

SOLOMON PROJECT TECHNICAL
MEMORANDUM NO. 22

Programmers' Considerations And Examples



Westinghouse

ELECTRIC CORPORATION

SOLOMON PROJECT TECHNICAL
MEMORANDUM NO. 22

Programmers' Considerations And Examples

4 November 1963

WESTINGHOUSE DEFENSE AND SPACE CENTER
Defense and Space Systems Operations
Baltimore, Maryland

This technical memorandum is published solely for information and use by project personnel and is not intended for external distribution. The material contained herein is PROPRIETARY.

TABLE OF CONTENTS

1. INTRODUCTION

Paragraph	Page
Introduction.	1-1

2. CONCEPT OF WRITING ONE PROGRAM TO CONTROL THE NETWORK AND NCU

Concept of Writing One Program to Control The Network And NCU .	2-1
---	-----

3. THE USE OF MODE CONTROL

The Use of Mode Control	3-1
-----------------------------------	-----

4. GEOMETRIC CONTROL

Geometric Control	4-1
-----------------------------	-----

5. USE OF THE BROADCAST REGISTER

Use of The Broadcast Register	5-1
---	-----

6. INTERCONNECTIONS BETWEEN PE'S

Interconnections Between PE's	6-1
---	-----

7. INDEX REGISTERS

Index Registers	7-1
---------------------------	-----

8. DOUBLE PRECISION ARITHMETIC

8.1 Double Precision Multiply	8-1
---	-----

8.2 Double Precision Add	8-2
------------------------------------	-----

9. INPUT/OUTPUT

9.1 General.	9-1
----------------------	-----

9.2 Contour Lines	9-1
-----------------------------	-----



Paragraph	Page
10. VARIABLE GEOMETRY	
Variable Geometry	10-1
11. COMMUNICATIONS BETWEEN UNITS (NCU, GPC, NETWORK)	
Communications Between Units (NCU, GPC, Network)	11-1
12. NUMBER OF POINTS PER PROCESSING ELEMENT	
Number of Points per Processing Element.	12-1
13. ASSOCIATIVE MEMORY PROPERTIES	
Associative Memory Properties	13-1
14. SPECIAL REGISTERS	
Special Registers	14-1
15. SOLOMON II CODING OF BAROTROPIC MODEL WEATHER STUDY	
SOLOMON II Coding of Barotropic Model Weather Study	15-1
16. SOLOMON II AS APPLIED TO A PRIMITIVE EQUATION MODEL	
16.1 Background Information	16-1
16.2 The Primitive Equation Model	16-5
16.3 Finite Differences	16-9
16.4 Initial Data	16-14
16.5 Averaging Procedure at Poles	16-18
16.6 SOLOMON II System	16-18
16.6.1 Computing North and South Pole Average Values	16-20
16.6.2 Method of Polar Data Transfer	16-24
16.6.3 Usage of the L-Buffer in Both Polar and Latitude Data Transfer	16-26
16.6.4 Computational Procedure	16-26
16.7 Assumptions in Coding	16-31
16.8 SOLOMON II Coding of Primitive Equation Model	16-32

Paragraph	Page
16.8.1 Variable Storage	16-33
16.8.2 Loop for Computing F_{ij} and ψ_{ij}	16-35
16.8.3 Relaxation Routine	16-39
16.8.4 Forecast Section	16-44

17. OTHER CONSIDERATIONS

17.1 Higher Order Differencing Schemes	17-1
17.2 Relaxation Techniques on the SOLOMON System	17-7

18. SEQUENTIAL PROBLEMS

Sequential Problems	18-1
-------------------------------	------

19. WAVE NUMBER SPACE COMPUTATION

Wave Number Space Computation	19-1
---	------

20. SUMMARY

Summary	20-1
-------------------	------

BIBLIOGRAPHY

Bibliography.	Bi-1
-----------------------	------



LIST OF ILLUSTRATIONS

Figure	Page
2-1 Network Control Unit and Processing Element Network	2-1
4-1 4 x 4 Sample Network	4-2
4-2 Boundary Conditions	4-3
4-3 Boundary PE's in Mode 0	4-4
4-4 Transfer of Data from PE Network to L-Buffer	4-5
6-1 PE's in Square Array in a Plane	6-1
6-2 Four Nearest Neighbors	6-2
8-1 Sign Position of Numbers	8-2
8-2 Sign Bits of Two Double Precision Numbers Line Up	8-3
11-1 Routes of Access Between NCU, GPU, and Network	11-1
13-1 Associative Memory Flow Chart	13-1
15-1 Flow Chart	15-2
16-1 Staggered Grid System	16-9
16-2 Staggered Grid System	16-11
16-3 Grid Point - PE Allocation	16-19
16-4 Configuration of Network	16-25
16-5 Diagram of L-Buffer Connections	16-27
16-6 Flow Chart for Primitive Equation Model	16-29
16-7 Flow Chart for Computing FLJ	16-32
16-8 Grid Point - PE Allocation for Initialization	16-36
17-1 Grid Point - PE Allocation	17-3
17-2 Four-Point-Per-PE Case	17-10
19-1 Diagrams for Network Configuration for Wave Space Computation, Initial State	19-5
19-2 First Shift to Right	19-6
19-3 First Row Completed and First Shift Down	19-7



1. INTRODUCTION

The purpose of this Technical Memorandum is to illustrate the capabilities of the parallel organized SOLOMON II Computer. Particular attention will be paid to the application of the SOLOMON II System to various aspects of the numerical weather forecasting problem. It is believed that the best possible method of demonstrating these capabilities is to illustrate a simple primitive equation model in detail plus other related problems which, taken together, contain types of operations typically encountered in numerical weather forecasting. In addition, a detailed coding of the primitive equation mode is presented to demonstrate the principles of SOLOMON II programming and to obtain a running time estimate for this model.

For those not familiar with SOLOMON II programming, it is suggested that the "SOLOMON II Programmers Reference Manual" be read concurrently, in order for the reader to obtain a better understanding of the SOLOMON II programs presented in this report. Also, for those interested in the background material on the numerical weather forecasting topics presented, it is recommended that the appropriate references, listed at the end of this report, be reviewed.

Also included in this report are brief descriptions of general physical characteristics and applicable subroutines associated with the SOLOMON II Computer. For more detailed information on the technical and physical aspects of the SOLOMON II System, reference should be made to SOLOMON Project Technical Memorandum No. 25, "SOLOMON II Physical Characteristics."



2. CONCEPT OF WRITING ONE PROGRAM TO CONTROL THE NETWORK AND NCU

Although the Network Control Unit (NCU) and the Processing Element (PE) Network appear to be two separate units (see figure 2-1), each with its own set of instructions, a program interspersing commands for both units will cause concurrent operation of the two.

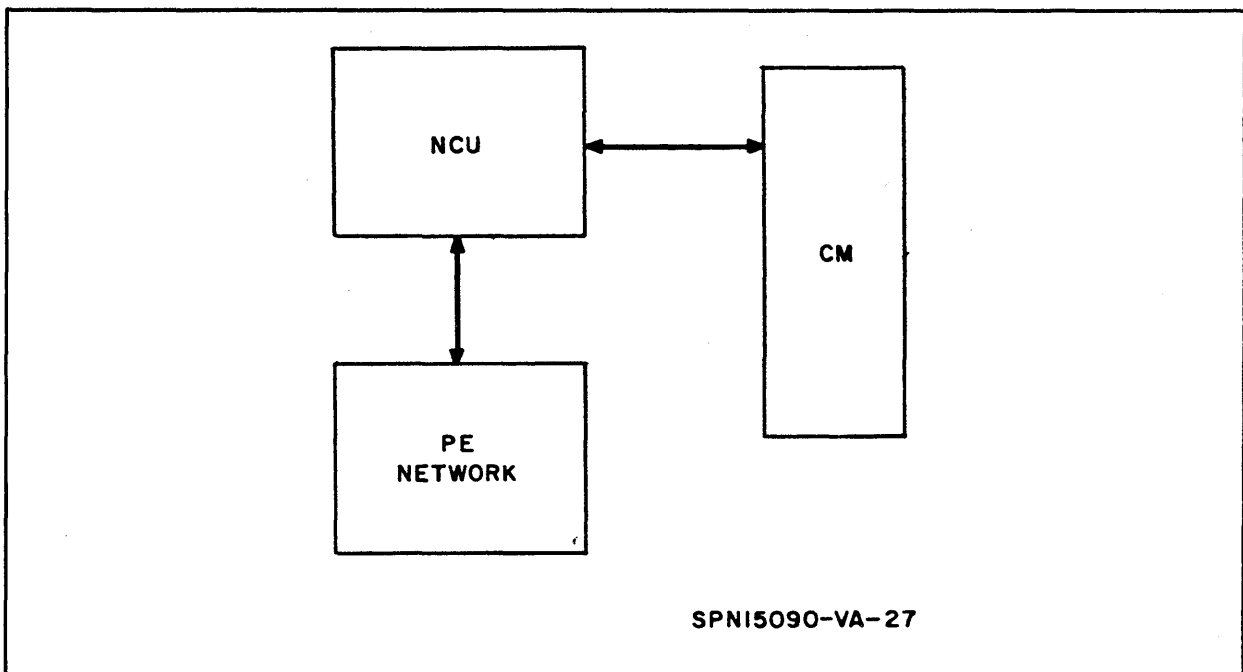


Figure 2-1. Network Control Unit and Processing Element Network

The NCU monitors all instructions. An instruction is brought from Central Memory (CM) into the NCU. If it is a PE instruction, it is sent to the Network for execution and the next instruction is brought from CM. By interspersing NCU commands and Network commands, the two units perform simultaneously.

Suppose, for example, a program contains a PE multiply instruction (execution time is 13 microseconds). The program can be written so that

about 6 NCU commands can be executed during this time. Thus in some cases most of the PE bookkeeping (i. e. , indexing, shifting, etc) can be done without loss of time.

A program written with interspersed commands will enable SOLOMON II to obtain the maximum in concurrent operation.

Also in line with the above, the operations of both the NCU and the PE Network can be executed by the same program. This is due to the fact that all instructions are extracted from NCU memory and those pertaining to the PE Network are given to the Network to execute via the NCU, while those pertaining to the NCU are executed by the NCU. Thus one program is capable of controlling both the NCU and the PE Network either separately or together (concurrently), depending upon how the two types of instructions are interspersed. It should be mentioned that the main computing power is in the Network while the NCU only performs operations such as indexing, bookkeeping, and data transfer.

3. THE USE OF MODE CONTROL

Mode control is one of two methods provided for referencing one or more groups of PE's. At any point in the program any PE may be set to one of four modes (0, 1, 2, 3). Once a PE is set to a mode it has the option of responding or not responding to a given instruction according to the PE mode and the mode control indicated in the instruction (i. e. , PE's in Mode 1 respond to instructions addressed to M1 but do not respond to instructions addressed to M 0, 2, 3). Thus, the mode status provides the programmer a means of control of special situations in a program.

Assume, for example, a weather problem where each PE represents a given grid point in the system. Assume also that the PE arrangement is a fixed square > 2 by 2 (this is purely an assumption, however, as the arrangement may be any configuration). Mode control will provide an effective method of differentiating between boundary points and internal mesh points in the square arrangement.

The equation $\nabla^2 \phi = F(x, y)$ will serve well as an example of mode control. An averaged ϕ must be found for both poles. (Assume we are working with the north pole. All points have a value for ϕ but only those adjacent to poles are to be averaged.) In a matrix array of PE's the uppermost row is assumed to be the north-most row. Mode control provides a method so that only those PE's concerned with the north row will be averaged and ensures that the internal mesh points will not respond to the averaging instructions. This method is as follows:

- a. Set all PE's to Mode 0
- b. Set the uppermost or north row to Mode 2
- c. Instructions for the averaging routine will reference those PE's in Mode 2 only. (This same routine will be repeated for the south pole ϕ .)



Therefore, in the above example, mode control provides boundary control.

The relaxation routine in a weather problem also depends on mode control for its completion. For each grid point there is an original assumption for $\psi = \frac{\phi_{ij}}{f_{45}}$. This ψ is then substituted and the resulting R_{ij}^m is tested against a given ϵ . If $\epsilon - R_{ij}^m < 0$, convergence has not occurred. A new value of ψ is computed and the loop is recycled; i.e., the new value of ψ is substituted and convergence is again tested.

The PE's are originally set to Mode 0. At the end of the first cycle, any PE which has not converged is set to Mode 2. This is accomplished by checking the sign bit; i.e., any PE with a negative result is set to Mode 2. The mode status is then tested. If any PE is in Mode 2, all PE's are again set to Mode 0 and the loop is recycled. Only when all PE's remain in Mode 0 at the end of a cycle has convergence occurred. Thus mode control allows the termination of a loop.

The above two examples for the use of mode control (boundary control and loop control) are a small sample of its many uses.

The availability of only four mode states may seem a restriction on the programmer but the ability to store and load mode states (mode map) into and from the PE core memory provides the programmer with unlimited (actually limited only by the amount of core storage available) mode states. The PE instruction set contains the necessary instructions for mode control. In addition, the programmer can change the mode state of PE's based on the result of an arithmetic operation, overflow conditions, bit checks, etc. Thus mode control provides the programmer with a powerful method of control.

4. GEOMETRIC CONTROL

A second method of idling some PE's is to use two registers which exert what is called geometric control. One register, called the row select register, is associated with the rows of the square array of PE's and the other, similarly associated with the columns, is called the column geometric register. The row geometric register contains exactly one bit corresponding to each row of PE's. The column geometric register contains exactly one bit corresponding to each column of PE's. The contents of the geometric control portion of the Machine Option register determines which of the four geometric possibilities is to be effective during an instruction:

- a. Neither register is active
- b. The row geometric register is active and the column geometric register is inactive.
- c. The column geometric register is active and the row geometric register is inactive.
- d. Both registers are active.

When either register is active, its effect is as follows. If the k-th bit in the row (column) geometric register is a 0, then the PE's in the k-th row (column) are all idle. If this bit is a 1, then these PE's are active, though still subject, of course, to mode control. When either of the registers is inactive, it does not idle any of the PE network; i. e., an inactive register is equivalent to a register filled entirely with 1's. In the 4 by 4 sample network shown in figure 4-1 below, the PE's behave as follows as a result of geometric control. In case a listed above, all 16 PE's are active. In case b, all 12 PE's which are shaded with horizontal or with both horizontal and vertical lines are active. In case c, the eight PE's shaded with vertical or with vertical and horizontal lines are active. In case d, the six PE's which are shaded with both horizontal and vertical lines are the only ones active.



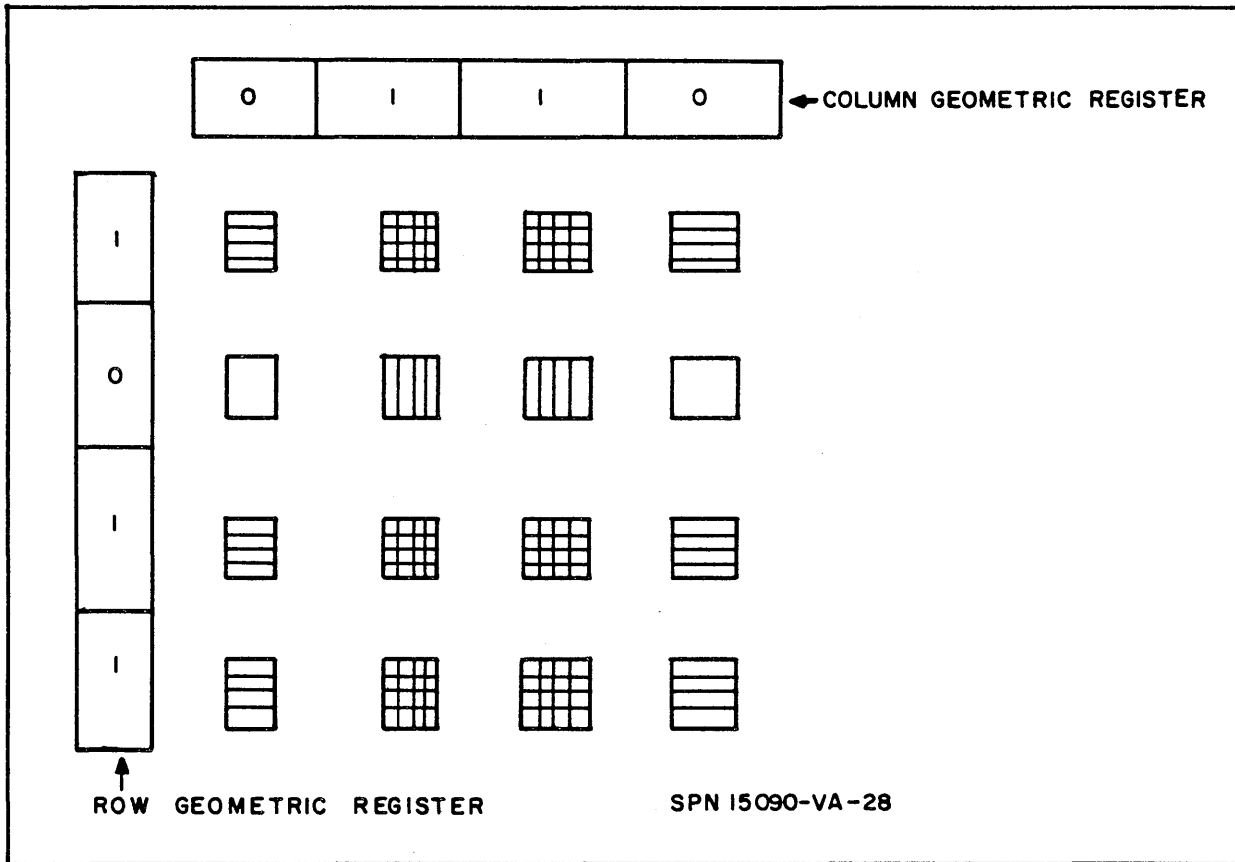


Figure 4-1. 4 x 4 Sample Network

Geometric control has many uses. The following examples will demonstrate some of these:

a. Establish Boundary Conditions (see figure 4-2)

Presume all PE's are in Mode 0. The GR would then be loaded with constants (k where $k = 011$). An instruction will then set all activated PE's to Mode 3, thus establishing the boundary conditions with all boundary PE's in Mode 0 (see figure 4-3). Actually for this simple case the programmer can specify that geometric control is active (both) and the mode state of the boundary PE's are immaterial.

b. Input of Data

As a continuation of a, suppose data is to be sent to boundary points only. Using the L-Buffer, where data are stored in locations 0, 1, ... 31, an

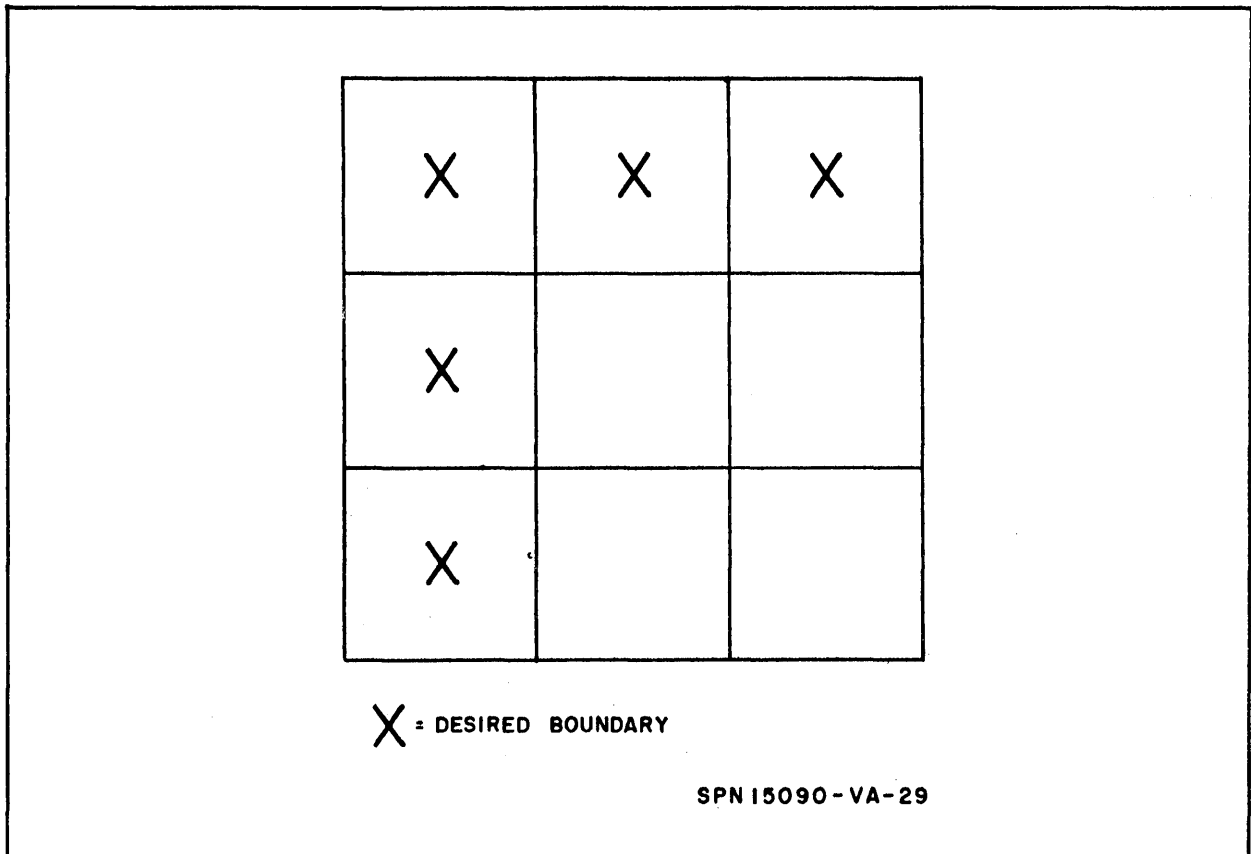


Figure 4-2. Boundary Conditions

instruction will transfer the contents of the L-Buffer to corresponding active PE's (as designated by 1's in CG and RG). Thus data is brought into the Network under geometric control.

c. Output of Data

If only certain data is to be output, geometric control is employed to transfer the data from the PE Network to the L-Buffer (see figure 4-4).

Suppose the PE containing the desired information is in Mode 3. Any instruction will command any column with active PE's (i.e., with a PE in Mode 3) to place a 1 in the corresponding bit position of the CG. The next instruction, with the combined use of geometric and mode control, will transfer data from the Mode 3 PE in column 1 to the L-Buffer for output. In this example geometric control provides output control.

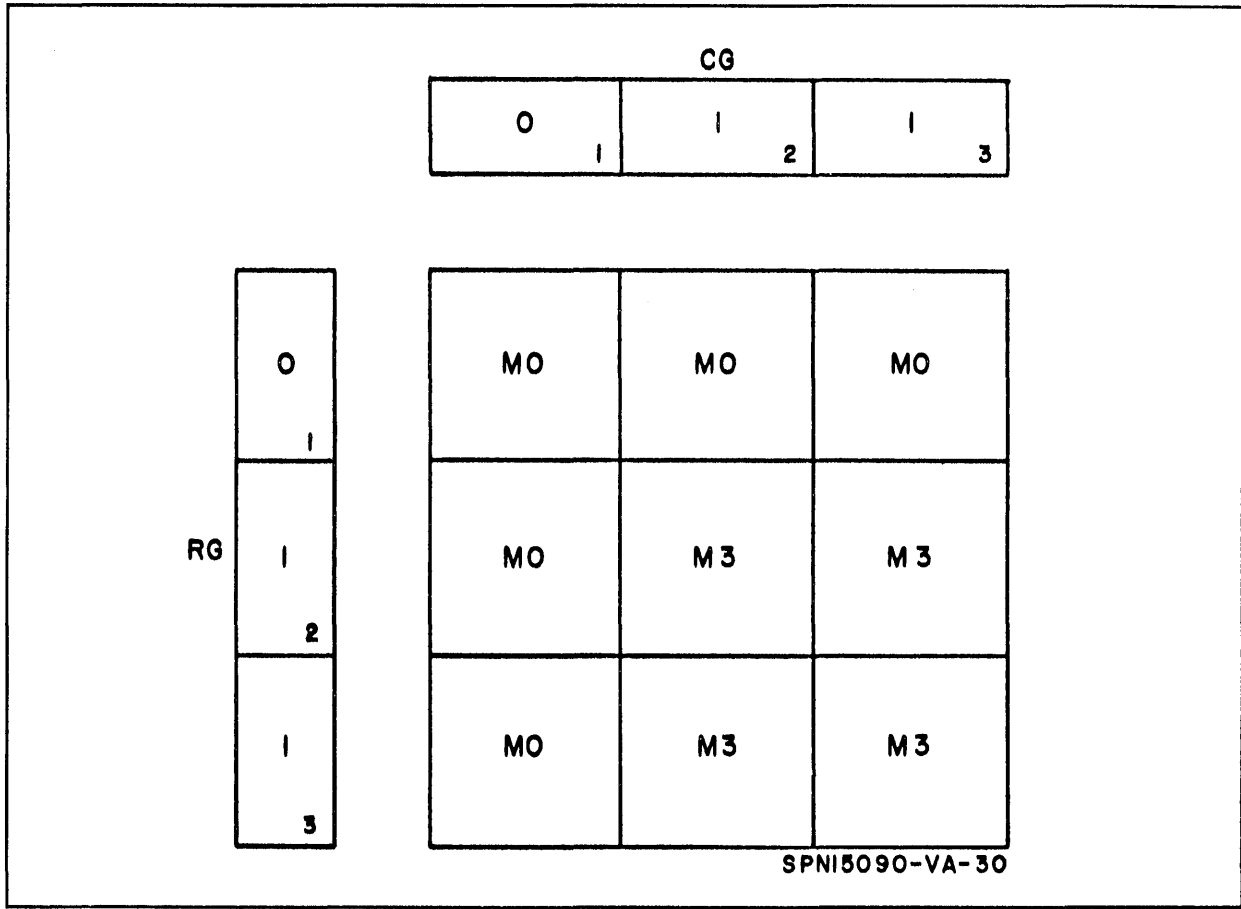


Figure 4-3. Boundary PE's in Mode 0

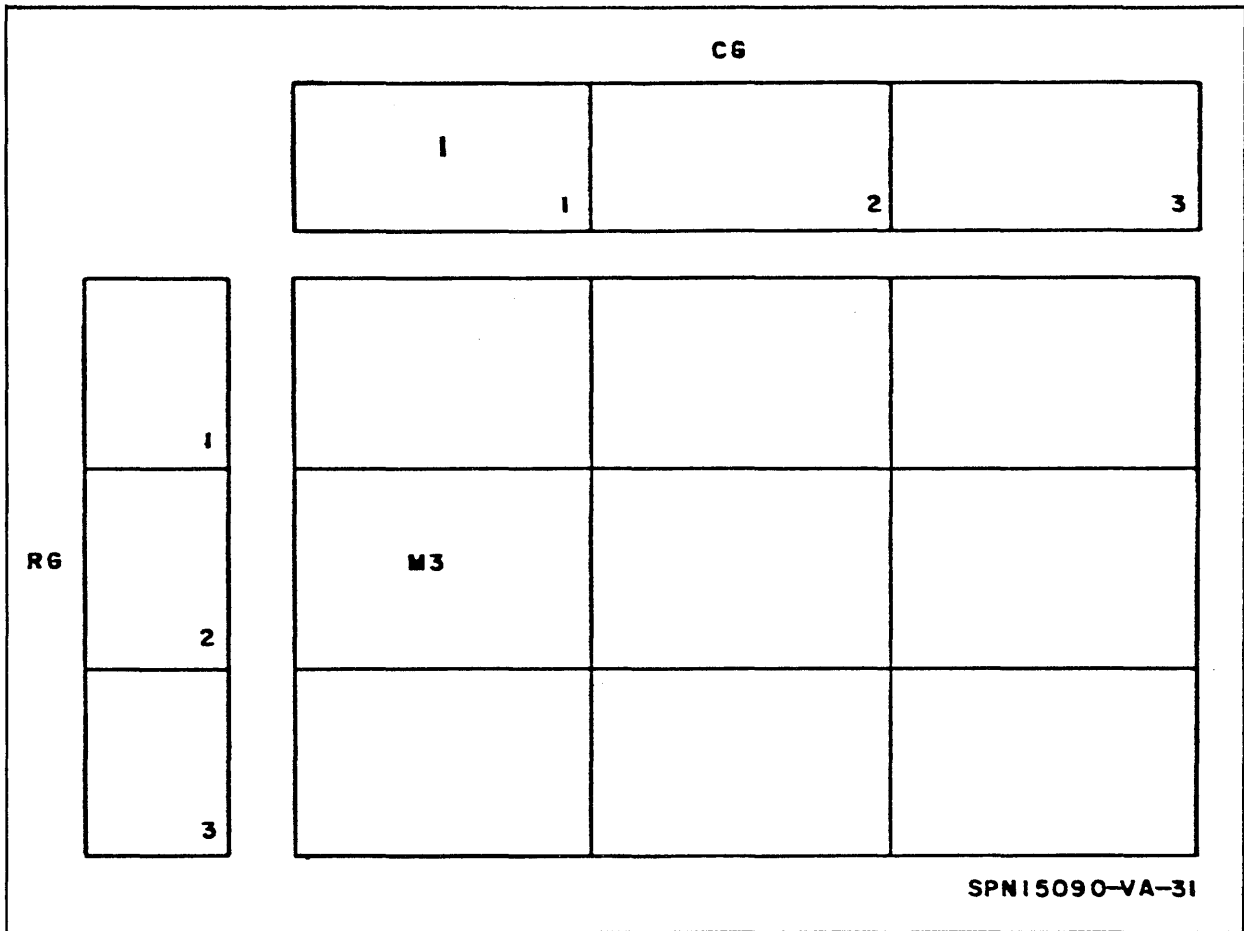


Figure 4-4. Transfer of Data from PE Network to L-Buffer

5. USE OF THE BROADCAST REGISTER

One of the features of SOLOMON is the Broadcast Register (B); a 20-bit register which transfers data from Central Memory (CM) to the PE Network.

Suppose a problem requires constants for its solution. These constants are stored in CM. One instruction transfers the data from CM to B thus making the constant available to all PE's. After the constant is in B, it may be used for various arithmetic operations in addition to being broadcast to all PE's. By using B the PE memory is preserved (one memory location as compared to one per PE).

The register is also adaptable (as an option) for special usage. For example, problems using polar boundaries require values to the north or south of the base PE. These values come from neighboring PE's for internal points. The acquisition of these values for the polar parameters, however, poses special problems. An easy solution is to use B as a north or south routing (i. e., provide north and south averaged values) thus permitting polar PE's to gain needed information.

Thus two uses of B are:

- a. Providing constants for use in all PE's
- b. Providing north or south routings for special problems such as supplying polar values when needed.



6. INTERCONNECTIONS BETWEEN PE'S

An important feature of the PE network is that neighboring PE's are connected so that each PE has available, as well as the data in its own memory, the data in the memory of its four nearest neighbors.

Ordinarily, the PE's are thought of as lying in a square array in a plane (see figure 6-1). The basic SOLOMON network is considered to have an array of 32 by 32 PE's. Without any design changes, however, modules of 256 (32 x 8) PE's can be added to or removed from the network. Thus the minimal network contains 32 by 8 PE's. For a PE which is interior to the array, the four nearest neighbors are designated as east (E), north (N), west (W), and south (S) (see figure 6-2). The PE's along the edges have only three nearest neighbors, while those on the corners have only two.

