NASA CONTRACTOR REPORT -119926

SUMC MULTIPROCESSOR CONFIGURATION CONTROL ANALYSIS AND SPECIFICATION

Prepared under Contract No. NAS8-18405 by

James R. Kennedy, Sr.

FIELD SERVICES DIVISION
Aerospace Systems Center
Spaceborne Executive Project

For: COMPUTER SYSTEMS DIVISION, COMPUTATION LABORATORY

NASA -- GEORGE C. MARSHALL SPACE FLIGHT CENTER

Huntsville, Alabama

June 14, 1971

SUMC MULTIPROCESSOR CONFIGURATION CONTROL ANALYSIS AND SPECIFICATION

By

James R. Kennedy, Sr.

Computer Sciences Corporation
Huntsville, Alabama

ABSTRACT

This report analyzes the problem of configuration control given a multiprocessor environment. A key point in the analysis is the assumption that periodic changes in operational mode are desirable. Two modes of operation are considered: a Democratic Multiprocessor mode and a Triple Modular Redundant (TMR) mode. A feasibility approach to switching requirements associated with mode changes is developed. In conjunction with the necessary mode switching requirements, instructions to perform the switching operations are specified, including a prose description of the effect of particular instructions on the switching devices, and various mode status indicators are defined and derived from the voting mechanism employed in the Triple Modular Redundant operation mode. The use of various indicators is discussed and an approach to separation of transient and hard system failures is suggested. In addition, the approach outlines a method for locking the system in a TMR mode in the event that failures are so prolific as to counter indicate a switch from the TMR mode back to the Democratic Multiprocessor mode. Conclusions are presented that estimate the switching time requirements based on the number of logic levels, and the number of gates necessary to perform the switching. Recommendations for further analysis and detailed specifications are made.

NASA-GEORGE C. MARSHALL SPACE FLIGHT CENTER

PRECEDING PAGE BLANK NOT FILMED

TABLE OF CONTENTS

Page

SUMMARY 1

SECTION I. INTRODUCTION 3

SECTION II. CONFIGURATION CONTROL SCHEME. 5

A. Configuration Baseline 5

1. Memory Units 5
2. Processor Units 7
3. Configurations 7

B. Spares Switching 9

1. DMP Mode Spares 9
2. TMR Mode Spares 16
3. DMP Spares Switching Controls 16
   a. System Map. 16
   b. Configuration Control 23
   c. Instructions. 28
4. TMR Spares Switching Controls 28
   a. Vote Logic and Disagree Detection . . 33
   b. Decision Logic 33
   c. Combined Vote, Decision, and Switch Control 36
   d. Configuration Control 40
   e. Instructions 45

C. Mode Switching 48

SECTION III. CONCLUSIONS 49

SECTION IV. RECOMMENDATIONS 51

APPENDIX A. 53

LIST OF ILLUSTRATIONS

Figure	Title	Page
1	Generic Memory Element	6
2	SUMC Processor Element	6
3	DMP Configuration Block Diagram	8
4	TMR Configuration Block Diagram	10
5	Processor Bus Scheme	11
6	Memory Bus Scheme	11
7	Plug Position to Bus Switching for DMP Memory Read Operation	13
8	"n" Bit Switch Type 1 x 3	15
9	Voting/Decision/Switch Control Logic Scheme #1 for TMR	17
10	TMR Configured as in Figure 4 and 9	18
11	VDSC Scheme #2 for TMR	19
12	Configuration Control and Mode Switching for SUMC-Memory Read/Write	20
13	DMP Configuration Setup	25
14	Democratic Multiprocessor	26
15	Multiple Simplex	27
16	Memory Spare Switching	29
17	Processor Spare Switching	30
18	Majority Vote and Disagree Detection Logic	34
19	Decision Logic	38
20	Vote, Decision and Switch Control Logic	39
21	TMR Configuration Setup	43
22	Use of TMR Failure Indicators	46

LIST OF TABLES

Table	Title	Page
1	Availability and Configuration Map	21
2	System Map	24
3	DMP Configuration Control Instructions	31
4	Truth Table and Boolean Equations for Majority Vote/ Disagree Detection	35
5	Decision Network Truth Table and Boolean Equations . . .	37
6	TMR Configuration System Map Supplement	41
7	Final System Map	42
8	Element and VDSC Connections to Buses Resulting in a TMR Configuration	44
9	TMR Configuration Control Instructions	47

DEFINITION OF SYMBOLS

Symbol	Definition
a	Memory/SUMC address and control path bit width
AM	Action Map
B	Number of processor bus triples (address, read, write)
b	Bank address register bit width
BA	Bank Address
BG	Bus Good
BM	Bus Used by Memory
BP	Bus Used by Processor
CMM	Connect Memory like Memory
CMSC	Configuration Map and Switch Control
CPP	Connect Processor like Processor
DMP	Democratic Multi-Processor (e.g. UNIVAC 1108/EXEC 8)
ED	Element Disconnected
EG	Element Good
EP	Element Plugged-In
IOU	Input/Output Unit
L	Multiple Channel Errors
M	Number of Memory Plug Positions
MAR	Memory Address Register
MB	Memory Bank
MR	Memory Register
μsec	10^{-6} second
P	Number of SUMC plug positions
PRR	Product/Remainder Register
R	Error(s) in TMR channel T1, T2, or T3
S	Single Channel Error(s)
SBG	Sense Bus Good
SBP	Sense Bus used by Processor
SBM	Sense Bus used by Memory
SIM	Switch In Memory
SIP	Switch In Processor
SM	Setup Map
SMB	Set Memory Bank
SMC	Sense Memory Connected
SMG	Sense Memory Good
SMP	Sense Memory Plugged-In
SOM	Switch Out Memory
SOP	Switch Out Processor
SPC	Sense Processor Connected
SPG	Sense Processor Good
SPP	Sense Processor Plugged-Jn
SUMC	Space Ultrareliable Modular Computer
SWJ	Switch and Jump

DEFINITION OF SYMBOLS (continued)

Symbol	Definition
TMR	Triple Modular Redundant
V	Majority Vote Logic
VD	Majority Vote and Disagree Detection Logic
VDSC	Vote, Decision and Switch Control
w	Memory/SUMC data path bit width

FOREWORD

The work reported herein was administered in the Systems Research Branch, Computer Systems Division, Computation Laboratory, MSFC, with Bobby C. Hodges assigned as Technical Monitor. In addition to his routine duties as Technical Monitor, Mr. Hodges has added significantly to our insight into and understanding of related NASA programs through careful planning, coordination with in-house effort, and encouragement.

SUMMARY

This report outlines the architectural aspects of multiprocessor configuration control. Methods and concepts are developed to support the setup of arbitrary configurations given a specified capability for switching system elements. Three configurations, a democratic multiprocessor, dedicated simplex, and triple modular redundant, are emphasized but feasibility for establishing a variety of other configurations is shown.

Spares switching to replace failed system elements is outlined through the use of flow diagrams based upon a derived set of instructions. Complete configuration setup is similarly outlined. Based upon the methods depicted, estimates are given in the conclusion for various parameters that characterize the approach. Some of these are:

○	Number of instructions	18	
○	Processor Replacement Time	27.5	μsec
○	Memory Replacement Time	42	μsec
○	Triple Modular Redundant Setup Time	415	μsec
○	Multiprocessor Setup Time	229	μsec

The functions of a Configuration Map and Switch Control unit are outlined. When these are combined with previously specified executive functions, the architecture and functional role of a system control unit (or executive controller) becomes apparent. Recommendations are made for further work to refine the definition of the executive controller and the SUMC processor role in communicating with the controller through primitives for process control and configuration control.

SECTION I. INTRODUCTION

Appendix A states the scope of work that guided the effort resulting in this report. The major objectives were met by the specification of a scheme for spares switching and configuration setup based upon an assumed bus structure and associated switching networks, triple modular redundant (TMR) networks, a derived set of instructions, and a configuration mapping scheme.

No consideration was given to power supply control; all switching is performed with respect to signal lines with power assumed to be "on."

Failure detection can result from the execution of appropriate diagnostics, or switching to a TMR mode of operation on a periodic basis when operating in a high throughput mode. Diagnostics are not discussed, but a procedure for mode switching is developed along with TMR failure detection and spares switching.

With respect to configuration optimization, "best" (as used in the scope of work) is taken to mean either maximum throughput with no internal redundancy or internally redundant (TMR, dual, or time redundant simplex). A method for achieving an arbitrary configuration with these best characteristics is given.

SECTION II. CONFIGURATION CONTROL SCHEME

This section outlines the assumptions concerning allowable modes of operation, and develops and discusses a technologically feasible set of switching elements necessary to support mode switching. In addition, spares switching is considered and a feasible approach to replacement of failed modules by operable spares is outlined. The concept of voting loops is introduced and serves as the basis for a generic switching arrangement specification.

A. Configuration Baseline

Two problems are considered in the development of a scheme for configuration switching:

- Replacement of failed modules by spares
- Mode Switching between a democratic multiprocessor (DMP) mode and a triple modular redundant (TMR) mode.

Block diagrams showing interfaces between various units are used as a basis for discussion of the concepts derived. In order to avoid confusion, it is necessary to discuss the functioning of these units with respect to one another, and show generic block diagram elements for each of the units considered. Specifically, memory units and processor units are discussed. Based upon the generic elements, a general configuration diagram for the democratic multiprocessor and the triple modular redundant systems is outlined. A method for configuring spares is given as a basis for a discussion of spares switching.

1. Memory Units. Figure 1 shows a generic memory element used to depict memory units in configuration block diagrams. The memory unit is considered to have a register, designated in the element as BA, which contains the memory element's bank address. For reasons to be made clear in subsequent text, the bank address register contents are specifiable under program control in contrast to the normal practice of manual entry through toggle switches or hard wiring.

MR represents a memory register used to transfer data to one of P processing units such that the data represents the contents of a previously specified memory address. The set of address and control lines entering

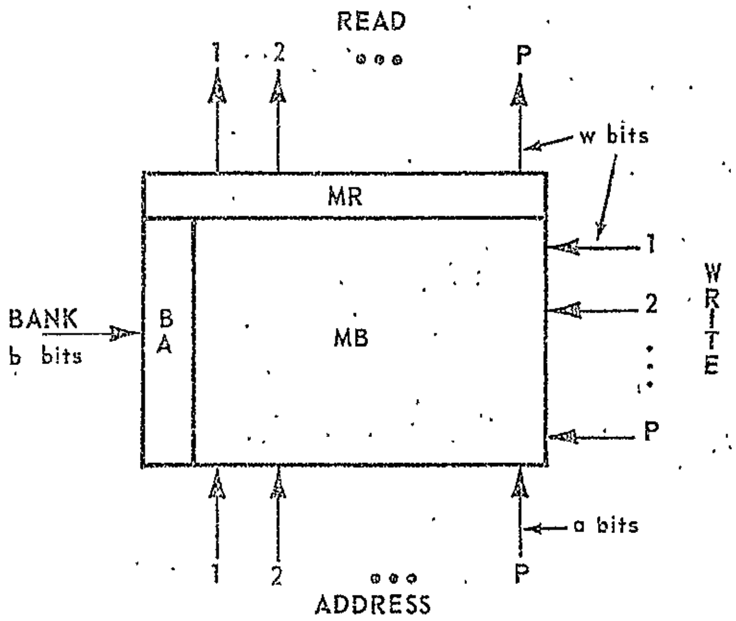


FIGURE 1

GENERIC MEMORY ELEMENT

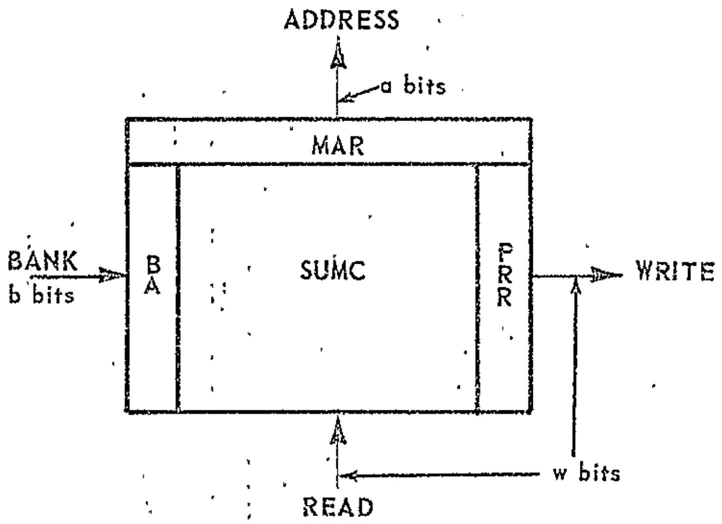


FIGURE 2

SUMC PROCESSOR ELEMENT

the memory element are sometimes referred to as processor access ports. The address supplied by a processor includes the memory element word address and the bank address of the element being referenced by the processor. In addition, various control lines for specifying read/restore, clear/write, etc., are included in the address group.

Operation of the memory element is assumed to take place as follows: A control line entering from one of the P processors corresponding to an arrow entering on the address side indicates that a memory reference is required; recognition of this control line by all memory elements causes them to gate the specified bank address into internal registers suitable for comparison with the contents of their respective BA register; a comparison match implies that the memory element is to perform a decoding operation on the remainder of the (word) address and gate the word contents into its memory register (MR); a bank address mismatch results in no action on the part of the memory element; when MR has been set to the addressed memory word contents, the memory element raises a control line to signal to the accessing SUMC processor unit that a read operation has been completed. In the case of a write operation, PRR contents from the appropriate accessing processor unit are gated into the memory element on the lines shown on the write side of the memory element. (The element is shown to have nominal read and write line widths of w bits each. This is possible if all control lines are included in the address group).

In the event that multiple processor units attempt to access the same memory element simultaneously, the memory element will respond to the accessing processor whose line number is the lowest by scanning from 1 through P consecutively. Higher numbered processor units are delayed until the lower numbered units are satisfied.

2. Processor Units. Figure 2 depicts a generic SUMC processor element as it relates to a memory element. It shows that in order to communicate with memory, the processor element must specify the appropriate bank address as part of MAR. Under some conditions, it may be necessary for the bank address portion of MAR to be explicitly specified to a processor element under program control. This requirement is analyzed in detail in subsequent discussions regarding TMR mode operations. The arrows going into and out of the SUMC processor element are typical and do not require explanation except to mention, as in the case of memory, all control lines are included in the MAR group.

3. Configurations. Figure 3 is a block diagram consisting of generic memory elements and SUMC processor elements illustrating the

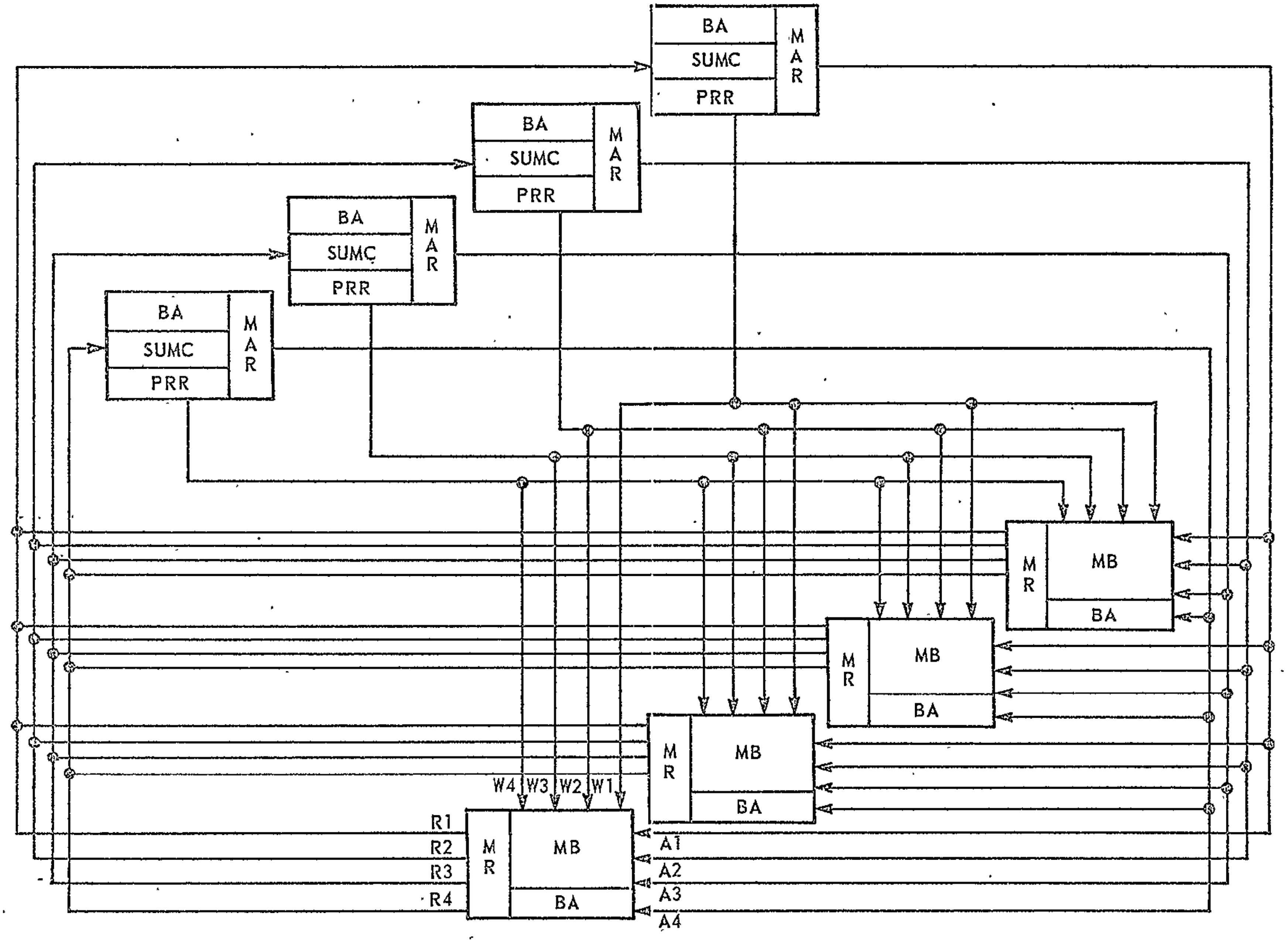


FIGURE 3

DMP CONFIGURATION BLOCK DIAGRAM

lumped data and control paths of a democratic multiprocessor system. The diagram is restricted to show only the lines necessary to support memory read and write operations initiated by the SUMC. Figure 4 represents a TMR configuration wherein Voting, Decision, and Switch Control (VDSC) logic is shown in the paths for address, read and write operations. SUMC processor and memory elements are shown such that each element is associated with a single TMR channel, and such that each channel represents a complete loop that does not overlap with either of the other two loops. In this particular figure, the following associations are depicted:

- Processor 3 (P3) with TMR channel 1 (T1)
- P1 with T2
- P2 with T3
- Memory number 2 (M2) with T1/and thereby P3
- M8 with T2/P1
- M3 with T3/P2.

