AN ABSTRACT OF THE THESIS OF

Donald Homer Cooley (Name) for the Master of Science (Degree)

in Electrical & Electronics Engineering (Major) presented on May 15, 1969 (Date)

Title: A MULTIPLE FORMAT TELEMETRY GENERATOR USING BINARY COUNTERS OR BINARY SHIFT REGISTERS

Abstract approved: Redacted for Privacy
Professor Robert A. Short

In recent years the types of engineering data accumulated by many telemetry systems have become so diversified and complex, and the environment in which these systems operate has become so restricting that telemetry systems with increased capabilities and decreased size are becoming a necessity. Advances in integrated circuitry techniques satisfy the size requirements and a great deal of work has been done to increase the capabilities of telemetry systems.

This paper describes two different designs, either of which can further increase the capabilities of a telemetry system. Both designs represent multiple format telemetry generators; they differ most significantly from one another by the fact that one design uses a binary counter as its basic element, and the other uses a binary shift register. Each design makes it possible for a telemetry system, during the mission, to alter the rate at which various engineering

measurements are taken, add new measurements or delete measurements.

A Multiple Format Telemetry Generator Using Binary Counters or Binary Shift Registers

by

Donald Homer Cooley

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

June 1970

APPROVED:

Redacted for Privacy

Associate Professor of Electrical and Electronics Engineering
in charge of major

Redacted for Privacy

Head of Department of Electrical and Electronics Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented May 15, 1969

Typed by Donna L. Olson for Donald Homer Cooley

ACKNOWLEDGEMENTS

The writer wishes to thank Professor Robert A. Short for his valuable help and comments in the writing of this thesis. The writer also wishes to thank the Jet Propulsion Laboratory, Pasadena, California, for supporting his work on this subject.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

A MULTIPLE FORMAT TELEMETRY GENERATOR USING BINARY COUNTERS OR BINARY SHIFT REGISTERS

I. INTRODUCTION

Telemetry systems are designed to monitor a group of channel outputs and transmit information concerning the magnitudes of these outputs. This paper is concerned with telemetry systems which employ time-division type multiplexing; meaning that each channel is sampled in sequence at discrete intervals in time. In order to transmit all the information contained in a channel output, it is necessary that the output be sampled at a rate which is at least twice the highest rate of variation of the output. Since the frequency content of various channels will differ, the most efficient rate, i.e. the minimum rate, at which to sample channels will differ for different channels.

Sequential sampling of the channel outputs is the function of the commutator. Between each sensor and its signal conditioners, e.g. modulator, A/D convertor, etc., is an electronic or mechanical switch. The opening and closing of the switch is controlled by the commutator. During a sampling period, one channel switch is closed while all others are open.

The format or sequence in which the channels are sampled is controlled by a device within the commutator termed the format

telemetry generator (FTG) or telemetry generator. For the purposes of this discussion, the format telemetry generator may be considered as a device which generates strings of n-bit binary numbers which are used to control up to 2^n channel switches. Each binary number represents a predetermined sensor, channel, and switch. If, during a given sampling period, the output of the FTG is some binary number $X_1, X_2, \ldots X_n$, then switch number $X_1, X_2, \ldots X_n$ is closed or in the "on" state during that period, allowing channel number $X_1, X_2, \ldots X_n$ to be interrogated. Figure 1 shows the interconnections between system components for a simplified telemetry system.

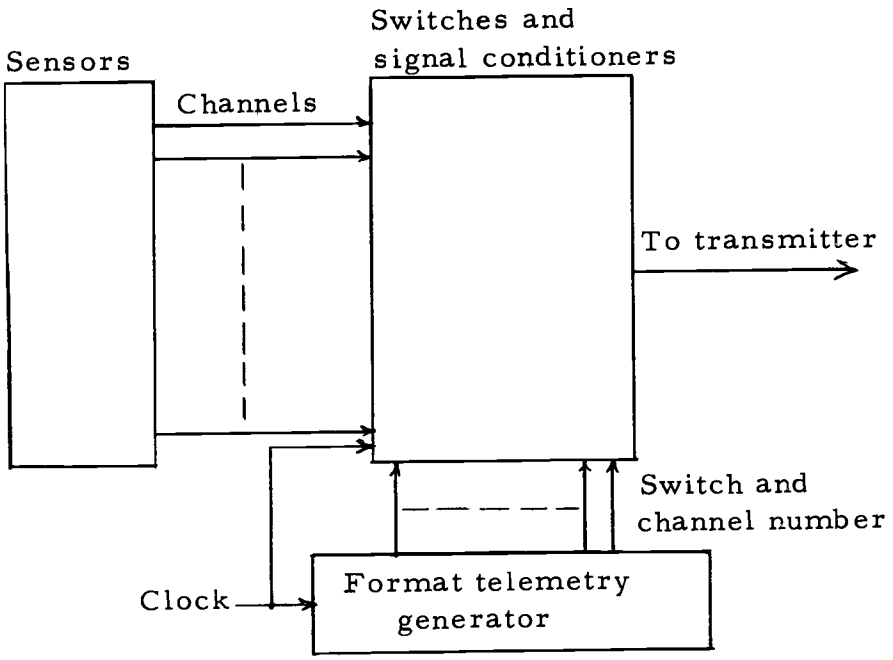


Figure 1. A basic telemetry system.

If the FTG is required to interrogate a group of channels, then the commutation sequence is the shortest sequence of channel numbers produced by the FTG such that each of the channels is sampled at least once. Since different channels may be sampled at different rates, some of the channels will be interrogated more than once during the commutation sequence. Each such sequence consists of an integer number of what are termed main frame or level one sequences. The length of the main frame sequence is N samples where N is the number of samples between like samples of the most frequently interrogated channel.

As an example, assume that the sequence of numbers in Figure 2 represents the decimal equivalent output of an FTG over some period of time for channel numbers 1, 2, ..., 10, 11.

1, 2, 5, 7, 1, 3, 6, 10, 1, 2, 5, 8, 1, 4, 6, 7,

1, 2, 5, 11, 1, 3, 6, 9, 1, 2, 5, 7, ...

Figure 2. FTG channel number output.

For the channel number sequence of Figure 2, the commutation sequence, as defined previously, is represented by the first twenty-four channel numbers. The main frame length is four samples.

The commutation sequence method of representation is obviously a cumbersome method for representing long sequences; a much better method of representation is to use a commutation diagram as

shown in Figure 3. The commutation diagram partitions the sequence into groups of sub-sequences in which channel numbers in the same sub-sequence represent channels sampled at the same rate.

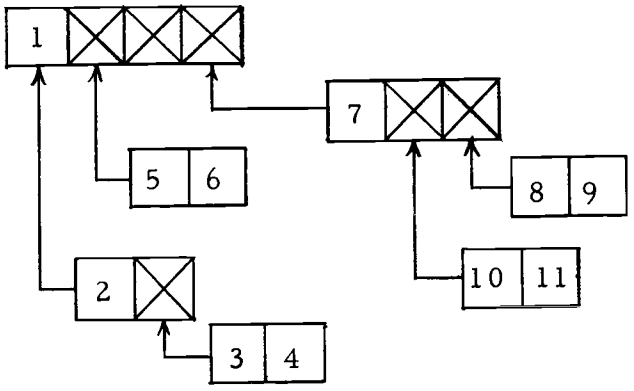


Figure 3. Commutation diagram for commutation sequence of Figure 3.

The X's of Figure 3 denote the fact that the channel numbers of those samples for which they appear are not constant but are functions of a sub-sequence. The arrows denote for which X - sample - periods a sub-sequence is the independent variable. Each sub-sequence is identified by the level and row in which it appears. The level of sub-sequence S is the number of sub-sequences which are a function of S. For example, the sub-sequence 7, X, X, is an independent variable for the main frame sequence and for itself; therefore, it is a level two sub-sequence. In order to distinguish between sub-sequences of the same level, they are further classified according to the row in which they appear. A row consists of all

sub-sequences which are either functions of or independent variables for the other sub-sequences of that row. From this definition, it can be seen that a given sub-sequence can be a member of more than one row; also, from the definition, sub-sequences at the same level cannot be in the same row since neither is a function of the other. The preceeding definitions are summarized for Figure 3 in Table I.

Table I. Row and Level Designation for Commutation Sequence of Figure 2.

Channel Number	Row Number	Level Number
1	1	1
2	2	2
3	2	3
4	2	3
5	3	2
6	3	2
7	4, 5	2
8	5	3
9	5	3
10	4	3
11	4	3

During telemetry missions, the optimum sampling rate for a given channel may vary over a wide range of rates. For instance, consider the atmospheric temperature measuring device on an

exploratory spacecraft. Until the time of encounter with the target planet, there is little need for temperature measurements; meaning that the output of the device need be sampled infrequently. On the other hand, the period of encounter necessitates a high sampling rate in order to obtain as much information as possible concerning the planet's atmospheric temperature.

In early telemetry systems, the need for variable sampling rates for each channel was not as significant a problem as in today's systems. The problem of variable sampling rates, when necessary, was accommodated by sampling each channel at a predetermined maximum rate even though this rate may be much higher than necessary during some periods of the mission (4, 8, 10). In more recent years, the number of channels interrogated by some telemetry systems has become so large that the amount of data to be transmitted must be reduced to a minimum in order that the allowable rate of data transmission is not exceeded. An effective method of reducing the amount of data to be transmitted is to sample all channels at the most efficient rate, thus requiring the commutator to be capable of producing different commutation sequences during different periods in the mission. Systems which require more than a single commutation sequence merely added another format telemetry generator for each sequence required (2, 7). More recently, a multiple format telemetry generator has been designed which will generate several

commutation sequences (6). However, this device still lacks a great deal of flexibility in that it is capable of generating only a limited number of predetermined commutation sequences, and once designed and built, it is limited to use in systems which require these same sequences.

The purpose of this paper is to discuss the design of a multiple format telemetry generator with greatly increased system and sequence flexibility over those of previous designs. There are two basic designs discussed in the text. One design uses binary shift registers for sequence control. This type of design could be adapted to a computer controlled or software type solution to the problem. The other design uses binary counters to control sequence selection.

For each design developed, a detailed logic diagram is given. The logic elements used are all commercially available as integrated circuits. All flip-flops are clocked j-k, r-s flip-flops. The "clocked" designation for the flip-flops means that the j and k inputs are AND-ed with the clock input so that the j input is a logical one only when both the j and clock inputs are ones, and the k input is a logical one only when both the k and clock inputs are ones. The j input is used to set the flip-flop to the one state and the k input resets the flip-flop to the zero state. With respect to the j-k inputs, transitions of a flip-flop from the zero state to the one state and vice versa occur during input transitions from the one to the zero state.

For example, if the output of flip-flop Q is a one and the k and clock inputs are ones, the Q output will not change to a zero until either the clock or k inputs change to a zero. This characteristic, and the fact that when both the j and k inputs are a one the flip-flop merely changes states on each clock pulse, formed the basis for the design of the binary or ripple counters used by the counter type telemetry generator. The three flip-flops connected as shown in Figure 4 represent a three stage or three-bit ripple counter. The term ripple is used to denote the fact that each time the Q_0 flip-flop changes from the one to the zero state it causes the Q_1 flip-flop to also change states. This characteristic is true for all flip-flops except the most significant bit flip-flop, i.e. in this case the Q_2 flip-flop.

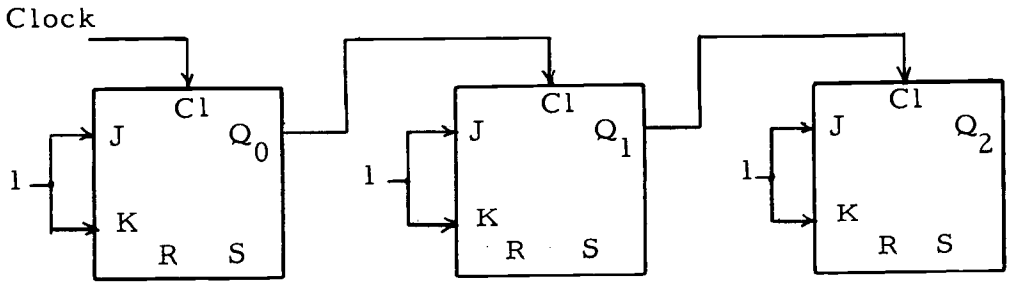


Figure 4. Three-stage ripple counter.

The r-s or reset-set inputs differ from the J-K inputs by the fact that they are not clocked and the r and s inputs cannot be

logical one's simultaneously. Appendix C of this discussion lists the logic equations for all logic elements. For all logic diagrams in this text a five-bit channel number is used.

II. FUNCTIONAL CHARACTERISTICS, CONSTRAINTS, AND METHODOLOGY

In Chapter I the basic function of a telemetry generator was briefly described. The first portion of this chapter will be devoted to a more detailed description of the functional characteristics of a multiple format telemetry generator. The constraints under which the designs were developed will then be discussed. The capability of a system to perform the required functions and the degree to which the constraints are met will be used to judge the relative merits of that system. Finally, a discussion of the method of solution or approach to the problem will be given in order to clarify the reasons for the types of designs developed.

Functional Characteristics

The necessary functional characteristics of a multiple format telemetry generator were considered as the following:

1. The generator must be externally controllable; that is, selection of a given commutation sequence is performed by communicating from the operator to the generator a predetermined set of commands. The commands are to be binary logic signals and the only function of the telemetry generator with respect to these signals is to decode them into the signals required to generate the desired

commutation sequence.

2. The generator must be capable of producing several different commutation sequences.
3. The system must operate under the control of a clock.
4. The output of the telemetry generator during a given clock period is to be one of a set of 2^n binary words, where there are 2^n information channels.
5. The design developed should represent a hardware as opposed to a software type solution.

Functional characteristic number one serves to define the nature of the control system and the function of the generator with respect to the control system. By restricting the nature of the control signals to binary logic signals, the control system within the telemetry generator has been greatly simplified without altering the basic nature of the problem; that is, multiple formatted sequence generation.

