BIBLIOGRAPHY

[1] D. Braito, *et al.*, "La calcolatrice CEP del centro Studi Calcolatrici Elettroniche dell'Univerità di Pisa (CSCE)," *Alta Frequenza*. vol. 30, pp. 873–876; December, 1961.
[2] M. V. Wilkes, "The best way to design an automatic calculating machine," *Manchester University Inaugural Conf.*, Manchester, England, pp. 16–18; July, 1951.
[3] M. V. Wilkes and J. B. Stringer, "Micro-programming and the design of the control circuits in an electronic digital computer," *Proc. Camb. Phil. Soc.*, vol. 49, pt. 2, pp. 230–238; April, 1953.
[4] M. V. Wilkes, "Microprogramming," *Proc. Eastern Joint Computer Conf.*, Philadelphia, Pa., pp. 18–20; December 3–5, 1958.
[5] A. Grasselli, "The Design of a Program-Modifiable Micro-Programmed Control Unit," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-11, pp. 336–339; June, 1962.
[6] H. Hagiwara, K. Amo, S. Matsushita, and H. Yamauchi, "The KT pilot computer, a micro-programmed computer with a phototransistor fixed memory," *IFIP Cong. '62*, Munich, Germany, pp. 684–689; August 2–September 1, 1962.
[7] G. P. Dinneen, I. L. Lebow and I. S. Reed, "The Logical Design of CC-24," *Proc. Eastern Joint Computer Conf.*, pp. 91–94; December 3–5, 1958.
[8] G. B. Gerace, "Un sistema di controllo ad alta velocità per calcolatrici elettroniche digitali," *Automazione e Strumentazione*, No. 11, pp. 533–538; November, 1961.
[9] G. Cecchini and G. B. Gerace, "L'organizzazione logica dell' unità di controllo della CEP," University of Pisa, Rome, Italy, Internal Rept. CSCE, Series I, No. 36; June, 1963.
[10] G. B. Gerace, "La temporizzazione, la minimizzazione dei tempi ed il controllo dinamico della CEP," University of Pisa, Rome, Italy; Internal Rept., CSCE, no. 29; December, 1961. Also in *Alta Frequenza*, vol. 32, pp. 184–191; March, 1963.
[11] C. Strachey, "Time sharing in large, fast computers," *Proc. IFIP'59*, UNESCO, Paris, France; June 15–20, 1959; pp. 336–341.
[12] B. J. Loopstra, "Input and output in the X-1 system," *Proc. IFIP'59*, UNESCO, Paris, France, pp. 342–344; June 15–20, 1959.
[13] E. F. Codd, "Multiprogramming," in "Advances in Computers," Academic Press, New York, N. Y., p. 78; 1962.
[14] G. Felton, "Multi-programming on the Ferranti Orion Computer System," *Proc. IFIP Congress '62*, Munich, Germany, pp. 570–571; August 27–September 1, 1962.
[15] M. V. Wilkes, *et al.*, "Panel on Priority Problems in Computer Systems," *Proc. IFIP Congress '62*, Munich, Germany, pp. 711–715, August 27–September 1, 1962.
[16] E. S. McDonough, "STRETCH Experiment in Multiprogramming," *ACM 62 Natl. Conf. Digest of Technical Papers*, p. 28; September 4–7, 1962.
[17] M. V. Wilkes, W. Renwick, and D. J. Wheeler, "The Design of the Control Unit of an Electronic Digital Computer," *Proc. IEEE*, vol. 105, pt. B, pp. 121–128; March, 1958.
[18] T. W. Kampe, "The Design of a General-Purpose Microprogram-Controlled Computer with Elementary Structure," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 208–213; June, 1960.
[19] L. B. Emelyanov-Yaroslavsky and A. A. Timofeev, "Microprogram Control for Digital Computers," *Proc. IFIP Congress '62*, Munich, Germany, Discussion, p. 569; August 27–September 1, 1962.
[20] T. Kilburn and L. Grimsdale, "A digital computer store with very short read time," *Proc. IEE*, pt. B, vol. 107, pp. 567–572; November, 1960.

Parallel Processing in a Restructurable Computer System*

G. ESTRIN†, MEMBER, IEEE, B. BUSSELL†, MEMBER, IEEE, R. TURN†, MEMBER, IEEE, AND J. BIBB†

Summary—**Pragmatic problem studies predict gains in computation speeds in a variety of computational tasks when executed on appropriate problem-oriented configurations of the variable structure computer. The economic feasibility of the system is based on utilization of essentially the same hardware in a variety of special purpose structures. This capability is achieved by programmed or physical restructuring of a part of the hardware. Existence of important classes of problems which the variable structure computer system promises to render practically computable, as well as use of the system for experiments in computer organization and for evaluation of new circuits and devices warrant construction of a variable structure computer. This paper describes the organization, programming, and hardware of a variable structure computer system presently under construction at UCLA.**

I. INTRODUCTION

AT THE PRESENT TIME there are numerous problems in almost all areas of computer application which fall outside the realm of practicable computability. In a finite system practicable computability is a function of such mundane factors as cost of computing and total delay from formulation of a problem to interpretation or results. Added to these factors are of course round-off, truncation, significance, problem reformulation, and machine reliability, all of which determine whether there is a relation between the symbols coming out of the machine and the answer to the orginal question posed. A problem which requires uninterrupted computation for periods greater than the mean time between component failures may not be practicably computable. Similarly, those classes of problems whose solutions are real-time dependent offer situations where lengthy computation time reduces the value of the solutions.