Although the processor elements shown in figures 3 and 4 are peculiar to the SUMC, it is clear that these diagrams are also representative of the wiring paths necessary to support memory read and write operations initiated from an input/output unit (IOU). The diagrams do not show the data routing and control loops associated with SUMC to IOU communications; the necessary switching and voting logic to support this loop is derivable from that developed to support the SUMC to memory operations.

Therefore, the development scheme is as follows: these figures are the basis for a development of the necessary switching and voting logic to support a data routing and control loop scheme that is generic in the sense that the general results can be applied to satisfy the requirements of other system loops. Although the resulting scheme may not be the most efficient for a particular combination of system elements, it is not the purpose of this report to perform a detailed design that is optimized for efficiency.

B. Spares Switching

1. DMP Mode Spares. With regard to the configuration depicted by figure 3, the maximum number of memory elements that may be on-line at any given time is limited of course by the width of the bank address field of the memory address register of the SUMC (which is assumed to be the same as the width of the BA register contained in each memory element). Figure 5 shows a simplified version (read only, say) of what can be referred to as a "processor bus" scheme similar to that implied by the diagram of

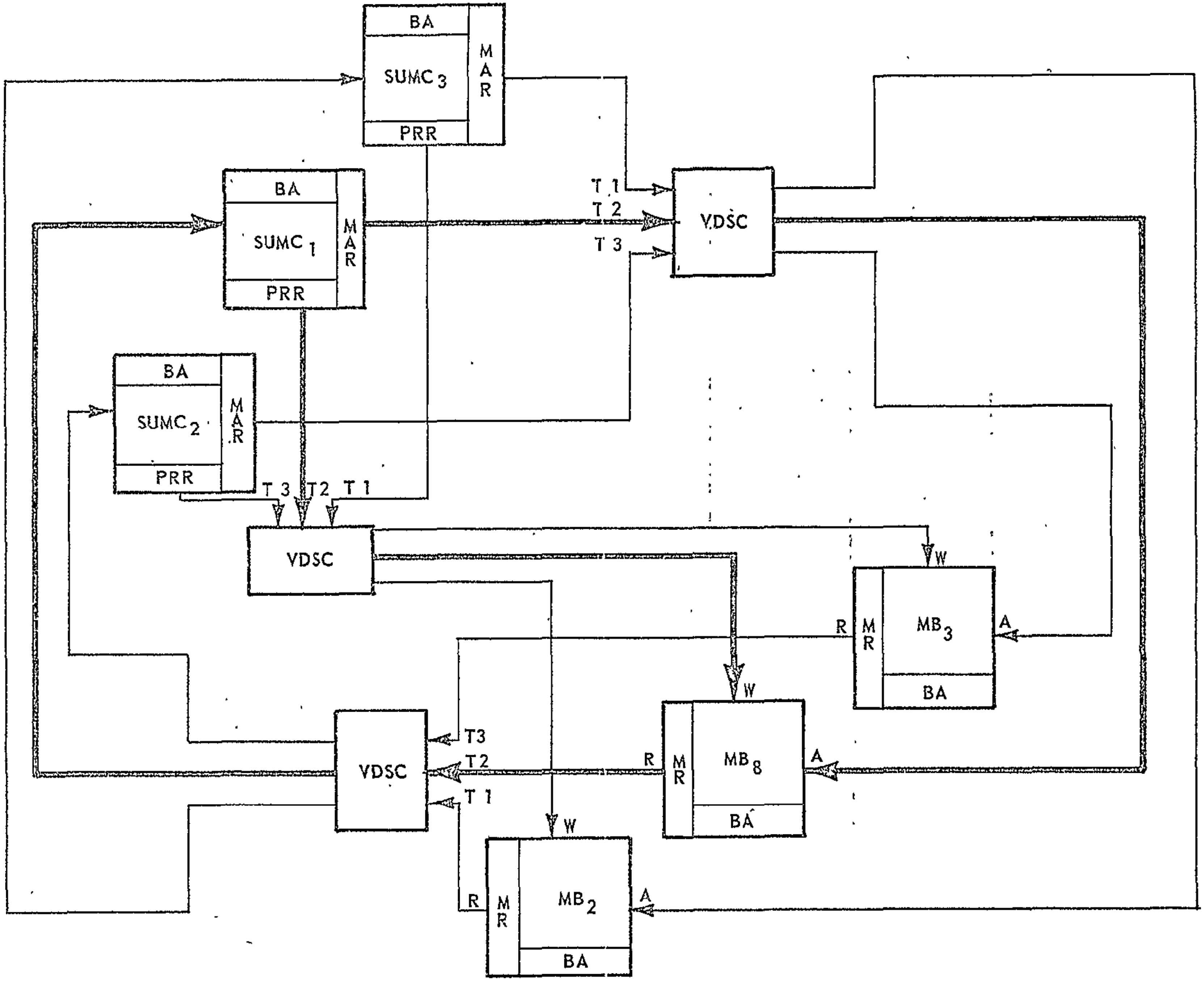


FIGURE 4

TMR CONFIGURATION BLOCK DIAGRAM

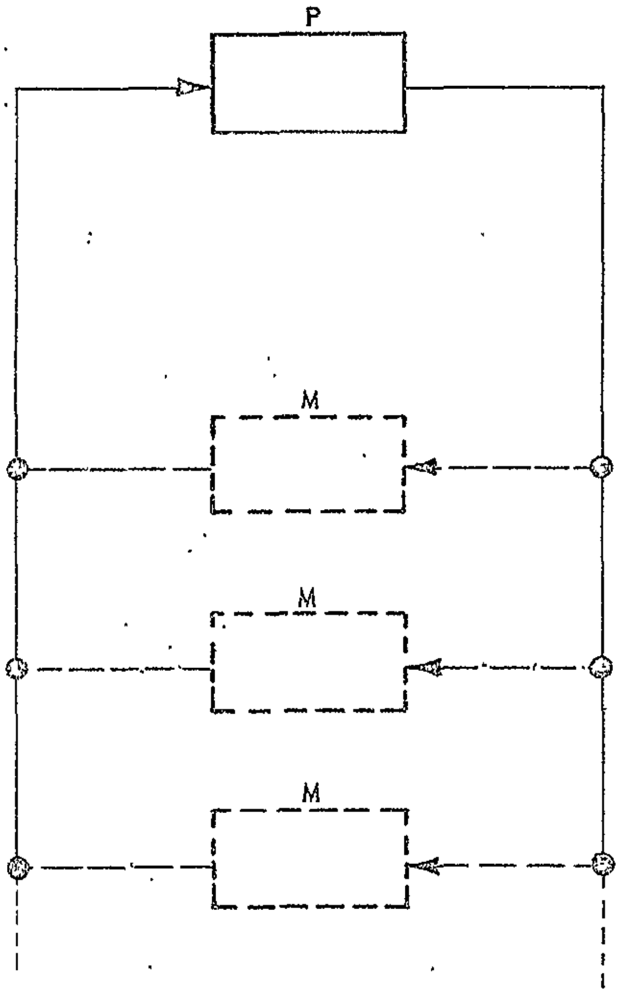


FIGURE 5

PROCESSOR BUS SCHEME

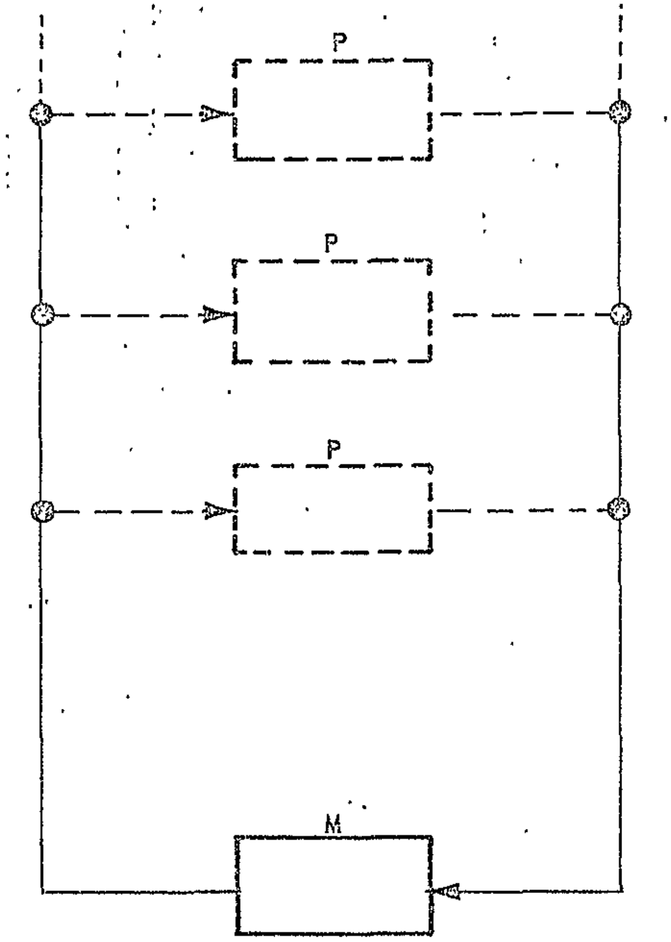


FIGURE 6

MEMORY BUS SCHEME

figure 3. In this scheme, memory elements are connected to a set of lines associated with a particular processor element. Each memory access request on the part of the processor is accompanied by a specification of the memory bank that is to respond to the access request. In a scheme of this type, a large bank address field in the memory address register could be provided for, and the associated bank address register in each memory element could be made correspondingly large. It would be possible to attach 2^n memory elements, where n represents the number of bits in the bank address. The distinguishing feature of the scheme shown in figure 5 is that the processor element is responsible for specifying the address of one of many memory elements attached to its bus.

Figure 6 shows what may be referred to as a "memory bus" scheme wherein each memory element has a single access bus to which multiple processor units may be attached. In this scheme, the memory element must be responsible for addressing the appropriate processor element attached to its bus.

The memory bus scheme obviously does not allow for simultaneous access of the memory element by multiple processors. On the other hand, the processor bus scheme, while it does not allow processor access to multiple memory elements simultaneously, can be structured to allow for multiple processor access to a single memory element simultaneously by phasing multiple memory element access ports to each processor bus such that contiguous word addresses are distributed across separate sub-elements within the memory element. Partly for the latter reason, but primarily for the reason that we are discussing what is essentially a von Neumann (sequential) processor organization, the processor bus scheme is the favored approach. In the case of associative processing, the memory bus scheme can offer certain advantages; combinations of the two schemes might even be attractive under some circumstances.

Figure 7 shows a (memory read only) processor bus scheme for spares switching based upon a fixed set of B processor buses to which both processor and memory elements may be connected under switch control. The switching actions make specific associations between fixed processor element plug-in positions and fixed memory element plug-in positions (and not the elements themselves). In this figure, the memory elements are represented by a generic plug position containing as many access ports as there are processor plug positions. It would be possible, of course, to have certain known memory plug positions which allowed access to a smaller number of processor plug positions.

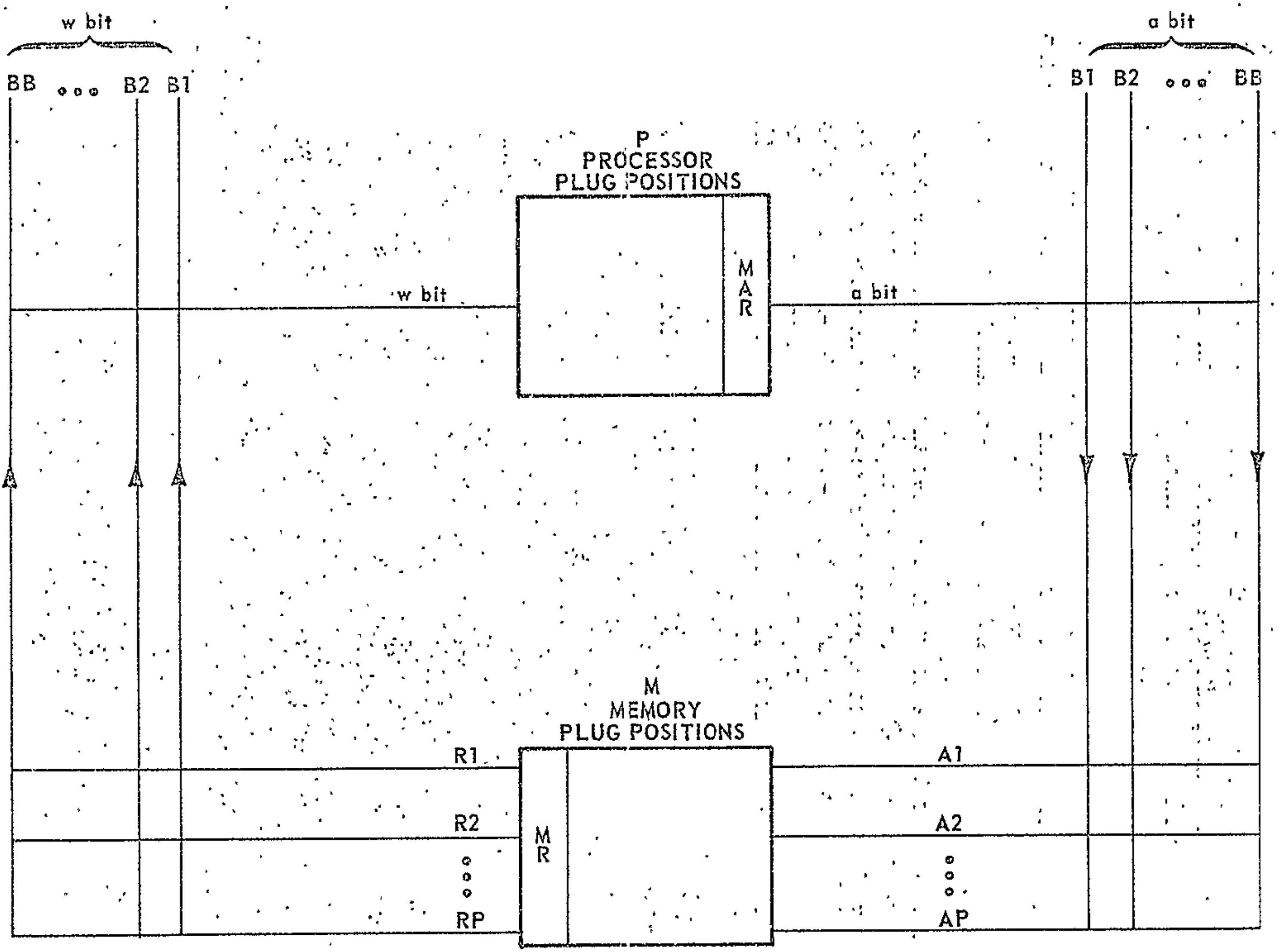


FIGURE 7

PLUG POSITION TO BUS SWITCHING FOR DMP MEMORY READ OPERATION

Figure 8 illustrates a logic gate switch that will distribute n-bits of data (D) to one (or none) of three possible n-bit buses labeled A, B, and C. This switch will be referred to as an "n-bit 1x3" switch. The symbology represented by "1x3" is intended to mean one set of input lines is connectable, through the use of enable/disable logic, to one and only one of three output line sets (or none). Similarly, 3x1 means that there are three input sources, one and only one of which is selectable as output. No figure is shown for the 3x1 type switch since it is clear that it can be obtained by simply reversing the "and" gates, labeled "E/D" in figure 8, and the direction arrows, while keeping the enable flip-flops connected as shown. (Diode switching matrix networks could also be considered for selection and distribution.)

Considering again only the read operation, it is clear that for each processor plug position, one a-bit 1xB switch and one w-bit Bx1 switch is sufficient to connect each processor plug position to any one of the B bus positions on both the a-bit and the w-bit buses. If each memory plug position has P processor access ports, then each memory plug position requires P w-bit 1xB switches and P a-bit Bx1 switches. The total number of switches required is seen to be:

- P a-bit 1xB
- P w-bit Bx1
- M•P a-bit Bx1
- M•P w-bit 1xB.