Functional requirement number two delineates the most basic function of the telemetry generator; the generation of the signals that control the sequence in which sensors are sampled and the ability to generate more than a single sequence. The flexibility with which these sequences can be selected represents the single most important characteristic.

The third requirement restricts the class of possible solutions

to those which operate synchronously as opposed to those which operate asynchronously. This is a necessity in order that synchronous decommutation of the data at the receiver be possible.

Functional requirement four also simplifies the system requirements without altering the basic nature of the problem by eliminating from consideration those devices external to the telemetry generator which are dependent upon the make-up of the telemetry generator. For instance, it is probable that the voltage levels of the output logic signals would not be sufficient to activate switches, thus requiring some type of signal conditioners.

The final functional characteristic places the burden of the responsibilities for sequence generation upon the generator itself and not upon the operator. It eliminates from consideration a design such as one which uses a programmable digital computer for sequence control (1, 9).

Constraints

The designs arrived at in this discussion represent a balance of the following areas of concern; system cost, necessary control logic and input control variables, sequence flexibility, and system flexibility. In order to simplify the task of comparing the cost of different designs, all logic elements, regardless of type, were assumed to have a uniform cost of one unit. Using this criterion, the

cost of a system is equal in magnitude to the number of elements used.

The control logic is that portion of the telemetry generator which regulates the selection of the commutation sequence as opposed to the portions which produce the channel numbers. The input control variables are those signals which must be sent to the generator by the operator in order to produce the desired sequence.

The most important consideration in the design of a multiple format telemetry generator is sequence or format flexibility. A great deal of added cost must be incurred in order to give the desired degree of format flexibility.

The final constraint on the system is that it possess some degree of system flexibility. By this it is meant that the telemetry generator should be compatible with commutation systems in which binary logic control signals are used or can be used. System flexibility also requires that a design be such that only minor changes are necessary if shorter length channel numbers are used than those for which the system was originally designed, e.g. a telemetry generator using n-bit channel numbers should be compatible with a telemetry generator which uses m-bit channel numbers, where $m \leq n$.

Methodology

Several approaches were investigated in arriving at the designs

discussed in this paper. Because of the number of sequences and length of many of the sequences, most approaches proved non-feasible. The designs finally arrived at represent a compromise between cost and sequence flexibility, i.e. in order to maintain the system cost within reasonable limits some sacrifice in sequence flexibility was necessary.

Although the number of samples per sequence can be very large, the number of samples in a single main frame sequence is generally relatively small. For example, the engineering telemetry generator of the Mariner '69 spacecraft uses a sequence length of over 4000 samples for 94 channels; however, the main frame length is only 20 samples (7). Exploitation of this characteristic formed the basic approach to the solution of the problem. Instead of picturing a particular telemetry format or sequence in its entirety, each was viewed as partitioned into sub-sequences. A commutation level was then made up of individual sub-sequences.

By partitioning the commutation sequences into smaller sub-sequences, two important advantages were realized. As already mentioned, much shorter sequence lengths are encountered with this approach, and more important, channel numbers are not repeated in any sub-sequence. It might be thought that because of these two advantages it would be possible to design a device capable of generating any of these shorter sub-sequences. Since no channel numbers are

repeated during a sub-sequence, there could be a one-to-one correspondance between the state of the machine and the required channel number; therefore, it would only be necessary to develop the next state equations to determine the logic requirements. A 2^N state machine would require N flip-flops; this would also signify N-bit channel numbers. Since all sequences must be generatable, the input logic for each flip-flop would have to be capable of generating any function of N variables. Decode logic would also be necessary to select the proper set of input equations for each sub-sequence generator.

As an example, assume that a four-bit channel number is used. Each of the four flip-flops would require 16 AND-gates and a single OR-gate to be capable of generating any function of the four variables. The four sets of input-function AND-gates would each require a decode logic section to select the proper input equation for each flip-flop. The least costly decode section would be a 16-bit register in which each bit of the register controlled an AND-gate. Therefore, such an approach as the preceeding would possess a cost of 132 units for each sub-sequence generator, or more generally, each sub-sequence generator would require $N(2^{N+1} + 1)$ cost units. This approach is obviously too costly for reasonable channel number bit lengths.

The two advantages of the sub-sequence approach make feasible

the use of either binary shift registers or counters to generate the sub-sequences. It should be noted that this type of approach does place an added restraint upon the system; that is, additional decode logic must be developed in order to distinguish between the various sub-sequence outputs.

This then, was the method of solution used: view the telemetry generator as being made up of a group of individual sub-sequence generators, the output of only one being used at any given time.

III. COUNTER TYPE TELEMETRY GENERATOR

The counter type telemetry generator is actually a group of individual binary counters, each capable of producing a sequence of switch numbers. There are groups of counters for each level of commutation. Each counter produces a predetermined portion of the commutation sequence. During any sampling period, a decoding section selects one individual counter output as the overall system output.

The degree of format flexibility, i.e. the number of different commutation sequences which can be generated, is dependent upon the number of sub-sequence generators available; the more available, the greater the format flexibility. The commutation sequence of Figure 3, for example, would require seven individual counters.

A block diagram showing the interconnections between functional sections is shown in Figure 5.

Operational Characteristics

Within the overall counter type telemetry generator (CTTG) there are four operational sections; sequence generation, clock control, sequence control, and output decoding. Each of these functional areas is logically distinct. That is, the logic components within one operational section are not shared by those of another section. For

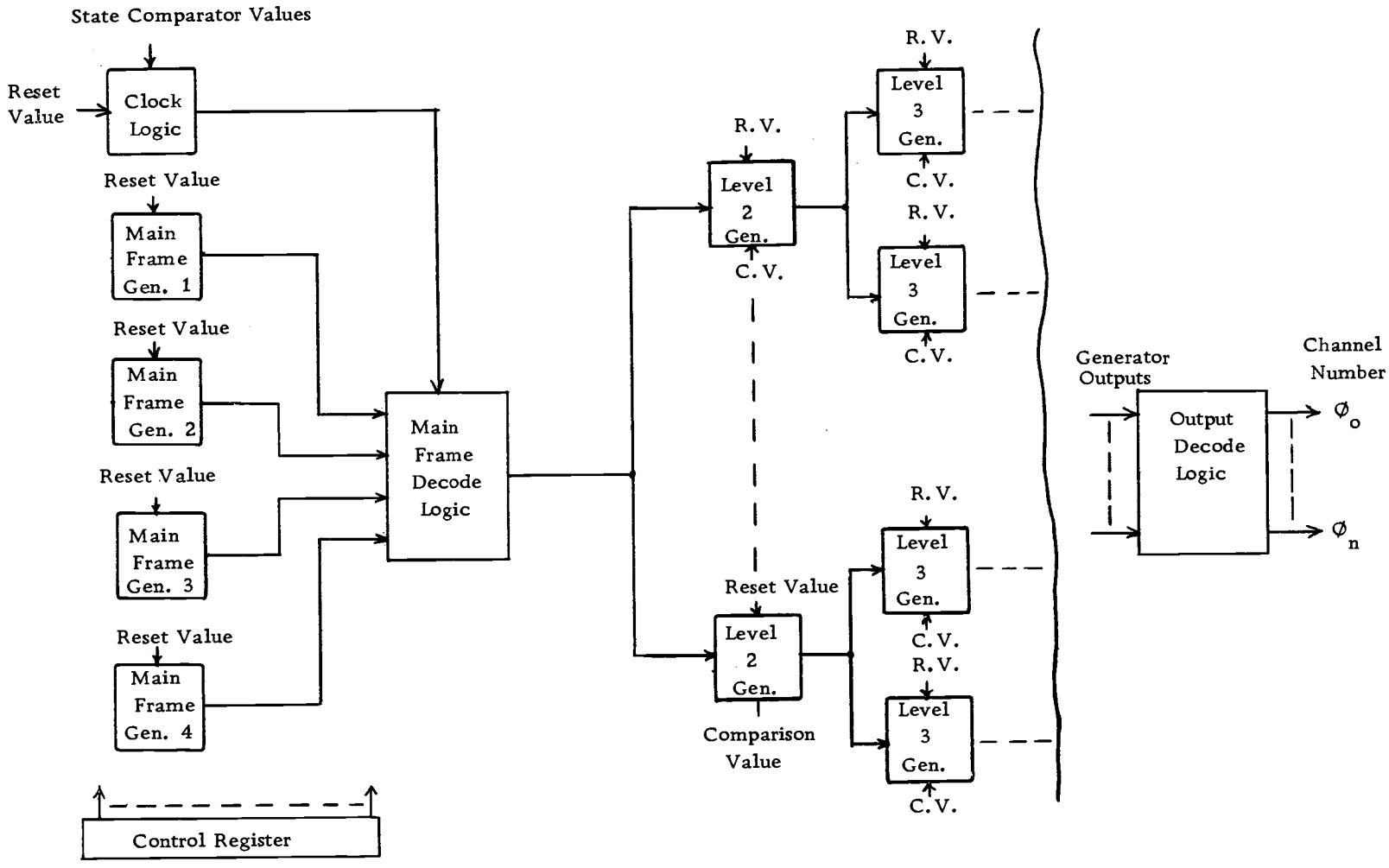


Figure 5. Block diagram for CTTG.

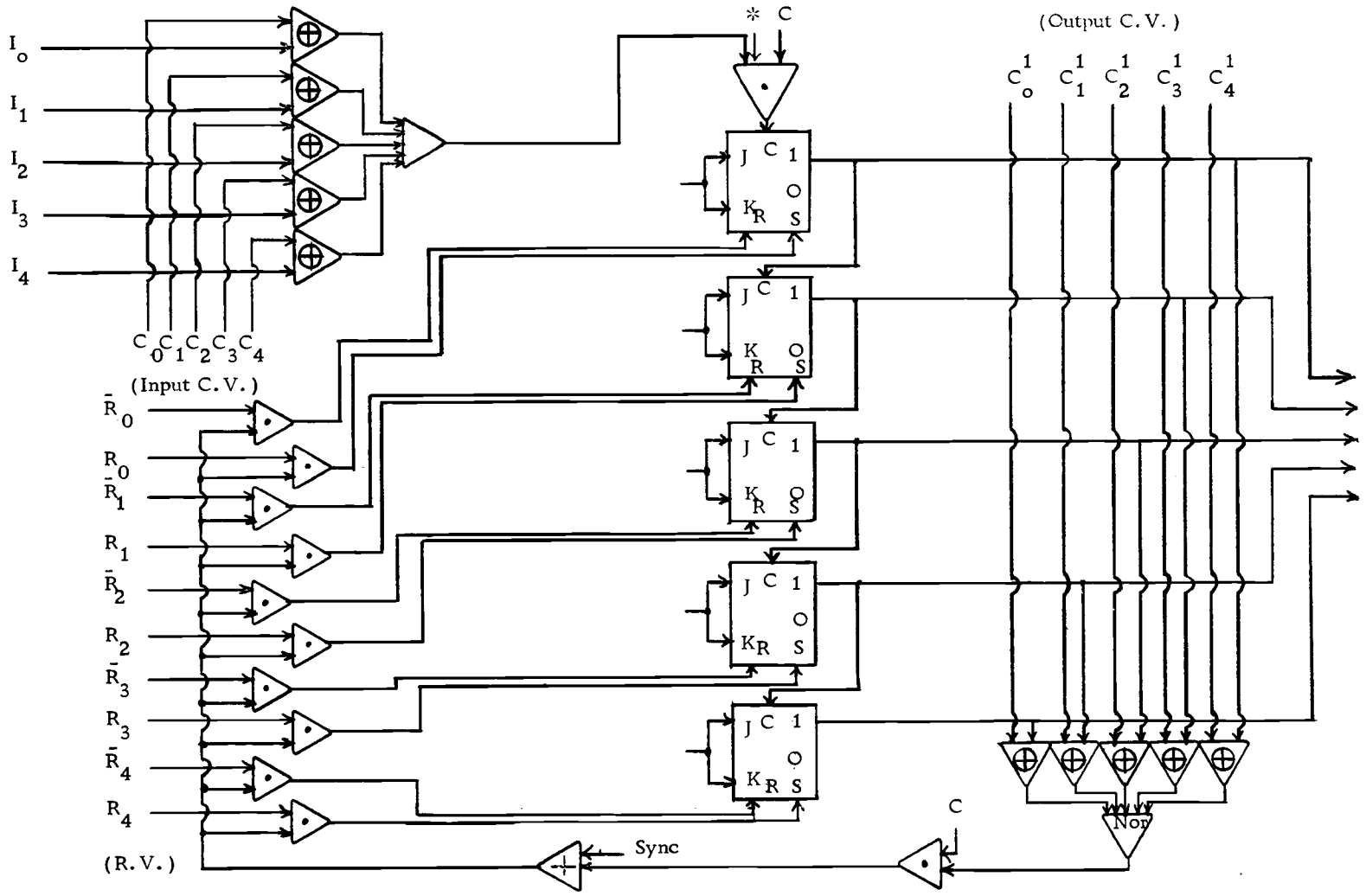
this reason, the operation characteristics of the overall system will be discussed from the standpoint of these operational sections.

Sequence Generation

The sequence generators are of two similar types. The level one or main frame sequence generators represent one type and the sub-sequence generators the other.

Figure 6 represents the logic diagram for a typical sub-sequence generator. The counter is basically a binary ripple counter with control logic to determine when it counts and the maximum and minimum values of the count sequence. The flip-flops used are clocked j-k flip-flops with auxilliary dc level r-s inputs, i.e., the r-s inputs are not clocked.

The minimum value of a count sequence is the reset value of Figure 6, and the maximum value is the output comparison value. Control of the counter is executed by the two comparators of Figure 6 and the clock control gate. The input variable C is a universal clock pulse common to all sub-sequence generators. The input comparator's output value acts as the control pulse for C. When the input value and the input comparison value are equal, the output of the comparator is a logical one. The counter will continue to increase its output value whenever its input comparator is in the logical one state, and the other comparison values are also ones, until its output value equals the output comparison



*All same row higher level sub-sequence generator input comparator values and inverse of all same row level n+1 input comparator values.

