UNCLASSIFIED

OAK RIDGE NATIONAL LABORATORY

Operated By

UNION CARBIDE NUCLEAR COMPANY

POST OFFICE BOX P
OAK RIDGE, TENNESSEE" External Transmittal
Authorized "**ORNL**
CENTRAL FILES NUMBER

56-3-61

DATE: March 15, 1956
SUBJECT: A New Approach to Computer Design

TO: [Redacted]

FROM: A. C. Downing, Jr.

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, express or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission to the extent that such employee or contractor prepares, handles or distributes, or provides access to, any information pursuant to his employment or contract with the Commission.

Photostat Price \$ 1.80Microfilm Price \$ 1.80Available from the
Office of Technical Services
Department of Commerce
Washington 25, D. C.**UNCLASSIFIED**

578-001

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

A NEW APPROACH TO COMPUTER DESIGN

A. C. Downing, Jr.
Mathematics Panel

In a general-purpose digital computer, the flow of information is in two directions -- from the memory to the arithmetic unit and from the arithmetic unit back to the memory. Throughout this paper, the term arithmetic unit is understood to include the control unit of the main computer. The proposals for many of the fastest contemporary machines point out that very often this flow of information does not take place smoothly because the speed of the arithmetic unit and the access time of the memory are not well balanced. The remedy advanced in the latest machines is to include a few hundred words of faster, very expensive, random access storage which is to be used in exactly the same fashion as the larger core or electrostatic memory. The purpose of these remarks is to propose another solution.

The problem is to improve the handling of the flow of information between the memory and arithmetic units. This suggests the possibility that some kind of special-purpose computer could be inserted between the arithmetic unit and the memory. Briefly, the function of this special-purpose computer (henceforth abbreviated SPC) would be to match the speeds of the arithmetic and the storage units of the main computer by anticipating

the needs of the arithmetic unit. This it could do by examining the sequence of instructions to be performed by the arithmetic unit and storing these instructions, and the words referred to in these instructions, in its own storage registers. By anticipating the needs of the arithmetic unit, the SPC can consult the memory independent of the action taking place in the arithmetic unit. Hence, during long operations, such as multiplication and division, valuable memory access times would not be wasted. Moreover, the memory would only have to be consulted once for words used sufficiently often.

It should be emphasized that the machine user would not have explicit control over the operation of the SPC. Programming would be exactly as in existing general-purpose computers having only one type of random access storage.

This suggests that a vital part of the SPC would be a directory which will enable it to determine which of its registers contains the word belonging in a specified memory location. Such a device could undoubtedly be designed, but, since the SPC serves a very special purpose, it seems desirable to take advantage of this fact.

One possible scheme for the logical design of the SPC would be to have the storage registers of the SPC long enough to accommodate a full word from the memory plus the address of that word and at least one additional bit to indicate whether the word originated in the memory or in the arithmetic unit. In the latter case, the word should eventually be stored in the main memory. The contents of such a register constitute a "sentence".

The sentence registers of the SPC would be arranged in two sequences* -- one sequence of registers $R_1, R_2, \dots, R_n$ for information moving toward the arithmetic unit and a second sequence of registers $T_1, T_2, \dots, T_m$ to store information coming from the arithmetic unit. This would include words already used by the arithmetic unit.** (See Figure 1.)

OPERATIONS PERFORMED BY THE SPC

The SPC will use the two sequences of registers as follows: Let the sentence in R_1 contain an instruction word. If the computer is a single-address computer with two commands per instruction word, as in the Oracle, then R_2 and R_3 would contain the words referred to in the left and the right commands, respectively, of this instruction word, together with their addresses. Normally R_4 would contain the next instruction word with its address, and so forth. As the main arithmetic unit accepts the instruction from R_1 , the sentence in R_2 is automatically transferred to R_1 , and in general the sentence in R_i is gated into R_{i-1} . As a word leaves R_1 to enter the arithmetic unit, it is also copied into the distribution register or D-register of the SPC. While the arithmetic unit is computing, the SPC scans each of the R-registers looking for a sentence containing the same address as the one in the D-register. If none is found,

*The author recognizes that some of his colleagues will be tempted to refer to the contents of such sequences as "paragraphs".

**Another possible arrangement is to factor the R-sequence into two separate sequences, one for instructions and the other for numbers.

the contents of D are inserted into T_1 at the end of the operation. If, however, the address in some R_j matches the address in D , then the sentence in D need not be transferred anywhere if the sentence in R_j is complete,* except in the case when the executed command modifies the sentence in D , as, for example, in a substitution or store order. In this case, the modified sentence in D is copied into the first R_j -register with the same address. At the same time, the indicator in R_j is set so that this new word will eventually be stored in the main memory. If the contents of D are to be inserted into T_1 , then T_1 must first be vacated. This means that the sentences in each T_i must be moved into T_{i+1} . The contents of T_m are either forgotten or stored depending upon the state of the storage indicator in T_m . It can happen that some T_j may be vacant, in which case it is possible to postpone storing the contents of T_m , since the contents of T_{j+1} need not be disturbed. The reason why some T_j may be vacant will be clear when the process for securing the information in the R-registers is explained. Before doing this, however, it is convenient to be more specific about the sentence structure in the two "paragraphs" in order that the operation of such a computer might be better examined.

