calculated, and the heading for the output format must be written.

As the computer enters the double iteration loop for the first time, the two indices, I and J, are set to 2 and 1 respectively. The I index accounts for the successive samples in the sampled data control system, while the J index distinguishes between the linearly quantized system (J = 1) and the truncated logarithmically quantized system (J = 2). The

```
    BAC-220 STANDARD VERSION   2/1/62
COMMENT      JOHN PEATMAN          CONTROLLER SIMULATION $

COMMENT   THIS PROGRAM SIMULATES THE CONTROL OF A CONTINUOUS
LINEAR PLANT USING  TWO  CONTROLLERS WHICH OPERATE ON THE
SAMPLED AND QUANTIZED ERROR BETWEEN THE SET POINT AND THE
OUTPUT.  THE CONTROLLERS PRODUCE AN OUTPUT DEPENDING ON THE
ERROR AND THE FIRST DIFFERENCE OF THE ERROR.  THE FIRST
CONTROLLER USES LINEAR QUANTIZATION WITH A QUANTUM SIZE OF
ONE.  THE   SECOND  CONTROLLER USES A TRUNCATED LOGARITHMIC
QUANTIZATION BASED UPON THE MOST SIGNIFICANT ONE IN THE
ERROR.  $

INTEGER  I, J, G, H,    QUANT(,), LOGKE, LOGKDELE, M(,)  $
ARRAY  QUANT(2,105), C(2,105), DC(2,105), M(2,105)  $
INPUT DATA (T, LOGKE, LOGKDELE, CSTART) $
OUTPUT HEADING (KE, KDELE) $
OUTPUT OUT ( (I-1)T, QUANT(1,I+1), C(1,I+1), DC(1,I+1),
     QUANT(2,I+1), C(2,I+1), DC(2,I+1)  )  $
FORMAT HEADINGFORM ( W3, *KE = *, X8.5, B15, *KDELTAE = *,
     X8.5, 3W0, B17, *SYSTEM WITH LINEAR QUANTIZATION*, B9,
     *LOG QUANT. + DECODING*, W4, B3, *NT*, B8,
     2(B5, *QUANTIZED*, B4, *C*, B5, *DERIVATIVE*), W4, B10,
     2(B10, *ERROR*, B14, *OF  C*), W0 ) $
FORMAT OUTFORM (    X10.5, 2(I14, X10.1, X10.1), W4 ) $


PROCEDURE LOGQUANT ( CABIN $ LOGCABIN ) $ BEGIN
  COMMENT   THIS FORMS A TRUNCATED LOGARITHMIC QUANTIZATION
  PLUS A SIGN.  THAT IS, LOGCABIN IS AN INTEGER EQUAL TO THE
  POSITION OF THE MOST SIGNIFICANT ONE IN THE BINARY
  REPRESENTATION OF CABIN (OR ZERO IF CABIN IS LESS THAN
  UNITY).  THE SIGN OF LOGCABIN IS THE SAME AS THE SIGN OF
  CABIN $

  INTEGER LOGCABIN, TEST, TRUNC $

  LOGCABIN = 0 $
  TEST = 1 $
  TRUNC = ABS(CABIN) $
REPEAT..  IF TRUNC GEQ TEST $
    BEGIN TEST = TEST . 2 $
    LOGCABIN = LOGCABIN + 1 $
    GO TO REPEAT  END $
  LOGCABIN = SIGN(CABIN) . LOGCABIN $
RETURN END LOGQUANT() $


PROCEDURE ANTILOG ( LOGCABIN $ CABIN ) $ BEGIN
  COMMENT   THIS IS THE INVERSE OF LOGQUANT.  IT PRODUCES
  AN INTEGER WHICH IS A POWER OF TWO (OR ZERO).  THE SIGN OF
  CABIN IS THE SAME AS THAT OF LOGCABIN $

  INTEGER  LOGCABIN, CABIN $

  EITHER IF LOGCABIN NEQ 0 $
  CABIN = SIGN(LOGCABIN) . ( 2 * ( ABS(LOGCABIN) - 1 ) ) $
  OTHERWISE $ CABIN = 0 $
RETURN END ANTILOG() $


START..  READ ($$DATA) $
         KE =  2.0 * LOGKE $
         KDELE = 2.0 * LOGKDELE $
         GAINADJ = 1.0 / SQRT(2.0) $
         I = 1 $
         QUANT(1,2) = QUANT(1,1) = -CSTART $
         LOGQUANT ( -CSTART $ QUANT(2,1) ) $
         QUANT(2,2) = QUANT(2,1) $
         ANTILOG ( QUANT(2,1) $ QUANT(2,1) ) $
         M(1,1) = M(2,1) = 0  $

         A = EXP(-T) $
         B = T.A $
         D = A+B $
         E = A-B $
         F = 1-D $
         C(2,2) = C(1,2) = CSTART  $
         DC(2,2) = DC(1,2) =  0.0  $

         WRITE ($$ HEADING, HEADINGFORM) $
         WRITE ($$ OUT, OUTFORM ) $

         FOR I = (2, 1, 100) $
           BEGIN FOR J = (1, 1, 2) $
             BEGIN EITHER IF J EQL 2 $
               BEGIN K = 1.0 $
               ANTILOG ( QUANT(2,I) $ QUANT(2,I) )  END $
             OTHERWISE $ K = GAINADJ $
             M(J,I) = K . ( KE . QUANT(J,I) + KDELE . (QUANT(J,I)
                  - QUANT(J,I-1))) + M(J,I-1)   $

PLANT..       C(J,I+1) =  D.C(J,I) + B.DC(J,I) + F.M(J,I) $
             DC(J,I+1) = -B.C(J,I) + E.DC(J,I) + B.M(J,I) $

             IF J EQL 1 $
               BEGIN QUANT(1,I+1) = -C(1,I+1) $
               GO TO BETA  END $
             LOGQUANT ( -C(2,I+1) $ QUANT(2,I+1) ) $
BETA..       END $
           WRITE ($$ OUT, OUTFORM ) $

ALPHA..   END $
         GO TO START $
         FINISH $
```

FIGURE A.1.1 BALGOL PROGRAM

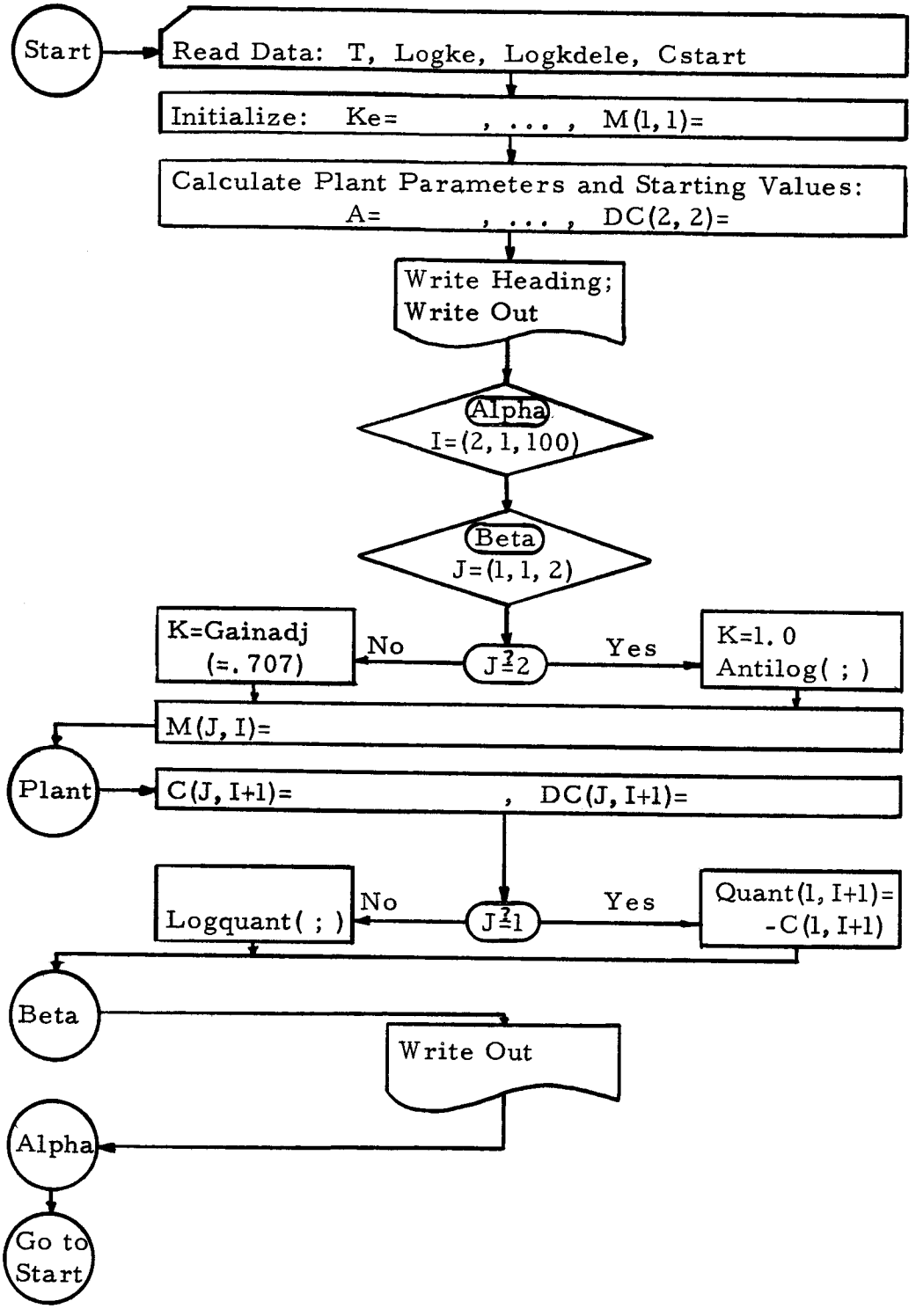


FIGURE A. 1. 2 FLOW DIAGRAM OF COMPUTER PROGRAM

computer performs instructions until the encircled Beta is reached.