Several observations can be made on the basis of the discussion thus far. The first is that B should be greater than the maximum number of connected processor plug positions. That is, spare buses are desirable. It is also desirable to have more processor plug positions than the maximum allowable number of on-line processor elements. The same is true of memory plug positions.

In a space station environment, it is possible for the number of on-board spares to exceed the number of available plug positions for any given type of element. In unmanned missions, each spare obviously corresponds to a unique plug position. In the (space station) case where certain plug positions may be vacant, it would be necessary to provide for an indicator associated with each plug position to show whether the plug position is vacant or not.

Recalling that figure 3 shows separate buses for read and write operations, each w-bits wide, and that figure 7 represents only the read operation, it is clear that a considerable reduction in piece-part count can be realized by sharing the w-bit bus between the read and write functions. Of course, this

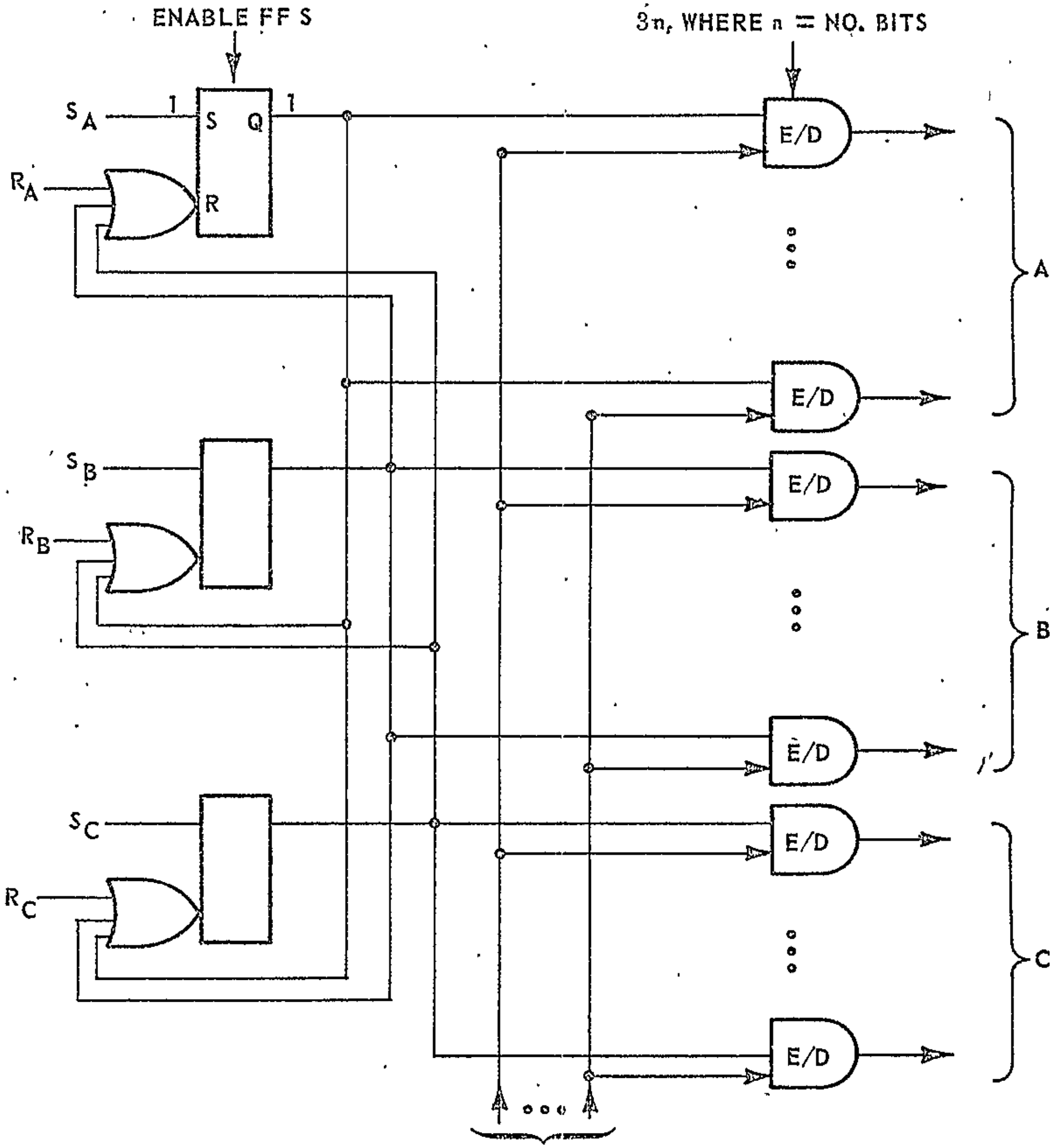


FIGURE 8

"n" bit SWITCH TYPE 1 x 3

requires bi-directional bus amplifiers and controls, thereby complicating matters somewhat. In addition, TMR mode operations are somewhat complicated by sharing. In the sequel, read and write buses will be kept separate and no effort is made to contrast the two approaches.

2. TMR Mode Spares. Based upon the DMP spares switching scheme depicted by figure 7, several methods are available for inclusion of VDSC elements for support of TMR mode operations. Figure 9 shows one such wherein the VDSC inputs are switched directly from the element output lines, and VDSC outputs are switched onto the appropriate processor buses. Figure 10 shows the switch closures for this method associated with setup of the TMR configuration depicted in figure 4. This scheme implies that the VDSC associated with the output of each element is located physically near groups of like elements in order to maintain short input lead lengths. Figure 11 indicates a method for overcoming the requirement for close proximity of VDSCs with their element types. In this figure, the VDSCs have both their inputs and outputs switched to processor buses. It is necessary, of course, to have a larger number of processor buses to support TMR mode operations with this method.

Through the incorporation of the voting, decision, and switching control circuitry into the spares switching network required for DMP spares switching, a fairly uniform method for TMR mode spares switching is possible. That is, the controls for switching failed elements out and spare elements in are the same as those required to support spares switching for DMP operations. In addition, the scheme shown in figure 11, although it requires additional buses, allows for full testing of all elements and their associated bus switching networks.

Figure 12 shows the final configuration control and mode switching setup for SUMC-memory read/write operations. In this diagram, three separate bus structures, one each for address, read, and write, are shown with a VDSC, having the appropriate bit-width, associated with each.

3. DMP Spares Switching Controls.

a. System Map. Based on the configuration depicted in figure 7 for DMP spares switching, table 1 shows a matrix representing an availability and configuration map that can be used to indicate all possible connections of elements to buses. Several indicators showing gross status of elements and buses are provided to assist as discussed below in program control of the switch networks.

An element is considered to be disconnected (ED = true) if it is not connected to one bus in each of the bus groups. Thus, a processor connected

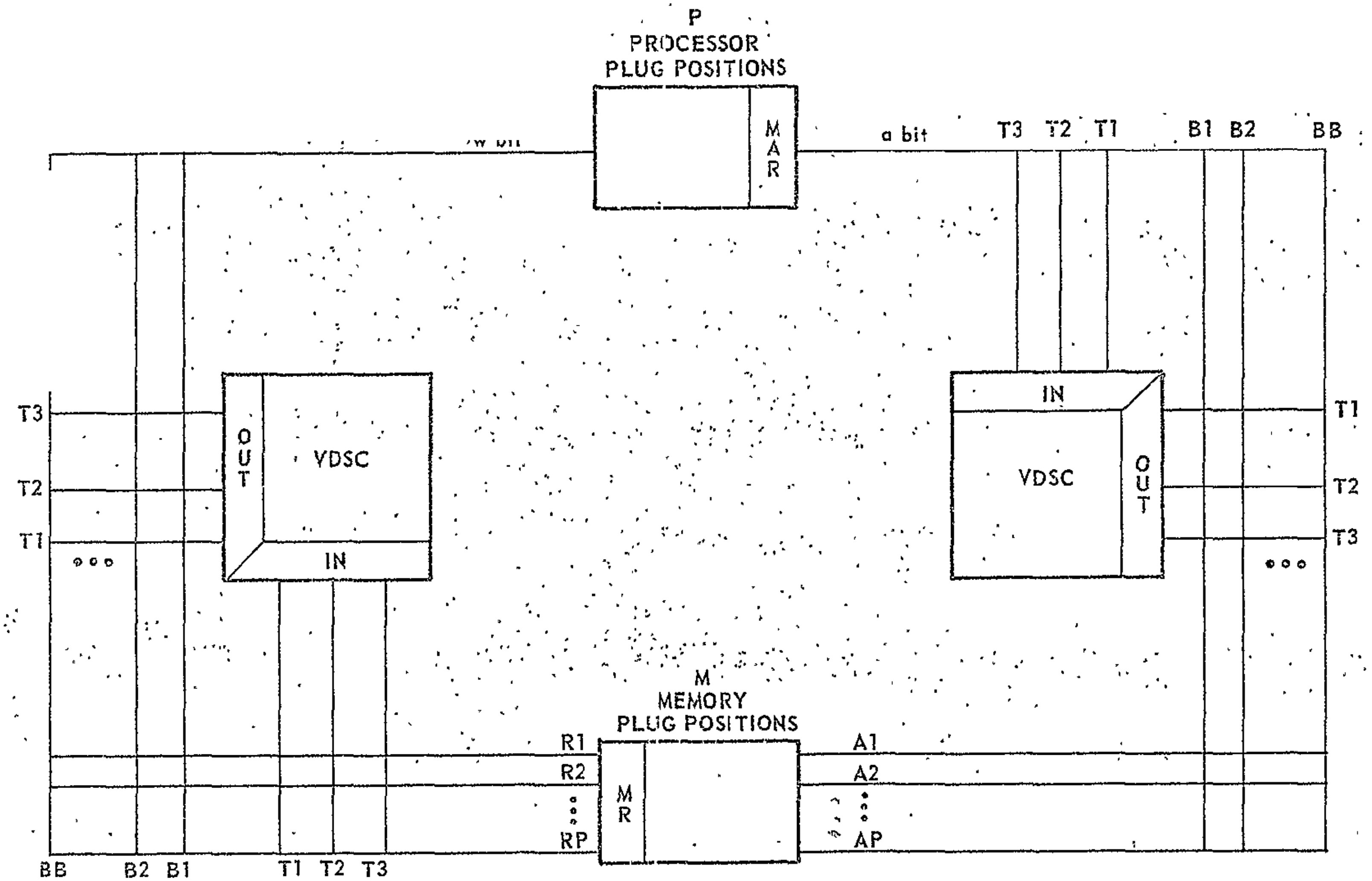


FIGURE 9

VOTING / DECISION / SWITCH CONTROL LOGIC SCHEME #1 FOR TMR

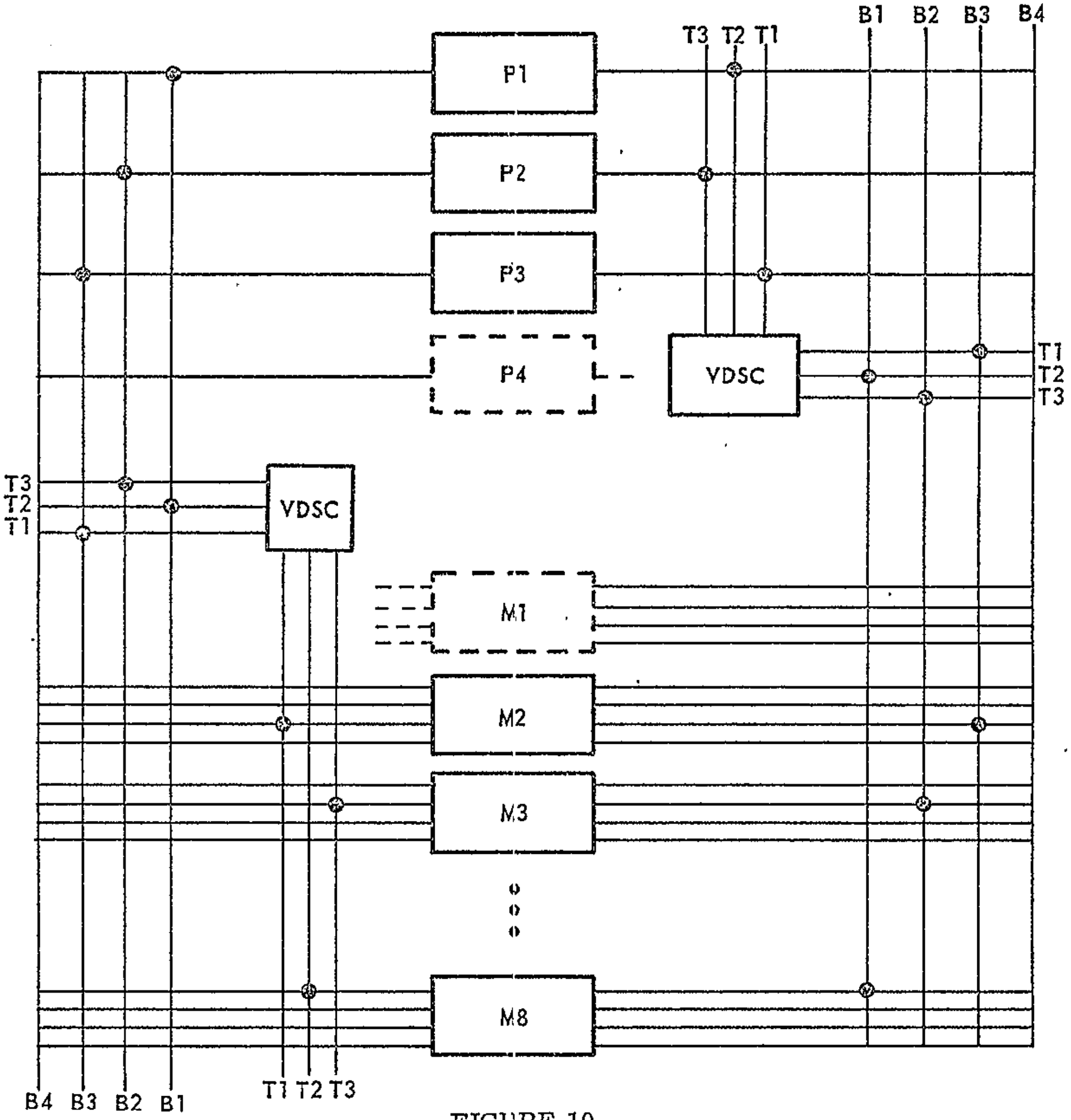


FIGURE 10

TMR CONFIGURED AS IN FIGURE 4 AND 9.

