REDUNDANCY TECHNIQUES TO IMPROVE THE RELIABILITY OF TWO LEVEL AND THREE LEVEL LOGIC CIRCUITS

By

Kasivisvanathan Vairavan

Bachelor of Engineering (Electrical)

June 1962

The University of Madras, India

A master's problem submitted to the Faculty of the School of Engineering and Applied Science of The George Washington University in partial satisfaction of the requirements for the degree of Master of Science in Engineering

June 1965

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

PREFACE

The object of this paper is to investigate into redundancy techniques to improve the reliability of logic circuits and to present some new concepts.

The author wishes to acknowledge the guidance and support given to him by Dr. N. T. Grisamore and Professor D. K. Anand, in the presentation of this paper.

CHAPTER I

INTRODUCTION

Reliability has assumed great importance in the past decade, chiefly because of the highly complex nature of some systems and the importance of their missions. Reliability may be defined as the probability that at any given time a system or equipment will operate within the prescribed range of precision and rate of performance. The deviation of the performance beyond the specified limits is considered as failure.

The conventional techniques for achieving high reliability include ensuring high component reliability, conservative design, one hundred per cent screening of parts, improved manufacturing and production processes and detailed analysis of failure reports and proper corrective action.

Certain applications, which include defense and space research satellites, demand very high reliability of systems and the conventional techniques may fail to yield the desired level of reliability. Failures may have to be avoided at any cost. Maintenance action(replacement of failed parts) may not be possible. In such cases, the lack of reliability could be a serious problem. The problem, in brief, is to construct reliable systems from less reliable components. This has been studied for a long time and redundancy has been found to be a natural solution.

CHAPTER II

REDUNDANCY

What is "Redundancy"? Any material or information, which forms a part of a system but which is not absolutely necessary for the proper operation of the system under normal conditions, is considered redundant. But when there is a malfunction in the system this redundant material or information could be designed to prevent failure of the system.

Redundancy in a digital system may be in the form of hardware or information. In the former type of redundancy, a single subsystem or a logic unit is replaced by redundant units. Obvious disadvantages of this type of redundancy are increased weight and cost; but with the recent rapid advances in microminiaturiazation, these do not present too big a problem.

Information redundancy is said to exist when a unit of information has more bits than are absolutely necessary to represent it. Error checking codes are a good example of information redundancy.

The purpose of this paper is to investigate some of the important redundancy techniques and to present some additional concepts. Both two and three level logics have been considered. An attempt has not been made to explore deeply into any of the additional concepts, although the applications of some of them have been briefly indicated.

CHAPTER III

REDUNDANCY TECHNIQUES IN TWO LEVEL LOGIC CIRCUITS

Majority Organ

Several authors including Moore and Shannon,[1] Tryon,[2] von Neumann[3] and Maithra[4] have developed different schemes to improve the reliability of systems through redundancy. The most basic and yet effective concept was introduced by von Neumann, using the principle of Majority Organ.

The use of majority organ for triple redundancy is as follows: The system is divided into a number of subsystems and each subsystem is replaced by three identical subsystems.

---

1 E. F. Moore and C. E. Shannon, "Reliable Circuits Using Less Reliable Realys", Jour. of the Franklin Inst., 262, 191-208 (Sept. 1956), 281-297 (Oct. 1956).

2 J. G. Tryon, "Quadded Logic", Redundancy Techniques for Computer Systems , Spartan Press, 1962.

3 J. von Neumann, "Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components", Automata Studies, Princeton University Press, 1956.

4 K. K. Maithra, "Stability of Logical Networks and its Application to Improvement of Reliabililty", IRE Trans. on Circuit Theory, Sept. 1961.

The outputs of the redundant subsystems are restored by the majority organ or the vote taker (Fig. 1). The output of the majority organ is the same as the majority of the inputs.

The reliability of the redundant configuration is

$$R_{ss} = r_{mo}\left[r^3 + 3r^2(1 - r)\right] = r_{mo}\left[r^2(3 - 2r)\right] \quad \ldots(1)$$

where r_{mo} = the reliability of the majority organ

r = the reliability of the non redundant system

A plot of the subsystem reliability R_{ss} is shown in figure 2, for the case of the perfect majority organ ($r_{mo} = 1$). It is seen that improvement in reliability is achieved only when $r > 0.5$.

For the more general case,

$$R_s = \left\{r_{mo}\left[\sum_{i=n+1}^{2n+1}\binom{2n+1}{i} r^i (1-r)^{2n+1-i}\right]\right\}^m \quad \ldots(2)$$

where R_s	- the redundant system reliability
m	- the number of subsystems into which the system is broken
r	- the non redundant subsystem reliability
$2n+1$	- the number of redundant subsystems
r_{mp}	- the reliability of the majority organ

The principle used in arriving at (2) is that the majority of the redundant units and the majority organ function properly to ensure success.

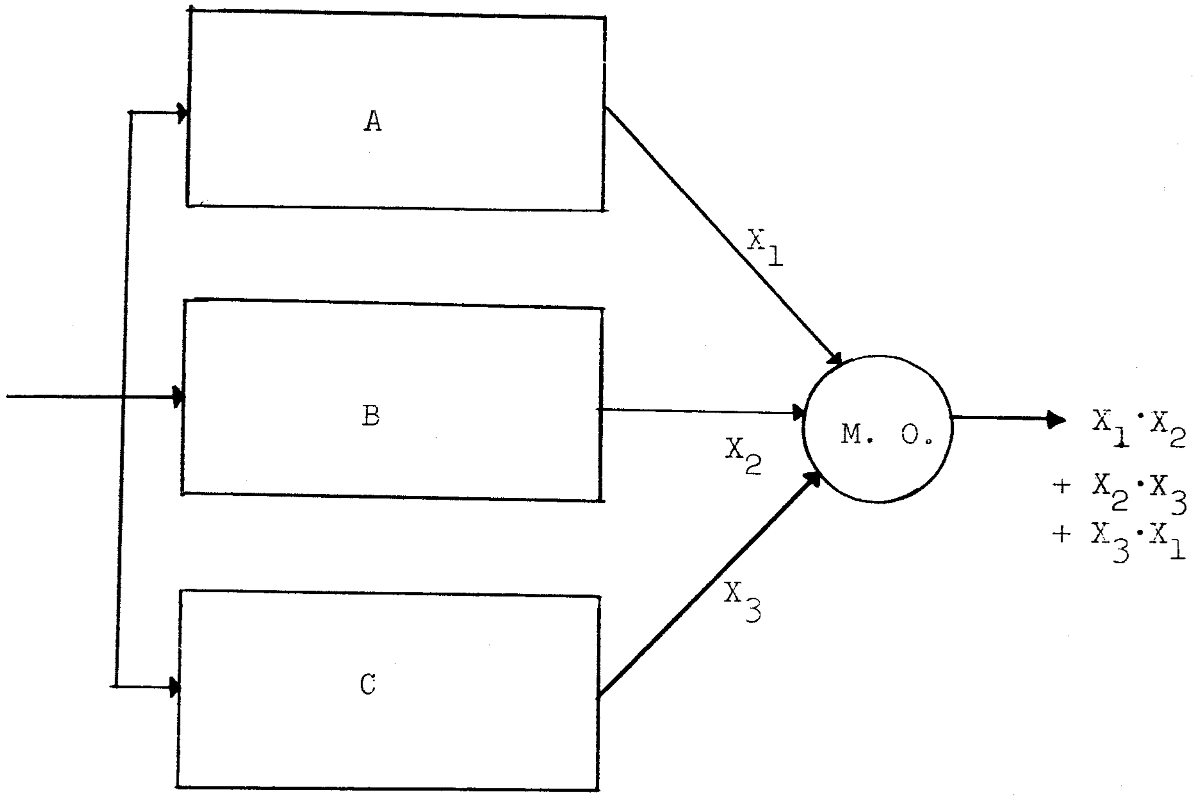


Figure 1 Triple Redundancy using Majority Organ

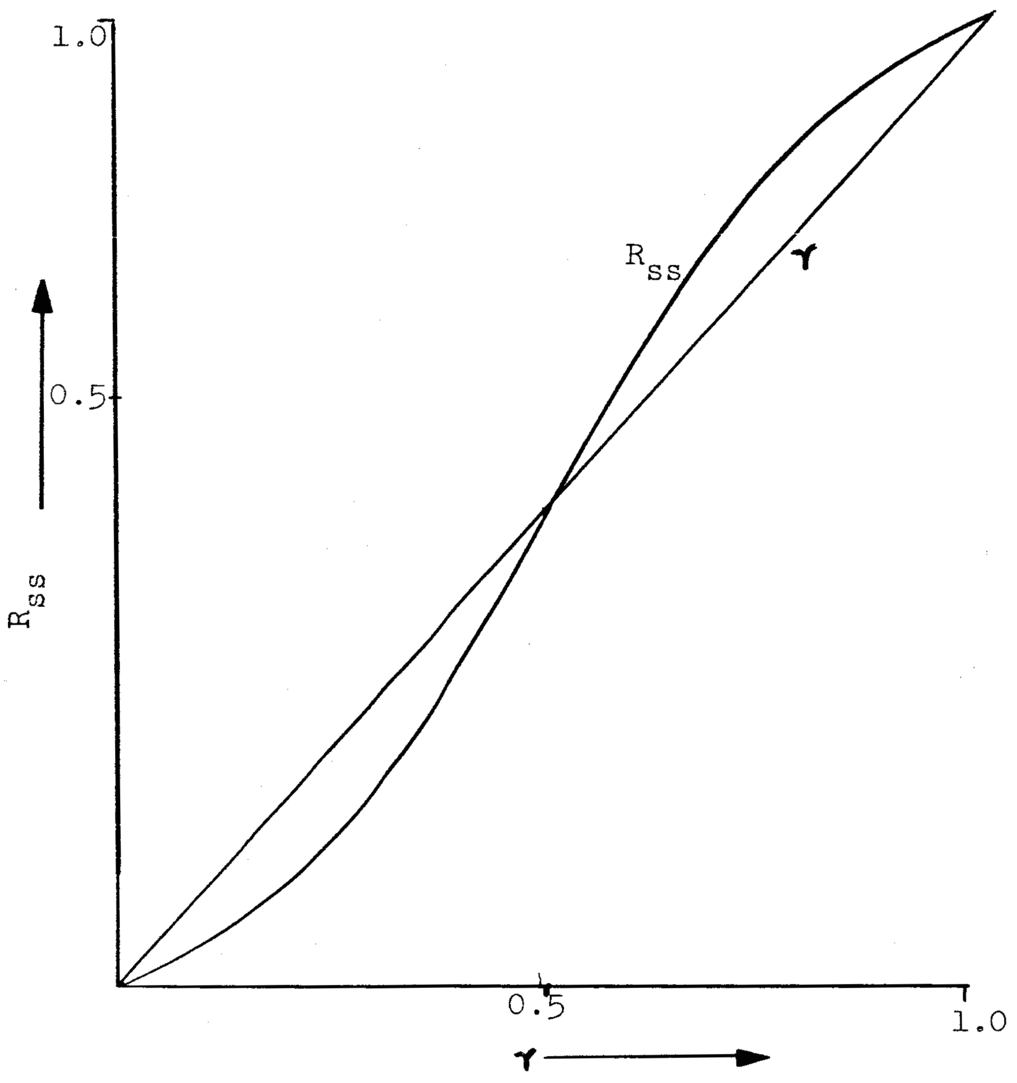


Figure 2 Redundant System Reliability

Detailed investigations[5,6] have been made into this technique and quantitative results have been established. It has been shown that the lower the level at which redundancy is introduced, the higher the reliability. In other words, the larger the number of subsystems, the higher the reliability. A closer examination of this is made in the Appendix B.

Grisamore and de Pian[7] have shown that in certain cases, reliability higher than that of the majority scheme could be achieved by choosing a proper function $s = f(x_1 x_2 \ldots x_n)$, where "s" is the output of the redundant configuration and $x_1 x_2 \ldots x_n$ are the outputs of the redundant units. The authors have described a linear average,

$$s = \frac{\Sigma x_i}{n}$$

although this is not the best function. Subsequently, Tsoukias[8] has shown that a nonlinear average,

$$s = \left[\frac{x_1^a + \cdots + x_n^a}{n}\right]^{1/a}$$

yields better results than the linear average method.

---

5 J. K. Knox Seith, "Improving the Reliability of Digital Systems by Redundancy and Restoring Organs", Stanford Electronics Laboratories, Technical Report, No. 4816-2

6 L. E. Dostert, Jr., B. A. McGill, D. O. Baechler, "Application of von Neumann Redundancy Techniques to the Reliable Design of Digital Computers", ARINC Research Corporation Final Summary Report, 167-1-293, April 1962.

7 L. de Pian and N. T. Grisamore, "Reliability Using Redundancy Concepts", IRE Trans. on Reliability and Control, 1960

8 Panos M. Tsoukias, "A study of the Averaging Method for Reliability Improvement", A Master's Thesis of the George Washington University.

Pierce[9] has proposed a scheme in which the outputs of the redundant units have "weighted" votes. A particular case of interest is when the weights are 1 and 0. The output of the redundant unit is compared with the output of the majority organ (which is assumed correct) and when the number of times the former differs from the latter exceeds a pre-specified value, the former is cut out.

Although a number of improved concepts have been proposed, the majority scheme is very practical and effective because of the simplicity and the ease with which it is realized. Whereas most of the proposed schemes are yet to be effectively implemented, the triple redundancy technique using majority principle has been successfully applied to the circuits of the computer used in the Saturn V Launch Vehicle.[10,11] Triple modular redundancy used in the above computer realizes

---

[9] W. H. Pierce, "Improving Reliability of Digital Systems by Redundancy and Adaption", Stanford Electronics Labs., Technical Report No. 1552-3.

[10] Dr. D. P. Rozenberg and H. L. Ergott, "Modular Redundancy for Spaceborne Computers", International Business Machines Corporation Internal Report No. 62-825-494. Oct. 1962.

[11] M. M. Dickinson, J. B. Jackson and G. C. Randa, "Saturn V Launch Vehicle Digital Computer and Data Adapter"--IBM Report No. 64-825-1179. September 1964.

twenty fold increase in reliability with three and one half times more components than a non-redundant system. Computer logic is divided into seven sections, each of which consists of three identical modules. The scheme is outlined in figure 3. The disagreement detector (D. D.) signals to the ground whenever the module outputs are not identical.