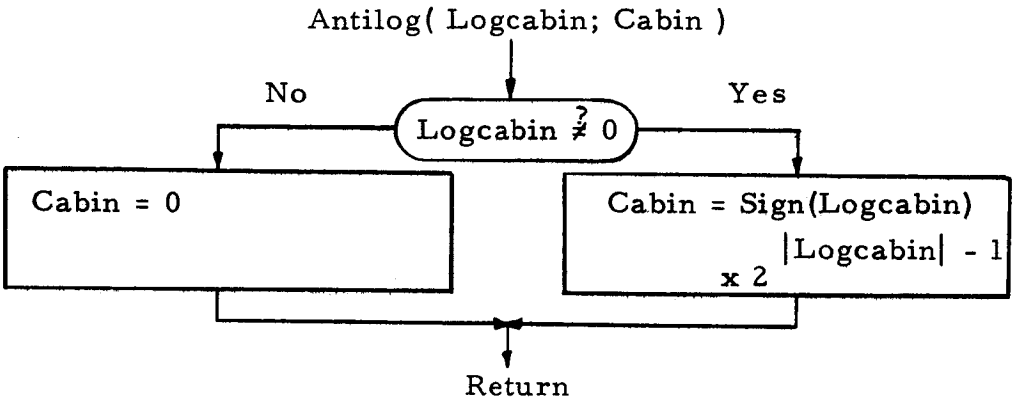
At Beta the loop test is performed for the index J. The computer asks: is J equal to (or greater than) 2? If so, continue on past the encircled Beta. If not, go back to the J diamond, increase J by one, and continue through this loop again. After going through the loop with J = 2, the computer passes through the encircled Beta, writes out the output data and reaches the I loop test point denoted by the encircled Alpha. This completes the computation and write out for both systems (J = 1 and J = 2) at one sampling interval. Now the next sampling interval is brought into consideration by increasing I by one. When I = 100 the computer passes through the encircled Alpha, returns to the start of the program and asks for new data. The program terminates when no more data is available.

Inside the loop the computer immediately comes upon the test on J represented by the oval encircling the question $J \stackrel{?}{=} 2$. The factor K then adjusts for the higher gain of the linearly quantized controller in an effort to provide similar performance from the two systems. The expression

Antilog(;) indicates the use of the subroutine shown in Figure A.1.3. This takes the integer variable written in place of the word "Logcabin" in Figure A.1.3, decodes this according to the algorithm shown, and sets the variable written in place of the word "Cabin" equal to this result.

Next, for either J = 1 or J = 2, the computer evaluates the plant input variable, M(J,I). Then the next values of the plant output, C(,), and its derivative DC(,) are calculated using equations 2-21 and 2-22 derived in Chapter II.

The computer completes the computations of the inner loop by taking one of two paths depending on whether J = 1 or J = 2. In either case the quantized system error, Quant(,), is formed. This assumes a system reference input of zero so that the error is the quantized equivalent of the output times minus one. To form the truncated logarithmically quantized error, Quant(2, I+1), the computer utilizes the subroutine described in Figure A.1.4. To form the linearly quantized error, Quant(1, I+1), the computer simply truncates that part of the error to the right of the decimal point by storing the floating point number, -C(1, I+1), in the integer address, Quant(1, I+1).

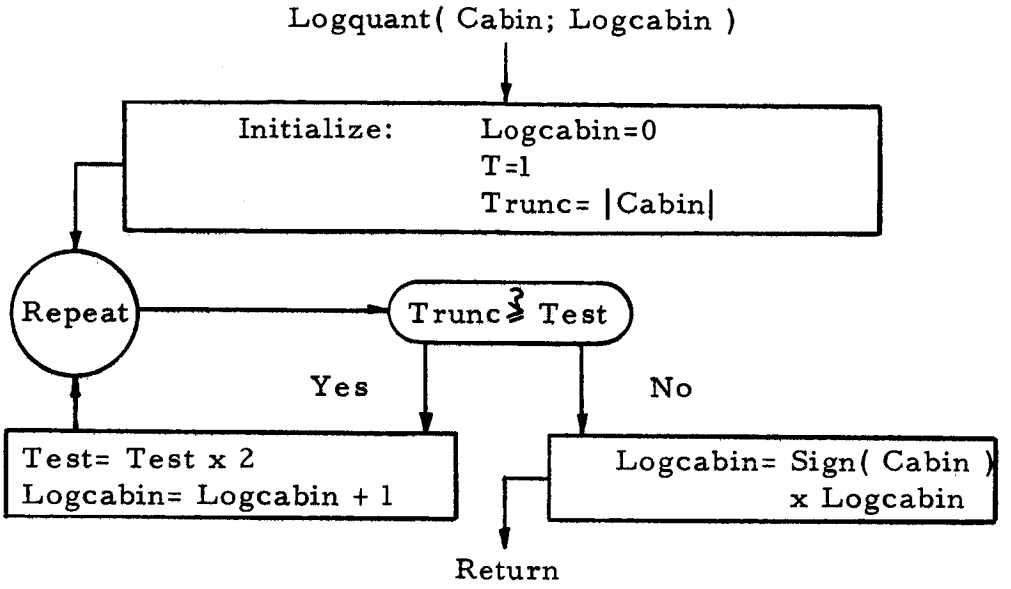


EXAMPLE: Logcabin = -4

$$\text{Cabin} = (-1) \times 2^{|-4| - 1}$$

$$= -8$$

FIGURE A.1.3 ANTILOG(;) SUBROUTINE



EXAMPLE: Cabin= -13.996 (Input)
Then Trunc= 13 since Trunc is an integer.
The subroutine proceeds as follows:

Time Through Iteration Loop	Test	Logcabin
0	1	0
1	2	1
2	4	2
3	8	3
4	16	4
4		-4

(Output is Logcabin= -4)

FIGURE A.1.4 LOGQUANT(;) SUBROUTINE

Returning to the actual computer program, Figure A.1.1, it should be noted that the controller parameters are entered as integers in logarithmic form. Consequently K_e and Logke are related by the equation

$$K_e = 2^{Logke} \qquad (A.1\text{-}1)$$

The choice of entering Logke rather than K_e was made in order to avoid the problem of obtaining the correct integer for Logke.

APPENDIX II

DETAILED CONTROL SYSTEM DESCRIPTION

Introduction

The function of this appendix is to discuss some details of the logical design of the controller (which were best omitted from Chapter III for the sake of clarity) and to discuss the characteristics of the auxillary equipment used in the experimental control system. Besides the plant, which was discussed in the beginning of Chapter IV, this auxillary equipment consists of the logic required to obtain the serial output information from the digital encoder and also the logic required to drive the stepping motor.

Controller Logic Details

One of the items which was not discussed previously in Chapter III is the complete wiring schematic of the controller. This is shown in Figure A.2.1. As was done in Chapter III, the letters encircled along the border of each functional block represent printed circuit board connector terminals. The eight-terminal connector shown at the top of this figure is mounted on the back of the chassis and provides