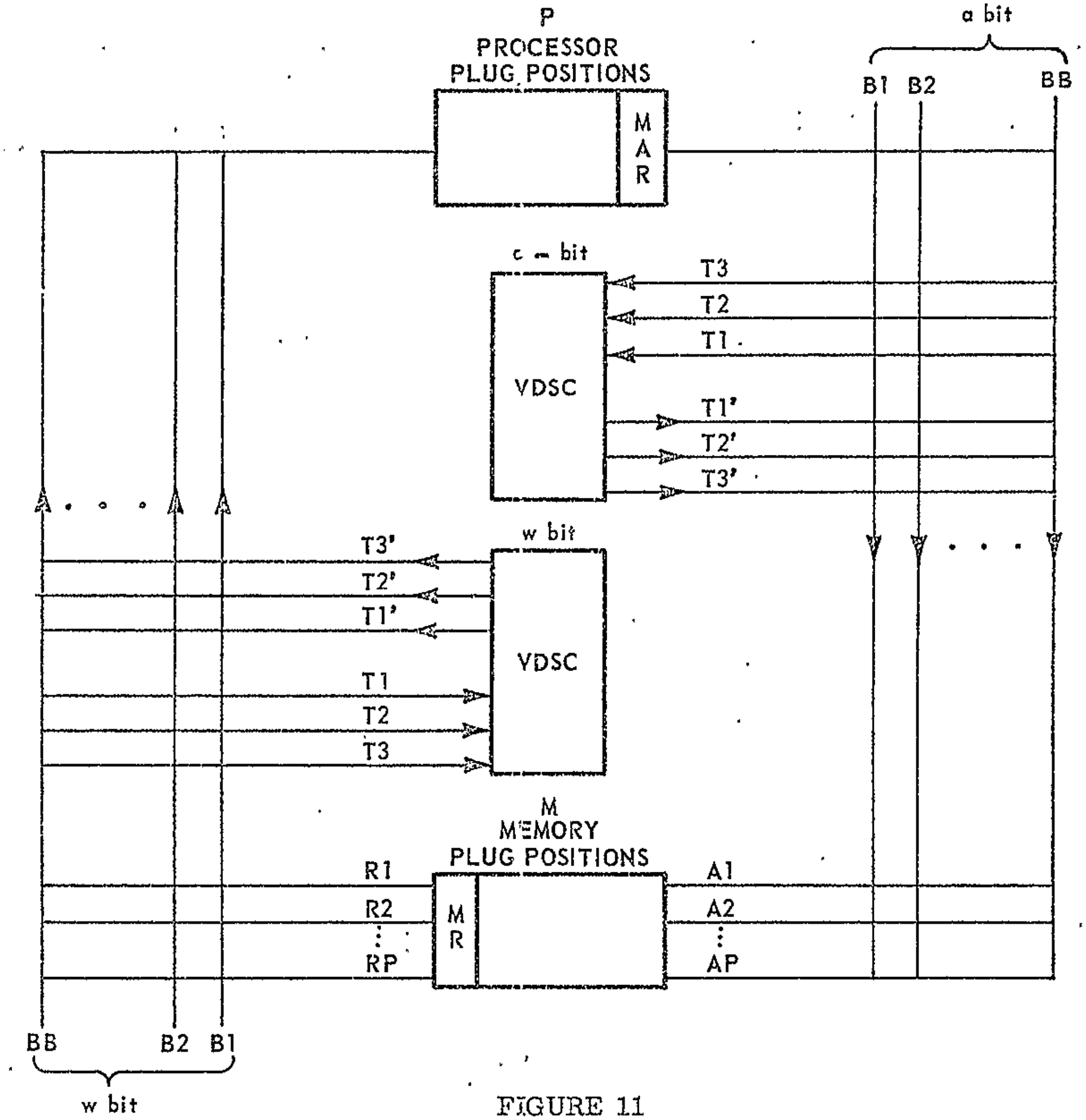


FIGURE 11

VDSC SCHEME#2 FOR TMR

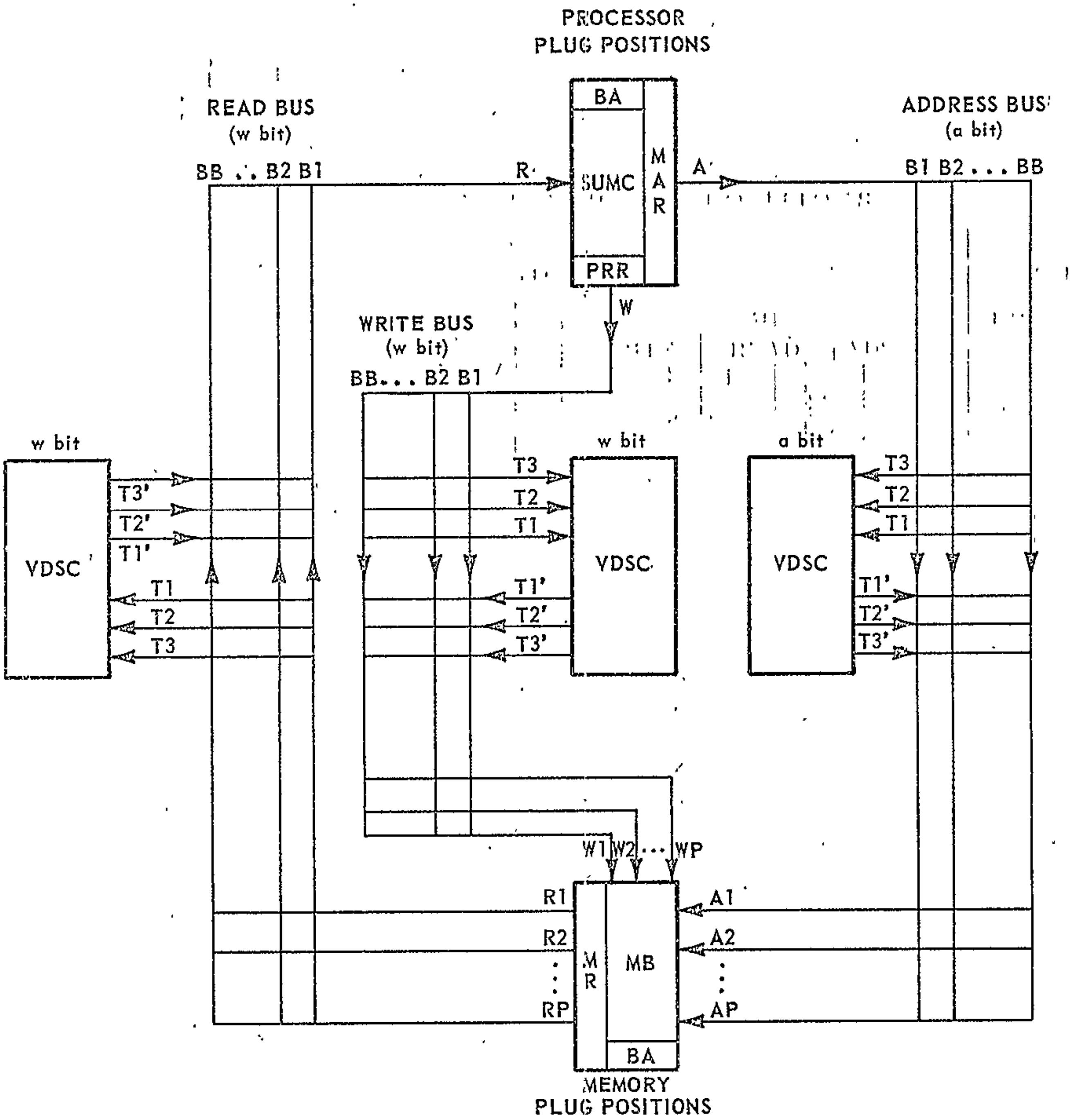


FIGURE 12

CONFIGURATION CONTROL AND MODE SWITCHING FOR SUMC - MEMORY READ/WRITE

TABLE 1. AVAILABILITY AND CONFIGURATION MAP

					BUS NUMBERS					
					ADDRESS	READ	WRITE			
					B1 B2···BB	B1 B2···BB	B1 B2···BB	ED	EP	EG
ELEMENT PLUG-IN POSITIONS	SUMC	P1		A		▨	▨			
				R	▨		▨			
				W	▨	▨				
		⋮								
		PP		A		▨	▨			
				R	▨		▨			
				W	▨	▨				
	MEMORY	M1	ADDRESS	A1		▨	▨			
				A2		▨	▨			
				⋮		▨	▨			
				AP		▨	▨			
			READ	R1	▨		▨			
				R2	▨		▨			
				⋮	▨		▨			
				RP	▨		▨			
			WRITE	W1	▨	▨				
				W2	▨	▨				
				⋮	▨	▨				
				WP	▨	▨				
		⋮								
		MM	ADDRESS	A1		▨	▨			
				A2		▨	▨			
				⋮		▨	▨			
				AP		▨	▨			
			READ	R1	▨		▨			
				R2	▨		▨			
				⋮	▨		▨			
				RP	▨		▨			
			WRITE	W1	▨	▨				
				W2	▨	▨				
				⋮	▨	▨				
				WP	▨	▨				
				BG						
				BP						
				BM						

ED: Element Disconnected
EP: Element Plugged-In
EG: Element Good

BG: Bus Good
BP: Bus Used by Processor
BM: Bus Used by Memory

to only the address and read buses is considered to be disconnected. (This meaning is broader and less flexible than could be applied.) A column is provided to indicate whether an element is plugged-in or not. This could be implemented through use of a special pin in the plug, for instance, such that EP = true only when the plug position is not vacant. Finally, an element is considered to be good (EG = true) when it is not known to have failed in some respect.

The indicator (BG) specifying a good bus is interpreted in the same way as EG. Since this table and the associated switching have been given a high degree of connection flexibility, it is possible to have element connections to bus groups such that the bus numbers vary from one group to the next. That is, P1 could be connected to address bus B1, read bus B2, and write bus B3. In order to allow for rapid configuration control, additional indicators are provided to show the use of a given bus by a processor (BP = true) and memory (BM = true).

Space for two maps is required: one, referred to as a setup map (SM), is provided as a working space that is used to build-up switch settings associated with a configuration being formed thus preventing possible conflicts with the prevailing configuration; the second, referred to as the action map (AM), is provided to record the current switch settings thus permitting a "copy" capability, and to save the status information outlined above. Once the SM has been formed for a specific configuration, its contents, indicating desired switch settings, is conveyed to the switch control logic for switching action and retention in the appropriate AM space.

Certain instructions are specified below for the purpose of indicating specific switch connections. These instructions initially operate on an empty SM space by setting the appropriate SM bits indicating the desired connections. Other instructions are defined for the purpose of sensing the status indicators discussed above. These instructions sense the indicator bits associated with the AM space. There are no status indicators (ED, EP, etc.) associated with the SM. Finally, an instruction is defined for the purpose of initiating the switching action implied by the bit pattern of the SM after which the contents of the SM replace the contents of the AM; upon replacement, the SM space may be cleared (for future configuration establishment), after which control of all connected processors is transferred to a specified location.

The AM is envisioned to be contained in a scratch memory that is firmware accessible only. The SM space may also reside in scratch memory if speed is important; main memory could be used otherwise. Since the storage of information placed into both SM and AM is performed by micro-orders rather than direct main memory code, the layout of table 1 can be considered as merely representative of the actual format which should be determined in such a way as to optimize firmware logic. This is a trade study area.

Since the action map contains all of the information indicated in table 1, whereas the setup map contains that subset of table 1 exclusive of ED, EP, EG, BG, BP, and BM, SM is not shown as a separate table, and discussions that follow will not distinguish between the two maps except where an ambiguity may arise.

With the understanding that connections exist only for address lines to address buses, read lines to read buses, and write lines to write buses, it is possible to reduce the map to a more condensed form as shown in table 2. This eliminates the shaded areas of table 1. The resolution potential of the ED status indicator is reduced by the condensation but the meaning outlined above is unchanged.

b. Configuration Control. The system map discussed previously is used under program control to construct various system configurations, and to assist in the replacement of spares when they exist. Several illustrations of operations that can be performed are provided here. In order to support these operations, a set of instructions is derived. The use of these instructions is indicated in the illustrations.

A feasible approach to setting up a class of configurations, one of which is the democratic multiprocessor configuration discussed previously, is shown in figure 13. The general scheme for configuration setup assumes that at least one processor element and one memory element are connected to a triple (one bus each address, read, and write groups) of buses to support program execution. This connection could be accomplished through the use of a cold-start or boot loading action. Reference is made to a boot processor and boot memory accordingly. The cold-start operation is assumed to result in an ultimate transfer of control to the "CONFIGURE" entry of figure 13. This routine first locates a triple of good, disconnected buses that will serve as a processor bus. Next, a good processor is identified and connected to the bus triple.

At this point a test of configuration-type is made. Figure 14 shows the operations performed in order to establish a democratic multiprocessor configuration having the maximum number of processors and memory elements connected as shown in figure 3. Figure 15 illustrates the operations performed to establish a multiple-simplex configuration wherein all operable processors are connected in such a way that each accesses a single memory element exclusively. Other configuration types are possible but not shown.

TABLE 2. SYSTEM MAP

				BUS NUMBERS											
				ADDRESS.			READ			WRITE					
				B1	···	BB	B1	···	BB	B1	···	BB	ED	EP	EG
ELEMENT PLUG POSITIONS	SUMC		P1		···			···			···				
			⋮		⋱			⋱			⋱		⋮	⋮	⋮
			PP		···			···			···				
	MEMORY	M1	P1		···			···			···				
			⋮		⋱			⋱			⋱		⋮		
			PP		···			···			···				
		⋮			⋮			⋮			⋮		⋮	⋮	⋮
		MM	P1		···			···			···				
			⋮		⋱			⋱			⋱		⋮		
			PP		···			···			···				
		BG			···			···			···				
		BP			···			···			···				
		BM			···			···			···				

ED: Element Disconnected
EP: Element Plugged-In
EG: Element Good
BG: Bus Good
BP: Bus Used by Processor
BM: Bus Used by Memory

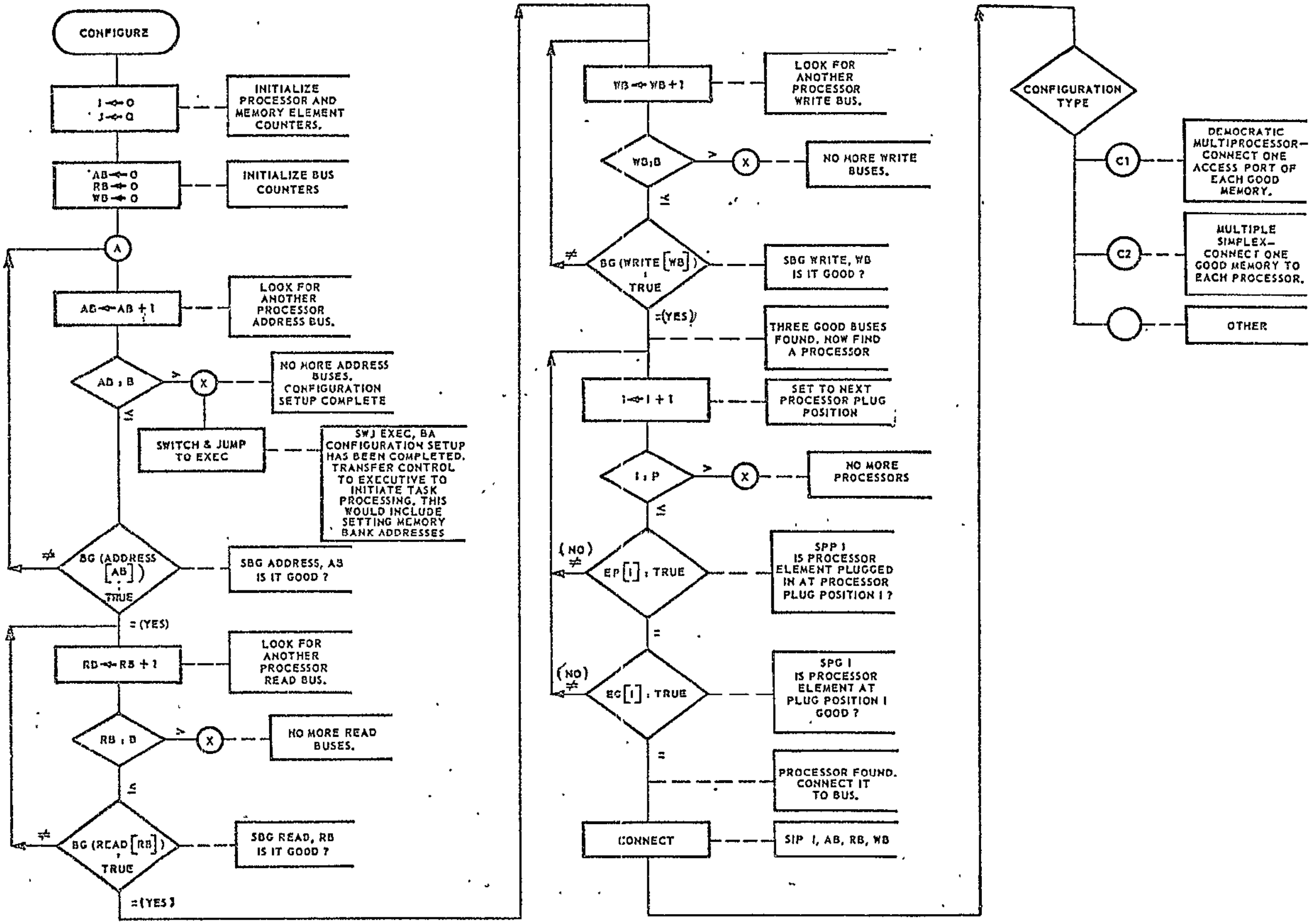

CONFIGURE
I ← O
J ← O
INITIALIZE PROCESSOR AND MEMORY ELEMENT COUNTERS.
AB ← O
RB ← O
WB ← O
INITIALIZE BUS COUNTERS
A
AB ← AB + 1
LOOK FOR ANOTHER PROCESSOR ADDRESS BUS.
AB : B
>
X
NO MORE ADDRESS BUSES. CONFIGURATION SETUP COMPLETE
≤
SWITCH & JUMP TO EXEC
SWJ EXEC, BA CONFIGURATION SETUP HAS BEEN COMPLETED. TRANSFER CONTROL TO EXECUTIVE TO INITIATE TASK PROCESSING. THIS WOULD INCLUDE SETTING MEMORY BANK ADDRESSES
≠
BG (ADDRESS [AB]) : TRUE
SBG ADDRESS, AB IS IT GOOD ?
= (YES)
RB ← RB + 1
LOOK FOR ANOTHER PROCESSOR READ BUS.
RB : B
>
X
NO MORE READ BUSES.
≤
≠
BG (READ [RB]) : TRUE
SBG READ, RB IS IT GOOD ?
= (YES)
WB ← WB + 1
LOOK FOR ANOTHER PROCESSOR WRITE BUS.
WB : B
>
X
NO MORE WRITE BUSES.
≤
≠
BG (WRITE [WB]) : TRUE
SBG WRITE, WB IS IT GOOD ?
= (YES)
THREE GOOD BUSES FOUND, NOW FIND A PROCESSOR
I ← I + 1
SET TO NEXT PROCESSOR PLUG POSITION
I : P
>
X
NO MORE PROCESSORS
≤
(NO) ≠
EP [I] : TRUE
SPP I IS PROCESSOR ELEMENT PLUGGED IN AT PROCESSOR PLUG POSITION I ?
=
(NO) ≠
EG [I] : TRUE
SPG I IS PROCESSOR ELEMENT AT PLUG POSITION I GOOD ?
=
PROCESSOR FOUND. CONNECT IT TO BUS.
CONNECT
SIP I, AB, RB, WB
CONFIGURATION TYPE
C1
DEMOCRATIC MULTIPROCESSOR– CONNECT ONE ACCESS PORT OF EACH GOOD MEMORY.
C2
MULTIPLE SIMPLEX– CONNECT ONE GOOD MEMORY TO EACH PROCESSOR.
OTHER

