September 30, 1958

Mr. H. F. Warner
Purchasing Agent
Central Purchasing Department
Westinghouse Electric Corporation
Bettis Atomic Power Division
P. O. Box 1526
Pittsburgh 30, Pennsylvania

Dear Mr. Warner:

In order to be of service to the Bettis Atomic Power Division in connection with the Naval Reactor Program, we have endeavored to keep you advised of our planning and progress in the development of computing equipment. To this end, on July 22, 1958, we proposed a solid-state IBM 709-TX system for your consideration to meet the interim requirements of the Program. We shall continue to assist you in your evaluations of that proposal.

On April 25, 1958, Dr. C. R. DeCarlo submitted to you a summary of our activities and plans as of that date concerning the specifications, delivery and price of certain large scale computing equipment.

I am informed that you had the opportunity of reviewing our progress in Poughkeepsie the first part of September, as well as the status of the high-speed computation program with Messrs. Dunwell and Sweeney. We are pleased to inform you that this program is proceeding as planned and we intend to deliver a prototype system to Los Alamos in May 1960.

Technical specifications for the system are in preliminary form, and we expect release of manuals for the purposes of programming and technical planning during the first quarter of 1959.

It is our sincere belief, based on our knowledge of the reactor design problem that equipment with the design specifications of our high-speed computation programs can best satisfy your ultimate requirements. We are pleased to present to you our proposal for such a system as outlined in Appendix A.

Attached to this letter as an Appendix B is a description of the equipment which IBM proposes to furnish. We have stressed the advantages to you, and simultaneously described the operations and characteristics of a system to best fit the requirements of the Naval Reactor Program.

Our present estimates indicate that the system described in the attachment would have a selling price of approximately $11,880,000. We are prepared to work with you at your earliest convenience in establishing the final configuration of the system to best meet the requirements of your Program. Those components of the final system which are commercially announced IBM products will be available to you on either a purchase or a lease basis at IBM's established prices and contract terms if announced prior to delivery. We expect that by February 1, 1959 we will be in a position to quote to you a firm selling price for the entire system, and in addition, negotiate with you an appropriate type of contract for those components which are not IBM commercially announced products. Such negotiated contract could, should you so desire, provide for progress payments prior to delivery and installment payments of any unpaid balance existing at the time of delivery.

Under appropriate contractual arrangements we would plan to manufacture and deliver the system to you in the first quarter of 1961.

This proposal, if not accepted by you or extended by IBM prior thereto will expire on March 31, 1959.

We look forward to being informed of your acceptance of this arrangement and will be pleased to answer any questions which will enable you to reach a favorable decision.

Very truly yours,

INTERNATIONAL BUSINESS MACHINES CORP.

By_____

J. E. Whitfield
DP Controller

JEW:hjw
Attachment

September 30, 1958

APPENDIX A

WESTINGHOUSE - BETTIS ATOMIC POWER DIVISION
STRETCH COMPUTER SYSTEM

One (1)	Central Processing Unit - including the indexing arithmetic unit, the computer control circuits, the arithmetic unit, the computer bus system, and the basic eight (8) channel exchange.
One (1)	Console
One (1)	Inquiry Unit
One (1)	Console and Inquiry Adapter
Six (6)	Magnetic Core Memory Units - each containing 16,384 words of 64 bits.
One (1)	High Speed Exchange
One (1)	Disk Control Unit
One (1)	High Speed Disk Unit - having a capacity of 1,048,576 words
One (1)	7500 Card Reader
One (1)	Card Reader Adapter
One (1)	7550 Card Punch
One (1)	Card Punch Adapter
One (1)	7400 Printer
One (1)	Printer Adapter
Two (2)	Tape Control Units
Eight (8)	729-IV Tape Units

THE APPLICATIONS OF STRETCH TO REACTOR DESIGN CALCULATIONS

During the deliberations of the joint Los Alamos-IBM Mathematical Planning Group, reactor design calculations were always one of the principal types considered. In particular, both the diffusion theory difference equations and the SNG transport theory were considered in the studies, including the timing simulation studies of STRETCH done on the 704.

However, it would be false to say that STRETCH was designed for reactor calculations. The design goal was always the best general purpose computer consistent with the state of the technical art, not one designed for one specific problem. In this way, we hope to provide flexibility for the computer to adapt to any new computing procedures which may be developed in this rapidly changing field.