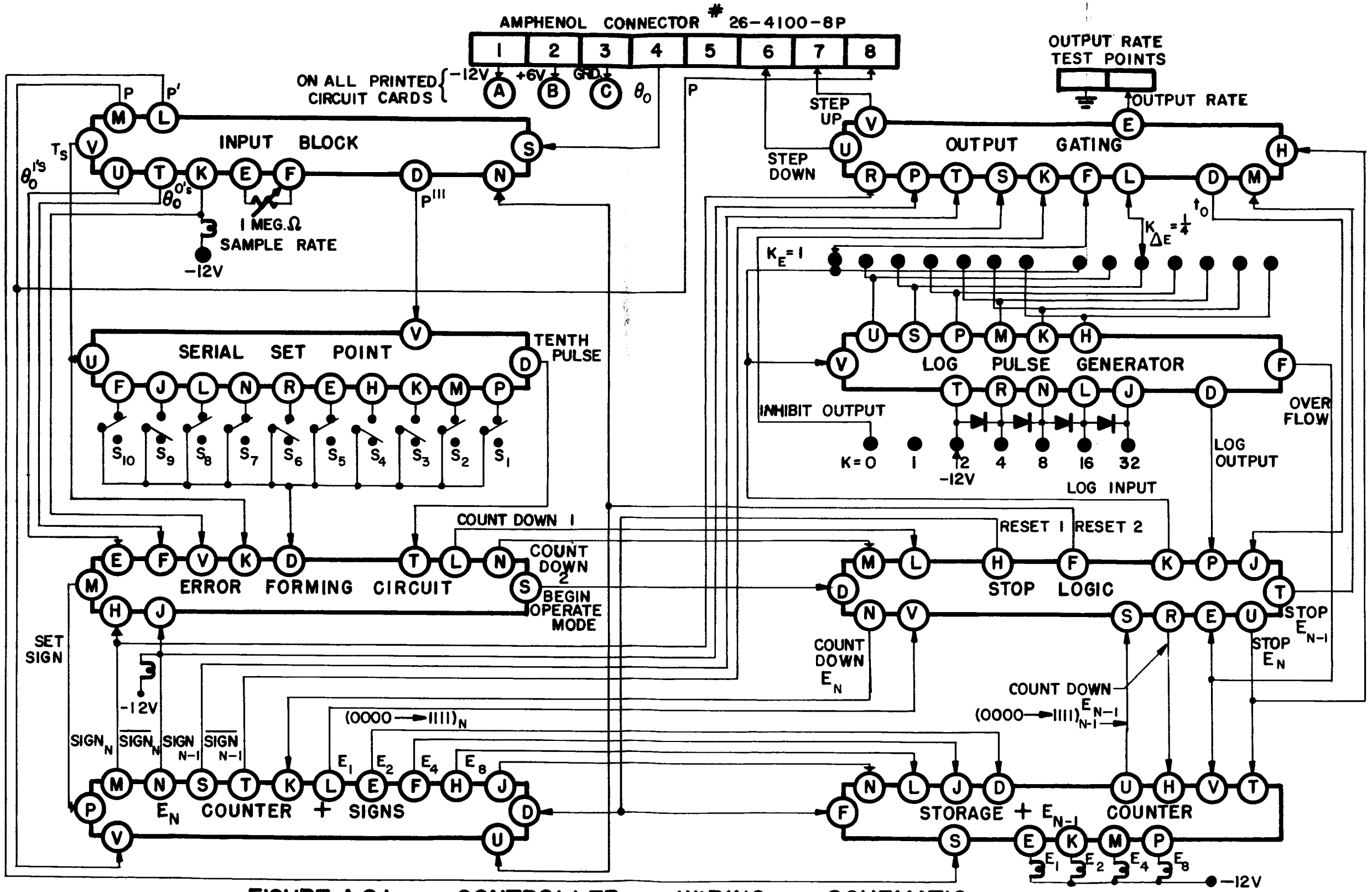


FIGURE A.2.1 CONTROLLER WIRING SCHEMATIC

the power required by the logic as well as the input and output signals of the controller.

The other item of consideration in this section is the circuitry used for those logic elements which have been previously represented symbolically. Figure A. 2. 2 shows the flip-flop circuit, while Figure A. 2. 3 shows the corresponding steering gate. If a (-)/(+) transition enters on the Set input, then the "1" Output transistor is turned off and the "1" Output goes to -12 volts. In order to trigger this flip-flop (i. e., to provide an input which will always change the state of the flip-flop) it is only necessary to tie the level inputs of two steering gates to the two outputs of the flip-flop. Then the two transition inputs of the steering gates are tied together to form the trigger input.

Figure A. 2. 4 illustrates the variety of pulse generator circuits used in the controller. Note that there is one steering gate configuration for use when the transition input is a relatively narrow pulse and another configuration for situations where this input is a (-)/(+) transition which is not closely preceded by a (+)/(-) transition. The extra diode in the first case allows the capacitor to charge quickly during the