Dual Redundancy

Let us now explore the possibility of using two identical units, whose outputs are X_1 and X_2. When the two units function properly, X_1 and X_2 are identical. C is a restoring unit and has two outputs Y_1 and Y_2. $Y_1 = X_1 \cdot \overline{X_2} + \overline{X_1} \cdot X_2$ is an error detecting signal and is "on" when X_1 and X_2 disagree. Y_2, the restored output, may be a logical "AND" or "OR" of the inputs X_1 , X_2. In other words,

$$Y_2 = X_1 \cdot X_2 \text{ or } Y_2 = X_1 + X_2$$

Although no great gain in reliability is achieved by this scheme it could be effectively used once details of circuits and logic are known. The exact extent of its use or the imporvement in reliability achieved depends on the nature of failure. Let us consider the case when A fails. If A fails in such a way that its output is always a logical 1, then $Y_2 = X_1 \cdot X_2$ would yield $Y_2 = X_2 \cdot 1 = X_2$, which is correct. However, if A fails in such a way that its output

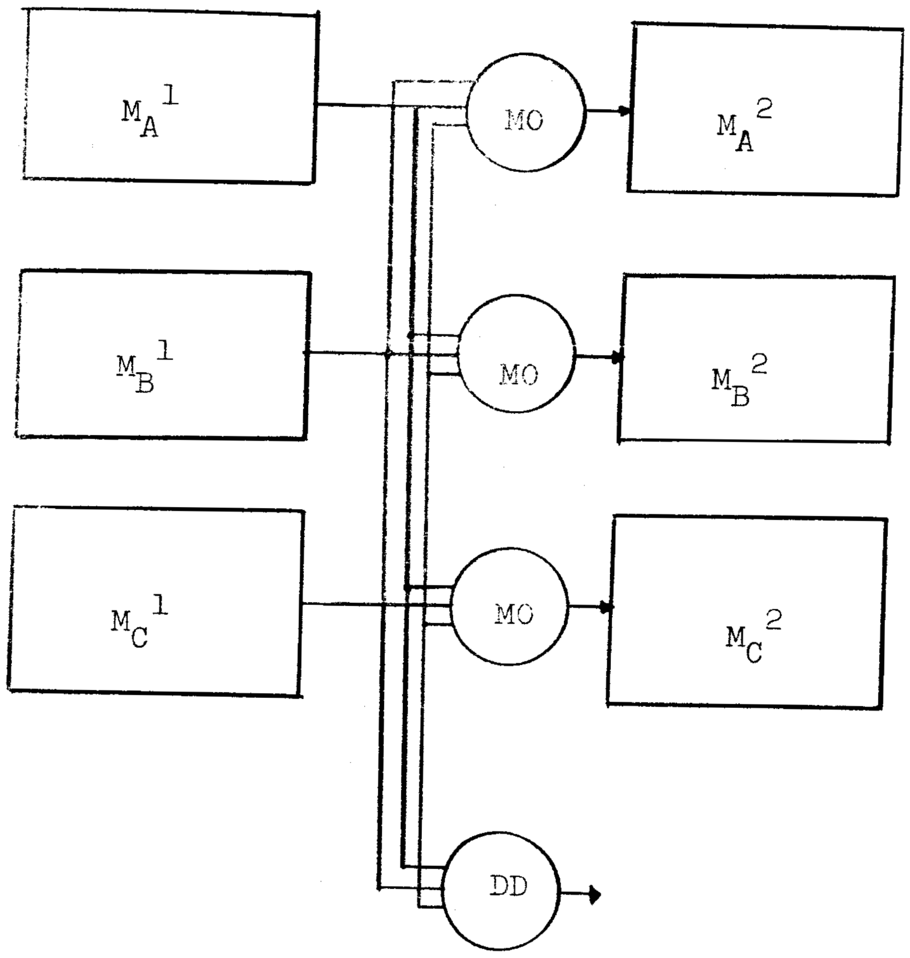


Figure 3 TRIPLE MODULAR REDUNDANCY WITH INPUT VOTING

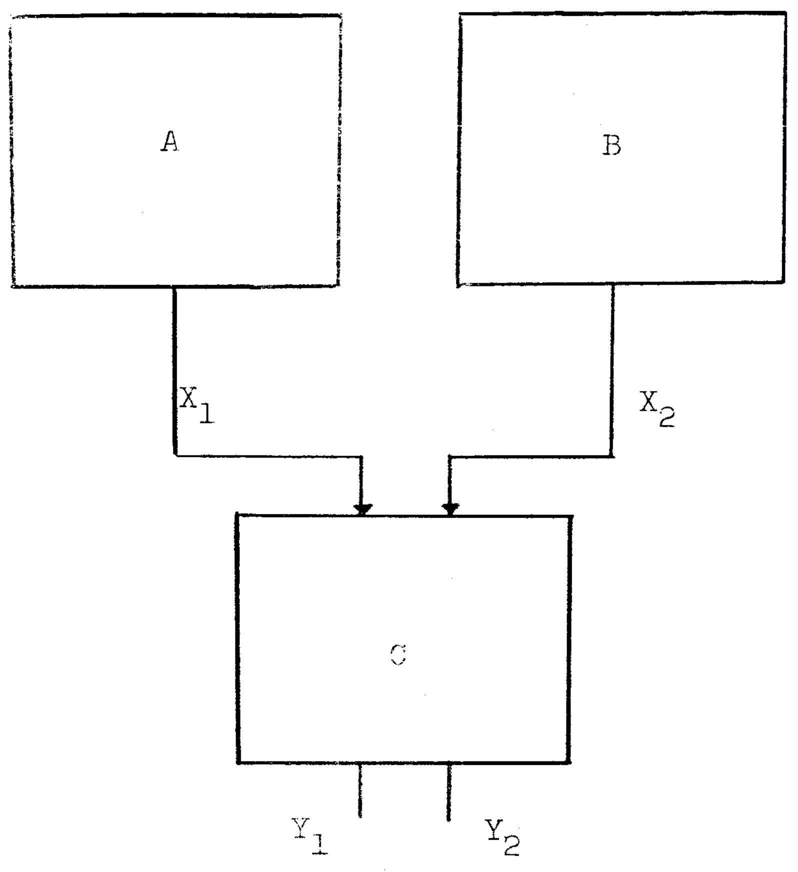


Figure 4 DUAL REDUNDANCY (A AND B IDENTICAL UNITS)

is always a logical 0, $Y_2 = X_2 \cdot 0 = 0$, which is the correct output only when X_2 is zero. Thus if a logical 1 is a more likely error, that is the output is 1 when it should actually be 0, $Y_2 = X_1 \cdot X_2$ would be more reliable. However if a logical 0 is a more likely error, that is the output is 0 when it sould actually be 1. $Y_2 = X_1 + X_2$ is more reliable. This is easily seen.

Let us now consider the "worst case" failure, the case in which the circuit yields a 0 when the correct value is 1 and 1 when the correct value is 0. Consider the following units of information:

a) 0110110 correct output information unit.

b) 1001001 output when the non-redundant unit fails.

c) 0000000 output, when $Y_2 = X_1 \cdot X_2$ and A fails (worst case failure).

d) 1111111 output when $Y_2 = X_1 + X_2$ and A fails.

An examination of the above shows that in the case of the worst case failure of a non-redundant unit, the "distance" (binary) between the correct and the incorrect information units is the length of the information unit itself. But, for the redundant configuration, the "distance" between the correct output and the incorrect output is reduced. In the example given above, the "gain in distance" using redundancy is 3 bits for $Y_2 = X_1 \cdot X_2$ and 4 bits for $Y_2 = X_1 + X_2$. It should be noted that the "gain" so achieved is also a function of the bits of the information unit. For, if the correct information unit was 1111111, $Y_2 = X_1 \cdot X_2$ (when A fails) yields 0000000 which would also have been the output if no redundancy was used. Thus no gain in distance is achieved. The above information block is the worst case for $Y_2 = X_1 + X_2$!

Let us now turn our attention to the signal Y_1. We recall that Y_1 detects failure only when one of the units (A or B) fails, not when both fail. A single signal Y_1 may mean a temporary failure. When Y_1 is continuously 1, it may mean a permanent failure. In such a case, we could stop processing and test for the failed configuration by running a test sequence (the correct output of which is known beforehand) and isolate the failed unit. Human interference is not totally necessary for this procedure. For, when the disagreement detector detects an arbitrarily long sequence of disagreeing bits, a control signal could be generated which automatically carrys out the diagnostic procedure. We shall not attempt a detailed investigation of this here.

The technique described in this section is effective in the following respects: (1) It detects the failure of a circuit (only when one of the two units fails). (2) The binary "distance" between the correct and incorrect information units is reduced generally. (3) When one of the redundant blocks fails, this scheme could be dèsigned to run a diagnostic sequence, which may or may not need human interference.

Information Redundancy

In section 1, the Majority Organ method of improving reliability in two level logic circuits was outlined. In section 2, the principle of dual redundancy was considered. Both the above techniques used hardware redundancy. In this section, we introduce the concept of "Information Reliability" and investigate into Information Redundancy.

The Concept of Information Reliability

When we start processing or handling information, we can assume that the information is correct or the probability of its being correct is 1. As the information is processed, or in other words, as the data bits pass through logic systems and storage devices, redundant or not, the processed information is likely to be not one hundred per cent correct, because of the failure of circuits(temporary or permanent) and noise. Thus it is easy to see that as information bits pass through physical devices, the probability that a bit is wrong increases. We could define the reliability of information as the probability that it is correct. Suppose we have a system whose reliability is 0.99 at a given time of operation; this could be interpreted to be the value of the reliability of the output information when the input information is cent per cent reliable.

The concept of reliability of information could be extended to information handled by any physical system, including storage, transmission, retrieval, where errors are likely to be committed. When we speak of increasing the reliability of systems, we mean to enhance information reliability, or more correctly, we wish to keep up the reliability of information as it is handled and processed because decisions of varying importance may be made based on it. The concept of information reliability is rather abstract. It is not easy to arrive at a mathematical expression representing information reliability. Before we estimate the reliability of information, we should be able to estimate the reliabilities of all physical devices handling the information unit.

Just as hardware redundancy was a natural method to enhance system reliability, information redundancy appears to be a possible method for improving information reliability.

Error detecting and correcting codes are examples of information redundancy. Generally the codes are used in information storage and transmission. The units of information have parity check bits, in addition to the information bits.

Although the codes have been applied to logic circuits, they are not thoroughly suited to logic systems. The difficulty arises from the fact that special encoders and decoders have to be designed for logical units. In this section a different approach is adopted.

Information Repetition

When a man computes, he may repeat his computation in order to check the validity of his original result. The same principle could be extended to a digital system. Each cycle of operations in a logic system could be repeated to ensure correct results.

When a circuit fails permanently the above procedure is not effective because the same errors are committed over and over. However, in some cases there may just be a temporary failure or noise interference. This is highly probable in space vehicles and other systems subjected to noise. In such cases information redundancy in the form of information repetition may be effective.

In the following paragraphs, a system, which uses the technique of repetition of logic operations, is illustrated.

In figure 5, A is a logic system with inputs X_1, X_2, ...X_n and outputs Y_1, Y_2,...Y_m. The set of flipflops B, C, and D serves the purpose of storing the processed information.

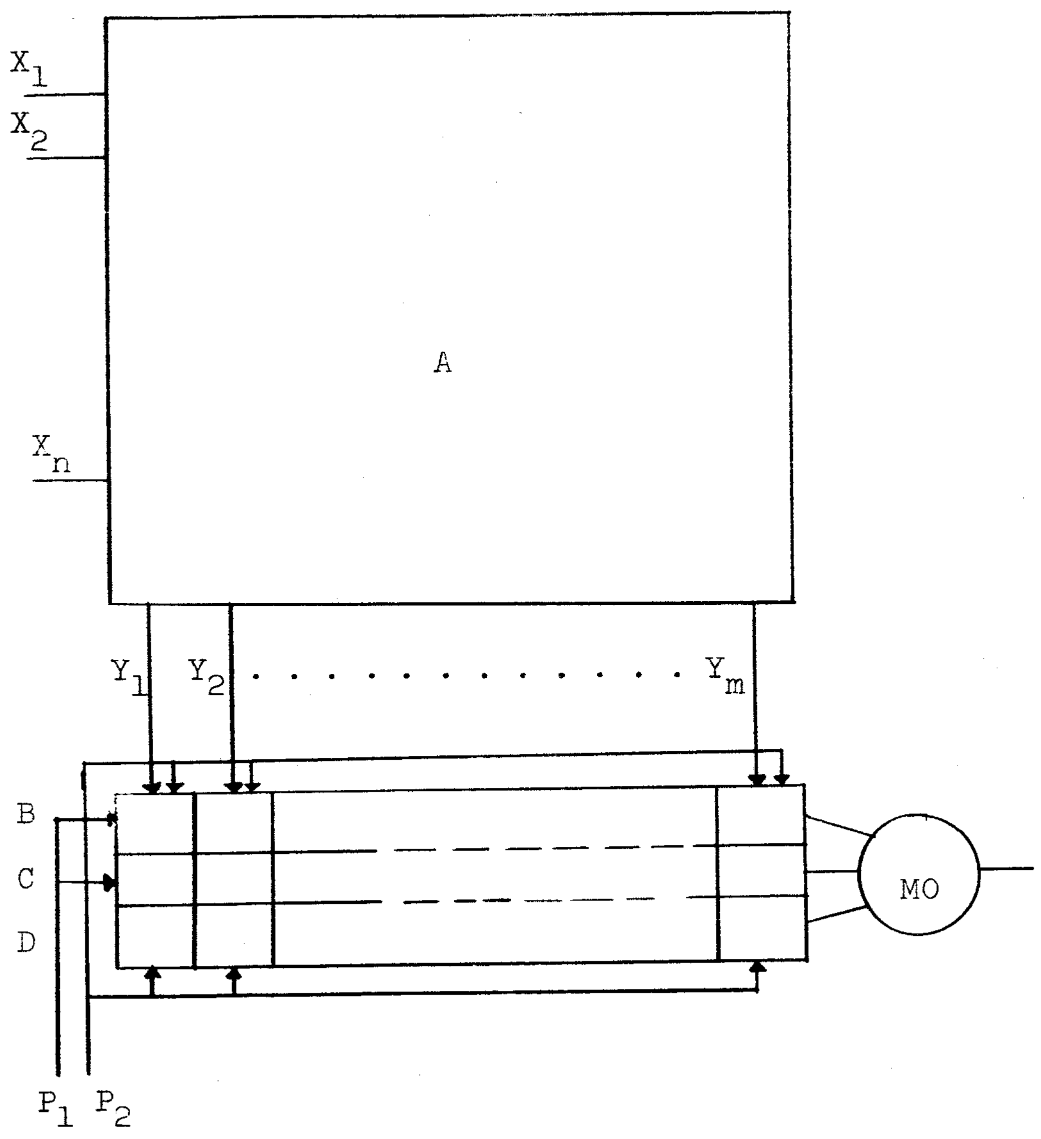


Figure 5 INFORMATION REDUNDANCY (PARALLEL OUTPUT)

The operation is as follows. The first set of operations is performed on the input data and the output is stored in the register B. Next, the same set of operations if performed on the same data and the results are entered in register B, whose contents now are shifted to C. Now the C register contains the information processed first and B contains the information processed next. Next the sequence is repeated for the second time, with the result that, the registers D, B and C now contain "redundant information". If the operations were not affected by noise the results in the registers should agree. The shifting operation in the downward ($\downarrow$) direction is performed by the shift pulse p_1.

