INTRODUCTION AND SUMMARY

The SOLOMON (Simultaneous Operation Linked Ordinal MOdular Network), a parallel network computer, is a new system involving the interconnections and programming, under the supervision of a central control unit, of many identical processing elements (as few or as many as a given problem requires), in an arrangement that can simulate directly the problem being solved.

The parallel network computer shows great promise in aiding progress in certain critically important areas limited by the capabilities of current computing systems. Many of these technical areas possess the common mathematical denominator of involving calculations with a matrix or mesh of numerical values, or more generally involving operations with sets of variables which permit simultaneous independent operation on each individual variable within the set. This group is typified by the solution of linear systems, the calculation of inverses and eigenvalues of matrices, correlation and autocorrelation, and numerical solution of systems of ordinary and partial differential equations. Such calculations are encountered throughout the entire spectrum of problems in data reduction, communication, character recognition, optimization, guidance and control, orbit calculations, hydrodynamics, heat flow, diffusion, radar data processing, and numerical weather forecasting.

An example of the type of problem permitting the use of the parallelism is the numerical solution of partial differential equations. Assuming the value of a function, $u$, is known on the boundary, $\Gamma$, of a region, the solution of the Laplace equation† can be calculated at each mesh point, $x, y$ in the region as illustrated in Figure 1.

Since the iteration formula is identical for each mesh point in the region, the arithmetic capability provided by a processing element corresponding to each point will enable one calculation of the equation; i.e., a single program execution, to improve the approximation at each of the mesh points simultaneously.

Figure 2 illustrates a basic array of processing elements. Each of these elements possesses 4096 bits of core storage, and the arithmetic capabilities to perform serial-by-bit arithmetic and logic. An additional capability possessed by each processing element is that of communication with other processing elements. Processing element $E$ can

---

*The applied research reported in this document has been made possible through support and sponsorship extended by the U.S. Air Force Rome Air Development Center and the U.S. Army Signal Research and Development Laboratory under Contract Number AF30(602)2724: Task 730J. It is published for technical information only, and does not necessarily represent recommendations or conclusions of the sponsoring agency.

transmit and receive data serially from its four nearest neighbors: the processing elements immediately to right, A; left, C; above, B; and below, D.

A fifth source of input data is available to the processing element matrix through the "broadcast input" option. This option utilizes a register in the central control to supply constants when needed by an arbitrary number of the processing elements during the same operation cycle. This constant is treated as a normal operand by the processing elements and results in the central control unit becoming a "fifth" nearest neighbor to all processing elements.

The processing element array is the core of the system concept; however, it is the method of controlling the array which turns this concept into a viable machine design. This method of control is the simplest possible in that the processing elements contain a minimum of control logic - the "multimodal" logic described below.

Figure 3 illustrates how the processing element array, a 32 x 32 network, is controlled by a single central control unit. Multimodal control permits the processing elements to alter control signals to the processing element network according to values of internal data. They are individually permitted to execute or ignore these central control signals.

Basically, the central control unit contains program storage (large capacity random-access memory), has the means to retrieve and interpret the stored instructions, and has the capability, subject to multimodal logic, to cause execution of these instructions within the array. Thus, at any given instant, each processing element in the system is capable of performing the same operation on the operands stored in the same memory location of each processing element. These operands, however, may all be different. The flow of control information from the control unit to the processing elements is indicated in Figure 3 by light lines. An instruction is retrieved from the program storage and transmitted to a register in central control. Within central control, the instruction is interpreted and the information contained is translated into a sequence of signals and transmitted from central control to the processing elements. Since this information must be provided to 1024 processing elements, it is necessary to branch this information and provide the necessary amplification and power. This is accomplished by transmission through branching levels, which provide the necessary power for transmission.