Throughout the less than two decades of history of large-scale digital computer systems the designers of computer hardware, logic, numerical procedures, and programming systems have endeavored to increase the speed of computation and thereby to increase the class of practicably computable problems. The early large increases in speed resulted from replacement of external programs by stored programs, and from discarding electromechanical relays as switching and memory de-

* Received August 19, 1963. Preparation of this paper was sponsored in part by the Office of Naval Research and the Atomic Energy Commission. Reproduction in whole or in part is permitted for any purpose of the U. S. Government.
† Department of Engineering, University of California, Los Angeles, Calif.

vices in favor of all-electronic devices. Further speed increases have resulted from development of faster switching devices for logic and memories, and from new logical design philosophies and system organizations.

At present, switching elements which are capable of changing state in approximately one nanosecond and large scale magnetic core memories with 300-nanosecond access times (1-microsecond cycle times) are a reality, while hot electron devices in laboratories promise switching times in the order of 10^{-13} seconds. Large aggregates of switching elements, however, characteristically show speed losses of several orders of magnitude in parallel transfer operations.

Coupled to the advances in computer hardware has been a rebirth of interest in numerical analysis.The new research in numerical mathematics, in turn, allowed further freedom in formulation of problems for digital solution. This resulted in stronger demands on the digital computer which spurred further technological advancements of computers. A step function increase in the communication between user and computer occurred as a result of the development of artificial languages and automatic compilers which reduced the tedium of problem preparation.

It is safe to conjecture that a general purpose computer constructed from the exotic hot electron switching elements and equipped with all the sophistication of present-day large scale computers will not exhaust the class of problems presently not practicably computable. A general purpose computer is principally a compromise in establishing a word length, in selecting the arithmetic algorithms, and in fixing the instruction list. It attempts to cater to a wide variety of problems and to solve such problems with a reasonable speed. However, it generally does not contain features which permit faster solution of very small classes of problems.

In recent years it has become increasingly apparent that significant increases in the class of practicably computable problems can be achieved by building computers oriented toward fast and efficient solution of restricted classes or problems. Such special purpose computers can be constructed from fastest available components, and they can utilize unusual system organization, instruction sets, number systems, word lengths, and arithmetic schemes, using parallel computation and hybrid subsystems in a manner and to an extent not economically feasible in general purpose computers.

The very principle of special purpose computation—design of hardware to enhance solution of a particular problem—also leads to a major drawback: it is difficult to adapt a special purpose computer to changes in problem formulation, in solution methods, or in computational needs. Further, economic considerations may force a facility with diverse problem interests to forego the computational power of special purpose computers in individual problem areas in favor of a general purpose computer system.

In 1959 G. Estrin [1] proposed the fixed plus variable structure computer organization as an approach to removal of structural rigidity from special purpose computation. In essence a variable structure computer system consists of a high-speed general purpose computer (the fixed part, F) operating in conjunction with a second system (the variable part, V) comprised of large and small high-speed digital substructures. The second system can be reorganized into a variety of problem-oriented special purpose configurations and this property—use of the same hardware in a variety of computing configurations—makes special purpose computation of a large class of diverse problems on the variable structure computer system feasible. The cooperation of the two systems occurs under the direction of a supervisory control unit (SC).

No such computer system exists at the present time nor can it be demonstrated that such a system would make a profitable product for industry today. There are certain large control systems which contain general purpose computers as parts but they do not contain the generality of reorganizability proposed for the variable structure computer. There are also highly parallel systems such as the French Gamma 60, the Ramo-Wooldridge RW-400, the CDC 6600, the proposed SOLOMON computer, and others, but where any of these systems consider changes in form it is at a much grosser level than defined in the variable structure system. It is in fact the definition of the mode of interaction of the variable structure part (V) with the general purpose computer (F) and the rules for growth of V which will make it possible to adapt new organizations and technology as a function of problem statements and time.

In its initial stages of evolution the variable structure system will serve most effectively in two ways: executing subsets of operations which occur rarely in conventional problems and which were therefore not considered economic in planning the repertoire of instructions of existing computer systems; and effecting simultaneous computation of operations whose order is not determined by the computational method. Thus, aside from its application as a system for solving problems which cannot be efficiently handled by conventional computers, the establishment of this system will encourage advances in parallel programming and in numerical methods which may effectively utilize simultaneity in computation. In so doing the variable structure system affords a new degree of freedom in problem solution, *i.e.*, in addition to consideration of change in the sequence of a fixed set of operations, it becomes reasonable to consider simultaneous use of new operations through electronic and mechanical reorganizations whenever they may be profitably applied.

This paper describes the organization, projected effectiveness, application, and hardware of a variable structure computer system which is presently being designed and constructed at UCLA.

II. System Organization

The concept of the variable structure computer system is based on the following premises:

1) In the solution of any given problem, a special purpose computer can be built to be more effective than a general purpose computer.

2) The essential sequential form[1] of many algorithms contains parts which may be executed simultaneously on different processors with a consequent reduction of the computation time.

3) Within the constraints of a finite hardware inventory, a greater number of computing substructures can be built if the inventory is restructurable than if it is committed to a nonvariable system.