Now another sequence of shift pulses p_2 is applied so that the bits are shifted along each register into the majority organ, (M. O.). The majority organ decides the output, which is the same as the majority of the inputs. Thus, for instance, if

D contains 1 0 0 1 1 1 0

B contains 1 0 0 1 1 1 0

C contains 0 0 1 1 1 1 1 , the output of the majority organ would be 1 0 0 1 1 1 0

It is not easy to calculate the reliability of this scheme. If there is a permanent failure, all the three redundant units of information would be wrong. The principle of the scheme is that when the system is subjected to noise, the probability that noise affects the system during two repeated operations is small. If each set of operations takes time "t" and if the probability that noise affects the system during the interval t is p, then the probability that noise affects the system during the successive intervals is p^2, assuming, of course, that noise occurrences are independent. The above tells us that by repeated operation, the probability of noise affecting the results is reduced.

It has been assumed that the registers and the majority organ function properly.

The obvious cost to be paid for this protection against noise interference is increased time. T, time for a total set of operations, is given as follows:

$$T = 3t + mt_p$$

where "t" is the time for a non-redundant set of operations, "t_p" shifting time for each bit and "m" the number of bits to be shifted. There is also an increase in weight and cost, introduced by the registers and the majority organ.

Let "cost" be taken as a factor which includes price as well as weight; then if -W_p- is the cost of a flipflop, W_t is the cost of the logic system and -W_m- is the cost of the majority organ, the cost of the redundant system may be given as

$$W^1 = W_t + 3mW_p + W_m$$

If the logic system under consideration has a serial output instead of parallel output, much less material would be required, but a longer time would be taken. Such a system with serial output is illustrated in figure 6. Three successive bits coming out of the system A correspond to the same information bit and hence they would all agree if there is no error. The bits stored in the flipflops B, C, and D are restored by the Majority Organ. Since the number of registers has been reduced, this is more reliable and less costly than the system with parallel output. But time for a cycle of operations is considerably increased.

Summing up, when the system is subjected to noise interference, hardware redundancy may not be necessary to prevent errors. In such a case, information redundancy would provide protection against system malfunction. The price paid for this is considerably increased time of processing data. There is more than a three-fold increase in time for the procedure outlined above.

When the object is to protect the system against both permanent circuit failure and intermittent noise, the natural solution would be to provide both hardware and information redundancies.

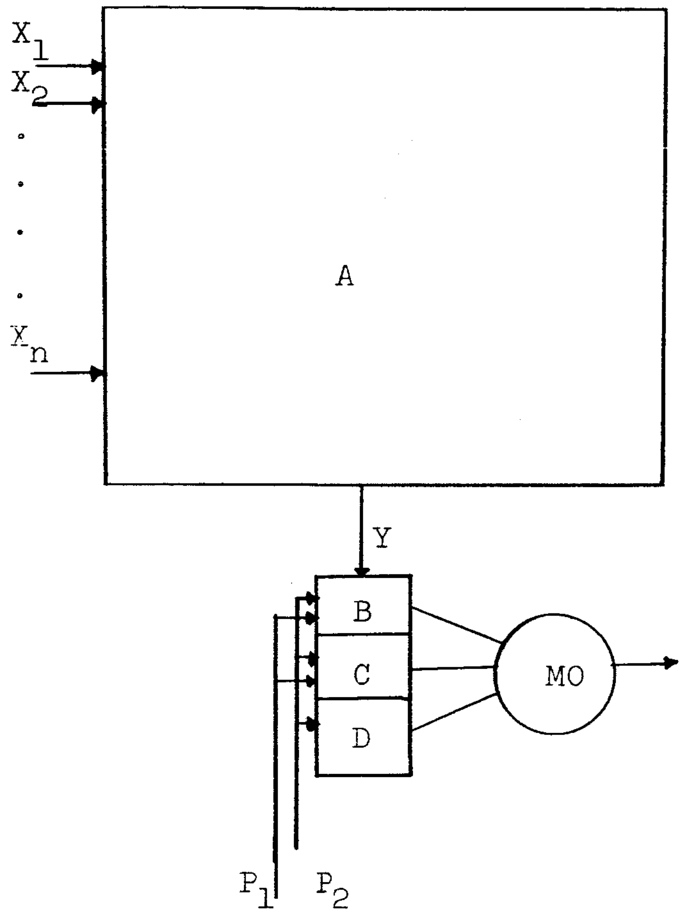


Fig. 6 INFORMATION REDUNDANCY (SERIAL OUTPUT)

Hardware and Information Redundancy

Let us now outline a scheme that includes both hardware and information redundancies. Such a scheme is illustrated in Fig. 7, which is just a triplicate version of the system described in the previous section. A, B and C are identical units. A set of flipflops, a_{11}, $a_{12}\ldots a_{33}$ constitutes three registers which store the output bits from the redundant units A, B and C. As described earlier, the operations are performed on the same set of data during successive bit times t_1, t_2 and t_3. The majority organs M_1, M_2 and M_3 decide on the majority of the output bits stored in the registers. The majority organ M_4 decides on the majority of the outputs M_1, M_2 and M_3. If the noise interference is assumed to be independent, it is intuitively clear that by performing redundant operations it is possible to reduce the probability of error due to noise.

Reliability Analysis

A quantitative analysis of the probability of successful operation of the above system, when noise affects it, is difficult. We shall make some assumptions that would considerably simplify the analysis.

Let the probability that noise affects the system during any one of the redundant operations be p. In other words, p is the probability that during any one of the redundant

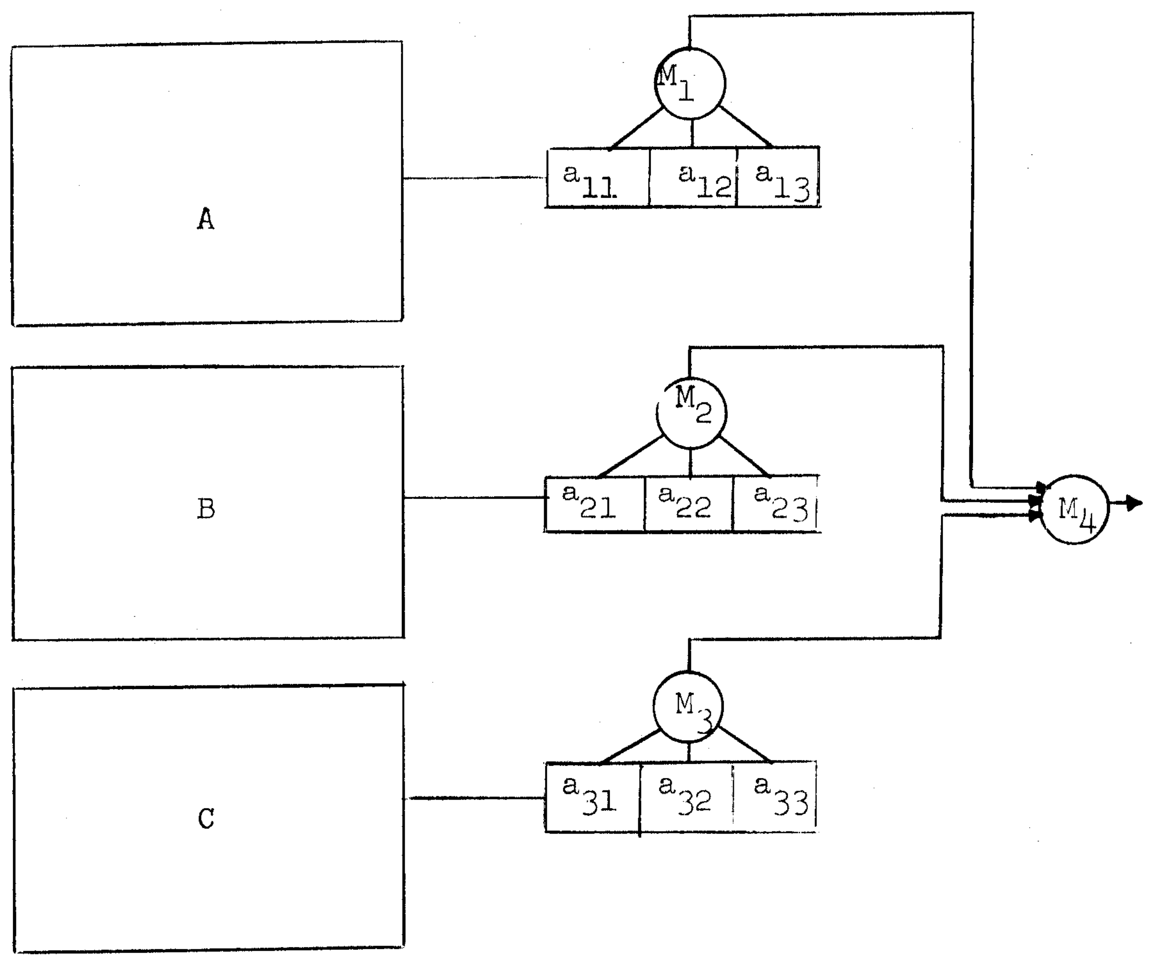
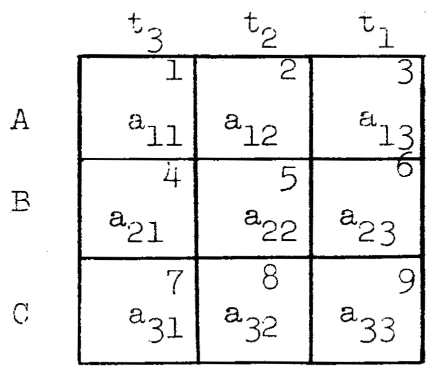


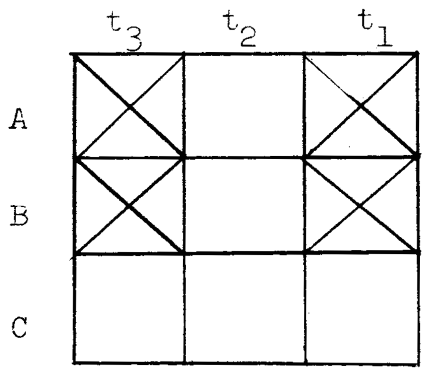
Fig. 7 COMBINATION OF HARDWARE AND INFORMATION REDUNDANCIES SCHEME 1

operations the output bit is wrong because of noise interference. The assumption we shall make is that the above events are independent. Then the probability that noise affects the system during t_1 and t_2 (during successive operations) is p^2 and the probability that noise affects the system during t_1, t_2 and t_3 is p^3. Let us further assume that the restoring units (the registers and the majority organs) are one hundred percent reliable.

Consider the map 1 shown below. Let each square correspond to a flipflop a_{ij} of the registers shown in figure 7. Then there is also a correspondence between the squares of the map and the output bits from the units A, B and C during t_1, t_2 and t_3. Thus the square 4 corresponds to the flipflop a_{21} as well as the output bit from unit B during t_3. Let us, for the convenience of analysis, indicate a wrong bit (the bit affected by noise) by crossing the corresponding square. Thus map 2 indicates that the output bits from units A and B during t_1 and t_3 are wrong.

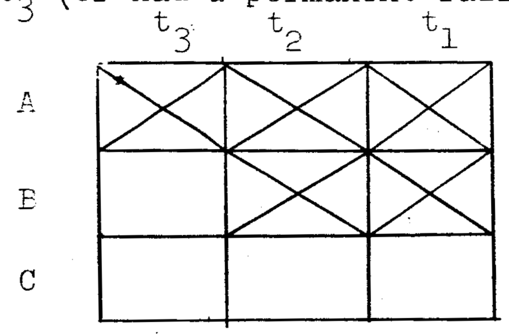


Map 1



Map 2

There are a total of $\sum_{i=0}^{9} \binom{9}{i}$ = 512 possible combinations of outcomes from units A, B and C during t_1, t_2 and t_3. The event $\binom{9}{0} = 1$ corresponds to the case of no failures. Out of the possible 511 possible combinations of failures $\left[\sum_{i=1}^{9} \binom{9}{i} = 511\right]$, the system described guards against 255 combinations. The principle used in arriving at the latter figure is that for the scheme to prevent system failures, no two registers in figure 7 should have two or more wrong bits. This is easily seen. Also, referring to map 1, no two rows should have two or more squares crossed. As an illustration, consider map 3. It tells us that unit A failed during t_1, t_2 and t_3 (or had a permanent failure) and unit B failed



Map 3

during t_1 and t_2. This means that the majority organs M_1 and M_2 (Refer to figure 7) would yield wrong results and M_3 would yield the correct result. Hence the output of M_4, the system output, would be wrong.