Reactor design calculations, even as presently done, strain computing machines in several distinct ways. The inner loops tend to be quite simple arithmetic but must be done a tremendous number of times. On the other hand the complexities of handling boundary conditions and other special cases tax the logical power of the machine. The mass of data required in a large multi-group problem creates a data processing job equalled in volume and rate by few other computer applications.

The STRETCH system represents a significant improvement in all these critical areas. The full power of its floating point arithmetic speed is brought to bear on the tight inner loops, while its logical commands are ideal for handling the special cases and editing problems. The exchange philosophy and the large-capacity high-speed RAMAC provide high data flow rates at the same time the arithmetic calculation is proceeding at top speed.

The increased speed and capacity of STRETCH can bring about a number of significant changes in reactor problems. The simplest extension will be to do existing calculations with finer spacial resolution or more neutron velocity groups. The next step would be to raise the dimensionality of the problems -2 dimensions where 1 dimension is common now and 3 dimensions instead of 2 in other problems. In addition it also means that scientists can study in detail the variations of reactor configurations due to depletion and poisoning with age.

Studies of the effects of rapid power cycling during the lifetime of the reactor could be done as a regular procedure in design. Studies of explosion safety of the reactor using the time-dependent transport equations may also become common place. The use of the more accurate but more difficult transport theory instead of diffusion theory in general will certainly be encouraged by STRETCH. Such developments will certainly become more important in future power-breeder designs.

Other non-nuclear problems which are still important to reactor designers such as thermal stress and heat transfer calculations could be incorporated directly into the basic reactor codes. Wigner energy calculations in solid moderators is another possible candidate.

The real pay-off of computing in reactor design occurs when it renders an experimental critical assembly unnecessary. The time and money which such a test requires is certainly tremendous. This is coupled with the fact that an experimental assembly cannot give as much detailed information as a good calculation. Of course, we cannot promise that STRETCH will render all experimental assemblies unnecessary, but as a new tool in the hands of your scientists and engineers it will certainly have an impact on the reactor field of such a magnitude that we can now only guess as to its nature.

Appendix B covers in detail the features of the prototype system which we are including in the system which we are proposing for application to your program. These features are all included in the STRETCH system which will be delivered to Los Alamos.

September 30, 1958

APPENDIX B

FEATURES OF THE STRETCH COMPUTER SYSTEM

Introduction

The purpose of the high-speed computing system, known as STRETCH, is to provide an increase in overall performance on very large technical computing problems of the order of 100 times the performance of present-day systems. The STRETCH computer provides a tool to solve important problems for which the capacity and speed of earlier computers is too low to permit completion of these problems in a reasonable length of time. The power of STRETCH also permits it to solve lesser problems more efficiently.

The STRETCH computer employs solid-state components throughout. The logical portion of the main computer and its working registers use high-speed transistors and circuits developed especially for this system. The use of relays and other electromechanical contacts is kept to a minimum, so as to maintain a degree of reliability commensurate with the use of solid-state components.

New magnetic core memories attain memory cycles which are several times shorter than those on previously available core memories. Special power transistors have been designed to drive the memories.

As will be described below the increased performance factor of 100 in the STRETCH system does not result from improvements in component technology alone. The new technology is teamed up with a high degree of overlapped operation, new input-output facilities, a more powerful instruction set, and other new features.

Several memory units can be cycled concurrently. Results produced can be stored in memory and one or more new instructions and operands can be fetched from memory, while the arithmetic unit is busy executing an instruction with data already in its registers. Automatic interlocks and safeguards are provided which adjust the flow of information and assure that the program is executed correctly. The programmer is in no way concerned with these matters. He writes the program as if it were executed sequentially, one instruction at a time.

Because of the overlapped operation of many parts of the system, it is quite difficult to predict the exact speed at which a given program will be executed. The duration of each operation depends on the data, on the availability of various units, and on the instructions which preceded it. While a time can be given for each operation alone, in terms of a formula or as an average, the overall time is not the sum of these times. Representative time estimates for problems have been obtained by simulating the many detailed time relationships in terms of that program.

Another factor in higher performance is in the handling of input-output and external devices. A large, high-speed magnetic disk unit with a one million word capacity is available to store large data arrays. Extensive buffering and multiplexing facilities permit simultaneous running of many units of lower speed. The input-output system is flexible enough so that almost any kind of device furnishing or accepting digital information can be connected to the computer.