4) Writing a compiler program for a large computer system is an effort measured in man years and is practical only if the computational characteristics (*e.g.*, instruction list and meaning of instructions) remain essentially fixed over the lifetime of the system.

These premises lead to a definition of the variable structure computer system as consisting of a fixed structure general purpose computer (F), an inventory of restructurable hardware (V), and a supervisory control unit (SC). A block diagram of the system appears in Fig. 1.

Inclusion of a fixed structure general purpose computer in the system is based on premise 4) since availability of an artificial language for programming of the system is essential. Additionally, because of its computational power, F can be used to execute complex but generally not most time-consuming parts of a problem. In the embryonic state of the variable structure system, F also executes the necessary input-output operations. The IBM 7090 computer has been selected as the initial F, although any high-speed general purpose computer may be used if it is possible to provide 1) easy and rapid access to the core, and 2) high-speed two-way communication between the fixed structure and the supervisory control.

The variable structure inventory, V, is open-ended and grows both through duplicating the modules already present in V, and through adding modules which contain new devices or circuits. The operations mechanized in V special purpose computers [premise 1)] are usually those which are among the more time consuming operations of a given problem. Together with F the special purpose computers in V form a parallel processor computer system [premise 2)].

[1] An essential sequential form may be defined in the context of unlimited parallelism. Thus, starting with a computer (i.e. instruction list and associated times) and an associated program, if one assumes that an unlimited number of the major computer components is available for the execution of a particular algorithm, then the essential sequential form is that sequence of series-parallel operations which minimize the computation time of the algorithm. If the hardware inventory is constrained, the procedure leading to the minimum time may be defined as a hardware-limited essential sequential form.

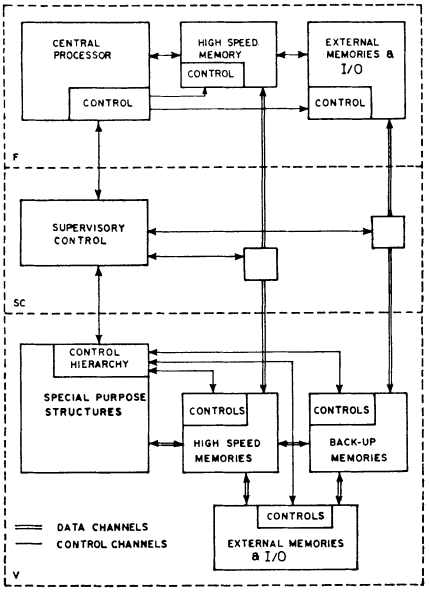

Fig. 1—Block diagram of the variable structure computer system.

Normally, *i.e.*, when it is not in a problem-oriented configuration, V is in a useful computing configuration, called the "standard state." In the standard state a number of electronically switchable configurations will be available. Such operations as computation of trigonometric and inverse trigonometric functions, logarithms, exponentials, nth powers and roots, complex airthmetic, certain matrix operations, and hyperbolic functions are almost certain to be available initially. Certain subsets of these operations may be executed simultaneously and both single and double precision computation may be available for some.

In this state V may be considered as a parallel extension of F. The computational characteristics of the standard state will be defined by an "intelligent" guess based on a number of problem studies and the standard state will vary as the inventory grows.

A. Pragmatic Investigations

The gross system organization described above was used as a framework for an investigation of the effectiveness of special purpose techniques and parallelism in computation. The measure of effectiveness was gain in speed as compared to the IBM 7090. No limit was based on the size of available inventory in order not to stifle inventiveness, but trivial solutions such as total table look-up were ruled out. The speeds of circuits in the available inventory were assumed to be similar to circuit speeds in the IBM 7090.

A summary of the promised speed gains and hardware requirements for some of these problem studies appears in Table I.

The studies in Table I indicate that special purpose structures associated with the general purpose computer F permit speed gains ranging from 2.5 to 1000.

TABLE I
SUMMARY OF SPEED GAINS

	Speed Gain	Hardware* Flip-Flops	Arithm. Units†	High-Speed Memory
Number sieve [2]	1000	340	—	—
Eigenvalues [3]	4	364	1	16 k
Dynamic programming [4]	2.5	80	1	
Parabolic partial differential eq. [5]	3.5	380	2	500
Log-exp—nth power [6]	8.5	125	1	1024
Exponential	6.0	125	1	1024
nth power	7.0			
Trig—inverse trig [7]	4	200	1	256
Random number gener. [8]	4	40	1 (serial)	

* For comparison the hardware count in flip-flops of the central processor (CPU) of the IBM 7090 is 420.
† Includes the necessary controls but excludes registers.

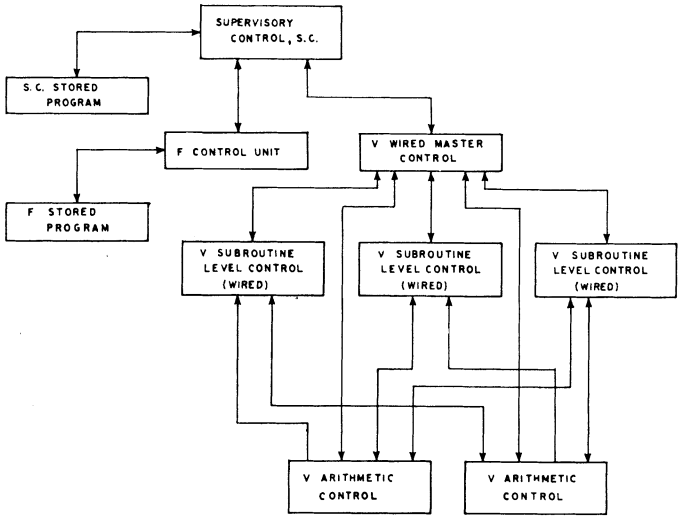

Fig. 2—Control hierarchy.