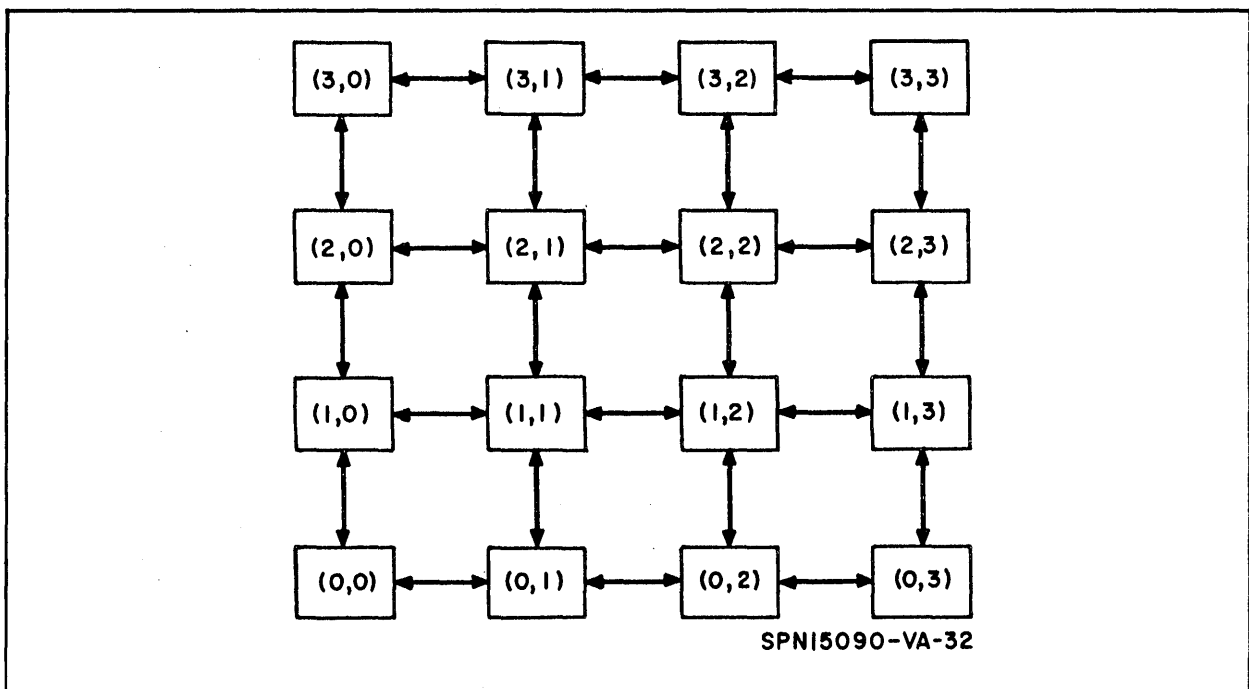


Figure 6-1. PE's in Square Array in a Plane



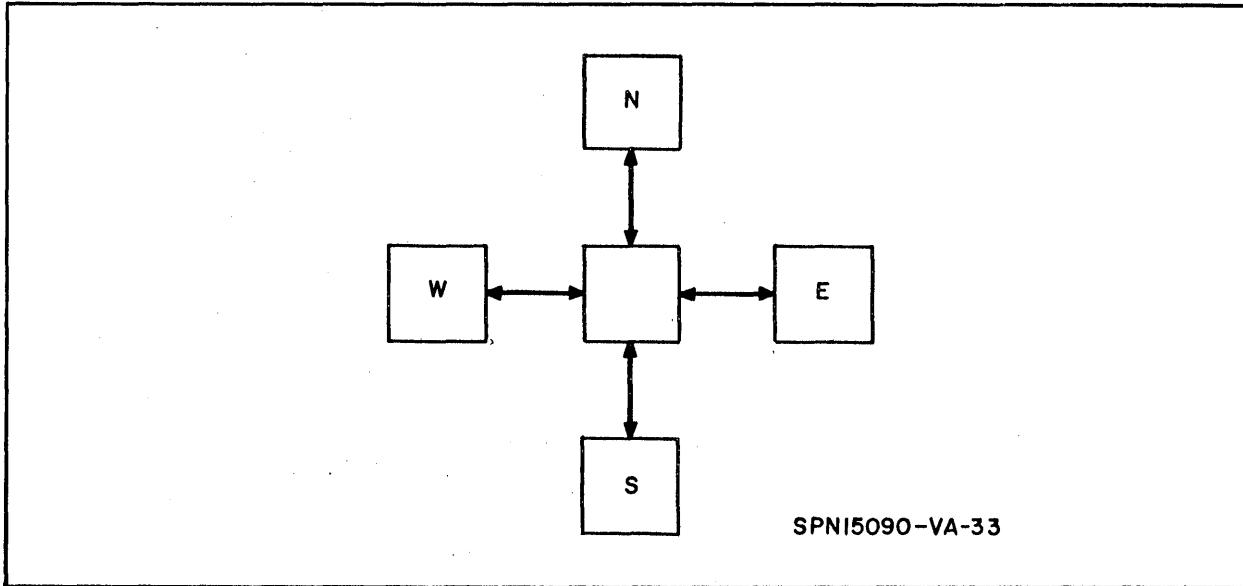


Figure 6-2. Four Nearest Neighbors

7. INDEX REGISTERS

The SOLOMON II System has seven index registers. The two main uses of the index registers are to provide loop control and to provide a means of picking an element out of an array.

The following example illustrates the use of an index register (in this case X1) to provide loop control. Suppose there is a series of N instructions that must be performed Z times. Then the following loop could be constructed:

	TIX	XI, Z	(Set X1 to Z)
Begin	Inst. 1		
	Inst. 2		
	.		
	.		
	.		
	Inst. N		
	SIX	X1, -1	Subtract 1 from X1
	JXZ	≠, X1, Begin	(if X1 ≠ 0, jump to Begin)

The loop is performed Z times. When the Z-th time is completed, X1 = 0 and the next instruction following this loop will be executed.

Suppose that there exists a one dimensional array of n elements stored sequentially as:

address	ELEMENT 0
address + 1	ELEMENT 1
address + 2	ELEMENT 2
.	
.	
address + n-1	ELEMENT n-1



Suppose it is now desirable to perform an operation on the Z-th element of this array ($0 < Z \leq n$). Also suppose that the desired operation is a TMP.

This could be performed as follows:

TIX	X, Z-1	Set index register X to Z-1
TMP	ELEMENT, X	

The address of ELEMENT is added to the contents of X and the element in this new address is transferred to the P register. This element is indeed the desired Z-th element of the array.

Since the SOLOMON II System has seven index registers, it is possible to have control over many loops and arrays with a minimum of manipulation. As a final comment on this topic, any one of the seven available index registers can be used to control loops in either the PE Network or in the Central Control Unit. Hence the programmer has available seven index registers that can be used to index PE instructions or NCU instructions.

8. DOUBLE PRECISION ARITHMETIC

8.1 DOUBLE PRECISION MULTIPLY

This program was written for application to problems which require greater than average precision. A double precision number (2 + 38 bits) is represented by 2 (1 + 19) bit numbers. A restriction on the location of the most significant and least significant 20 bits is not necessary but in general they will be located in sequential memory locations. A double precision number can be expressed as the sum of the most significant 19 bits plus the least significant 19 bits as $A + \Delta A$. The product of two such numbers yields the identity:

$$(A + \Delta A) (B + \Delta B) \equiv AB + \Delta AB + A\Delta B + \Delta A\Delta B \quad (8-1)$$

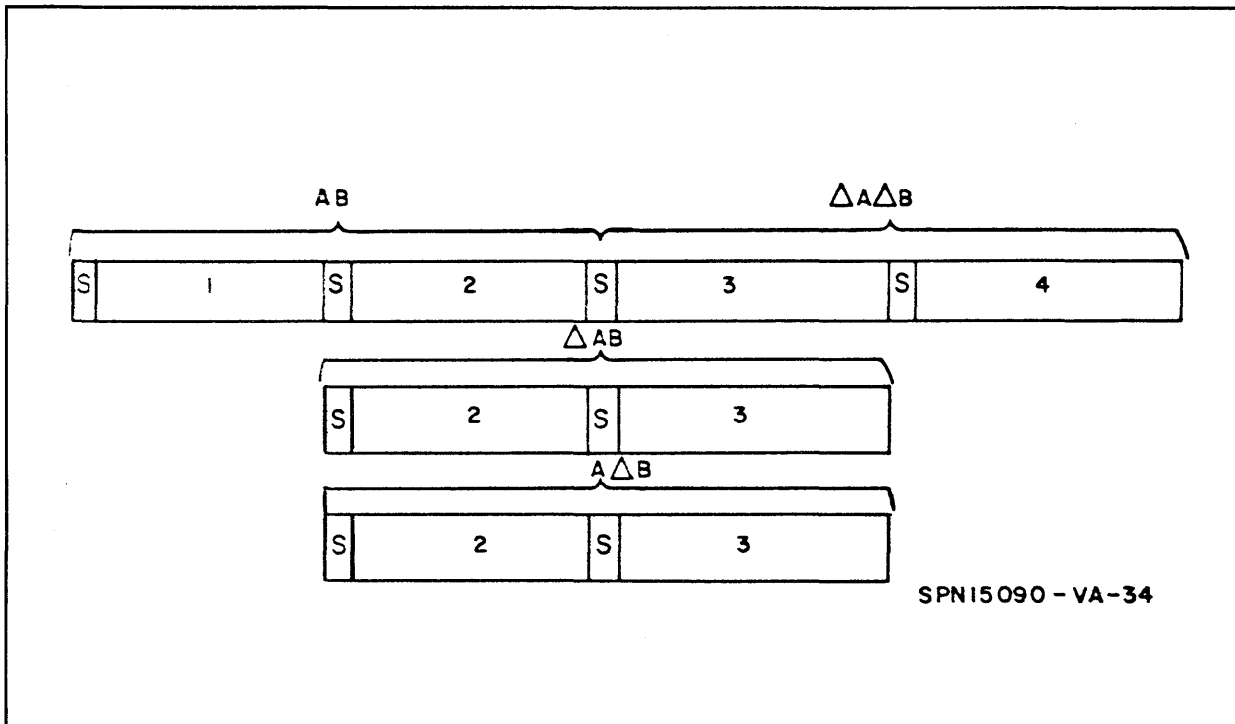
In fixed point form, $(A + \Delta A)$ can be expressed as $(A \cdot 2^{19} + \Delta A \cdot 2^0)$. Thus the right side of equation 8-1 can be expressed as in equation 8-2

$$AB \cdot 2^{38} + \Delta AB \cdot 2^{19} + A\Delta B \cdot 2^{19} + \Delta A\Delta B \cdot 2^0 \quad (8-2)$$

Note from equation 8-2 that the sign positions of the numbers will line up as shown in figure 8-1.

One obtains the 80-bit product by adding the appropriate single length blocks of 20 bits. Care must be taken to preserve the carry bit from the addition of each pair of blocks. The carry bits are added into the next column of blocks. The use carry (UC) option on the addition operation adds in the carry flip-flop. In general, it is impossible to get overflow in the multiply operation. However, since the 2's complement representation is used for negative numbers it is possible to have $(-1) \times (-1) = +1$ which causes an overflow. This is the only case. It is not necessary to sense overflow in the addition operation except in the final column. If overflow occurs in the second or third column, only the sign bit will be wrong. But these are changed to match the sign of column 1 by the algebraic shift of 0 places at the end of





SPN15090 - VA-34

Figure 8-1. Sign Position of Numbers

the program. If overflow occurs in column 1, a jump instruction transfers the computer out of the program to an instruction called OUT. (OUT must be designated in the main program.)

The resultant is stored in four locations called R1, R2, R3, and R4. R1 is the most significant 19 bits and R4 is the least significant. Note that the same digit occupies all four sign bits located within the 80-bit result.

8.2 DOUBLE PRECISION ADD

This program was written to accompany the double precision multiply program. A double precision number is expressed as $(A + \Delta A)$ where A is the most significant 19 bits and ΔA is the least significant. Since the sign bits of two double precision numbers line up, the numbers can be added in blocks of 19 bits (see figure 8-2).

The carry from column 2 is added to column 1 by the UC option on addition. The necessity of detecting overflow is left to the programmer. The most significant 19 bits of the result are stored in the P Register and the least



SOLOMON II CODING FORM

PROGRAM: DOUBLE PRECISION MULTIPLY DATE:

PROGRAMMER: R JAMES BENNETT

PAGE 1 OF 3

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22		72 73 80
					COMPUTE: AB, AΔB, ΔAB, ΔAΔB	TIME
			TRQ	1	A	22
			MQ	1	B	1 30
			TPM	1	AB1	22
			TQM	1	AB2	22
			TRQ	1	A	22
			MQ	1	ΔB	1 30
			TPM	1	AΔB2	22
			TQM	1	AΔB3	22
			TRQ	1	ΔA	22
			MQ	1	B	1 30
			TPM	1	ΔA B2	22
			TQM	1	ΔA B3	22
			TRQ	1	ΔA	22
			MQ	1	ΔB	1 30
			TPM	1	ΔA ΔB3	22
			TQM	1	R4	22
						78.4 μSEC

2000A-1





SOLOMON II CODING FORM

PROGRAM: DOUBLE PRECISION MULTIPLY DATE:

PROGRAMMER: R JAMES BENNETT

PAGE 2 OF 3

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22		72 73 80
X					COMPUTE R3, R2, R1 ADJUST SIGNS	TIME
		SQZ	/		Q → O	22
		TRP	/		ΔAΔB3	22
		AP	/		ΔAB3	22
		QSC	/		C → Q	22
		TQM	/		CARRY	22
		AP	/		AΔB3	22
		TPM	/		R3	22
		TRP	/		CARRY	22
		AP	/		AB2	22
		QSC	/		C → Q	22
		TQM	/		CARRY	22
		AP	/		ΔAB2	22
		QSC	/		C → Q	22
		AP	/		AΔB2	22
		TPM	/		R2	22
		EPQ	/		Q → P	22
		APC	/		CARRY	22
		APC	/		3AB1	22
		JMD	/		3OUT	30
		TPM	/		R1	22
		TRQ	/		R2	22
		SPQRA	/	C		22
		TQM	/		R2	22
		TRQ	/		R3	22
		SPQRA	/			22
		TQM	/		R3	22

200041

58.0 M SEC.
+ 78.4
136 U SEC

9. INPUT/OUTPUT

9.1 GENERAL

In general, the I/O operations in the SOLOMON II System are quite conventional in that data channels are used to transmit data between central memory and the standard peripheral devices such as magnetic tape, disk, high speed printer, and typewriter. However, the transmission of data between central memory and the PE is rather unique. This transmission of data is accomplished by a serial-to-parallel converter called the L-Buffer. The actual details of transferring data is covered in SOLOMON Technical Memorandum No. 25, "SOLOMON II Physical Characteristics."

To store information into the PE memories, the information must first be read into central memory by tape, punched cards, etc. From the central memory, it is transferred to the L-Buffer and from there to the PE memories.

The output from the PE Network process is the exact reverse procedure. The desired information is transferred from the PE memories to the L-Buffer and from there to the printer or other peripheral devices.

9.2 CONTOUR LINES

The final output for many forecasting models are contour lines of weather parameters such as pressure and temperature. For example, at the end of the 500-mb forecast, the desired output consists of the 500-mb height field. Assume that it is desired to obtain the contour for 500-mb surface at $19,000 \pm \pm 50$ feet. The following series of instructions can be used:

MSI	Set all PE's to Mode 0
TMB	Load Broadcast Register with 19,000
TRP	Load P with height



SR	B	$(P)-(B) \rightarrow P$
TMB		Load Broadcast Register with 50
MSLA		Set all PE's in which $ P < B$ to Mode 2

Each PE has two flip-flops to store the mode status and for this case any PE which had answered yes would have a $(10)_2$ combination in its mode flip-flops. The Mode 2 PE's will define the contour line and this line can be displayed on a TV tube by the use of a display device. The same procedure can be used to obtain the remaining contour lines of the height field and then a picture of the TV tube can be taken for the final output. This process of displaying contour lines could be a continuous process and would allow the Meteorologist to observe the progress of the forecast as it is being generated. This procedure would seem to be a powerful tool for a Meteorologist working to experiment with different models, different finite differencing approximations, or the general heating of the atmosphere.

The contour output scheme should also provide a speed advantage over the present method of feeding the data to an analog plotter. However, no time estimate is available at this time.

10. VARIABLE GEOMETRY

Three alternative configurations, described below, are available in addition to the planar one described previously. The choice of the 4 possible configurations is made by the programmer. (Additional configurations are available as a special option.)

By joining the east and west edges of the planar array, the network becomes cylindrical, with open north and south ends. Using figure 6-1 as an example, a cylinder would be formed by connecting the following pairs of PE's: (0, 3) and (0, 0), (1, 3) and (1, 0), (2, 3) and (2, 0), (3, 3) and (3, 0).

Similarly, by joining the north and south edges of the planar array, a cylinder with open east and west ends is formed.

If the connections specified in the last two options are made at the same time, the figure becomes a torus. In this case, every PE in the network has four nearest neighbors.

The possibility of selecting one of several configurations, depending on the problem, is attractive.

For example, in a physical problem which is most conveniently described in cylindrical coordinates, the connection between the two opposite edges of the planar array is automatically made. Without this provision, program steps would have to be written to create this connection artificially. To join the most easterly and most westerly columns by programming would involve the following: every time all the PE's consult their western neighbors, the same instruction can be executed by the PE's in all columns except the most westerly one. This column's west neighbors are in the most easterly column. At least two more instructions (as seen later) would be required to get the required neighborly information from the east side of the square array to the west side.



11. COMMUNICATIONS BETWEEN UNITS (NCU, GPC, NETWORK)

Communications between the Network, the Network Control Unit, and the General Purpose Unit are outlined by the schematic diagram shown in figure 11-1.

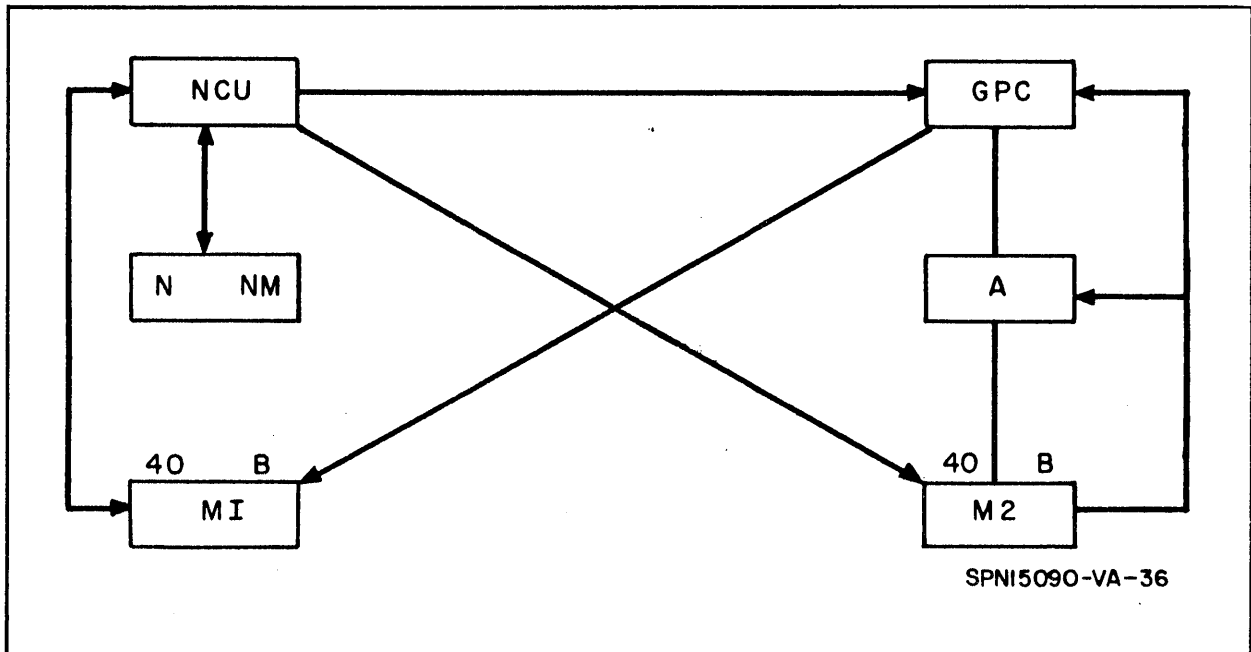


Figure 11-1. Routes of Access Between NCU, GPU, and Network

In the diagram above:

- N, NM - PE Network and PE memory, respectively
- GPC - General Purpose Control
- M1 - Memory for NCU (40 bits)
- M2 - Memory for GPC (40 bits)
- A - Arithmetic Unit (associated with GPC) Both A and GPC together comprise the General Purpose Unit.



As seen from the above diagram, direct access is available for the NCU and GPC to each other's memory. This implies that the NCU is able to communicate directly with the GPC memory and conversely, the GPU is able to communicate directly with the NCU memory. Communication with respect to the Network is supplied only through the Network Control Unit. The PE Network does not have access to the General Purpose Unit directly but must communicate through the Network Control unit, then to the GPU by the instruction HELP. All data transfer from or to the PE Network must take place through the NCU (via either the L-Buffer or Broadcast Register). Also, all indexing instructions to the PE Network are executed and controlled by the NCU only.

12. NUMBER OF POINTS PER PROCESSING ELEMENT

Each PE is capable of handling any desired number of grid points on a finite difference mesh (within the required storage limits for those points). In general, this grid point - PE allocation is determined mainly by the requirements (size) of the finite difference network being solved and the number of Processing Elements available (for this report 32 x 32 or 1024 Processing Elements). The allocation can be as small as one grid point per PE or many grid points per PE subject to the storage limitations of those points. The only restriction imposed on this grid point allocation other than storage is that it be uniform throughout the entire Network, otherwise some special programming considerations must be applied.

It might be mentioned here that as the number of grid points per PE increases, it becomes more desirable to use some form of a loop to avoid the tedious task of straight line coding for all points, provided the same instructions apply at each point. This technique will result in a slight time loss due to the indexing instructions necessary to control the loop but could be very valuable in situations where the grid point allocation per PE is very large. In writing a loop for a set of points with a Processing Element, there are two factors which must be considered.

- a. The routing must be changed for various points (implies modification of the routing field in the instruction).
- b. The positions with the PE must be incremented (implies indexing of positions).

Of the two factors given above, changing the routing appears to be the most formidable. This can be handled by either of the following two methods:



a. The routing can be indexed by indirect addressing (SOLOMON Project of Technical Memorandum No. 23 "SOLOMON II Programmers' Reference Manual"). This method consists of placing all of the required addresses to be used for each position in a sequential table in Central Control Memory and using indexing controls on the PE Network to provide the correct addresses associated with each position in turn. This corresponds to a "table look-up" and could result in a slight loss of time since the PE Network may be inactive while the fetching procedure of the correct addresses is being performed.

b. Indexing the routing can be avoided by simply coding a separate subroutine for each position (or groups of positions), controllable by indexing from Central Control. Each separate subroutine would contain the necessary operations and routings required for the calculations to be performed at that particular position. For the case where there are many points per Processing Element, a single subroutine can be written to take care of all interior points within the PE while the necessary subroutines can be written to account for the peripheral PE positions (one for each corner and one for each side containing other than corner points). In general, the corner points will require special treatment since it is at these positions where the greatest variation in routing will occur. An example of this type of program for the eight-point-per-Processing-Element case is illustrated in the coding for the initialization section of the sample forecast problem which is presented later in this memorandum.

13. ASSOCIATIVE MEMORY PROPERTIES

The purpose of this section is threefold: to assign a definition unique to associative memory programs, to develop and illustrate an example of associative memory programs, and to provide a flow chart (see figure 13-1) and program coding for a more comprehensive study.

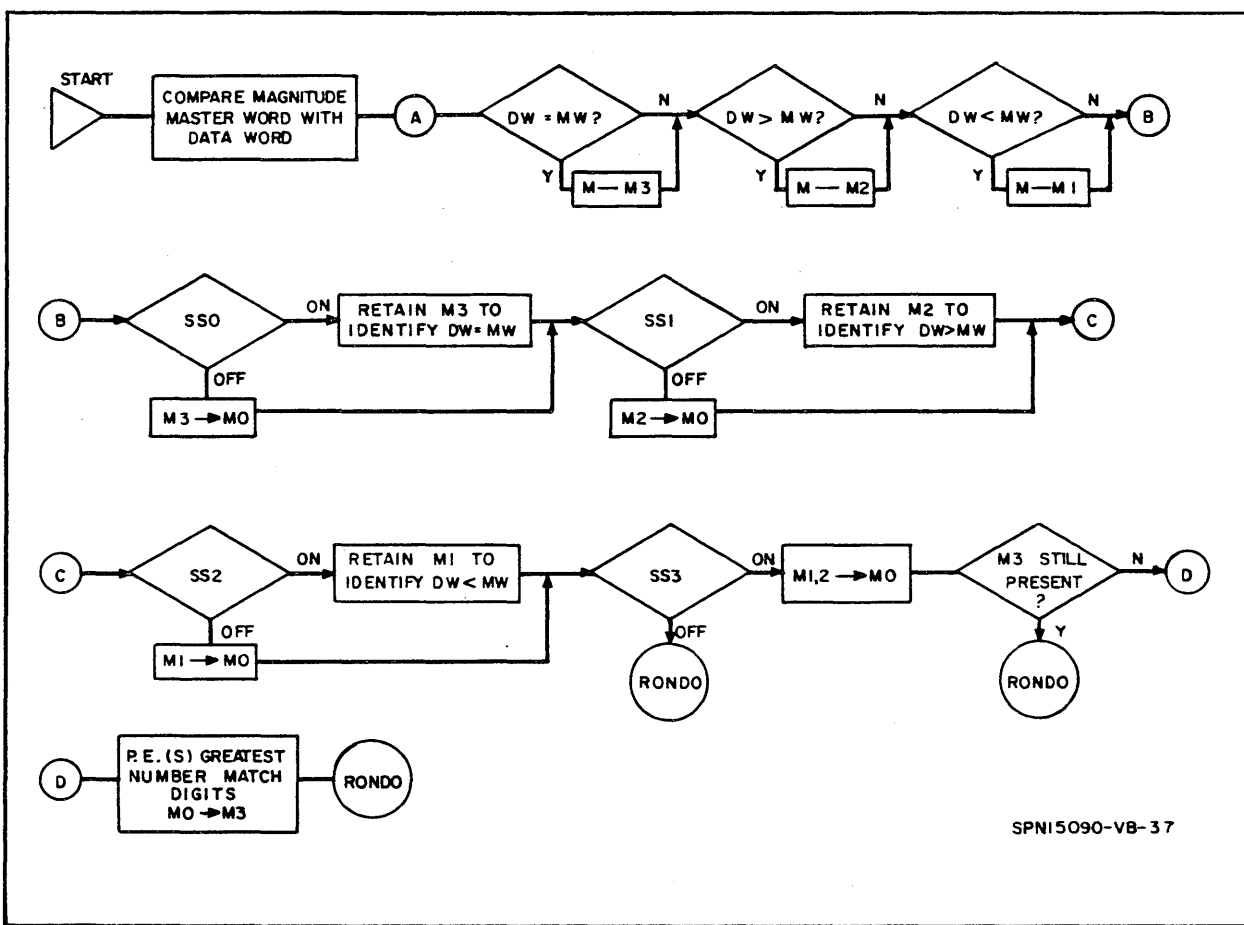


Figure 13-1. Associative Memory Flow Chart

To define associative memory programs, consider a normal program (with or without parallel processing). The program will call for specified

operations to be performed which involve specified values stored at specified memory locations. In contrast, associative memory programs may utilize a parallel process machine to perform unknown functions with unknown data stored in unknown memory locations. The term unknown here implies that the three involved factors are not rigidly designed into the program and are not determinable through visual examination of the program coding. During the development of the model program, the above mentioned three unknowns will be reiterated as they become a set of interrelated factors which establish the function and effect of the program.

In examining the exact function or purpose of the model program, consider that it may represent one facet of a complex integrated system of data processing rather than being complete and independent. That is to say, the program will represent to an experienced programmer or systems analyst a simple example of data maneuvers and comparisons which are now ordinarily handled in a very awkward and time consuming manner. Supporting this statement, the several major accomplishments of the program will be listed:

- a. Simultaneous magnitude or similarity comparison of a large set of data words to some controlling factor (herein called the master word). This corresponds to the first function block on the attached flow chart.
- b. Categorizing of the data words into one of the following four groups:
 - (1) Data word = master word
 - (2) Data word > master word
 - (3) Data word < master word
 - (4) Greatest similarity based on matching bits or absence of bits in corresponding bit positions.
- c. Controlling which classes of data, or combination of classes, will be assigned to further processing and which classes are not to be retained as unique categories for selection.

Consider that the model program will first compare and categorize each data word (hereafter called DW and stored in processing element memory)

by a few simple steps relating the DW and the master word (MW). First simultaneous subtraction is performed and the difference MW-DW is stored back into each PE. The magnitude of the difference is examined to establish one of the following three relationships:

- a. $MW-DW = 0$; Equality
- b. $MW-DW < 0$; $DW > MW$
- c. $MW-DW > 0$; $DW < MW$

Actually, the first two conditions cited are tested for and the third is assumed through this elimination process. This represents the operations from point A to point B on the flow chart; and as the chart points out, the data is being categorized during these tests by assigning one of four PE mode settings to each group of PE's. The operations begin with all PE's set to mode zero and the following modes are assigned to represent each group:

- a. Equality; Mode 3
- b. $DW > MW$; Mode 2
- c. $DW < MW$; Mode 1

Note that at this point the SOLOMON II has used a half dozen instructions to compare and categorize 1024 data words in its PE's by relating them to the controlling MW. Mode selection would now be capable of assigning any or all of the data groups to one or more subroutines. The similarity test has not been made at this point since it would have no meaning when used with the above three tests. Equality would obviously mean complete similarity. However, a lesser than word and a greater than word may be equally similar to the master word.

The program now utilizes 3 of the 40 sense switches set from the SOLOMON II Console Panel to determine whether the data groups should retain their unique mode setting or be returned to Mode 0. A 4th switch is tested to see if the similarity test is desired. These processes begin at point B on the flow chart.



At point D, an interesting similarity test is achieved by simultaneously creating a word containing bits in all matching digits and then adding the bits to determine which word or words are most similar to the master word.

First the matching bits are established by the Boolean AND function and the resulting word is stored. $MW + DW$ is then performed as an OR function. This is complemented and another OR adds this and the result of the AND operations. The results contain bits in all matching digits.

Having identified similarity by PE Mode State 3, the program would now make the final of several possible exits to the switch test rendezvous point. This corresponds to point RONDO on the flow chart. At this point consider the associative memory program in the light of the original definition. The unknown function is satisfied by the ambiguity of DW magnitude, sense switch settings, mode states, etc. The unknown data is the undefined contents of 1024 PE data words. Finally, the unknown locations are illustrated in that, at point RONDO or other points, data categories based on magnitude or similarity could be assigned an exit into subsequent processing unique to their classification without directly addressing a specific location or set of location in the computer memory.



1		2		9		10		15		16		21		22		ADDRESS AND COMMENTS												72		73		80	
LOCATION		INST.		OPTIONS																						SEQUENCE							
START		MSI										0		M → MO.																			
		TMR												MASTER																			
		TRP												B																			
		SR												DW																			
		TPM												TEMP1																			
		TZP												O'S → P.																			
		MSE		0								3		DW = MW, MO → M3.																			
		MSPG		0								2		DW > MW, MO → M2.																			
		MSI		0								1		DW < MW, MO → M1.																			
SENSE		TSSW												BEGIN SWITCH SENSING.																			
		WNIM												C0N1																			
		TMX												C0N6																			
		JWZ												K1, 1																			
K1		MSI		3								0		IF SSO ON, → M+1.																			
K1+1		TSSW												EQUALITY BACK TO MO.																			
		WNIM												C0N2																			
		JWZ												K2, 1																			
K2		MSI		2								0		GREATER BACK TO MO.																			
K2+1		TSSW																															
		WNIM																															
		JWZ												K3, 1																			
K3		MSI		1								0		DW < MW; M → MO.																			
K3+1		TSSW																															
		WNIM																															
		JWZ												K4, 1																			
K4		JP												R0ND0																			
K4+1		MSI		1		2						0		SS3 OFF, OMIT BIT MATCH TEST.																			
		JNA		3										R0ND0																			
		TRP												B																			
		PAR												DW																			

20004-1



CODING FORM

13-5

PROGRAM: ASSOC. MEMORY MODEL DATE:

PROGRAMMER: DEL GUSENIUS

PAGE 1 OF 3



CODING FORM

PROGRAM: ASSOC. MEMORY MODEL DATE:

PROGRAMMER: DEL GUSENIUS

PAGE 2 OF 3

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22	72 73	80
			TPM		TEMP1	
			TRP		B	
			POR		DW	
			NP			
			POR		TEMP1	
			TZNM		TEMP1	
			TZNM		TEMP2	
		LOOP	TZQ			
			EPQ			
			MLSQ0	0	1	
			AP	1	CONS	
			AP		TEMP1	
			TPM		TEMP1	
			MSI	1	0	
			TZP	0		
			AP		CONS	
			AP		TEMP2	
			TPM		TEMP2	
			MSE	0	2CONS	
			JNA	2	EXIT1	
			EPQ	0		
			SPRL		1	
			JP		LOOP	
		EXIT1	MSI	2	0	
			TZP	0		
			TRQ		TEMP1	
			SQC	4		
			MSLQ	0	1	
			JNA	1	KS	
		KS	MSI			





CODING FORM

13-7/13-8

PROGRAM: ASSOC. MEMORY MODEL DATE:

PROGRAMMER: DEL GUSENIUS

PAGE 3 OF 3

T	LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS	SEQUENCE			
	1	2	9	10	15	16		21	22	72	73
	K5+1		SQC		1		19				
			MSLQØ		1		2				
			JNA		2		K6, 1				
	K6		MSI		1		2				
	K6+1		SQC		2		19				
			MSLQØ		2		3				
			JNA		3		K7, 1				
	K7		MSI		2		3				
	K7+1		SQC		3		19				
			MSI		12		0				
			MSLQØ		3		1				
			JNA		1		K8, 1				
	K8		MSI		3		1				
	K8+1		SQC		1		19				
			MSLQØ		1		2				
			JNA		2		K9, 1				
	K9		MSI		1		2				
	K9+1		MSI		13		0				
			MSI		2		3				
			JP				RØNDØ				
	RØNDØ										
							PE(S) WITH GNMB → M3.				
							SELECTED DATA NOW MARKED.				
							RENDEZVOUS POINT, ALL ROUTINES.				