FIGURE 13

DMP CONFIGURATION SETUP

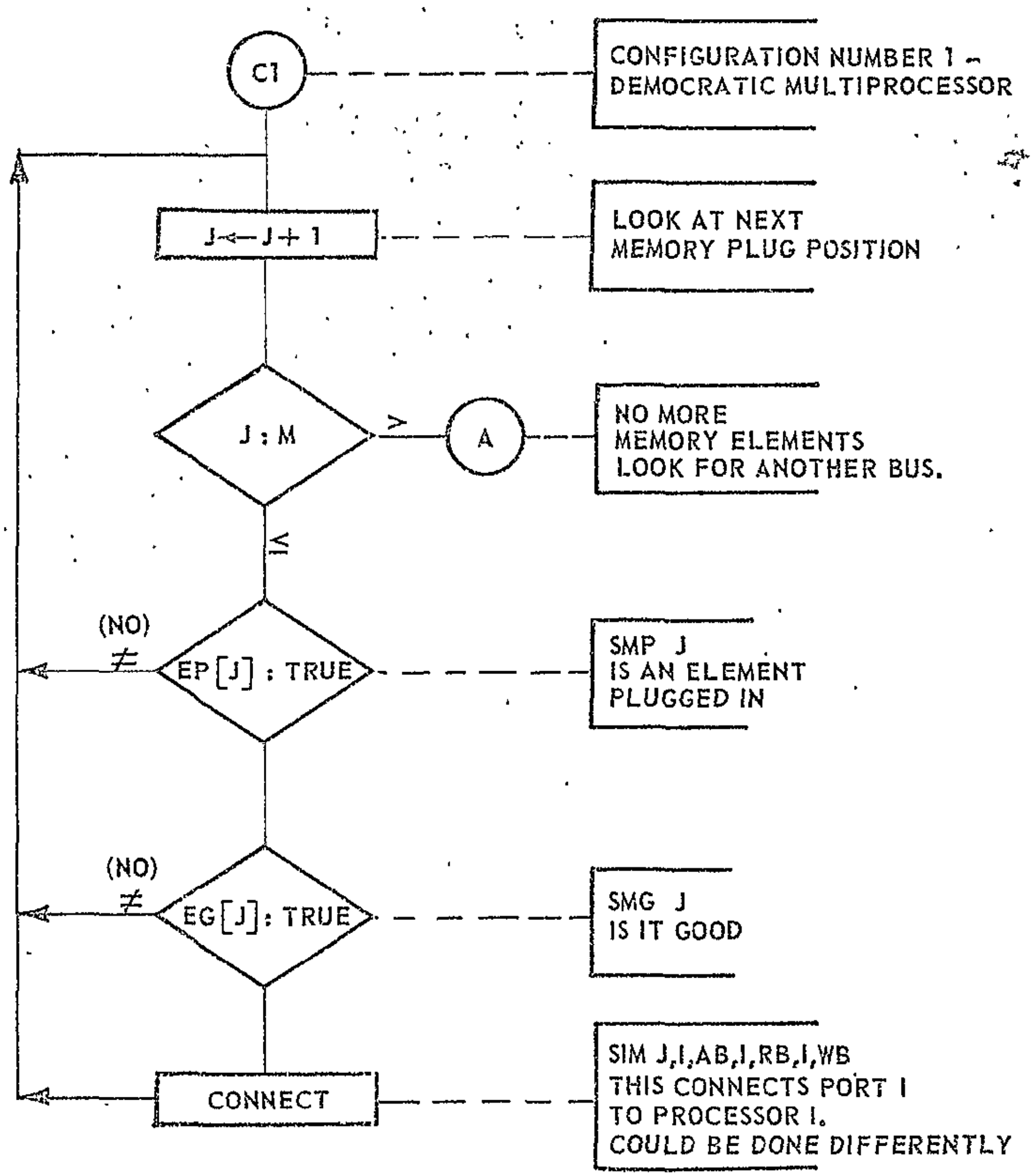


FIGURE 14
DEMOCRATIC MULTIPROCESSOR

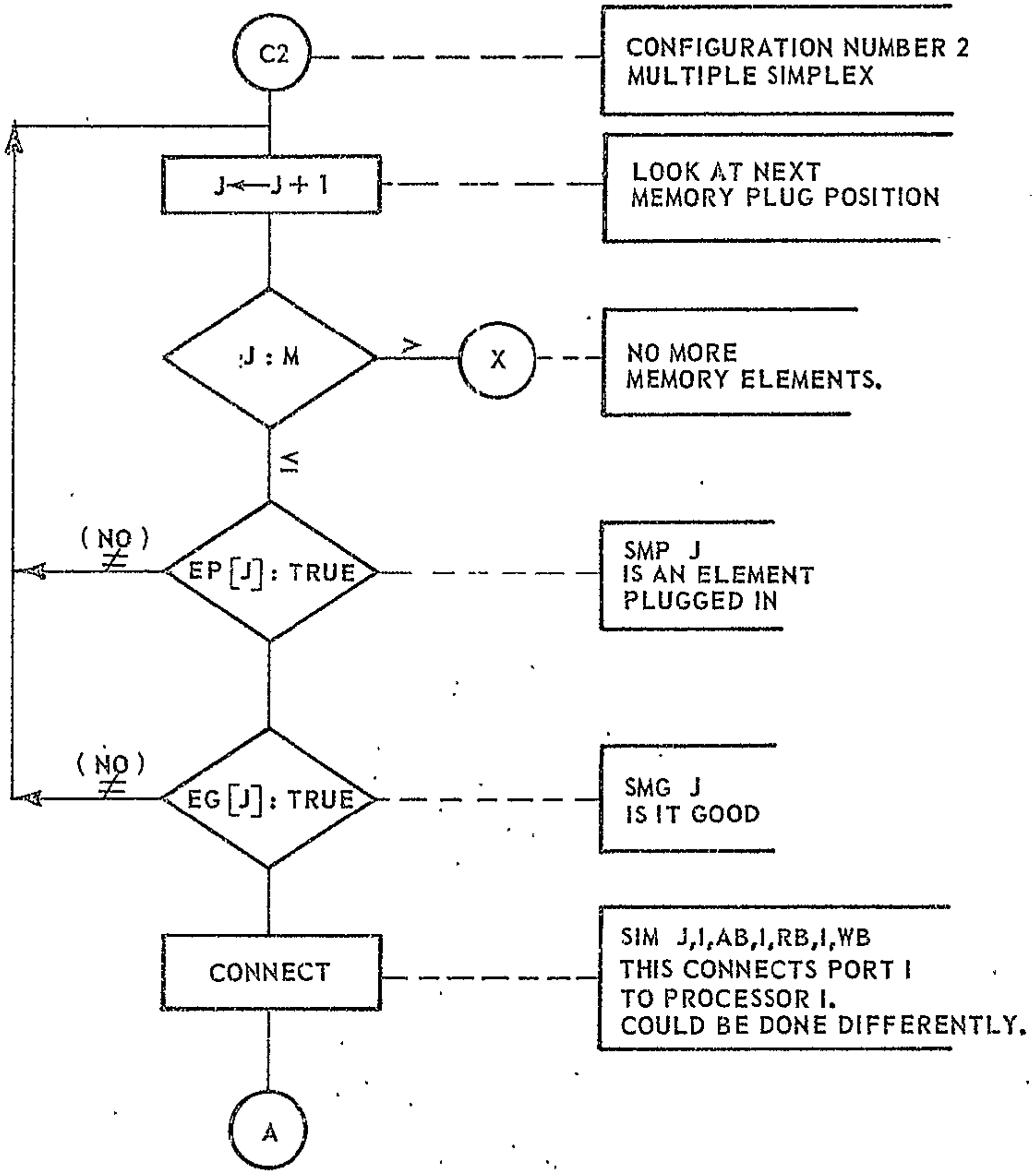


FIGURE 15

MULTIPLE SIMPLEX

When there are no additional buses or processors that can be connected to act as part of the configuration, switching is invoked and control transfers to an executive routine to initiate task processing. This procedure, which would include setting the bank addresses for all memory elements in use, is not illustrated because of the variety of operation possibilities.

Figure 16 illustrates memory spare switching making use of the derived instructions. It is assumed, as indicated in the figure, that the number of the bad memory plug position and the bank address of the bad memory element is known at the time of transfer to the memory spare switching algorithm.

Figure 17 illustrates processor spare switching wherein the number of the bad processor plug position is known. As noted in the figure, it might be desirable, in the case where no spare processor is available, to disconnect each memory element's access port from bus triples associated with the bad processor.

c. Instructions. The instructions specified in table 3 are a result of analysis of the requirements associated with the algorithms for configuration control and spares switching illustrated in figures 13 through 17. For the most part, the table contains sufficient explanation to illustrate the functional nature of each instruction. The table indicates which map (action or setup) is affected by each instruction where appropriate.

It should be noted that if a strict (hard wired) association is made among bus triples (i.e. selection and distribution switch elements are "ganged" together such that a switching action to connect an element to, for instance, an address bus will also connect the element to specific read and write buses), considerable simplification in the instructions and arguments is possible. Two of the columns of bus types in table 2 could be eliminated resulting in a single column for each bus triple. If, in addition, a hard wired association is made between processor elements and memory access ports, an additional level of simplification would be possible. These simplifications are not shown in this report; they are considered to be subject matter for future trade studies aimed toward achieving a good balance between flexibility and complexity.

4. <u>TMR Spares Switching Controls</u>. In order to establish a rationale for spares switching while operating in the TMR mode, a method for determining which element failed and for signaling a spares switching routine is outlined below. This method and the associated signaling mechanism then becomes the basis for spares switching. Majority voting and disagree detection is first reviewed. Then a method is outlined wherein the results of detecting a disagreement is used to make a decision as to whether certain specified actions are to be taken. Indicators set by the decision logic will then be used to control spares switching.

INPUT: J = BAD MEMORY PLUG POSITION.
BA = BANK ADDRESS OF BAD MEMORY.

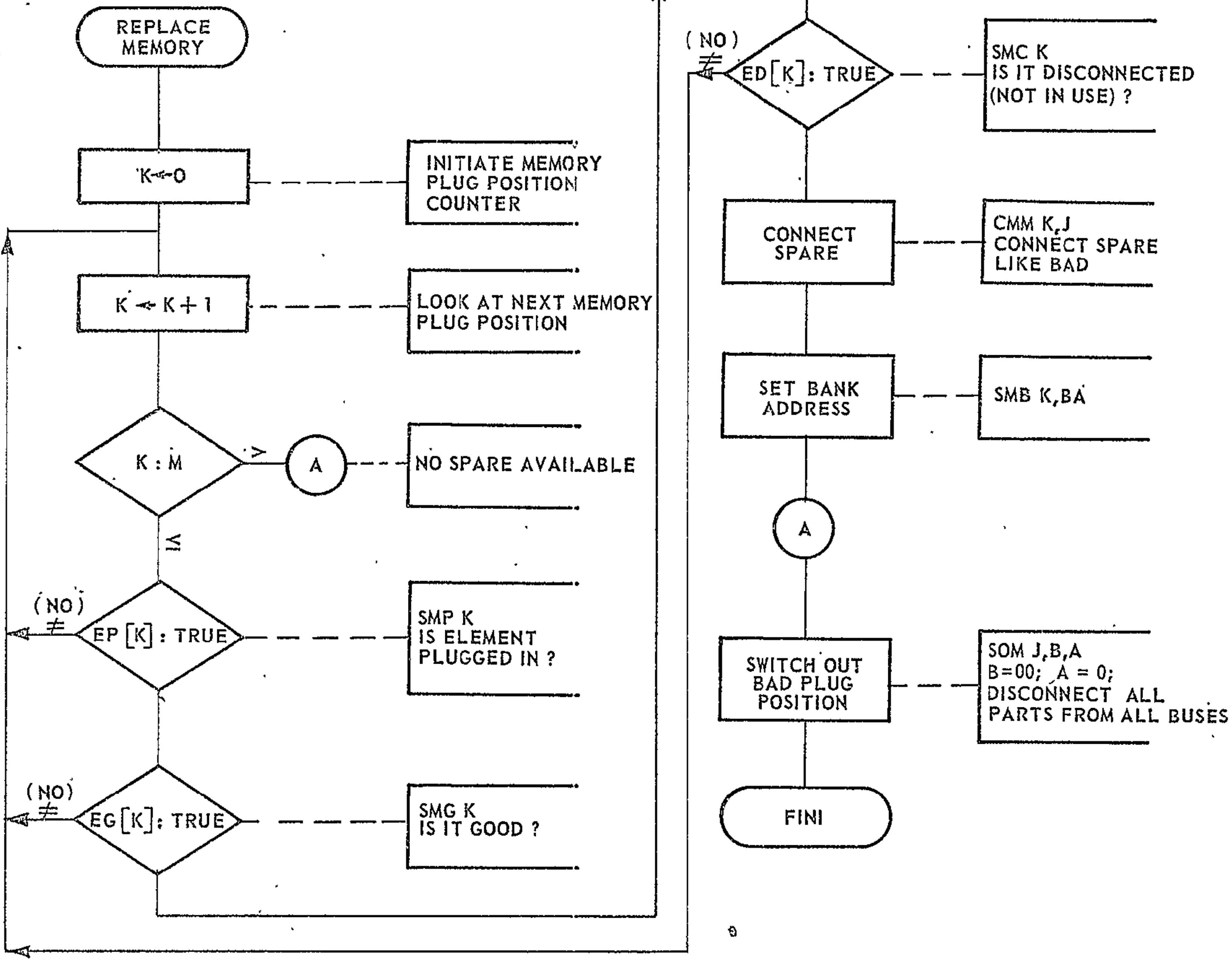
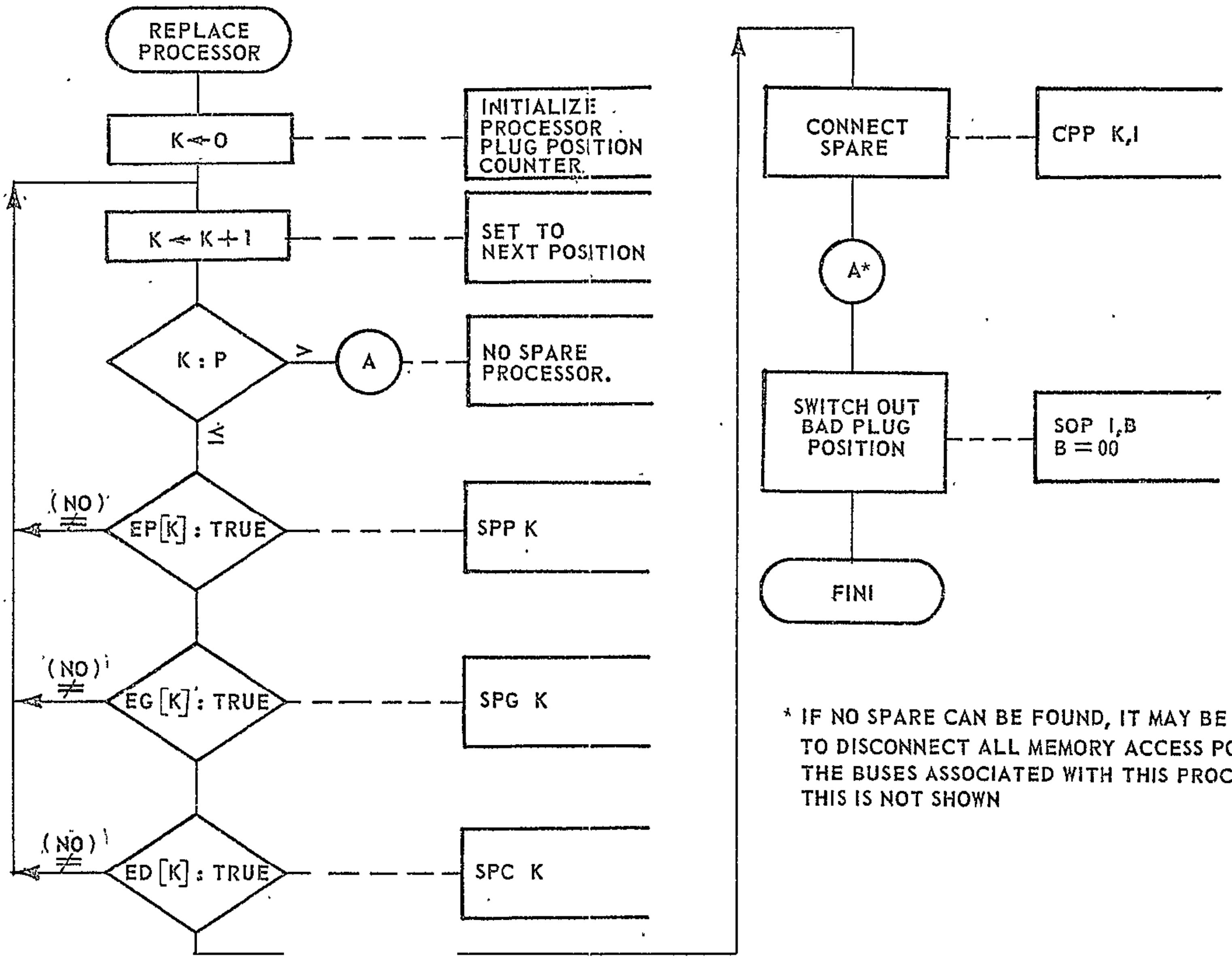


FIGURE 16

MEMORY SPARE SWITCHING

INPUT: I = BAD PROCESSOR PLUG POSITION



* IF NO SPARE CAN BE FOUND, IT MAY BE DESIRABLE TO DISCONNECT ALL MEMORY ACCESS PORTS FROM THE BUSES ASSOCIATED WITH THIS PROCESSOR, THIS IS NOT SHOWN

FIGURE 17

PROCESSOR SPARE SWITCHING

TABLE 3. DMP CONFIGURATION CONTROL INSTRUCTIONS

Map	Instruction		Meaning
	Op Code	Arguments	
SM	SIP	PP, AB, RB, WB	Switch-in processor element located in plug position PP connecting it to address bus number AB, read bus number RB, and write bus number WB.
SM	SIM	MM, AN, AB, RN, RB, WN, WB	Switch-in memory element located in plug position MM connecting its address port number AN to address bus number AB, its read port number RN to read bus number RB, and its write port number WN to write bus number WB.