Table 1 lists the probabilities of the occurrence of the events shown in column 2 of the same table. For example, the probability that a register contains wrong information bits in the second and third units (corresponding to t_2 and t_1) is $r (1 - p) p^2$, where r is the reliability of the

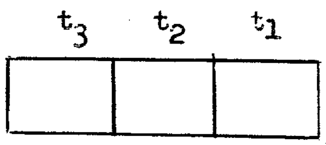
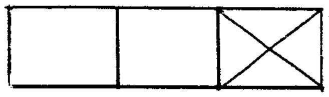
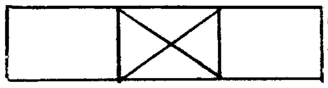
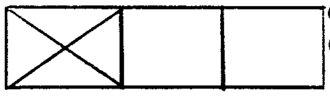
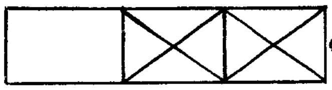
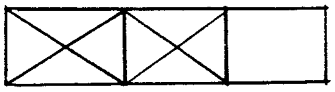
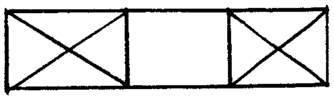
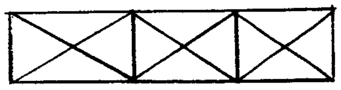
	EVENTS (Errors in bits stored in each register)	ASSOCIATED PROBABILITY (r- reliability of non-redundant system p- probability that noise affects the system during t_1, t_2, or t_3.)
1	 no error	$r(1-p)^3$
2	 error during t_1	$r(1-p)^2p$
3	 error during t_2	$r(1-p)^2p$
4	 error during t_3	$r(1-p)^2p$
5	 error during t_1 and t_2	$r(1-p)\ p^2$
6	 error during t_2 and t_3	$r(1-p)\ p^2$
7	 error during t_1 and t_3	$r(1-p)\ p^2$
8	 error during t_1 and t_2, t_3	$rp^3 + 1-r$ (1-r corresponds to the probability to permanent failure)

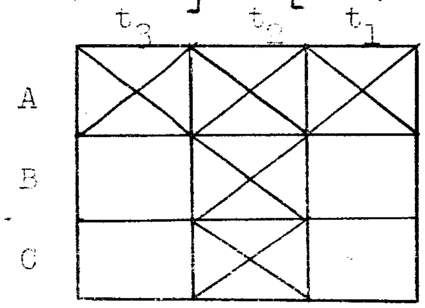
TABLE 1: ERRORS AND ASSOCIATED PROBABILITIES

system.[11a]

The events listed in the table are exclusive and constitute a universe (i. e., the total probability of the 8 events listed is 1). Let us note that the event with 3 errors implies that either that system (A, B or C) has been affected by noise at t_1, t_2 and t_3 or a permanent failure has occurred.

Using table 1, we can estimate the probability of any of the 512 events that could occur, when we take into account all the three registers. Thus the probability of the event shown below is

$$\left[rp^3 + (1 - r)\right] \times \left[rp\,(1 - p)^2\right]^2$$



Incidentally, the above event does not result in system failure although five out of nine bits are wrong!

Reliability of the scheme described would be the probability that no two registers contain two or more wrong units of information. There are 256 such events. The probabilities associated with these events are the following:

11a Reliability, r, of the system takes into account permanent failures.

1) The probability that all 9 bits are error free:

$P_1 = [r\ (1 - p)^3]^3$. There is one such event.

2) The probability that any one of 9 bits is wrong:

$P_2 = 9\ [rp\ (1 - p)^2\ xr\ (1 - p)^3\ xr\ (1 - p)^3]$

$= 9\ [r^3\ (1 - p)^8\ p]$. Note that $\binom{9}{1} = 9$

3) The probability that any two bits are wrong

$P_3 = 36r^3\ (1 - p)^7 p^2$, noting that $\binom{9}{2} = 36$.

4) The probability that any three bits are wrong

P_4 = The probability that all three wrong bits are in the same register + the probability that the three wrong bits are distributed

$= 3r^2\ (1 - p)^6\ (1 - r + rp^3) + 81r^3\ (1 - p)^6 p^3$; note that $\binom{9}{3} = 84$

5) The probability that there are 4 errors which are swamped

P_5 = The probability that there are 3 in one and 1 in another + the probability that there are 2 in one and 1 in each of the remaining units.

$= 18\ (1 - r + rp^3)\ \ (1 - p)^5\ pr^2 + 81\ [r^3\ p^4\ (1 - p)^5]$

The coefficients 18 and 81 correspond to the number of the events.

6) The probability that there are 5 errors which are swamped

P_6 = the probability that there are 3 in one and one in each of the remaining two units.

$= 27\ (1 - r + rp^3)\ \ (r^2\ p^2\ (1 - p)^4)$; note that there are 27 such events.

The total probability that the redundant system we have described operates correctly is $\sum_{i=1}^{6} P_i$.

The above probability was calculated for several values of r and p and the results are shown in table 2. We should note here that the foregoing analysis holds only under the assumption that noise interference is independent.

Another way of implementing combined redundancy is shown in figure 8. The difference between this system and the system just analyzed is clear from the corresponding figures. A study of the second system would show us that this again swamps 256 out of the possible 512 combinations of errors in the bits stored in the registers. However the errors swamped are not all the same as the ones swamped by the first system. An analysis of this scheme is identical to the analysis of the scheme first described and hence will not repeated.

We should mention here that there may be some compensating failures in the system, including the registers and the majority organs. For instance, in the scheme shown in figure 7, if the majority organ M_1 and the unit A fail, the output from M_1 would still be correct! This holds, of course, only under the assumption that failures are "worst case". We can easily show that there are several such instances of compensating errors. But we shall not make a detailed analysis of the compensating errors here.

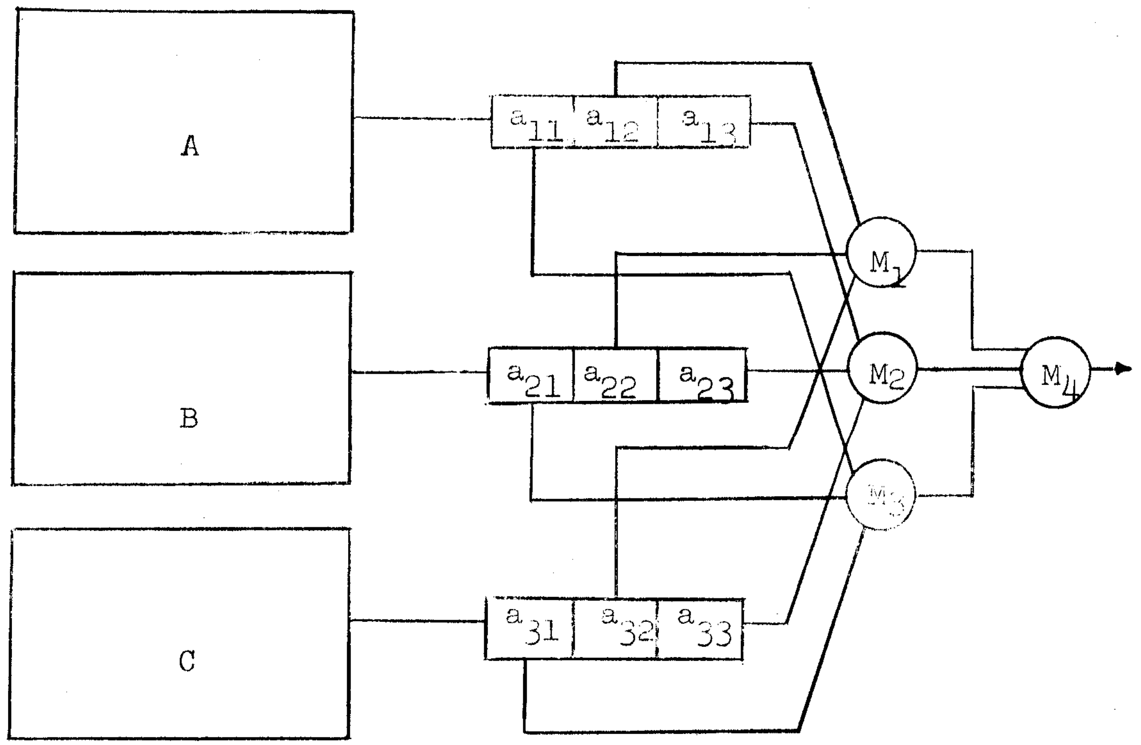


FIGURE 8: COMBINATION OF HARDWARE AND INFORMATION REDUNDANCIES

SCHEME 2

r non redundant system reliability	p probability that noise affects the system output at t_1, t_2, or t_3	P probability that redundant system output is correct	r(1-p) probability that non redundant system output is correct
.6000	.1000	.6236	.5400
	.2000	.5563	.4800
	.3000	.4557	.4270
	.4000	.3360	.3600
	.5000	.2160	.3000
.7000	.1000	.7589	.6300
	.2000	.6867	.5600
	.3000	.5730	.4900
	.4000	.4306	.4200
	.5000	.2817	.3500
.8000	.1000	.8736	.7200
	.2000	.8048	.6400
	.3000	.6867	.5600
	.4000	.5276	.4800
	.5000	.3520	.4000
.8500	.1000	.9199	.7650
	.2000	.8566	.6800
	.3000	.7404	.5950
	.4000	.5760	.5100
	.5000	.3883	.4250
.9000	.1000	.9569	.8100
	.2000	.9021	.7200
	.3000	.7910	.6300
	.4000	.6235	.5400
	.5000	.4254	.4500
.9500	.1000	.9834	.8500
	.2000	.9402	.7600
	.3000	.8379	.6650
	.4000	.6703	.5700
	.5000	.4625	.4750

TABLE 2: ESTIMATES OF RELIABILITY USING HARDWARE AND INFORMATION REDUNDANCIES

CHAPTER IV

REDUNDANCY TECHNIQUES IN THREE LEVEL LOGIC CIRCUITS

Redundancy techniques to improve the reliability of two level logic circuits have been proposed and studied by several authors. In this chapter, we shall extend some of the concepts to three level logic circuits.

Three-valued Logic

Almost all of the present day digital circuits use two valued logic. A signal may be "On" or "Off", that is, the truth value is 1 or 0. There is, however, no theoretical limitation to the value of the logic. In three valued logic, a variable may assume one of three values: 0, 1 or 2.

Several people have explored the possibility of designing digital systems using three level (ternary) logic circuits. The advantage of three level logic is that more information could be represented using lesser hardware. Thus the size of the machine could be reduced. In addition to this, some of the peculiarities of three level logic could be favorably exploited.

Majority Technique

Let us extend the concept of triple redundancy to improve the reliability of three level logic circuits.

The redundant system representation is the same as that for two level logic. Hence we could refer to figure 1, page 7. We recall that A, B and C are redundant identical systems. The outputs X_1, X_2 and X_3 may now have nay one of the values 0, 1 or 2. Correct operation implies that all the three be the same. In case of failure, they may be different.

There are 27 possible combinations of X_1, X_2, X_3.

X_1-	0 1 2	0 1 2	0 1 2-	0 1 2	0 1 2	0 1 2-	0 1 2	0 1 2	0 1 2
X_2-	0 0 0	1 1 1	2 2 2-	0 0 0	1 1 1	2 2 2-	0 0 0	1 1 1	2 2 2
X_3-	0 0 0	0 0 0	0 0 0-	1 1 1	1 1 1	1 1 1-	2 2 2	2 2 2	2 2 2

When the three systems function correctly $X_1 = X_2 = X_3 = 0$, 1 or 2.

The Majority Organ may be designed to produce an output which is the same as the majority of the inputs, as in the case of two valued logic. However, there is a small problem. There are 6 combination of X_1, X_2, X_3 out of the possible 27 in which all the three differ from one another. They are,

X_1	0 0	1 1	2 2
X_2	1 2	0 2	0 1
X_3	2 1	2 0	1 0

How does the Majority Organ decide?

A closer study shows that the above apparently ambiguous situation comes under the case when two of the three redundant systems fail! Since, when all the three differ, no two can be correct. Further, triple redundancy is not expected to give protection against system failure, when two subsystems fail. In other words, when two subsystems fail, system malfunctions as in the case of two level logic. Hence it does not really matter what the output is, when three input signals to the Majority Organ are different.

The probability of successful operation or the reliability of the system may be given by the expression

$$R = r\,(p^3 + 3p^2\,(1-p)) = r\,(p^2\,(3-2p))$$

where r = reliability of the Majority Organ and

p = reliability of the system.

It is appropriate for us to clarify what we mean by failure in three level logic. We shall say that a system has failed if the actual output does not correspond to the correct output--whether the former differs from the latter by one or two levels makes no difference.

Synthesis of the Majority Organ

Before we attempt to synthesize the Majority Organ let us consider some of the characteristics of the algebra of n valued logic. A "cycle" operation in n valued logic is defined as follows: If $A = t_k$ (that is, the truth value of A is t_k), cycle A = A = (t_k + 1) mod n. The cycle operation for three valued logic is as follows:

A	0	1	2
$\bar{A}$	1	2	0
$\bar{\bar{A}}$	2	0	1

A is obtained by cycling A.

Post[12] and Webb[13] have defined functionally complete operations for n valued logic. An operation is said to be functionally complete when any function of the variables could be expressed in terms of the defined operation, which is called the "Primitive Operation". An example of such an operation in two level logic is "NAND".

Webb[14] has defined the functionally complete operation "/" for n valued logic. If A = t_k, B = t_j, then A/B = (max t_k, t_{j+1}) mod n. This operation in three valued logic is shown below:

---

[12] E. L. Post, "Introduction to a General Theory of Propositions", Amer. J. Math., 43 (1921), pp. 163-185.

[13] D. L. Webb, "Generation of Any N Valued Logic by One Binary Operation", Proc. Nat. Acad. U. S. A. 21 (1935), pp. 252-254.

[14] D. L. Webb, "Definition of Post's Generalized Negative and Maximum in terms of One Binary Operation", Amer. J. Math., 58 (1936), pp. 193-194.

	A	0 1 2	0 1 2	0 1 2
	B	0 0 0	1 1 1	2 2 2
A/B=A+B	=	1 2 0	2 2 0	0 0 0

In addition to the above function (/), "AND" and "cycle" form a pair of functionally complete functions. The "OR" and "cycle" form another pair. In three valued logic, the "AND" and "OR" operations are defined as follows:

	A	0 1 2	0 1 2	0 1 2
	B	0 0 0	1 1 1	2 2 2
A . B (min. value)	=	0 0 0	0 1 1	0 1 2
A + B (max. value)	=	0 1 2	1 1 2	2 2 2

The simplest method of implementing the majority organ is to use "AND" and "OR"operations. Circuits to perform theseoperations in three level logic have been designed and tested.[15]

The output $Y = X_1X_2 + X_2X_3 + X_3X_4$.

Figure 9 shows how the majority organ is implemented using the above operations.