Figure 6. CTTG sub-sequence generator level n logic diagram.

value. When this occurs, the reset control gate is then enabled so that on the next pulse C the counter will reset to its minimum value. The input value for a sub-sequence generator is the output value from the next higher level sequence generator in the same row.

The main frame sequence generators differ from the sub-sequence generators by virtue of the fact that they have a different set of count control variables, and by the fact that they do not possess output comparator logic. Figure 7 represents the logic diagram for a main frame sequence generator.

This type generator operates in a manner similar to that of a sub-sequence generator. Whenever the clock logic state counter is equal to f(a)[1], the ripple counter counts on each clock pulse C. The synchronization input, as with all generators, is used to reset the counter to its initial value. The clock logic reset control also resets the main frame counter to its initial value.

---

[1] The function f(a) represents a Boolean function of the output variables of the clock logic state counter. For example, if four main frame sequence generators were used, a 2 bit (a_o, a_1) four state counter would be needed to discriminate between generators. For generator one:

$$f(a) = \bar{a}_o \cdot \bar{a}_1 = \begin{cases} 1, \text{ if } a_o = a_1 = 0 \\ 0, \text{ if } a_o \text{ and/or } a_1 = 1 \end{cases}$$

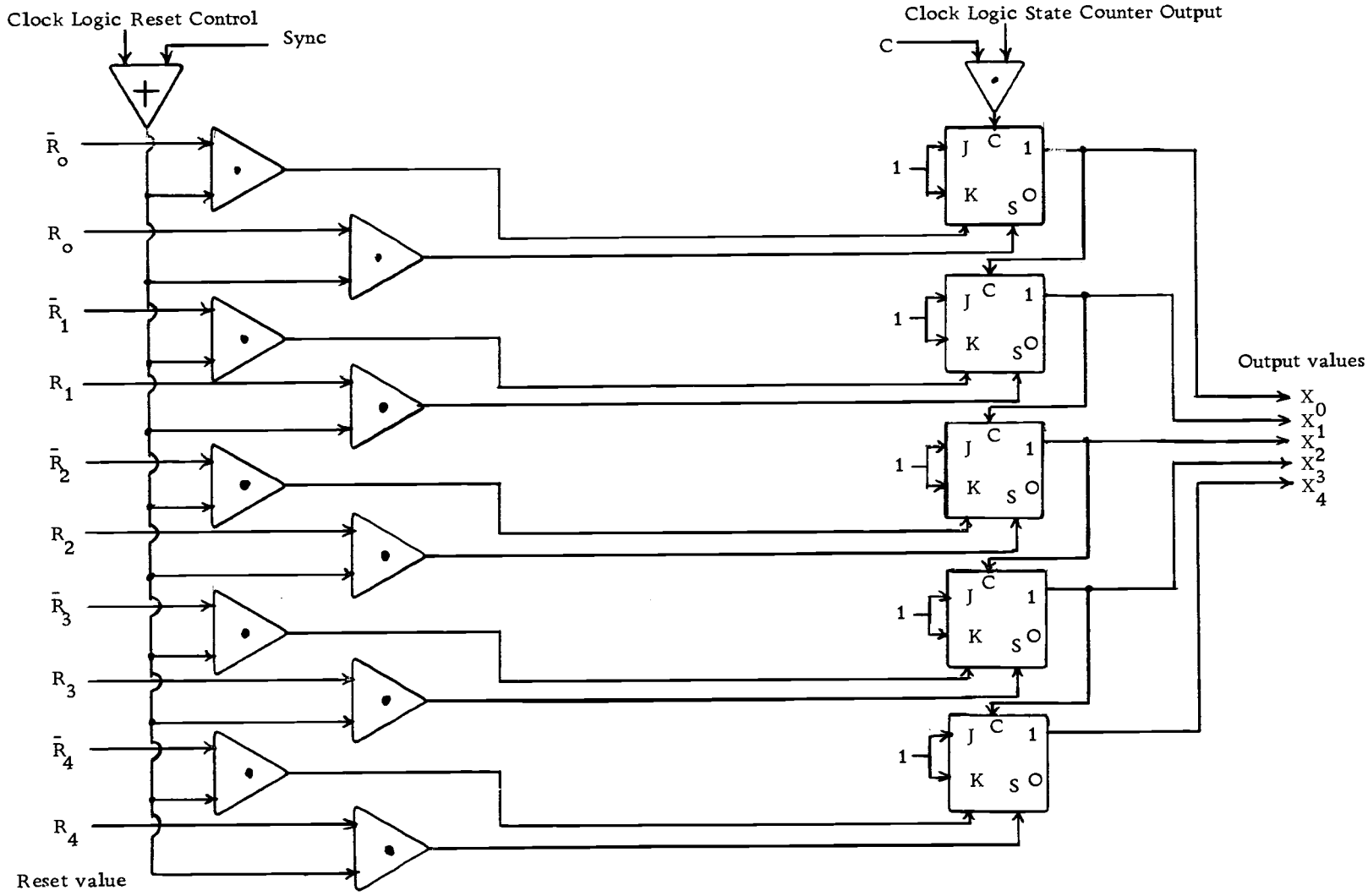


Figure 7. Main frame sequence generator logic diagram.

Clock Logic

The timing of the main frame sequence generators is the function of the clock logic section (Figure 8). The clock counter of the clock logic section is a binary ripple counter which counts on each clock pulse C. The minimum value for the counter is zero and the maximum value is N-1, where N is the number of samples in a main frame sequence. The output comparator functions in the same manner as those of the sequence generators; that is, it is used to perform the command function of resetting the clock counter when its maximum value is reached. The state comparator is actually a group of comparators each of which compares the output of the clock counter with that value of the clock counter at which the state counter should count up one. For four main frame sequence generators,three comparators within the state comparator would be needed. An OR-gate to combine these comparator outputs is also needed. The output of the state counter represents which main frame sequence generator output is to be used during that sample period. Along with the selection of the generator outputs, the state counter output also acts as the control variable for the count command for the main frame generator counters.

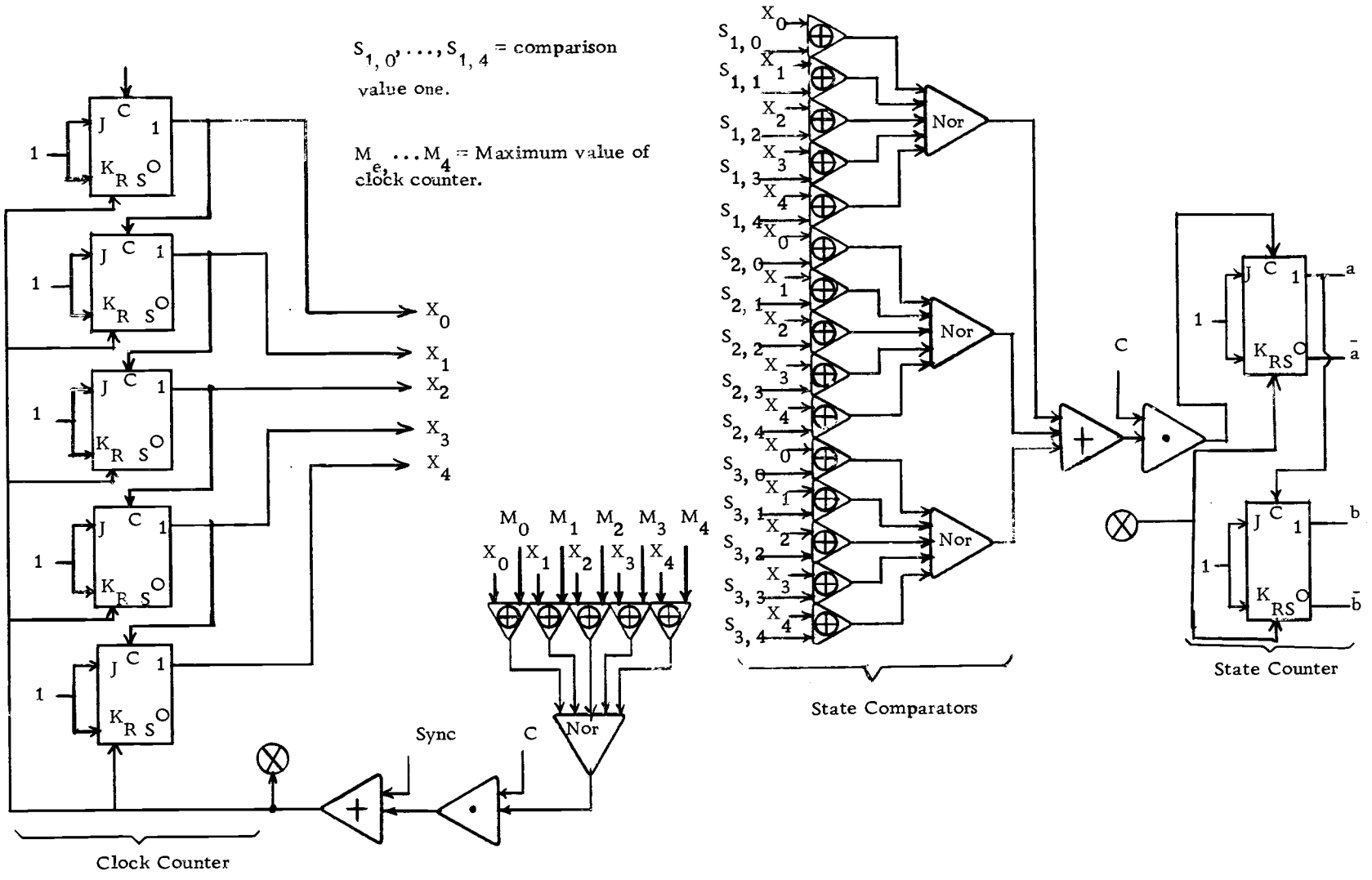


Figure 8. CTTG clock logic section logic diagram.

Decode Logic

There are two distinct decode sections within the CTTG. The basic function of both sections is to derive a single channel number ouput from a group of channel number inputs. This function was implemented in both cases by the use of two level AND-OR networks and clocked j-k flip-flops; there being one such network for each bit of a channel number.

The least complex decode section is shown in Figure 5 as the main frame decode logic. Figure 9 shows the logic diagram for this decode section. Since a five-bit channel number is used, there are five[2] distinct AND-OR networks, and since four main frame sequence generators are used, there are four AND-gates for each bit of a channel number. Each AND-gate has three inputs. Two of these inputs act as control inputs, each of the four sets of these inputs represent one of the four states of the clock logic section. The third input represents one bit of the channel number output from a main frame sequence generator. Each OR-gate combines four AND-gate outputs into a single output. The combination of the binary outputs of these five gates signifies the desired channel number output from the main

---

[2] All block and logic diagrams are referenced to a five-bit channel number. Extensions to longer or shorter word lengths are relatively simple. A five-bit channel number means that up to thirty-two channels can be controlled by the commutator.

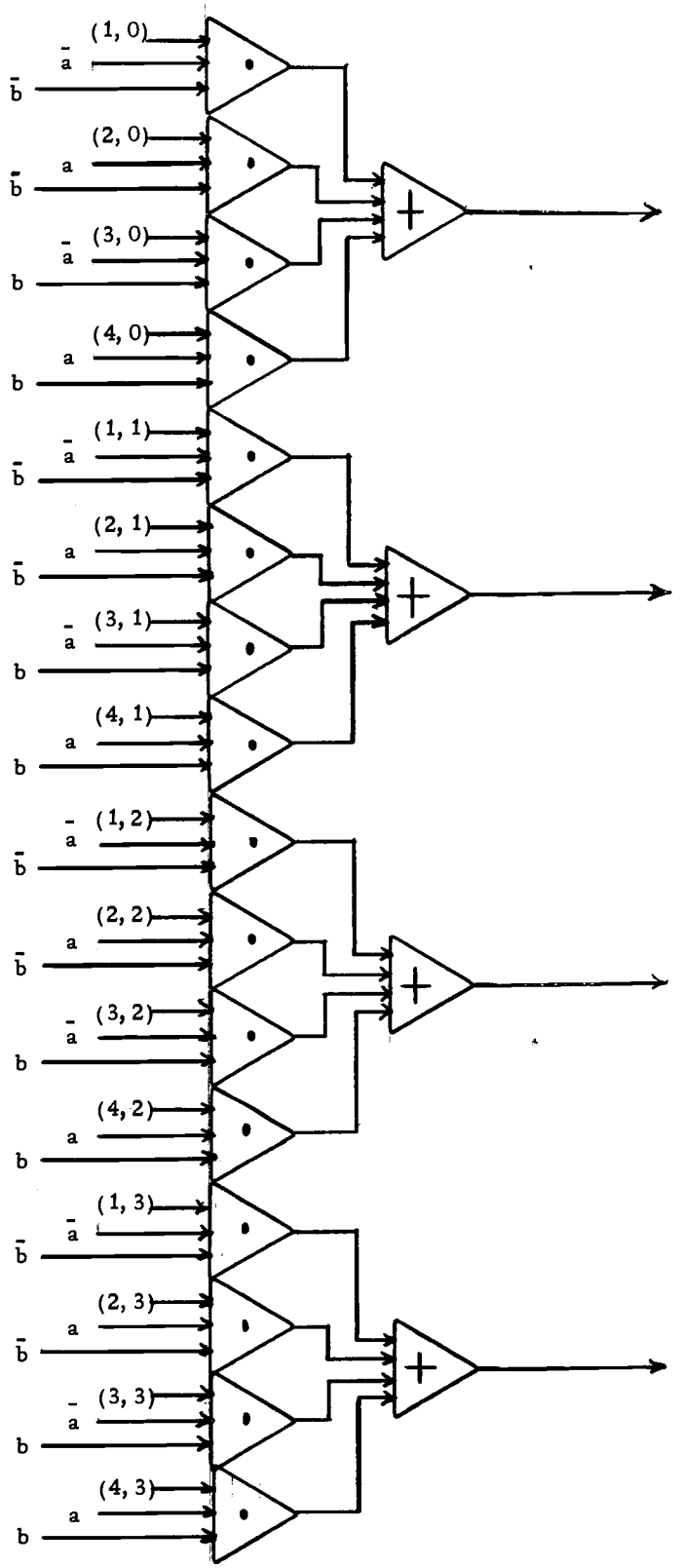


Figure 9. Main frame decode section logic diagram.

frame sequence generators.