As described above, each processing element in the network possesses the capability of communicating with its four adjacent elements. The "edge" processing elements,
however, do not possess a full complement of neighbors. The resulting free connections are used for input-output application. This makes possible very high data exchange rates between the central computer and external devices through the input-output subsystem. These rates could be still further increased by providing longer "edges"; i.e., by the use of a nonsquare network array.

Two input-output exchange systems are used by the input output equipment. The primary exchange system is a high speed system operating at a data rate near that of the processing element network. This system consists of magnetic tapes and rotating magnetic memories and serves the network with data storage during large net problems.

The secondary exchange system provides the user with communication with the primary exchange system through conventional high speed printers, and tape transports. The data at the output of this system is compatible with most conventional devices.

The Processing Element

The processing element (PE) logic, illustrated in Figure 4, basically consists of two parts: the processing element memory, and the arithmetic and multimodal control logic.

The multimodal control within each processing element provides the capability for individual elements to alter the program flow as a function of the data which it is currently processing. This capability permits the processing element to classify data and make judgments on the course of programming which it should follow. Whenever individual elements are in a different mode of operation than specified by central control, they will not execute the specified command.

During each arithmetic operation, one word will be read serially from each of the two memory frames associated with a unique processing element. The operand in frame one will be transmitted by central control command, either to the internal adder or to that of one of the four adjacent elements which are its nearest neighbors in the network array. The five gates labeled A in Figure 4 control the routing of information from frame one. Since only one of these may be activated during a single operation, a word in frame one can be entered in the operation select logic of only one of the five processing elements. The frame-two operand can be routed only into the unit's full adder.

Each PE in the system will communicate with a corresponding unit, thereby producing a flow of information between processing elements during network operations.

Word addressing of the memory is performed by the matrix switches in the central control unit. These switches convert the address from the binary form of the instruction to the one-of-n form required for addressing the memory frame. Provision is made for special addressing of specific memory locations for temporary storage of multiplier and quotient during multiplication and division. Successive bits are shifted into the PE logic by two digit counters in central control.

Three different types of storage are permitted: (1) the sum can be routed into frame one while the original word in frame two is rewritten; (2) the sum can be routed into frame two while the word in frame one is rewritten; and (3) information can be interchanged between frames. Note that in the first two operations, the word which was located in the memory frame into which the sum is routed is destroyed. No information is altered during the third type operation.

Multimodal Operation: Multimodal operation gives the processing element the additional capability for altering program flow and tagging information on the basis of internal data. Any command given by the
central control unit to the PE matrix is executed by the processing element only when the mode control signals from the control sequencer are identical with the coding of the processing element mode register. The central control unit may activate any combination of four states permitting a given command to be executed by individual elements in different multimodal states.

Upon comparison of a command field with the multimodal state, the execute signal is energized enabling the processing element to execute the command. When this signal is not energized, the command is not executed. If external routing is specified, the numbers in frame one are routed to the specified neighbors, regardless of mode state. The processing elements in a nonselected mode will not accept information; they will return bits of information read from memory to their respective frames unaltered.

Internally controlled mode advancement will take place as "condition met" signals are received from the arithmetic sum network. These signals transfer those processing elements which have satisfied the condition for transfer to a mode of operation specified by central control.

Modes can also be changed by the programmer by using special commands. The set mode command will automatically set all addressed processing elements into the directed mode state.

Commands for loading modes operate on the mode control flip-flops loading into or loading from two bits in memory specified by central control. The store mode command does not alter the contents of the mode control flip-flops. By programming, this capability can be used to tag information or results of calculations.

Row and Column Selection: In a number of matrix calculations, the use of a series of load and store mode commands to do row selection becomes quite cumbersome. Including the capability for commands to select particular rows or columns for the operations saves both time and processing element memory storage.

During row-column operations, the mode control logic is altered by the selection control. Processing elements execute commands in a manner identical to the ordinary mode control operations. Nonselected rows will transmit required operands, but will not alter their memory contents. The selected
processing elements operate in the conventional manner.