An attempt was made to synthesize the majority organ

[15] N. F. J. Matthews, "An Algebra of Three Valued Logic and Its Application to Digital Circuitry", George Washington University Thesis, 1959.

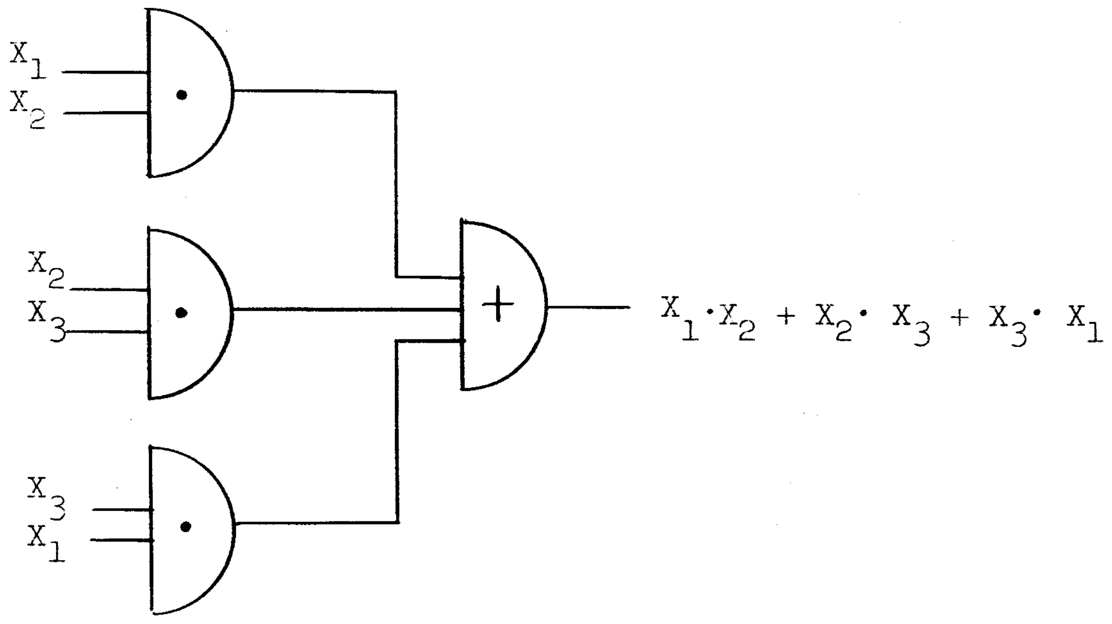


Figure 9 THE MAJORITY ORGAN

using the functionally complete operation "/" defined by Webb. The circuit for this has been designed and tested.[16]

The result of the study showed that the majority organ so synthesized is much too complex and does not satisfy the basic necessity of redundancy to imporve reliability, namely simplicity. The standard basis and the designation number of the desired output, together with the "Don't-Care" terms (shown by X) are shown below.

X_1--0 1 2 0 1 2 0 1 2--0 1 2 0 1 2 0 1 2--0 1 2 0 1 2 0 1 2

X_2--0 0 0 1 1 1 2 2 2--0 0 0 1 1 1 2 2 2--0 0 0 1 1 1 2 2 2

X_3--0 0 0 0 0 0 0 0 0--1 1 1 1 1 1 1 1 1--2 2 2 2 2 2 2 2 2

Majority Organ output

= 0 0 0 0 1 X 0 X 2--0 1 X 1 1 1 X 1 2--0 X 2 X 1 2 2 2 2

$X_1X_2 + X_2X_3 + X_3X_1$

= 0 0 0 0 1 1 0 1 2--0 1 1 1 1 1 1 1 2--0 1 2 0 1 2 2 2 2

It is seen that there are six "Don't Care" terms each corresponding to the input combination of three different values

[16]David Hardy Nelson, "An adder-Subtracter Using Three Valued Logic", George Washington University Thesis, 1962

The function synthesized using the operation is

$$F = \overline{\overline{\overline{\overline{\overline{X_3}}+1} + \overline{\overline{\overline{\overline{X_2}}+1}+\overline{\overline{X_1}} + \overline{X_2+1}+\overline{\overline{X_1}}}} + \overline{X_3+1}+\overline{X_1+X_2} + \overline{\overline{\overline{X_3}+1} + \overline{\overline{\overline{\overline{\overline{X_2}}+1}+\overline{\overline{X_1}}} + \overline{\overline{X_2}+1} + \overline{\overline{X_2+1}+\overline{\overline{X_1}}}}}}$$

The majority organ so obtained is shown in figure 10. It is readily seen that the majority organ so obtained is complex compared to the one implemented using "AND", "OR", operations. Hence the latter is preferred.

Extension from $\eta = 3$ to $2n + 1$ redundancy in three level logic is straight forward. The fact that the outputs of the redundant units may assume any one of three values does not pose any special problem. As in two valued logic, the redundant system yields correct output when $n + 1$ of the $2n + 1$ circuits function properly.

Let us consider the case $\eta = 5$, that is, $n = 2$. <u>At least</u> three units and the majority organ should function correctly for success. If the redundant outputs are X_1, X_2, X_3, X_4, X_5, the "Majority Function", would be $X_1 \cdot X_2 \cdot X_3 + X_2 \cdot X_3 \cdot X_4 + X_3 \cdot X_4 \cdot X_5 + X_4 \cdot X_5 \cdot X_1 + X_5 \cdot X_1 \cdot X_2 + X_2 \cdot X_3 \cdot X_5 + X_2 \cdot X_4 \cdot X_5 + X_3 \cdot X_5 \cdot X_1 + X_3 \cdot X_4 \cdot X_1 + X_1 \cdot X_2 \cdot X_4$. We have already defined the "AND"and "OR"

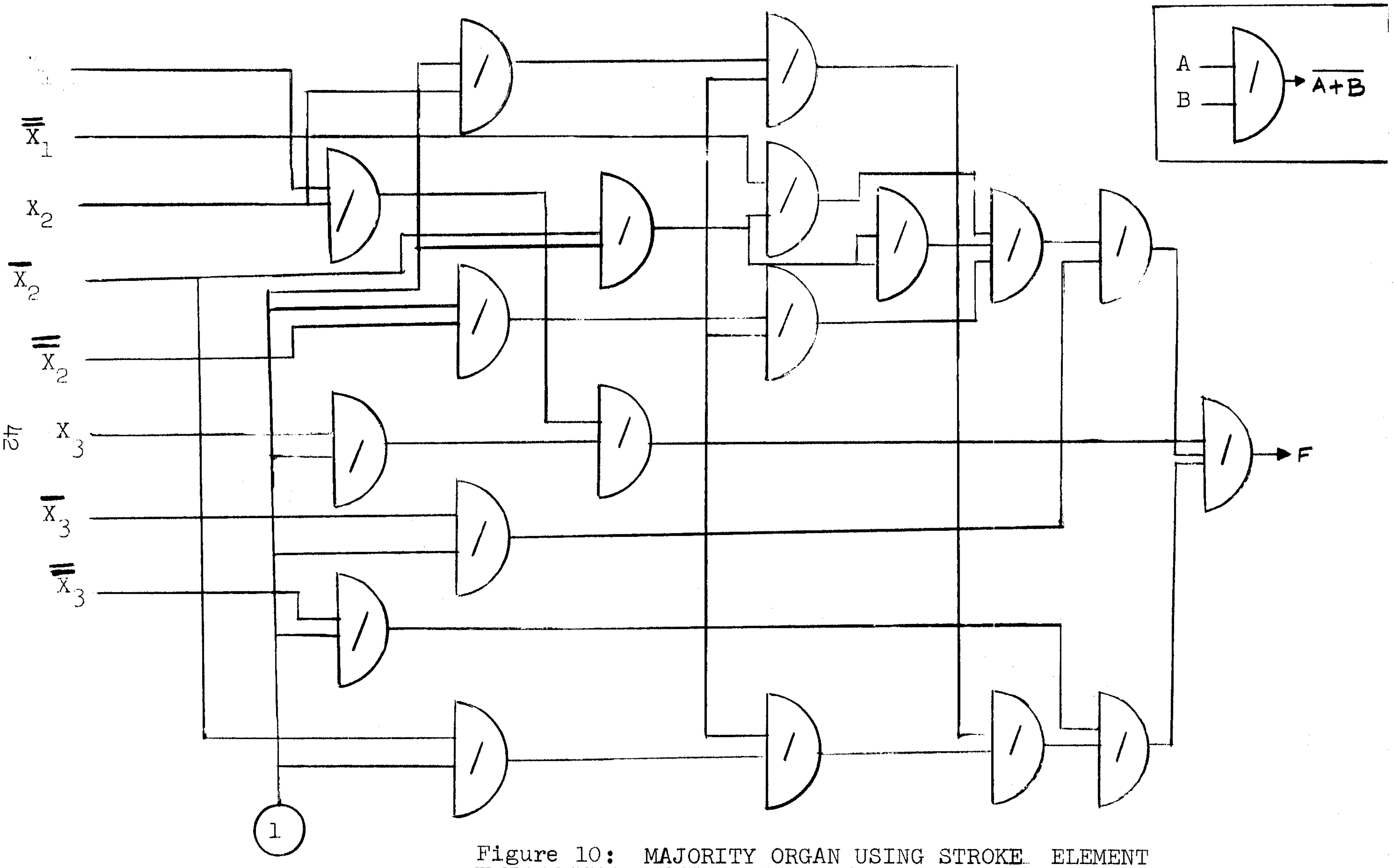


Figure 10: MAJORITY ORGAN USING STROKE ELEMENT IN THREE LEVEL LOGIC

operations in three level logic.

An interesting situation arises when $X_1 = 0$, $X_2 = 0$, $X_3 = 1$, $X_4 = 1$, and $X_5 = 2$. The implication is that at least three have malfunctioned; since, if three function correctly we should have three identical outputs. In the above situation the output would be 1, which may or may not be correct.

Dual Redundancy

We will now extend the concept of dual redundancy to three valued logic. Referring to figure 4 (page 13), we recall that A and B are two identical units and C is an additional unit. During normal operations, A and B give the same output. In case of failure of A or B or both, the outputs will not be identical. As in two valued logic we can design the restoring unit to have two outputs, Y_1 and Y_2. We may use the existence of three levels to give us additional information about the failure. We can design Y_2 to attain logical 1, when the outputs of A and B differ by one level and logical 2, when the outputs differ by two levels. The "restored" output Y_1 may be synthesized to yield the minimum value of X_1, X_2 the outputs of the identical units. This could have well been the maximum value of X_1, X_2. Actually a knowledge of the circuit used will be helpful in making the above decision.

Y_2 would then be synthesized as follows:

X_1	0 1 2	0 1 2	0 1 2
X_2	0 0 0	1 1 1	2 2 2
Y_2	0 1 2	1 0 1	2 1 0

Using "AND", "OR" and "CYCLE" elements, we have

$$Y_2 = \overline{(\overline{\overline{X}}_2+X_2)}\cdot X_1+\overline{(\overline{\overline{X}}_2+X_2)}\cdot\overline{\overline{X}}_1\cdot(X_1+\overline{X}_1)+\overline{(\overline{\overline{X}}_2+\overline{X}_2)}\cdot(\overline{(\overline{X}_1+X_1)}+X_1\cdot\overline{X}_1).$$

Dual redundancy is thus effective in detecting errors (when one of the two units fails) in three level systems as well.

CHAPTER V

SUMMARY

The following work has been done.

1. Redundancy techniques to improve the reliability of two level logic circuits were investigated. The majority principle was outlined. In the Appendix B, a quantitative study of this is made and a pair of terms are defined. A computer program has been written to determine the redundant configuration compatible with system requirements.

2. The concept of dual redundancy was considered and it was shown how this could be used to enhance the reliability of logic circuits.

3. The concept of "Information Reliability" was introduced. A technique for protecting systems against noise interference was presented.

4. The majority technique used in two level logic was extended to three level logic circuits.

5. A brief study of quadded logic and triplet scheme is made in Appendix C.

CHAPTER VI

CONCLUSIONS

From the current study, the following conclusions have been reached.

Hardware redundancy is very effective when permanent failures occur; it may not be as effective where there is a noise interference. When there is temporary circuit failure, hardware redundancy may not be really necessary--a simple repetition of information (information or operation redundancy) may be sufficient. Where there is a noise interference, it is quite likely that all redundant systems are affected. Once again information redundancy would be better suited. The trade off is between material (price, weight, etc.) and time.

On the other hand, when there is a permanent circuit failure, information redundancy is useless. When errors are committed, they are repeated. In such a case, hardware redundancy appears to be the only solution.

When the system is to be protected against permanent failures and temporary failures as well as noise, both hardware and information redundancies may be used. Increased cost, time and weight are the price to be paid.

APPENDIX A

Failure Model

In the discussions we have assumed "worst-case" failure, that is, the output of the failed system in two level logic is complementary to the correct output (0 for 1 and 1 for 0). A circuit failure may result in the output being a 0 or a 1 all the time. Hence the assumption that the failed circuit yields 0 for 1 and 1 for 0 is not correct. However, this model gives us the most pessimistic evaluation of the reliability of the system and hence desirable. It is readily seen that, in general, systems, redundant or not, would be more reliable than may be expected from an evaluation using the above model.

Failures have been assumed to be independent of each other. This is sufficiently true from the practical point of view.

APPENDIX B

In the following paragraphs we shall make a brief quantitative study of the redundancy technique using restoring organ (outlined in Chapter III).

The general layout of a system using output voting is shown in Figure 11. The system if broken into m subsystems. There are (η = 2n + 1, an odd number) identical units at each subsystem level. The majority of the redundant subsystems and the majority organ should operate correctly for the successful operation of the system.

The reliability of the system, as stated in Chapter III, is as follows:

$$R_S = \left[\sum_{i=n+1}^{\eta} \binom{\eta}{i} r^i (1-r)^{\eta-i} \right]^m r_{mo}^m$$

η = 2n + 1. The other variables have already been defined. (See Chapter III, Majority Organ).

Reliability of a subsystem, r, may be expressed by the usual exponential low, $r = e^{-\lambda t}$, where λ is the failure rate, in failures/unit time, of the subsystem. If each one of the subsystems is assumed to be of equal size and to consist of N components, and the failure rate of each of those components is λ_c failure/hour, then the failure rate of the subsystem may be given as $N\lambda_c$ fail./hr. . Let the failure rate of each of the components of the majority organ be λ_m failure/hour. The failure rate for the majority organ may then be given as $\left[\binom{\eta}{n+1}+1\right]\lambda_m$ failure/hour, $\binom{\eta}{n+1}$ + 1 being the number of components.