The second decode section is shown in Figures 5 and 10 as the output decode logic. The only difference between this section and the main frame decode section is the fact that for this section a much larger group of channel numbers must be decoded in order to determine the proper output. One of the outputs which is decoded by this section is that of the main frame decoder.

The inputs to a given AND-gate for the output decoder are determined in the following manner. Consider the zero bit output of generator six, labeled T (6, 0) in Figure 10. The inputs to that bit's AND-gate are Z (6, 0), its input comparator value (C6), all higher level input comparator values in the same row, and the complement of all next level lower input comparator values in the same row. This means that except for transistion periods, all generator outputs except one are ANDed with a logical zero, the remaining output being ANDed with a logical one. The OR-gates of Figure 10 combine the AND-gate outputs for each bit of a channel number to form the inputs to the clocked j-k flip-flops. During a given clock period, the channel number represented by the outputs of these flip-flops denotes the channel which is to be sampled during that period.

Control Inputs

In order to select the desired commutation sequence, each

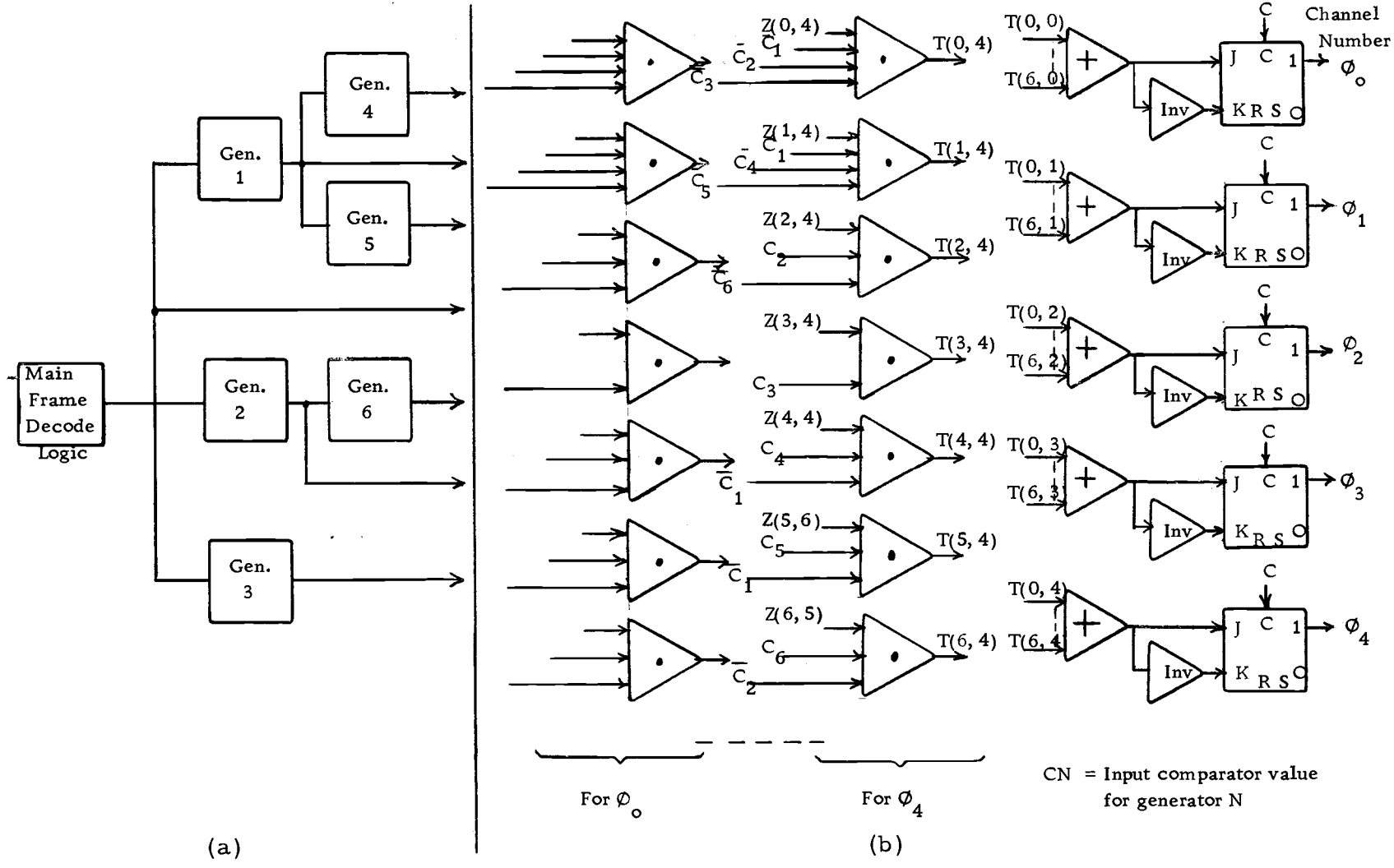


Figure 10. (a) CTTG sequence generator configuration, (b) output decode logic for (a).

individual sequence generator and the clock logic section require certain control inputs. These control variables are the outputs of a register as shown in Figure 5. Once the commutation sequence has been determined it is then necessary to load the control register with the necessary values and finally, synchronize all generators with a sync pulse. Generation of the desired commutation sequence will begin with the initiation of the operation of the master clock. An example of the selection of the necessary control variables and the operation of the CTTG is given in Appendix A.

Format Limitations

The most important limitation in the flexibility with which individual commutation formats may be selected is the result of the fact that binary counters are used as the sequence generators. Because of this fact, each channel number in a sub-sequence must be numerically adjacent to one of the other channel numbers in that same sub-sequence. Two numbers are numerically adjacent if the absolute value of their difference is one. For example, the commutation scheme of Figure 11 should not be realizable with ordinary binary counters since a level two sequence generator could not change from a value of 11 to a value of 19 in one count. The main frame sequence generators are not as severely restricted by this format limitation since each generator is capable of producing a portion of

the main frame sequence. In other words, for four main frame sequence generators, four non-adjacent channel numbers could be tolerated.

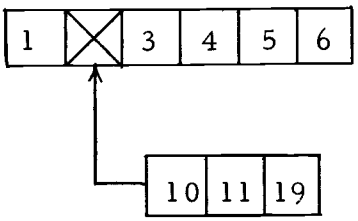


Figure 11. Unrealizable commutation sequence.

The only other format limitation, as was stated previously, is dependent upon the number of individual sequence generators available. The greater the number of generators available, the greater the format flexibility.

IV. SHIFT REGISTER TYPE TELEMETRY GENERATOR

The design of the shift register type telemetry generator (SRTTG) was based upon the same approach as that discussed for the CTTG; however, because of the inherent differences between shift registers and counters, the SRRTG lacks a great deal of the modularity of the CTTG. Figures 12, 13, and 14 show the block and logic diagrams for the SRTTG.

The SRTTG uses variable length shift registers in order to produce different length commutation sequences. One shift register is required for the generation of the main frame sequence, and two additional shift registers for each additional lower level of sequence generation. Unlike the decode sections of the CTTG, multiple additions are performed on certain register contents in order to determine the required channel number.

Operational Characteristics

As with the CTTG, the SRTTG can be divided into operational sections; namely, the sequence generating shift registers and their associated comparator, reset, and length control logic, the clock logic, and the output decode logic. The shift registers used are variable length, i.e. input length control variables determine which stage of the shift registers will be used as the last or feedback stage. The

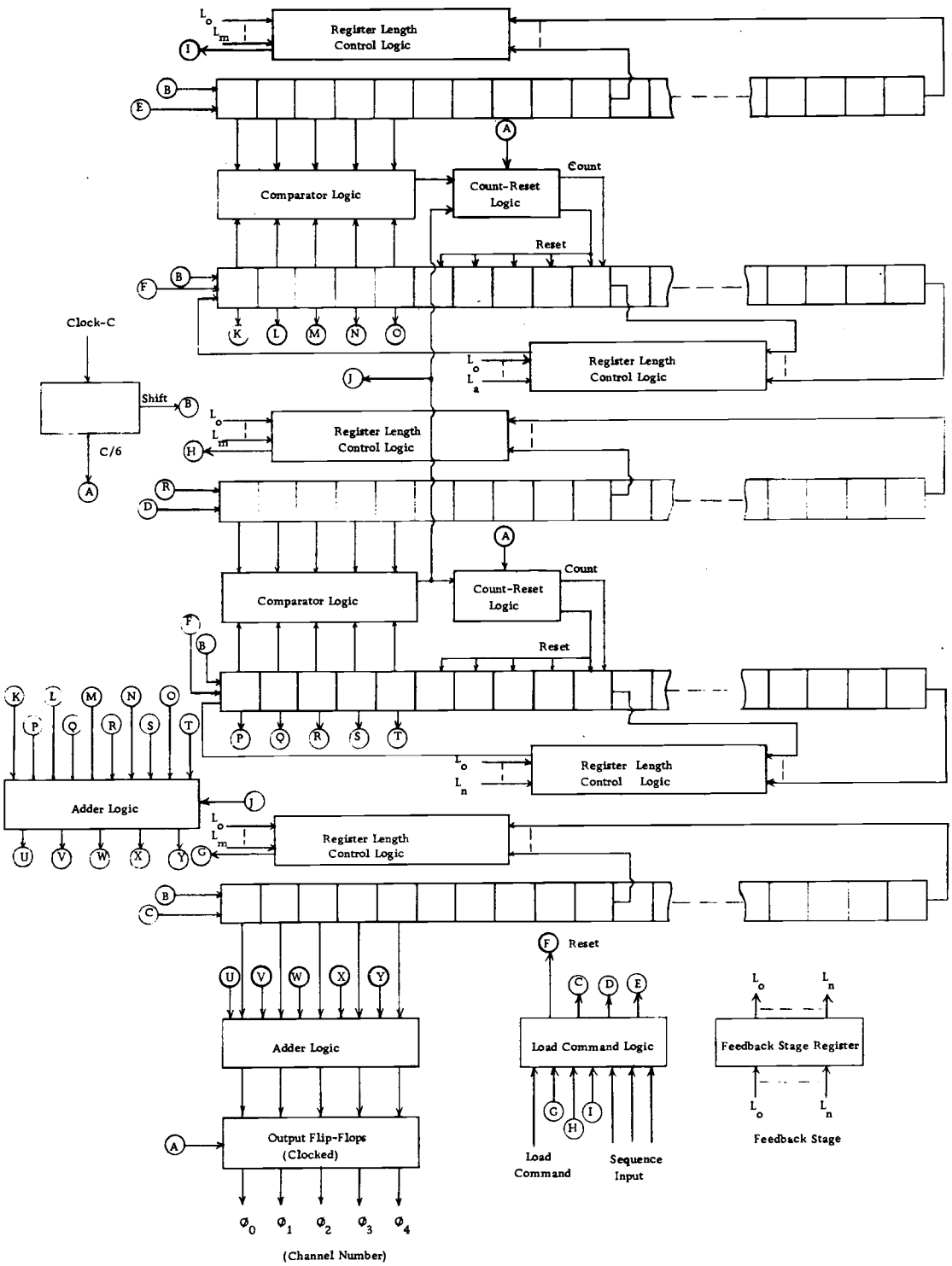


Figure 12. SRTTG block diagram.

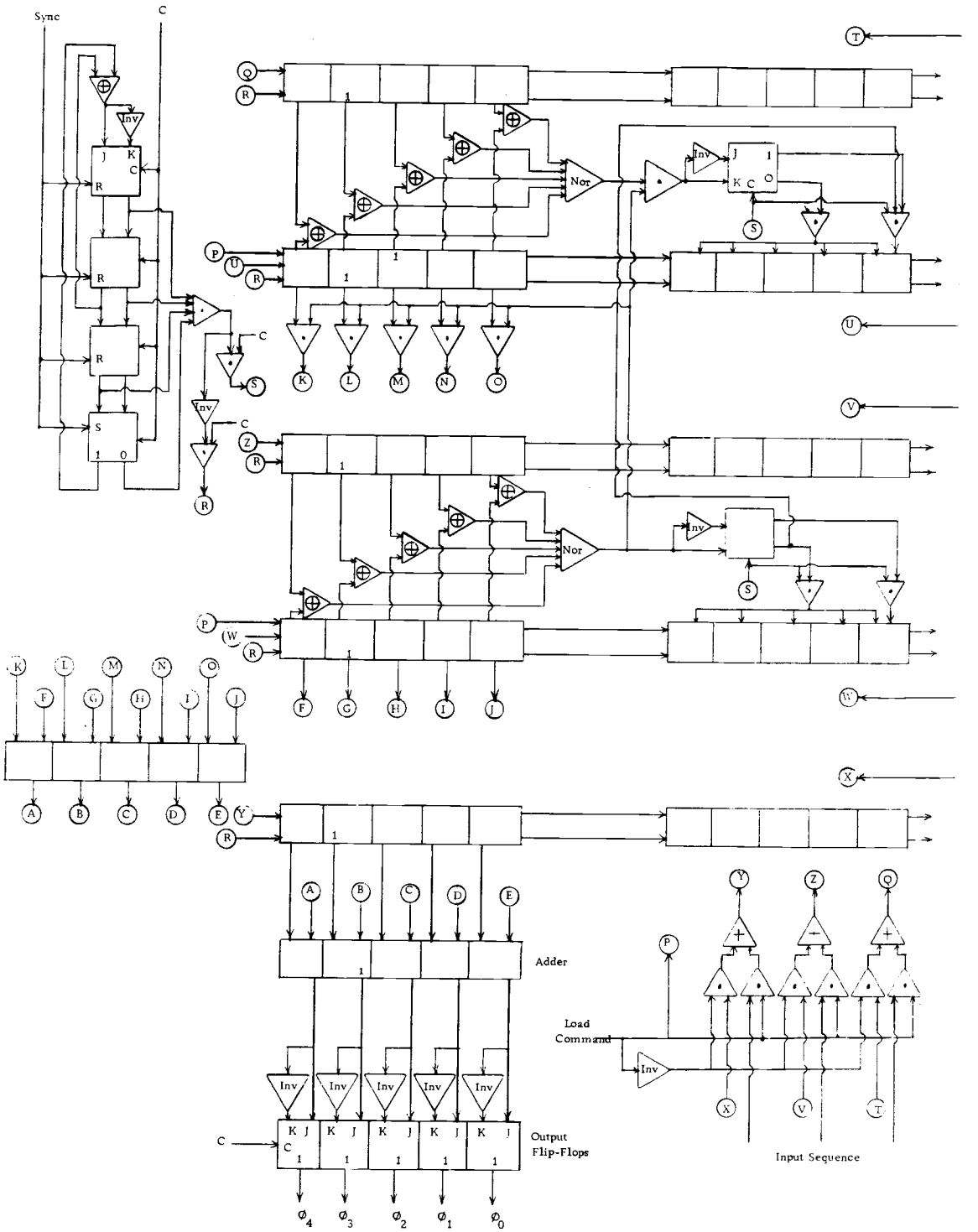


Figure 13. SRTTG logic diagram.