AM	SOP	PP
		B
			Switch-out processor element in plug position PP disconnecting it from
			if B = 00, all buses,
			if B = 01, address bus only,
			if B = 10, read bus only, or
			if B = 11, write bus only.

AM	SOM	MM B A	Switch-out memory element in plug position MM disconnecting it from if B = 00, all buses, if B = 01, address bus only, if B = 10, read bus only, or if B = 11, write bus only, at its access port number, if A = 0; all ports, otherwise,only that port specified by A ≠ 0.
	SMB	MM BA	Set the bank address of the memory element located in plug position MM to contain BA.
AM	SPC	PP	Sense processor-connect status; if processor plug position PP is connected to a set of buses (i.e., ED [PP]* is false), skip the next instruction. (When a processor plug position is vacant, it is assumed that it is disconnected from all buses. The disconnect operation should occur automatically upon manual unplugging or under program control.)

*The functional notation X [Y] is read "property X of thing Y" or, simply, "X of Y."

TABLE 3. DMP CONFIGURATION CONTROL INSTRUCTIONS (Continued)

Map	Instruction		Meaning
	Op Code	Arguments	
AM	SPP	PP	Sense processor-plugged-in status; if processor plug position PP has an element plugged in (i. e., EP [PP] is true), skip the next instruction.
AM	SPG	PP	Sense processor-good status; if processor plug position PP has a good element plugged in (i.e., EG [PP] is true), skip the next instruction.
AM	SMC	MM	Sense memory-connect status; similar to SPC.
AM	SMP	MM	Sense memory-plugged-in status; similar to SPP.
AM	SMG	MM	Sense memory-good status; similar to SPG.

AM	SBG	B
		BN
			Sense bus-good status; if bus number BN in bus group
			if B = 00, all bus groups,
			if B = 01, address only,
			if B = 10, read only,
			if B = 11, write only
			is marked good (i. e., BG [BN] is true), skip the next instruction.
			Note: If B = 00, the test is made on $\text{BG}[\text{BN}]_{\text{address}}$, $\text{BG}[\text{BN}]_{\text{read}}$, and $\text{BG}[\text{BN}]_{\text{write}}$. In other words, bus BN in all three groups must be good to cause a skip.

AM	CMM	M1 M2	Connect Memory-Memory; connect memory plug position M1 as indicated in the Action Map for memory plug position M2.
AM	CPP	P1 P2	Connect Processor-Processor; connect processor plug position P1 as indicated in the Action Map for processor plug position P2.
SM/ AM	SWJ	A BA	Switch and jump; transfer Setup Map information to the switch control logic for switching and save it in the Action Map. Status indicators are set in AM to show associated connections and SM is cleared. Control of all connected processors is transferred simultaneously to memory location A of the memory element whose bank address is BA.

a. Vote Logic and Disagree Detection. Figure 18 depicts a logic network that will output a majority vote (upper portion of figure) based upon three inputs denoted as c1, c2, and c3 for the i-th bit. Inset A shows a block diagram symbol, denoted as V, that produces an n-bit majority, M, given the three n-bit channels of input C1, C2, and C3.

Shown in the lower portion of figure 18 is a network that will result in a true value for that channel presumed to be in error when a disagreement exists. The error for the i-th bit is designated in the diagram for the j-th channel as e_j^i. Inset B shows a block diagram symbol, denoted VD, for the combined majority vote and disagree detection logic.

The disagree detection logic could be based upon the use of exclusive-or components, of course. However, while the number of gates is comparable, an additional level is required if an exclusive-or is considered to be a higher level logic component based upon "and" and "or" gates.

The network of figure 18 is based upon the truth table and Boolean equations shown in table 4 for majority vote and disagree detection.

b. Decision Logic. Based upon the n-bits specifying a disagree for each of the three channels, a decision can be made as to whether switching should be invoked. Clearly, there are numerous ways to make such a decision. A particular way is discussed here as a basis for control requirements.

Referring to the disagree detection logic of figure 18, we note that if $e_j^i = 1$, then $e_k^i = 0$ for $k \neq j$, unless the logic has failed. Assuming that the logic is good, and letting

$$E_j = e_j^1 + e_j^2 + \ldots + e_j^n, \text{ for } j = 1, 2, 3,$$

then $E_j = E_k = 1$ for $j \neq k$, implies at least two different bit positions in channels j and k have failed. Further, if $E_1 = E_2 = E_3 = 1$, then at least three different bit positions in channels 1, 2, and 3 have failed. Either of these conditions means that two or more of the three participating channels have failed; that is, they cannot be reliably operated in simplex or dual-redundant modes. Of course the errors could be transient. In any case, a mode change while such failures are prevalent is contraindicated and a variable, L, is therefore defined to signal this condition. L might be used to lock the system in a TMR mode. Another valid approach might be to use the variable L to start a counter to count consecutive double or triple failures for the purpose of separating transient from solid failures. A similar result based upon a different approach might be to use L to set a clock for, say, one second to allow a burst of transient

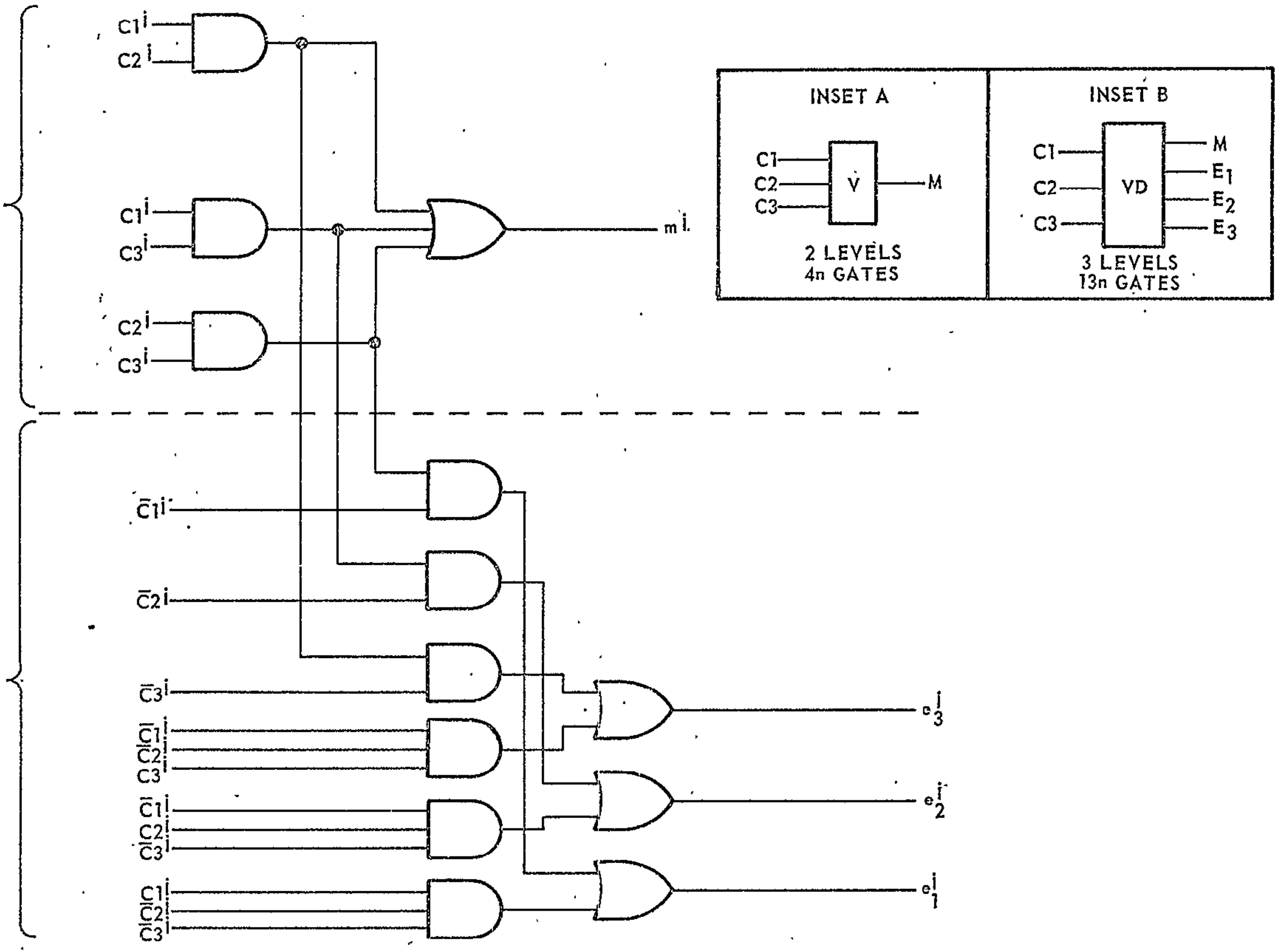


FIGURE 18

MAJORITY VOTE AND DISAGREE DETECTION LOGIC

TABLE 4. TRUTH TABLE AND BOOLEAN EQUATIONS FOR MAJORITY VOTE/DISAGREE DETECTION

TRUTH TABLE									BOOLEAN EQUATIONS
VARIABLES	STATES								
	0	1	2	3	4	5	6	7	
C1	0	1	0	1	0	1	0	1	
C2	0	0	1	1	0	0	1	1	
C3	0	0	0	0	1	1	1	1	
m	0	0	0	1	0	1	1	1	$C1 \cdot C2 + C1 \cdot C3 + C2 \cdot C3$
e1	0	1	0	0	0	0	1	0	$C1 \cdot \overline{C2} \cdot \overline{C3} + \overline{C1} \cdot C2 \cdot C3$
e2	0	0	1	0	0	1	0	0	$\overline{C1} \cdot C2 \cdot \overline{C3} + C1 \cdot \overline{C2} \cdot C3$
e3	0	0	0	1	1	0	0	0	$C1 \cdot C2 \cdot \overline{C3} + \overline{C1} \cdot \overline{C2} \cdot C3$

C1 – Channel 1 input
C2 – Channel 2 input
C3 – Channel 3 input
m – Majority
e1 – Error in channel 1
e2 – Error in channel 2
e3 – Error in channel 3

errors related to physical phenomena to subside. If L is true after the clock expires, the errors might be considered to be solid.

Table 5 gives a truth table and Boolean equations for the decision network shown in figure 19. The inputs consist of n-bit error indicators from each of the three channels. Outputs are L, which is simply a majority vote on the OR'ed error bits, S, an indicator that an error in one and only one of the channels has occurred, and R_1, R_2, and R_3, which are used to designate the channel in error. All outputs are single bit.

c. Combined Vote, Decision, and Switch Control. Combining the networks of figure 18 and 19, figure 20 depicts the components sufficient to perform TMR voting, disagreeing channel detection, switching decision logic, and element performance status indicators that can be used by an executive routine or digital logic to switch spares or influence configuration mode switching. This network is denoted in the inset as VDSC where three channels of data originating at a system element, such as the SUMC memory address register, are shown entering on the left side and exiting in majority vote form on the right side for distribution to the next system element, such as memory. Single bit indicators as discussed above are shown leaving the top of the box. Note that the network of figure 20 makes use of both the majority vote alone (V) and the majority vote with disagree detection (VD).

While it is not the purpose of this report to develop a philosophy for spares switching decisions, a brief discussion is in order. Switching could be invoked on the basis of either S or L. Since S indicates single channel errors while L indicates multiple channel errors, they should be treated differently in any decision concerning switching. In contrast to the procedures outlined previously, S could be used to start a count of consecutive errors on a given channel. Switching the offending element on a channel might then be based upon the occurrence of consecutive errors rather than a single error. Provided good spare elements are available, intuitively it seems that no harm is done by leaving a failing element in the TMR configuration since majority voting cancels most errors. On the other hand, concern over the system's viability increases when L is true. L may therefore be considered as the controlling variable regarding switching. If this is the case, it is clear that at least two failing elements should be replaced; it is not clear what action should be taken when all three elements are failing. Obviously, if sufficient spares are available, a complete change of all elements could be made. However, it is suspected that L will be "on" either as a result of transient failures throughout a given mission or as a result of a general decay in system performance in the terminal stages of a long-duration mission. While transient failures will, more or less, take care of themselves, a radical methodology, such as "thrashing switch action," will be required in the latter case. Thrashing

TABLE 5. DECISION NETWORK TRUTH TABLE AND BOOLEAN EQUATIONS

TRUTH TABLE									BOOLEAN EQUATIONS
VARIABLES	STATES								
	0	1	2	3	4	5	6	7	
E_1	0	1	0	1	0	1	0	1	
E_2	0	0	1	1	0	0	1	1	
E_3	0	0	0	0	1	1	1	1	
S	0	1	1	0	1	0	0	0	$R_1 + R_2 + R_3$
R_1	0	1	0	0	0	0	0	0	$E_1 \cdot \overline{E_2} \cdot \overline{E_3}$
R_2	0	0	1	0	0	0	0	0	$\overline{E_1} \cdot E_2 \cdot \overline{E_3}$
R_3	0	0	0	0	1	0	0	0	$\overline{E_1} \cdot \overline{E_2} \cdot E_3$
L	0	0	0	1	0	1	1	1	$E_1 \cdot E_2 + E_1 \cdot E_3 + E_2 \cdot E_3$

E_1 - Or of n-bits denoting errors in channel 1.
E_2 - Or of n-bits denoting errors in channel 2.
E_3 - Or of n-bits denoting errors in channel 3.
S - Signal or Switch.
R_1 - Channel 1 Error.
R_2 - Channel 2 Error.
R_3 - Channel 3 Error.
L - Lock. Majority of channels have errors.