For this application the ratio of speed increase to hardware addition as a measure of effectiveness is not justified. The hardware is shared between all special purpose structures which can be constructed from a particular inventory and its cost should be ammortized over all applications over the lifetime of the variable structure system, just as is the hardware in a general purpose computer.

B. Control Hierarchy

All special purpose structures generated during the problem investigations use wired program macrocommands as a means of obtaining speed in computation. Frequently, as in a number of the above examples, changes in V structures are primarily in the control units. Therefore a special emphasis in the design of the variable structure system is placed on the design of control units, and a model suitable for automatic design has been formulated [7].

In mechanizing a wired program in V it is desirable to take advantage of any elementary or macro-operations which are available in the standard state. This leads to the design of control units which use already available macrocommand controllers, forming a control hierarchy.

A typical configuration of the control hierarchy of the variable structure computer system is depicted in Fig. 2. On the lowest control level are control units which execute arithmetic and logical operations that is, these control units correspond roughly to the arithmetic control unit in a general purpose computer. A control unit on this (lowest) level can generate only elementary commands which cause only a single control action to take place (*e.g.*, transfer from one register to another, shift by one position). One such control unit is associated with each independent arithmetic unit.

On the next higher control level are control units which may execute elementary functions, complex arithmetic, matrix operations, etc. The control units on this level usually correspond to stored program subroutines available in subroutine libraries of general purpose computers. Each such control unit may use all compound commands generated by lower level control units as its elementary commands, that is, the set of operations is more inclusive at the higher level of hierarchy. A number of consecutively higher control levels may exist each of which may specify operations performed by lower level control units as elementary commands. These control units may use stored programs when extreme flexibility is necessary.

On the highest level in the control hierarchy is the supervisory control which 1) controls the execution of computations both in F and V; 2) coordinates information exchanges between F, V, peripheral equipment, and among the autonomous computing structures in V; and 3) performs interlocking functions necessary in parallel processing. For efficient performance of this task the supervisory control may use both wired and stored programs.

Each of the control units, regardless of its control level, has to perform three basic functions: 1) generate a sequence of internal states, which correspond to sequential operations specified by the program being mechanized, 2) during each sequence state generate a subset of available elementary and compound commands, and 3) in each sequence state provide for the generation of the next sequence state as specified by the program. The next sequence state may be different from a natural sequence due to conditional branching specified by the program. Only the first of the basic functions is independent of the program provided that a large enough number of sequence states is available. The other two, necessarily, depend on the programmed algorithm. Thus, for a fixed data flow structure a change in the computational algorithm implies a change in the logical structure of the control unit. Several models of restructurable control units can be formulated. A very general approach may involve construction of branching and command matrices which permit transition from a given sequence state to any other sequence state for any

combination of completion signals, and the generation of any subset of command signals in any given sequence state. A wired program for execution of a given algorithm could be established by enabling certain subsets of intersections of these matrices under the control of internal or external programs (*e.g.*, punched cards, photographic masks in conjunction with photoelectric switches at intersections). The major advantage of such a general model is in the decreased restructuring time. Disadvantages include the high initial cost and low utilization of the hardware (Studies of the control requirements for elementary function generation have indicated that only a small fraction of the total intersections are enabled for the execution of a given algorithm.), as well as a deterioration of the execution times of the control unit due to the large arrays of switching circuits.

A less general model involves building M branching logic units and M command generation matrices to correspond with the M electronically switchable wired programs mechanized by the given control unit. In this case each of the branching logic and command matrices contains only those intersections which are required for the execution of the program. A change in the set of prewired programs, however, will require mechanical restructuring of the control unit. A model of the control unit of this type is shown in Fig. 3.

In general the control hierarchy of a particular configuration of the variable structure computer system contains several simultaneously operating control units, each of which may operate at a different basic speed. Synchronous communication between such control units may present large problems and considerable deterioration in transfer speeds, hence an asynchronous design philosophy has been adopted. Asynchronous design is based on testing for completion of an operation before the next operation is initiated. Totally asynchronous design as formalized by Muller and Bartky [9] requires that *correct* completion of every elementary operation is verified. This results in very reliable operation, but large amounts of hardware are required resulting in considerable delays. A less sophisticated asynchronous design simulates the operation times of the circuits by delay units and no completion testing is performed.

The degree of asynchronism initially adopted for the control units in the variable structure computer is a compromise between these two extreme cases. In particular, the completion signal of a command is its "echo" returning from the point of farthest application. This does not provide complete verification of completion of the operation, but does permit realistic inclusion of circuit delays.

The completion signals of macrocommands are generated by special circuits in the respective control units. A detailed description of the adopted model of the control unit, examples of designs for executing trigonometric and inverse trigonometric functions, and initial steps toward automatic logical design are given in Turn [7].