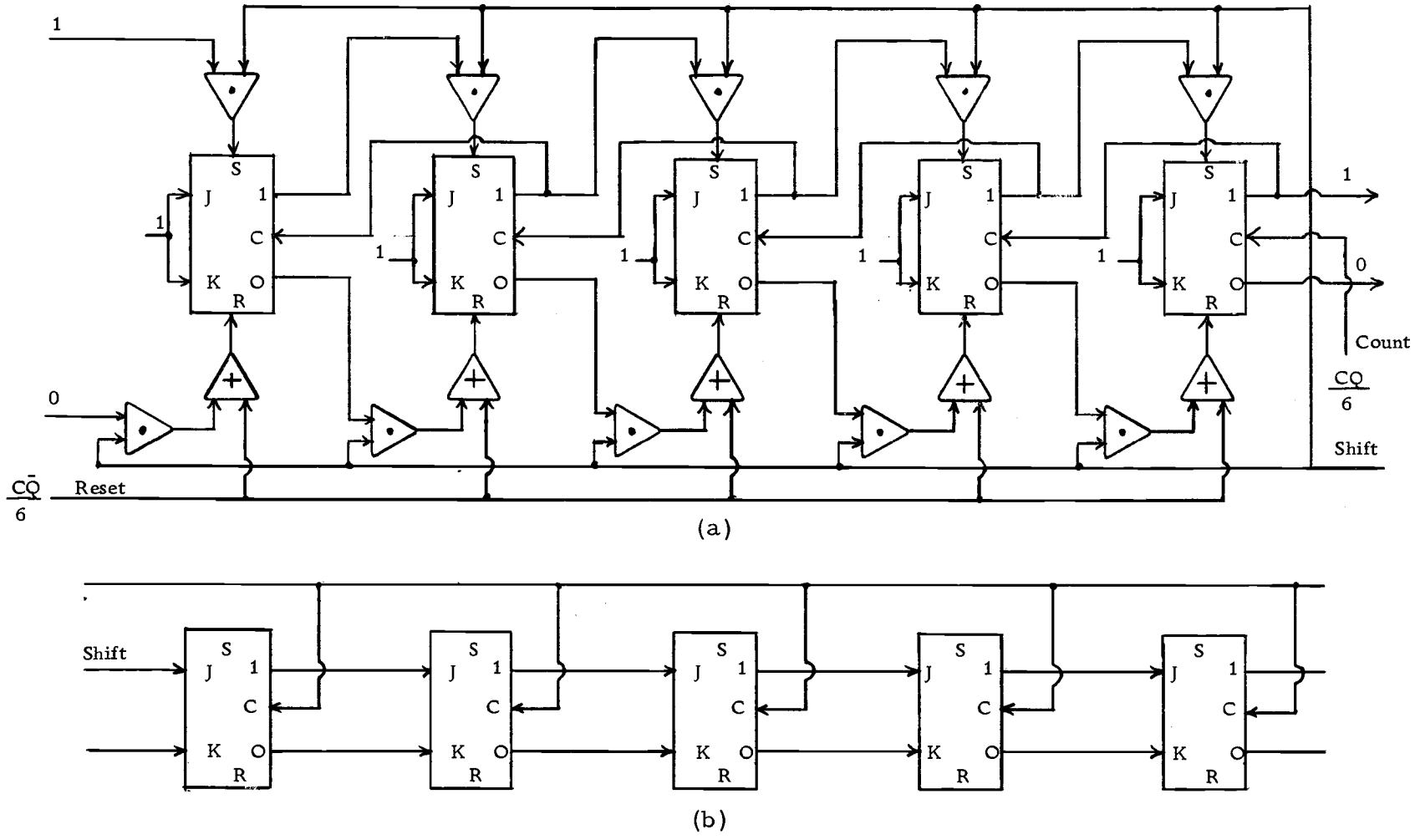


Figure 14. (a) SRTTG shift, reset, and count logic (b) SRTTG shift logic.

clock logic section is merely a divide-by-n+1 counter[3] where n is the number of bits per channel number. The output decode logic section bears no resemblance to that of the CTTG; it is actually a group of parallel binary adders with the output of the final adder being the desired channel number.

Shift Registers

There are three operations performed for a three level SRTTG during each sequence of n+1 clock pulses, where n is the number of bits per channel number. During the first n pulses, the contents of each register are shifted circularly to the right. During pulse n+1 the output flip-flops are set to the result of the addition of the first n bits of registers one and two. The contents of register four are also added to this sum if the first n bits of registers two and three are alike, i.e., the comparator output is a logical one. In this manner, register one is used as the first level of commutation and registers two and three as the second level and registers four and five as the third level of commutation.

Also, during pulse n+1, the contents of flip-flops n+1 through 2n of register two are either reset to all zeroes if Q1 is in the zero

---

[3] A divide-by-X counter has two outputs, namely, Q and $\overline{Q}$. The output is Q when the number of clock pulses divided by X is an integer; if not, the output is $\overline{Q}$.

state or increased by one if Q1 is in the one state. The same stages of register four are reset if Q1 and Q2 are zero, increased by one if Q1 is zero and Q2 is one, or left unchanged when Q1 is one. Finally, during this same pulse, Q1 is reset to zero if the comparator output for registers two and three is one; otherwise, Q1 is set to a one. Flip-flop Q2 is reset to a zero if both comparator outputs are one; otherwise, it is set to a one. In this manner, the contents of registers two and four are manipulated so that two levels of subcommutation can be achieved.

Clock Logic

There are two clocked commands, the shift command and the register manipulation command. In order to generate these pulses, a divide-by-n+1 counter is used as shown in Figure 13. This counter's output is the register manipulation command. The inverse of this output AND-ed with the master clock pulse forms the shift command. Figure 15 is a timing diagram for these pulses.

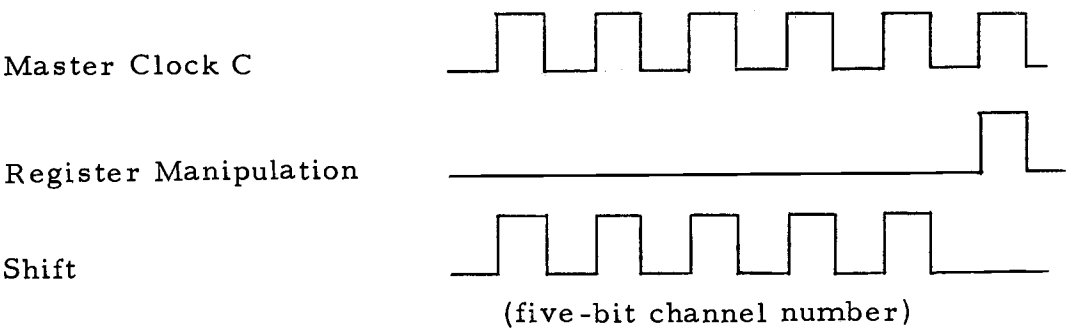


Figure 15. SRTTG timing diagram.

Shift Register Feedback Length Control Logic

The length control logic determines the feedback stage for each register. Since the length of a main frame may change with different commutation sequences, it is necessary that the length of each shift register be variable. The actual form of this logic is shown in Figure 16 for four feedback stages.

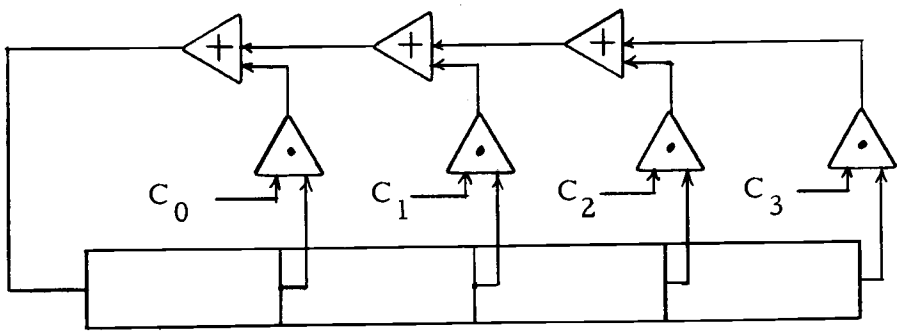


Figure 16. Feedback length control logic.

When the feedback stage is Q_n (Figure 16), length control variable C_n is one and all other C_i's are a zero. Although sub-commutation sequences may be shorter or longer than the main frame sequence, it is only necessary that the main frame length be externally controlled. This is due to the fact that the sub-sequence lengths for levels two and three are actually determined by the input control variables for registers three and four.

Control Logic Inputs

Load, reset, and feedback length control inputs are needed for the SRTTG. The load and reset inputs are identical; they remain a logical one while the commutation sequence is being loaded. Once the registers are loaded, they are changed to a logical zero. In Figure 12, it is shown that there is no load command input for registers two and four. The reason for this fact is that the initial state for both of these registers is the all zeroes state. When the reset command is a logical one, each shift command causes the left-most flip-flop in registers two and four to be reset to a logical zero regardless of the actual input fed back.

The load command input enables the operator to serially load the shift registers. When in the one state, the load command enables the commutation sequence loading gate and "turns off" the feedback input gate, i.e. the output of the feedback input gate is a logical zero regardless of the gate's input.

Although the registers are loaded sequentially, the generation of the first switch number occurs during clock pulse n+1. In this manner, the loss of operation time which would normally be associated with a serial input scheme is circumvented. A serial input scheme was used because it offers the greatest compatability with serial command transmission, this being the most common method

of transmitting commands to the telemetry system.

As already stated, the feedback length control inputs determine which stage of the registers will be used as the feedback stage. The feedback control variables are formed from the outputs of a separate register. If the number of feedback stages is not extremely large, each stage of the register could be used to represent a feedback length control variable, with only one stage in the one state at any time. If the number of feedback stages is too large, then the n length control stages of the register can be used as the inputs to a decode matrix with 2^n outputs. Each of the 2^n outputs would be a feedback stage control variable.

Format Limitations

Unlike the CTTG, the SRTTG possesses no numerical adjacency requirements for the first level of commutation. However, it does possess the same numerical adjacency requirement for all lower levels of commutation.

Another important format limitation of the SRTTG is the fact that commutation below the second level can only occur in the last word of the previous level. The reason for this is the fact that the contents of the commutation registers below the second level are only added to those of the first level commutator when the next higher level register contains the last channel number for that sample.

Figure 17 shows an example of this limitation.

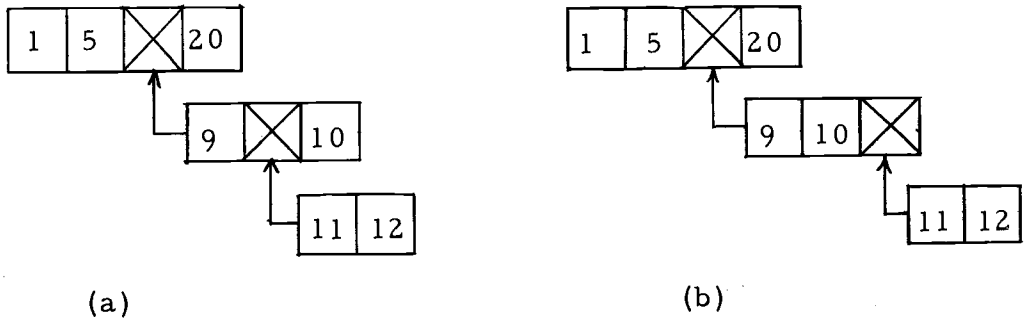


Figure 17. (a) Unrealizable commutation scheme,
(b) Realizable commutation scheme.

The most important format limitation of the SRTTG is the fact that sub-sequences in the same row but different levels must be numerically adjacent. This means that the first channel number of sub-sequence n+1, row i, must be numerically adjacent to the last channel number of sub-sequence n, row i.

Finally, as with the CTTG, format flexibility is also dependent upon the number of registers available. The more shift registers used, the more levels of sub-commutation which can be obtained.