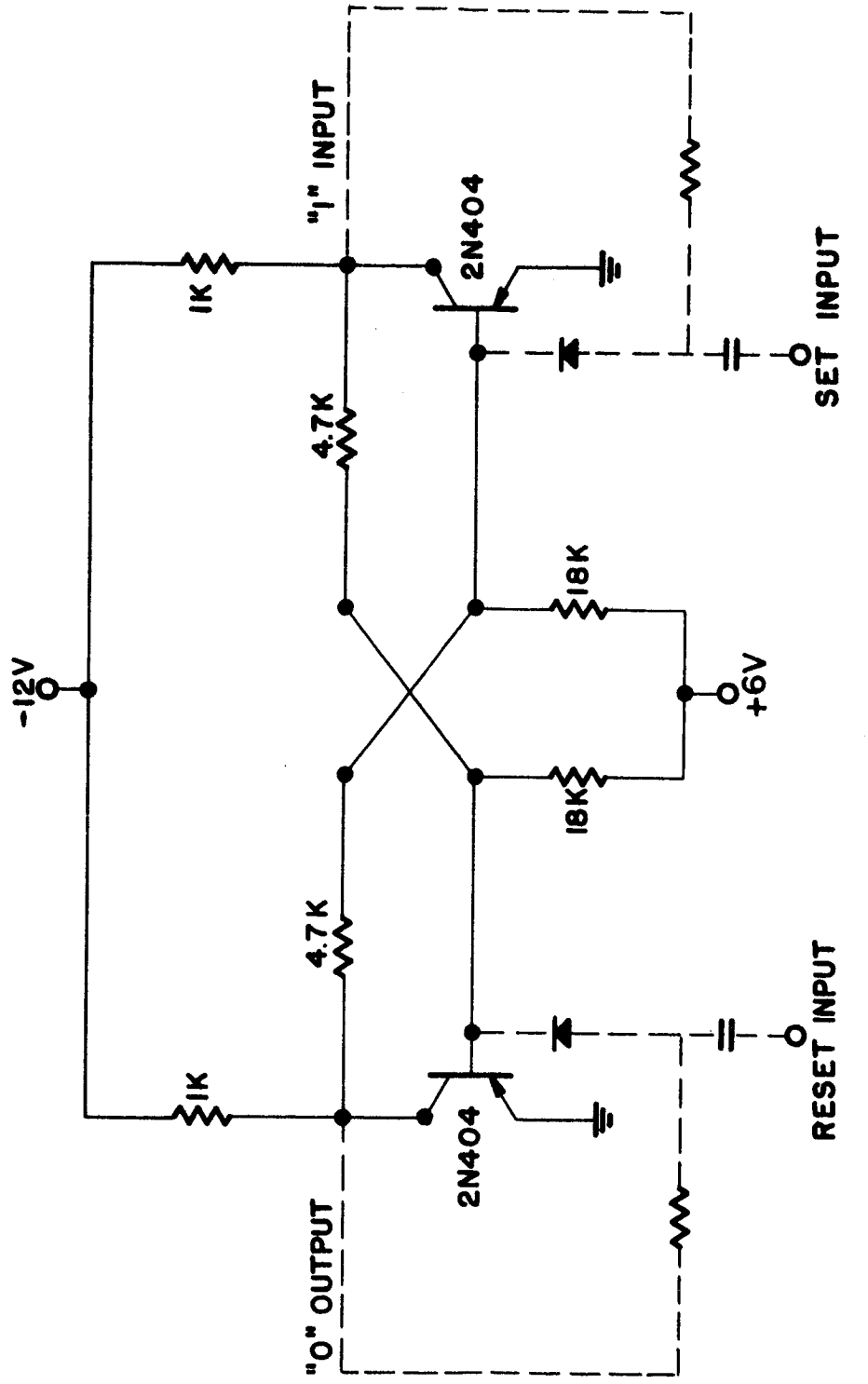


FIGURE A.22 FLIPFLOP CIRCUIT

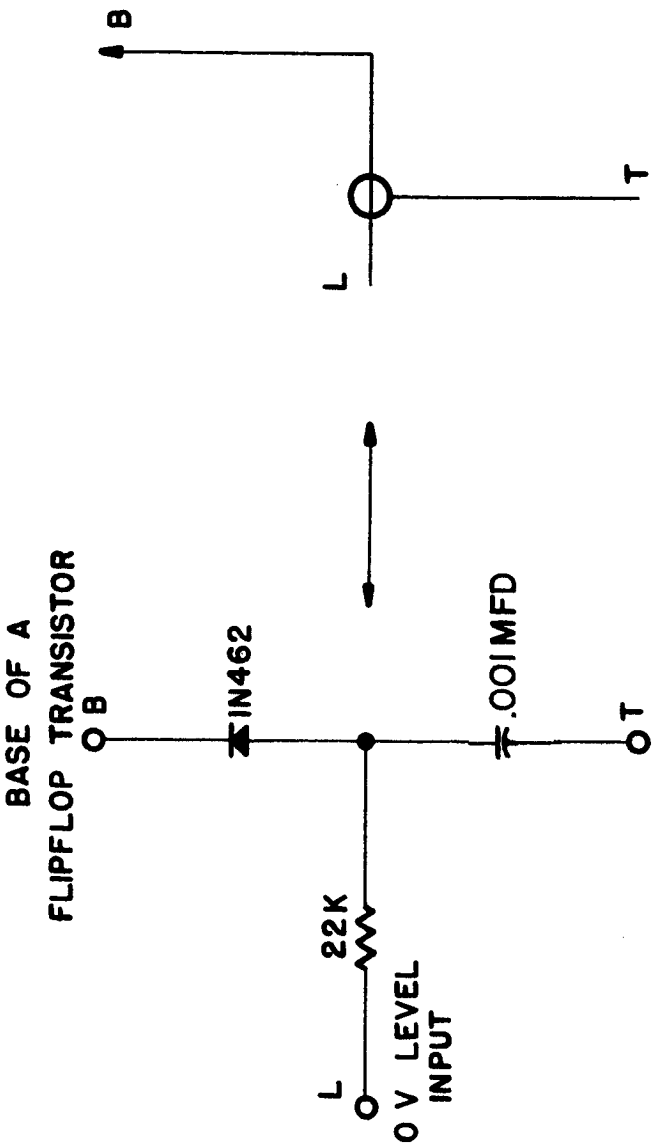


FIGURE A.23 STEERING GATE CIRCUIT

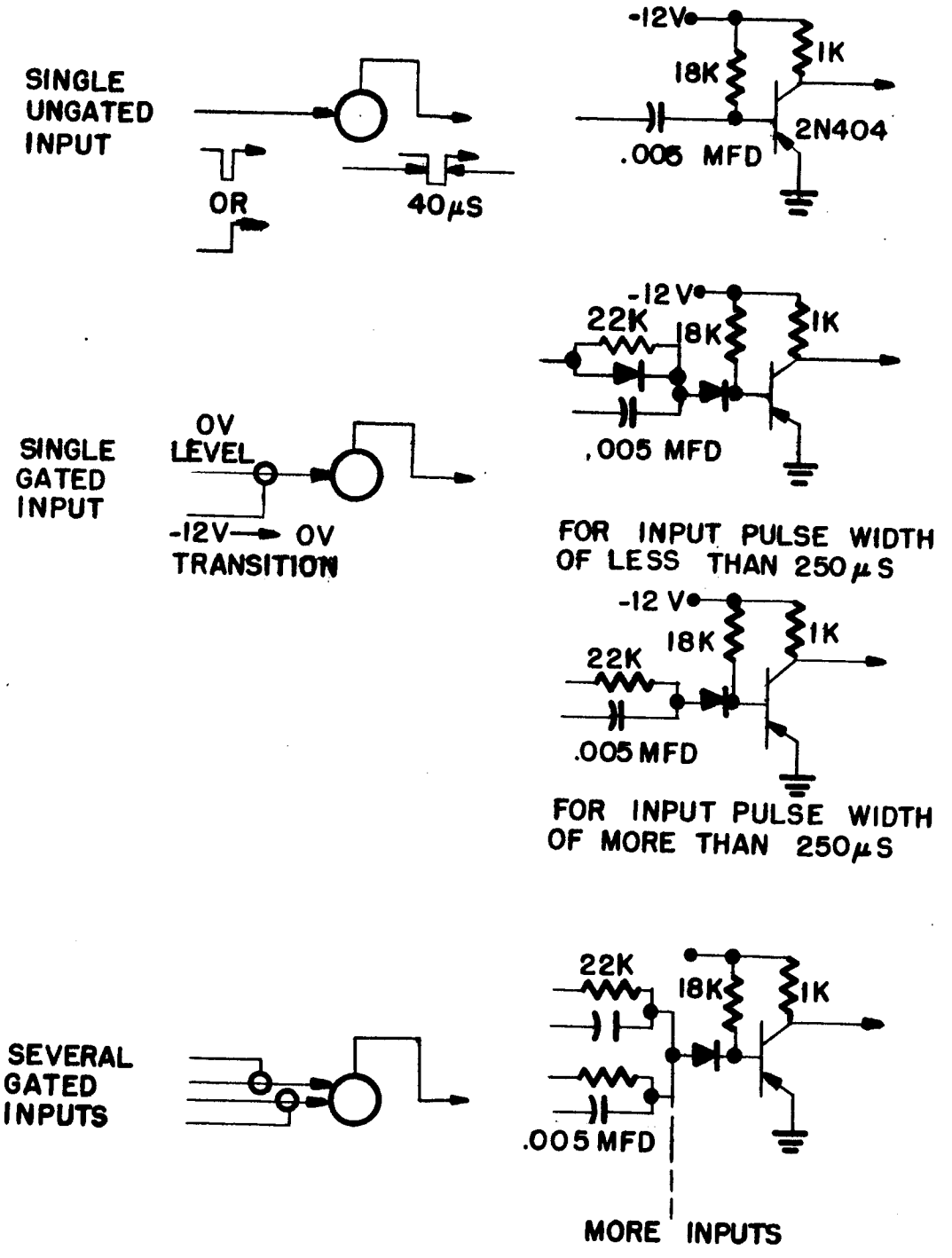


FIGURE A. 2.4 PULSE GENERATORS

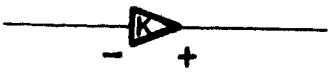
negative portion of the input pulse.

The light driver circuit, required to drive the relatively low impedance lights from the higher impedance logic circuits, is shown in Figure A. 2. 5. In this circuit the light is connected from the collector of the transistor to -12 volts. Then if the input goes to -12 volts the transistor is turned on and this turns on the light.

A resistor-diode AND gate is used in several of the functional blocks and is shown in Figure A. 2. 6. The signs on the symbol mean that if both the A input AND the B input are at zero volts then the output is at zero volts; otherwise the output is at -12 volts.

There are two specially designed free running pulse generators used in the controller. The one shown in Figure A. 2. 7 controls the output rate to the stepping motor. The unijunction transistor[7] is used in a relaxation oscillator in which the capacitor charges until this transistor fires. The voltage across the unijunction transistor drops during the capacitor discharge and this is used to form a pulse on the output. By carefully biasing the output transistor, the edges of the pulse are kept steep. The pulse width is controlled by

SYMBOL:



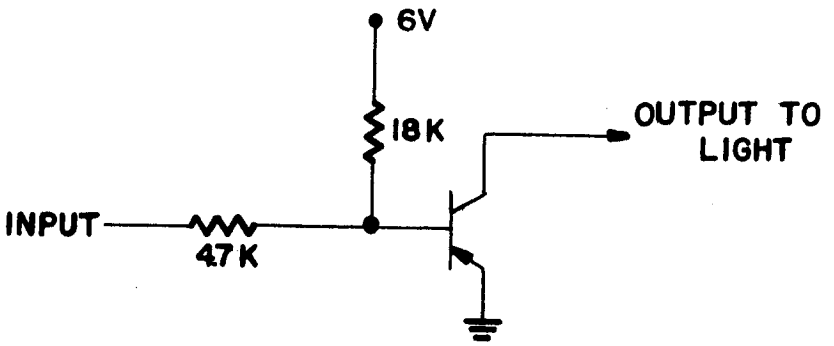


FIGURE A.2.5 LIGHT DRIVER

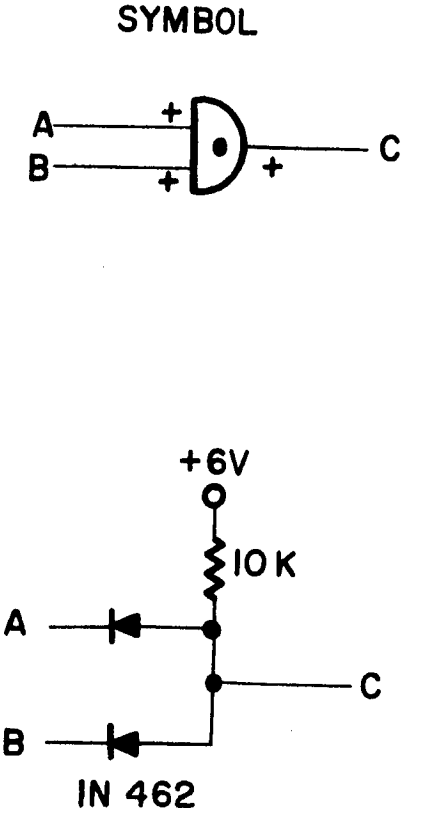
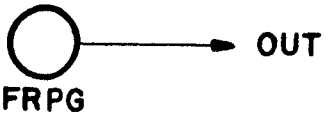
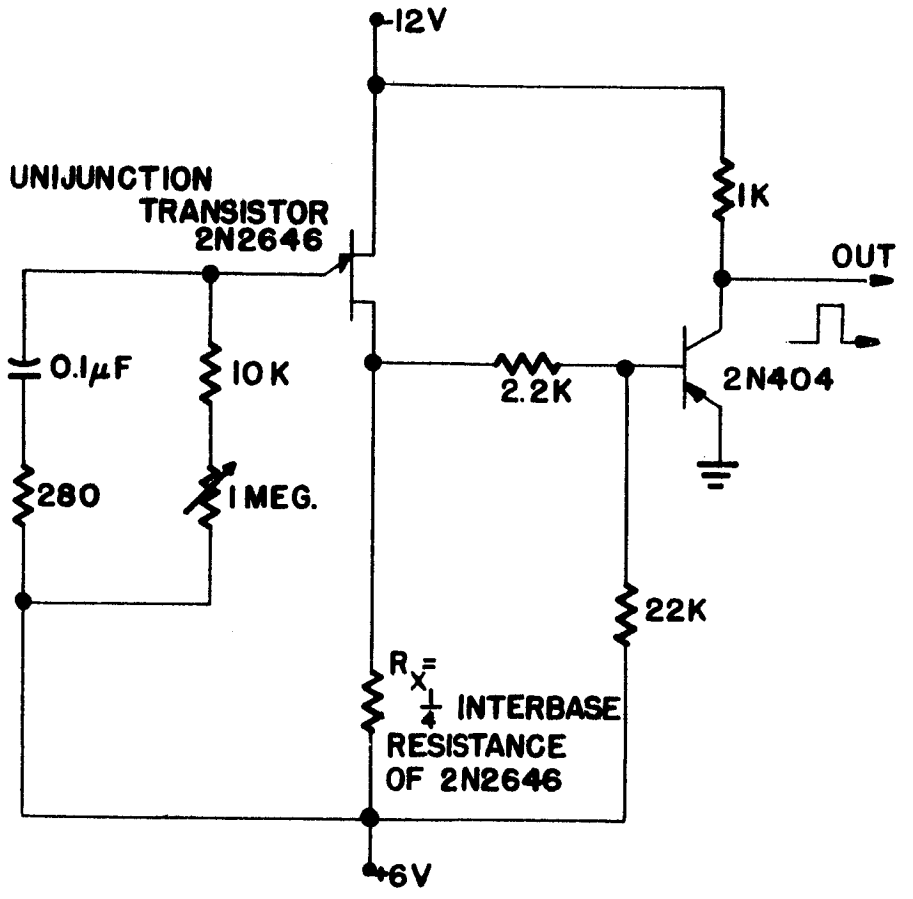


FIGURE A.2.6 RESISTOR-DIODE AND GATE

SYMBOL:





OUTPUT	PULSE	WIDTH	=	90 MICROSEC.
OUTPUT	RISE	TIME	=	0.5 MICROSEC.
PERIOD	,	MIN.	=	2.5 MILLI SEC.
PERIOD	,	MAX.	=	60 MILLISEC.

