COMPANY CONFIDENTIAL

STRETCH AUTOMATION OF DESIGN

Leonard S. Snyder

Now that you haveall heard about the STRETCH system, its whys and wherefores, the question in your mind is "How?". (In a number of minds, the answer is already "Never".) There is a problem, we will all agree. There are only so many people, especially experienced people, and only so much time. If we approached the project with the same methods we have used previously, there would be much doubt about our schedule.

This, then, is the purpose of the STRETCH Automation of Design program, to provide methods of designing and checking the STRETCH machines more quickly and accurately. We hope to do this by using programmers and coders early in the project whose work will save time later on.

The first objective of the program will, therefore, be to reduce the over-all time needed for the design of the STRETCH machine by reducing engineering time. The emphasis on time is not in the main to save money. Reducing the engineering time required for changes and design, permits us to make logical changes late in the program.

It also permits utilization of newly developed and approved components until late in the program. Thus, it will keep the machine design flexible until the last minute and produce a machine with the least built-in obsolesence.

Another valuable product of computer design procedures will be statistical analysis of machine circuits. We will be able to compile data on the use of present circuits to indicate the relative value of development on the various circuits. We will be able to evaluate any new components or logical circuits by an analysis of the circuitry before and after their insertion into a hypothetical machine.

One of the obvious reasons for using a computer in a machine design program is for machine simulation. We hope to be able to simulate the STRETCH machine on the detailed block diagram level. The program will assume that the circuits are black boxes which work as specified by the circuit design group. The computer will trace signals from circuit to circuit testing the logical connections and timing between circuits. This will permit us to test many different engineering versions of the machine easily and inexpensively. It will also check final design to prevent unnecessary changes after release.

The foregoing is a statement of what we on STRETCH would like to do. What are the possibilities? It has been stated that anything can be done by a computer that can be set down in a simple step by step procedure. The obvious possibilities for a machine design program are such activities as back panel wiring, compilation of Bills of Material, calculation of power requirements, a statistical analysis of circuits and the assignment of non-logical circuit elements such as line drivers. Some less obvious possibilities are arranging the circuits to some optimum configuration, pluggable unit layout, and rearrangement of logic upon the introduction of a new logical element. Machine simulation on the block diagram level is obviously possible and necessary but may be difficult.

The ultimate would be to have the engineers present to the computer, a Boolean algebra expression for a proposed computer. The computer would then develop factored expressions, insert necessary non-logical elements, compile power requirements, assign pluggable unit locations, develop back panel wiring and assemble a bill of material. This sounds fantastic and may be further in the future than STRETCH itself. It may even require a STRETCH machine to do this type of a job in a reasonable time. However, much is realizable now.

For our present purposes we can use this work as each step is completed. The Kingston Military Products people are already using computers to lay out the wiring for etched cards. The program for back panel wiring is a simple extension of this whose further requirement may only be more memory. The assignment of pluggable units or circuits geographically should be no harder than a commercial inventory problem which computes re-order times and levels of safe inventory. A program which is even more closely related to an inventory problem is that of compiling bills of material and input power requirements.

This brings us, in a backward manner, to the present work of our group. Since the first problem for machine designers is that of the logic of machines, we decided to get some experience in the manipulation of logical expressions.

For this purpose, we selected the 705 Card Reader control circuits. We have tabulated the 705 Card Reader circuits and formed Boolean algebra expressions for the circuits as they exist in the machine using the 704. Programs have been flow-charted for forming the alternational normal forms of these expressions and reducing them to the simplest form.

The method we are experimenting with is as follows:

Each circuit has been given a number. Cards are punched with the circuit type, the number of the circuit, and the numbers assigned to a maximum of four input circuits. If there are more than four inputs, imaginary circuits are created to keep the table words down to only four inputs. The circuits are then tabulated on the drum. A symbol for the logic, and the numbers given to the inputs of each circuit are placed at a drum address corresponding to the circuits own number. When it is desired to form the expression for any line, the address of its driving circuit is used as a starting point. We then can search through the table. Since each circuit has its input addresses associated with it, we can trace back in the machine logic until we come to what we call base-points. These base-points are such devices as triggers, single shots and inputs from cables. The expression will then use as factors these base-points.

This table will also be used in the near future to supply statistical information about the prevalence of "And" circuits with more than two inputs, number of cathode followers, etc.

The notation used within the Boolean algebra expression consists of a weight, a logic symbol and an address for each logical circuit. Figure 1 shows a typical circuit, devoid of non-logical elements, its normal Boolean algebra expression and the form used in this experiment. The symbol "O" indicates an Or circuit, the symbol "A" indicates an And circuit and the symbol "I" indicates an Inverter. This is true both of the circuit block diagram and the machine form of the expression. In addition, the symbol "H" indicates an input to the circuitry in the machine form of the expression. The address is the four digits to the right of the letter and the weight is printed to the left of the letter.

The address indicates the number given to the circuit. This can be a physical location, but is now an indication of the circuit's place in the Systems Diagrams. The logic symbols are presently limited to Ands, Ors, Denials, Halts and Non-logical elements. Other symbols are available if we desire to expand the number of connectives. The weight, which is assigned each circuit, is an indirect replacement for the parenthesis in logical equations. It indicates that the connective or logical symbol associated with it, applies to all the expression to either side of the connective until a connective of equal or higher weight is encountered.