In combination with the multimodal operation, row-column selection is a useful programming tool. The hardware in central control consists of a holding register (switches) whose output is gated to either rows or columns. Two holding registers could permit simultaneous row and column selection.

Arithmetic Operation: Each of the two memory frames communicates with a two-bit memory buffer shown in Figure 4. Each buffer holds the bit just read from memory, along with the result of the logical operation which is about to be written into memory. The frame-one memory buffer includes a control flip-flop which can sense a bit set into the buffer during division or multiplication, as well as the condition of the summing network during a logical or arithmetic operation.

The frame-select switch controls the routing of the sum and memory bits according to the command from central control.

The arithmetic portion of the PE consists of two parts: operation selection logic and a modified full adder. Subtraction is performed by the addition of the complement of the subtrahend to the minuend. SOLOMON uses 2's complement arithmetic in the serial processing element to eliminate the endaround carry required when using 1's complement. The 2's complement is formed by an addition of 1 during the first bit cycle by initially setting the carry to a 1 and gating the complement of each bit. Therefore, a 1 is automatically added during a cycle when no carry is normally present. Negative numbers are stored in 2's complement form within the processing element memory.

Multiplication is accomplished by a series of shifts and additions. Prior to the start of the multiplication, the multiplier is stored in a specific memory location in frame one. The multiplicand is also stored in frame one. When the multiplication signal arrives, the matrix switch is set so that the first multiplier bit is read out of memory into the flip-flop. Gating then modifies the central control signals according to the value of the multiplier bit. They will provide either a set of zeros, or permit addition of the multiplicand to the partial product which is stored in frame two. To maintain the alignment of bits, the second ring is started one bit earlier than ring one. Note that the product need not be stored in the same processing element as the multiplicand and multiplier, since the multiplication can be implemented in conjunction with any of the four nearest neighbors.

Division is implemented through a nonrestoring technique. When a divisor is smaller than or equal to the dividend, a test is applied to determine if the quotient will not possess any significant bits. When this occurs, the processing element will enable the overflow signal and not divide. The central control unit can then choose to ignore the overflow or stop the computer for the operator to make corrections.

Logical operations such as "and," "or," and "exclusive or" are included in the command repertoire. Other operations such as comparisons are implemented by varying combinations of the control signals that permit flexibility within the processing element logic.

Overflow (when addition or subtraction results in a value greater than the range of the computer) can be sensed and may initiate programmed corrective routines in central control. The execute signal is also transmitted to central control for sensing to determine that some processing elements have met or have not met the transfer conditions.

Processing Element Memory Organization: Each processing element includes two memory frames each with its own read and write circuitry. Each frame has a capacity of sixty-four 32-bit words. These frames are physically organized into stacks as shown in Figure 5. The frame-one planes and earlier than ring one. Note that the product need not be stored in the same processing element as the multiplicand and multiplier, since the multiplication can be implemented in conjunction with any of the four nearest neighbors.

Division is implemented through a nonrestoring technique. When a divisor is smaller than or equal to the dividend, a test is applied to determine if the quotient will not possess any significant bits. When this occurs, the processing element will enable the overflow signal and not divide. The central control unit can then choose to ignore the overflow or stop the computer for the operator to make corrections.

Logical operations such as "and," "or," and "exclusive or" are included in the command repertoire. Other operations such as comparisons are implemented by varying combinations of the control signals that permit flexibility within the processing element logic.

Overflow (when addition or subtraction results in a value greater than the range of the computer) can be sensed and may initiate programmed corrective routines in central control. The execute signal is also transmitted to central control for sensing to determine that some processing elements have met or have not met the transfer conditions.

Processing Element Memory Organization: Each processing element includes two memory frames each with its own read and write circuitry. Each frame has a capacity of sixty-four 32-bit words. These frames are physically organized into stacks as shown in Figure 5. The frame-one planes and
frame—two planes make up separate stacks (stack I and stack II). These stacks are driven in parallel from central control.