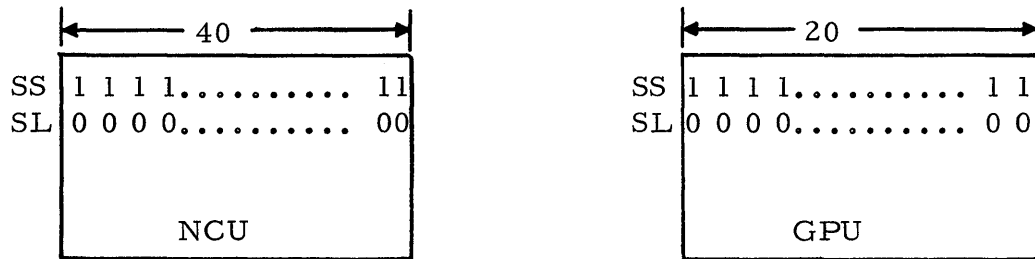
2000A-1

14. SPECIAL REGISTERS

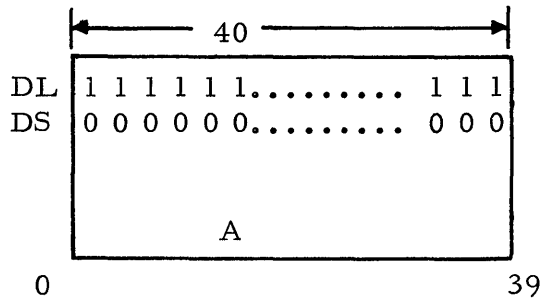
The SOLOMON II System has the following special registers available:

M ϕ	Machine Option Register (geometric control and edge connections)
CG	Column Geometric Register
RG	Row Geometric Register
B	Broadcast Register (Option No. 1 or No. 2)
SL	Sense Lights Register
SS	Sense Switch Register
DL	Data Lights on Arithmetic Unit Register
DS	Data Switches on Arithmetic Unit Register

The SL and SS Registers are located in the following positions:



The DL and DS Registers are located in the following positions:



15. SOLOMON II CODING OF BAROTROPIC MODEL WEATHER STUDY

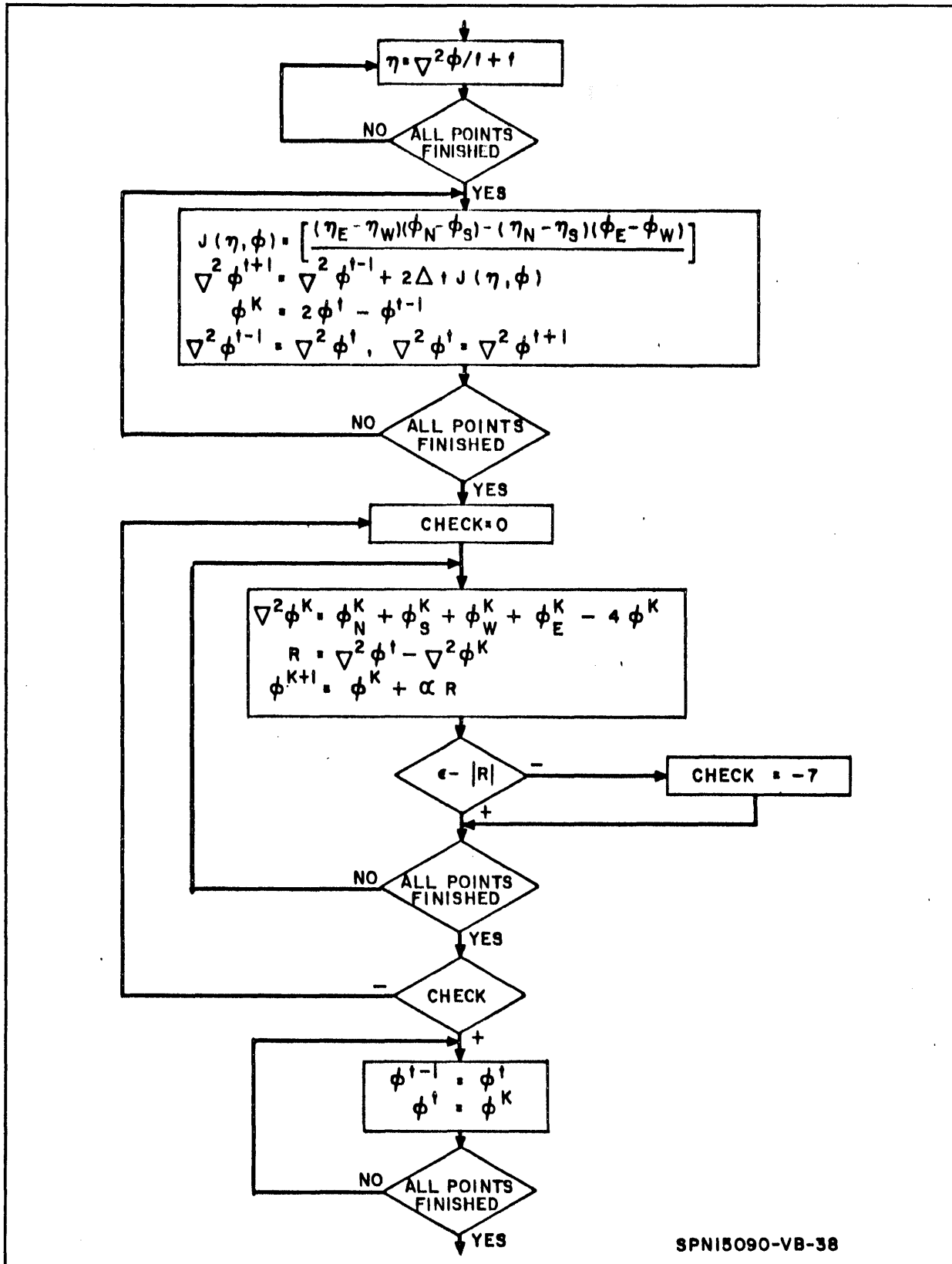
This section deals with a recent weather study program written for the SOLOMON II System. The new program is a follow-up to the SOLOMON program presented for the Barotropic Model; Technical Memorandum No. 14, 6/12/63, (see flow chart figure 15-1). Unlike the SOLOMON System, SOLOMON II is a single address machine of fixed word length (20 bits). Add time has been decreased by a factor of 10 and multiply time by more than a factor of 42. New instructions have been added and the mnemonics of the old instructions have been changed. In short, the present program is considerably different from the old one.

The data for the first data point of each PE is programmed and included in this section. Programming for the remaining points is similar. In the previous program, a four-point-per-PE allocation was used. The same is true for this case.

The execution time for the present program is obtained by multiplying the times for the appropriate sets of instructions by 4 and adding in the times for operations pertaining to all points. The computing time for the IBM 7090 was cited in the previous report as approximately 1.61×10^6 microseconds. The SOLOMON II computing time for this case is 804 microseconds. This gives an overall speed ratio of more than 2000/1 in favor of the SOLOMON II System.

In this program, an attempt was made to keep the same variable names for quantities as were used in the previous report. On the first page, η is computed for point No. 1. The second page has the instructions for computing $J(\eta, \phi)$, $\nabla^2 \phi^{k+1}$ and ϕ^k for point No. 1. The subscripts are adjusted in





SPN15090-VB-38

Figure 15-1. Flow Chart



the variables for later use. The relaxation process is programmed for the first point on page 3. The fourth page contains the test for convergence in all PE's while the last page lists the instructions for resetting the subscripts. The execution times for individual instructions and sets of instructions are listed in the program.



W SOLOMON II CODING FORM

PROGRAM: WEATHER STUDY

DATE:

PROGRAMMER: R JAMES BENNETT

PAGE 1 OF 6

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22		72 73 80
*					COMPUTE: $\eta = \nabla^2 \phi / f + f$	TIME
		TRQ	A		MODE WORD FOR I	22
		MSQ	A		" " " "	22
		TRQ	I		MODEL I	22
		MIQ	I		$f^{-1} i_j$	130
		AP	I		f_{ij}	22
		IMP	I		η_{ij}	22
						240 SEC

20004-1





SOLOMON II CODING FORM

PROGRAM: WEATHER STUDY

DATE:

PROGRAMMER: R JAMES BENNETT

PAGE 3 OF 6

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE				
1	2	9	10	15	16	21	22	72	73	80
X										
		TRP		1		MDELI		$\nabla^2 \phi^T \rightarrow$		22
		TPM		1		DELI		$\nabla^2 \phi^T - 1$		22
		TQM		1		MDELI		$\nabla^2 \phi^{T+1} \rightarrow \nabla^2 \phi^T$		22
										98.2 μ SEC

200041





SOLOMON II CODING FORM

PROGRAM: WEATHER STUDY		DATE:		PROGRAMMER: R JAMES BENNETT		PAGE 4 OF 6		
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE
1 2	9 10	15 16	21 22					72 73 80
X				RELAXATION				TIME
	RJBI	TZNM	A	CHECK				22
		TRQ	A	MODE WORD FOR 1				22
		MSQ	A	" " " "				22
		TRP	1	KPHI1				22
		SPLA	1	2				22
		TPM	1	NDELIK				22
		TRP	1	KPHI2				22
		AP	1	KPHI3				22
		AP	1	KPHI2				22
		AP	1	KPHI3				22
		SR	1	NDELIK				22
		TPM	1	NDELIK				22
		TRP	1	NDELI				22
		SR	1	NDELIK				22
		TMP		EPS				20
		TPM	1	RI				22
	1	MSDIAL	1	2B, EPS				22
		QSM	2					22
		TQM	2	CHECK				22
		MSI	2					22
		TM3		ALPHA				20
		TRQ	1	RI				22
		MSQ	1	B, ALPHA				130
		AP	1	KPHI1				22
		TPM	1	KPHI1				22

2000A-1

65.4 SEC



SOLOMON II CODING FORM

15-8

PROGRAM: WEATHER STUDY DATE:

PROGRAMMER: R JAMES BENNETT

PAGE 5 OF 6

LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS												SEQUENCE				
1	2	9	10	15	16	21	22													72	73	80
X								COMM LOOP														
			TRQ		A			CHECK														22
			MSG		A																	22
			JMD		Z			RJB1														30
																						7.4
																						μSEC.

20004-1





SALOMON II CODING FORM

15-9/15-10

PROGRAM: WEATHER STUDY DATE:

PROGRAMMER: R. JAMES BENNETT

PAGE 6 OF 6

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22		72 73 80
*					REPLACEMENT	TIME
		TRQ	A		MODE WORD FOR I	22
		MSG	A		" " " "	22
		TRP	I		PHI I $\phi^E \rightarrow$	22
		TPM	I		PHI I ϕ^{E-1}	22
		TRP	I		KPHI I $\phi^K \rightarrow$	22
		TPM	I		MPHI I ϕ^E	22
						132 μ SEC

Z0004-1

16. SOLOMON II AS APPLIED TO A PRIMITIVE EQUATION MODEL

This section deals with the handling of a primitive equation model by the SOLOMON II System in connection with numerical weather forecasting. This primitive equation model, while being of a simple nature, was felt to adequately illustrate many of the various types of computation typically required by atmospheric circulation models in general. Included will be the following main topics: (a) presentation of the model and the method of solution; (b) detailed coding of this model on the SOLOMON II System with time estimate; and (c) a full discussion on the procedures used by the SOLOMON II System.

For purposes of this report, emphasis is being placed on the finite difference method of integrating the hydrodynamic equations. However, some attention will be given to an alternate method of prediction which involves computations in wave number space. This method of prediction will, in general, require a much different scheme than that using the finite difference technique, since a given wave component is capable of interacting with many other wave components rather than merely the nearest four or eight neighbors.

In the first SOLOMON technical report on Numerical Weather Prediction (SOLOMON Project Technical Memorandum No. 14) the Barotropic Model a two address machine was presented and its handling of the problem described. This report deals with the SOLOMON II System which is a single address machine but whose instruction execution times have been significantly reduced.

16.1 BACKGROUND INFORMATION

Research in the field of numerical weather forecasting has provided methods of daily routine weather prediction which have given reasonably good



results in forecasting the upper and surface flow patterns, despite various simplifying assumptions in order to make the dynamic equations solvable. These methods have largely consisted of taking the basic equations of motion and continuity, utilizing assumptions (such as the hydrostatic and geostrophic equations) and finally filtering the equations to remove high speed and frequency gravity and sound waves from the solutions obtained. This last step was considered necessary to allow a longer time step in the forecast equation or equations and still retain computational stability. This filtering can take on several forms depending on the type of filtering desired. In fact, two common filtering procedures are the two equations mentioned above, the hydrostatic and the geostrophic equations. Other filtering procedures are the utilization of various appropriate smoothing operators which are applied to the derivatives of the dependent variables and sometimes to the variables themselves, chosen in such a way as to eliminate the high frequency waves and still retain the solutions containing the longer and more meteorologically significant (Rossby) waves, unchanged. In general, these computations were carried out on a hemispherical rather than a global scale.

However at present, more and more attention is being turned to the "Primitive Equations" (the basic hydrodynamic equations of the atmospheric circulation in their original form without filtering). These primitive equations take on different forms depending upon the model being used but basically attempt to represent the following atmospheric effects:

- a. Friction effects on the fluid motion caused by ground and wind shear (i. e., air and ground interface; air to air interface due to wind shear)
- b. Heating effects on the general circulation. These effects can be divided up into the following categories:
 - (1) Short wave radiation (radiation which enters the atmosphere and is absorbed by ground and air)
 - (2) Long wave radiation (radiation which leaves earth's surface, part of which is absorbed by atmosphere)

(3) Boundary layer convection (small scale convection at boundary layers)

(4) Tropospheric convection (larger scale convection throughout atmosphere)

c. Atmospheric moisture content (Heat of evaporation and or condensation)

d. Terrain effects (topography)

e. Vertical advection (motion or flux in the vertical)

In a more complete general circulation model, the above mentioned effects are accounted for in part by the following descriptive equations:

a. The equations of motion (frictional effects, terrain effects, vertical motion)

b. Equation of continuity (vertical advection, surface pressure tendency)

c. Thermodynamic equation (atmospheric heating by all processes except condensation/evaporation)

d. Equation of moisture (heat gain or loss due to condensation/evaporation)

The primitive equations, although in existence for many years, have never (until recently) been considered for daily numerical weather prediction for four important reasons. These reasons are:

a. The primitive equations require a large data accumulation and input, for several levels. The present system of data collection is not nearly extensive enough to permit an accurate reconstruction of the initial state of the atmosphere, particularly for the large number of levels required for most models presently under consideration. However for research considerations, this problem is handled by simply setting the general circulation in a state of rest (i. e., calm winds, constant temperature, constant pressure in the horizontal and constant variation with height, no vertical motion, no heating, throughout the atmosphere), applying the heating effects, and



finally observing the resulting circulation patterns caused by the heating induced. In theory, this process would be repeated for several variations of heating distributions. For routine forecasting, a minimum of levels must be used to utilize the limited data now available for analysis. For research work, this presents no problem, since most or all of the work is done on a theoretical basis.

b. Since the primitive equations are essentially unfiltered (short-high frequency waves retained), the time step to ensure computational stability is necessarily much smaller than in the filtered equations. In the filtered equations, it is possible to use approximately a 1/2-hour time step and still obtain a nondiverging solution. In the primitive equations, this time step must be cut down to 5 or at the most 10 minutes to ensure a stable computation. Obviously, the usage of so short a time step requires much more computation to obtain a forecast for the same future time period than in the filtered case. This problem can be solved by utilizing a faster parallel-type computer capable of handling these computations in a minimum of time.

c. A third problem is the absolute magnitude of the computations involved on a hemispherical or global grid. For example, a hemispherical grid network consists of 10,000 grid points. Moreover, there are six levels to be considered in the model plus the surface. This gives a total of (10,000) (7) or 70,000 grid points to be considered. At each grid point, we are to carry 5 parameters plus storage for certain constants (for each grid point). These would include; coriolis parameter, map scale factor, various trigonometric functions - if a change in the coordinate system is desired - and certain absolute constants. Also, there must be a working storage to provide space for the logical and arithmetic operations to take place at each point. The computational problem becomes tremendous even for a fairly simple primitive equation model. It is clear that a computer of a parallel nature is extremely desirable particularly when a great many of the computations involve a large network where the same computation is performed

at each point, excluding the boundary points. These are treated separately as explained in SOLOMON Project Technical Memorandum No. 14.

d. Direct measurement of the heating effects is extremely difficult if not impossible at present. Hence researchers are led to empirical formulae for the various types of heating effects or must leave them out entirely. Hopefully, a faster and more efficient computer will allow researchers to discover and formulate more realistic expressions for the rate of change of heating as it actually occurs in the real atmosphere.

16.2 THE PRIMITIVE EQUATION MODEL

As a preliminary consideration on the primitive equations, it was felt that a three parameter model would suffice for the sake of simplicity. This model will illustrate many of the computations involved in a primitive equation study without considering full scale atmospheric effects. As an added feature, a somewhat different finite differencing scheme will be considered to illustrate the flexibility of the SOLOMON II System. This finite differencing scheme (Eliassen and Platzman) involves essentially a staggered grid system which employs not only the nearest four neighboring grid points but also the four corner points as well, for the finite difference formulation of the dependent variables and their derivatives. The usual centered time, center space finite differencing method will be used, conventional in many forecast problems of this nature.

The primitive equation model chosen is one containing three dependent variables and having a free surface, otherwise known as the Hydrostatic System. This model has the following simplifying assumptions:

- a. Hydrostatic balance
- b. No frictional effects
- c. Homogeneous and incompressible atmosphere, with a free surface
- d. Heating effects have been disregarded

From the basic vector equation of motion in the atmosphere:

$$\frac{d\vec{V}}{dt} = -\alpha \nabla p - 2 \vec{\Omega} \times \vec{V} + g$$



the following scalar equations of motion in rectangular coordinates can be derived

$$\frac{\partial U}{\partial t} + U \frac{\partial U}{\partial X} + V \frac{\partial U}{\partial Y} - f V + \frac{\partial \phi}{\partial X} = 0$$

$$\frac{\partial V}{\partial t} + U \frac{\partial V}{\partial X} + V \frac{\partial V}{\partial Y} + f U + \frac{\partial \phi}{\partial Y} = 0$$

The third equation necessary to solve this system is the continuity equation:

$$\frac{1}{\rho} \frac{d\rho}{dt} = - \nabla \cdot \mathbf{V}_3$$

But for our system using the assumptions of incompressibility and free surface, the continuity equation can be put into the form:

$$\frac{\partial \phi}{\partial t} + U \frac{\partial \phi}{\partial X} + V \frac{\partial \phi}{\partial Y} + \phi \left(\frac{\partial U}{\partial X} + \frac{\partial V}{\partial Y} \right) = 0$$

where

U = horizontal component of velocity (west to east)

V = horizontal component of velocity (south to north)

X = coordinate distance (west to east)

Y = coordinate distance (south to north)

ϕ = geopotential height, given by $g Z$

Z = height of an isobaric surface

f = coriolis parameter given by $2 \Omega \sin \theta$

$\vec{\Omega}$ = angular velocity vector of earth's rotation

θ = latitude

α = specific volume

ρ = dry air density

g = gravitational acceleration

∇ = dell operator

p = atmospheric pressure

\mathbf{V} = horizontal velocity vector

\mathbf{V}_3 = three dimensional velocity vector



For the SOLOMON II System, it is extremely convenient to use the spherical coordinate system to ensure proper continuity at the equator required by a global model. This has the added feature of allowing mass transport across the equator without the fitting and interpolation procedure of rectangular grid systems at the equator. The two polar points where singularities occur are to be handled separately and in a manner to be outlined later. With this adoption, the equations of motion can be expressed in spherical coordinates as:

$$a \ddot{\lambda} \cos \theta = 2 a \dot{\theta} (\dot{\lambda} + \Omega) \sin \theta - \frac{1}{a \cos \theta} \frac{\partial \phi}{\partial \lambda}$$

$$a \ddot{\theta} = - a \dot{\lambda} \cos \theta (\dot{\lambda} + 2 \Omega) \sin \theta - \frac{1}{a} \frac{\partial \phi}{\partial \theta}$$

where

λ = longitude

θ = latitude

a = radius of earth (taken as a constant)

Ω = angular velocity of earth's rotation

The total derivative operator is now given by:

$$\left(\dot{} \right) = \frac{d}{dt} = \frac{\partial}{\partial t} + \dot{\lambda} \frac{\partial}{\partial \lambda} + \dot{\theta} \frac{\partial}{\partial \theta} + \dot{\rho} \frac{\partial}{\partial \rho}$$

However in this coordinate system the operator $\frac{\partial}{\partial \rho}$ vanishes since $\dot{\lambda}$ and $\dot{\theta}$ can be taken to be independent of the vertical coordinate ρ . To provide consistency throughout the system the following terms have been omitted.

$2 \Omega \dot{a} \cos \theta$ - interpreted physically as the vertical component of coriolis term

$2 \dot{a} \dot{\theta}$

interpreted physically as inertial terms

$2 \dot{\lambda} \dot{a} \cos \theta$

The equation of continuity in spherical coordinates becomes:

$$\frac{\partial \phi}{\partial t} = - \nabla \cdot (\phi \vec{V}) = - \frac{\partial}{\partial \lambda} (\phi \dot{\lambda}) - \sin \theta \frac{\partial}{\partial \theta} (\phi \dot{\theta} \cos \theta)$$

Here and in the equations of motion, differences of gravity with height have been neglected. Also for purposes of expanding the above equations, the following spherical relationships are available:

$$U = a \dot{\lambda} \cos \theta \quad \partial Y = a \partial \theta$$

$$V = a \dot{\theta} \quad \partial X = a \cos \theta \partial \lambda$$

Hence the equations for the model can be expanded into the following form:

$$\frac{\partial U}{\partial t} + \frac{U}{a \cos \theta} \frac{\partial U}{\partial \lambda} + \frac{V}{a} \frac{\partial U}{\partial \theta} - \frac{U V \tan \theta}{a} - f V + \frac{1}{a \cos \theta} \frac{\partial \phi}{\partial \lambda} = 0$$

$$\frac{\partial V}{\partial t} + \frac{U}{a \cos \theta} \frac{\partial V}{\partial \lambda} + \frac{V}{a} \frac{\partial V}{\partial \theta} - \frac{U^2 \tan \theta}{a} + f U + \frac{1}{a} \frac{\partial \phi}{\partial \theta} = 0$$

$$\frac{\partial \phi}{\partial t} + \frac{U}{a \cos \theta} \frac{\partial \phi}{\partial \lambda} + \frac{V}{a} \frac{\partial \phi}{\partial \theta} + \frac{\phi}{a} \left(\frac{1}{\cos \theta} \frac{\partial Y}{\partial \lambda} + \frac{\partial V}{\partial \theta} - V \tan \theta \right) = 0$$

In meteorological problems of this nature, it is usually desirable to use a conformal mapping transformation in order to map the solution from a curved to a flat surface. For this particular case, a Mercator Projection was selected for convenience at the equator. This transformation can be made by the following relationships:

$$X = a \lambda \quad Y = -a \ln h$$

$$V = a \dot{\lambda} \cos \theta = M^{-1} \dot{X} \quad V = a \dot{\theta} = M^{-1} \dot{Y}$$

$$h = \cos \theta (1 + \sin \theta)^{-1}$$

$$M = \sec \theta$$

Putting these relationships into the spherical equations yields the following system expressed in map coordinates:

$$\frac{\partial U}{\partial t} = -M \left(U \frac{\partial U}{\partial X} + V \frac{\partial U}{\partial Y} + \frac{\partial \phi}{\partial X} \right) + f V \left(1 + \frac{M}{2\Omega a} U \right)$$

$$\frac{\partial V}{\partial t} = -M \left(U \frac{\partial V}{\partial X} + V \frac{\partial V}{\partial Y} + \frac{\partial \phi}{\partial Y} \right) - f U \left(1 + \frac{M}{2\Omega a} U \right)$$

$$\frac{\partial \phi}{\partial t} = -M \left(\frac{\partial}{\partial X} (U \phi) + \frac{\partial}{\partial Y} (V \phi) - \frac{f}{2\Omega} V \phi \right)$$

16.3 FINITE DIFFERENCES

As was mentioned previously, the finite differencing scheme to be used in this illustration is the one proposed by Eliassen and Platzman, the "staggered grid" system. Basically, this system consists of "staggering" the dependent variables in a systematic manner over the entire network. This is illustrated by the following diagram. Note that not all parameters are stored at all points in the network but are staggered in a definite pattern throughout the entire grid system (see figure 16-1).

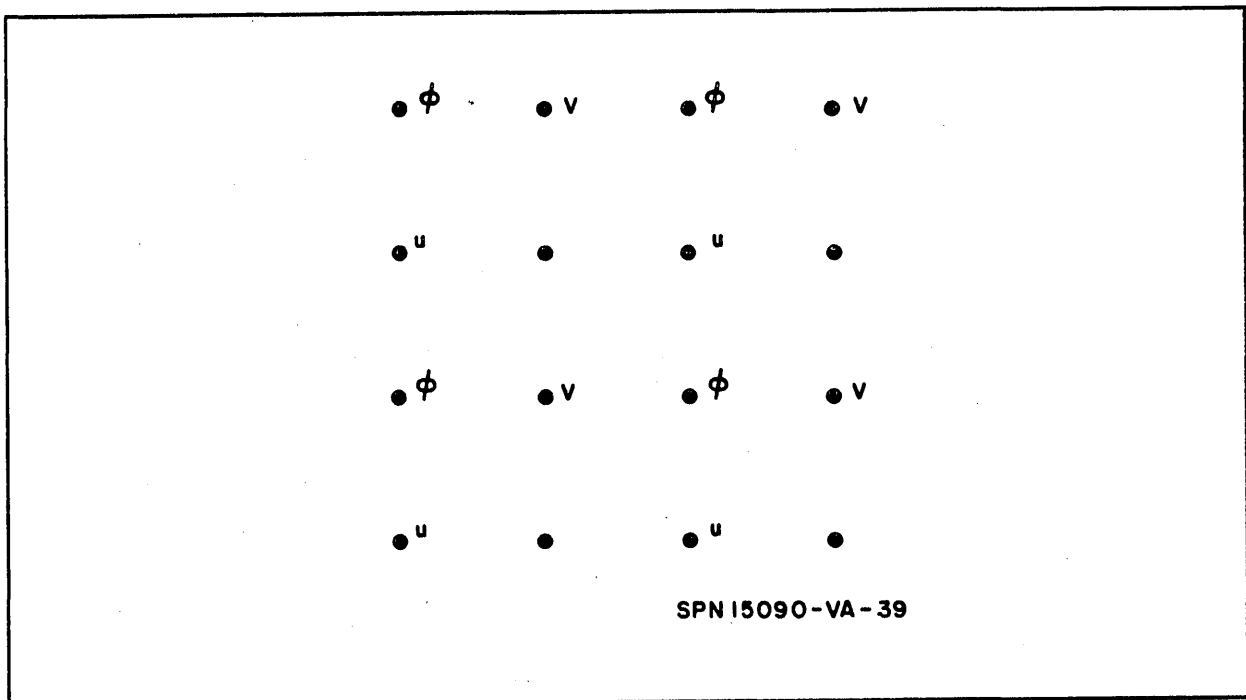


Figure 16-1. Staggered Grid System

Another feature added to this scheme was to systematically stagger the dependent variables in time as well as in space. This was done by setting for each integral time step (i. e., $0, \Delta t, 2 \Delta t, 3 \Delta t, \dots, n \Delta t$) the grid distribution shown above for unprimed variables $U, V,$ and ϕ . For the $1/2$ integral time steps (i. e., $1/2 \Delta t, 3/2 \Delta t, 5/2 \Delta t, \dots, (1 + 1/2 \Delta t)$) the grid distribution shown below for the primed variables, $U', V',$ and ϕ' was used. Hence, there are two variable distributions depending upon time step being

considered. The dependent variables alternate positions on the grid with each 1/2 integral time step. This system has been shown to be more efficient than a grid system having all dependent variables stored at all grid points for all times. The variation of this grid system with space and time is shown in figure 16-2 below.