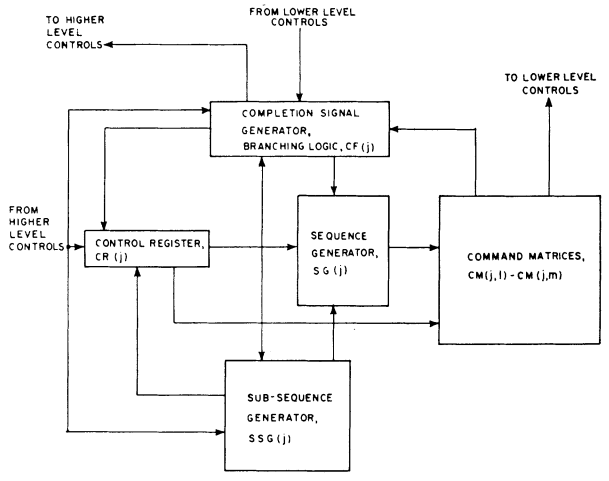

Fig. 3—Model of an electronically switchable control unit.

III. Programming

Solution of a problem on any digital computer system begins with the selection of numerical procedures. In general, a choice is made on the basis of required accuracy, computation time, convergence rate, storage requirement, and availability in the subroutine library. If the system also possesses parallel computing capability (*i.e.*, several independent processors), suitability of the numerical procedure for parallel computation enters the selection process.

In considering parallel computing it is often helpful to distinguish between two types of parallelism. One type permits simultaneous computations in various portions of the given model of the physical system; that is, computing in several local neighborhoods at the same time (global parallelism). This type of parallelism is exhibited in the proposed SOLOMON computer [10]. The other type depends upon the parallelism allowable by the mathematical formulation of the algorithm; that is, parallelism in operations associated with some local neighborhood of a point (local parallelism). An algorithm demonstrating local parallelism is discussed in a paper by W. S. Dorn [11].

Programming of computers with parallel computing capability leads to problems of optimal allocation of tasks to processors and of optimal sequencing of tasks on a given processor. These optimization problems defy easy solution and most of the present operating supervisory routines use simple first-come-first-served queuing discipline.

The capability of reorganizing the variable structure computer system into problem-oriented special purpose configurations permits further freedom in choosing numerical procedures and, simultaneously, complicates the problem of optimal programming. Numerical procedures are now examined for parallelism, both global and local, and for features which show promise of gains in speed when mechanized in V.

Programming of the variable structure computer system, in addition to assignment and sequencing of

computational statements, also includes specification of the structure of V (within the constraint of available inventory). This implies that programming of this system should be done by a team consisting of a numerical analyst, a programmer, and a computer design engineer. In order to expedite the task of such a programming team, a number of automatic and semiautomatic procedures are being formulated. A procedure and associated computer program have been completed for suboptimal *a priori* assignment of the subcomputations of a gross computational task for processing in a set of predesignated special or general purpose computing configurations available in the variable structure system [12], [13]. An automatic procedure for designing a wired program control unit for executing the computation of a given algorithm is being formulated [7].

A suitable programming language for the variable structure computer system is under study. Initially, however, the FORTRAN programming language of F (the IBM 7090) augmented by special commands for F and V interaction, and wired programs in V is used.

The procedure from problem statement to execution on a restructured configuration of the system is, in general, iterative. Initially an attempt is made to utilize special purpose structures which have already been designed for incorporation in the standard state or for solving other problems (all such configurations are retained as the "computational history" of V. A maximum acceptable cost level is established and the following steps are taken:

1) Initially the numerical procedures are selected on the basis of availability in the computational history of V. Any necessary numerical procedure not in the computational history is treated as a subroutine for F.

2) An automatic assignment procedure is applied as described by Estrin [12]. This results in a specification of the special purpose structures which must be constructed in assignment of computations to be performed by the $F+V$ system, and in sequencing of the computations.

3) If the expected cost of computation does not satisfy the requirements of the user, new numerical procedures which promise further reduction of the computation time are selected, and a computer design engineer aided with automatic procedures for designing control units generates the designs of the required special purpose structures. Step 2) above is now repeated.

4) If an acceptable prediction of the expected cost has been obtained the proposed assignment and sequencing of computations is used to generate a stored program for F, wired programs for higher levels of the V control hierarchy (by the design engineer using automatic procedures), and both stored and wired programs for the supervisory control. Simultaneously, restructuring procedures described in Section IV-C are initiated.

5) When programming and restructuring steps have been completed the problem may be put on the system. It is to be expected that modifications in the program and structure lead to iterative application of steps 4) and 5); hence a problem may occupy the system for a considerable time. At intervals when changes in program and in V structure are made, F may be used to perform routine computations (*i.e.*, the system is multiprogrammed).

Programming of a variable structure computer system requires considerable effort and time. It is expected that only problems which warrant this effort are investigated.

IV. Physical Realization

The concept of a variable structure computer system depends heavily upon the ability to effect fast, reliable, and inexpensive physical changes in the interconnections between its elements. These requirements are unique to the variable structure computer system: computers usually experience the "pain" of interconnecting their circuits only once and, thus, it is feasible to expend a considerable time in wiring and in ascertaining that the connections so made are correct and reliable. Consequently, the circuits, their packaging and interconnection schemes in the variable structure computer, must reflect the required flexibility such that the predicted system gains due to organization will not be destroyed.