The reliability of the redundant system is now expressed in terms of failure rates as

$$R_s=\left[e^{-\lambda_m\left[\binom{\eta}{n+1}+1\right]t}\left[\sum_{i=n+1}^{\eta}\binom{\eta}{i}e^{-N\lambda_c it}\left(1-e^{-N\lambda_c t}\right)^{\eta-i}\right]\right]$$

On the following four pages various plots of interest are shown. They are self explanatory.

Although considerable increase in reliability can be achieved using the above technique, increased weight and cost may be limiting factors. For instance, weight is an important factor in space vehicles. Let us define a factor which would give us a fair idea of the increased weight or cost or both. Let α be the ratio of the "cost" of the majority organ per input to that of the original non-redundant system. The cost shall include weight and price. Let us, for simplicity, just consider the weight.[17] Then α is the ratio of

[17] Weight is the dominant factor in some important applications like space vehicles.

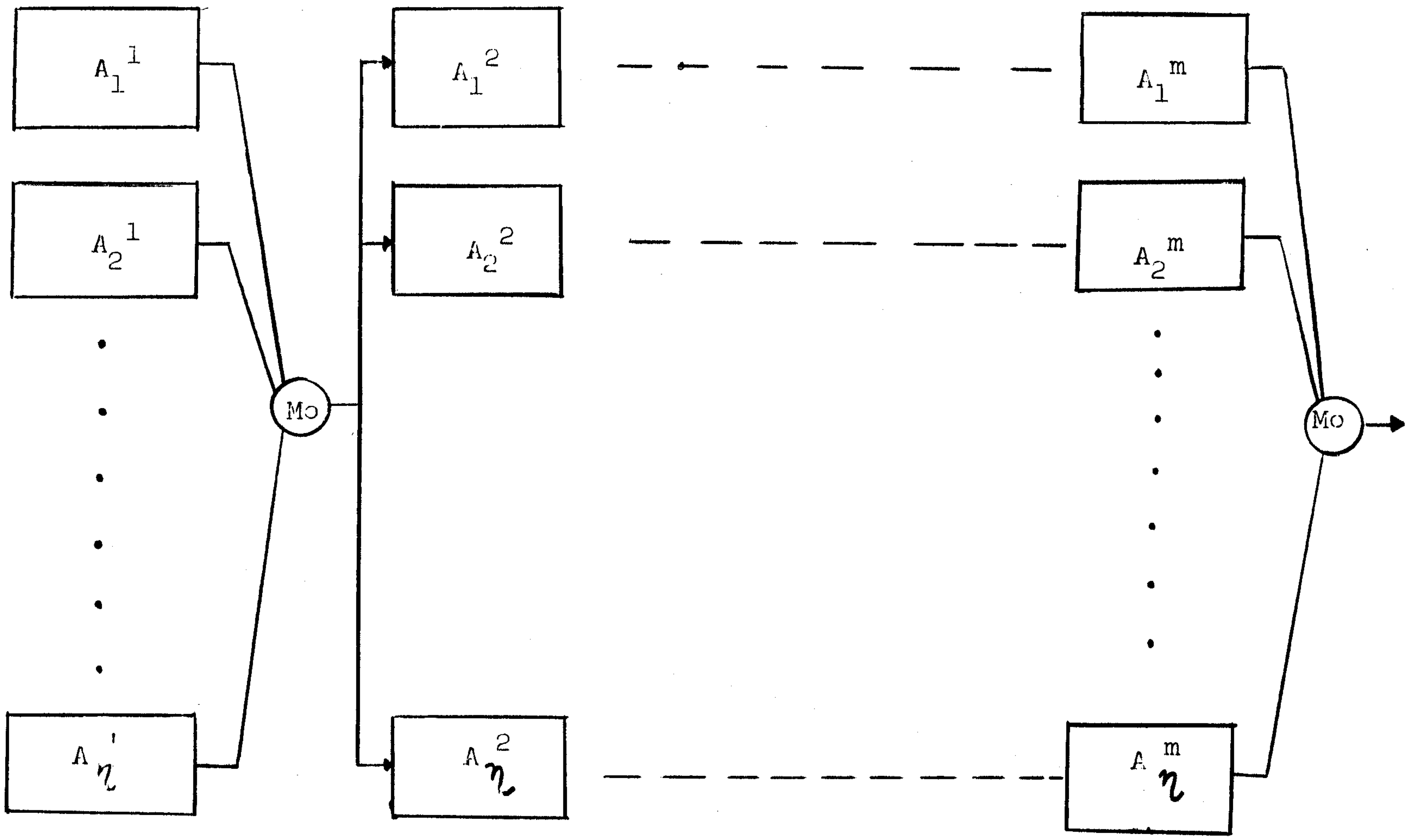


Figure 11: "η" REDUNDANCY WITH OUTPUT VOTING

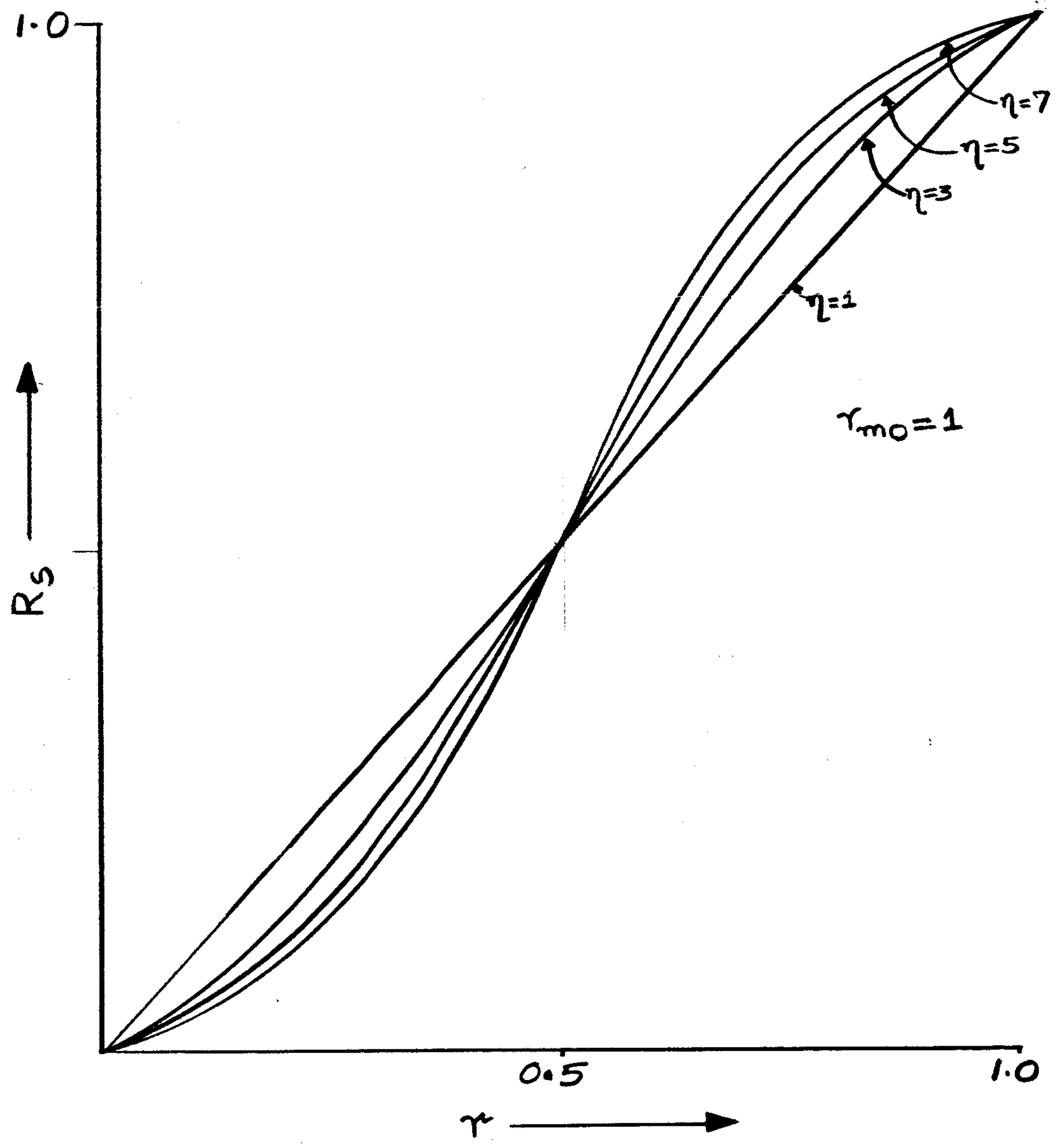


Figure 12 a RELIABILITY OF "η" REDUNDANT SYSTEM

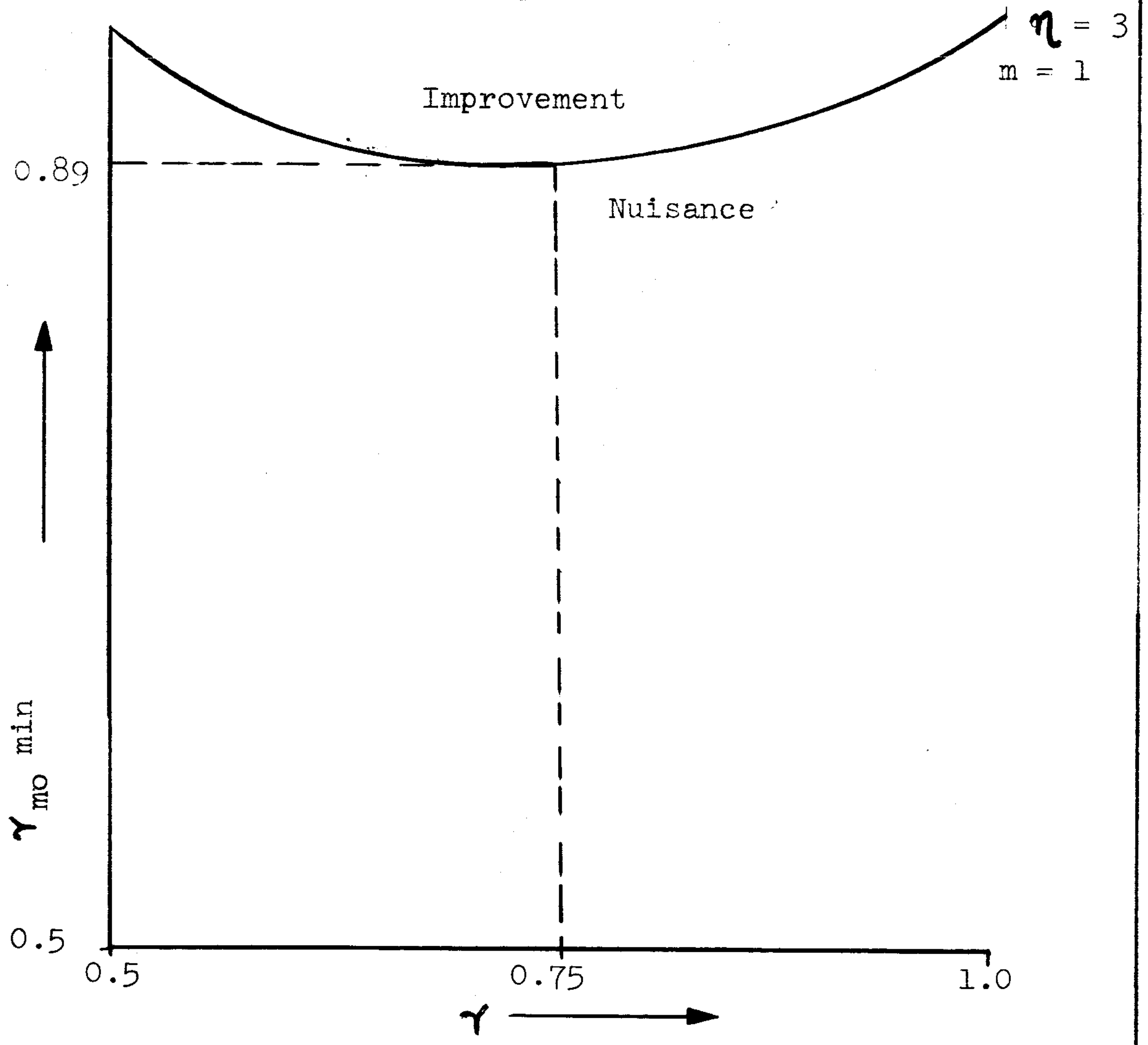


Figure 12 b MINIMUM MAJORITY ORGAN RELIABILITY FOR THE IMPROVEMENT IN RELIABILITY OF SYSTEM WITH TRIPLE REDUNDANCY

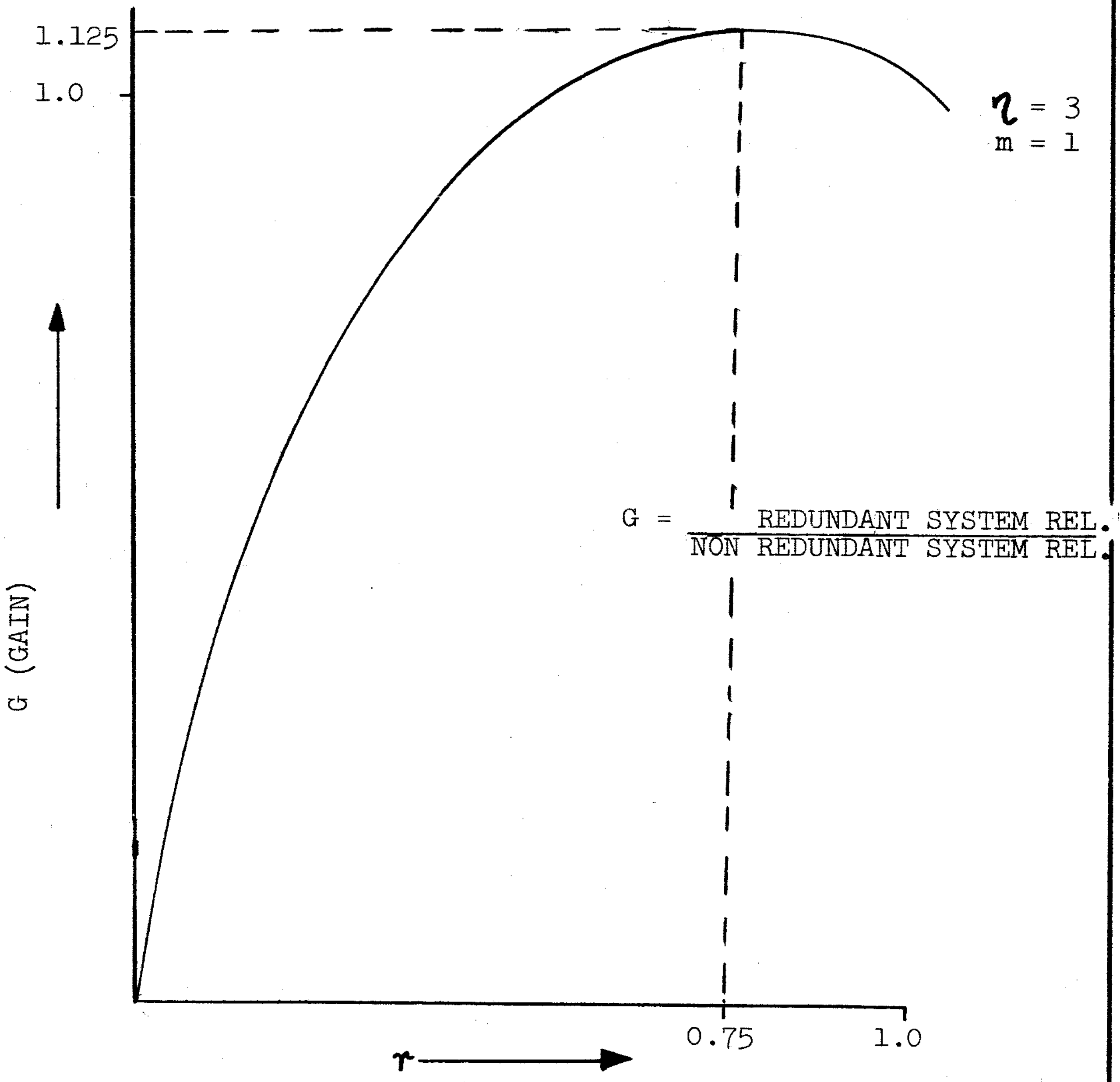


Figure 12 c GAIN IN RELIABILITY USING REDUNDANCY

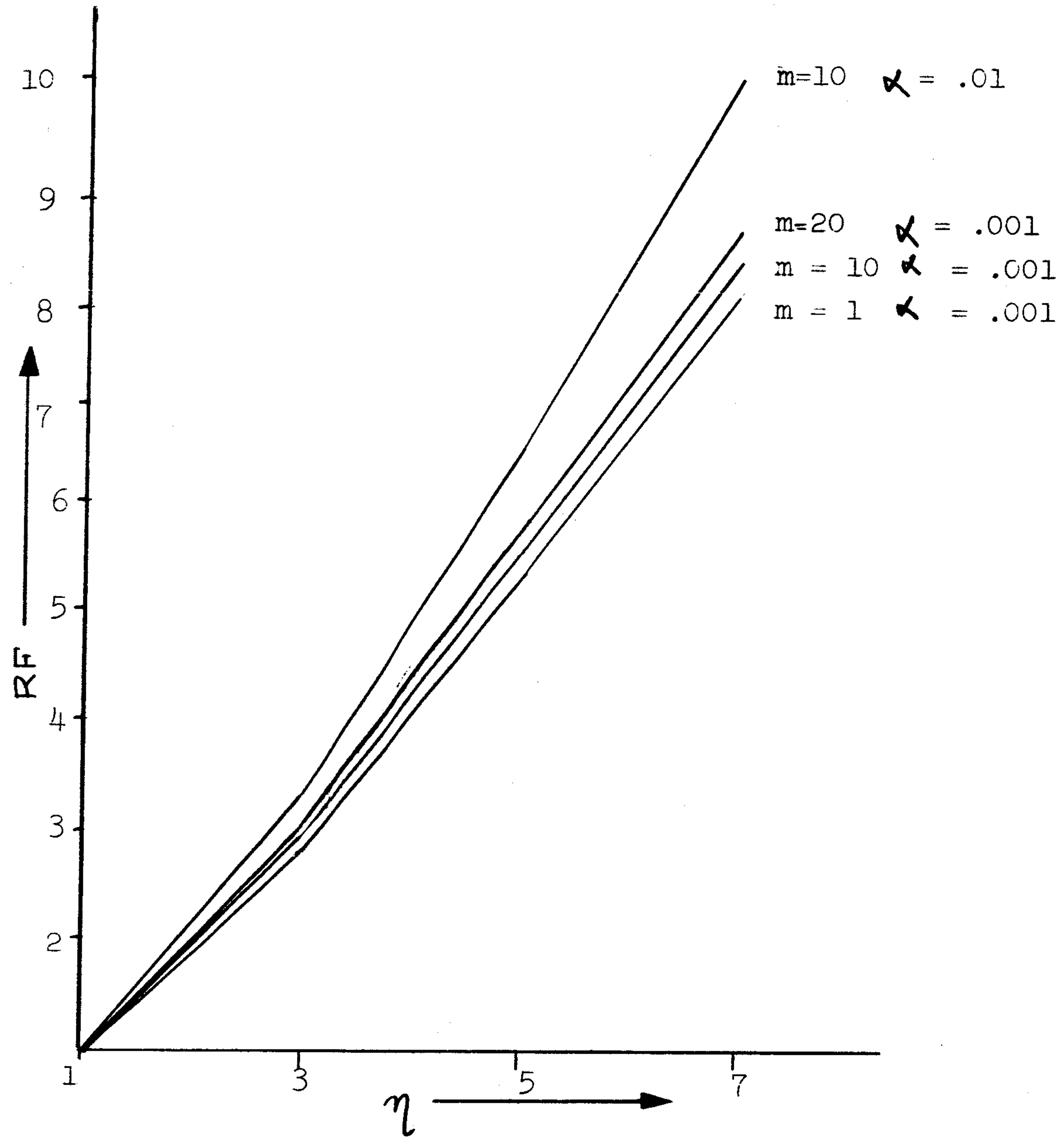


Figure 12 d REDUNDANCE FACTOR(R F) FOR DIFFERENT CONFIGURATIONS the weight of the majority organ (per input) to the weight of the non-redundant system. Let k be the wieght (a cost factor, in general) of the system.

Let us now define Redundance Factor as the ratio of the redundant system cost to the cost of the non-redundant system. The cost of the redundant system is $\eta k + \binom{\eta}{n+1} m \alpha k$ and the cost of the non-redundant system is k.

Redundance Factor, RF, $= \left[\eta k + \binom{\eta}{n+1} m \alpha k\right] / k = \eta + \binom{\eta}{n+1} m\alpha$

The limiting values of RF may be obtained as follows: As α approaches zero (the majority organ is of negligible cost compared to the non-redundant system), RF $= \eta$

As α approaches one (the cost of the majority organ is of the order of the non-redundant system), RF $= \eta + \binom{\eta}{n+1}$ m

For triple redundancy, the above limits reduce to 3 and 3 + 3m.

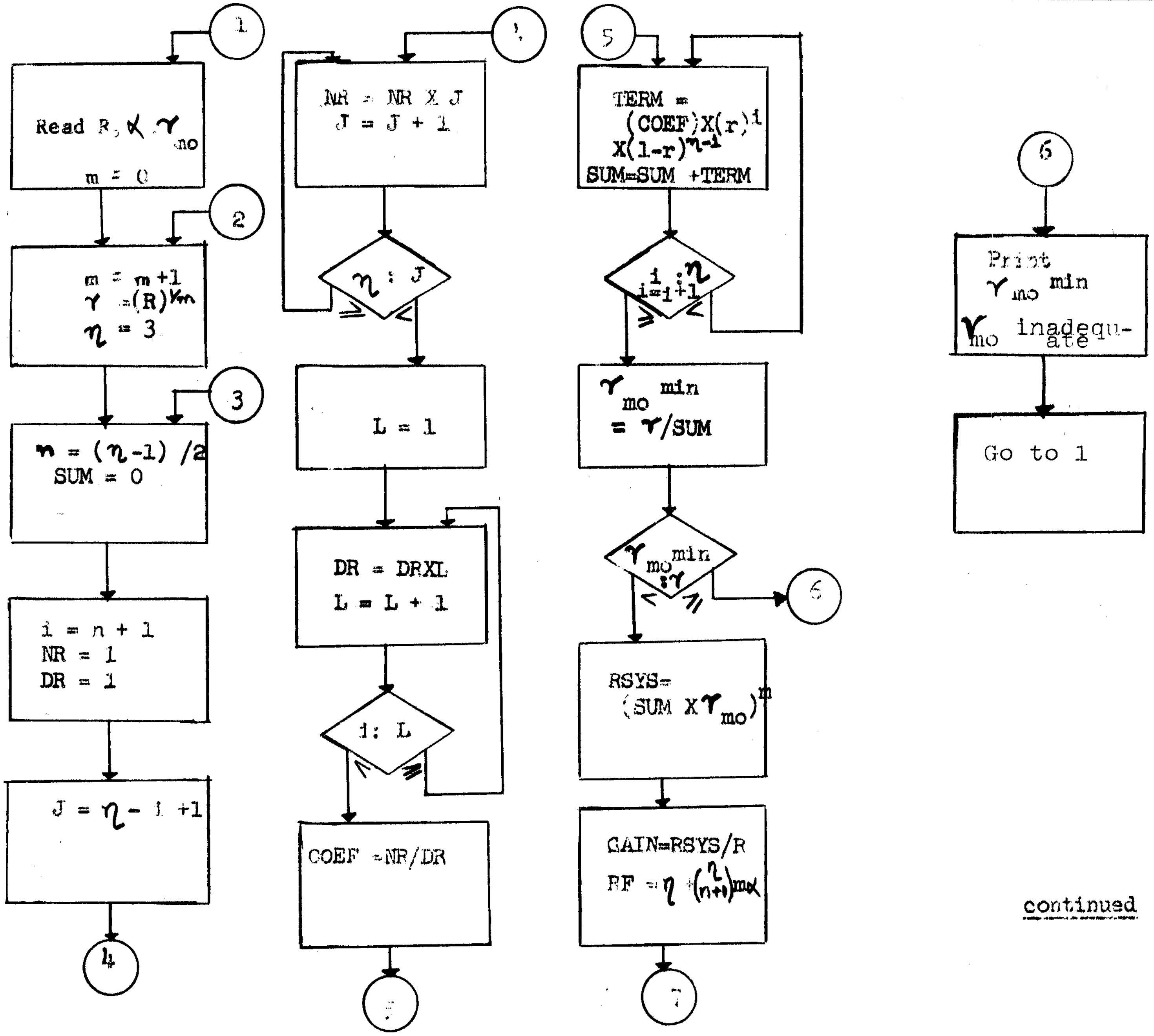
Let us define gain in reliability, G as follows

G = Reliability of redundant system/reliability of non-redundant system.

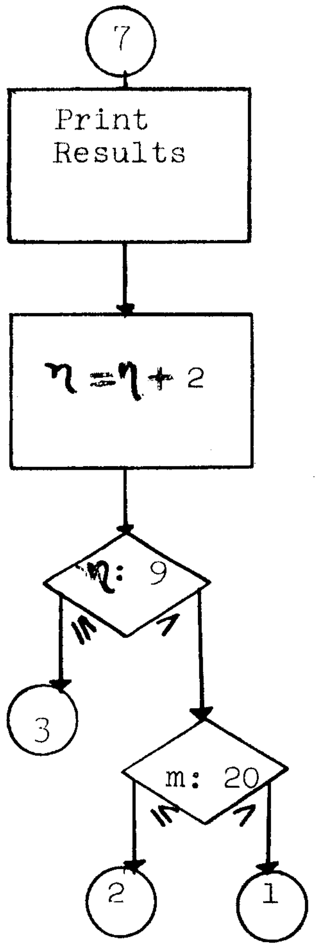
Then,