V. DESIGN EVALUATION

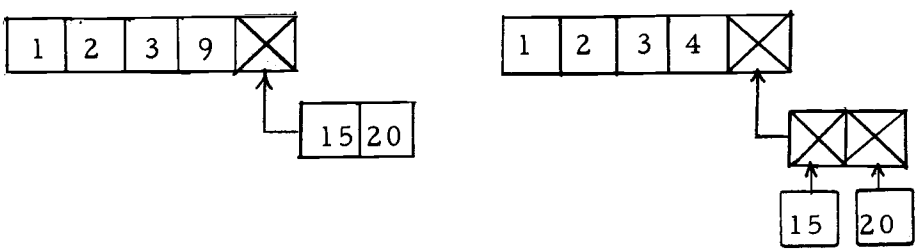
In Chapter II, the constraints and functional characteristics by which a design was to be judged were discussed, and in Chapters III and IV the functional characteristics of the CTTG and SRTTG were discussed. The majority of this chapter will be devoted to a comparison of the functional characteristics of the SRTTG and CTTG using the constraints and functional characteristics of Chapter II as standards. Portions of this chapter will also be devoted to discussions of reliability and methods of format flexibility enhancement.

Functional Characteristics

The necessary functional characteristics of a multiple format telemetry generator are external controllability, multiple sequence capability, synchronous operation, binary output, and hardware type solution. Both the CTTG and SRTTG meet all of these requirements; in fact, the only characteristics of both generators which are significantly different are sequence flexibility and control requirements. Both designs operate synchronously, are practical hardware solutions, and utilize binary logic signals as their outputs.

If we consider only those sequences which one system can generate while the other cannot, without placing any limitations on the number of elements in either system, then only the format

limitations discussed in Chapters III and IV need to be judged. There are two format limitations possessed by the SRTTG and not by the CTTG; they are, only one level in n sub-commutation per row and sub-commutation sequences in the same row and one level apart must be numerically adjacent. The CTTG possesses the capability to circumvent the first limitation when two or more level n+1 sequence generators are attached to the same level n generator. The lack of this format limitation would also sometimes allow an operator to circumvent the numerical adjacency requirement for sub-sequences. The commutation sequence of Figure 18 exhibits a non-adjacency which could be realized with a CTTG. The second limitation is eliminated by the CTTG regardless of its configuration.



(a) (b)

Figure 18. (a) Non-adjacent commutation scheme,
(b) CTTG circumvention of non-adjacency problem.

The control requirements of each system are entirely different. In the case of the SRTTG, if there are N levels of commutation and the main frame sequence is M samples long then (N) (M) + 2 control

words[4] are required. For the CTTG three control words for each sub-sequence generator, two control words for each main frame sequence generator, and 1+L control words[5] for the clock logic section are required.

The fact that the SRTTG does not require a control register for (N) (M) of its control words while the CTTG requires a control register for all control words is not of significance at present, since we are only concerned with what the operator must do in order to control the sequence generation. The advantages of not needing a control register for (N) (M) of its control words will be considered when the relative cost of the SRTTG is discussed.

A SRTTG with a five-bit channel number and a capability for two commutation levels would require:

$$(32)\ (2) + 2 = 66 \quad \text{control words}$$

A CTTG with sixteen main frame sequence generators and sixteen second level sequence generators would require 97 control words.[6]

---

[4] One control word for the feedback length control and one control word for the load reset command.

[5] Where L is the number of main frame sequence generators.

[6] Sixteen main frame sequence generators make possible the generation of any main frame sequence for a five-bit channel number. Sixteen second level sequence generators give the CTTG the same capabilities as the SRTTG with two levels of commutation.

From the preceeding, it can be said when both generators possess maximum flexibility, the CTTG offers an increase in format flexibility over the SRTTG. The maximum added flexibility is attainable only when the CTTG possesses the capability for at least three levels of commutation. It can also be said that the SRTTG requires fewer control word changes than the CTTG when the commutation sequence must be altered. This latter advantage is much less of an advantage than the former.

Constraints

In the preceeding section, it was shown that both types of designs are comparable from the standpoint of functional characteristics. In this section, a comparison will be presented of how well each design remains within the constraints of Chapter II.

System Cost

The system cost of both designs represents the second most important constraint, sequence flexibility being the most important. In Appendix B the cost of the CTTG and SRTTG is computed for five and six-bit channel numbers. As in the previous section, each design possessed the capability for two levels of commutation. From the calculations, it can be seen that the SRTTG is significantly cheaper than the CTTG. Although the SRTTG is less costly than the

CTTG, it is important to note that the CTTG, because of its greater modularity, can be decreased in cost by merely removing individual sequence generators. This is the most significant advantage of the CTTG over the SRTTG, increased cost flexibility, or greater ease of modification, due to its greater modularity. A CTTG could be built with only a few individual sequence generators and more could be added if and when necessary without making any significant changes in the original system. For the SRTTG, it would be comparably practical to add words to the length of the shift register; however, if another level of commutation is desired, it is necessary to add a set of two shift registers equal in length to the main frame register. Therefore, the SRTTG is less costly than the CTTG when both telemetry generators possess equal format flexibility; however, when less than maximum format flexibility is required the CTTG can be cheaper.

Control Logic and Control Variables

Preceeding sections have already discussed the cost, number of control variables, and required operator responses for each design. It is difficult to compare the control logic of the two designs since for the SRTTG the shift registers themselves function as both control logic and sequence generation logic; whereas, the CTTG the control logic is a separate entity from the sequence generation

logic. Neither design, however, exhibits significant advantages over the other with respect to control flexibility.

Sequence Flexibility

It is the purpose of this section to discuss the sequence flexibility of each design not from the standpoint of comparing one design with the other, but from the standpoint of overall sequence flexibility with respect to all possible commutation sequences. Sequence flexibility for both designs can be discussed in two parts, main frame sequence flexibility and sub-sequence flexibility. It is important to note that we are only concerned with sampling rates and not sampling order. For this reason, it makes no difference when sampling channels one and two, for example if channel one is sampled before channel two or vice versa.

The SRTTG possesses maximum main frame sequence flexibility when, for an n-bit channel number, it possesses the capability for generating main frame sequences of up to 2^n samples. For the CTTG, 2^{n-1} main frame sequence generators are necessary for maximum main frame sequence flexibility. Any main frame sequence of length greater than 2^{n-1} samples must possess channel number adjacencies so that a single generator can be used to produce more than one channel number. Therefore, both designs are capable of producing any main frame sequence required.

Sequence flexibility with respect to sub-commutation sequences is much more constrained. Both the CTTG and SRTTG, for the most part, require numerical adjacency for the channel numbers of a sub-commutation sequence. This requirement of both designs at times would make the generation of the most optimum channel number sequence non-feasible. The added constraint of sub-sequence adjacency further limits the SRTTG.

System Flexibility

The system flexibility of both designs is, in the opinion of this author, very good. Neither design was developed with respect to a specific commutator; therefore, either telemetry generator, assuming it possessed proper channel number lengths could be utilized in commutators requiring the aforementioned functional characteristics.

A major advantage of the CTTG with respect to system flexibility is the fact that if the required commutation sequences are known before the telemetry generator is installed, it is possible with the CTTG to use only as many individual sequence generators as necessary. With the SRTTG, it was shown previously that such flexibility would be less feasible.

Reliability

Neither of the two telemetry generators was designed under the

constraint of maximum reliability. However, in order to adequately compare both designs, the effect of component failures must be considered. Since the effects of any single component failure are dependent upon the actual commutation sequence, this discussion will be concerned with higher level failures such as the failure of a shift register or counter.

For the CTTG, failure of a sub-sequence generator would probably make that generator and all lower level generators in the same row unusable. All higher level generators and those not in the same row as the faulty generator would still be usable.

If one of the main frame sequence generators should fail, more difficulty would arise than in the case of a sub-sequence generator failure. Unlike the sub-sequence generators which can be removed from use at the perogative of the operator, the clock logic section enables the operator to use main frame sequence generator n+1 only after generator n is used. Therefore, if a main frame sequence generator failed, it would probably still have to be sampled for one sample period. This requirement might not be a problem in cases where the output of a malfunctioning generator were constant. However, if the generator failed in such a manner that its output were not constant, e.g. the reset gate failed, then at times an erroneous channel number would have to be tolerated.

There are two major types of clock logic failures; a failure in

which the output of the clock logic section remains constant, or a failure in which the output of the clock logic section is some sequence which is not in the control of the operator. For a constant output failure, only one main frame sequence generator could be used, greatly reducing sequence flexibility; however, one would still be able to interrogate all channels. For a sequential non-random output failure, the telemetry generator could still be of some use depending on the characteristics of the sequence. If the sequence was random, then the generator would be unusable because the operator would be unable to determine which channel, during a given sample period, was being sampled.

Decode logic section failures are of three types, failure of an AND-gate, failure of an OR-gate, and failure of an output flip-flop. Failure of either of the latter two logic elements would make it impossible to generate half of the channel numbers. This conclusion assumes that the logic elements fail in the one or zero output. Should one of the AND-gates fail in the zero output, it would restrict the possible channel number sequences for its sequence generator. Should the AND-gate fail in the one output, once again it would make it impossible to produce one-half of the channel numbers.

Failures similar to those already discussed are generally more catastrophic in nature for the SRTTG. If one of the flip-flops of a shift register should fail, then the longest main frame sequence

would have to use the feedback stage preceding that flip-flop as the last stage. A failure of the clock logic would cause the generator to produce a single channel number output, i.e. it would fail either in the shift or output mode. A failure of one of the output flip-flops would have the same effect as for the CTTG. When both designs possess maximum sequence flexibility, the SRTTG will be less likely to exhibit a component failure simply because it has fewer elements.

With respect to usability after a component failure, the CTTG exhibits a greater ability to circumvent component failures than the SRTTG. Unlike the SRTTG, because of its modularity, it is feasible to use redundant generators at each commutation level to account for generator failures. Finally, it should be noted that in order to determine which component of a generator has failed, it is necessary to be able to determine which channels are being sampled. This would require some knowledge of the relative magnitudes of the channel outputs since these and not the channel numbers are the only information available at the receiver.

Format Flexibility Enhancement

Both the CTTG and SRTTG exhibit format limitations. This section will be devoted to a discussion of ways in which the format flexibility of both designs can be increased.

Numerical Adjacency

As has already been discussed, both designs are seriously format limited for sub-sequences by the numerical adjacency requirement. The use of an additional shift register for each sub-commutation level could partially alleviate this problem for the SRTTG. The additional register could be used to allow the sub-sequence channel number to be increased by more than one digit each time it is sampled.

The CTTG could be partially alleviated of the numerical adjacency requirement if designed in the manner shown in Figure 19. In this case, a counter, comparator, two AND-gates, and a flip-flop have been added in order to allow variable generator count rates. If for instance, the output comparator value is 15, the reset value is one, and the counter comparator value is two, the output of the generator would be 1, 4, 7, 10, and 13. The obvious constraint is that the input comparator can be in the one state no more than once every three clock pulses.

The numerical adjacency constraint could be completely removed by using a comparator as the control for each channel switch. As shown in Figure 20, the output switch numbers would be the input to each of the switch comparators. The logic for the comparators is the same as that for the CTTG comparators. The price of this

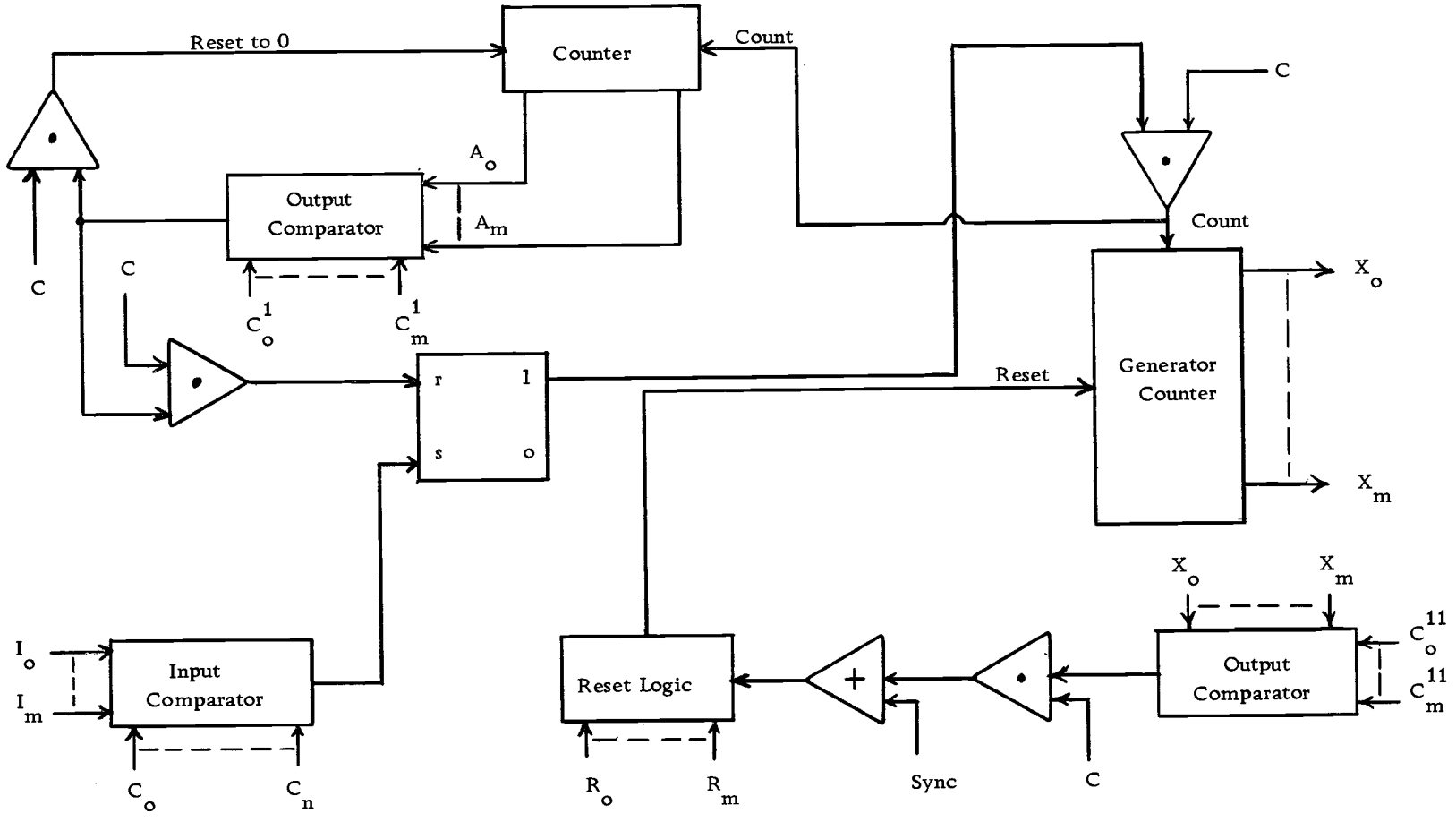


Figure 19. Variable count sub-sequence generator block diagram.

added flexibility is a significant increase in cost. As opposed to using 2^n n-input AND-gates for output decoding, the comparator method would require $n2^n$ EXCLUSIVE-OR-gates plus $n2^n$ additional flip-flops for the control register.