A. Modular Structure

The present first pass at module design has resulted in specification of two basic types of switching circuits and a modular scheme of packaging and construction. The basic building blocks are two types of pluggable modules: one module contains four amplifiers and associated input logic for signal inversion, amplification, or high-speed storage (in its flip-flop interconnection); the other module is primarily for combinatoric application. A combinatoric module consists of ten diodes and four output drivers. A prewired pluggable cap is attached to define its logical characteristics and to perform structure change on the switching circuit level. Fig. 4 displays the packaging designs of the basic modules.

Each basic module may be inserted into any of the 36 positions of a motherboard (Fig. 5) which also serves as a low impedance multilayer power distribution system. Connections between basic modules on a motherboard are provided by a replaceable flexible printed wiring harness (Fig. 6) which connects to the feed-through pins. Restructuring of the system on the motherboard level is effected by replacing one wiring harness with another.

Ten motherboards are mounted in each of the three bays of a frame. Each bay contains a multilayer power distribution system with an independent set of voltage regulators. Mounted on each end of a motherboard is an endplate on which are mounted four 50-pin connectors. These connectors are used for communication between

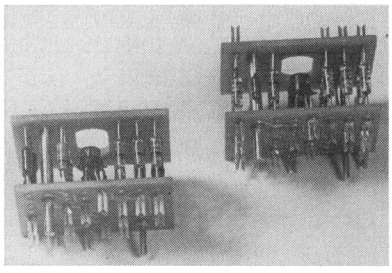

Fig. 4—Basic modules.

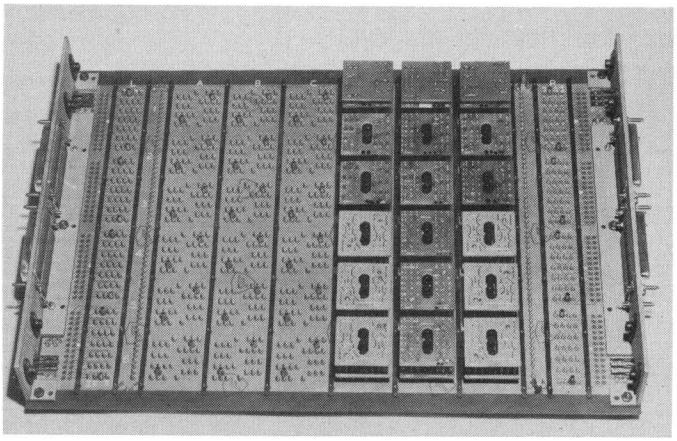

Fig. 5—Motherboard.

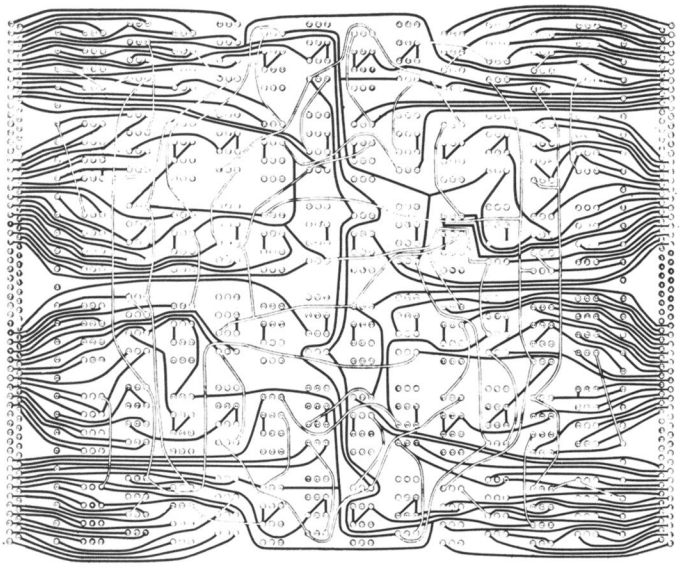

Fig. 6—Wiring harness for the motherboard.

motherboards. Although standard wiring techniques could be used for a restructurable intermotherboard connection system, they do not lend themselves easily to rapid large scale rewiring of the system, and they do not allow a prediction of signal deterioration for any general connection. In an attempt to eliminate the problems which arise with the use of standard wiring procedures, a restructurable wiring system based on the use of a mass-produced, printed wiring cable is under study.

A basic printed wiring cable contains 100 long lines which are crossed (on the reverse side) at five locations by sets of 100 cross lines. At each crossing the kth long line is connected to the kth cross line by a plated through hole. This permits establishing of signal paths which enter the cable on a cross line travel down the cable and exit at a different crossing. The parts of the long line not in the signal path, as well as the cross lines in the signal path but not used are disconnected from the path by physically "punching out" the connection. This philosophy of disconnecting rather than making connections should lead to greater reliability.

The basis cables may be used to effect vertical connections between motherboards on different levels on the frame, or used as lead-ins to a horizontal cable tray. The cross lines of a basic cable may be connected, by use of removable pins, to any socket of a connector on the end-plate of a motherboard or connector on the horizontal cable tray.

In general, a particular basic cable is reusable only in situations where the number of changes is small and repairs can be made. The cost in basic cables in restructuring is reduced by mass production techniques. The known geometry of such a wiring system should permit prediction of deterioration of a given signal as a function of length of the path and number of connection points.