$$G = \left[r_{mo} \sum_{i=n+1}^{\eta} \binom{\eta}{i} r^{i} (1-r)^{\eta - i} \right]^{m} / r^{m}$$

A computer program (FORTRAN II D) was written to evaluate the effectiveness of different redundant configurations using the principle of output voting. The flow chart for the program is given on the following page. The program determines the minimum value of the majority organ reliability for imporvement in system reliability for different configurations. Given the majority organ reliability, the program determines the redundant system reliability, gain and redundance factor. From the results obtained we could determine the configuration that is compatible with system requirements. The program was run on the IBM 1620 computer for a few arbitarily chosen values of the parameters (this was done just to test the program) and the results are shown in the following few pages.


1
Read R, α, γ_mo
m = 0
2
m = m+1
γ = (R)^{1/m}
η = 3
3
n = (η-1)/2
SUM = 0
4
i = n + 1
NR = 1
DR = 1
J = η - i +1
4
NR = NR X J
J = J + 1
η : J
≥
<
L = 1
DR = DRXL
L = L + 1
i : L
<
≥
COEF = NR/DR
5
TERM = (COEF)X(r)^i X(1-r)^{η-i}
SUM=SUM +TERM
i : η
i=i+1
≥
<
γ_mo min = γ/SUM
γ_mo min : γ
<
≥
6
RSYS= (SUM X γ_mo)^m
GAIN=RSYS/R
RF = η (η/(n+1)) mα
7
6
Print γ_mo min
γ_mo inadequate
Go to 1

continued



R - non redundant system Reliability

γ_{mo} - majority organ reliability

$\gamma_{mo}min$ - minimum majority organ reliability

RSYS - Redundant system reliability

M - number of sub-systems into which the system is broken

RF - redundance factor

Figure 13: FLOW DIAGRAM OF COMPUTER PROGRAM TO DETERMINE SYSTEM CONFIGURATION

I) R(non redundant system reliability .85
r_{mo} (majority organ reliability) = .995 κ = .001

η	m	r_{mo}min	Redundant system reliability RSYS	Gain	RF redundance factor
3	1	.9049	.9345	1.099	3.003
	5	.9709	.9606	1.130	3.015
	10	.9846	.9438	1.110	3.030
5	1	.8732	.9690	1.139	5.010
	5	.9683	.9737	1.146	5.051
	10	.9839	.9507	1.118	5.100
7	1	.8604	.9829	1.156	7.035
	5	.9680	.9750	1.147	7.175
	10	.9838	.9510	1.119	7.350

II) R = .90 r_{mo} = .99 κ = .001

η	m	r_{mo}min	RSYS	Gain	RF
3	1	.9259	.9622	1.069	3.003
	5	.9804	.9448	1.050	3.015
	10	.9898	.9014	1.002	3.030
5	1	.9077	.9815	1.091	5.010
	5	.9792	.9505	1.056	5.050
	10	.9895	.9042	1.005	5.100
7	1	.9024	.9872	1.097	7.035
	5	.9791	.9509	1.057	7.175
	10	.9895	.9042	1.005	7.350

TABLE 3: ESTIMATES OF RELIABILITY USING HARDWARE REDUNDANCY (sample results of computer program)

APPENDIX C

We shall briefly outline two of the more important and often mentioned redundancy techniques: (a) Quadded logic proposed by Tryon and (b) Triplet scheme introduced by Maithra.