For the purpose of this problem, the following distributions were used for each variable being computed. From these, the equivalent staggered grid finite difference equations for the hydrostatic system were derived. There are two sets of equations with three per set; one set for the unprimed variables and the other set for the primed variables. These are as follows:

(Δ = grid size increment)

$$\begin{array}{cccccc} \dot{U}V' & \dot{\phi}' & \dot{U}V' & -\dot{1}\dot{1} & \dot{0}\dot{1} & \dot{1}\dot{1} \\ \dot{\phi} & \dot{U}'V & \dot{\phi} & -\dot{1}\dot{0} & \dot{0}\dot{0} & \dot{1}\dot{0} \\ \dot{U}V' & \dot{\phi}' & \dot{U}V' & -\dot{1}\dot{-1} & \dot{0}\dot{-1} & \dot{1}\dot{-1} \end{array}$$

$$U'_{00n} = U'_{00n-1} - M_{00} \left(\frac{\Delta t}{\Delta}\right) \left\{ \phi_{10} - \phi_{-10} + \frac{1}{4} \left[(U_{11} + U_{-11})(U_{11} - U_{-11}) + (U_{1-1} + U_{-1-1})(U_{1-1} - U_{-1-1}) \right] + \frac{1}{2} V_{00} (U_{11} - U_{1-1} + U_{-11} - U_{-1-1}) \right\} + V_{00} \Delta t f_{00} \left[1 + \frac{M_{00}}{8\Omega a} (U_{-11} + U_{1-1} + U_{11} + U_{-1-1}) \right]$$

$$\begin{array}{cccccc} \dot{U}'V & \dot{\phi} & \dot{U}'V & -\dot{1}\dot{1} & \dot{0}\dot{1} & \dot{1}\dot{1} \\ \dot{\phi}' & \dot{U}V' & \dot{\phi}' & -\dot{1}\dot{0} & \dot{0}\dot{0} & \dot{1}\dot{0} \\ \dot{U}'V & \dot{\phi} & \dot{U}'V & -\dot{1}\dot{-1} & \dot{0}\dot{-1} & \dot{1}\dot{-1} \end{array}$$

$$V'_{00n} = V'_{00n-1} - M_{00} \left(\frac{\Delta t}{\Delta}\right) \left\{ \phi_{01} - \phi_{0-1} + \frac{1}{2} U_{00} (V_{11} - V_{-11} + V_{1-1} - V_{-1-1}) + \frac{1}{4} \left[(V_{11} + V_{1-1})(V_{11} - V_{1-1}) + (V_{-11} + V_{-1-1})(V_{-11} - V_{-1-1}) \right] \right\} - U_{00} \Delta t f_{00} \left(1 + \frac{M_{00}}{2\Omega a} V_{00} \right)$$

$$\begin{array}{cccccc} \dot{\phi} & \dot{U}'V & \dot{\phi} & -\dot{1}\dot{1} & \dot{0}\dot{1} & \dot{1}\dot{1} \\ \dot{U}V' & \dot{\phi}' & \dot{U}V' & -\dot{1}\dot{0} & \dot{0}\dot{0} & \dot{1}\dot{0} \\ \dot{\phi} & \dot{U}'V & \dot{\phi} & -\dot{1}\dot{-1} & \dot{0}\dot{-1} & \dot{1}\dot{-1} \end{array}$$

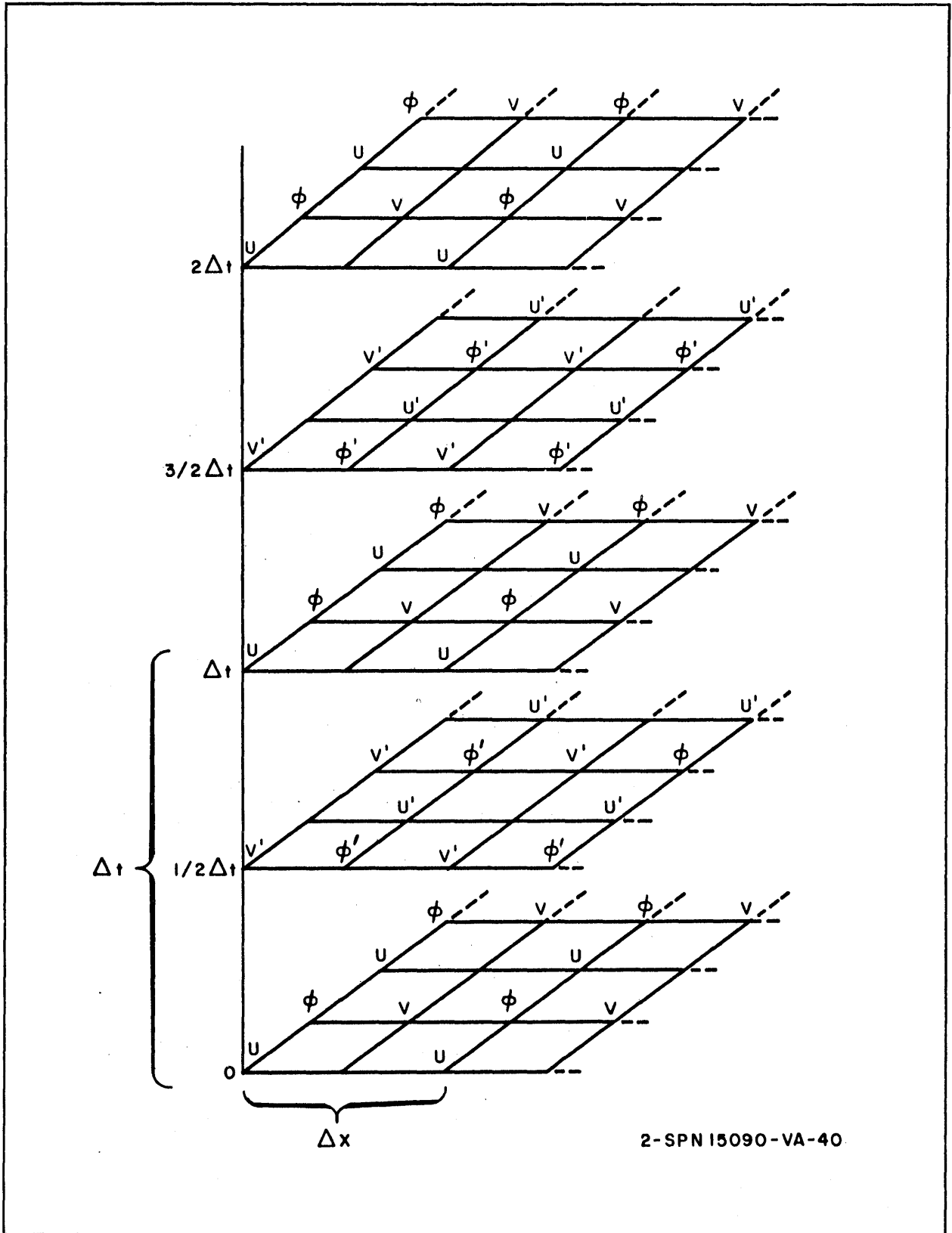


Figure 16-2. Staggered Grid System

$$\begin{aligned} \phi'_{00n} = & \phi'_{00n-1} - \frac{1}{2} M_{00} \left(\frac{\Delta t}{\Delta}\right) \left\{ U_{10} (\phi_{11} + \phi_{1-1}) - U_{-10} (\phi_{-11} + \phi_{-1-1}) \right. \\ & + V_{01} (\phi_{11} + \phi_{-11}) - V_{0-1} (\phi_{1-1} + \phi_{-1-1}) - \frac{f_{00}}{16\Omega} (V_{01} + V_{0-1}) (\phi_{-11} \\ & \left. + \phi_{1-1} + \phi_{11} + \phi_{-1-1}) \right\} \end{aligned}$$

$$\begin{array}{cccccc} \dot{U}'V & \dot{\phi} & \dot{U}'V & -\dot{11} & \dot{01} & \dot{11} \\ \dot{\phi}' & \dot{U}'V' & \dot{\phi}' & -\dot{10} & \dot{00} & \dot{10} \\ \dot{U}'V & \dot{\phi} & \dot{U}'V & -\dot{1-1} & \dot{0-1} & \dot{1-1} \end{array}$$

$$\begin{aligned} U_{00n+1} = & U_{00n} - M_{00} \left(\frac{\Delta t}{\Delta}\right) \left\{ \phi'_{10} - \phi'_{-10} + \frac{1}{4} \left[(U'_{11} + U'_{-11}) (U'_{11} - U'_{-11}) \right. \right. \\ & \left. \left. + (U'_{1-1} + U'_{-1-1}) (U'_{1-1} - U'_{-1-1}) \right] + \frac{1}{2} V'_{00} (U'_{11} - U'_{1-1} + U'_{-11} \right. \\ & \left. - U'_{-1-1}) \right\} + V'_{00} f_{00} \Delta t \left[1 + \frac{M_{00}}{8\Omega a} (U'_{11} + U'_{1-1} + U'_{-1-1} + U'_{-11}) \right] \end{aligned}$$

$$\begin{array}{cccccc} \dot{U}'V' & \dot{\phi}' & \dot{U}'V' & -\dot{11} & \dot{01} & \dot{11} \\ \dot{\phi}' & \dot{U}'V & \dot{\phi}' & -\dot{10} & \dot{00} & \dot{10} \\ \dot{U}'V' & \dot{\phi}' & \dot{U}'V' & -\dot{1-1} & \dot{0-1} & \dot{1-1} \end{array}$$

$$\begin{aligned} V_{00n+1} = & V_{00n} - M_{00} \left(\frac{\Delta t}{\Delta}\right) \left\{ \phi'_{01} - \phi'_{0-1} + \frac{1}{2} V'_{00} (V'_{11} - V'_{-11} + V'_{1-1} \right. \\ & \left. - V'_{-1-1}) + \frac{1}{4} \left[(V'_{11} + V'_{1-1}) (V'_{11} - V'_{1-1}) + (V'_{-11} + V'_{-1-1}) (V'_{-11} \right. \right. \\ & \left. \left. - V'_{-1-1}) \right] \right\} - U'_{00} f_{00} \Delta t \left[1 + \frac{M_{00}}{2\Omega a} V'_{00} \right] \end{aligned}$$

$$\begin{array}{cccccc} \dot{\phi}' & \dot{U}'V' & \dot{\phi}' & -\dot{11} & \dot{01} & \dot{11} \\ \dot{U}'V & \dot{\phi} & \dot{U}'V & -\dot{10} & \dot{00} & \dot{10} \\ \dot{\phi}' & \dot{U}'V' & \dot{\phi}' & -\dot{1-1} & \dot{0-1} & \dot{1-1} \end{array}$$

$$\begin{aligned} \phi'_{00n+1} = & \phi'_{00n} - \frac{1}{2} M_{00} \left(\frac{\Delta t}{\Delta}\right) \left\{ U'_{10} (\phi'_{11} + \phi'_{1-1}) - U'_{-10} (\phi'_{-11} + \phi'_{-1-1}) \right. \\ & + V'_{01} (\phi'_{11} + \phi'_{-11}) - V'_{0-1} (\phi'_{1-1} + \phi'_{-1-1}) - \frac{f_{00}}{16\Omega} (V'_{01} + V'_{0-1}) (\phi'_{-11} \\ & \left. + \phi'_{1-1} + \phi'_{11} + \phi'_{-1-1}) \right\} \end{aligned}$$

It is worth mentioning that the finite differencing scheme used in this report (and subsequent coding) is one typically used in general circulation computations and experimentation. This is particularly true when the spherical-polar coordinate system is used, due to the convergence of the longitude lines near or at the polar regions. When making use of such a coordinate system, it is particularly desirable to use a finite differencing scheme which is not affected appreciably by grid size and still retain computational stability. Since, generally, the governing equations of a general circulation model are nonlinear in nature, the criterion for computational stability becomes even more critical than for the linear case.

As it turns out, the type of instability which has hampered general circulation computation the most is the so-called nonlinear computational instability of the finite difference analogue representing the physical system under consideration. This type of instability, upon investigation, was found to have the following characteristics:

- a. This instability appears only in nonlinear finite difference equations (finite difference analogue of the nonlinear partial differential equations)
- b. This type of instability cannot be avoided or postponed by simply reducing the time interval used in the forecast equations.
- c. The occurrence of this type of instability is, for all practical purposes, independent of the grid size being used.

Extensive research has been conducted in this area and schemes for eliminating this type of instability have been proposed for various situations. Fundamentally, these finite differencing methods are based on energy constraints imposed on the entire system (i. e., conservation of energy, kinetic and potential, or conservation of mean square vorticity). The staggered grid system plays an important part in these schemes when applied to the primitive equation models.

For the hydrostatic system, the sum of both potential and kinetic energy must be approximately conserved. From the above consideration, the



staggered grid technique used can be shown to be sufficient along with the usual centered time and spatial finite differencing forms to retain long-range nonlinear computational stability and continuity. The staggered grid system has other distinct advantages since it requires less storage than the usual types of grid system allocations and finally allows twice as many grid points for roughly the same amount of computation.

16.4 INITIAL DATA

To initiate the computational procedure, it is necessary to obtain an initial wind field (u, v) along with a geopotential height field for the given synoptic situation. Since, in general, upper wind data are either not available or are not sufficiently accurate for numerical forecasting purposes, a theoretical method must be used to obtain these data initially. It is possible using geostrophic and hydrostatic considerations to express a theoretical wind field that is in a state of balance with the existing geopotential height field. To do this, it is necessary to introduce the concept of a stream function or in simple terms, an air particle (wind) trajectory field. The balance equation relating the stream function and its corresponding geopotential height using the above assumptions, is:

$$\nabla \cdot (f \nabla \psi) - 2(\psi_{xy}^2 - \psi_{xx} \psi_{yy}) = \nabla^2 \phi$$

where

f = Coriolis parameter

ψ = stream function

∇ = horizontal dell operator

ϕ = geopotential height (obtainable from a weather map or upper atmospheric data)

k = unit vector in the vertical

The wind field can then be expressed in terms of this stream function by the formulae:

$$\vec{V} = \vec{k} \times \nabla \psi$$

or in component form:

$$U = -\psi_y \quad V = \psi_x$$

For many situations in numerical weather forecasting, a good degree of accuracy and realism is retained by omitting the nonlinear terms in the balance equation. Hence to a good degree of approximation, the initial stream function can be determined from the linear balance equation:

$$\nabla \cdot (f \nabla \psi) = \nabla^2 \phi$$

and the wind field as before by

$$U = -\psi_y \quad V = \psi_x$$

For consistency in this model, these relations for initialization must be transformed into spherical and Mercator Coordinates respectively. For the spherical transformation, the following identities can be employed:

$$\nabla^2 \psi = \frac{1}{a^2 \cos \theta} \frac{\partial}{\partial \theta} \left(\cos \theta \frac{\partial \psi}{\partial \theta} \right) + \frac{1}{a^2 \cos^2 \theta} \frac{\partial^2 \psi}{\partial \lambda^2}$$

$$\nabla^2 \phi = \frac{1}{a^2 \cos \theta} \frac{\partial}{\partial \theta} \left(\cos \theta \frac{\partial \phi}{\partial \theta} \right) + \frac{1}{a^2 \cos^2 \theta} \frac{\partial^2 \phi}{\partial \lambda^2}$$

$$\nabla f \cdot \nabla \psi = \frac{1}{a} \left[\sec^2 \theta \frac{\partial f}{\partial \lambda} \frac{\partial \psi}{\partial \lambda} + \frac{\partial f}{\partial \theta} \frac{\partial \psi}{\partial \theta} \right]$$

The linear balance equation in spherical coordinates is then:

$$\begin{aligned} f \left[\frac{1}{a^2 \cos \theta} \frac{\partial}{\partial \theta} \left(\cos \theta \frac{\partial \psi}{\partial \theta} \right) + \frac{1}{a^2 \cos^2 \theta} \frac{\partial^2 \psi}{\partial \lambda^2} \right] + \frac{1}{a} \frac{\partial f}{\partial \theta} \frac{\partial \psi}{\partial \theta} \\ = \frac{1}{a^2 \cos \theta} \frac{\partial}{\partial \theta} \left(\cos \theta \frac{\partial \phi}{\partial \theta} \right) + \frac{1}{a^2 \cos^2 \theta} \frac{\partial^2 \phi}{\partial \lambda^2} \quad \text{since } \frac{\partial f}{\partial \lambda} = 0 \end{aligned}$$

The transformed wind relations are:

$$u = -\frac{1}{a} \frac{\partial \psi}{\partial \theta} \quad v = \frac{1}{a \cos \theta} \frac{\partial \psi}{\partial \lambda}$$



Finally, with the aid of the Mercator Transformations, the system can be cast into the form:

$$\frac{\partial f}{\partial y} \frac{\partial \psi}{\partial y} + f \left(\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \right) = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2}$$

$$u = -\frac{1}{\cos \theta} \frac{\partial \psi}{\partial y} \quad v = \frac{1}{\cos \theta} \frac{\partial \psi}{\partial x}$$

The linear balance equation will be solved by a method equivalent to the Liebmann relaxation technique which consists of computing a residual from an initial guess, then applying a corrector formula (over-or-under-relaxation) to reduce the residual at each point as much as possible. This procedure is repeated until the residual is reduced to less than or equal to specified limit (measure of error between computed and true solution). For this situation, the residual is computed by the formula:

$$R_{ij}^m = f_{ij} \left(\psi_{i, j+1}^m + \psi_{i+1, j}^m + \psi_{i-1, j}^{m+1} + \psi_{i, j-1}^{m+1} - 4 \psi_{ij}^m \right) + (f_{ij+1} - f_{ij-1}) (\psi_{i, j+1} - \psi_{ij-1}) - F_{ij}$$

where

m = order of approximation

$$F_{ij} = \phi_{i+1, j} + \phi_{i-1, j} + \phi_{ij+1} + \phi_{i, j-1} - 4 \phi_{ij}$$

The predictor - corrector formula is given by:

$$\psi_{ij}^{m+1} = \psi_{ij}^m + \frac{\alpha}{4} R_{ij}^m$$

The test for convergence to the true solution can be written in the form

$$\epsilon - |R_{ij}^m| > 0$$

where

α is a relaxation coefficient generally in the range: $0.2 < \alpha < 2$

ϵ is a predetermined convergence criterion

The procedure given above is repeated until the convergence criterion given above is met at all points or until it has cycled a stipulated number of times (implies nonconvergence to solution). To initiate the relaxation procedure, a guess function is used. This consists, for this case, of the geostrophic relation:

$$\psi_{ij} = \frac{1}{f_{45}} \phi_{ij}$$

Once the stream function has been found, the wind field is then obtained by the finite difference formulas:

$$U = -\frac{1}{\Delta} M (\psi_{i, j+1} - \psi_{ij-1})$$

$$V = \frac{1}{\Delta} M (\psi_{i+1, j} - \psi_{i-1, j})$$

where for this example, both the stream function and the wind components are specified at the poles.

From the initialization procedure, the initial fields of U, V, and ϕ are obtained. To start the forecast, the following conditions are used: at time $t = 0$

$$U = U'$$

$$V = V'$$

$$\phi = \phi'$$

Now use the finite difference equations to solve for the primed variables at time $t = 1/2 \Delta t$. This is done by making the following substitution:

$$U'_{00n-1} = U'(t = 0)$$

$$V'_{00n-1} = V'(t = 0)$$

$$\phi'_{00n-1} = \phi'(t = 0)$$

and replacing Δt by $\Delta t/2$ for the first time step. All time steps thereafter, the normal finite difference equations are used, solving for the unprimed



and primed variables in succession. This is done for all parameters U, V, and ϕ ; U' , V' , and ϕ' .

16.5 AVERAGING PROCEDURE AT POLES

As was mentioned previously, a special treatment is required at the polar regions where singularities occur in this system. In this problem, new polar values for each variable are generated by averaging the grid point values closest to the pole (i. e., nearest two sets of latitude points) containing that particular variable. This is done after new values for that particular variable have been computed at all interior points. The new averaged value is then substituted in place of the old one. Expressed mathematically, this is:

$$\begin{aligned} \phi_{\rho} &= \sum_{i=1}^N \phi_i / N & \phi'_{\rho} &= \sum_{i=1}^N \phi'_i / N \\ U'_{\rho} &= \sum_{i=1}^N U_i / N & U'_{\rho} &= \sum_{i=1}^N U'_i / N \\ V'_{\rho} &= \sum_{i=1}^N V_i / N & V'_{\rho} &= \sum_{i=1}^N V'_i / N \end{aligned}$$

Where N = number of grid points containing a particular variable. Note, due to the staggered grid configuration, N will, in all cases, be equal to 1/2 the total number of grid points contained along a given latitude line (for this example N = 64). Also, due to the variable allocation on this grid system, this averaging procedure will take place on either the 1st or 2nd latitude circle nearest the pole but not on both simultaneously for a given variable.

16.6 SOLOMON II SYSTEM

This section will deal with the actual implementation of the SOLOMON II System to this particular problem. The concept of the Parallel Processing

Element array as applied to a simple forecasting model has already been presented in SOLOMON Project Technical Memorandum No. 14. For this case, the grid point allocation was set to four points per Processing Element. Since, in general, the current Processing Element array being considered is a 32 x 32 array, it was decided to set 8 staggered grid points per Processing Element. It was found that this allocation (4 latitude points across and 2 longitude points down per PE) would give a resolution of 2.8125 degrees. This would allow a grid network of 128 points around a latitude circle and 64 grid points along a longitude line from pole to pole. It was felt that this resolution was adequate for the purposes of this numerical forecasting problem. According to the staggered grid system, the grid point distribution per PE was used as shown in figure 16-3.

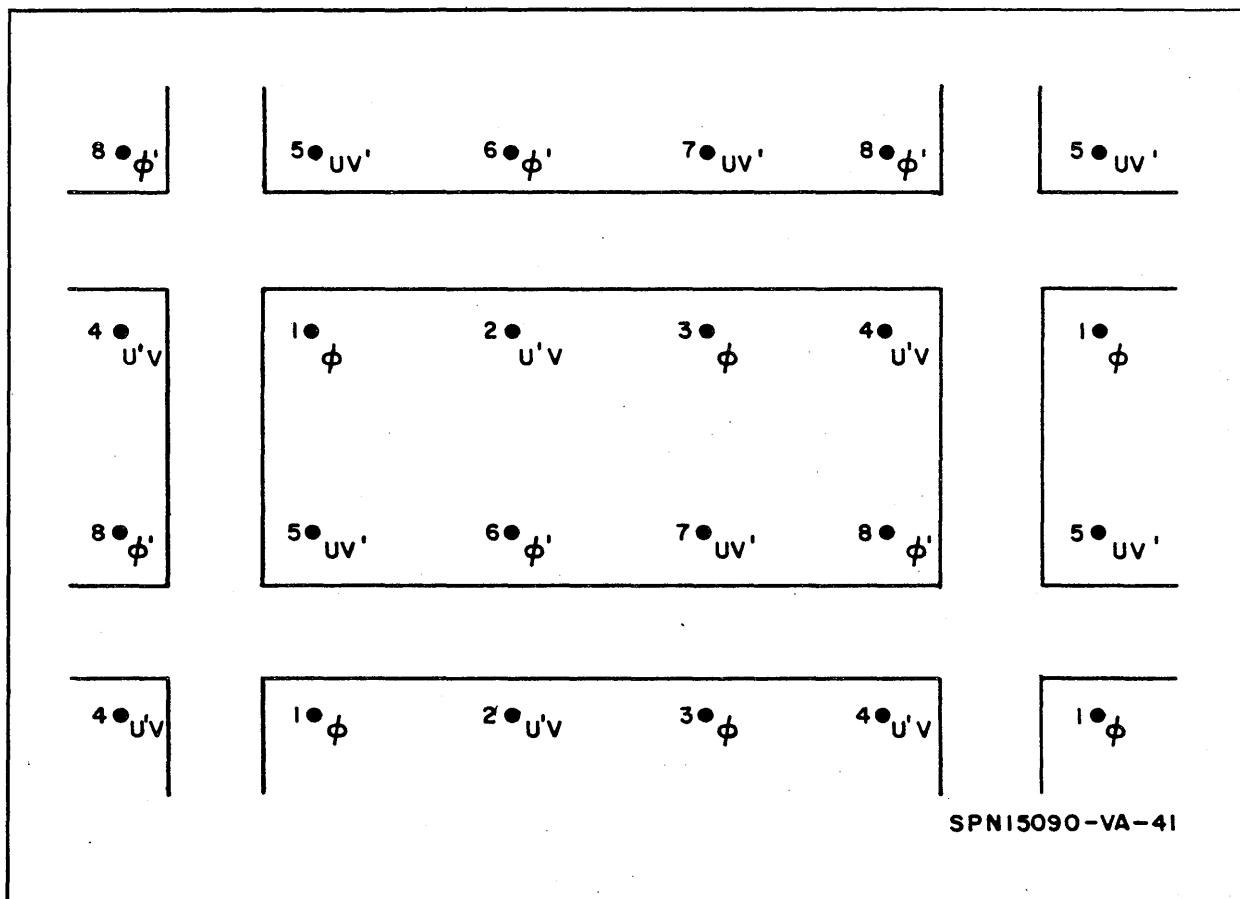


Figure 16-3. Grid Point - PE Allocation

With this allocation, the computations are programmed for each point within the PE. Due to the dependent variable distribution, the computations will not be the same for each point within the PE but will be the same for corresponding points in all other PE's in the network. Hence, the concept of parallel computation applies equally well to a staggered grid system as well as a conventional one. Therefore, the finite difference equations for this physical system are applied to their respective points in each PE until the computations are performed at every point within the PE (and hence all PE's).

16.6.1 Computing North and South Pole Average Values

Once each half time step, new average values of the U , V , and ϕ values in both the 0th and 31st rows must be computed for use as north and south pole values. What must be formed are 6 sums each with 64 summands. Several methods of achieving this result have been examined; the fastest one found is shown in the following program. Rows 0, 1, and 2 are used to compute \bar{U} , \bar{V} , and $\overline{\text{PHI}}$, respectively, for use as south pole values; rows 31, 30, and 29 are used to compute \bar{U} , \bar{V} , and $\overline{\text{PHI}}$ for use as north pole values.

The program has two parts. The first, given on page 1, computes the average of the pair of values stored within each PE and distributes these initial averages across rows 0, 1, and 2 and 31, 30, and 29. The second part, given on pages 2 and 3, computes the average of the 32 values stored across each of these rows and places the 6 average values in the L-Buffer. The program is written to minimize loss of significance; shifts are performed only after additions, and overflow is compensated for. Note here that in those operations that indicate an interest in overflow detection, $\text{AP}\phi$ for example, the last mode given is that to which PE's are to be set if they produce a result which overflows. Thus, the instruction

$\text{AP}\phi$ 20 TO


says, all PE's in mode 2, add the contents of TO to the contents of P; PE's in which overflow occurs, go to mode 0.




CODING FORM

PROGRAM: AVERAGE VALUES		DATE:		PROGRAMMER: M. L. GRAHAM		PAGE 1 OF 3			
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE	
1 2	9 10	15 16	21 22	23	24	25	26	72 73	80
*				D I S T R I B U T E D A T A					
		TRQ	A	M O D E				0 31	U
		MSQ	A					1 30	V
								1 29	PHI
		TRP	1	N, P H I 1				26	
		APQ	1 3	N, P H I 3				30	
		TRP	2	S, P H I 1				3	
		APQ	2 3	S, P H I 3				2 2	PHI
		TRP	0	V 2				2 1	V
		APQ	0 3	V 4				0 0	U
		SPRA	A	1					
		AP	3	K = - 1					
		MSQ							
		TPM	0 1 2	T O					
		TRP	1	N, T O					
		TRP	2	S, T O					
		TRP	0	U 3					
		APQ	0 3	U 5					
		SPRA	0 3	1					
		AP	3	K = - 1					
		TPM	A	T O					

16-22
15090A

Computer and Data Systems 

		 CODING FORM																															16-22										
PROGRAM: AVERAGE VALUES		DATE:			PROGRAMMER: M. L. GRAHAM																							PAGE 2 OF 3															
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS																															SEQUENCE								
1	2	9	10	15	16	21	22																																72	73	80		
		SQC		A				C	O	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31			
		MSQ		A				M	O	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3			
								A		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X			
								B		X				X			X			X			X			X			X			X			X			X					
		AP0		130		W,	TO	C			X	X			X	X						X	X					X	X						X	X							
		SPRA		013		1		D			X					X									X										X								
A		AP		0		K = -1																																					
		TPM		013		TO																																					
		TRP		2		W,	TO																																				
		AP0		23		E,	TO																																				
		SPRA		23		1																																					
		AP		3		K = -1																																					
B		TPM		23		TO																																					
		MSQ		A																																							
		TRP		3		W,	TO																																				
		TPM		3		TO																																					
		TRP		0		W,	TO																																				
C		TPM		0		TO																																					
		TRP		1		W,	TO																																				
		AP0		10		E,	TO																																				
		SPRA		01		1																																					
		AP		0		K = -1																																					
D		TPM		01		TO																																					

20004-1



CODING FORM

PROGRAM: AVERAGE VALUES

DATE: M. L. GRAHAM

PROGRAMMER:

PAGE 3 OF 3

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22		72 73 80
			T M C G		G+13	ACTIVATE COLUMN 13
			T M M Ø		G C	SET GEOMETRIC CONTROL FOR COLUMNS
			T N L	Ø 1	T Ø	
			T M M Ø		Z E R Ø	INHIBIT GEOMETRIC CONTROL
			S Ø C	A		SET MODES OF COLUMNS
			M S Ø	A		5 TO 1 21 TO 2 OTHERS Ø
			A P Ø	1 Ø	C T Ø	
			T M C G		G+29	
			T M M Ø		G C	
			T N L		T Ø	
			T M M Ø		G C	
			A P Ø	2 Ø	C T Ø	
			S P R A	Ø 1 2	1	
			A P	Ø	K = - 1	
			T P M	A	T Ø	
			M S Ø			
			T M C G		G+21	
			T M M Ø		G C	
			T N L		T Ø	
			T M M Ø		Z E R Ø	
			A P Ø	1 Ø	C T Ø	
			S P R A	Ø 1	1	
			A P	Ø	K = - 1	
			T M C G		G+5	
			T M M Ø		G C	
			T N L		T Ø	

20004-1

16.6.2 Method of Polar Data Transfer

The method providing access to polar data becomes an important consideration when actual programming is attempted. Due to the nature of the SOLOMON II System, there are two options which can be made available using the Broadcast Register. These are: to provide access to all PE's when a constant is required by the Network, and to provide a variable to the north row of PE's (row 0) when a north routing is required while all other rows obtain their variable from the row of PE's above; e.g., row 0 would receive its value from the B Register but rows 1, 2 ... 31 would receive their data from rows 0, 1, ... 30. A similar situation exists for a south routing. To obtain this special feature, the programmer must give up the horizontal cylindrical connection. This last option will allow instructions requiring north or south routings to be executed for all PE's including those nearest the poles without a transfer of polar data from the Broadcast Register to a temporary location in those PE's concerned. This last feature will simplify the programming considerably for Network problems in general.

In regard to the transfer of PE data to the Network Control Unit, the L-Buffer will be employed. The required averaged data from the PE's will, upon instruction, transfer to an address in the L-Buffer Memory. The Network Control Unit will then, upon instruction, retrieve these data from the L-Buffer Memory and store the averaged values in the NCU Memory so they can be loaded into the B register when required. The hardware used in transferring plus the PE Network itself is shown in figure 16-4.

In general, global circulation models require cyclic continuity in the constant latitude direction. This implies that the values at the right and left hand (east and west) boundaries must be equivalent at all points on the boundary for all times. This condition is well simulated on the SOLOMON II System by simply connecting the PE Network in the form of a cylinder, with the top and bottom set as the north and south poles, respectively. This

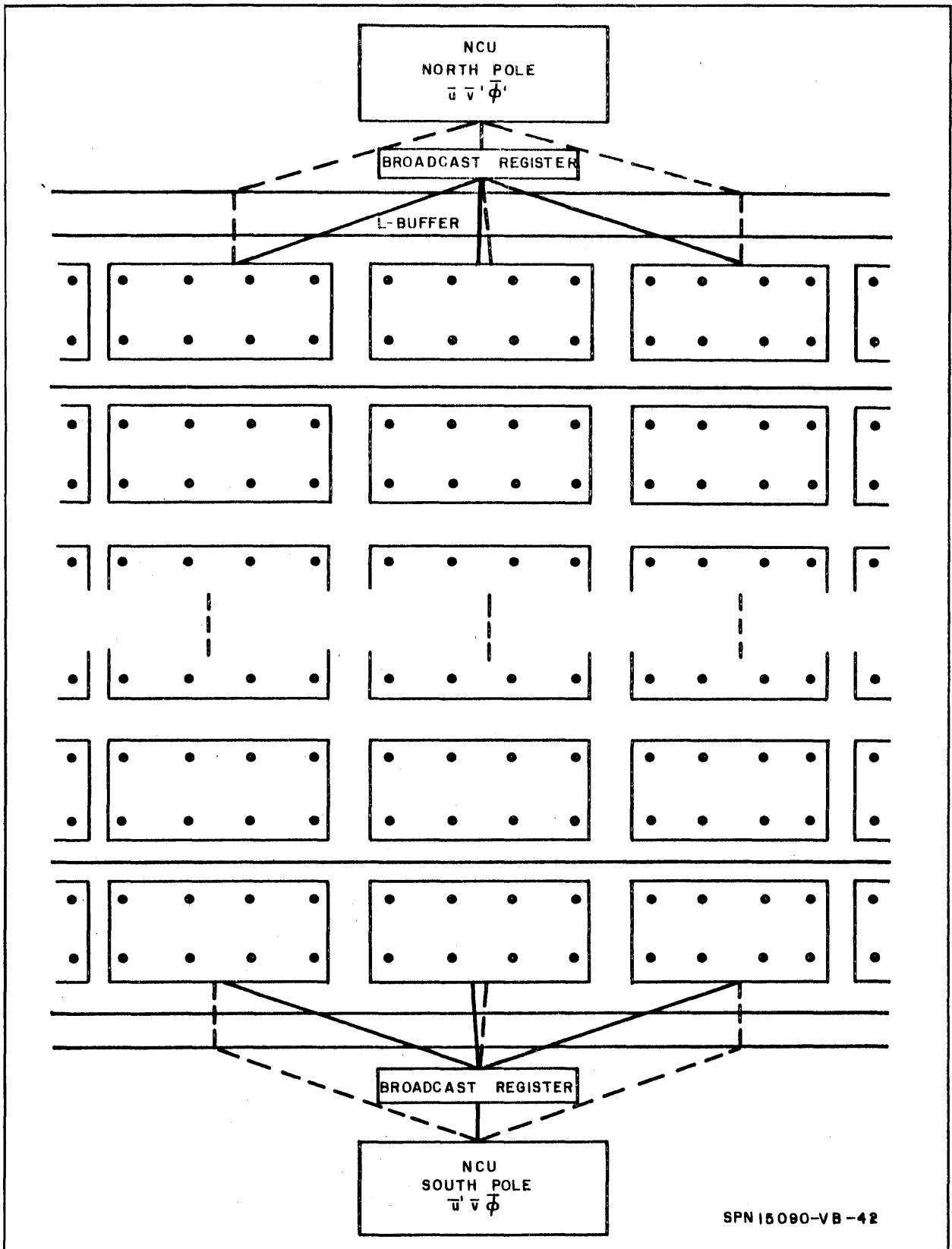


Figure 16-4. Configuration of Network



configuration ensures the cyclic continuity required by global or hemispherical forecasts on the western and eastern boundaries. As has been stated previously, the Network Control Unit (and the Broadcast Register) will serve as the top and bottom of the cylinder.