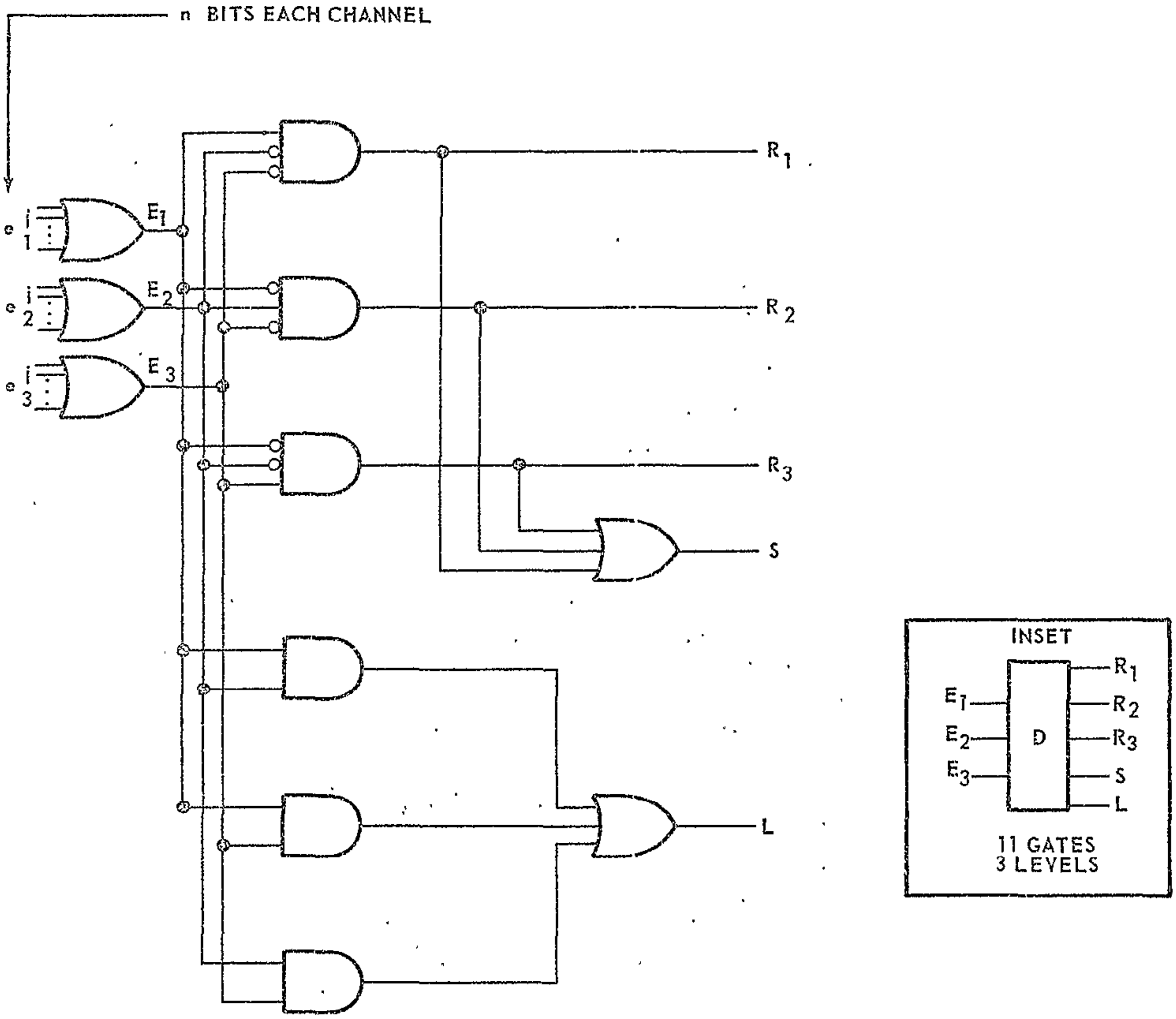


FIGURE 19

DECISION LOGIC

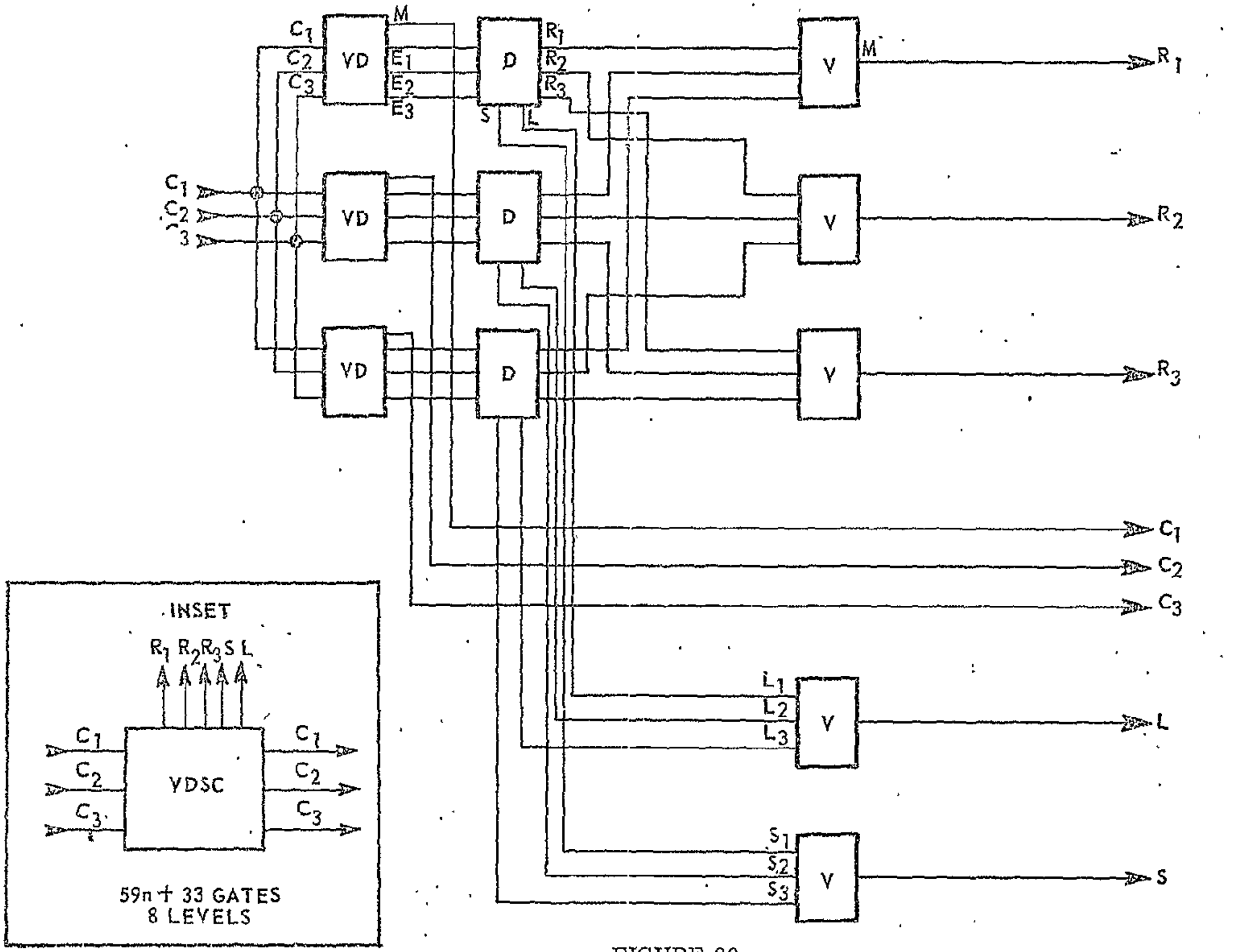


FIGURE 20

VOTE, DECISION AND SWITCH CONTROL LOGIC

switch action refers to an approach based upon treating previously failed and replaced elements as valid spares. For example, if a given jump operation cannot be performed by any of the three SUMCs comprising the TMR configuration, it may be possible to find two previously failed spares that can perform the jump. Or, one previously failed spare might be found that could correctly agree with one of the others. After the jump operation has been performed, it may also be that the next operation cannot be performed by the pair of processors that accomplished the jump operation. In this case, spares comprised of the same components which were previously switched out may be switched back in and successfully accomplish the next operation.

d. Configuration Control. All of the elements shown in the configuration control and mode switching diagram of figure 12 have been discussed and approaches to functional performance have been outlined from a feasibility point of view. The system map of table 2 is expanded to include the VDSC elements required for TMR mode operations in table 6 which outlines the additional system map entry formats.

Three additional columns in the supplement to the system map are included to contain condition indicators associated with R1, R2, R3, S, and L as developed in figure 20. These condition indicators apply, of course, only to the input triples T1, T2, and T3 as shown in table 6. The intersection of column R with row T1 represents R1, the intersection of column R with row T2 represents R2, etc. When a R indicator is turned on by a VDSC, the action map can be searched on the corresponding VDSC bus column to find the element connected to the same bus. This element is the offending element and is therefore a potential candidate for replacement by a spare. The searching operation to identify the connected element would be simplified if the action map has certain associative features that could provide the identification of the element connected to a given bus upon request.

Table 6 can be compressed in much the same way as was table 1. Table 7 represents a final system map including the compressed table 6 information. Notice that it is necessary to retain R, S, and L indicators in uncompressed form.

Based upon the TMR mode spares concepts discussed thus far, figure 21 outlines a program flow for setting-up a TMR configuration and switching accordingly. Table 8 provides a matrix showing the resulting connections. (This matric could be stored for use as an aid in spares switching eliminating the bus search discussed above.)

The algorithm of figure 21 recognizes that there may be a lack of sufficient buses or elements of various types required to establish a TMR configuration. Therefore, tests and flow control branches are depicted that

TABLE 6. TMR CONFIGURATION SYSTEM MAP SUPPLEMENT

				BUS NUMBERS					
				ADDRESS	READ	WRITE			
				B1 B2 ··· BB	B1 B2 ··· BB	B1 B2 ··· BB	R	S	L
PROCESSOR-TO-MEMORY VDSC PLUG POSITIONS	ADDRESS	IN	T1		▨	▨			
			T2		▨	▨			
			T3		▨	▨			
		OUT	T1		▨	▨	▨	▨	▨
			T2		▨	▨	▨	▨	▨
			T3		▨	▨	▨	▨	▨
	READ	IN	T1	▨		▨			
			T2	▨		▨			
			T3	▨		▨			
		OUT	T1	▨		▨	▨	▨	▨
			T2	▨		▨	▨	▨	▨
			T3	▨		▨	▨	▨	▨
	WRITE	IN	T1	▨	▨				
			T2	▨	▨				
			T3	▨	▨				
		OUT	T1	▨	▨				
			T2	▨	▨				
			T3	▨	▨				

TABLE 7. FINAL SYSTEM MAP

				BUS NUMBERS											
				ADDRESS			READ			WRITE					
				B1	...	BB	B1	...	BB	B1	...	BB	ED	EP	EG
ELEMENT PLUG POSITIONS	SUMC		P1		...			...			...				
			⋮		⋱			⋱			⋱		⋮	⋮	⋮
			PP		...			...			...				
	MEMORY	M1	P1		...			...			...				
			⋮		⋱			⋱			⋱		⋮		
			PP		...			...			...				
		⋮	⋮		⋮			⋮			⋮		⋮	⋮	⋮
		MM	P1		...			...			...				
			⋮		⋱			⋱			⋱		⋮		
			PP		...			...			...				
	VDSC(S/M)	IN	T1												
			T2												
			T3												
		OUT	T1												
			T2												
			T3												
			BG		...			...			...				
			BP		...			...			...				
			BM		...			...			...				

		ADDRESS			READ			WRITE		
		R	S	L	R	S	L	R	S	L
VDSC(S/M) IN	T1									
	T2									
	T3									

ED:	Element Disconnected
EP:	Element Plugged-In
EG:	Element Good
BG:	Bus Good
BP:	Bus Used by Processor
BM:	Bus Used by Memory

R:	Error(s) in TMR channel T1, T2, or T3
S:	Single Channel Error(s)
L:	Multiple Channel Errors

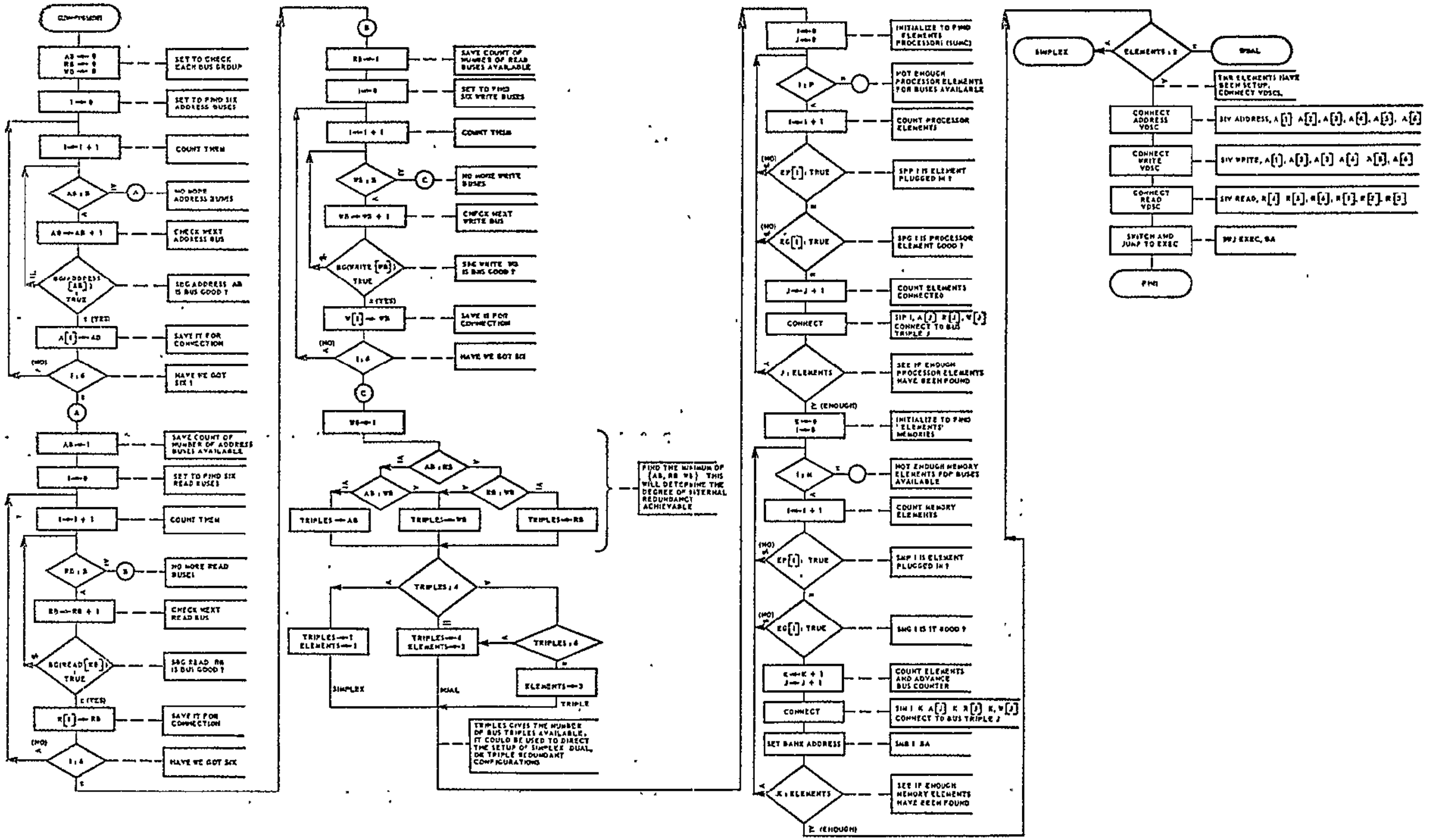

CONFIGURATION
AB ← 0
RB ← 0
WB ← 0
SET TO CHECK EACH BUS GROUP
I ← 0
SET TO FIND SIX ADDRESS BUSES
I ← I + 1
COUNT THEM
AB : 8
≥
A
<
NO MORE ADDRESS BUSES
AB ← AB + 1
CHECK NEXT ADDRESS BUS
BG(ADDRESS [AB]) : TRUE
≠
= (YES)
SBG ADDRESS AB IS BUS GOOD ?
A[I] ← AB
SAVE IT FOR CONNECTION
I : 6
(NO) <
=
HAVE WE GOT SIX ?
A
AB ← I
SAVE COUNT OF NUMBER OF ADDRESS BUSES AVAILABLE
I ← 0
SET TO FIND SIX READ BUSES
I ← I + 1
COUNT THEM
RB : 8
≥
B
<
NO MORE READ BUSES
RB ← RB + 1
CHECK NEXT READ BUS
BG(READ [RB]) : TRUE
≠
= (YES)
SBG READ RB IS BUS GOOD ?
R[I] ← RB
SAVE IT FOR CONNECTION
I : 6
(NO) <
=
HAVE WE GOT SIX
B
RB ← I
SAVE COUNT OF NUMBER OF READ BUSES AVAILABLE
I ← 0
SET TO FIND SIX WRITE BUSES
I ← I + 1
COUNT THEM
WB : 8
≥
C
<
NO MORE WRITE BUSES
WB ← WB + 1
CHECK NEXT WRITE BUS
BG(WRITE [WB]) : TRUE
≠
= (YES)
SBG WRITE WB IS BUS GOOD ?
W[I] ← WB
SAVE IT FOR CONNECTION
I : 6
(NO) <
HAVE WE GOT SIX
C
WB ← I
AB : RB
≤
>
AB : WB
RB : WB
TRIPLES ← AB
TRIPLES ← WB
TRIPLES ← RB
FIND THE MINIMUM OF {AB, RB, WB} THIS WILL DETERMINE THE DEGREE OF INTERNAL REDUNDANCY ACHIEVABLE
TRIPLES : 4
<
>
=
TRIPLES ← 1
ELEMENTS ← 1
TRIPLES ← 4
ELEMENTS ← 2
TRIPLES : 6
ELEMENTS ← 3
SIMPLEX
DUAL
TRIPLE
TRIPLES GIVES THE NUMBER OF BUS TRIPLES AVAILABLE. IT COULD BE USED TO DIRECT THE SETUP OF SIMPLEX, DUAL, OR TRIPLE REDUNDANT CONFIGURATIONS
I ← 0
J ← 0
INITIALIZE TO FIND ELEMENTS PROCESSORS (SUMC)
I : P
=
NOT ENOUGH PROCESSOR ELEMENTS FOR BUSES AVAILABLE
I ← I + 1
COUNT PROCESSOR ELEMENTS
EP[I] : TRUE
(NO)
SPP I IS ELEMENT PLUGGED IN ?
EG[I] : TRUE
SPG I IS PROCESSOR ELEMENT GOOD ?
J ← J + 1
COUNT ELEMENTS CONNECTED
CONNECT
SIP I, A[J] R[J], W[J] CONNECT TO BUS TRIPLE J
J : ELEMENTS
≥ (ENOUGH)
SEE IF ENOUGH PROCESSOR ELEMENTS HAVE BEEN FOUND
K ← 0
I ← 0
INITIALIZE TO FIND 'ELEMENTS' MEMORIES
I : M
NOT ENOUGH MEMORY ELEMENTS FOR BUSES AVAILABLE
I ← I + 1
COUNT MEMORY ELEMENTS
SMP I IS ELEMENT PLUGGED IN ?
SMG I IS IT GOOD ?
K ← K + 1
J ← J + 1
COUNT ELEMENTS AND ADVANCE BUS COUNTER
SIM I K A[J] K R[J] K, W[J] CONNECT TO BUS TRIPLE J
SET BANK ADDRESS
SMB I BA
K : ELEMENTS
SEE IF ENOUGH MEMORY ELEMENTS HAVE BEEN FOUND
SIMPLEX
ELEMENTS : 2
DUAL
TMR ELEMENTS HAVE BEEN SETUP, CONNECT VDSCS.
CONNECT ADDRESS VDSC
SIV ADDRESS, A[1] A[2], A[3], A[4], A[5], A[6]
CONNECT WRITE VDSC
SIV WRITE, A[1], A[2], A[3] A[4] A[5], A[6]
CONNECT READ VDSC
SIV READ, R[1] R[2], R[4], R[1], R[2], R[3]
SWITCH AND JUMP TO EXEC
SWJ EXEC, BA
FINI

FIGURE 21
TMR CONFIGURATION SETUP

TABLE 8. ELEMENT AND VDSC CONNECTIONS TO BUSES RESULTING IN A TMR CONFIGURATION

			BUSES			
VDSC's			AB	RB	WB	ELEMENTS
ADDRESS/WRITE	IN	T1	A [1]		W [1]	P1
		T2	2		2	P2
		T3	3		3	P3
	OUT	T1	4		4	M1
		T2	5		5	M2
		T3	6		6	M3
READ	OUT	T1		R [1]		P1
		T2		2		P2
		T3		3		P3
	IN	T1		4		M1
		T2		5		M2
		T3		6		M3

AB:	Address buses
RB:	Read buses
WB:	Write buses
A:	Array of six address buses
R:	Array of six read buses
W:	Array of six write buses

indicate how an attempt to establish a TMR configuration can be made to result in the establishment of a dual modular redundant system or a simplex system which could provide time redundancy through multiple calculations. All details are not shown; however, there is sufficient flow logic to indicate the necessary steps.