B. Switching Circuits

Two types of switching circuits are presently candidates for initial V circuitry. Both types will be incorporated in the initial V inventory to permit further evaluation. One type uses a diode-inverter logical stage followed by another inverter as the basic amplifier module and diode-inverter units in the combinatoric modules (Fig. 7). The other type uses a diode-emitter follower logical stage followed by an inverter as the amplifier module and diode-emitter follower units as combinatoric modules Fig. 8. Both types of amplifiers have a fan-in of 3 and fan-out of 10. The fan-in of the amplifiers can be increased by attaching combinatoric circuits.

The rise and fall times of all circuits are approximately 2 nanoseconds. The delay time (measured at 50 per cent values of signal swings) of the first type amplifier (diode—inverter) is 10 nanoseconds without load and 15 nanoseconds with a fan-in of 10 and driving maximum load. The delay time of the second type amplifier (diode-emitter follower) is 7 nanoseconds unloaded and 10 nanoseconds with fan-in of 10 and under full load.

The variable structure computer system permits addition of new modules to its hardware inventory at any time in history. Hence the design of new circuits for future inclusion in the hardware inventory will be carried on concurrently with evaluating the performance of the current circuits. Nonsaturating circuits and tunnel diode applications in particular will receive a great deal of attention.

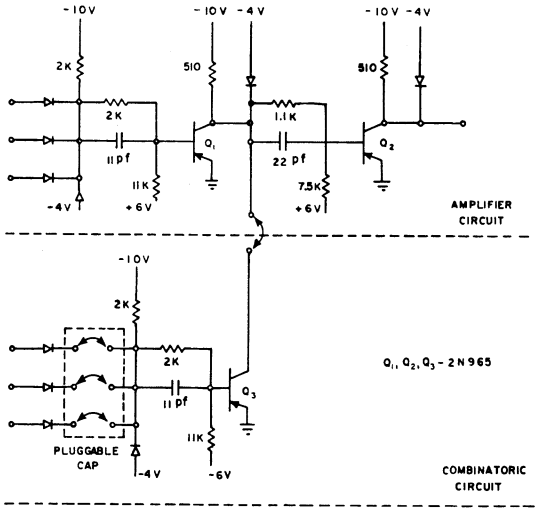

Fig. 7—Circuit diagrams of diode-inverter amplifier and combinatoric modules.

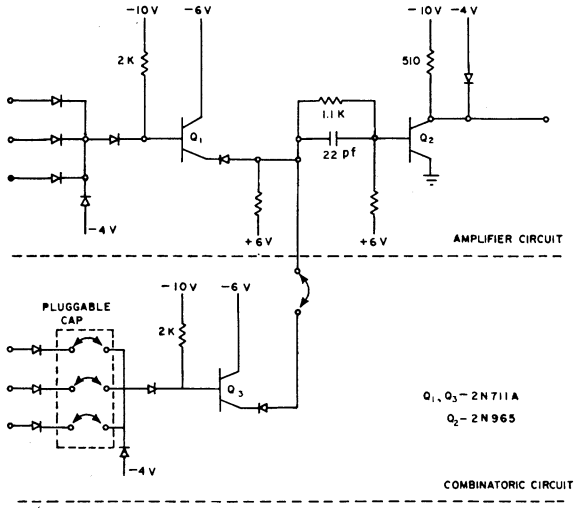

Fig. 8—Circuit diagrams of diode-emitter follower amplifier and combinatoric modules.

C. Restructuring of the System

Restructuring of the variable structure computer system may be accomplished by electronic means (enabling of sets of gates) or by physical changes in module locations and their interconnections. While electronic restructuring permits rapid switching between a small set of preconstructed configurations, it is the capability for physical reconstructuring which permits large-scale sharing of V hardware between different special purpose configurations. A major physical restructuring of V requires a considerable design and construction effort. Given a formulation of the changes which must be made as referred to the standard state, the following steps may have to be executed:

1) Rewiring of the caps of the combinatoric modules.

2) Generation of layouts for motherboard wiring harnesses. A program for automation of this step is now being written. Output of this program will be a tape to control an automatic x-y plotter which is equipped with a programmable intensity light pencil for drawing the artwork for exposing a predrilled and plated-through raw harness stock. The harness stock is mass produced and stored as part of the V hardware inventory.

3) Specification of the intermotherboard connections. An automatic system for effecting the connections (*e.g.*, automatic wiring machine for conventional interconnections, automatic disconnecting of lines in the printed cable system described above) is essential and will receive high priority.

4) Removal of old harnesses and mounting of the new harnesses on motherboards. Testing of these subsystems for proper wiring and circuit operation.

5) Assembly and testing of the intermotherboard and interframe wiring systems.

6) Installation of the new motherboards and wiring system in a frame and connection into the system.

7) Debugging of the whole system. Throughout steps 1) through 5) the variable structure computer system can remain operative (if the motherboard inventory is sufficiently large) either in its current problem oriented configuration or in the standard state. During step 6) the F part of the system remains operative.

The interconnection and restructuring procedures as described above permit conceptual realization of flexible structure changes. The validity of these concepts will be tested when the first significant amount of inventory has been delivered and nontrivial structure changes can be effected. At present, prototype modules and motherboards, as well as intermotherboard interconnection harnesses have been constructed. Larger scale construction is in process.