16.6.3 Usage of the L-Buffer in Both Polar and Latitude Data Transfer

In certain portions of the following SOLOMON II Program, advantage was taken of certain parameters which were a function of latitude only (i. e., constant for a given latitude line). In these instances, the L-Buffer can be employed to provide each latitude line (row of PE's) with such factors as the coriolis parameter and map scale factor when they are required by a given PE instruction. Upon instruction, these parameters can be transferred via the L-Buffer into the PE Network, thereby saving storage within the PE Network itself since for a given row of PE's (at each position taken one at a time) these quantities are constant.

It has already been demonstrated how the L-Buffer can be used to transfer polar data from the Network into the NCU Memory for the averaging process. Since this is along columns of PE's rather than rows, this implies that this facility can be used in either direction simply by supplying the necessary connections from the L-Buffer to both rows and columns of Processing Elements. This circuitry is shown in figure 16-5. It can be seen that this facility can be an extremely useful tool in models of this sort where the grid coordinates lie along lines of constant latitude and longitude.

When one set of connections (i. e., to PE columns) is being utilized, the other set (i. e., to PE rows) is disconnected (or conversely) by control.

16.6.4 Computational Procedure

- a. Read in initial geopotential height field, constants, and latitude data
- b. Solve by relaxation the equation

$$\frac{\partial f}{\partial y} \frac{\partial \psi}{\partial y} + f \left(\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \right) = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2}$$

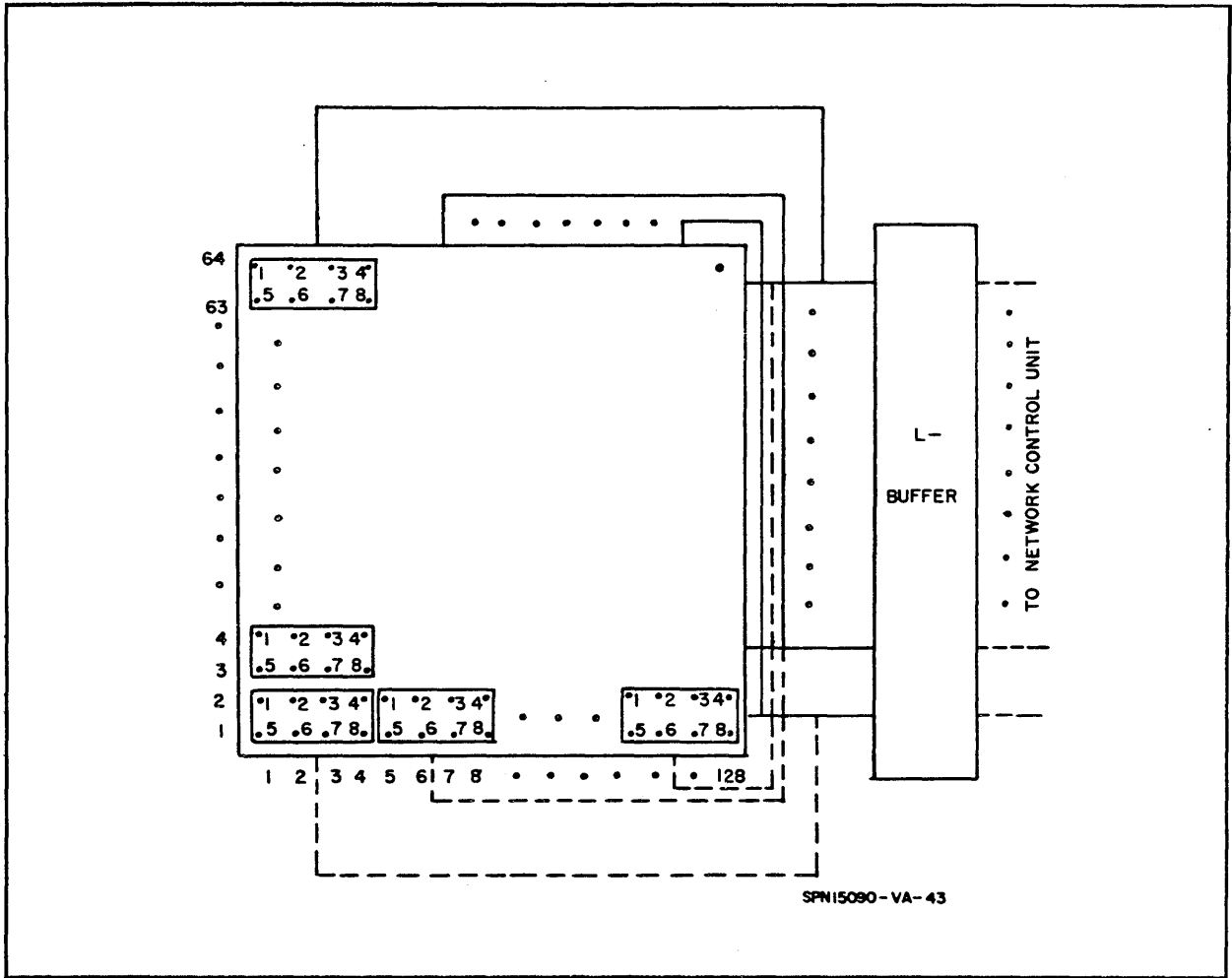


Figure 16-5. Diagram of L-Buffer Connections

for the initial stream function using as an initial guess

$$\psi = \frac{1}{f} \phi$$

As boundary conditions set

$$\psi_{Np} = \frac{1}{2\Omega} \phi_{Np} \quad Np = \text{north pole}$$

$$\psi_{Sp} = -\frac{1}{2\Omega} \phi_{Sp} \quad sp = \text{south pole}$$

c. Solve for the wind field by

$$u = -\frac{1}{\cos \theta} \frac{\partial \psi}{\partial y} \quad v = \frac{1}{\cos \theta} \frac{\partial \psi}{\partial x}$$



Take $v = u = 0$ at poles

d. Solve the forecast equations

$$\frac{\partial u}{\partial t} = -M \left(u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + \frac{\partial \phi}{\partial \lambda} \right) + f v \left(1 + \frac{M}{2\Omega a} u \right)$$

$$\frac{\partial v}{\partial t} = -M \left(u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + \frac{\partial \phi}{\partial y} \right) - f u \left(1 + \frac{M}{2\Omega a} u \right)$$

$$\frac{\partial \phi}{\partial t} = -M \left[\frac{\partial}{\partial x} (u \phi) + \frac{\partial}{\partial y} (v \phi) - \frac{f}{\partial \Omega} v \phi \right]$$

for the desired number of time steps. See figure 16-6.

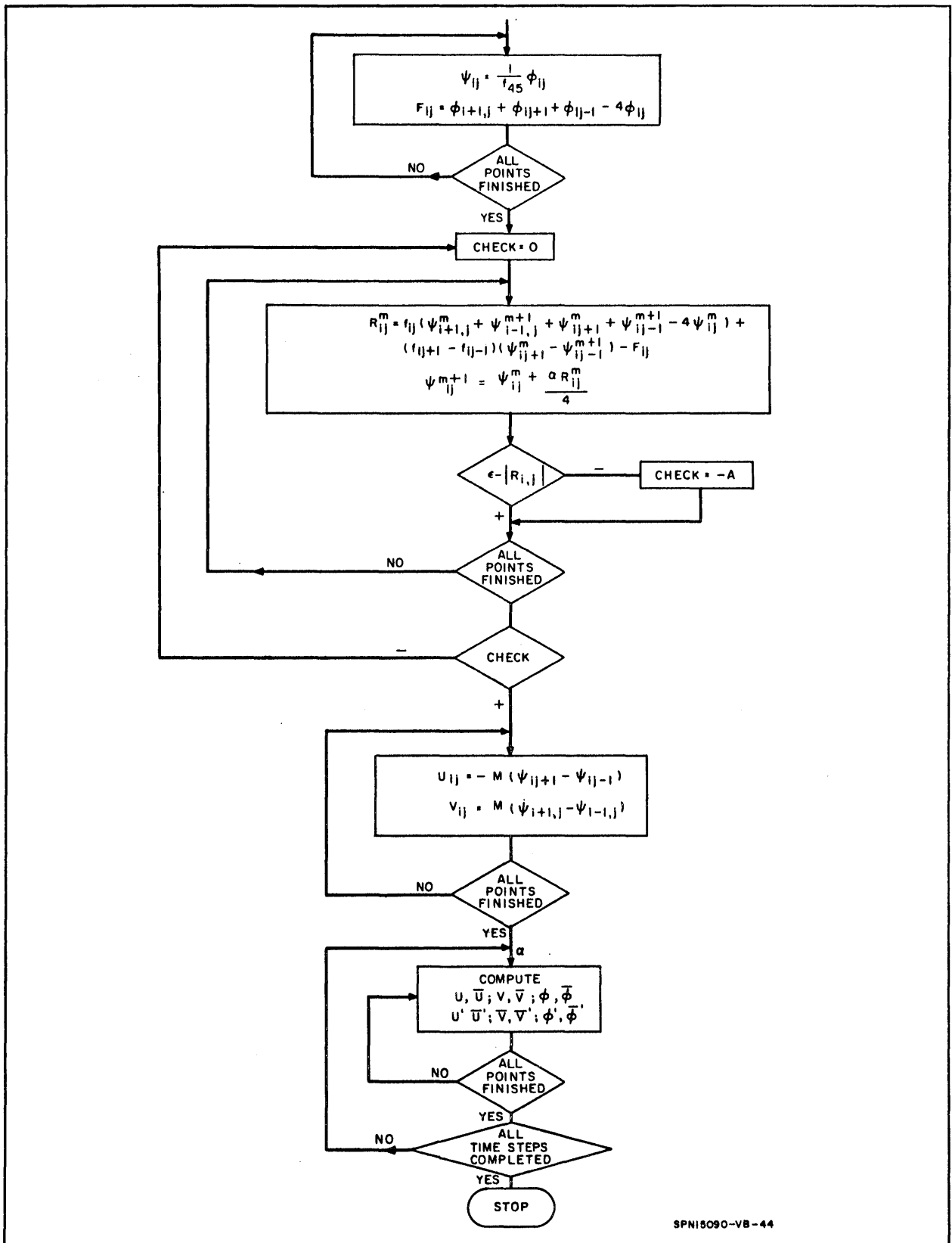


Figure 16-6. Flow Chart for Primitive Equation Model

This page intentionally left blank

16.7 ASSUMPTIONS IN CODING

The computational procedure for this forecasting model is given in the preceding pages and shown in figure 16-7. It should be stated that this sample model does not represent an operational forecasting problem in that it has been simplified in the following manner:

a. It is assumed that the initial geopotential height field has been read in and stored in the PE Network. Also constants and parameters, depending upon latitude, are stored in Central Control Memory.

b. This problem has been written in a fixed point mode, but some of the detailed scaling operations have been omitted (i. e. , the shift instructions necessary to line up the binary points of the two operands, prior to addition or subtraction have been left out; also, the necessary shifts to perform the required multiplications have been included but the magnitude of the shifts have been omitted).

c. It is assumed that the number of iterations needed for convergence of the relaxation procedure do not differ significantly for the SOLOMON II System and a sequential type of computer.*

d. No checking or correcting of overflow conditions has been programmed.

e. The given program represents the main computational parts, namely:

- (1) Initialization for stream function (one relaxation)
- (2) Computation of initial wind field from stream function
- (3) A typical time integration of the six finite difference forecast equations.

And does not include input/output of data, initial, final, or intermediate.

f. The running time estimates are based on the following three sections of the program. These are:

* A numerical investigation was conducted on this assumption, and tests were made on varying grid sizes for a particular equation. For the results and discussion of this experiment see paragraph 17.2 of this report.



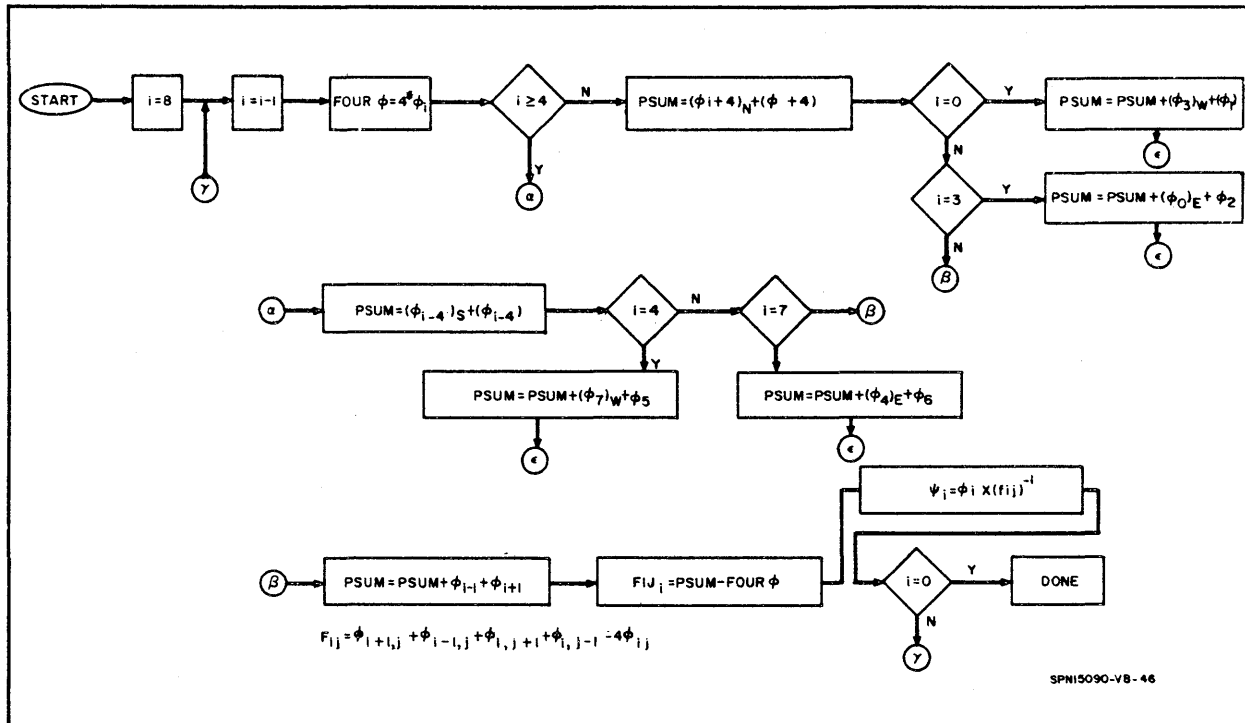


Figure 16-7. Flow Chart for Computing FIJ

- (1) Running time for one iteration of the relaxation procedure
- (2) Time for computing initial guess for relaxation procedure plus computing wind field from stream function
- (3) Computation time for a typical time integration of the six forecast equations (not the first time step).

16.8 SOLOMON II CODING OF PRIMITIVE EQUATION MODEL

The coding for this model is divided up into the following sections:

- a. Computation of initial guess function from given geopotential height field
- b. Coding of the relaxation procedure required to find the initial stream function
- c. Computation of initial wind field from stream function
- d. Computation of forecast variables and averaging procedure at poles.

To obtain the initial stream function, computations were performed at all eight points in the Processing Element (all points in the Network). From this stream function, the wind components U and V were then computed for those points required by the staggered grid system (in coding, positions No. 2, 4, 5, 7 in PE's). Also, as a matter of convenience in indexing, the eight points per Processing Element were numbered 0 through 7 for the initialization section, then renumbered 1 through 8 for the forecasting portion. Finally, the quantities $f\Delta t$, $-1/2 N \frac{\Delta t}{\Delta}$, $-M \frac{\Delta t}{\Delta}$, $-f\Delta t \frac{M}{8 \Omega a}$, $-\frac{f}{16 \Omega}$, $-f\Delta t \frac{M}{2 \Omega a}$ are assumed to be stored in Central Control Memory. Constants for the system are a and ϵ which are broadcast to the PE Network via the Broadcast Register. Quantities such as $1/2$, $1/4$, etc are handled simply by shifting the operand the required number of places (left shift-binary system). The same applies to factors like 2, 4, etc (right shift-binary system).

16.8.1 Variable Storage

a. Initialization

PHI, 1	ϕ , modified by Index Register No. 1 (positions 0 through 7).
PSI, 1	ψ , modified by Index Register No. 1 (position 0 through 7).
FIJ, 1	Storage for F_{ij} in finite difference balance equation, modified by Index Register No. 1 (positions 0 through 7).
RIJ, 1	Storage for residual R_{ij}^m computed during relaxation procedure modified by Index Register No. 1 (positions 0 through 7).
f, 1	Coriolis parameter f_{ij} , modified by Index Register No. 1 (positions 0 through 7).
FOUR ϕ , OP1,	Temporary storage locations
FOUR , OP2,	
4ALPHA	Relaxation coefficient (constant for all positions)
EPS	Convergence criterion (constant for all positions)
CHECK	Storage location reserved for checking of convergence criterion, in PE's

b. Computation of Wind Field.

PSI 0-7	ψ at positions 0 through 7
UPI	U at position No. 1



V1	V at position No. 1
UP3	U' at position No. 3
V3	V at position No. 3
U4	U at position No. 4
VP4	V' at position No. 4
U6	U at position No. 6
VP6	V at position No. 6

c. Forecast Section (positions above are now renumbered 1 through 8 for each Processing Element).

UP2	U' at position No. 2
UP4	U' at position No. 4
VP5	V' at position No. 5
VP7	V' at position No. 7
PHIP6	ϕ' at position No. 6
PHIP8	ϕ' at position No. 8
U5	U at position No. 5
U7	U at position No. 7
V2	V at position No. 2
V4	V at position No. 4
PHI1	ϕ at position No. 1
PHI3	ϕ at position No. 3
N	North routing
S	South routing
E	East routing
W	West routing
B	Broadcast Register
TEMP XY	Temporary storage at position X, number or letter Y
A, AB	Temporary storage, position No. 5
G, GH	Temporary storage, position No. 2
K, D	Temporary storage, position No. 1
C, CD	Temporary storage, position No. 2



E, EF	Temporary storage, position No. 5
L, F	Temporary storage, position No. 6
M, MN	Temporary storage, position No. 4
O, OP	Temporary storage, position No. 7
Q, P	Temporary storage, position No. 8
R, RS	Temporary storage, position No. 7
W, Y	Temporary storage, position No. 4
Z, F	Temporary storage, position No. 3

16.8.2 Loop for Computing F_{ij} and ψ_{ij}

Each PE contains 8 data points, i. e., 0, 1, ..., 7. It can be seen from figure 16-8 that only the corner points need special handling (i. e., only points 0, 3, 4, and 7 require special routing). The expression for F_{ij} is:

$$F_{ij} = \phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{ij}$$

Again it can be seen from figure 16-7 that if $i \geq 4$, a south routing is required for the evaluation and if $i \leq 3$ a north routing is required. (i = index for points 0 through 7.)

Essentially, the loop is performed under the control of one Index Register. The first important test is to determine whether $i > 3$, (i. e., determine whether a north or south routing is required). Suppose it is found that a north routing is required; immediately, it is known that two of the four neighbors can be summed, (i. e., that $\phi_{i+4,N}$) and ϕ_{i+4} can be added. Next, it must be determined whether i is either 0 or 3 since these 2 points require special routing. Three possibilities are present for this case: $i=0$, $0 < i < 3$, and $i=3$. The appropriate case can be determined by the use of index test instructions and index increments (see instruction sheets). Suppose it is determined that $i=0$. Then a west routing is required. If $i=3$, an east routing

NOTE: In actual practice, many of these storage locations would be consolidated, thereby reducing considerably the required amount of working storage. For purposes of illustration, this was not done. Also the absence of a prefix implies internal routing. Finally, all above parameters are stored in the PE Network with the exception of 4ALPHA and EPS.



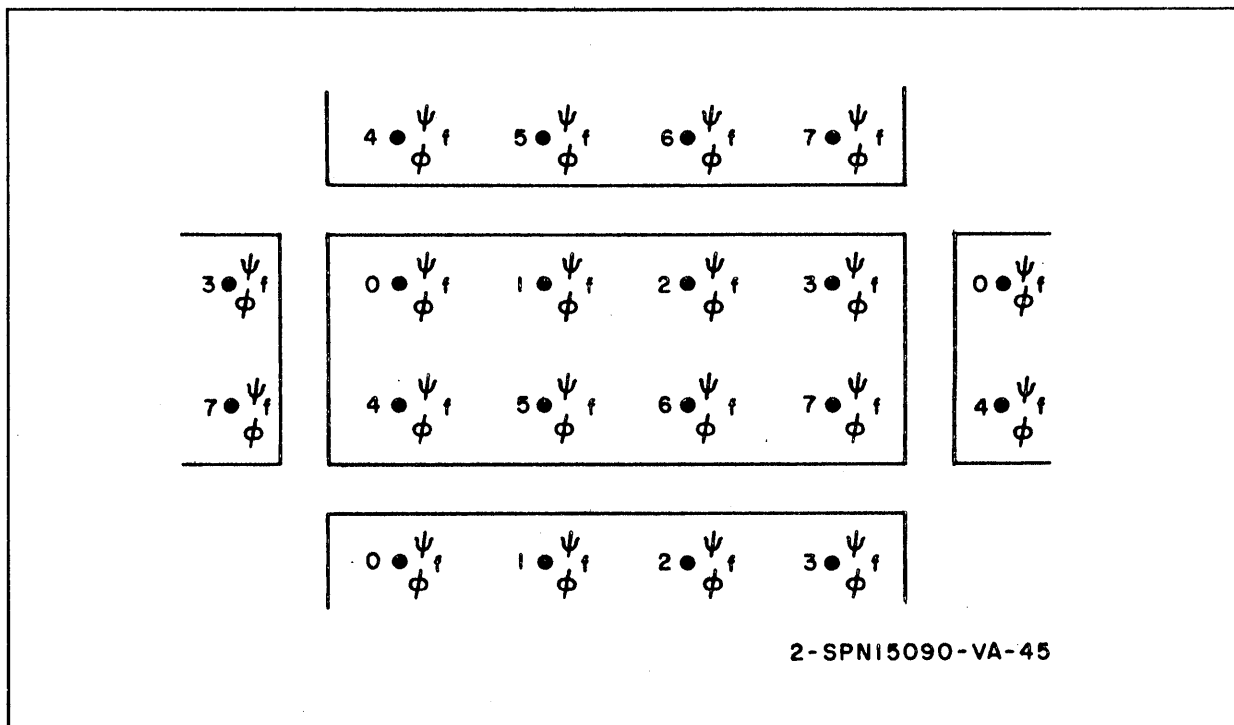


Figure 16-8. Grid Point - PE Allocation for Initialization

is required and if $i \neq 0$, 3 no special routing is necessary (in addition to the north routing already determined for the first test).

At this point it is only necessary that the remaining internal neighboring points be added into the previous sum and for the subtraction of $4\phi_i$ to be performed.

Suppose that $i \geq 4$. It must then be determined whether $i=4$, $4 < i < 7$, or $i=7$. From this point, the procedure is exactly the same as above.

After each F_{ij} is computed, the initial guess for ψ_i is computed. This consists only of a multiplication since the guess function for ψ_i is $\psi_i = \phi_i \times (f_{45})^{-1}$

Any questions that remain on this routine may be answered by examining the coding sheets provided.



CODING FORM

PROGRAM: *COMPUTATION OF F12* DATE:

PROGRAMMER: *M.D. GENTRY*

PAGE *1* OF *33*

T	LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS	SEQUENCE			
	1	2	9	10	15	16		21	22	72	73
							LOAD V WITH 4		1		STORAGE
							X1 → 8		1		ADDRESS
	OVER						X1 - 1 → X1		1		Φ ₀ X
							Φ ₀ → P		2	2	Φ ₀ X+1
							4Φ ₀ → P		2	2	...
							(P) → FOURPHI		2	2	Φ ₀ X+7
							TEST - IS I ≥ 4 JUMP IF T		2		
							LOAD BR REG WITH Φ ₀ NORTH POLE		2		ψ ₀ w
							X1 + 1 → X1		1		ψ ₀ w+1
							(Φ ₀ +1)NORTH → P		2	2	...
							(P)+(Φ ₀ +1) → P		2	2	ψ ₀ w+7
							X1 - 4 → X1		1		
							TEST - IS I = 0 JUMP IF I ≠ 0		2		
							X1 → 3		1		
							(P)+(Φ ₀ +3)WEST → P		2	2	
							X1 → 1		1		
							(P)+(Φ ₀ +1) → P		2	2	
							X1 → 0		1		
							JUMP TO COMP		2		
	CL						X1 - 3 → X1		1		
							TEST - IS I = 3 JUMP IF I ≠ 3		2		
							(P)+(Φ ₀ -3)EAST → P		2	2	
							X1 → 2		1		
							(P)+(Φ ₀ -1) → P		2	2	
							X1 → 3		1		
							JUMP TO COMP		2		
	GREATER						LOAD BR REG WITH Φ ₀ SOUTH POLE		2		
							X1 - 4 → X1		1		
							(Φ ₀ -4) → P		2	2	
							(P)+(Φ ₀ -4) → P		2	2	

20004-1



CODING FORM

PROGRAM:		DATE:		PROGRAMMER: M. D. GENTRY		PAGE 2 OF 33				
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
		JXWZ	1		C2	TEST - IS C=4	JUMP IF C≠4			2
		TIX	1		7	XI → 7				1
		AP			W, PHI, 1	(P)+(Q-3)W → P				2 2
		TIX	1		5	XI → 5				1
		AP			PHI, 1	(P)+(Q-1) → P				2 2
		TIX	1		4	XI → 4				1
		JP			COMP					2
C2		SIX	1		3	XI-3 → XI				1
		JXWZ			ADDZ	TEST - IS C=7	JUMP IF C≠7			2
		TIX	1		4	XI → 4				1
		AP			E, PHI, 1	(P)+(Q-3)E → P				2 2
		TIX	1		6	XI → 6				1
		AP			PHI, 1	(P)+(Q-1) → P				2 2
		TIX	1		7	XI → 7				1
		JP			COMP					2
ADDZ		AIX			1	XI+1 → XI				1
ADD3		AIX			2	XI+2 → XI				1
		AP			PHI, 1	(P)+(Q-1) → P				2 2
		AIX			2	XI+2 → XI				1
		AP			PHI, 1	(P)+(Q-1) → P				2 2
		SIX	1		1	XI-1 → XI				1
COMP		SR			FOURPHI	Σ NEIGHBORS - 4Q → P				2 2
		TPM			FIJ, 1	(P) → FIJ _i				2 2
		TRQ			PHI, 1	(P) → Q				2 2
		MQ			F ₁₂	(Q) · F ₁₂ → Q				1 3
		SPIQLA			N	FOR SCALING (PQ)				2 2
		TPM			PSI, 1	(P) → 1/2				2 2
		JXWZ			OVER					2

16.8.3 Relaxation Routine

The relaxation routine is very similiar to the routine for calculating F_{ij} . However the relaxation routine has a few features which the F_{ij} does not have and does not need.

First, Index Register No. 3 is used to count the number of iterations and is checked against a previously determined upper limit. If the number of iterations reaches this value before convergence is achieved, the program will be stopped. A stop such as this will indicate either faulty data or a poor choice of upper limit.

Also, after the residual is calculated for each point, the degree of convergence is checked. If convergence has not occurred, a mode control indicator is set. After the eighth residual has been computed and checked for convergence, a check is made to see if this indicator was set during any of the convergence comparisons. If the indicator was set, the program again picks up the first point and continues through the remaining seven points. This process is repeated until convergence has been reached for each of the eight points.





CODING FORM

PROGRAM: <i>RELAXATION</i>		DATE:		PROGRAMMER: <i>M. D. GENTRY</i>		PAGE <i>3 OF 33</i>		
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE
1 2	9 10	15 16	21 22					72 73 80
		TIX	1	8				1
		TRNM		COMCK			COMCK = STORAGE FOR CONSTANT 0	2
		TIX	3	1			IR3 COUNTS ITERATIONS	1
	BEG	TRNM		CHECK			CHECK = MODE STORAGE LOCATION	2 2
	TRY	MSI		0			SET ALL PEYS TO MODE 0	2 2
		SIX	1	1			X1 - 1 → X1	1
		TRP		PSI, 1			ψ → P	2 2
		SPLA		2			4ψ → P	2 2
		TPM		FOURPSI			(P) → FOURPSI	2 2
		JXGE	1	GTR			TEST - IS ≥ 4 JUMP IF T	2
		TMB		FNP			LOAD BR REG WITH FNORTHPOLE	2
		AIX	1	4			X1 + 1 → X1	1
		TRP		N, F, 1			(F) → P	2 2
		SR		F, 1			(P) - F _i → P	2 2
		TPM		OP1			(P) → OP1 OP1 = (F _{i+1}) _N - (F _i + 1)	2 2
		TMB		PSIMP			LOAD BR REG WITH ψ NORTH POLE	2
		TRP		N, PSI, 1			(ψ _{next}) _N → P	2 2
		SR		PSI, 1			(P) - ψ _{next} → P	2 2
		TPM		OP2			(P) → OP2 OP2 = (ψ _{next}) _N - ψ _{next}	2 2
		TRP		N, PSI, 1			(ψ _{next}) _N → P	2 2
		AP		PSI, 1			(P) + ψ _{next} → P	2 2
		SIX	1	4			X1 - 1 → X1	1
		JXNZ	1	C3			TEST - IS ≠ 0 JUMP ≠ 0	2
		TIX	1	3			X1 → 3	1
		AP		W, PSI, 1			(P) + (ψ _{next}) _W → P	2 2
		TIX	1	1			X1 → 1	1
		AP		PSI, 1			(P) + ψ _{next} → P	2 2
		TIX	1	0			X1 → 0	1
		JP		COMP 1				2
								1
	C3	SIX	1	3			X1 - 3 → X1	2





PROGRAM:		DATE:		PROGRAMMER: M. D. GENTRY		PAGE 4 OF 33		16-41		
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
		JXWZ	1			ADD 33	TEST - IS I = 3		JUMP I ≠ 3	2
		AP				E, PSI, 1	(P) + (Y _{L+3}) _E → P			2 2
		TIX	1			2	X1 → 2			1
		AP				PSI, 1	(P) + (Y _{L-1}) → P			2 2
		TIX	1			3	X1 → 3			1
		JP				COMPL				2
GTR		SIX	1			4	X1 - 4 → X1			1
		TMB				FSP	LOAD BR REG WITH f SOUTH POLE			2
		TRP				F, 1	f _i → P			2 2
		SR				S, F, 1	(P) - (f _{i-4}) _S → P			2 2
		TPM				OP1	(P) → OP1			2 2
		TMB				PSI, 1	LOAD BR REG WITH Y SOUTH POLE			2
		TRP				PSI, 1	Y _L → P			2 2
		SR				S, PSI, 1	(P) + (Y _{L-4}) _S → P			2 2
		TPM				OP2				2 2
		TRP				S, PSI, 1	(Y _{L-4}) _S → P			2 2
		AP				PSI, 1	(P) + (Y _{L-4}) → P			2 2
		JXWZ	1			C4	TEST - IS I = 4		JUMP I ≠ 4	2
		TIX	1			7	X1 → 7			1
		AP				W, PSI, 1	(P) + (Y _{L+3}) _W → P			2 2
		TIX	1			5	X1 → 5			1
		AP				PSI, 1	(P) + (Y _{L+4}) → P			2 2
		TIX	1			4	X1 → 4			1
		JP				COMPL	[X1 - 3 → X1]			2
C4		SIX				3				1
		JXWZ	1			ADD 77	TEST - IS I = 7		JUMP IF I ≠ 7	2
		TIX	1			4	X1 → 4			1
		AP				E, PSI, 1	(P) + (Y _{L+3}) _E → P			2 2
		TIX	1			6	X1 → 6			1
		AP				PSI, 1	(P) + (Y _{L-1}) → P			2 2

2000A-1



CODING FORM

PROGRAM:		DATE:		PROGRAMMER: M.D.GENTRY		PAGE 5 OF 33	
1	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS			SEQUENCE
1 2	9 10	15 16	21 22				72 73 80
		TLX	1	7	X1 → 7		1
		JP			COMPI		2
ADD 77	ALX	1		4	X1 + 4 → X1		1
ABD 33	ALX	1		2	X1 + 2 → X1		1
	AP				(P) + Y ₆₋₁ → P		2 2
	ALX	1		2	X1 + 2 → X1		1
	AP				(P) + Y ₆₋₁ → P		2 2
	SLX	1		1	X1 - 1 → X1		1
COMPI	SR				FOURPSI		2 2
	EPQ				(P) - 4 Y ₆ → P		2 2
	MQ				(P) → (Q)		2 2
	SPQLA				F, 1		1 3
	TRM				FOR SCALING		2 2
	SR				(P) → OP3	OP3 = f _i (Σ NEIGHBORS - 4 Y _i)	2 2
	TMQ				(P) → FIJ		
	MQ				OP1		2 2
	SPQLA				(OP1) → Q		2 2
	AP				OP2		1 3
	SR				FOR SCALING		2 2
	TRM				(P) + (OP3) → P		2 2
	TMQ				(P) - FIJ _i → P		2 2
	MQ				(P) → RIJ _i		2 2
	SPQLA				RIJ, 1		2 2
	AP				ALPHA = $\frac{\alpha}{\beta}$		2
	TRM				(RIJ _i) → Q		2 2
	MQ				B		1 3
	SPQLA				(Q)(B) → Q		2 2
	AP				FOR SCALING		2 2
	TRM				(P) + Y _i → P		2 2
	TMQ				(P) → Y _i		2 2
	TRM				EPS = -E		2
	TRM				(RIJ _i) → P		2 2
	APA				B		2 2
	MSDG				(P) - E → P		2 2
					ZCOWCK		2 2
					if P > 0. CHANGE PIE TO MODE 2		2 2





CODING FORM

PROGRAM:		DATE:		PROGRAMMER: M. D. GENTRY		PAGE 6 OF 33					
1	2	9	10	15	16	21	22	72	73	80	
LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS		SEQUENCE			
		QSM		2				STORE MODE FFS IN Q1A, 17		22	
		TQM		2		CHECK		(Q) → CHECK		22	
		TXN2				TRY		IF ALL 8 POINTS HAVEN'T BEEN DONE, DO NEXT POINT		22	
		TRQ				CHECK		(CHECK) → Q		22	
		MSG						LOAD MODE FROM Q1A, 19		22	
		AIX		3		J		X3+1 → X3		1	
		SIX		3		CHECKNO		CHECKNO = MAXIMUM NUMBER OF ITERATIONS		1	
		JXZ		3		ERROR		JUMP TO ERROR IF X3 = CHECKNO		2	
		AIX		3		CHECKNO				1	
		TX		1		B		X1 → B		1	
		JMD		2		BEG		NOT COMPLETE CONVERGENCE, DO ANOTHER ITERATION		3	

16.8.4 Forecast Section

The forecast coding is divided into 12 parts. These are separate codings to compute:

U	at position No. 5	(Refer to paragraph 16.6.1 for the detailed code used to obtain the averaged values for all variables)
V	at position No. 2	
ϕ	at position No. 1	
U'	at position No. 2	
V'	at position No. 5	
ϕ'	at position No. 6	
U	at position No. 7	
V	at position No. 4	
ϕ	at position No. 3	
U'	at position No. 4	
V'	at position No. 7	
ϕ'	at position No. 8	

Symbolic explanations for the coding instructions are given in the right hand columns and are mostly straightforward in nature. One situation should be described in detail however - the question of routings other than the nearest four neighbors. As has already been discussed, a given base PE is capable of communicating directly only with its nearest four neighbors, itself, or Central Control. Hence, when data is required from locations other than these regions of access, special provision must be made to transfer the required piece of data to an address where it can be fetched readily by the base PE. In this sample problem, situations arise where data is required from corner neighbors (i.e., nw, sw, se, or ne neighboring PE's) by the given finite difference scheme. Since this corner routing is not available, it was necessary to transfer the required data into an appropriate neighboring PE where it is directly accessible by the base PE. In general, this access to corner PE's merely requires two additional instructions. This type of technique was applied to various portions of this sample coding where the

forecast variables were being computed at corner positions of the PE's. Along this line, this same technique can be applied to any order finite differencing scheme if required values are stored in Processing Elements outside the nearest eight neighbors.