(a). Quadded Logic

The scheme consits of constructing the original network in quadruplicate. The quadded networks are fed with identical inputs. They produce identical outputs when there is no failure or malfunction. If there is a failure, proper interconnections of the quadding help swamp the error in the subsequent levels. The pattern of the connections is the basis for the suppression of errors. Tryon has described the proper patterns.

The scheme prevents system failure due to a single component failure and takes care of many multiple errors, as well. The non-redundant version of a network is shown in figure 14a. The redundant version is shown in fig. 14b.

As an illustration, let us consider the case when a_1 fails, yielding an output of 0 instead of 1. We, readily, see that in the next level, this is corrected at c_1 (ORgate) and at c_3 (ORgate) by the correct output, 1, from a_3. However, if a_1 fails, yielding 1 for 0, and the correct output from b_1, b_2, b_3, b_4 is 0, the failure propagates to one more level before it is corrected. This may be easily checked.

An exact estimate of the improvement in reliability of this has not been made, because the errors arising from the failures are suppressed depending on the location of the failures. Multiple failures will not affect the system, provided they are not close together. A "worst case" estimate is made by assuming that at least three of the four gates should operate properly, for success. If the probability of correct operation of a single gate is p, the probability of successful operation of the quadded scheme is

$$\begin{aligned} P &= p^4 + 4p^3 (1 - p) \\ &= p^3 (4 - 3p) \end{aligned}$$

A plot of the P versus p is shown in figure 14c. It is seen that only when $p > 0.76$ there is improvement in reliability.

b). <u>Triplet Scheme</u>

The triplet scheme consists in replacing a single gate by three gates (See figure 15.) The principle used is that there are redundant states in each of the blocks A, B and C of the triplet.

Maithra has shown that for maximum reliability, if the triplet is to perform OR function, the "normal" functions of

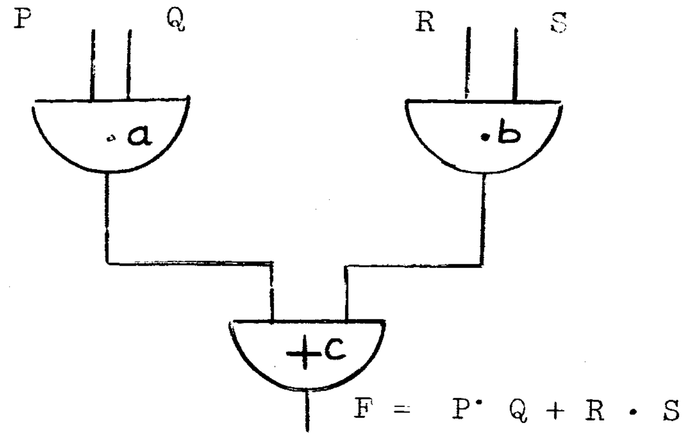


Figure 14a NON REDUNDANT VERSION

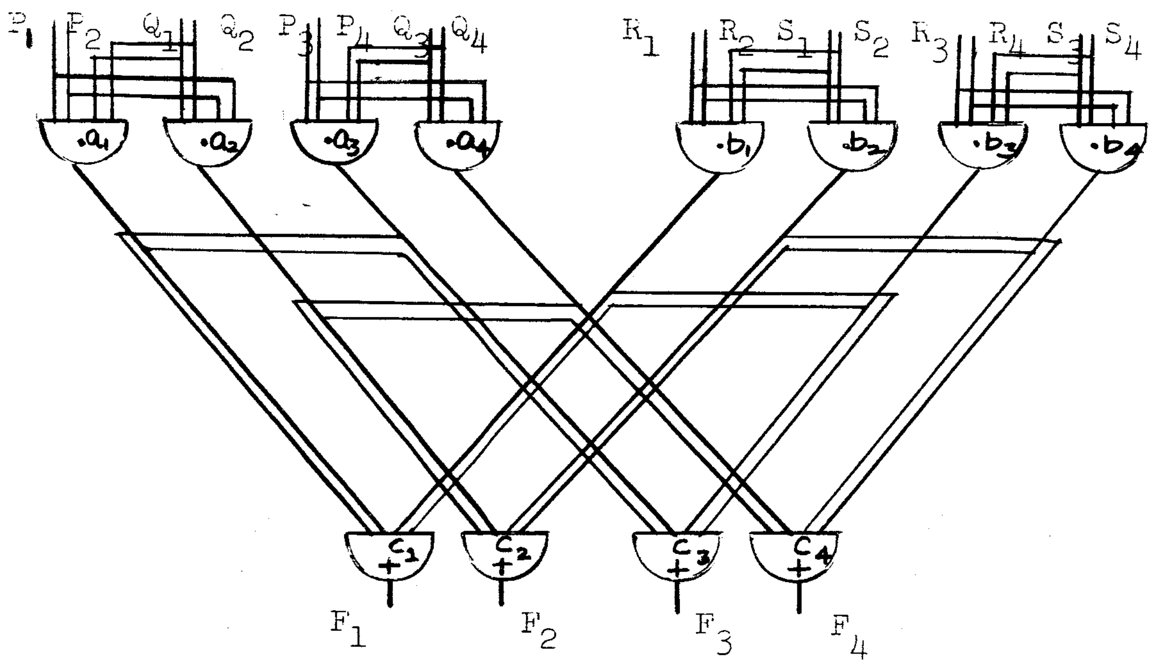


Figure 14b QUADDED VERSION

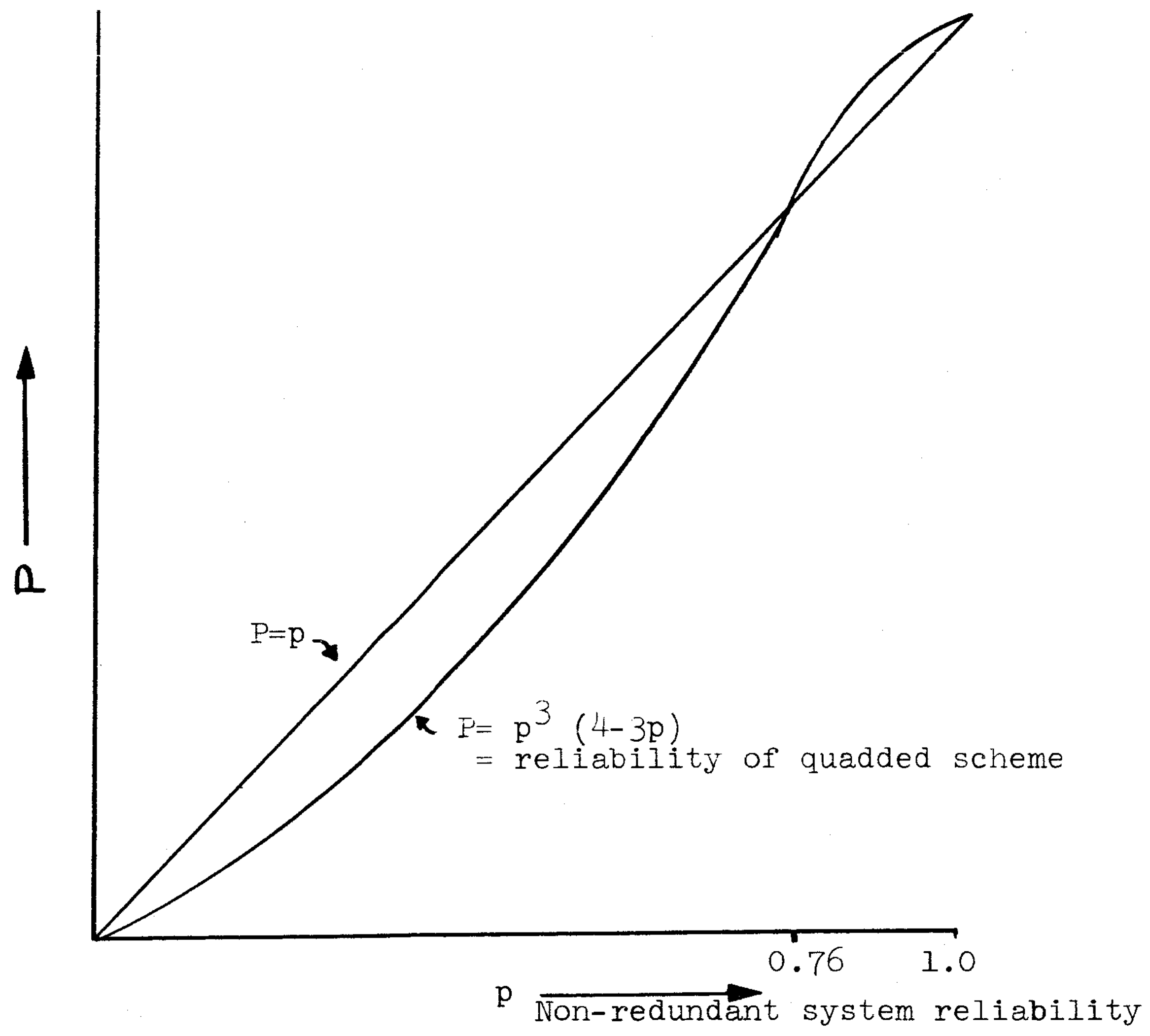


Figure 14 c RELIABILITY OF TRYON'S QUADDED SCHEME

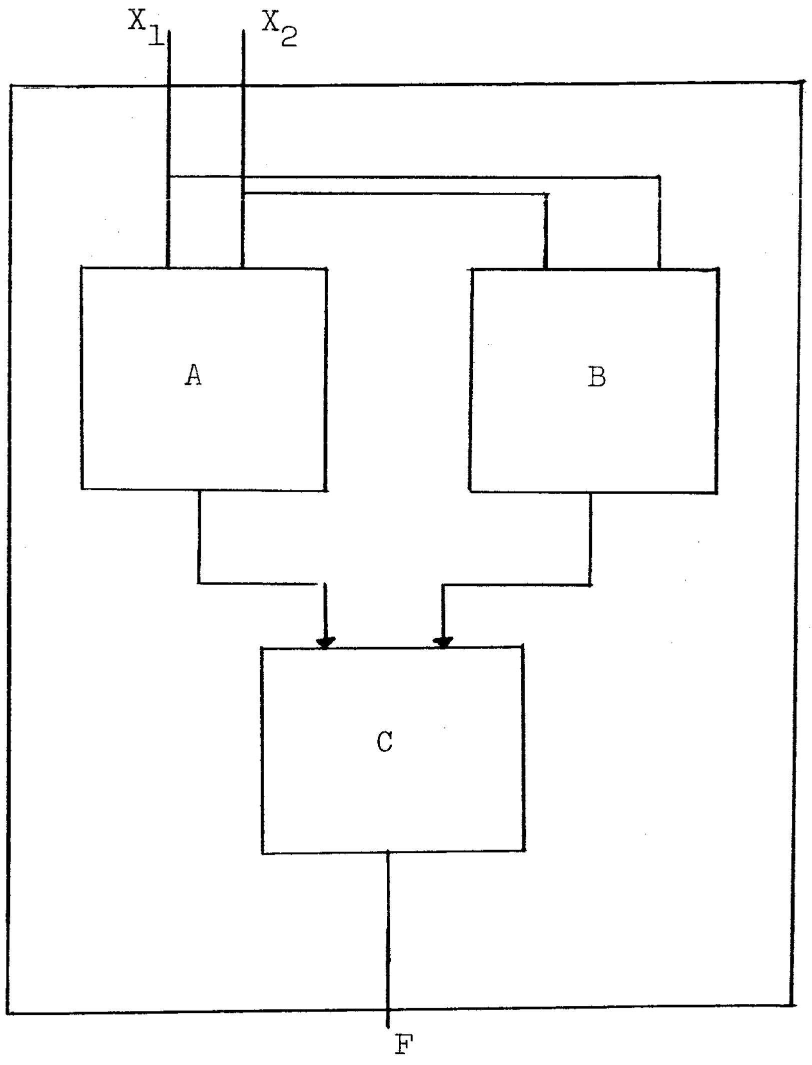


Figure 15 TRIPLET SCHEME

A, B and C should also be OR. Supposing, the triplet has been designed for an OR function and B malfunctions yielding $X_1 \cdot X_2$ (where X_1, X_2 are the inputs), the output of the triplet would be $X_1 + X_2 + X_1 \cdot X_2 = X_1 + X_2$, assuming A and C work properly. There are more such redundant states. Similarly for "AND" function, the normal function of A, B and C should be AND.

The analysis has not been extended to a NOT gate and is incomplete. In spite of the shortcomings, the method is a novel one.

REFERENCES

P. Rosenbloom: The elements of mathematical logic. Dover Publications. 1956.

Proceedings of the Symposium of Redundancy Techniques for Computing Systems. Spartan Books Co. Washington, D. C., 1962.

J. von Neuman: Probabilistic Logics, Automata Studies, Princeton University Press, 1956.

E. F. Moore and C. E. Shannon: Reliable circuits using less reliable relays, Journal of the Franklin Institute, September 1956.

W. H. Pierce: Improving Reliability of Digital Systems by Redundancy and Adaptation, Stanford Electronic Laboratories, Technical Report No. 1552-3; July 17, 1961.

S. R. Calabro: Reliability Principles and Practices, McGraw-Hill Book Co., Inc., 1962.

N. F. J. Matthews: An algebra of three valued logic and its application to digital circuitry, Master's thesis, The George Washington University, June 1959.