FIGURE A.2.7 FREE RUNNING PULSE GENERATOR FOR OUTPUT GATING

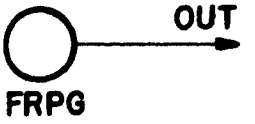
the RC time constant of the discharge. Figure A. 2. 8 shows the free running pulse generator used for the sampling rate. Here the pulse width is long because of the relatively high internal resistance of the tantalum capacitor. It is interesting to note how steep the pulse edges are in spite of this.

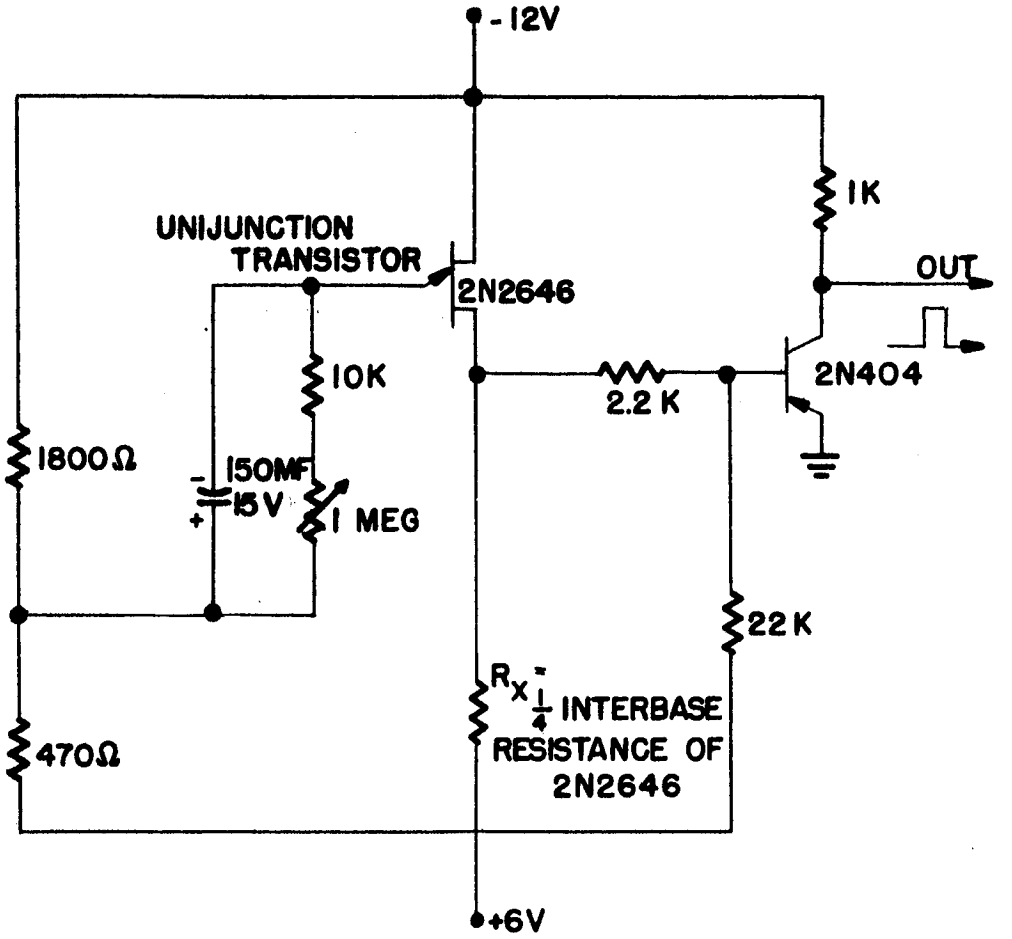
Another special purpose logic circuit is the resistor-transistor OR gate used in the Input Block and shown here in Figure A. 2. 9. The function of this gate is to provide one output pulse for each input 0 or 1 pulse. To accomplish this, two signals are first derived. One signal represents zeros as small positive pulses above ground level and the other represents ones as large positive pulses above -12 volts. The OR gate input resistances and the bias from the +6 volt supply are then chosen to switch the transistor off reliably whenever one of these pulses occurs, and to turn it on in the absence of either of these pulses.

Figure A. 2.10 shows the resistor-diode OR gate used in the Input Block. Figure A. 2.11 shows a transistor-resistor-diode AND gate used in the Serial Set Point.

The one-shot used in the Logarithmic Pulse Generator is shown in Figure A. 2.12. This circuit is the same as the

SYMBOL:





OUTPUT PULSE WIDTH = 5 MILLISEC.
OUTPUT RISE TIME = 0.5 MICROSEC.
PERIOD , MIN. = 3 SEC.
PERIOD , MAX. = 240 SEC.

FIGURE A.2.8 FREE RUNNING PULSE GENERATOR FOR INPUT BLOCK

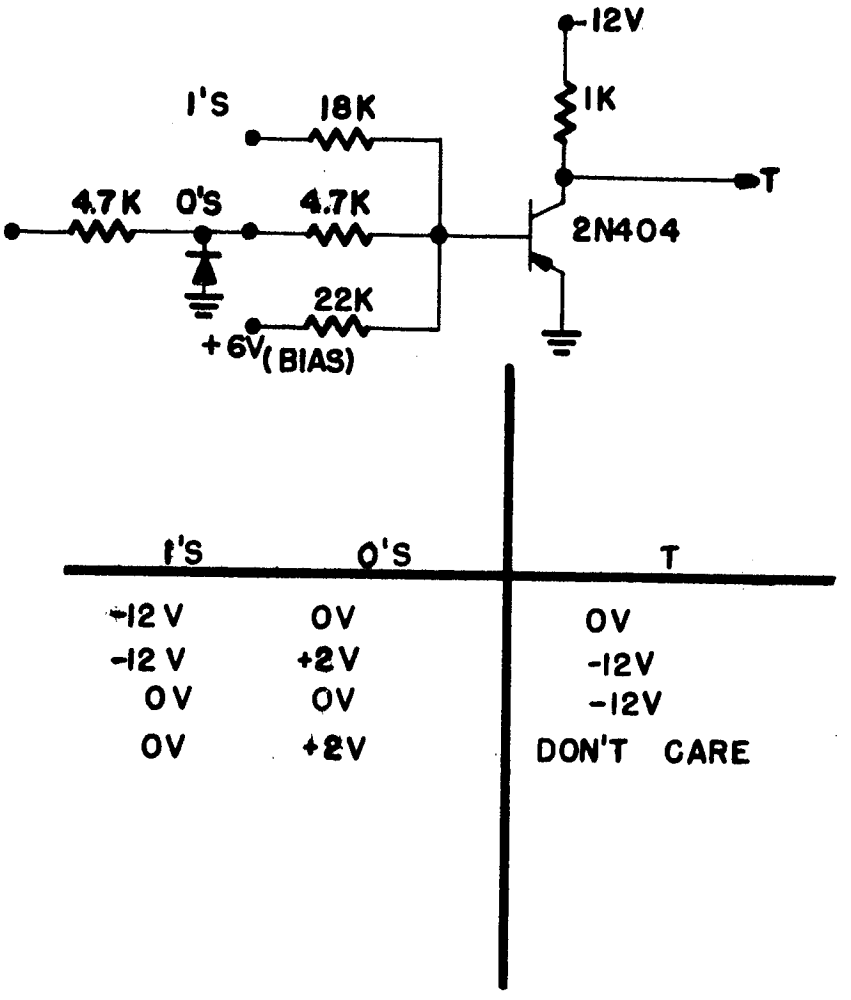
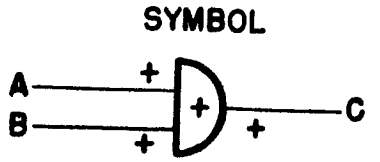


FIGURE A.2.9 RESISTOR—TRANSISTOR OR GATE FOR INPUT BLOCK



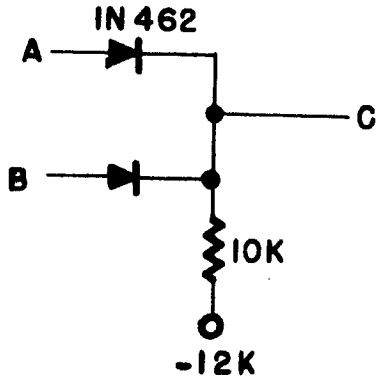


FIGURE A. 2.10 RESISTOR– DIODE OR GATE

SYMBOL:

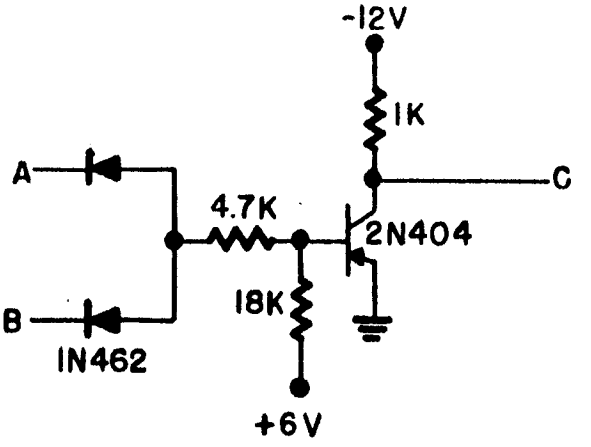
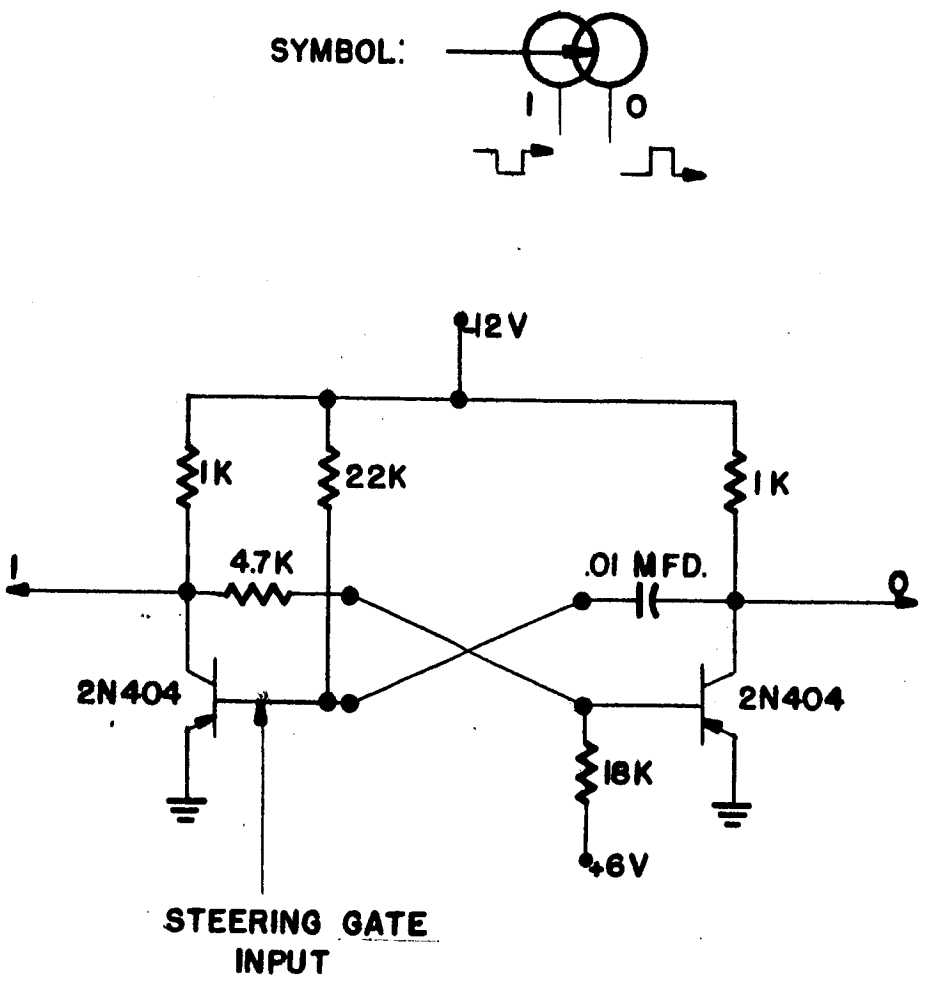


FIGURE A.2.11 RESISTOR–DIODE–TRANSISTOR AND GATE



200 MICROSEC. OUTPUT PULSE WIDTH

FIGURE A. 2.12 ONE - SHOT

flip-flop circuit but with one of the cross-over networks modified with an RC circuit to make it monostable.

Serial Output from the Encoder

One of the initial constraints placed on the design of the controller was that it should accept serial data from the encoder. The logic used to take the parallel information from the encoder and put it in serial form is shown in Figure A.2.13. The pulse, P, from the controller begins the operation by presetting a ten bit shift register with the encoder data. This pulse also resets the Stop flip-flop which then gates pulses into a pulse generator. The pulse generator, the scale-of-ten counter, and the Stop flip-flop form a little loop which allows exactly ten pulses to be emitted by the pulse generator. In this way ten shift pulses are generated which place the encoder information, one bit at a time, on the two lines labelled OUT and $\overline{\text{OUT}}$ (where these two lines are complements of each other). Each pulse is also used to set either flip-flop A or flip-flop B depending on whether the OUT signal represents a one or a zero. These flip-flops are used to control the output circuit in such a way as to obtain an output, Θ_o, of -9 volts is A is set, of +4.5 volts if B is set,

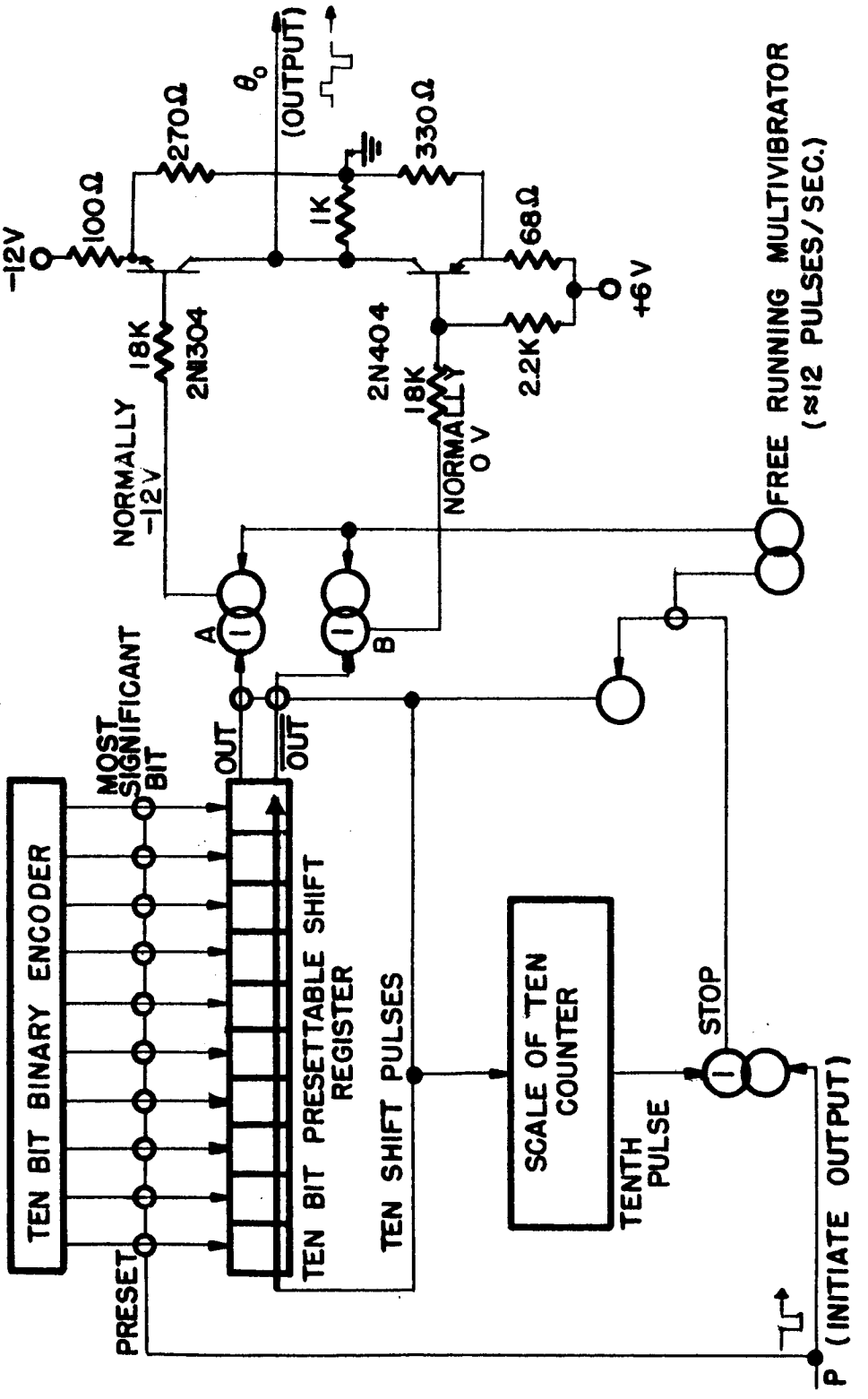


FIGURE A2.13 SERIAL OUTPUT LOGIC

and of 0 volts if neither flip-flop is set. The two output transistors operate as switches in which the 1 K ohm resistor acts as a load while the remaining resistors provide the proper biasing conditions.

Stepping Motor Logic

The stepping motor used in conjunction with the controller is bifilar wound. This means that there are four windings which are ninety degrees apart from each other electrically. For convenience, the field directions due to the energizing of these windings can be designated as 0^{o}, 90^{o}, 180^{o}, 270^{o}. Then in order to make the motor step in one direction it is only necessary to make the field direction change in increments of less than 180^{o} (to avoid ambiguity) and in either an increasing direction of field (e.g., 0^{o}, 90^{o}, 180^{o}, 270^{o}, 0^{o}, ...) or a decreasing direction of field (e.g., 0^{o}, 270^{o}, 180^{o}, 90^{o}, 0^{o}, ...).