16-46
15090A

Computer and Data Systems



PROGRAM:		DATE:		PROGRAMMER: M.D. GENTRY		PAGE 7 OF 33				
LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS		SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
			TMB			PS1WP	LOAD BR REG WITH ψ NORTH POLE		2	
			IO			DELTAM	DELTAM $\frac{M}{\Delta}$ LOAD L-BUFFER		2	
			TRP			PS15	(PS15) \rightarrow P		2	2
			SR			M, PS15	(P) - ψ_{SN} \rightarrow P		2	2
			EPQ				(P) \leftrightarrow (Q)		2	2
			MQ			DELTAM			1	3
			SPQLA			M	FOR SCALING		2	2
			TPM			UP1			2	2
			TRP			PS12			2	2
			SR			PS10			2	2
			EPQ						2	2
			MQ			DELTAM			1	3
			SPQLA			M			2	2
			TPM			V1			2	2
			TRP			PS17			2	2
			SR			M, PS17			2	2
			EPQ						2	2
			MQ			DELTAM			1	3
			SPQLA			M			2	2
			TPM			UP3			2	2
			TRP			E, PS10			2	2
			SR			PS12			2	2
			EPQ						2	2
			MQ			DELTAM			1	3
			SPQLA			M			2	2
			TPM			V3			2	2
			TMB			PS1SP	PS1SP = ψ SOUTH POLE		2	
			TRP			PS10	PS10 SOUTH		2	2
			SR			PS10			2	2
			EPQ						2	2

20004-1



CODING FORM

16-46




CODING FORM

PROGRAM:		DATE:		PROGRAMMER: <i>M D GENTRY</i>		PAGE: <i>8 OF 33</i>				
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
		MQ				DELTA M			13	
		SPQLA				N			2.2	
		TPM				U4			2.2	
		TRP				PS15			2.2	
		SR				W, PS17			2.2	
		EPQ							2.2	
		MQ				DELTA M			13	
		SPQLA							2.2	
		TPM				VP4			2.2	
		TRP				S, PS12			2.2	
		SR				PS12			2.2	
		EPQ							2.2	
		MQ				DELTA M			13	
		SPQLA							2.2	
		TPM				U6			2.2	
		TRP				PS17			2.2	
		SR				PS15			2.2	
		EPQ							2.2	
		MQ				DELTA M			13	
		SPQLA							2.2	
		TPM				VP6			2.2	
									198.0	

16-48
15090A

Computer and Data Systems 

 CODING FORM												16-48			
PROGRAM: <i>Compute. Error Positron #3</i> DATE:										PROGRAMMER: <i>C. L. TENLEY</i>		PAGE <i>9</i> OF <i>33</i>			
T	LOCATION			INST.		OPTIONS		ADDRESS AND COMMENTS				SEQUENCE			
	1	2	9	10	15	16	21	22					72	73	80
				TMB					PHI AVERAGED						2
				TRP											2.2
				AP											2.2
				EPQ											2.2
				MQ				U' 2							1.3
				SPQ LA											2.2
				TPM				TEMP 3 I							2.2
				TRP				N, PHI ' 8							2.2
				AP				PHI ' 8							2.2
				EPQ											2.2
				MQ				U' 4							1.3
				SPQ LA											2.2
				SR				TEMP 3 I							2.2
				TPM				TEMP 3 I							2.2
				TRP				PHI ' 8							2.2
				AP				PHI ' 6							2.2
				TPM				F							2.2
				TRP				F							2.2
				EPQ				F							2.2
				MQ				V' 7							1.3
				SPQ LA											2.2
				TPM				Z							2.2
				TMB				Z							2.
				TRP				N, Z							2.2
				SR				Z							2.2
				AP				TEMP 3 I							2.2
				TPM				TEMP 3 I							2.2
				IO				TEMP 3 A							2
				IO				TEMP 3 B							2
				TMB				V' AVERAGED							2.

20004-1



CODING FORM

PROGRAM: *Compute Ocean Position #3* DATE:

PROGRAMMER: C. L. TENLEY

PAGE 10 OF 33

T 1 2	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE	
					72 73	80
	TRP			N, V' 7		2. 2
	AP			V' 7		2. 2
	TPM			Z		2. 2
	TMB			Φ' AVERAGED		2. 2
	TRP			N, F		2. 2
	AP			F		2. 2
	EPQ					2. 2
	MQ			Z		1 3
	SPQRA			N PLACES		2. 2
	MQ			TEMP 3A		1 3
	SPQLA			N PLACES		2. 2
	AP			TEMP 3I		2. 2
	EPQ					2. 2
	MQ			TEMP 3B		1 3
	SPQLA			N PLACES		2. 2
	AP			PHI 3		2. 2
	TPM			PHI 3		2. 2
						1 6 7. 0

2000A-1



CODING FORM

PROGRAM: *U₀₀ n factor #2* DATE:

PROGRAMMER: C. L. TENLEY

PAGE 11 OF 33

T 1 2	LOCATION 9 10	INST. 13 16	OPTIONS 21 22	ADDRESS AND COMMENTS	SEQUENCE	
					72 73	80
		TRP	A	U_{i-1}	U_{ii}	2.2
		AP		$(U_{i-1} + U_{i-1})$	$(U_{ii} + U_{-ii})$	2.2
		TPM		C		2.2
		TRP		U_{i-1}	U_{ii}	2.2
		SR		$(U_{i-1} - U_{i-1})$	$(U_{ii} - U_{-ii})$	2.2
		EPQ		P → Q		2.2
		MQ		C		2.2
		SPQLA		$(U_{i-1} + (U_{i-1})(U_{i-1} - U_{i-1}))$	$(U_{ii} + U_{-ii})(U_{ii} - U_{-ii})$	13
		TPM		SCALING		2.2
		TMB		CD		2.2
		AP		ZERO		2.
		SPRA		N, CD	$[U_{i-1} + U_{i-1}](U_{i-1} - U_{i-1}) + (U_{ii} + U_{-ii})(U_{ii} - U_{-ii})$	2.2
		TPM		2	$\frac{1}{2}[\quad]$	2.2
		TMB		TEMP2I	$\frac{1}{2}[\quad]$	2.2
		TRP		TIPSUM		2.
		SR		N, C	$U_{ii} + U_{-i}$	2.2
		EPQ		C	$(U_{ii} + U_{-ii}) - (U_{i-1} + U_{i-1})$	2.2
		MQ		P → Q		2.2
		SPQLA		V2	$V_{00}[(U_{ii} + U_{-ii}) - (U_{i-1} + U_{i-1})]$	13
		AP		DIVIDE AND SCALE	$\frac{1}{2}V_{00}[\quad]$	2.2
		TPM		TEMP2I	$\frac{1}{2}[\quad] + \frac{1}{2}V_{00}[\quad]$	2.2
		IO		TEMP2A - $M_{00} \frac{\Delta F}{2}$		2.2
		TRP		CCM → LBUF	Φ_{10}	2
						2.2

200041






CODING FORM

PROGRAM: V_{co} n fraction 2				DATE:				PROGRAMMER: C. L. TENLEY				PAGE 12 OF 33				
LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS						SEQUENCE				
1	2	9	10	15	16	21	22							72	73	80
			SR				PHI1	$\phi_{10} - \phi_{-10}$								2.2
			AP				TEMP2I	$\{ \phi_{10} - \phi_{-10} + \frac{1}{4} [] + \frac{1}{2} V_{co} [] \}$								2.2
			EPQ				P → Q									2.2
			MQ				TEMP2A	$-M_{oo} \frac{\Delta T}{A} \{ \}$								13
			SPQLA				SCALE									2.2
			TPM				TEMP2I	$-M_{oo} \frac{\Delta T}{A} \{ \}$								2.2
			IO				TEMP2A	$f_{AT} \frac{M}{A} []$								2
			IO				TEMP2B	f_{AT}								2
			TMB				C									2.
			TRP				N.C	$V_{11} + V_{-11}$								2.2
			AP				C	$[(V_{11} + V_{-11}) + (V_{-11} + V_{-1-1})]$								2.2
			EPQ				P → Q									2.2
			MQ				TEMP2A	$f_{AT} \frac{M}{A} []$								13
			SPQLA				SCALE									2.2
			AP				TEMP2B	$f_{AT} + f_{AT} \frac{M}{A} []$								2.2
			EPQ				P → Q									2.2
			MQ				V2	$V_{co} \{ f_{AT} + f_{AT} \frac{M}{A} [] \}$								13
			SPQLA				SCALE									2.2
			AP				TEMP2I	$V_{co} \{ f_{AT} + f_{AT} \frac{M}{A} [] \} - M_{oo} \frac{\Delta T}{A} \{ \}$								2.2
			AP				UP2	$V_{co} n + 1$								2.2
			TPM				UP2	$V_{co} [1] - M_{oo} \frac{\Delta T}{A} \{ \phi_{10} - \phi_{-10} + \frac{1}{4} [(V_{11} + V_{-11}) (V_{1-1} - V_{-1-1}) + (V_{11} + V_{-11}) (V_{11} - V_{-11})] + \frac{1}{2} V_{co} [(V_{11} + V_{-11}) - (V_{-11} + V_{-1-1})] \}$								2.2
								$+ V_{co} \{ f_{AT} + f_{AT} \frac{M}{A} [(V_{11} + V_{-11}) + (V_{11} + V_{-1-1})] \}$								14.9.6

2000A-1

 CODING FORM 16-52														
PROGRAM: <i>COMPUTATION OF V_{avg}</i>				DATE: <i>POSITION #5</i>				PROGRAMMER: <i>M.D. GENTRY</i>				PAGE <i>13</i> OF <i>33</i>		
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS							SEQUENCE			
1	2	9	10	15	16	21	22					72	73	80
	2	TMB				P SUM								
	2.2	M S I	A			0		SET MODES TO 0						
	2.2	T R P				V ₂		$(V_{i-1}) \rightarrow P$						
	2.2	A P				S, V ₂		$(V_{i-1}) + (P) \rightarrow P$						
	2.2	T P M				E'		$(P) \rightarrow E'$ $E' = V_{i-1} + V_{i-1}$						
	2.2	T R P				V ₂		$V_i \rightarrow P$						
	2.2	S R				S, V ₂		$(P) - V_{i-1} \rightarrow P$						
	2.2	E P G						$(P) \leftrightarrow (Q)$						
1	3	M Q				E'								
	2.2	S P Q L A				N								
	2.2	T P M				E F		$E F = (V_i + V_{i-1})(V_{i-1} - V_{i-1})$						
	2.2	T R P				S, V ₄		$(V_4)_s \rightarrow P$						
	2.2	T P M				TEMP 5 1		$(P) \rightarrow TEMP 5 1$						
	2.2	T R P				W, V ₄		$V_{-i} \rightarrow P$						
	2.2	A P				W, TEMP 5 2								
	2.2	T P M				TEMP 5 2		$TEMP 5 2 = V_{-i} + V_{-i-1}$						
	2.2	T R P				W, V ₄		$V_{-i} \rightarrow P$						
	2.2	S R				W, TEMP 5 1		$(P) - V_{-i-1} \rightarrow P$						
	2.2	E P G						$(P) \leftrightarrow (Q)$						
1	3	M Q				TEMP 5 2								
	2.2	S P Q L A				E F		$(P) = (V_{-i} + V_{-i-1})(V_{-i} - V_{-i-1})$						
	2.2	A P						$(P) = (V_{-i} + V_{-i-1})(V_{-i} - V_{-i-1}) + (V_{-i} + V_{-i-1})(V_{-i} - V_{-i-1})$						



CODING FORM

PROGRAM:		DATE:		PROGRAMMER: M. D. GENTRY		PAGE 14 OF 33	
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS		SEQUENCE	
1 2	9 10	15 16	21 22			72 73	80
	2. 2	SPRA		2	$\frac{1}{2}(P) \rightarrow P$		
	2. 2	TPM		TEMP51	$(P) \rightarrow TEMP51$		
	2. 2	TRP		E	$(V_n + V_{n+1}) \rightarrow P$		
	2. 2	SR		TEMP52	$(P) = (V_n + V_{n+1}) - (V_{n-1} + V_{n-2})$		
	2. 2	EPQ			$(P) \leftarrow (Q)$		
1 3		MQ		US	$US = U_{oc}$		
	2. 2	SPQLA		N-1	$(P) = \frac{1}{2} U_{oc} (V_n + V_{n+1} + V_{n-1} + V_{n-2})$		
	2. 2	AP		TEMP51	$(P) + \frac{1}{2} [\quad] \rightarrow P$		
	2. 2	TPM		TEMP52			
	2	IO		TEMP5A	$TEMP5A \leftarrow -M \frac{\Delta t}{2}$ LOAD LBUFFER		
	2. 2	TRP		PH11	$\phi_1 \rightarrow P$		
	2. 2	SR		S, PH11	$(P) - \phi_{oc-1} \rightarrow P$		
	2. 2	AP		TEMP51	$(P) + (TEMP5A) \rightarrow P$		
	2. 2	EPQ			$(P) \leftarrow (Q)$		
1 3		MQ		TEMP5A			
	2. 2	SPQLA		N			
	2. 2	TPM		TEMP51			
	2	IO		TEMP5A	$TEMP5A \leftarrow -f \Delta t \frac{M}{2U_{oc}}$		
	2	IO		TEMP5B	$TEMP5B \leftarrow -f \Delta t$		
	2. 2	TRQ		US	$U_{oc} \rightarrow Q$		
1 3		MQ		TEMP5A			
	2. 2	SPQLA		N	$(P) = -U_{oc} (f \Delta t \frac{M}{2U_{oc}})$		
	2. 2	AP		TEMP5B	$(P) - f \Delta t \rightarrow P$		
	2. 2	EPQ			$(P) \leftarrow (Q)$		
1 3		MQ		US			
	2. 2	SPQLA		N	$(P) = U_{oc} [U_{oc} (f \Delta t \frac{M}{2U_{oc}}) + f \Delta t]$		
	2. 2	AP		TEMP51			
	2. 2	AP		VP5			
	2. 2	TPM		TEMP51	$(TEMP51) = V_{ocm-1} - U_{oc} [U_{oc} (f \Delta t \frac{M}{2U_{oc}}) + f \Delta t] - M \frac{\Delta t}{2} [\phi_{oc-1} - \phi_{oc} + \frac{1}{2} f] + \frac{1}{2} U_{oc} (\quad)$		
1 7 6	2. 2						

2000A-1

		CODING FORM										16-54						
PROGRAM:		ϕ' on position #6										DATE:		PROGRAMMER:		PAGE 15 OF 33		
T	LOCATION	INST.		OPTIONS		ADDRESS AND COMMENTS										SEQUENCE		
		9	10	15	16											21	22	72
	2.		TMB															
	2.2		TRP															
	2.2		AP															
	2.2		EPQ															
	13		MQ															
	2.2		SPQLA															
	2.2		TPM															
	2.2		TRP															
	2.2		AP															
	2.2		EPQ															
	13		MQ															
	2.2		SPQLA															
	2.2		SR															
	2.2		TPM															
	2.		TMB															
	2.2		TRP															
	2.2		AP															
	2.2		TPM															
	2.2		TRP															
	2.2		EPQ															
	13		MQ															
	2.2		SPQLA															
	2.2		TPM															

2000A-1



CODING FORM

PROGRAM: ϕ_{00n} position #6

DATE:

PROGRAMMER:

PAGE 16 OF 33

1	LOCATION		INST.	OPTIONS		ADDRESS AND COMMENTS	SEQUENCE	
	2	9 10		15 16	21 22		72 73	80
	2	2	SR			S, L		
	2	2	AP			TEMP 6 I		
	2	2	TPM			TEMP 6 A		
	2		IO			TEMP 6 B		
	2		IO			TEMP 6 B		
	2		TMB			VAVE		
	2	2	TRP			V2		
	2	2	AP			S, V2		
	2	2	TPM			L		
	2		TRB			B		
	2	2	TRP			B		
	2	2	AP			S, B		
	2	2	EPQ			P → Q		
1	3		MQ			L		
	2	2	SIPQLA			SCALE		
1	3		MQ			TEMP 6 A		
	2	2	SIPQLA			SCALE		
	2	2	AP			TEMP 6 I		
	2	2	EPQ			P → Q		
1	3		MQ			TEMP 6 B		
	2	2	SIPQLA			SCALE		
	2	2	AP			PHIG		
	2	2	TPM			PHIG		
1	6	4	. 8					

20004-1

16-56
15090A



CODING FORM

16-56

PROGRAM: *Compute U^{ooc} Position #4* DATE:

PROGRAMMER: C L TENLEY

PAGE 17 OF 33

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22	72 73	80
		2. 2	TRP		U_{t-1}	
		2. 2	AP		$U_{t-1} + U_{-1-1}$	
		2. 2	T PM		$U_{t-1} + U_{-1-1}$	
		2. 2	TRP		U_{t-1}	
		2. 2	SR		$U_{t-1} - U_{-1-1}$	
		2. 2	E P Q			
13		M Q			$(U_{t-1} - U_{-1-1})(U_{t-1} + U_{-1-1})$	
		2. 2	SP Q LA		N PLACES	
		2. 2	T PM		$(U_{t-1} - U_{-1-1})(U_{t-1} + U_{-1-1})$	
		2. 2	T MB		MN	
		2. 2	AP		$\frac{1}{4}[U_{t-1} - U_{-1-1})(U_{t-1} + U_{-1-1}) + (U_{t-1} + U_{-1-1})(U_{t-1} - U_{-1-1})]$	
		2. 2	SP Q RA		2	
		2. 2	T PM		TEMP 4 I	
		2. 2	T MB		M	
		2. 2	TRP		$U_{t-1} + U_{-1-1}$	
		2. 2	SR		$(U_{t-1} + U_{-1-1}) - (U_{t-1} + U_{-1-1})$	
		2. 2	E P Q			
13		M Q			$V_{00}[(U_{t-1} + U_{-1-1}) - (U_{t-1} + U_{-1-1})]$	
		2. 2	SP Q LA		N - 1 PLACES	
		2. 2	AP		SCALING $\frac{1}{2}V_{00}[]$	
		2. 2	T PM		$\frac{1}{2}V_{00}[] + \frac{1}{4}[]$	
		2. 2	T PM		TEMP 4 I	
		2. 2	IO		TEMP 4 A	
		2. 2	TRP		$-M_{00} \Delta \frac{1}{4} \rightarrow L$ BUFFER	
		2. 2	SR		ϕ_{10}	
		2. 2	AP		PHI 3	
		2. 2	TRP		$\phi_{10} - \phi_{-10}$	
		2. 2	AP		$\phi_{10} - \phi_{-10} + \frac{1}{4}[] + \frac{1}{2}V_{00}[]$	
		2. 2	E P Q			
13		M Q			$-M_{00} \Delta \frac{1}{4} \{ \phi_{10} - \phi_{-10} + \frac{1}{4}[] + \frac{1}{2}V_{00}[] \}$	
		2. 2	SP Q LA		N PLACES	
		2. 2	T PM		FOR SCALING	
		2. 2	IO		TEMP 2 A	
		2. 2	TRP		$-M_{00} \Delta \frac{1}{4} \{ \phi_{10} - \phi_{-10} + \frac{1}{4}[] + \frac{1}{2}V_{00}[] \}$	
		2. 2	IO		TEMP 2 A	
		2. 2	TRP		$\Delta \frac{1}{4} \{ \phi_{10} - \phi_{-10} + \frac{1}{4}[] + \frac{1}{2}V_{00}[] \}$	
		2. 2	IO		TEMP 2 A	

20004-1

Computer and Data Systems





CODING FORM

PROGRAM: Compute $U_{oo,n}$ Position #4 DATE:

PROGRAMMER: C.L. TENLEY

PAGE 18 OF 33

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE				
1	2	9	10	15	16	21	22	72	73	80
		2	IO			TEMP 2 B				
		2.2	TRP							
		2.2	AP							
		2.2	EPQ							
		13	MQ			TEMP 2 A				
		2.2	SPQLA							
		2.2	AP							
		2.2	EPQ							
		13	MQ			V4				
		2.2	SPQLA							
		2.2	AP							
		2.2	AP							
		2.2	TPM							
		147.6								

2000A-1



CODING FORM

PROGRAM: *V¹⁰⁰ Position #1*

DATE:

PROGRAMMER: C L TENLEY

PAGE 19 OF 33

T	LOCATION		INST.	OPTIONS		ADDRESS AND COMMENTS	SEQUENCE					
	1	2		9	10		15	16	21	22	72	73
		2.		TMB			V P AVERAGED					
		2.	2	TRP			V 4			V 11		
		2.	2	AP			S, V 4			V 1-1		
		2.	2	TPM			⊖			V 11 + V 1-1		⊖ - ALPHA 0
		2.	2	TRP			V 4			V 11		
		2.	2	SR			S, V 4			V 11 - V 1-1		
		2.	2	EPQ								
	1	3		MC			⊖			(V 11 - V 1-1)(V 11 + V 1-1)		
		2.	2	SPOLA			N PLACES			SCALING		
		2.	2	TPM			⊖ P					
		2.	2	TRP			V 2			V -11		
		2.	2	AP			S, V 2			V -11 + V 1-1-1		
		2.	2	TPM			TEMP 72			V -11 + V 1-1-1		
		2.	2	TRP			V 2			V -11		
		2.	2	SR			S, V 2			V -11 - V 1-1-1		
		2.	2	EPQ								
	1	3		MC			TEMP 72			(V -11 + V 1-1-1)(V -11 - V 1-1-1)		
		2.	2	SPOLA			N PLACES			SCALING		
		2.	2	AP			⊖ P			(V -11 + V 1-1-1)(V -11 - V 1-1-1) + (V 11 - V 1-1)(V 11 + V 1-1)		
		2.	2	SPRA			2			$\frac{1}{2} [\quad]$		
		2.	2	TPM			TEMP 71					
		2.	2	TRP			⊖			V 11 + V 1-1		
		2.	2	SR			TEMP 72			(V 11 + V 1-1) - (V -11 + V 1-1)(V -11 - V 1-1)		
		2.	2	EPQ								
	1	3		MC			U 7					
		2.	2	SPOLA			N-1 PLACES			SCALING $\frac{1}{2} U 7 [\quad]$		
		2.	2	AP			TEMP 71					
		2.	2	TPM			TEMP 71			$\frac{1}{2} [\quad] + \frac{1}{2} U 7 [\quad]$		
		2		IC			TEMP 7A			-M $\frac{1}{2} [\quad]$ - L BUFFER		
		2.		TMB			⊖					

20004-1



CODING FORM

PROGRAM: <i>Voon Position #1</i>										DATE:										PROGRAMMER: C. L. TENLEY										PAGE 20 OF 33									
LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS		SEQUENCE																															
1	2	9	10	15	16	21	22	72	73	80																													
	2.2		TRP				PHI3																																
	2.2		SR				S, PHI3																																
	2.2		AP				TEMP 71																																
	2.2		EPQ																																				
13			MQ				TEMP 7A																																
	2.2		SPOLA				N PLACES																																
	2.2		TPM				TEMP 71																																
	2		IO				TEMP 7A																																
	2		IO				TEMP 7B																																
	2.2		TRP				U7																																
	2.2		EPQ																																				
13			MQ				TEMP 7A																																
	2.2		SPOLA				N PLACES																																
	2.2		AP				TEMP 7B																																
	2.2		EPQ																																				
13			MQ				U7																																
	2.2		SPOLA				N PLACES																																
	2.2		AP				TEMP 71																																
	2.2		AP				VPT																																
17	16																																						

20004-1

16-60
15090A

PROGRAM: <i>Con Position # 8</i>										DATE:										PROGRAMMER: C. L. TENLEY										PAGE 21 OF 33										16-60									
LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS		SEQUENCE																																									
1	2	9	10	15	16	21	22	72	73	80																																							
	2.		TMB				PHI AVERAGED																																										
	2.	2	TRP				PHI 3			Φ_{11}																																							
	2.	2	AP				S, PHI 3			$\Phi_{11} + \Phi_{11}$																																							
	2.	2	EPQ																																														
	1	3	MQ				U7			$U_{10}(\Phi_{11} + \Phi_{11})$																																							
	2.	2	SPQLA				N PLACES			SCALING																																							
	2.	2	TPM				TEMP 81			$U_{10}(\Phi_{11} + \Phi_{11})$																																							
	2.	2	TRP				S, PHI 1			Φ_{11}																																							
	2.	2	TPM				TEMP 82			Φ_{11}																																							
	2.	2	TRP				E, PHI			Φ_{11}																																							
	2.	2	AP				E, TEMP 82			$\Phi_{11} + \Phi_{11}$																																							
	2.	2	EPQ																																														
	1	3	MQ				E, U5			$U_{10}(\Phi_{11} + \Phi_{11})$																																							
	2.	2	SPQLA				N PLACES			SCALING																																							
	2.	2	SR				TEMP 81			$U_{10}(\Phi_{11} + \Phi_{11}) - U_{10}(\Phi_{11} + \Phi_{11})$																																							
	2.	2	TPM				TEMP 81																																										
	2.		TMB				Q																																										
	2.	2	TRP				E, PHI 1			Φ_{11}																																							
	2.	2	AP				PHI 3			$\Phi_{11} - \Phi_{11}$																																							
	2.	2	TPM				P			$\Phi_{11} - \Phi_{11}$																																							
	2.	2	TRP																																														
	2.	2	EPQ																																														
	1	3	MQ							$V_{01}(\Phi_{11} - \Phi_{11})$																																							
	2.	2	SPQLA				N PLACES			SCALING																																							
	2.	2	TPM				Q																																										
	2.	2	SR				S, Q			$V_{01}(\Phi_{11} - \Phi_{11}) - V_{10}(\Phi_{11} + \Phi_{11})$																																							
	2.	2	AP				TEMP 81			$V_{01}(\Phi_{11} - \Phi_{11}) - V_{10}(\Phi_{11} + \Phi_{11}) + V_{10}(\Phi_{11} + \Phi_{11}) - V_{10}(\Phi_{11} + \Phi_{11})$																																							
	2.	2	TPM				TEMP 81																																										
	2		IC				TEMP 8A			$-\frac{1}{2} M_{12} \rightarrow L$ BUFFER																																							
	2		IO				TEMP 8B			$-\frac{1}{2} M_{12} \rightarrow L$ BUFFER																																							

20004-1



CODING FORM


PROGRAM: ϕ_{002} Position #8

DATE:

PROGRAMMER: C. L. TENLEY

PAGE 22 OF 33

T	LOCATION		INST.	OPTIONS		ADDRESS AND COMMENTS	SEQUENCE	
	1 2	9 10		15 16	21 22		72 73	80
		2.	TMB			V AVERAGED		
		2-2	TRP			V4	V_{01}	
		2-2	AP			S, V4	$V_{01} + V_{0+1}$	
		2-2	TPM			Q	$V_{01} + V_{0-1}$	
		2.	TMB			P AVERAGED		
		2-2	TRP			P	$\phi_{11} + \phi_{-11}$	
		2-2	AP			S, P	$(\phi_{11} + \phi_{-11}) + (\phi_{+11} + \phi_{-11})$	
		2-2	EPQ					
	1 3		MQ			Q	$V_{01} + V_{0+1}[(\phi_{11} + \phi_{-11}) + (\phi_{-1-1} + \phi_{+1-1})]$	
	2-2		SPQRA			N PLACES	SCALING	
	1 3		MQ			TEMP 8A	$-\frac{f}{16} [V_{01} + V_{0+1}(\phi_{11} + \phi_{-11} + \phi_{-1-1} + \phi_{+1-1})]$	
	2-2		SPOLA			N PLACES	SCALING	
	2-2		AP			TEMP 81	$\{ -\frac{f}{16} [\quad] + V_{01}(\quad) - V_{-10}(\quad) + V_{10}(\quad) - V_{-10}(\quad) \}$	
	2-2		EPQ					
	1 3		MQ			TEMP 8B	$-\frac{1}{2} M \Delta \{ \quad \}$	
	2-2		SPOLA			N PLACES	SCALING	
	2-2		AP			PHIP 8	$\phi_{002-1} - \frac{1}{2} M \frac{\Delta t}{\Delta} \{ \quad \}$	
	2-2		TPM			PHIP 8	$\phi_{002-1} - \frac{1}{2} M \frac{\Delta t}{\Delta} \{ -\frac{f}{16} [V_{01} + V_{0+1}(\phi_{11} + \phi_{-11} + \phi_{+1-1} + \phi_{-1-1})] + V_{01}(\phi_{11} - \phi_{-11}) - V_{-10}(\phi_{+11} + \phi_{-1-1}) + V_{10}(\phi_{11} + \phi_{-1-1}) - V_{-10}(\phi_{-11} + \phi_{+1-1}) \}$	
	1 6 9	2						