If the identity of a failed SUMC or memory is known, the flow logic of figures 16 and 17 is sufficient to accomplish spares switching in the TMR mode with certain restrictions. Based on the memory spare switching scheme of figure 16, a TMR spare memory should be designed such that it will ignore access requests for as long as it takes to execute the SOM shown below label A. That is, after the spare memory has been connected and its bank address set, it should ignore access requests for a delay time equivalent to the execution time of the SOM. This allows the SOM instruction to be fetched from all TMR memories that are connected with the exception of the newly connected spare. If the delay is not incorporated in memory elements, it is clear that a problem arises due to the fact that two memory elements are connected to the same bus during execution of both the SMB and the SOM instruction of figure 16. A similar consideration must be given in the case of processor spare switching as shown in figure 17. That is, after the CPP instruction which connects the spare SUMC has been completed, the SUMC must delay fetching instructions for the time required to execute the SOP instruction shown below label A of figure 17.

The status indicators R, S, and L are associated only with the action map as is the case with all other such indicators. On the other hand, the bus connections for the VDSCs are associated with both the setup map and the action map. In order to show the use of the failure indicators R, S, and L, figure 22 is provided as an illustration of program controlled response to the setting (interrupt?) of an S or L. The algorithm shown tests the various indicators to determine which elements are indicated as having failed. These elements are replaced through the use of spares switching procedures outlined in figures 16 and 17. The information contained in table 8 may be referenced with regard to figure 22 in order to identify element plug positions and perform the indicated replacements.

e. Instructions. As a result of the considerations outlined in figures 21 and 22 and the additions to the system map, the instructions depicted in table 9 are suggested. They are sufficient to accomplish TMR configuration control and spares switching. The format of table 9 is the same as that of table 3. Recalling that figure 2 indicated the possibility of a special bank address register for processor elements, it can now be stated that such a register is not necessary on the basis of sufficient configuration control capability without it.

INPUT: BANK ADDRESS (BA) OF TMR MEMORIES.

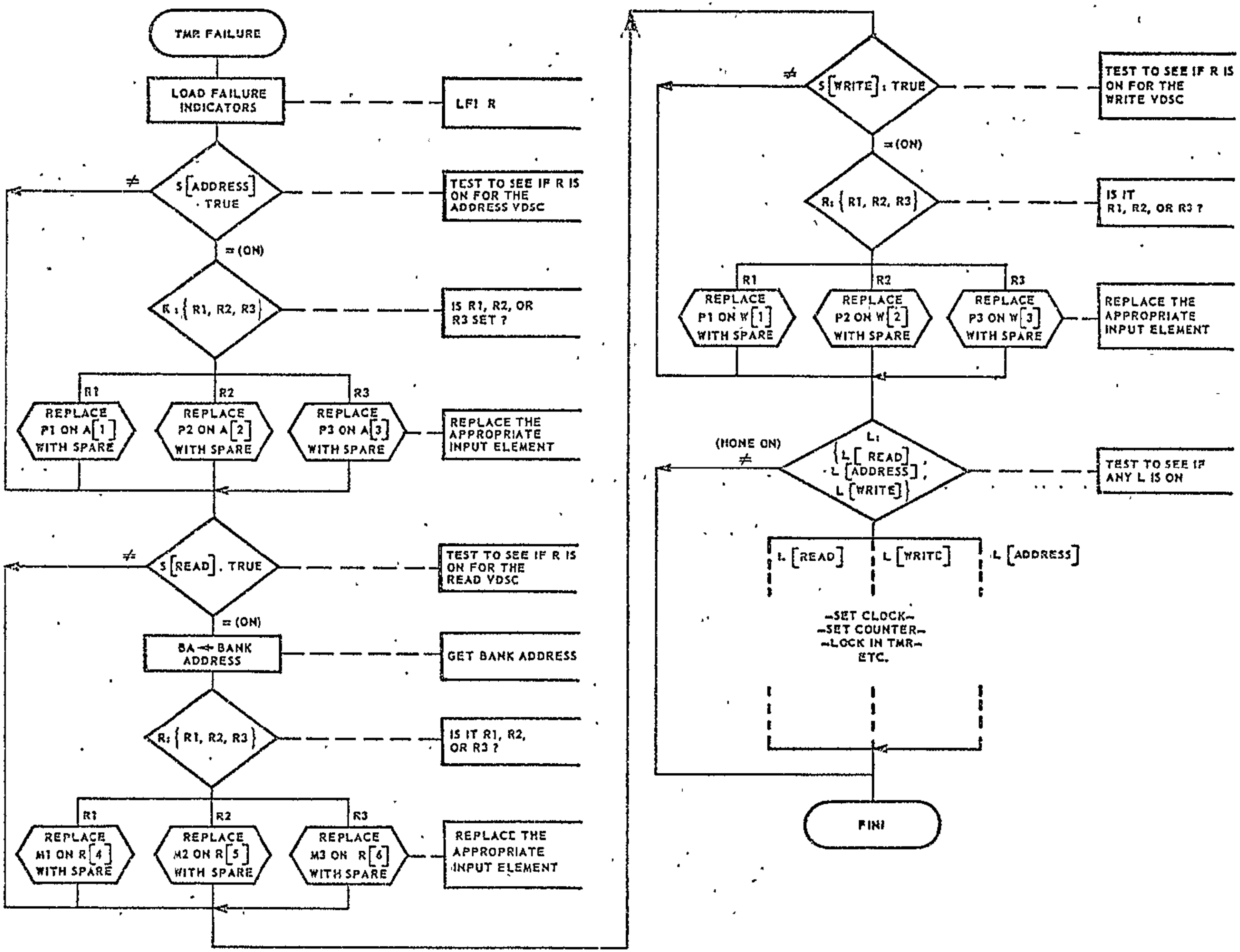

TMR FAILURE
LOAD FAILURE INDICATORS
LFI R
S [ADDRESS] · TRUE
≠
= (ON)
TEST TO SEE IF R IS ON FOR THE ADDRESS VDSC
R : {R1, R2, R3}
IS R1, R2, OR R3 SET ?
R1
R2
R3
REPLACE P1 ON A [1] WITH SPARE
REPLACE P2 ON A [2] WITH SPARE
REPLACE P3 ON A [3] WITH SPARE
REPLACE THE APPROPRIATE INPUT ELEMENT
S [READ] . TRUE
TEST TO SEE IF R IS ON FOR THE READ VDSC
BA ← BANK ADDRESS
GET BANK ADDRESS
R: {R1, R2, R3}
IS IT R1, R2, OR R3 ?
REPLACE M1 ON R [4] WITH SPARE
REPLACE M2 ON R [5] WITH SPARE
REPLACE M3 ON R [6] WITH SPARE
REPLACE THE APPROPRIATE INPUT ELEMENT
S [WRITE] : TRUE
TEST TO SEE IF R IS ON FOR THE WRITE VDSC
R: {R1, R2, R3}
IS IT R1, R2, OR R3 ?
REPLACE P1 ON W [1] WITH SPARE
REPLACE P2 ON W [2] WITH SPARE
REPLACE P3 ON W [3] WITH SPARE
REPLACE THE APPROPRIATE INPUT ELEMENT
(NONE ON)
L: {L [READ], L [ADDRESS], L [WRITE]}
TEST TO SEE IF ANY L IS ON
L [READ]
L [WRITE]
L [ADDRESS]
–SET CLOCK–
–SET COUNTER–
–LOCK IN TMR–
ETC.
FINI

FIGURE 22

USE OF TMR FAILURE INDICATORS

TABLE 9. TMR CONFIGURATION CONTROL INSTRUCTIONS

Map	Instruction		Meaning
	Op Code	Arguments	
SM	SIV	V IT1 IT2 IT3 OT1 OT2 OT3	Switch-in VDSC. The VDSC in plug position V is connected as follows: input T1 to bus IT1 input T2 to bus IT2 input T3 to bus IT3 output T1 to bus OT1 output T2 to bus OT2 output T3 to bus OT3.

AM	SOV	V
		T
			Switch-out VDSC. The VDSC in plug position V is disconnected from its buses as indicated by T:
			if T = 01, input T1 only,
			if T = 10, input T2 only,
			if T = 11, input T3 only,
			if T = 00, all input and output.

AM	LFI	R1	Load failure indicators. The R, S, and L status indicators for all bus sets are loaded into register R1 for program testing (no specific field format considered at this time).

C. Mode Switching

It is clear at this point that no special consideration needs to be given to the problem of changing the operational mode from DMP to TMR, or from TMR to DMP. It is assumed that control programs in either mode will be aware of requirements to change the configuration mode at the time such a change is necessary. This being the case, it is sufficient to transfer control to the DMP configuration setup procedure of figure 13 or the TMR configuration setup procedure of figure 21 at the appropriate time.

The simplicity of mode switching under the concepts outlined herein hinge primarily upon two factors:

- The use of two system maps, one for setup and the other for dynamic configuration recording and maintenance, and

- The switch and jump instruction which accomplishes the actual switching operations and transfers control of all connected processors.

In the case of TMR configuration setup, the jump and switch instruction must have the facility for synchronizing all processors beginning with the fetch operation for the first instruction executed in the TMR mode.

One of the major benefits of configuration control as outlined herein is that setup maps for various configurations can be predetermined and stored in main memory for future use. For instance, setup maps for configurations having a variety of reliability or throughput characteristics could be stored in such a way that a table search could be made to obtain the setup map for a configuration exhibiting a required reliability, power consumption, or throughput. A capability for structuring setup maps in-flight would also be of great value in overcoming low probability contingencies. Such a capability is easily implemented.

SECTION III. CONCLUSIONS

The report outlines schemes for configuration control, including spares switching and mode control. They are believed to represent a feasible approach to the dynamic control of a multi-element system including internally redundant modes of operation. Based upon the logic networks outlined, and instruction repertoire and flow diagrams developed, the following derived parameters characterize the overall method:

○	Number of DMP configuration setup instructions -	46
	Maximum DMP configuration setup time - 30P+9M+7** =	229 μsec*
○	Number of TMR configuration setup instructions -	127
	Maximum TMR configuration setup time - 18B+8(P+M)+203 =	415 μsec
○	Number of processor replacement instructions -	14
	Average processor replacement time (search 1/2) - 4.5P+5 =	27.5 μsec
○	Number of memory replacement instructions -	15
	Average memory replacement time (search 1/2) - 4.5M+6 =	42 μsec
○	Number of logic components to support switching -	
	Gates - B(3P[M+1] +[PM+7] [a+2w] +18) =	29,682
	Flip-flops - 3B(P[M+1] +6) =	918
○	Number of logic components to support TMR -	
	Gates - 59(a+2w)+99 =	6,117
	Flip-flops -	15
○	Estimate of number of logic levels -	
	Data: DMP -	4
	TMR -	12
	Switching: DMP - 9P(M+1) =	405
	TMR -	162

* 1 μsec average instruction time

**P = 5 SUMCs
M = 8 Memories
B = 6 Buses
a = 34 bits
w = 34 bits

o Number of special instructions to support configuration control -	18
o Minimum number of buses (processor/memory only)	
Address -	6
Read -	6
Write -	6

The approach taken in the development of the various controls is a somewhat generalized approach in that very few simplifications of a pragmatic nature are made. It is felt that such simplifications should be the result of appropriate trade studies which are beyond the scope of this effort. Many simplifications may be made in the instruction repertoire outlined. These are mentioned in the report in proper context. The instruction repertoire is not dependent upon the details of the mechanization of switching as depicted in the report. On the other hand, the repertoire is heavily dependent upon a flexible busing structure and upon the concept of a dual system map such as outlined herein. Should either of these features be significantly altered from the baseline approach, it is likely that revisions in the instruction repertoire will be necessary.

The SUMC processors must all be capable of executing the derived instruction repertoire. No problem arises with regard to the setup map in that each processor can contain local scratch memory for this purpose. However, there can be only one space for the action map which is common to all system elements. Status lines must be available at each processor for sensing action map indicators. Also, lines must be available for the transfer of SM data to the AM.

Therefore, a single device is required to contain the action map and perform the necessary switching operations. This device is referred to as the Configuration Map and Switch Control (CMSC) unit.

If the CMSC is also allocated the function of synchronizing the SUMC processors when the SWJ (Switch and Jump) instruction is executed, then no processor-to-processor communications is necessary to support configuration control.

SECTION IV. RECOMMENDATIONS

It is recommended that the concepts outlined herein be evaluated for feasibility by comparison with contrasting approaches to configuration control. It is also recommended that the approach be considered as a weighted forcing function to be used in trade studies oriented toward developing the busing structure associated with the SUMC multiprocessor development effort.

Further, it is recommended that micro-order logic diagrams sufficient to implement the instruction repertoire outlined herein be developed and used, along with the characterizing parameters listed in the conclusions, to bound the complete method in terms of read only memory requirements, scratch pad memory requirements, and instruction execution times.

Because of the large potential reduction in piece parts, sharing a set of buses between read and write memory operations should be examined. Also, if read only data and/or instructions could be lumped into a set of common memory banks, no write lines (or write buses) would be required thereby offering considerable reductions. This contingency should be borne in mind for future use.

The details concerning switching and voting with an IOU should be developed to complete the set of requirements. The methods outlined in this report are applicable and will simplify the task.

Finally, the CMSC (Configuration Map and Switch Control unit) should be functionally designed to determine the optimum AM (Action Map) structure and logic for switching, status reporting, and SUMC synchronization. It is felt that these functions should be combined physically with the software and trap support functions outlined in reference 1. The combination then constitutes a single separate system control unit that performs all of the following functions:

- Configuration Mapping
- Configuration Switching
- Configuration Status Reporting
- Process Dispatching
- Process State Error Analysis and Recovery
- Process State Transition Monitoring
- Job Stack (Ready List) Manipulation
- Adaptive Configuration Determination

[1] Kennedy, J. R.: Spâceborne Computer Executive Routine Functional Specification - Volume III: Executive Routine Primitives and Process Control - Final Report prepared under contract NAS8-24930, Computer Sciences Corporation, March 1971.

PRECEDING PAGE BLANK NOT FILMED

APPENDIX A. SCOPE OF WORK

CONFIGURATION CONTROL TASK
(Job M10050, 1 March 1971)

Purpose

The purpose of this task is to analyze the problem of control of the availability of hardware units (modules) in a multiprocessor computer configuration. The requirement in real time space systems for failure tolerant operation necessitates a highly flexible scheme for detecting and isolating module failures, verifying the failure, switching good modules online and bad modules offline, initiating testing procedures, managing the substitution of Line Replaceable Units, etc. A configuration control capability can be an integrated collection of capabilities and operational schemes to accomplish some or all of these functions. This task will specify the required capabilities in the case of a multiprocessor system in a space (high reliability, real time) environment, and develop approaches for implementing the capabilities on the SUMC.

Approach

The method of accomplishment will be to develop a set of features that are desirable for constitution of the configuration control capability. The features will then be used to develop an integrated description and method of operation. These will then be characterized by their major points of importance and played against the SUMC architecture in order to derive an implementation scheme consisting of any necessary special instructions, processor-to-processor intercommunications, register and memory organizations, or special control devices and paths. All findings will be detailed in a final report and optional slide presentation.

Plan

The present plan is to meet with cognizant S&E-ASTR-C personnel in order to determine the functions desired of the configuration control capability. This meeting is optional in that a set of functions can be assumed as follows:

- Power on/off,
- Switch online/offline,
- Switch idle/executing,
- Switch idle/self-test,

- Detect failure,
- Maintain configuration map,
- Transform from map_1 to map_2,
- Determine "best" map,
- Switch modules to conform to a map,
- Bootstrap/cold start, and
- Derivatives to be determined.

The meeting should be of a steering nature and is therefore recommended. The latest documented architecture of the SUMC will be ascertained in order to avoid developmental conflicts. CSC will assume an informal working relation exists.

APPROVAL

SUMC MULTIPROCESSOR CONFIGURATION CONTROL
ANALYSIS AND SPECIFICATION

By James R. Kennedy, Sr.

The information in this report has been reviewed for security classification. Review of any information concerning Department of Defense or Atomic Energy Commission programs has been made by the MSFC Security Classification Officer. This report, in its entirety, has been determined to be unclassified.

This document has also been reviewed and approved for technical accuracy.

Dr. H. Hoelzer, Director
S&E-COMP-DIR

2 0