Both sets of X-drivers are controlled from central control and select one of 64 words in each frame (see Figure 6). The word selected from frame 1 can be different from the word from frame 2. While the selected X-drivers are turned on (read-write), the Y-drivers will sequence through the bit positions with a series of read-write pulses, and thus cycle the bits of both selected words serially into the sense amplifiers. The Y-drivers of stack I can be offset by any number of bit positions from the Y-drivers of stack II. The outputs of the sense amplifiers are sent to the buffer flip-flop register where they are gated by the processing element either into its own adder or to a designated nearest neighbor.

![Figure 6. Basic Operation of Memory Unit](https://www.computerhistory.org)

Writing the information back into the memory will be accomplished in a similar fashion, with the exception that the flip-flop buffer will control the digit driver, which, in turn, determines whether a one or a zero is written.

Drivers of convenient size and complexity can be built to drive up to 128 frame pairs. Therefore, a 1024-frame memory requires eight sets of X- and Y-drivers. When expanded systems are desired, an additional unit is provided with each module of 128 processing elements.

Central Control

The Control Unit: The purpose of the Control Unit, Figure 7, is to control the operations of the SOLOMON computer complex and to maintain the proper command distribution. This unit is the only one which addresses the program memory. All indexing, whether it is to be performed on an input-output command or a processing element command is completed within this unit. The unit has control over all broadcast and index registers. Loading, transferring, and other operations upon these registers are accomplished while processing element matrix sequencing and input-output control is taking place.

The control unit has sufficient command decoding logic to discriminate between four basic types of commands: (1) processing element commands, (2) input-output control commands, (3) program control commands, and (4) commands which transfer information between the matrix and the input-output equipment.

In processing element commands, the control unit addresses the program memory sequentially, and receives the instruction. The control unit ascertains that the instruction is intended for the sequencer, and then does the required indexing. When the sequencer has completed its previous instructions, the indexed command is transferred in parallel to the network sequencer. The control unit then addresses the program for the next instruction.

The input-output control commands are partially decoded and the addresses are indexed. When the input-output control has completed its previous instruction, the information from the program memory is transferred to the input-output executor.

The program control commands are completely executed within the control unit unless inputs are required from other control complexes. If the former is true, the control unit will complete the instruction and then return to the program memory for new instructions. When inputs from the other units are necessary, the control unit must wait until the device is not busy. Then the command will be executed. This class of instructions includes console control interrupt tests, switch test and control program, transfer instructions, and register controls.

The final class of instructions are those where both the input-output control and network sequencer are required to act together. With this type, both the sequencer and the input-output executor must have completed their previous instructions, then, with indexing completed, the command is transferred in parallel to the network sequencer, the
control of the input-output executor is transferred to the sequencer, and the control unit is released to obtain the next instruction.

This organization of control will provide the programmer with the flexibility to maintain a maximum utilization of all control equipment, see Figure 8. Interweaving instructions will provide a maximum amount of command overlap and results in the highest speed.

Network Sequencer: The sequencer, shown in Figure 9, is the portion of the SOLOMON system whose major objective is to provide control signals for commands involving the processing element matrix. The device provides control signals to the input-output unit during information transfers.

The unit has two major functions, one to decode commands and provide control pulses and levels to the processing element matrix, and the second to control memory addressing.

Memory address control is provided by two digit counters and two matrix switches. These counters are loaded by the control unit and will advance or decrement under control of the operation register and controller.

While memory addressing is mainly accomplished by counters, the processing element controls are more varied. After the operation register is loaded from the control unit, it is decoded and supplies command signals to the controller. The controller supplies all the time-varying signals to the processing element matrix and transmits control pulses to other sequencer registers.
The controller has the capability of selecting a broadcast option which will enable the central control to act as a "fifth" nearest neighbor to the network. The frame one operand is replaced by one of the broadcast registers in the Control Unit. This option will be selected by the programmer whenever a single constant is required by a large number of processing elements during the same operation cycle.