 CODING FORM 16-62																										
PROGRAM: $\phi_{00} n+1$ position #1																		DATE:			PROGRAMMER:			PAGE 23 OF 33		
T	LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS															SEQUENCE				
1	2	9	10	15	16	21	22																72	73	80	
		2		TMB				ϕ AVE																		
		2.2		TRP				N, PHIP 8																		
		2.2		TPM				TEMP I 1																		
		2.2		TRP				W, PHIP 8																		
		2.2		AP				W, TEMP I 1																		
		2.2		EPQ				P \rightarrow Q																		
	1	3		MQ				W, UP 4																		
		2.2		SPQLA				SCALE																		
		2.2		TPM				TEMP I 1																		
		2.2		TRP				N, PHIP 6																		
		2.2		AP				PHIP 6																		
		2.2		EPQ				P \rightarrow Q																		
	1	3		MQ				UP 2																		
		2.2		SPQLA				SCALE																		
		2.2		SR				TEMP I 1																		
		2.2		TPM				TEMP I 1																		
		2.2		TRP				PHIP 6																		
		2.2		AP				W, PHIP 8																		
		2.2		TPM				D																		
		2.2		TRP				D																		
		2.2		EPQ				P \rightarrow Q																		
	1	3		MQ				VP 5																		
		2.2		SPQLA				SCALE																		



CODING FORM


PROGRAM: $\phi_{00 n+1}$ position # 1 DATE:

PROGRAMMER:

PAGE 24 OF 33

T	LOCATION		INST.	OPTIONS		ADDRESS AND COMMENTS	SEQUENCE					
	1	2		9	10		15	16	21	22	72	73
		2.2	TPM									
		2	TMB									
		2.2	TRP									
		2.2	SR									
		2.2	AP									
		2.2	TPM									
		2	IO									
		2	IO									
		2	TMB									
		2.2	TRP									
		2.2	AP									
		2.2	TPM									
		2	TMB									
		2.2	TRP									
		2.2	AP									
		2.2	EPQ									
	13		MQ									
	2.2		SPQLA									
	13		MQ									
	2.2		SPQLA									
	2.2		AP									
	2.2		EPQ									
	13		MQ									
	2.2		SPQLA									
	2.2		AP									
	2.2		TPM									
	171	4										

20004-1

		 CODING FORM										16-64			
PROGRAM: <i>Vcont1 position # 2</i>		DATE:					PROGRAMMER: <i>C. L. TENLEY</i>					PAGE <i>25</i> OF <i>33</i>			
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS							SEQUENCE				
1	2	9	10	15	16	21	22					72	73	80	
	2		TMB			V' AVE									
	2.2		TRP			N, VP7		V' _t							
	2.2		AP			VP7		V' _{t-1}							
	2.2		TPM			H		V' _t + V' _{t-1}							
	2.2		TRP			N, VP7		V' _t							
	2.2		SR			VP7		V' _t - V' _{t-1}							
	2.2		EPQ			P → Q									
	13		MIQ			H									
	2.2		SPQLA			SCALE									
	2.2		TPM			TEMP2I		(V' _t + V' _{t-1})(V' _t - V' _{t-1})							
	2.2		TRP			N, VP5		V' _{t-1}							
	2.2		AP			VP5		(V' _t + V' _{t-1})							
	2.2		TPM			G									
	2.2		TRP			N, VP5		V' _{t-1}							
	2.2		SR			VP5		(V' _{t-1} - V' _{t-2})							
	2.2		EPQ			P → Q									
	13		MIQ			G		(V' _{t-1} + V' _{t-2})(V' _{t-1} - V' _{t-2})							
	2.2		SPQLA			SCALE									
	2.2		AP			TEMP2I		(V' _t + V' _{t-1})(V' _t - V' _{t-1}) + (V' _{t-1} + V' _{t-2})(V' _{t-1} - V' _{t-2})							
	2.2		SPRA			2		$\frac{1}{4} [\quad]$							
	2.2		TPM			TEMP2I		$\frac{1}{4} [\quad]$							
	2.2		TRP			H		V' _t + V' _{t-1}							
	2.2		SR			G		(V' _t + V' _{t-1}) - (V' _{t-1} + V' _{t-2})							



CODING FORM

PROGRAM: V_{00} n+1 position #2 DATE:

PROGRAMMER: C. L. TENLEY

PAGE 26 OF 33

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE				
1	2	9	10	15	16	21	22	72	73	80
		2.2	EPQ				P → Q			
	13		MQ				U'2			
		2.2	SPQLA				DEVIDE & SCALE			
		2.2	AP				TEMP2I			
		2.2	TPM				TEMP2I			
		2	IO				TEMP2A			
		2	TMB				ϕ'			
		2.2	TRP				N, PH1P6			
		2.2	SR				PH1P6			
		2.2	AP				TEMP2I			
		2.2	EPQ				P → G			
	13		MQ				TEMP2A			
		2.2	SPQLA				SCALE			
		2.2	TPM				TEMP2I			
		2	IO				TEMP2A			
		2	IO				TEMP2B			
		2.2	TRQ				U'2			
	13		MQ				TEMP2A			
		2.2	SPQLA				SCALE			
		2.2	AP				TEMP2B			
		2.2	EPQ				P → G			
	13		MQ				UP2			
		2.2	SPQLA				SCALE			
		2.2	AP				TEMP2I			
		2.2	AP				V2			
		2.2	TPM				V2			
	17	1-6								

2000A-1



CODING FORM

PROGRAM: *Compute Voo 111 Position #4* DATE:

PROGRAMMER: C.L. TENLEY

PAGE 21 OF 33

T	LOCATION		INST.	OPTIONS		ADDRESS AND COMMENTS	SEQUENCE					
	1	2		9	10		15	16	21	22	72	73
		2. 0	TMB			V' AVERAGED						
		2. 2	TRP			N, VP 5	V' -31					
		2. 2	TPM			TEMP 4A						
		2. 2	TRP			E, TEMP 4A						
		2. 2	AP			E, VP 5						
		2. 2	TPM			W	$V'_{11} + V'_{1-1}$					
		2. 2	TRP			E, TEMP 4A	V' 11					
		2. 2	SR			E, VP 5	$V'_{11} - V'_{1-1}$					
		2. 2	EPQ			W	$(V'_{11} + V'_{1+1})(V'_{11} - V'_{1-1})$					
		2. 2	SPQLA			N PLACES	SCALING					
		2. 2	TPM			TEMP 41						
		2. 2	TRP			N, VP 7	V' -11					
		2. 2	AP			VP 7	V' -1-1					
		2. 2	TPM			Y	$V'_{-11} + V'_{-1-1}$					
		2. 2	TRP			N, VP 7	V' -11					
		2. 2	SR			VP 7	$V'_{-1-1} - V'_{-1-1}$					
		2. 2	EPQ			Y	$(V'_{-11} + V'_{-1+1})(V'_{-11} - V'_{-1-1})$					
		1 3. 0	MC			Y	$(V'_{-11} + V'_{-1+1})(V'_{-11} - V'_{-1-1})$					
		2. 2	SPQLA			N PLACES	SCALING					
		2. 2	AP			TEMP 41	$\frac{1}{4} [(V'_{11} + V'_{1+1})(V'_{11} - V'_{1-1}) + (V'_{-11} + V'_{-1+1})(V'_{-11} - V'_{-1-1})]$					
		2. 2	SPRIA			2						
		2. 2	TPM			TEMP 41						
		2. 2	TRP			W	$V'_{11} + V'_{1+1}$					
		2. 2	SR			Y	$(V'_{11} + V'_{1+1}) - (V'_{-11} + V'_{-1+1})$					
		2. 2	EPQ									
		1 3. 0	MC			UP 4	$U'_{00} [(V'_{11} + V'_{1+1}) - (V'_{-11} + V'_{-1+1})]$					
		2. 2	SPQLA			N-1 PLACES	SCALING $\frac{1}{2} U'_{00} [\quad]$					
		2. 2	AP			TEMP 41	$\frac{1}{2} U'_{00} [\quad] + \frac{1}{4} [\quad]$					
		2. 2	TPM			TEMP 41						

20004-1





CODING FORM

PROGRAM: *UCC-1 POSITION #5* DATE:

PROGRAMMER: *M. D. GENTRY*

PAGE 29 OF 33

T		LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1	2	9 10	15 16	21 22		72 73 80
		2. 2	TRP		UP2 U_{i-1} , U_{i-1}	
		2. 2	AP		W, UP4 U_{i-1} , U_{i-1}	
		2. 2	TPM		A $U_{i-1} + U_{i-1}$, $U_{i-1} + U_{i-1}$	
		2. 2	TRP		UP2 U_{i-1} , U_{i-1}	
		2. 2	SR		W, UP4 $U_{i-1} - U_{i-1}$, $U_{i-1} - U_{i-1}$	
		2. 2	EPO			
		13. 0	MQ		A $(U_{i-1} - U_{i-1})(U_{i-1} + U_{i-1})$, $(U_{i-1} - U_{i-1})(U_{i-1} + U_{i-1})$	
		2. 2	SPQLA		N	
		2. 2	TPM		AB	
		2. 0	TMB		ZERO	ZERO $\hat{=}$ LOCATION CONTAINING
		2. 2	AP		S, AB	$AB + S$, AB
		2. 2	SPRA		Z	$\frac{1}{2}[(AB) + (S, AB)]$
		2. 2	TPM		TEMP51	
		2. 2	TRP		A	$(U_{i-1} + U_{i-1})$
		2. 2	TMP		TOPSUM	$TOPSUM \hat{=} 2(CPSUM)$
		2. 2	SR		S, A	$(U_{i-1} + U_{i-1}) - (U_{i-1} + U_{i-1})$
		2. 2	EPO			
		13. 0	MQ		VPS	$V_{P0} [(A) - (S, A)]$
		2. 2	SPQLA		N-1	$\frac{1}{2} V_{P0} [(A) - (S, A)]$
		2. 2	AP		TEMP51	
		2. 2	TPM		TEMP51	$\frac{1}{2} [(AB) + (S, AB)] + \frac{1}{2} V_{P0} [(A) - (S, A)]$
		2. 0	IO		TEMP5A	$TEMP5A = \frac{1}{2} V_{P0} \frac{1}{2}$

20004-1





CODING FORM

PROGRAM:		DATE:		PROGRAMMER: M. D. GENTRY		PAGE 30 OF 33				
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
	2.2	TRP				PH1P6				
	2.2	SR				W, PH1PB				
	2.2	AP				TEMP51				
	2.2	EPQ					$\phi_{10} - \phi_{10} + \frac{1}{2} [] + \frac{1}{2} V_{b0} []$			
	13.0	MQ				TEMP5A				
	2.2	SPQLA				N				
	2.2	TPM				TEMP51				
	2.0	IC				TEMP5A				
	2.0	IO				TEMP5B				
	2.2	TRP				A				
	2.2	AP				S, A				
	2.2	EPQ					$U_{11} + U_{11}$			
	13.0	MQ				TEMP5A				
	2.2	SPQLA				N				
	2.2	AP				TEMP5B				
	2.2	EPQ								
	13.0	MQ				VPS				
	2.2	SPQLA				N				
	2.2	AP				TEMP51				
	2.2	AP				US				
	2.2	TPM				US				
	147.8									

2000A-1



CODING FORM

PROGRAM: U₀₀ n₁₁ Position #1

DATE:

PROGRAMMER: C L TENLEY

PAGE 31 OF 33

T	I	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE	
						72	73
		2.2	TRP		U ⁱ 4		
		2.2	AP		U ⁱ -11		
		2.2	TPM		U ⁱ 11 + U ⁱ -11		
		2.2	TRP		U ⁱ 11		
		2.2	SR		U ⁱ 11 - U ⁱ -11		
		2.2	EPQ				
		13	MQ		(U ⁱ 11 + U ⁱ -11)(U ⁱ 11 - U ⁱ -11)		
		2.2	SPQLA		N PLACES		
		2.2	TPM		S CALING		
		2	TMB		RS		
		2.2	AP		S, RS		
		2.2	SPRA		2		
		2.2	TPM		TEMP 7I		
		2.2	TRP		R		
		2	TMB		R		
		2.2	SR		S, R		
		2.2	EPQ				
		13	MQ		V _{op} 7		
		2.2	SPQLA		N-1		
		2.2	AP		TEMP 7I		
		2.2	TPM		TEMP 7I		
		2	IO		TEMP 7A		
		2.2	TRP		PHI P 8		
		2.2	SR		PHI P 6		
		2.2	AP		TEMP 7I		
		2.2	EPQ				
		13	MQ		TEMP 7A		
		2.2	SPQLA		N PLACES		
		2.2	TPM		TEMP 7I		
		2	IO		TEMP 7A		

2000A-1





CODING FORM

16-71

PROGRAM: U₀₀n1 Position #1

DATE:

PROGRAMMER: C L TENNEY

PAGE 32

OF 33

T 1 2	LOCATION 9 10	INST. 15 16	OPTIONS 21 22	ADDRESS AND COMMENTS	SEQUENCE	
					72 73	80
	2	IC		TEMP 7B		
	2.2	TRP		R		
	2.2	AP		S, R		
	2.2	EPQ				
	13	MQ		TEMP 7A		
	2.2	SPOQA		N PLACES		
	2.2	AP		TEMP 7B		
	2.2	EPQ				
	13	MQ		VP 1		
	2.2	SPOQA		N PLACES		
	2.2	AP		TEMP 7I		
	2.2	AP		U 7		
	2.2	TPM		U 7		
	147.6					

2000A-1



CODING FORM

16-72

PROGRAM:

DATE:

PROGRAMMER:

PAGE 33 OF 33

T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS	SEQUENCE
1 2	9 10	15 16	21 22		72 73 80
				THIS PAGE INTENTIONALLY LEFT BLANK	

2000A-1

16-72
15090A

Computer and Data Systems



17. OTHER CONSIDERATIONS

17.1 HIGHER ORDER DIFFERENCING SCHEMES

Due to the design of the SOLOMON System, there are several modifications that can be made in numerical forecasting problems of this nature. It has been found that use of a spherical grid system mapped onto some type of conformal map projection can create computational difficulties due to either decreasing grid size as the polar regions are approached or lack of resolution in certain critical regions due to the mapping technique. To compensate for the latter difficulty, it is very feasible to introduce varying mesh lengths at certain latitudes (or at all latitudes) to give the desired resolution (care must be taken however not to create regions of discontinuities where the grid mesh is changed significantly). Using the PE Network, one can store the various values of the grid mesh size without additional effort, rather than taking the grid size as being constant. One can therefore compensate for any distortion produced by the mapping technique and obtain the desired resolution in both latitude and longitude units.

More sophisticated finite differencing schemes have either largely or totally eliminated the former problem. As was mentioned earlier, these schemes are based on energy conservation and involve grid points other than just the nearest four neighbors. If the four-grid-point-per-PE allocation is chosen, it is possible to utilize the parallel computation technique to the fullest extent. Suppose, for example, the following expression is to be evaluated.

$$Q = C_1 \left[(A_{11} + A_{1-1}) (a_{20} + a_{00}) - (A_{-11} + A_{-1-1}) (a_{00} + a_{-20}) \right] \\ + C_2 \left[(a_{21} + a_{01}) (\beta_{21} - \beta_{01}) - (a_{2-1} + a_{-01}) (\beta_{2-1} - \beta_{-01}) \right]$$

$$+C_3 \left[A'_{10} (a_{21} + a_{0-1}) - A'_{-10} (a_{01} + a_{-2-1}) + \beta'_{10} (\beta_{01} + \beta_{2-1}) \right. \\ \left. - \beta'_{-10} (\beta_{-21} + \beta_{0-1}) \right]$$

using the following variable distribution and PE allocation as shown in figure 7-1.

The code for the SOLOMON System is on the following coding forms.

The first set of instructions 1 through 13 computes the value of the quantities, $(A_{11} + A_{1-1}) (a_{20} + a_{00})$, $(A_{-11} + A_{-1-1}) (a_{00} + a_{-20})$. To do this it is necessary only to compute the value of $(A_{11} + A_{1-1}) (a_{20} + a_{00})$ at the base PE since the value of $(A_{-11} + A_{-1-1}) (a_{00} + a_{-20})$ will be computed automatically by the west PE for its base No. 3 point (i. e., $(A_{11} + A_{1-1}) (a_{20} + a_{00})$ for west neighboring PE). Once $(A_{11} + A_{1-1}) (a_{20} + a_{00})$ has been computed at all PE's for point No. 3 and stored in some temporary location, all that is necessary to form the required difference is to specify a west routing for that temporary location and subtract this from the quantity stored in the same temporary location (same address) of the base PE. An averaged A must be routed to the southern boundaries of PE's via the Broadcast Register prior to the operations' similar to various situations in the forecasting problem. This same general procedure can be applied to each of the terms in the given expression. For example, the next set of instructions 18 through 29 compute the quantities $(a_{21} + a_{01}) (\beta_{21} - \beta_{01})$ and $(a_{2-1} + a_{0-1}) (\beta_{2-1} - \beta_{0-1})$; $(a_{21} + a_{01}) (\beta_{21} - \beta_{01})$ for the base PE; and $(a_{2-1} + a_{0-1}) (\beta_{2-1} - \beta_{0-1})$ for the south neighbor. Again care must be taken to supply this quantity for the southern boundary rows of PE's (in this case zero since the difference of the same averaged value is zero) via a south routing from the Broadcast Register. The instruction sets 14 through 17 and 30 through 34 are used to multiply the given quantities by their respective constants and to add the resulting quantities together. The instruction set 35 through 54 computes, in the same manner, the quantities $A'_{10} (a_{21} + a_{0-1})$, $A'_{-10} (a_{01} + a_{-2-1})$, $\beta'_{10} (\beta_{01} + \beta_{2-1})$, and

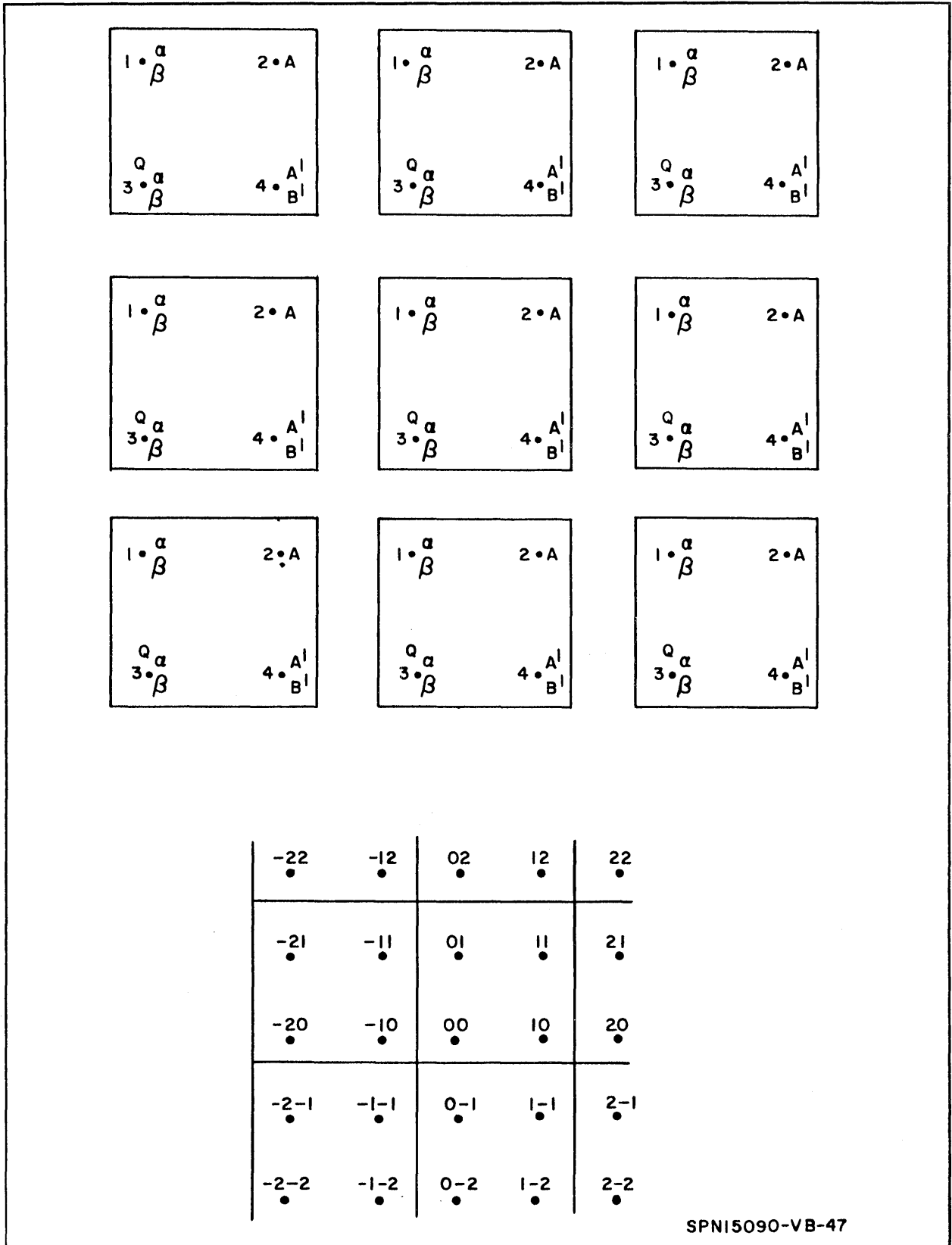


Figure 17-1. Grid Point - PE Allocation



$$Q = C_1 [(A_{11} + A_{1-1})(a_{20} + a_{00}) - (A_{-11} + A_{-1-1})(a_{00} + a_{20})] + C_2 [(\alpha_{21} + \alpha_{01})(\beta_{21} - \beta_{01}) - (\alpha_{2-1} + \alpha_{0-1})(\beta_{2-1} - \beta_{0-1})] + C_3 [A'_{10}(\alpha_{21} + \alpha_{0-1}) - A'_{10}(\alpha_{01} + \alpha_{2-1}) + B'_{10}(\beta_{01} + \beta_{2-1}) - B'_{10}(\beta_{21} + \beta_{0-1})]$$

Compute Expression "Q"				W	CODING FORM		17-4	
PROGRAM:		DATE:		PROGRAMMER:		PAGE 1	OF 3	
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE
1	2	9 10	15 16	21 22				72 73
1	2	9 10	15 16	21 22				72 73
		MSI		MODE O	SET ALL PE'S TO	MODE O		0
		TMB		A				1
		TRP		A2	A ₁₁	, A ₋₁₁		2
		AP		S, A2	A ₁₁ + A ₁₋₁	, A ₋₁₁ + A ₁₋₁		3
		TPM		CC	A ₁₁ + A ₁₋₁	, A ₋₁₁ + A ₁₋₁		4
		TMB		CI				5
		TRP		E, a ₃	a ₂₀	, a ₀₀		6
		AP		a ₃	a ₂₀ + a ₀₀	, a ₂₀ + a ₀₀		7
		EPQ						8
		MC		CC	(A ₁₁ + A ₁₋₁)(a ₂₀ + a ₀₀)	, (A ₋₁₁ + A ₁₋₁)(a ₀₀ + a ₂₀)		9
		SPQ LA		N PLACES	FOR SCALING			10
		TPM		CC				11
		TRP		CC				12
		SR		W, CC	[(A ₁₁ + A ₁₋₁)(a ₂₀ + a ₀₀) - (A ₋₁₁ + A ₁₋₁)(a ₀₀ + a ₂₀)]			13
		EPQ						14
		MC		CI	C ₁ []			15
		SPQ LA		N PLACES	FOR SCALING			16
		TPM		CC	C ₁ [(A ₁₁ + A ₁₋₁)(a ₂₀ + a ₀₀) - (A ₋₁₁ + A ₁₋₁)(a ₀₀ + a ₂₀)]			17
		TRP		E, ALPHA I	α ₂₁	, α ₂₋₁		18
		AP		ALPHA I	α ₂₁ + α ₀₁	, α ₂₋₁ + α ₀₋₁		19
		TPM		DD	α ₂₁ + α ₀₁	, α ₂₋₁ + α ₀₋₁		20
		TRP		E, BETA I	β ₂₁	, β ₂₋₁		21
		SR		BETA I	β ₂₁ - β ₀₁	, β ₂₋₁ - β ₀₋₁		22
		EPQ						23
		MC		DD	(α ₂₁ + α ₀₁)(β ₂₁ - β ₀₁)	, (α ₂₋₁ + α ₀₋₁)(β ₂₋₁ - β ₀₋₁)		24
		SPQ LA		N PLACES	FOR SCALING			25
		TPM		DD	(α ₂₁ + α ₀₁)(β ₂₁ - β ₀₁)	, (α ₂₋₁ + α ₀₋₁)(β ₂₋₁ - β ₀₋₁)		26
		TMB		DD	(α ₂₁ + α ₀₁)			27
		TRP		DD	(α ₂₁ + α ₀₁)(β ₂₁ - β ₀₁)			28
		SR		S, DD	(α ₂₁ + α ₀₁)(β ₂₁ - β ₀₁) - (α ₂₋₁ + α ₀₋₁)(β ₂₋₁ - β ₀₋₁)			29

2000A-1





PROGRAM: COMPUTE EXPRESSION		DATE:		PROGRAMMER:		PAGE 2 OF 3		17-5		
T	LOCATION	INST.	OPTIONS	ADDRESS AND COMMENTS				SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
		TMB				C ₂				30
		EPC								31
		MGC				C ₂				32
		SPCLA				N PLACES				33
		AP				C ₁				34
		TMB				ALPHA				35
		TRP				E, ALPHA I				36
		AP				S, ALPHA I				37
		EPC								38
		MGC				A ₄				39
		SPCLA				N PLACES				40
		TPM				DD				41
		TRP				DD				42
		SR				W, DD				43
		TPM				EE				44
		TMB				BETA				45
		TRP				W, BETA I				46
		AP				S, BETA I				47
		EPC								48
		MGC				W, B ₄				49
		SPCLA				N PLACES				50
		TPM				DD				51
		TRP				DD				52
		SR				DD				53
		AP				EE				54
		TMB				C ₃				55
		EPC								56
		MGC				C ₃				57
		SPCLA				N PLACES				58
		AP				C ₁				59

2000A-1



CODING FORM



PROGRAM:		DATE:		PROGRAMMER:		PAGE 3 OF 3				
LOCATION		INST.		OPTIONS		ADDRESS AND COMMENTS		SEQUENCE		
1	2	9	10	15	16	21	22	72	73	80
		T	P	M		C	3			60
$C_1 [(A_{11} + A_{11})(a_{20} + a_{20}) - (A_{11} + A_{11})(a_{20} + a_{20})]$ $+ C_2 [(a_{21} + a_{21})(b_{21} - b_{21}) - (a_{21} + a_{21})(b_{21} - b_{21})]$ $+ C_3 [A_{10}(a_{21} + a_{21}) - A_{10}(a_{21} + a_{21}) + B_{10}(b_{21} + b_{21}) - B_{10}(b_{21} + b_{21})]$										

2000A-1



CODING FORM

17-6

Expression "4"

$(\beta_{-10} (\beta_{-21} + \beta_{0-1}))$, and combines these values as required. The instruction set 55 through 60 performs the necessary coefficient multiplication plus the final addition and storage of the expression Q.

It can be seen that this parallel computation feature is at an optimum whenever the four-point PE allocation is used. In the forecast program, this feature was utilized only partially since it was desired to use the eight-point allocation. This is not a necessary feature but turns out to be a convenient one for finite difference schemes more involved than the conventional type.

17.2 RELAXATION TECHNIQUES ON THE SOLOMON SYSTEM

Strictly speaking, the assumption in paragraph 16.7 is not exactly true. As it turns out, due to the parallel processing of all Processing Elements (1024 points at a time), slightly fewer relaxations are required by the SOLOMON II System (4 points per PE) than by the sequential type of processing to obtain the same degree of convergence. This difference is very small (zero for the first case) but becomes more significant as the matrix size is increased. A numerical analysis was carried out to investigate this feature of convergence as accomplished by the two types of processing. This test consisted of solving Laplace's equation (finite difference analogue) for different grid sizes. The number of iterations (relaxations) required for convergence by both types of processing plus the absolute magnitude of the convergence (maximum difference), were then tabulated, and compared to determine which system provided the fastest rate of convergence. This test was conducted by a 7094 program and yielded the following results: (In all cases, a four-point-Processing-Element allocation was used. The initial field guess was zero and the boundary values were the same for all cases.).

<u>Grid Size</u>	<u>No. Iterations Sequential System</u>	<u>No. Iterations SOLOMON II System</u>
4 x 4	16	16
6 x 6	53	47
8 x 8	92	90
10 x 10	148	145
12 x 12	216	211



The computer program was instructed to stop if the number of iterations required to achieve convergence exceeded 250. Hence for the remaining cases, a relative difference in the magnitude of convergence by the two systems was all that was available. The remaining data are as follows, based on 250 iterations:

<u>Grid Size</u>	<u>Maximum Difference, Sequential System</u>	<u>Maximum Difference SOLOMON II System</u>
14 x 14	0.954×10^{-6}	0.477×10^{-6}
16 x 16	0.175×10^{-4}	0.846×10^{-5}
18 x 18	0.144×10^{-3}	0.669×10^{-4}
20 x 20	0.593×10^{-3}	0.264×10^{-3}

Note, for the sequential processing, a Liebmann relaxation technique was used.

It must be noted that the above statistics were obtained from a particular case with a particular grid point-PE allocation and do not, in general, represent exactly the rates of convergence for all types of problems requiring relaxation or iteration procedures for solution. It can be seen that for this case, however, the SOLOMON II System requires fewer iterations than the sequential method of processing to provide a convergent solution. In fact the sequential computer should be programmed to take advantage of this faster method of relaxation.

It is also appropriate to discuss the main types of relaxation procedures and their implementation by the SOLOMON II System. In general, relaxation techniques can be divided into two main schemes:

- a. A procedure by which no new data are used to generate the current values (Richardson's Method)
- b. A procedure by which two new data points are used to obtain the current values (Liebmann's Method).

These two methods are best illustrated by an example. Consider the Helmholtz equation:

$$\nabla^2 \phi - a\phi = F(x, y)$$

For the first case, the residual is given by:

$$r_{ij}^m = \phi_{i+1, j}^m + \phi_{i-1, j}^m + \phi_{i, j+1}^m + \phi_{i, j-1}^m - (4+a) \phi_{ij}^m - F_{ij}$$

m = the order of approximation.

Here, the residual (and hence the new value of ϕ_{ij} denoted by ϕ_{ij}^{m+1}) for each point is based entirely on old values computed by the previous relaxation.