The next step of our project will be a reduction to the alternational normal form. This form is an expression consisting of "And" connectives feeding a single "Or" connective. This reduction is to be done by first removing all denials except the denial of base-points. We then expand "Ors" feeding "Ands" until only "Ands" feed to a common "Or".

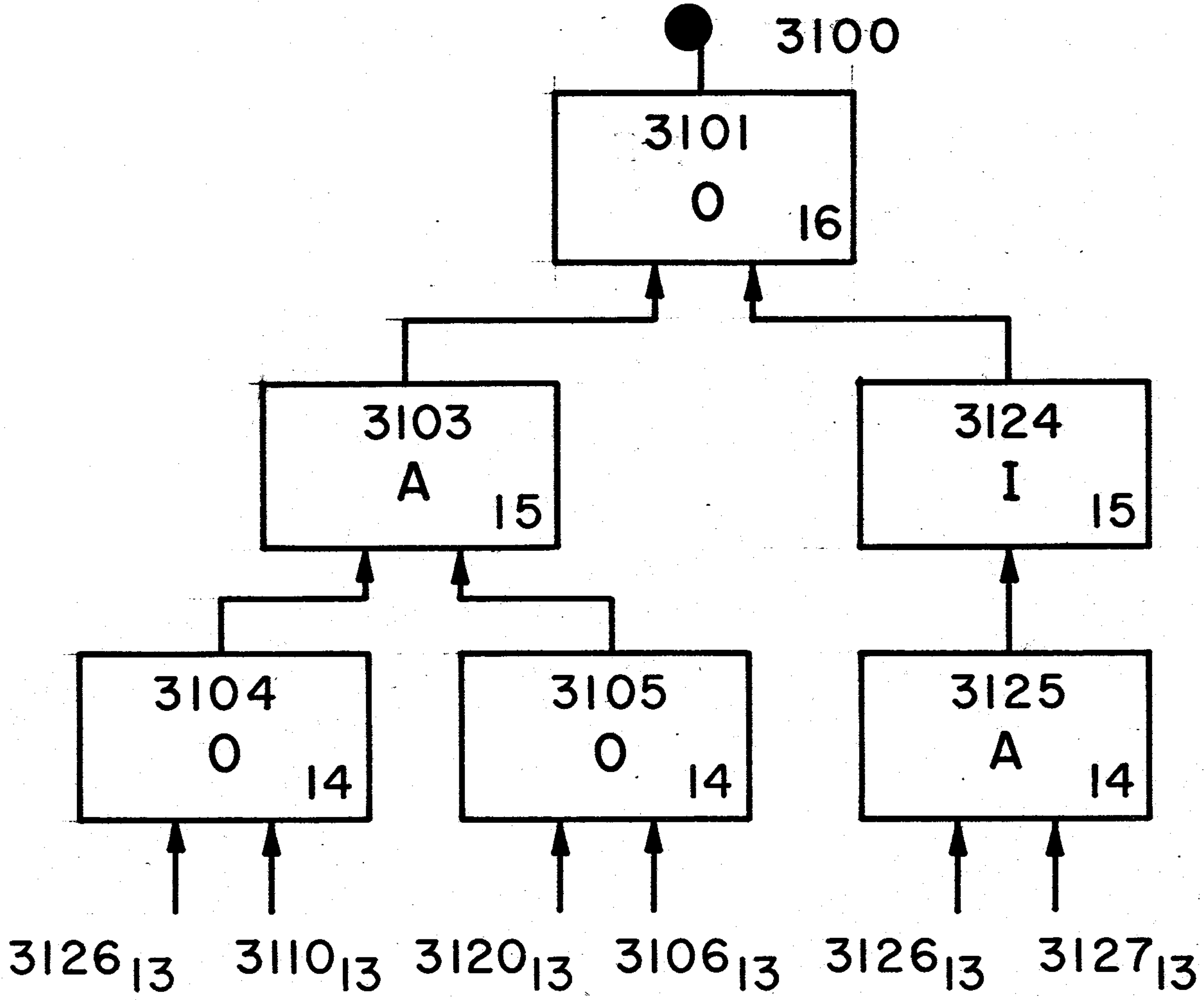
The method of simplification to the alternational normal form we are using was derived from a paper by W. V. Quine in the American Mathematical Monthly, November 1955, entitled "A Way to Simplify Truth Functions".

Professor Quine has proven that his method will result in the simplest alternational normal form. His method does not required as much memory as other methods which have been proposed for mechanization. This, I feel, is sufficient justification for its use. When an expression has been simplified, it will be tested for inconsistencies and validity. We will then attempt to factor these expressions and compare the result with the machine as designed. This will give us a measure of the success of our experiment. Then we will have some idea of how close we can come at the present to the ideal.

Leonard S. Snyder

LSS/mms

FIGURE NO.



PRESENT FORM

$$(3126+3110)\cdot(3120+3106) \quad + \quad \overline{(3126)(3127)}$$

MACHINE FORM

13 H 3126	14 O 3104	13 H 3110	15 A 3103	13 H 3106	14 O 3105
13 H 3120	16 O 3101	13 H 3126	14 A 3125	13 H 3127	15 I 3124