ADDITIONAL DETAILS

If the word length of the main memory is λ binary places, if the main memory has 2^α addressible locations and if $2^\beta \leq n$, where n

* It may be that design considerations will make it desirable that, when the same sentence appears in more than one R-register, only the register with the lowest index contain a complete sentence. This is discussed later.

is the number of R-registers, then each of the T-registers will have $\lambda + \alpha + 2$ bits, and each R-register will have $\lambda + \alpha + \beta + 4$ bits. The β bits will serve as the index specifying some other R-register. The additional 2 or 4 bits serve as control indicators or instructions for the SPC. These binary indicators are denoted by b_1 , b_2 , b_3 , and b_4 . Superscripts may be used to specify to which register they belong. If a register is not vacant, then $b_1 = 1$. If the word in that register originated in the arithmetic unit, as the result of a store or an extract order, then $b_2 = 1$. Indicators b_3 and b_4 exist only in the R-registers since this sequence carries information to the arithmetic unit. The indicator b_3 records whether or not the word is to be interpreted as an instruction, and b_4 indicates whether the word should be ignored entirely by the arithmetic unit. The latter indicator is cleared to zero in every R-register whenever the arithmetic unit signals that the decision made in connection with a conditional transfer order was not the one anticipated by the SPC.

The sentence in any T-register is a quadruple $(w_\lambda, a_\alpha, b_1, b_2)$ comprised of a word, w_λ , from location a_α in the main memory, and two indicators. The sequence of T-registers contains words which have already been used or processed by the arithmetic unit.

The sentence in any register R_j is a septuple $(w_\lambda, a_\alpha, a_\beta, b_1, b_2, b_3, b_4)$. The integer a_β , $0 \leq a_\beta < n$, serves to remind the SPC that the register R_{j+a_β} is the first register following R_j which must contain the same sentence as R_j .

The contents of R_1 can be gated into the arithmetic unit directly. Whenever this happens, the appropriate indicator in R_1 will show that R_1 is vacant. Moreover, each R_i ($i = 2, \dots, n$) will automatically be gated into R_{i-1} whenever R_{i-1} is vacant, and, as this occurs the indicator $b_1^{(i)}$ is set to 0 (i.e., "vacant") and simultaneously $b_1^{(i-1)}$ is set equal to 1.

The control counter for the entire computer is in the SPC and from it the SPC obtains the address of the next instruction it must process. Before fetching any word from the memory, the SPC scans the address portions of the R- and T-registers to see if the desired word is already available. The scanning of the R-registers starts with R_n , proceeding to R_{n-1} , etc. As each non-vacant register is interrogated, a counter which is initially zero is increased by one. The contents of this counter are substituted into the a_β portion of the first register, R_j , which has the desired address. No information is copied from R_j . Thus, the septuple inserted into R_n will not contain the word w_λ , nor the indication of whether it originated in the arithmetic unit. These will be filled in later as the sentence, now in R_j , leaves the D-register. Note that with the index a_β known, the SPC does not have to scan the registers to determine where to put the sentence when it leaves the D-register. If $a_\beta = 0$, the sentence in D goes into T_1 ; if $a_\beta = i > 0$, then part of the contents of D, namely w_λ and b_2 , are copied into R_1 . (See Figure 2.)

If the desired address is not located in the R-sequence, then the T-sequence is scanned. The direction of scanning does not matter here

since no address can occur in more than one T-register. For if it did, then it would have previously existed in an R-register and a T-register, which is impossible. If the address is found in some T-register, w_λ and b_2 of that sentence are copied into the composition register or C-register, where the entire sentence is composed before it is copied into R_n . Meanwhile, the particular T-register is marked vacant, since all the information it contained is now located in R_n . Finally, if the word must be obtained from the main memory, this is done.

If the sentence placed in R_n is identical with the sentence in another R-register and if the main word in this sentence is to be used as an instruction, then the SPC cannot conveniently compose the sentences to follow this one since it does not know the addresses referred to in this instruction. However, when the same instruction sentence occurs twice in the R-sequence, then the list of instructions to be performed comprises a loop, and if the loop is not too short and the SPC is not too slow, then the loop can be copied as the individual words reach the D-register.

This brings up the problem of conditional transfers of control. As the SPC is interpreting the list of instructions to be performed, it must have some way of deciding whether or not the condition will usually be satisfied. Such decisions will be made by the programmer and be indicated by his choice of transfer order. This means that the order code for the computer will have twice as many possible transfer of control commands as a corresponding computer without a built-in SPC.

Finally, it should be observed that any sentence in the R-sequence cannot occur simultaneously in the T-sequence and, as previously mentioned, every sentence in the T-sequence is distinct. Moreover, the information which must be extracted from a T-register by the composition register is only w_λ and b_2 . Similarly, the information gated from the distribution register into a specified R_j is w_λ and b_2 ; and the partial sentence set up in some $R_j \neq R_n$ from the C-register is only the index a_β .

These facts tend to simplify the circuitry for the SPC, which would nevertheless be expensive. The total number of registers which would be required to substantially reduce the number of memory accesses appears to be small, however, compared with the several hundred all-electronic storage registers presently considered as a solution to the problem of how to handle the flow of information between arithmetic unit and primary memory without compromising the inherent speed of the arithmetic unit.

FIGURE 1.

FIGURE 2.