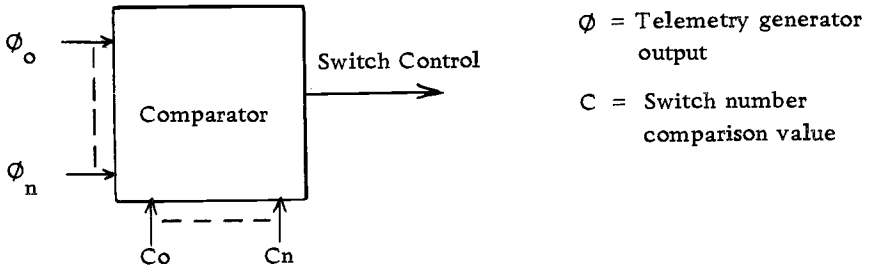


Figure 20. Switch control comparator.

Channel Number Selection

One of the most important considerations is the assignment of switch numbers for each channel. Because of the numerical adjacency limitations, it is necessary that switch numbers be grouped according to the sample rates for those channels which they represent. For instance, if channels ten and twelve were to be sampled at equal rates, it would be illogical to assign channel number eleven to a channel which was to be sampled at some other rate. By the proper assignment of switch numbers many of the limitations imposed by the adjacency requirements can be abated.

VI. CONCLUSIONS

As is evident from the discussion of the preceding chapters, any design which possesses some degree of format flexibility will be complex and costly as compared to previous single format telemetry generators. In this respect the two designs developed in this paper are both complex and costly. However, with the advent of integrated circuits, an even more inhibiting characteristic, excessive size, is partially eliminated. Shift registers of fifty-bits are now commercially available as integrated circuits. Binary counters, logic gates, and binary adders are also available commercially as integrated circuits.

The CTTG exhibited a significant degree of increased format flexibility over the SRTTG. Both designs, when equipped properly, i.e. possess the capability for more than one commutation level, can produce a great many different commutation sequences. Obviously, neither the SRTTG nor the CTTG could economically replace a telemetry generator requiring a single commutation sequence. However, if several sequences are required, either the SRTTG or the CTTG would probably represent a savings in system cost as opposed to using several individual telemetry generators.

The most difficult to answer question concerning the two designs is which design, the CTTG or the SRTTG, is the better. The question

is difficult to answer because one must first decide upon the relative merits of the various characteristics of both designs. From an engineering standpoint, the most important characteristics for a specific system are cost, format flexibility, and reliability. If maximum format flexibility or maximum reliability were the most important considerations, then the CTTG would be the most applicable design. If cost and maximum main frame sequence flexibility were the most important considerations, then the SRTTG would be the most applicable design. If all three characteristics were of equal importance, then the CTTG would be the most applicable design to use.

Finally, the most significant restriction of both designs is the numerical adjacency requirement. As previously discussed, this problem can be partially circumvented by proper channel number selection and/or by making changes in the logic elements of the generators, e.g. adding variable count generators to the CTTG. However, if these adjustments or a combination thereof do not permit the necessary format flexibility, then it would probably be necessary to program a computer to produce the required sequence, or use individual telemetry generators for each sequence.

BIBLIOGRAPHY

1. Blaine, G. James and G. E. Brechling. A versatile aerospace telemetry system. In: Aerospace Systems Conference, Seattle. p. 362-371. (IEEE Publication 10 C 22) (Supplement to IEEE Transactions on Aerospace and Electronic Systems, vol. AES-4, no. 4)

2. Carl, Christopher. Surveyor telemetry processing system. In: Conference on Computer Technology, Manchester, England, 1967. London, 1967. p. 214-221. (Institution of Electrical Engineers. Conference Publication no. 32)

3. Cory, J. D. and P. Garner. A micro-PCM encoder and multiplexer. In: Proceedings of the 1965 National Telemetering Conference, Houston. [Pittsburgh] Instrument Society of America, 1965. p. 121-125.

4. Foster, Leroy Edward. Telemetry systems. New York, Wiley, 1965. 308p.

5. Gatland, Kenneth William (ed.). Telecommunication satellites. London, Iliffe, 1964. 441p.

6. Hardin, R. H. A multiple format telemetry programmer. In: Proceedings of the 1967 National telemetering Conference, San Francisco. [New York] American Institute of Aeronautics and Astronautics, 1967. p. 190-195.

7. Jet Propulsion Laboratory. Mariner Mars 1969 software system TCP real time telemetry. Pasadena, June 1, 1968 (SD-69-5-2000A)

8. Lenk, John D. Understanding telemetry circuits. New York, Bobbs, Merrill, 1966. 160p.

9. Levy, C. Concepts for a stored program data acquisition system. In: Proceedings of the 1967 National Telemetering Conference, San Francisco. [New York] American Institute of Aeronautics and Astronautics, 1967. p. 190-195.

10. Stiltz, Harry L. (ed.). Aerospace telemetry. Englewood Cliffs, New Jersey, Prentice Hall, 1961, 2 vols.

11. Tepper, Marvin. Fundamentals of radio telemetry. New York. Rider, 1959. 116p.

APPENDICES

APPENDIX A

Commutation Examples

This appendix will be devoted to a discussion of the methods for determining the input control variables for the CTTG and the SRTTG after one has been given the channel sampling rates. One of the commutation formats of the Mariner '69 spacecraft has been used as the example (7). For the SRTTG it is assumed that there are three commutation levels and the CTTG is assumed to be of the configuration shown in Figure 21. As with both designs, in order to determine the input control variables, the simplest procedure is to first develop a commutation diagram and from that determine the necessary control variables.

Table 2. Channel Sampling Rates for Mariner '69 Spacecraft.

Number of Channels	Rate at Which Each Channel is Sampled
17	N[1]
27	N/10
10	N/100
40	N/200

[1] N is the main frame or highest sampling rate in the sequence.

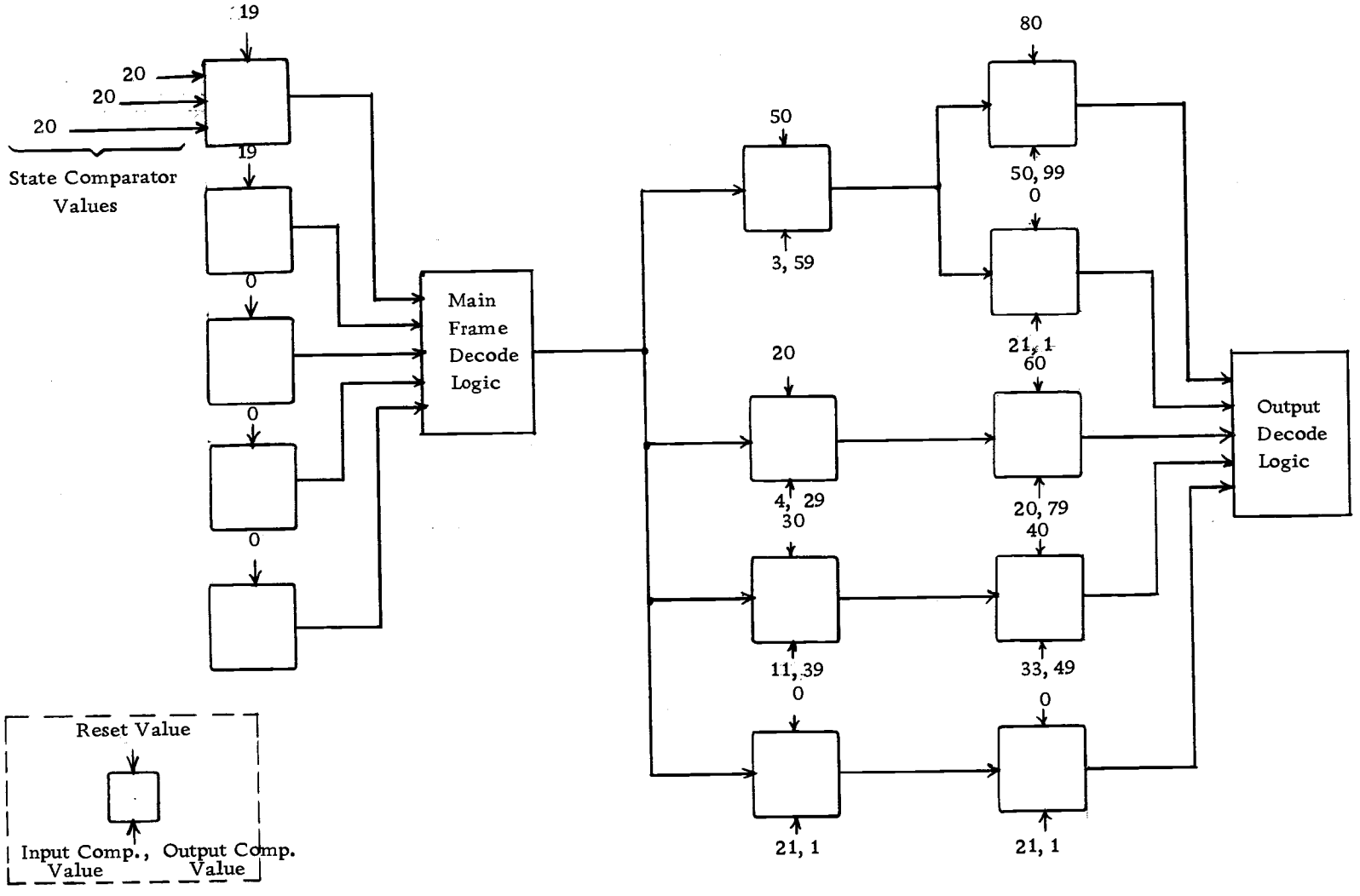


Figure 21. CTTG commutation example.

Commutation Diagram

It can be shown that the rate at which a channel is sampled with respect to its position in a commutation diagram is the following:[2]

$$\text{(sampling rate for a)} = N/(X_{n,i})(X_{n-1,i}) \cdots (X_{2,i})$$
$$\text{(level n row i channel)}$$

From Table 2, it can be seen that there are four distinct sampling rates. For the 27 channels at rate N/10, three level two sequences of ten samples each would suffice, leaving three unused samples. For the ten samples at rate N/100, one level three sequence of ten samples in the same row as a level two sequence of ten samples will give the desired sampling rate. This means that a level two sequence of nine samples and one unused sample for its level three sequence would actually be needed. For the 40 channels at rate N/200, two groups of twenty samples each in the same rows as level two sequences of ten samples each would suffice. Therefore, from the sampling rates of Table 1, the commutation diagram of Figure 22 can be developed. Since no specific channel numbers were given, they were assigned so as to require only one main frame sequence generator.

In this case, the sampling rates for each channel were chosen

---

[2] $X_{n,i}$ is the number of level n row i samples in the commutation diagram.

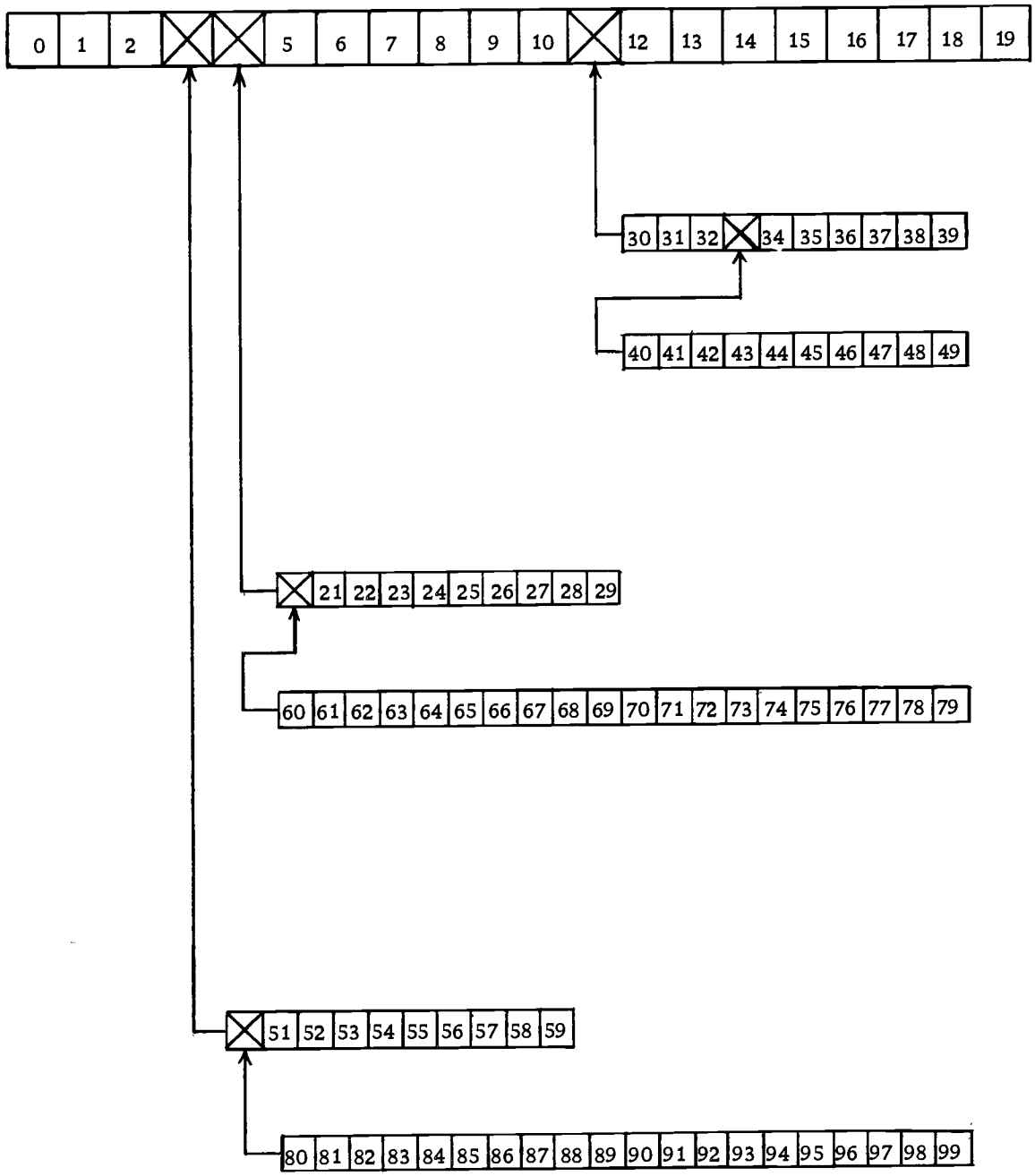


Figure 22. Commutation diagram for Mariner '69 commutation scheme - CTTG.

so that a commutation diagram would correspond exactly to all sampling rates. However, many times this will not be the case; for example, if there were only two channels to be sampled, one at rate N and the other at rate N/2 a commutation scheme could not be developed which would duplicate this scheme without possessing unused sampling periods. Therefore, as will often be the case, certain channels will have to be sampled at rates higher than their frequency contents require.