During input-output operations, the controller sends the coded signals to the input-output control unit for the duration of the operation. The input-output unit then transmits the control pulses to the input-output equipment to maintain exact synchronization between the two subsystems.

The Input Output System: The input output system, Figure 10, is organized to facilitate information transfer between the network.
and auxiliary equipment with high efficiency.

The primary information exchange is used as auxiliary network storage in large mesh problems. The exchange consists of wide multi-channel magnetic tape whose function is to keep a high speed drum loaded. This drum operating with parallel channels can provide a bit rate comparable with the network rate. Other advanced storage devices are being considered in this area.

This primary exchange system communicates through the input output buffer with the secondary exchange system. This secondary exchange system provides the user with compatibility with existing systems such as conventional magnetic tape, card reader and punch, high speed printers, paper tape units and other devices. The Format Converter provides conversions from the binary form required by the network to the format required by the peripheral devices.

Both the primary exchange and the core buffer can communicate with the processing element network in two ways, the "edge" elements and geometric control.

In the first method, since each element has the capability of communication with its four adjacent neighbors a processing element located on the "edge" of the network has one or more free connections. These unused connections are connected to the input-output register.

Figure 11 shows the $32 \times 32$ matrix array with the capability to select the particular edge into which information is to be written. This flexibility increases the ease of loading the processing element network.

Additional flexibility is obtained through "geometric control." This technique employs special information transfer commands that enable the programmer to select specific rows or columns which are to communicate directly with the input output register. This eliminates the necessity of multiple transfers to obtain or load information directly into the interior of the matrix.

Programming

In the overall evaluation of any computer system, consideration must be given both to
the mathematical formulation of the problem to be solved and to the way in which this is reduced to a sequence of computer operations. Both of these aspects have a particular significance in the SOLOMON system. It is becoming increasingly apparent that the traditional methods of numerical analysis will be largely inapplicable to the new generation of computers represented by SOLOMON.

Present analytical and programming methods have evolved directly and with little change from centuries of hand calculation and later from use of desk calculators. For this reason, such methods have been considered "natural," and the organization of present computing systems may be directly compared to the computing complex consisting of a human operator, an explicit computational algorithm, and mechanisms (or persons) capable of performing the required sequence of calculations. Heretofore, no serious attempt has been made to employ an approach to computer design which is based not on human capability and "natural" computation methods, but on potential computer capability and entirely new methods "natural" to the computer innovation.

Current developments demand an alteration of this state of affairs. The SOLOMON system, as a consequence of its radically different organization, requires new techniques in numerical analysis and programming for its effective utilization.

The main question is the amenability of basic mathematical functions and processes to the parallel approach. Specifically, in the SOLOMON system a fixed number of processing elements under common control must be efficiently used to perform calculations which will vary both in size and in basic type. A fixed number of processing elements can both speed the solution of a given problem of fixed size and perform substantially different but mathematically related calculations simultaneously. The former capability takes advantage of the parallelism which is intrinsic to many problems but which has not been previously exploited.

Figure 11. Input Variations Possible With 32 X 32 Array
The latter stems from the capability to represent broad classes of functions and operators mathematically in normal forms which distinguish between the represented quantities only by the values of constants that occur in their representations; examples of this are the representation of continuous functions by polynomials, of analytic functions by Taylor series, and of linear operators by matrices.

A previous paper describes in detail the application of the SOLOMON system to problems in partial differential equations and certain matrix calculations. Results to date establish a performance advantage between 60 and 200 for the SOLOMON Computer compared to currently available large scale digital systems.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the assistance received through many discussions with their associates during the conception and development of the SOLOMON system, and, in particular, E.R. Higgins, W.H. Leonard, and Dr. J.C. Tu.


Jeeves, T. A., op cit.