There are three different ways in which the motor can be stepped in each direction. First, a relatively low torque stepping can be achieved by energizing one winding at a time in the sequence 0^{o}, 90^{o}, 180^{o}, 270^{o}, 0^{o}, ... An alternative approach is to energize two windings at a time to obtain

relatively high torque stepping. Here the sequence is 45^{o}, 135^{o}, 225^{o}, 315^{o}, 45^{o}, ... where, for example, the 45^{o} field direction comes from exciting the 0^{o} winding and the 90^{o} winding simultaneously. The higher torque is due to the higher field strength which results from energizing two windings at a time. A third mode of stepping alternates the above stepping modes to obtain the sequence 0^{o}, 45^{o}, 90^{o}, 135^{o}, ... This mode alternates high and low torque steps but it has the added feature of doubling the resolution of the output.

The second mode of operation is used here because of the simplicity of implementation and also because the higher torque involved permits a higher stepping speed. In order to produce the desired sequencing on the windings, note that when the 0^{o} winding is energized, the 180^{o} winding is not and vice-versa. Consequently the signals for energizing these two windings are logical complements of each other and they can be obtained from the two outputs of a flip-flop. A second flip-flop can be used for the 90^{o} winding and the 270^{o} winding. Then to step in one direction it is only necessary to join the two flip-flops into a counter which counts in the sequence 00,

01, 11, 10, 00, ... This count sequence, in which only one bit changes at a time, is called a two-bit Gray code. To step the motor in the opposite direction requires the reverse sequence, 00, 10, 11, 01, 00, ... Figure A.2.14 shows a two bit bidirectional Gray code counter. The four outputs are labelled "X." In this figure the four circuits between the four "X" points and the four "Y" points are identical. The emitter follower provides current gain to drive the higher powered transistor switch which in turn opens and closes the connection between the "Y" point and ground.

Since switching off the current in the stepping motor windings produces a large inductive "kick," the four diodes provide an alternative path in which the current can decay more slowly. However this reduces the maximum stepping speed, so the 7.5 ohm damping resistor represents a compromise between the maximum voltage which can be tolerated by the switching transistors and the stepping speed.

This circuit will step the motor continuously at rates up to 150 pulses/second before pulses are lost. However the motor may skip pulses in starting up to this rate, so 100 pulses/second represents a safe maximum rate for use as an

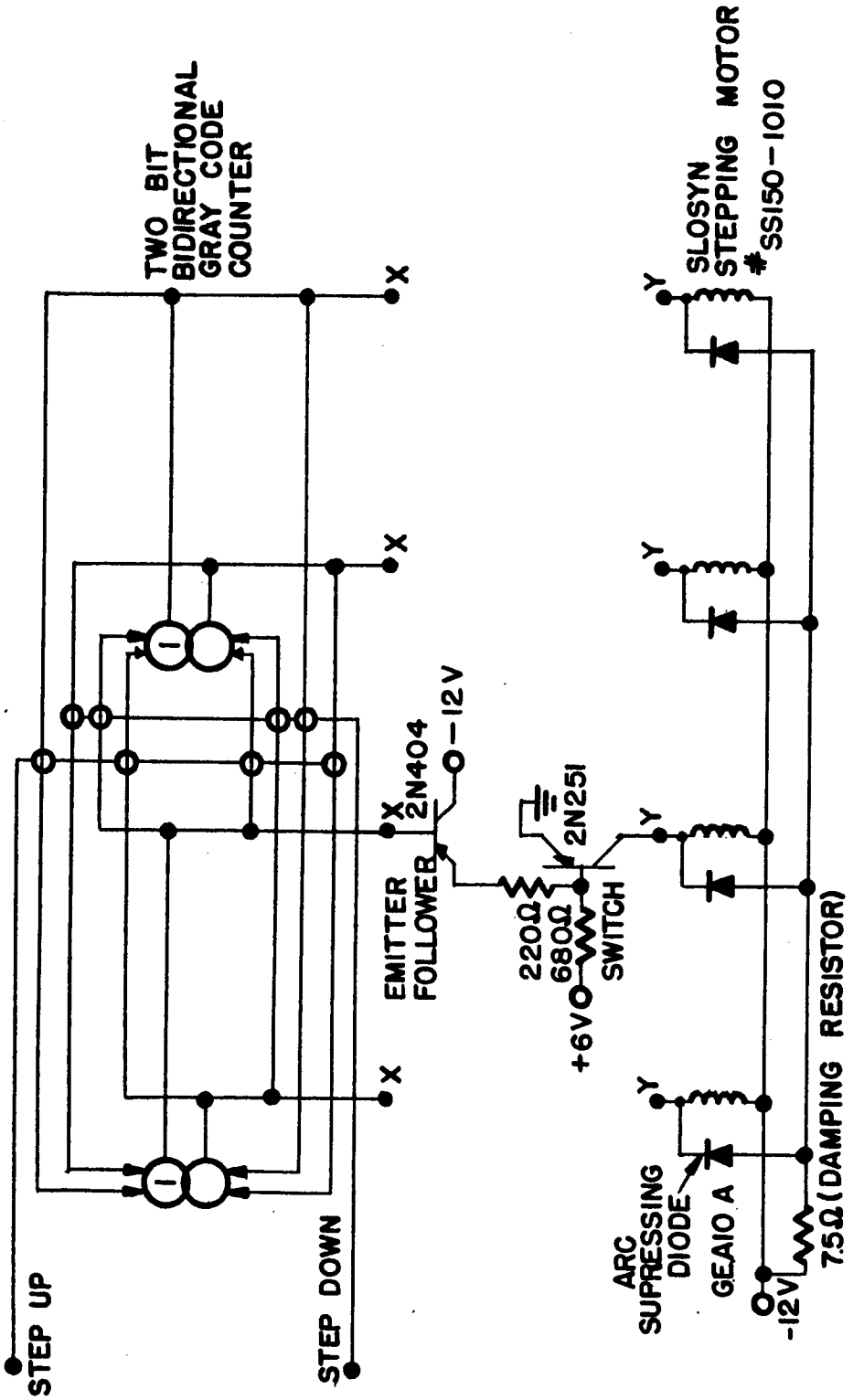


FIGURE A.2.14 STEPPING MOTOR DRIVE CIRCUIT

adder-subtractor in conjunction with the controller.

APPENDIX III

IDENTIFICATION OF THE PLANT

The Form of the Transfer Function

A common approach to this problem consists of two steps. First the form of the equation between input and output is derived from theoretical or intuitive considerations and then linearized (if necessary). Then it only remains to evaluate the parameters in this linearized form from experimental data.

The form of the equation can be derived from a few fundamental relationships for the plant, shown in Figure 4.1. First, the rate out of either barrel increases with the head in that barrel. Furthermore the head in either barrel is proportional to the integral of the difference between the input flow rate and the output flow rate. These two statements can be expressed as

$$W_{OUT} \approx K_1 \cdot \text{Head} \qquad \text{(A. 3-1)}$$

$$\text{Head} = K_2 \int (W_{IN} - W_{OUT})\, dt \qquad \text{(A. 3-2)}$$

Equation A. 3-1 is a linearized form of a nonlinear relationship and therefore to be meaningful, Head should be measured from

the operating point (e.g., the middle of the barrel). On the other hand, equation A.3-2 is an exact relationship, depending only on the liquid being incompressible and the barrels being cylindrical. In order to obtain the plant transfer function, these two equations are converted to Laplace transforms. Then for the upper barrel the ratio of W_{OUT} to W_{IN} is needed:

$$\frac{W_{OUT}(s)}{W_{IN}(s)} = \frac{1}{1 + T_u s} \qquad \text{(A.3-3)}$$

where $T_u = \frac{1}{K_1 K_2}$ (A.3-4)

For the lower barrel the ratio of Head to W_{IN} is needed:

$$\frac{\text{Head}(s)}{W_{IN}(s)} = \frac{K_L}{1 + T_L s} \qquad \text{(A.3-5)}$$

where $T_L = \frac{1}{K_1 K_2}$ (A.3-6)

and $K_L = \frac{1}{K_1}$ (A.3-7)

The time constants for the two barrels are given different subscripts because the K's may be different for the two barrels.

Finally, a relationship is needed between the input stepping of the motor, M, and the flow output of the controlled valve. Since a linearized form is desired, it is only necessary to write

$$\frac{W_{VALVE}(s)}{M(s)} = K_3 \qquad \text{(A. 3-8)}$$

Also for the encoder

$$\frac{\Theta_o(s)}{Head(s)} = K_4 \qquad \text{(A. 3-9)}$$

Now it is only necessary to relate the various inputs and outputs. Note that since in every case the liquid out of a valve free-falls to the next container, there is no "loading" by one stage upon a previous stage. Consequently the overall transfer function is simply the products of the individual transfer functions and can be written

$$\frac{\Theta_o(s)}{M(s)} = \frac{K_p}{(1 + T_u s)(1 + T_L s)} \frac{\text{output quanta}}{\text{step}} \qquad \text{(A. 3-10)}$$

where T_u and T_L have been given previously, and

$$K_p = K_L K_3 K_4 \qquad \text{(A. 3-11)}$$

One further simplifying condition has been utilized. The valve between the two barrels and the valve out of the lower barrel are adjusted so that when the stepping motor is set at the midpoint in its range of travel, the two barrels are both half filled. The K_1's of the two barrels are then considered to be approximately equal. Furthermore K_2 depends only on the cross sectional area of a barrel, which is the same

for the two barrels. Consequently a single time constant, T_p, can be defined and the plant equation written as

$$\frac{\Theta_o(s)}{M(s)} = \frac{K_p}{(1 + T_p s)^2} \quad \frac{\text{output quanta}}{\text{step}} \qquad \text{(A. 3-12)}$$

where $T_p = T_u = T_L$ (A. 3-13)

Fitting the Parameters

The second step in the plant identification process, that of determining the parameters of equation A. 3-12 can be handled by finding the individual parameters of each component of the plant, or alternatively by using experimental input-output data and fitting K_p and T_p to this data. This latter approach is to be preferred since it avoids the accumulation of errors which can occur with the former approach.