CTTG Control Variables

Once the commutation diagram has been developed, all that is required is to load the control register with the necessary input comparison, output comparison, and reset values. For the example, only one main frame sequence generator would be required since there are no non-adjacent channel numbers. The required reset value for main frame sequence generator one is zero; this value can also be used for each of the unused main frame sequence generators. Since the main frame sequence is 20 samples in length, the maximum value comparator comparison value should be 19. The comparison value for the number one comparator of the clock state counter should be zero since this is the first value of the clock counter sequence. The remaining comparator values should be >19 so the clock state counter output will always be a zero. As an example

of the control variables for a sub-sequence generator, consider the sequence of channel numbers 20 through 29. The reset value for the generator producing this sequence would be 20 and the output comparison value would be 29. The input comparison value would be four since that is the value of the main frame sequence generator during that sub-sequence sample. For any sub-sequence generator not used, it is only necessary that its input comparison value be set to a value so that its input comparator output is never a logical one. Using the preceding procedure for each sequence generator, the control variable values of Figure 21 can be obtained.

SRTTG Control Variables

Because of certain of the format limitations of the SRTTG, the commutation diagram of Figure 22 must be altered as shown in Figure 23. The channel numbers have also been changed in Figure 23, but the sampling rates are the same as those in Figure 22.

Using Figure 23, since the main frame sequence length is 20 samples, the feedback length control value[3] is 19. Figure 24 shows the values which must be loaded into each shift register in order to produce the desired commutation sequence. If a given sample of

---

[3] The first word of each register cannot be used as a feedback stage, thus eliminating the need for one set of feedback gates.

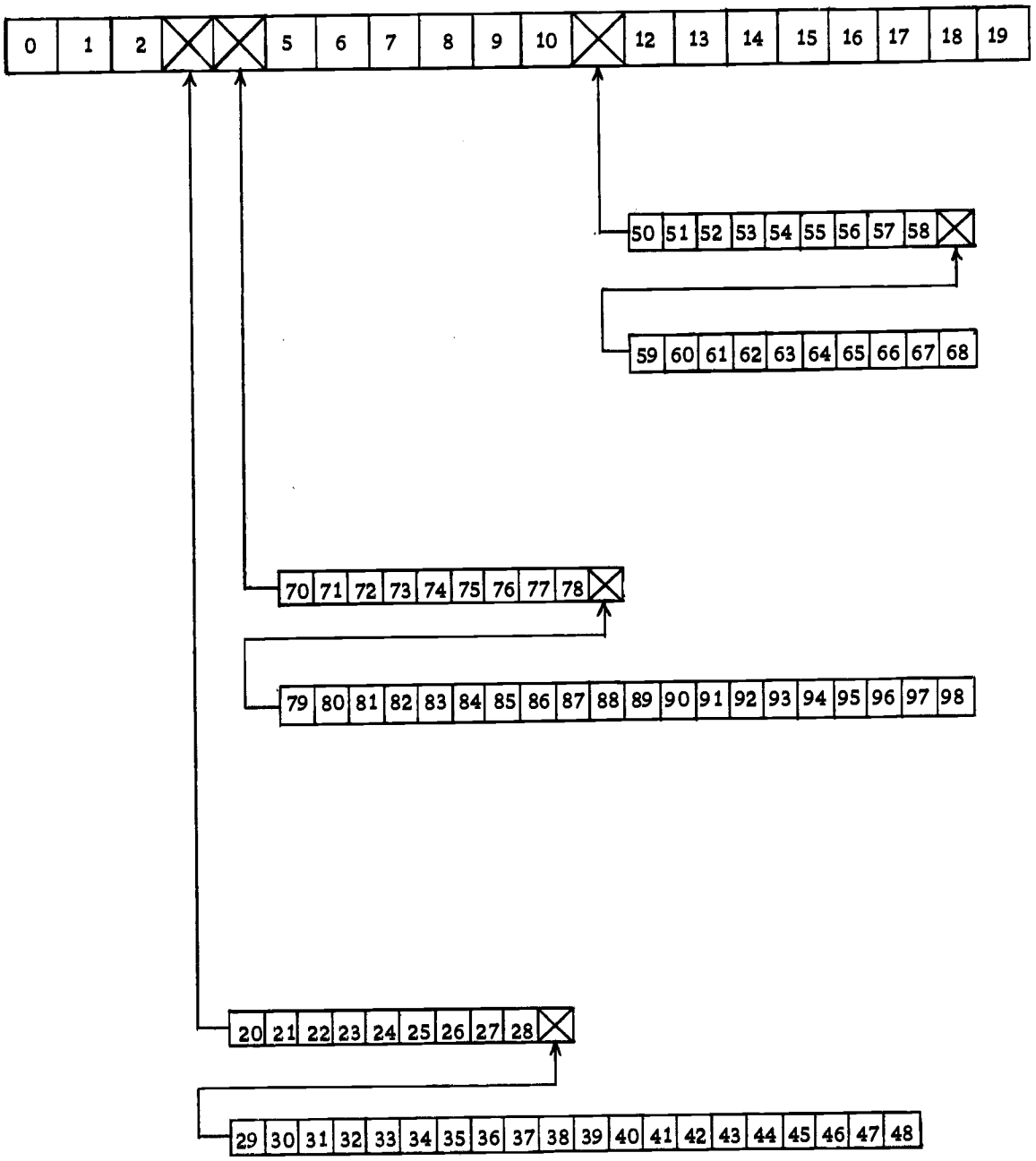
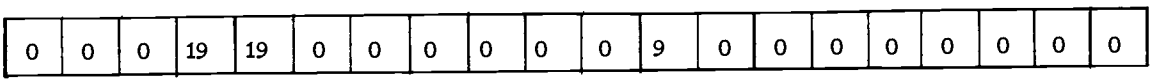
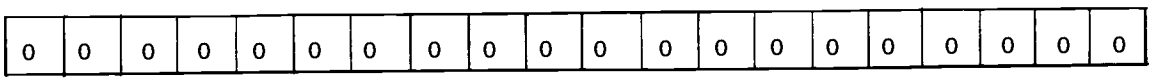


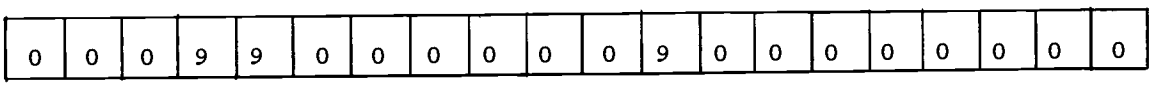
Figure 23. Commutation diagram for Mariner '69 commutation scheme - SRTTG.



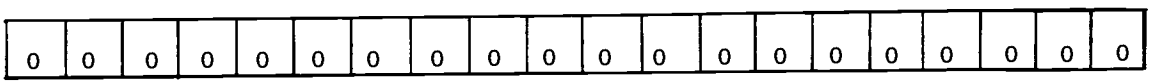
Register 5



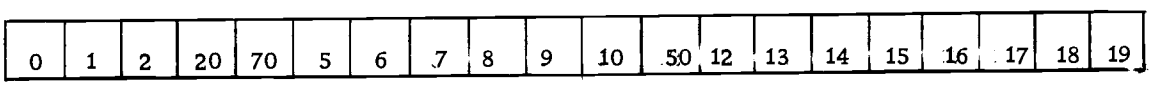
Register 4



Register 3



Register 2



Register 1

Figure 24. SRTTG input values for Mariner '69 commutation scheme.

register three contains a value other than zero, then the value of register one for that sample is the minimum channel number for the second level sequence and the sum of the values of registers one and three is the maximum channel number. If a given register five sample contains a value other than zero, then the sum of the values of registers one and three is the minimum third level channel number, and the sum of the values of registers one, three, and five is the maximum third level channel number in that row.

APPENDIX B

Cost Diagnosis

CTTG Cost Diagnosis

It is assumed that there is an unlimited number of inputs for each gate.

M_s = number of sub-sequence generators

M_m = number of main frame sequence generators

n = number of bits per channel number

Sub-Sequence Generator

$$\text{Cost} = (M_s)(5)(n+1)$$

Main Frame Sequence Generator

$$\text{Cost} = (M_m)(3n + 2)$$

Clock Logic Section

$$\text{Cost} = 2n + (M_m)(n+1) + 5 + Q$$

Where Q is the smallest integer $\geq \log_2 (M_s)$

Decode Logic

$$\text{Cost} = (n)(M_m + 1) + (n)(M_s + 4)$$

Control Register

$$\text{Cost} = ((M_s)(3) + 2M_m + 1)(N)$$

Total Cost:

$$\text{Cost} = (n)(8 + 7M_m + 9M_s) + 5M_s + 3M_m + 5 + Q$$

For $n = 5$, $M_s = 16$, and $M_m = 16$

$$\text{Cost} = 5(8 + 112 + 1444) + 80 + 48 + 5 + 4$$

$$\text{Cost} = 1457$$

For $n = 6$, $M_s = 32$, and $M_m = 32$

$$\text{Cost} = 3386$$

SRTTG Cost Diagnosis

n = number of bits per channel number

L = number of commutation levels

$N = 2^n$

Shift Register Cost

$$\text{Cost} = (N)(n)(1 + (L-1)\,2)$$

Adder Cost[4]

$$\text{Cost} = (n)(L-1)$$

Register Manipulation Logic Cost

$$\text{Cost} = (L-1)(n\ 4) + (n)(L'-2) + (3)(L-1)$$

$$L' = \begin{matrix}(L \text{ if } L_1 \geq 2)\\ (2 \text{ if } L_1 < 2)\end{matrix}$$

[4]It is assumed that a single unit will suffice for each adder.

Output flip-flop logic cost

$$\text{Cost} = 2n$$

Feedback Length Control Logic Cost

$$\text{Cost} = (2^n-2)\,2 + 1 + n + (2^n-2)\,2 + 1$$

Clock Logic Cost[5]

$$\text{Cost} = R$$

Load Reset Control Logic Cost

$$\text{Cost} = 1$$

Total Cost:

$$\text{Cost} = n\,((2)\,(L)\,(N) - N + (3)\,(L)-1) - 12 + R + 4N$$

For $n = 5$, $N = 32$, $L = 2$, and $R = 10$

$$\text{Cost} = (5)\,(128 - 32 + 3) - 12 + 10 + 128$$

$$\text{Cost} = 625$$

For $n = 5$, $N = 64$, $L = 2$, and $R = 10$

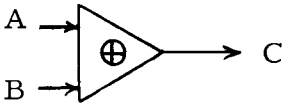
$$\text{Cost} = 928$$

---

[5]The number of logic elements in the clock logic section is a variable function of channel number word length.

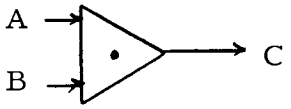
APPENDIX C

Next State and Truth Table Representations For Logic Elements



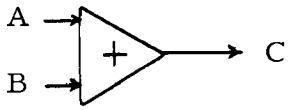
$A \oplus B = C$

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0



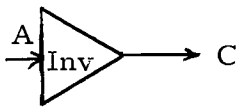
$A \cdot B = C$

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1



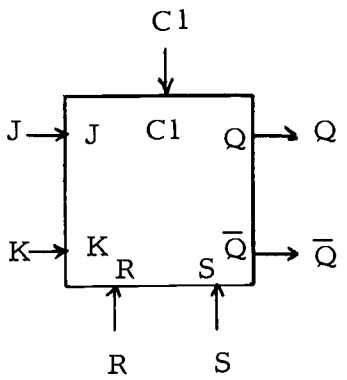
$A + B = C$

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1



$\overline{A} = C$

A	C
0	1
1	0



Clocked, J-K, R-S flip-flop

$$Q^{n+1} = \overline{R} \cdot \overline{S} \cdot Cl \cdot (\overline{K} \cdot Q^n + J \cdot Q^n) + \overline{J} \cdot \overline{K} \cdot \overline{R} \cdot Q^n + S$$

$R \cdot S \neq 1$

$Q^n \triangleq$ Output value of flip-flop Q during clock pulse n