For the second case (Liebmann relaxation) the residual is computed by:

$$r_{ij}^m = \phi_{i+1, j}^m + \phi_{i, j+1}^m + \phi_{i-1, j}^{m+1} + \phi_{i, j-1}^{m+1} - (4+a) \phi_{ij}^m - F_{ij}$$

where now r_{ij} is based on three old values of ϕ namely $\phi_{i+1, j}^m$, $\phi_{i, j+1}^m$, ϕ_{ij}^m and two new ones, $\phi_{i-1, j}^{m+1}$, $\phi_{i, j-1}^{m+1}$. In a sequential process (increasing i, j) these values are always available at each point and utilization of this new data has been shown to produce a faster rate of convergence. The new values of ϕ_{ij} are computed by a predictor-corrector formula given by:

$$\phi_{ij}^{m+1} = \phi_{ij}^m + \frac{a}{4+a} R_{ij}^m$$

a = a relaxation coefficient in the range

$$0.2 \leq a \leq 2.$$

At first glance, it would appear that the SOLOMON II System with its parallel processing feature would be forced to use the Richardson relaxation method. Upon closer investigation however, it is seen that for more than one point per PE allocation, a pseudo-Liebmann relaxation scheme can be generated. For example, take the four-point-per-PE case (see figure 17-2) computations at the first point (position No. 1) would have no new data available from which to work. Hence this first case is very similar to the Richardson method. For the second point, (position No. 2), one new datum is available, namely the value just computed at position No. 1, ($\phi_{i-1, j}^{m+1}$). For the third point (position No. 3) again the value previously computed at position No. 1 is available (now $\phi_{i, j+1}^{m+1}$). Finally, for the fourth point (position No. 4),



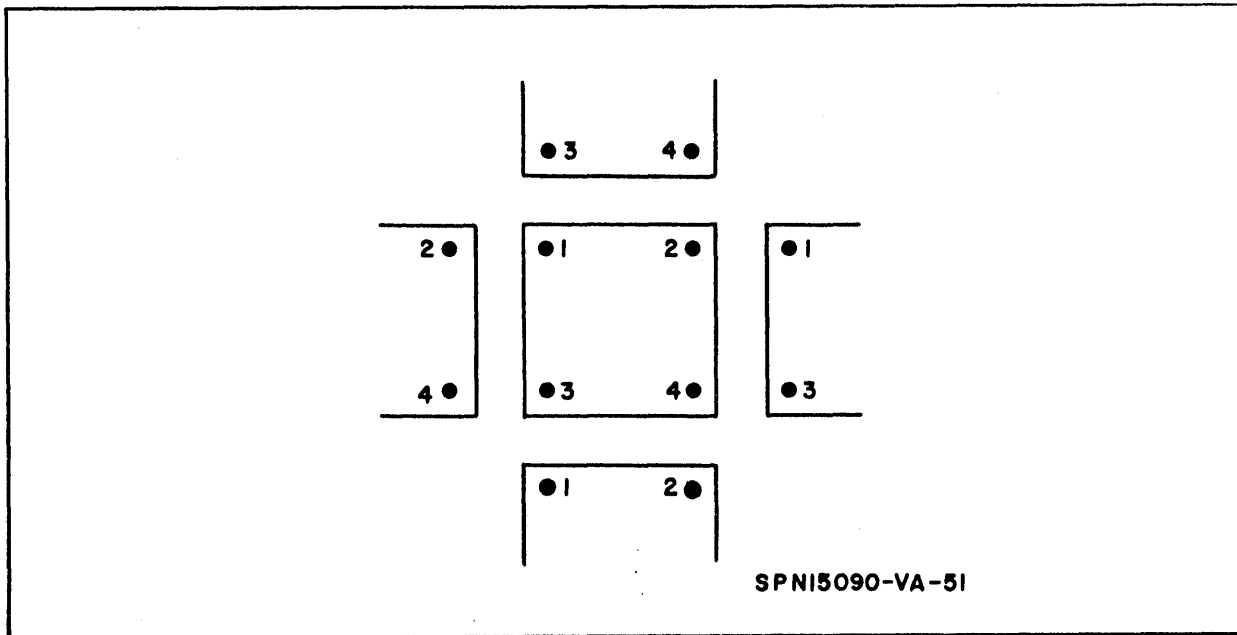


Figure 17-2. Four-Point-Per-PE Case

two new values are available, namely the new values at positions 2 and 3 ($\phi_{i, j+1}^{m+1}$, $\phi_{i-1, j}^{m+1}$, respectively). (In actuality, the values at positions No. 2 and No. 1 are also available for positions No. 3 and No. 4, respectively, but for this example are not required by the finite difference formula.)

It can be seen that generalization of the grid point - PE allocation will produce various combinations of available new and old data. It must be remembered that the above descriptive operations accounted only for Internal PE data and that corresponding new values (for a given position in the PE) are also generated concurrently in neighboring PE's. These new values may also be available, depending upon the requirements of the finite difference scheme being used. Hence it is conceivable that values for a given point can be computed from almost entirely new data. This is realized at position No. 4 for this example where new values at position 2 and 3 (and hence S, 3 and E, 3 are available; $\phi_{i, j+1}^{m+1}$, $\phi_{i-1, j}^{m+1}$, $\phi_{i, j-1}^{m+1}$, $\phi_{i+1, j}^{m+1}$, respectively). For the one-point-per-PE allocation, the above system obviously reduces to the Richardson method of relaxation.

18. SEQUENTIAL PROBLEMS

Suppose it is desirable to run a sequential problem or a sequential subproblem. If it is a sequential problem, there is only one logical choice. The GPU has the speed capability of approximately twice that of the IBM 7094. Thus any sequential problem could be run with greater speed in the GPU than in the PE Network.

However, suppose at some point in a program, it is desired to execute a sequential subproblem. For example, assume that it is necessary to find the square root of some number or numbers. Now a choice exists. Mainly, would the operation be performed faster in the GPU or in the PE Network. This decision is left entirely to the programmer.

If it is necessary to find the square root of one number, the GPU would be the logical choice since it would be the faster unit. However, as the number of values increase, the GPU loses its advantage.

Now assume that there are 1000 numbers which must have their square root calculated. In this case the PE Network would be the faster choice since the calculations are performed in parallel rather than sequentially.

To perform the desired operations in parallel, a scheme such as follows could be used:

- a. Store mode in memory location M
- b. Set all PE's to Mode X ($X > 0$)
- c. Set PE's applicable to the desired operation to Mode X-1
- d. Program the desired operation addressing all commands to Mode X-1
- e. Load original modes into the PE's from the memory location M
- f. Continue on original program



19. WAVE NUMBER SPACE COMPUTATION

Another possible method of numerical weather forecasting is the integration of the dynamic equations in wave number space. This method consists of transforming the prediction equations expressed in spatial coordinates into a wave number coordinate system leaving the time variable unchanged. This transformation results in a system of ordinary differential equations with time as the independent variable but whose spatial parts now contain interaction terms of various wave number components. The transformation is accomplished by the substitution: (m, n = wave numbers)

$$\phi(x, y, t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \phi_1(m, n, t) e^{i(k m x + l n y)}$$

for planar harmonics and:

$$\phi(\theta, \lambda, t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \phi_2(m, n, t) e^{i(k m \theta + l n \lambda)}$$

for spherical harmonics: where each term in the above series expansions must satisfy the given boundary conditions and ϕ is any dependent variable (single parameter model). For planar harmonics, substitution of the above expression into the dynamic equation of motion reduces the problem to the following general set of ordinary differential equations.

$$\ddot{\phi}_i = a_i + \sum_j b_{ij} \phi_j + \sum_{jk} c_{ijk} \phi_j \phi_k$$

where

ϕ_i = the i th dependent variable

a_i = a constant forcing function for each i

b_{ij} = the coefficients of the linear terms

c_{ijk} = the coefficients of the nonlinear (advection) terms.



The above mentioned coefficients are, generally, functions of the spectrum of wave numbers only and are of the form:

$$\begin{aligned}b_{ij} &= B(J) \\c_{ijk} &= C(J, K) \\ \phi_j &= \phi(J, N) \\ \phi_k &= \phi(M - J, N - K)\end{aligned}$$

where M, N equal wave numbers; J, K are indices of wave number spectrum under consideration. The series expansion is necessarily truncated to include the desired wave components, and still retain computational feasibility. To initiate the forecast procedure, the initial values of ϕ in wave number coordinate must be specified.

At first glance, this type of solution appears to lend itself solely to a sequential type of processing. This is due to the fact that a given wave component is capable of interacting with any other wave component within the limits of the summation, rather than merely the nearest four or eight neighbors as is the case with the conventional finite differencing process. This is equivalent to saying that derivatives at a given point (wave number) depend upon given information in the entire domain rather than just neighboring information.

However, recalling that the PE Network can be connected into a two dimensional cylinder (i. e., connections to ends of rows and columns), a possible method for utilizing the network for this type of problem can be outlined. For this shape, the same set of communications apply as before, except for this case, just the internal arithmetic operations will be considered (i. e., operations within base PE). A possible procedure would be to represent each wave number component M, N by a single Processing Element and perform the internal operations to form the various required interaction coefficients for that particular set of wave number components. The next step would then be to shift the original wave component data one Processing Element to

the right (i. e., to east neighbor), then repeat the above computations, leaving the original data stored in the original base PE. This procedure could then be repeated 32 times until all possible combinations for that particular row (and hence all rows in the network) have been completed. The third step would be to shift one more PE to the right (made necessary to restore original data to its original location), then shift all data down one row of Processing Elements (and hence shift data down one row for all PE's). The procedure outlined previously is then repeated for this next configuration of wave number data until all 32 PE's are again accounted for. The data for this row of calculations are then shifted down again one row and the computations again shifted across the new row. This double shifting procedure is repeated until all possible combinations of interaction coefficients have been computed (i. e., entire spectrum of wave numbers is covered). Access from the bottom to top rows of Processing Elements and from right- to left-hand columns of Processing Elements is supplied by connections to and from these components, respectively. At each shift along a row, 32×32 or 1024 calculations per interaction coefficient are performed. For each completed row, $32 \times 32 \times 32$ or 32,768 calculations per coefficient are performed. For the entire spectrum after 32 shifts across and 32 shifts down, $32 \times 32 \times 32 \times 32$ or 1,048,576 calculations per coefficient are performed. This method will compute all of the required interaction coefficients and store them provided enough storage is available. To alleviate the storage problem, it may be possible to perform the summation concurrently with the shifting procedure, thereby accumulating the necessary interaction coefficients and parameters as required by the series expansions. This method has not yet been fully investigated but comes to mind as a possible means of employing the PE Network to good advantage for this type of problem. An alternate method would obviously be to use the General Purpose and Network Control Units to perform the required operations sequentially. The speed in this latter case would be on a par with twice that



of the IBM 7094. A third possibility (and perhaps the most feasible) would be to use both the PE Network and the Central Control Units simultaneously; the PE Network to compute the interaction coefficients by the method outlined previously and the Control Units (General Purpose and Network Units) to perform the required summation and bookkeeping. The interaction coefficients could be read out of PE memory into the Network Control unit via the L-Buffer as fast as they are computed, thereby making them available for the summation process to be performed in the General Purpose Unit concurrently. In any case, the final transformation of the solution from wave number coordinates back into spatial coordinates can be handled efficiently by the network. See figures 19-1, 19-2, and 19-3.

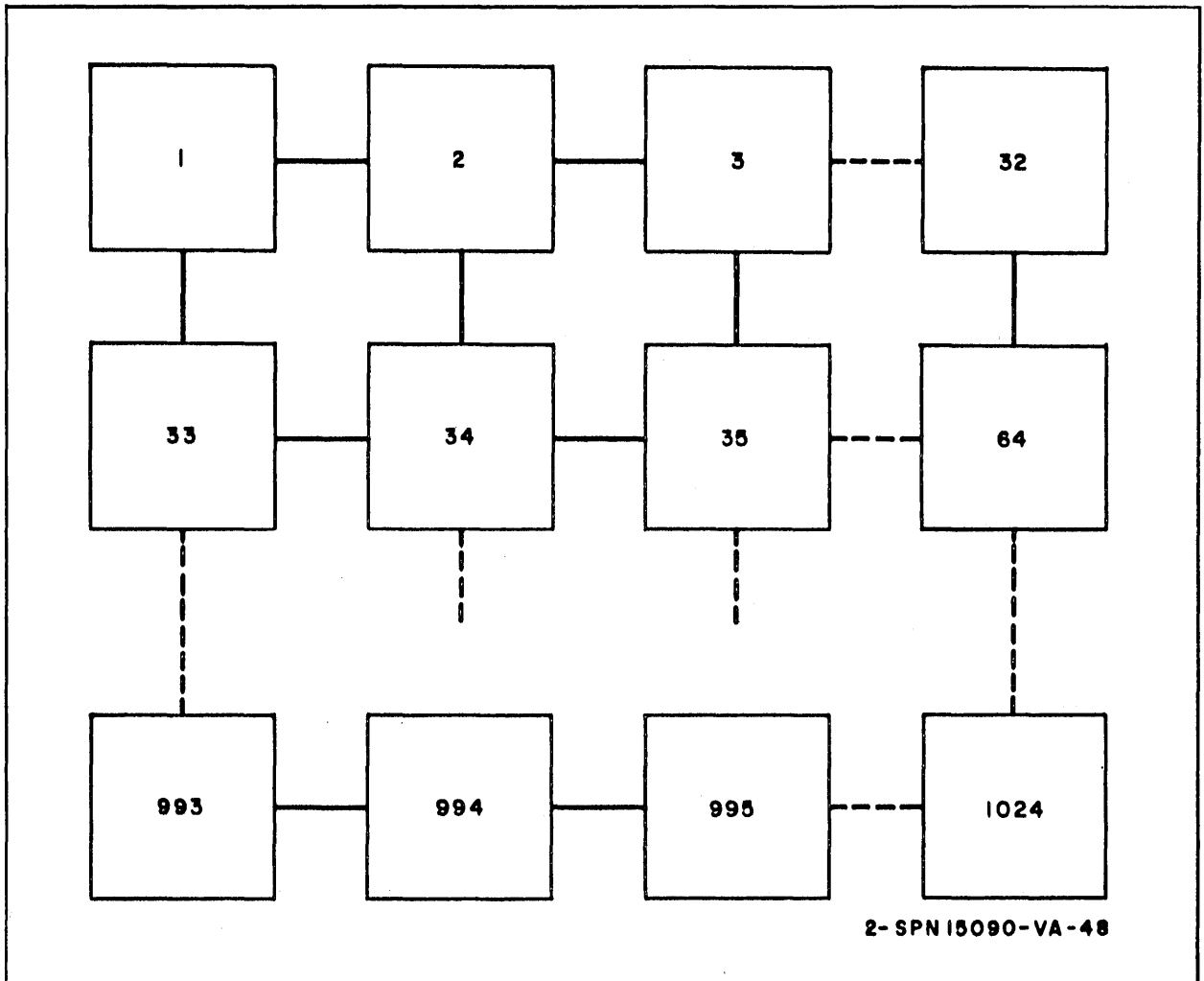


Figure 19-1. Diagrams for Network Configuration for Wave Space Computation, Initial State

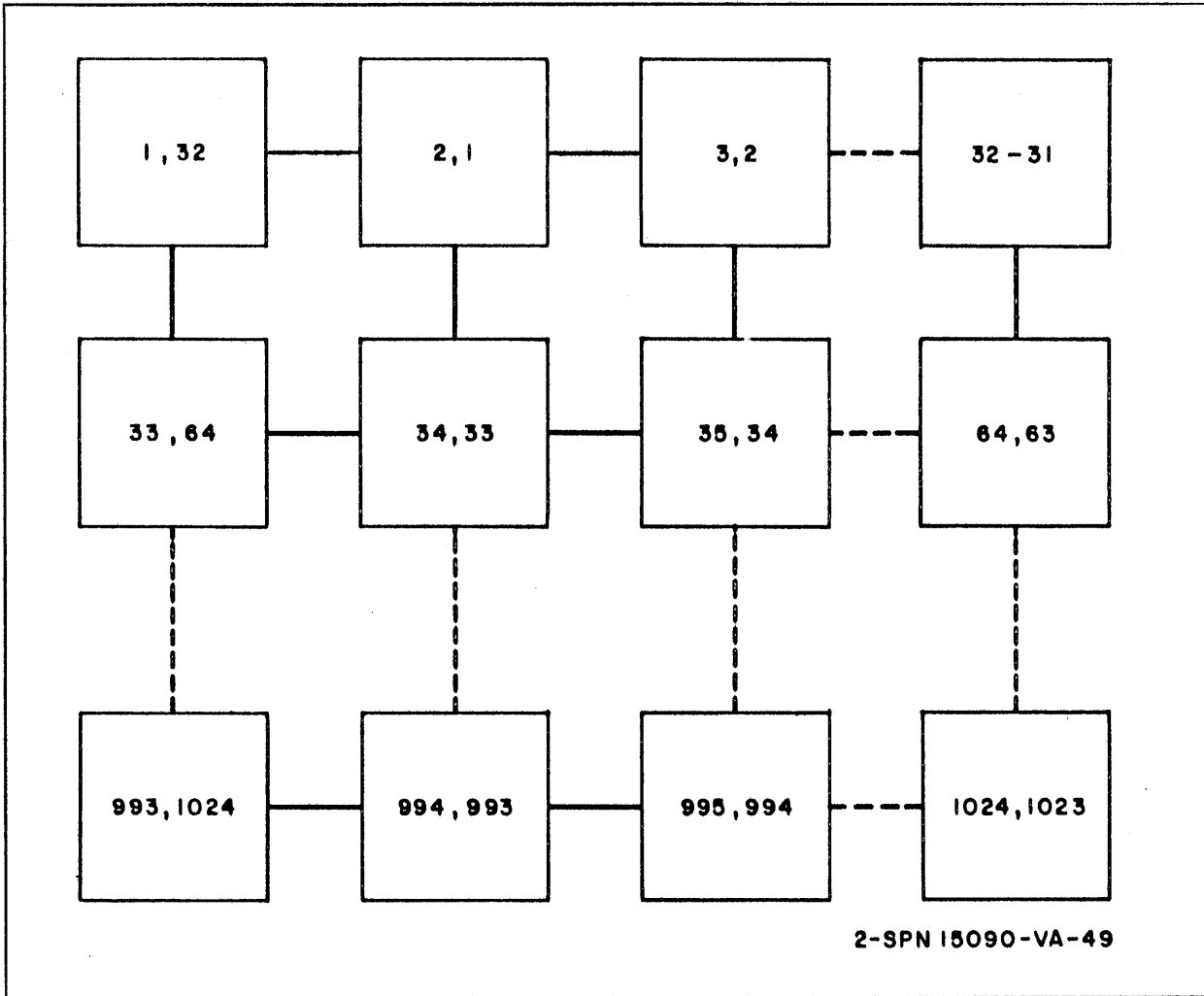


Figure 19-2. First Shift to Right

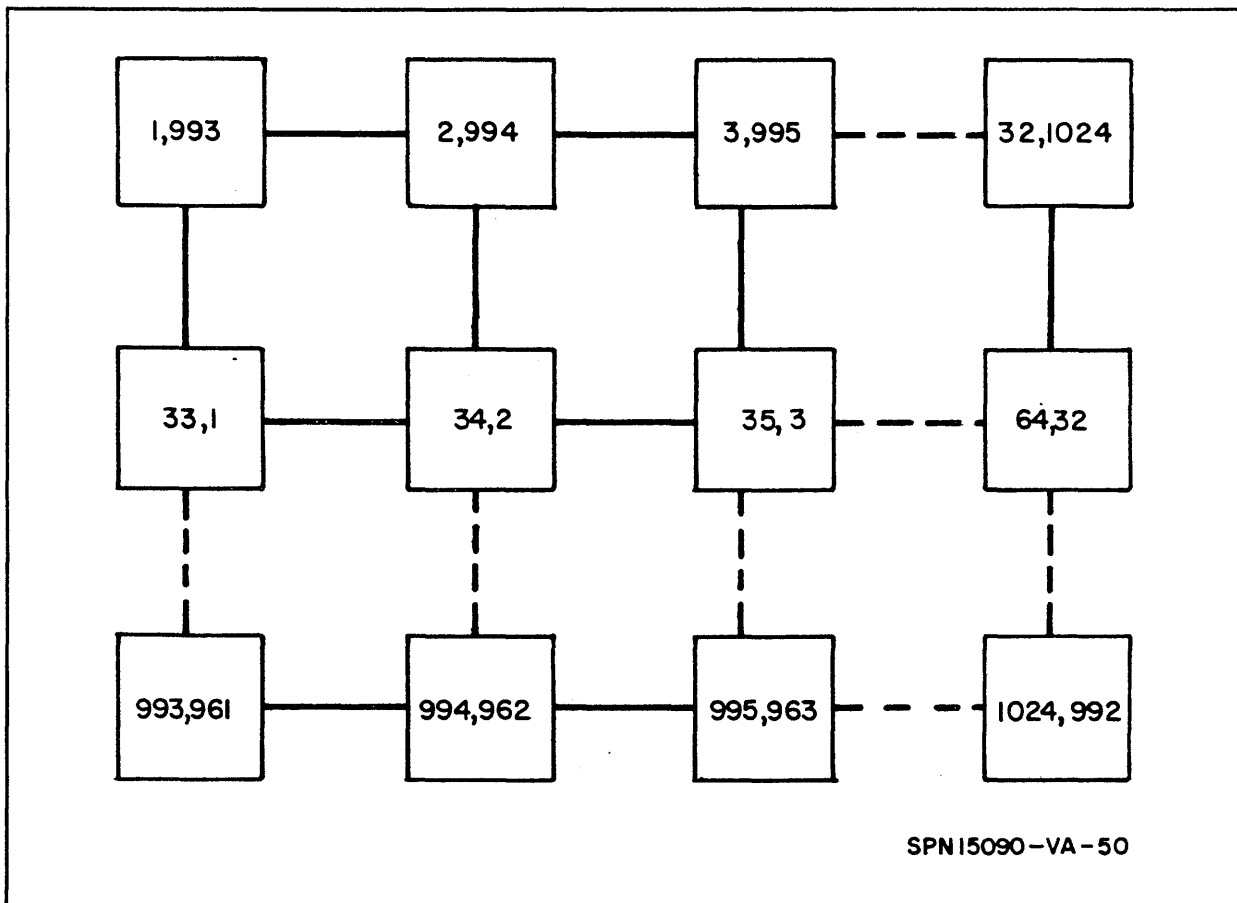


Figure 19-3. First Row Completed and First Shift Down

20. SUMMARY

The purpose of this report was to demonstrate the application of the SOLOMON II Computer to particular problems. The major emphasis has been placed on numerical weather prediction problems. To clarify these applications, detailed codes have been supplied, where applicable, to enable the reader to gain a fuller understanding of the basic characteristics of the SOLOMON II System. The example programs illustrated the concept of writing one program to control the NCU and the PE Network. An analogous situation for a sequential computer would be the separation of bookkeeping (NCU instructions) and arithmetic (PE instructions) operations.

Mode control provides the programmer with a powerful method of controlling the network. In the relaxation routine, modes were used to control boundary points. For the relaxation example the boundary was a simple geometric shape (rectangular) and indexing on a sequential computer was straightforward. However, as the boundary becomes irregular, indexing on a sequential computer becomes much more difficult and, perhaps more important, time consuming. As an example of the utility of mode control, consider the effect of oceans on atmospheric movements. It has become increasingly apparent that the oceans are a vast reservoir of heat energy and hence play an important part in atmospheric heat transfer (at sea and air interface) at various portions of the earth's surface. Consideration of oceanographic analysis implies incorporation of the continental boundaries into the atmospheric models; boundaries which are, in general, extremely irregular in nature. To account for these irregular boundaries at the earth's surface, an extremely complicated indexing and bookkeeping subroutine would be



required to perform this type of computation on a sequential computer. This problem can be eliminated completely, with little effort, on the SOLOMON II Computer by the use of Mode Control. All points which lie on a continental boundary can be set to a different mode, thereby removing once and for all, the time consuming procedure of locating these points sequentially. The use of Mode Control with respect to variable geometry has already been described in the first portion of this report.

Short range forecasting provides another situation where Mode Control can be an extremely useful facility. For this type of problem, there are various sets and sub-sets of points to be considered in addition to variable boundary conditions (i. e. , edge and points whose values vary with time and space). These sets and sub-sets of points can be either on the boundaries or in the interior field. For this type of problem, each set or sub-set can be assigned a different mode, and the necessary operation performed separately for each set. (The method for obtaining these modes is explained in Section 3.) At first glance, this appears to be an extremely inefficient method of handling this type of problem due to the fact that there may only be a few points in a given set. However, due to the speed which the SOLOMON II System can execute the necessary instructions, all of the necessary sets or sub-sets of points can still be handled faster by Mode Control in the Network than by a sequentially organized computer. Generally speaking, the treatment of the variable boundaries varies but can again be handled by Mode Control in the same manner as for the oceanographic case. The main use of Geometric Control (a second method of network control) is during input/output. These procedures are well explained in the text of the report.

With regard to the Broadcast Register, it was stated that two options were available; to supply constants to the entire Network and to provide a north or south routine to adjacent rows of PE's respectively. The coding of the primitive equation model was written on the premise that both options were available. It should be noted that the second option is not a necessity but merely

serves to facilitate the programming. Without this option, the averaged values would have to be transferred into the nearest row of PE's (either north or south rows) in some temporary location before the forecast could proceed. Additional programming would now be necessary since two boundaries have been created which will not have available a north or south routing; hence some of the features used in the given coding could not be utilized as conveniently for this case. Mode Control would then be used to account for the boundary rows of PE's. It can be seen, therefore, that this second option provides the programmer with two important advantages.

a. All Processing Elements (including boundary PE's) can execute the given set of instructions concurrently.

b. No special programming is required at the "boundaries" except for loading the Broadcast Register with the appropriate value.

Through the routing facility, each PE has access to

a. Central memory via the Broadcast Register and L-Buffer.

b. Any of the memories of the four nearest neighbors.

c. Its own memory.

This routing facility is extremely useful in the solution of partial differential equations. This is true even in situations where data are required from Processing Elements other than the nearest four neighbors, by a given Base PE; to illustrate these features, the process of obtaining solutions to a sample global circulation model by the method of finite differences has been greatly stressed. Extensive research has shown this representation to be an extremely powerful method for solving the hydrodynamic equations of the atmosphere on a large scale for an extended period of time. Care must be taken, however, to ensure that these finite difference representations be approximately conservative in nature. For primitive equation models in general, the trend has been toward higher order differencing schemes in order to conserve energy (as much as possible) and hence preserve long range stability. It has been briefly illustrated in paragraph 17.1 how the SOLOMON II



System can be readily applied to higher order differencing schemes. From the discussion on the number of points per PE, it can be seen that this allocation can be set to any number (Section 12). (Four points per PE turns out to be an optimum case but this in no way restricts the programmer to this distribution.) It has also been shown in the forecast section of the coding (paragraph 16.8) how access is gained to data in "corner" Processing Elements with just two additional instructions. Hence the adoption of higher order finite differencing schemes will present little or no programming difficulty and should give a significant time ratio in favor of the SOLOMON II Computer, over a sequentially organized system.

It has been stated previously that the SOLOMON II System is a fixed word length machine. It has also been found that a 20-bit word will ensure sufficient accuracy for practically all numerical forecasting purposes. If however, it is desired to increase the accuracy beyond this limit, a facility for doing this is provided in the double precision routine (Section 8) where 40-bit operations can be performed in the Network. This double precision routine will, however, be roughly three times as slow as a SOLOMON II designed for a 40-bit-word length.

It should be noted that the forecasting models now being developed will place demands on a computing system that cannot be adequately satisfied by sequentially organized computers. For example, the model proposed by Philips* requires 9×10^9 multiplications for the computation alone, requiring 5.5 hours of 7090 time to generate a 24-hour forecast. Long range forecasts may go up to 96 hours, while experimental atmospheric circulation forecasts could range up to 200 days or more. The bookkeeping instructions (checking for overflow, scaling in a fixed point routine, input/output, etc) may increase the execution time by a factor of 2 or 3. Also this model requires roughly 270,000 words of data, implying that the complete data set cannot be stored in the core memory and must be retrieved and restored for each time step.

* N.A. Philips, "Advances in Computers," Numerical Weather Prediction, Academic Press, New York, 1960, pp. 43-86.

In atmospheric circulation computations, it is generally required that the data for two time steps (i. e., data at time $t-1$, and time t) be retained for computation of the forecast variables at time $t+1$. In addition to this, a working storage must be provided for the necessary operations to obtain these required new values. In the SOLOMON II System, the storage available per Processing Element is, for a 20-bit word, 2000 words in a 40,000-bit memory. The total PE array storage is 1024×2000 or 2,048,000 words; $1024 \times 40,000$ or 40,960,000 bits. These data are in the random access core memory. The storage per PE can be increased to 8,000 words in a 160,000-bit memory or a total Network storage of 8,192,000 words (163,840,000 bits). To augment the PE core storage, the SOLOMON II System will have a disk storage system. Up to 16 disks are available and each disk can store 53,760,000 bits or $53,760,000/20 = 2,688,000$ words. These disks are controlled by a data channel and communicate with the PE's via the L-Buffer. Data rates are available in SOLOMON Project Technical Memorandum No. 25, "The SOLOMON II Physical Characteristics."

For many general circulation models, this Network storage will be sufficient to store most (if not all) of the required levels of data at one time, thereby reducing considerably (if not completely) transferring of data from disk into the Network and back again due to storage limitations. For example, take the case of a 9-level model containing 10,000 grid points per level with 7 parameters to be carried at each point. The total apparent storage would then be $10,000 \times 9 \times 7$ or 630,000 words. But this must be available for three time steps at a time (i. e., time steps $t-1$, t , and working storage for time $t+1$) which gives a total storage requirement of $630,000 \times 3$ or 1,890,000 words. Since the PE Network has 2,048,000 words of storage, it can be seen that even for this large storage requirement, the problem could conceivably be stored completely in the Network (3 time steps at a time) and run without unnecessary data transfer each time step. This feature would



also provide faster and more complete forms of output since the entire level of data would be available at a given time. This is particularly desirable when requesting output on a global basis.

The associative memory property of SOLOMON II provides an extremely fast method of table look-up. Since a major portion of compiling time is consumed in table look-up, the table look-up (and large storage) should provide a reduction in compiling times.

As was stated in paragraph 16.8, the time estimate for coding of the primitive equation model was subdivided into the following three categories - computation time for initial guess field and initial data; computation time for one iteration of the relaxation procedure; and computation time for one time step in the forecast equations (to include averaging procedure at poles). From the results obtained from the coding, the following table can be constructed:

TABLE 20-1
TIME ESTIMATES

Computation	Time (Minoseconds)
Wind field	198.0
$F_{i,j}$	<u>451.0</u>
Total	649.0
One relaxation for initialization	<u>1070.0</u>
ϕ Position No. 3 (per time step)	167.0
U' Position No. 2 (per time step)	149.6
V' Position No. 5 (per time step)	176.2
ϕ' Position No. 6 (per time step)	164.8
U' Position No. 4 (per time step)	147.6
V' Position No. 1 (per time step)	171.6
ϕ' Position No. 8 (per time step)	169.2
ϕ Position No. 1 (per time step)	171.4
V Position No. 2 (per time step)	171.6
V Position No. 4 (per time step)	165.4

TABLE 20-1 (Continued)

Computation	Time (Minoseconds)
U Position No. 5 (per time step)	147.8
U Position No. 7 (per time step)	147.6
Averaging procedure (per time step)	<u>290.0</u>
Total	2239.8

From the table given above, the total time estimates for the three respective sections are: computation of wind field and $F_{i,j}$ - 649 microseconds; one iteration of relaxation procedure - 1070 microseconds; one time step in forecast equations - 2239.8 microseconds.

The results from the Barotropic Model are:

Barotropic Model - 804 microseconds.

The time ratio 2000:1 obtained for the SOLOMON II System compared to the IBM 7090 with respect to the Barotropic Model is extremely large. This is an indication of the speed advantages which can be obtained on the parallel organized SOLOMON II System, over a sequentially organized computer (7090) for the idealized case considered.

A running time estimate was made on the primitive equation model presented in this paper for the SOLOMON II System. Since a corresponding coding for a sequential computer is not yet available, a direct comparison between the two systems is not possible at the present time. It is felt, however, that when this is done, the result will show the SOLOMON II Computer to be substantially faster (by more than an order of magnitude) than the fastest sequentially organized computer currently on the market.



BIBLIOGRAPHY

1. Haltiner and Martin, "Dynamical and Physical Meteorology," McGraw-Hill Book Co., New York, 1957.
2. N.A. Philips, "Advances in Computers," Numerical Weather Prediction, Academic Press, New York, 1960, pp. 43-86.
3. N.A. Philips, "Numerical Integration of the Primitive Equations on the Hemisphere," Monthly Weather Review, September 1959, U.S. Dept. of Commerce, Weather Bureau, pp. 333-345.
4. B. Saltzman, "Numerical Solution for an Idealized Barotropic Flow," Studies of the General Circulation III, M.I.T. Dept. of Meteorology, Boston, Mass., December 1959, pp. 125-146.
5. F.G. Shuman, "Numerical Experiments with the Primitive Equations," Proceedings of International Symposium on Numerical Weather Prediction in Tokyo, 7-13 Nov. 1960, Meteorological Society of Japan, March 1962, pp. 85-107.
6. P.D. Thompson, "Numerical Weather Analysis and Prediction," Macmillian Company, New York, 1961.
7. "SOLOMON II Reference Manual," SOLOMON Project Technical Memorandum No. 23, Westinghouse Defense and Space Center, Systems Division, 1963.
8. "SOLOMON II Physical Characteristics," SOLOMON Project Technical Memorandum No. 25, Westinghouse Defense and Space Center, Systems Division, 1963.