V. Conclusions

Presently a variable structure computer system is in design and early construction stages at UCLA. It may be predicted that the following forms will evolve:

1) Sequential computational procedures and the multiprogrammed handling of problem groups will exert a force in the direction of defining a group of general purpose computer configurations coupled to the extent that they time-share expensive substructures such as high-speed memory or high-speed arithmetic units.

2) Computational procedures with high inherent parallelism will tend to force configurations operating under global control.

Many of the fruits of creative programs such as that of the University of Illinois have already found their way into new generation computers. Since the new philosophy established at UCLA does not even have to wait for the completion of a total system to generate tests of its power, it should result in continued contributions to digital technology as well as problem solutions immediately.

BIBLIOGRAPHY

[1] G. Estrin, "Organization of computer systems: the fixed plus variable structure computer," *Proc. Western Joint Computer Conf.*, pp. 33–40; 1960.
[2] D. G. Cantor, G. Estrin, A. Fraenkel, and R. Turn, "A very high speed digital number sieve," *Math. Computation*, vol. 16, pp. 141–154; April, 1962.
[3] G. Estrin and C. R. Viswanathan, "Organization of a fixed-plus-variable structure computer for computation of eigenvalues and eigenevectors of real symmetric matrices," *J. ACM*, vol. 9, pp. 41–60; January, 1962.
[4] M. Aoki and G. Estrin, "The Fixed-Plus-Variable Computer System in Dynamic Programming Formulation of Control System Optimization Problems, Part I," Dept. of Engrg., University of California, Los Angeles, Rept. No. 60–66; May, 1961.
[5] B. Bussell and G. Estrin, "An Evaluation of the Effectiveness of Parallel Processing," *Proc. Pacific Computer Conf.*, pp. 201–220; May, 1963.
[6] D. G. Cantor, G. Estrin, and R. Turn, "Logarithmic and exponential function evaluations in a variable structure digital computer," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-11, pp. 155–164; April, 1962.
[7] R. Turn, "Assignment of Inventory of a Variable Structure Computer," Dept. of Engrg., University of California, Los Angeles, Rept. No. 63-5; January, 1963.
[8] M. Aoki, G. Estrin, and T. Tang, "Parallelism in computer organization—random number generation in the fixed-plus-variable structure computer," *Proc. Western Joint Computer Conf.*, pp. 157–172; May, 1961.
[9] D. E. Muller and W. S. Bartky, "A Theory of Asynchronous Circuits," Digital Computer Lab., University of Illinois, Urbana, Part I, Rept. No. 75, November, 1956; Part II, Rept. No. 78; March, 1957.
[10] D. Slotnick, *et al.*, "The SOLOMON Computer," *Proc. Eastern Joint Computer Conf.*, vol. 22, pp. 97–107; December, 1962.
[11] W. S. Dorn, "Generalizations of Horner's rule for polynomial evaluation," *IBM J. Res. & Dev.*, vol. 6, pp. 239–245; April, 1962.
[12] G. Estrin and R. Turn, "Automatic Assignment of Computations in a Variable Structure Computer," this issue, p. 753.
13] E. C. Russell, "Automatic Assignment of Computational Tasks in a Variable Structure Computer," Dept. of Engrg., University of California, Los Angeles, Rept. No. 63–45; October, 1963.

Automatic Assignment of Computations in a Variable Structure Computer System*

G. ESTRIN†, MEMBER, IEEE, AND R. TURN†, MEMBER, IEEE

Summary—**The problem of optimal assignment of subcomputations of a computational task to autonomous computing structures of a variable structure computing system is investigated. In particular, it is desired to determine which computing structures should be constructed from the hardware inventory of the variable structure computing system and which subcomputations should be executed on which computing structures, and in what sequence, so as to minimize the total cost of computation (cost of restructuring of the system and the cost of the actual execution time). A successive approximation assignment procedure is formulated. The procedure requires representation of the computational task by a directed graph and an estimate of the number of traversals of each computational loop, as well as the branching probabilities of each conditional branching operation. Computer programs for automatic execution of the assignment procedure have been written. A numerical experiment on a problem from the area of the x-ray diffraction analysis of crystal structures indicates that the procedure is computationally practical, and also demonstrates that execution of the problem in a variable structure system leads to a considerable gain over a system of three modern high-speed general purpose computers.**

I. INTRODUCTION

THE CONCEPT of a variable structure computer system and its characteristics in processing certain classes of computational tasks have been discussed in a number of papers [1]–[7]. These discussions propose a computer system which consists of a modern general-purpose computer F, an inventory of computing hardware V, and a supervisory control unit SC. The most important characteristic of this system is that the hardware in V may be restructured, both by electronic and mechanical means, into different problem-oriented computing configurations. Special purpose design techniques (*e.g.*, wired programs) and parallel processing capability are used to obtain reductions in processing time as compared to more sequential general purpose computers.

Programming of a variable structure computer system requires the usual implicit specification of the seqence of operations to be performed and proper location of associated data. Additional generality is introduced by the freedom to select a number of special purpose computing configurations from the currently available inventory of V and to define the interaction between these parallel processors sufficiently to specify the SC. Such a programming task is clearly enormous and automatic or semiautomatic programming procedures are essential for effective use of the variable structure system organization.

This paper is concerned with one part of the over-all task of programming a variable structure computer: optimal *a priori* assignment for processing the subcomputations of a gross computational task in a set of pre-

* Received September 19, 1963.
† Department of Engineering, University of California, Los Angeles, Calif.