The step response of the transfer function given by equation A. 3-12 is

$$\Theta_o\left(\frac{t}{T_p}\right) = K_p\left[1 - \left(1 + \frac{t}{T_p}\right)e^{-\frac{t}{T_p}}\right] \qquad \text{(A.3-14)}$$

If this is evaluated at several points the results are

$$\Theta_o(1) = 0.265\ K_p \qquad \text{(A. 3-15a)}$$

$$\Theta_o(2) = 0.593\ K_p \qquad \text{(A. 3-15b)}$$

$$\Theta_o(3) = 0.802\ K_p \qquad \text{(A. 3-15c)}$$

$$\Theta_o(\infty) = 1.000\ K_p \qquad \text{(A. 3-15d)}$$

The last equation provides a simple test for evaluating K_p, and given a curve which obeys equation A. 3-14 exactly, any of the first three equations can be used to evaluate T_p. In the case of real data, a visual fit to the step response using all four points is to be preferred. To obtain satisfactory results requires that the step response be large relative to disturbances to the plant (e. g., water pressure variations in the supply line) and yet small enough for the assumption of linearity to be sufficiently valid. (In the event that these two conflicting requirements should prohibit the use of this approach, the fit can be made to the more exact impluse response found by the much more involved procedure of cross-correlating the input and the output under closed loop operating conditions).

Figure A. 3.1a shows a typical disturbance response of the plant which results from variations in the water supply pressure. This characteristic is given in order to explain the oscillatory tail of the open loop response shown in Figure A. 3.1b. This open loop response was made with a 512 step input, or one-eighth of the total valve travel. The output change of 368 quanta represents about a nine inch change in

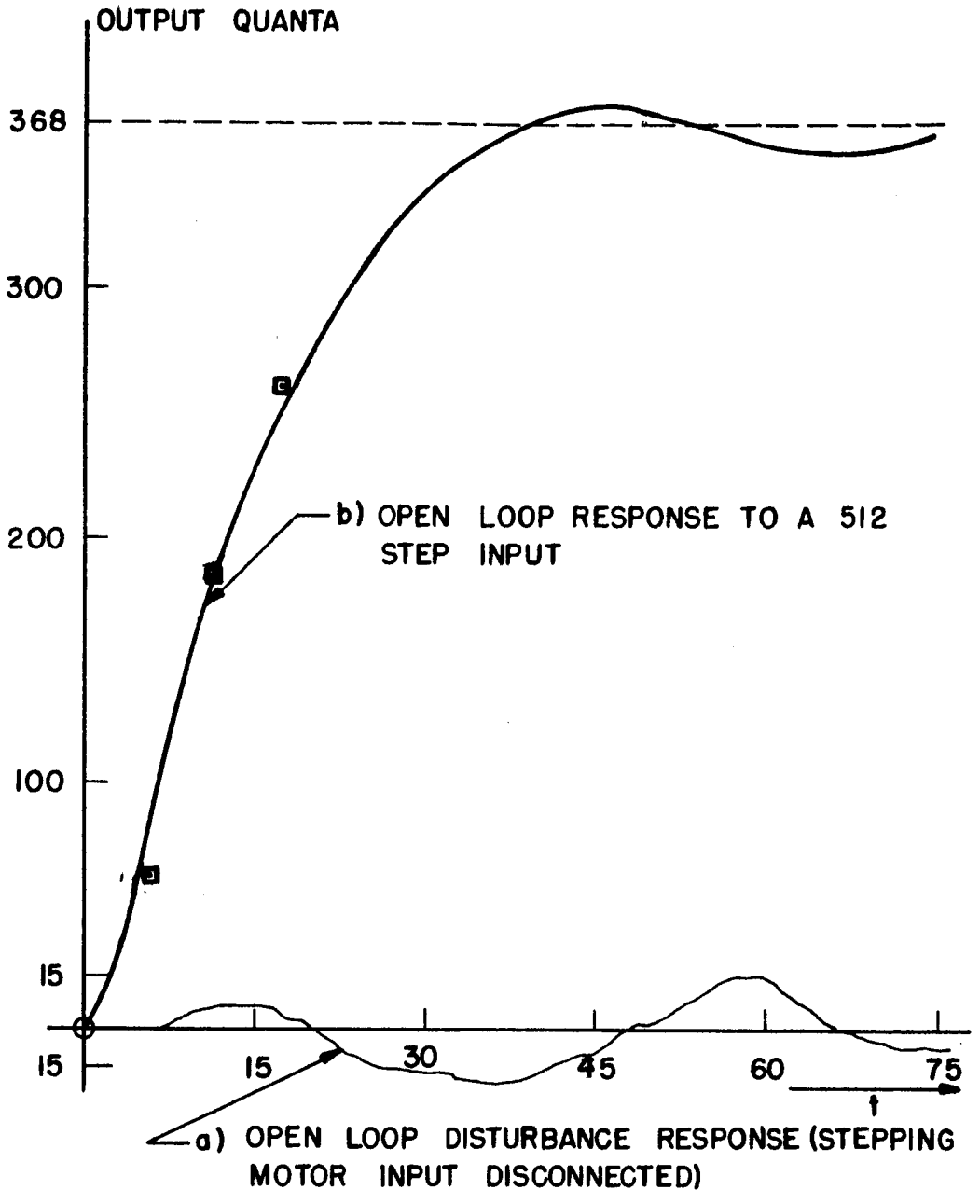


FIGURE A3-1 PLANT RESPONSE CHARACTERISTICS

the output level. Both the input and the output changes were centered about the midpoints of their range of travel.

The points of equation A. 3-15 are shown encircled on Figure A. 3. 1b. Note that these do not fit exactly, presumably because of the nonlinearity of the plant. The values which result from this fit are:

$$K_p = 0,72 \qquad \text{(A, 3-16)}$$

$$T_p = 428 \text{ seconds} \qquad \text{(A, 3-17)}$$

APPENDIX IV

PARTS LIST FOR THE DIGITAL CONTROLLER

The prices listed below for electronic parts are, for the most part, taken from Allied Radio's 1964 Industrial Electronics Catalog. The quantities upon which the prices are based are those quantities listed below for one controller.

Quantity	Description	Price
	Chassis aluminum - .090" x 1$\frac{1}{2}$ sq. ft.	2.00
	Welding - chassis	6.00
	- front panel	3.00
	Polishing and anodizing of chassis and front panel	6.00
10	Miniature DPDT switches - Alco # MST205N	22.00
3	Rotary switches, 1 pole, 12 position, ceramic - Centralab type PA2001	7.11
6	Indicating lights plus mounting sockets - Dialco # 39-28-1433	8.22
3	Knobs	.60
1	Potentiometer, 1 megohm, linear taper - Centralab type HML	2.55
	One sided 2 ounce copper clad fiber-glass epoxy laminate, Type G-10, 1/16" x 1-3/4 sq. ft.	6.00

Quantity	Description	Price
8	Printed circuit board connectors, 18 terminal - Amphenol # 143-018-01	10.48
1	Cable connector, 8 terminal - Amphenol # 26-4100-8P	1.02
124	2N404 transistors @ .28	34.72
2	2N2646 unijunction transistors @ 1.35	2.70
186	1N462 diodes @ .30	55.80
519	Resistors, 1/4 watt, 10% @ .038	19.72
125	Disc capacitors, .001 Mfd. - Centralab Series DD (1000 V) @ .10	12.50
33	Disc capacitors, .005 Mfd. - Centralab Series CK (50 V) @ .14	4.62
1	E-Z Trimpot, 1 megohm - Bourns type 3068-P	1.65
1	Tantalum capacitor, 150 Mfd., 15 V - Sprague # MIL CS13-AD151K	4.50
1	Mylar-paper capacitor, 0.1 Mfd., 200 V - Sprague # 2TM-P10	.21
	TOTAL	$211.40

LIST OF REFERENCES

1. Johnstone, R. E. and Thring, M. W., Pilot Plants, Models, and Scale Up Methods in Chemical Engineering. New York: McGraw-Hill Book Company, Inc., 1957, Chapter 16.

2. "Users See No Big Changes for DDC," ISA Journal, Vol. 11, No. 6, June 1964, pp. 20, 22.

3. Phister, M., Logical Design of Digital Computers. New York: John Wiley and Sons, Inc., 1958, Chapter 6.

4. Hartmanis, J., "On the State Assignment Problem for Sequential Machines I," IRE Transactions on Electronic Computers, Vol. EC-10, June 1961, pp. 157-165.

5. Stearns, R. E. and Hartmanis, J., "On the State Assignment Problem for Sequential Machines II," IRE Transactions on Electronic Computers, Vol. EC-10, December 1961, pp. 593-603.

6. van der Grinten, P. M. E. M., "Finding Optimum Controller Settings," Control Engineering, Vol. 10, No. 12, December 1963, pp. 51-56.

7. General Electric Transistor Manual, General Electric, 1962, Chapter 13.