The instruction set which has been developed for this system exhibits new and powerful features. As a result, fewer instructions are usually required to write a given program. Moreover, most of the instructions used in the inner loops of floating point arithmetic problems will be only half a word, or 32 bits, long. Fewer and shorter instructions mean less storage space for programs, fewer accesses to memory, and fewer instructions executed. All of this contributes to high performance.

Much of the emphasis in the instruction set is on efficient floating point, indexing, and branching instructions, which form the heart of a great many programs in the technical field. It is equally important, though, to have available good instructions for housekeeping functions, for editing input-output data, and for processing data other than floating point numbers. As an example, a stored program computer will normally spend considerable time in preparing its own programs. To simplify such tasks, a complete set of instructions is provided for processing fields and records of varying length in any desired code.

The use of solid-state components makes the system inherently more reliable. Automatic error detection, together with means of locating faults and other maintenance aids, serve to reduce the time required to identify and repair those machine faults which do arise. Automatic error correction is provided in areas where this is expected to improve performance materially.

A final contribution to performance is made by features to improve operating techniques. Routine operating functions are, as much as possible, placed under direct control of the stored program. Input-output units require a minimum of manual set-up. All rearrangement and control is done by programming. Multiple input-output channels, buffering, and multiprogramming facilities make it possible to do independent input-output data conversion efficiently on units attached to the computer without significantly increasing the computing time on the major job. Physical switching of units or manual transfer of tape reels is minimized.

A new approach is employed in the operating console. The keys, switches, lights, digital display, and associated typewriter are all subject to program control. These facilities are provided to permit close communication between man and machine when human supervision is desired. With two or more programs ready for operation, the machine need not wait idly while the user thinks. All console facilities are designed to take advantage of the intervention techniques made possible by multiprogramming. The use of programmed definition of console functions also permits protective and logging functions which would be quite uneconomical in hardware.

SYSTEM ORGANIZATION

Memories

The computing system uses magnetic core memories with very short access times. A memory word consists of 64 information bits and 8 non-addressable redundancy bits. Memory and bus action is checked, and any single bit errors are automatically detected and corrected without slowing the speed of operation. Double bit errors are detected. The address space in instructions provides for addressing directly any of the 218 (262,144) word locations on all operations.

Main Memory

The main memory of a system is a magnetic core memory with a read-write cycle time of the order of two microseconds. Each unit of memory consists of 16,384 (214) words.

Each memory unit operates independently. Several memory references may be in process at the same time. In order that programs may take advantage of this, successive addresses are distributed among different units.

Index Word Memory

A separate fast magnetic core memory with addresses 16 to 31, is used for index words. This memory has a capacity of 16 words and is directly associated with the computer.

Special Registers

Many of the special registers of the machine are directly addressable and are assigned addresses 0 to 15.

Input and Output

Input to the system passes from the input devices to memory through the Exchange. The Exchange assembles successive 64-bit words from the flow of input information and stores the assembled words in successive memory locations without tying up the Central Processing Unit. The CPU specifies the locations and number of input words to be read, and the Exchange completes the operation.

The same Exchange operates similarly for output, fetching successive memory words and disassembling them for the output devices independently of the CPU. External storage devices are operated via the Exchange as if they were input and output.

The Exchange has the capability of operating eight independent input-output units. Each of the Exchange channels can accept information at a rate of up to 500,000 bits per second.

A wide variety of input-output units can be operated by the Exchange. These include punched card readers and punches, medium and high-speed printers, magnetic tapes, operator's consoles, and keyboard inquiry stations. For some kinds of units, several can be attached to a single exchange channel. When this is done, of course, only one of these units can be operated at a time.

The high-speed Exchange, like the basic Exchange, operates directly with memory, executing instructions as specified by the computer.

The very high speed magnetic disk file is capable of reading or writing a full 64-bit word each 8 microseconds. The disk unit has a capacity of 1,048,576 words.

The Central Processing Unit

The Central Processing Unit performs arithmetic and logical operations upon operands taken from memory. Operations are specified one at a time by instructions, which are also taken from memory. Each instruction fundamentally specifies an operation and an operand. The operand specification is made up of an address and an index word address. Part of the index word contents are added to the address in the instruction to get an effective address. The effective address actually designates the operand used. The additions needed to derive the effective address and to modify index words is performed in an independent index arithmetic unit.

The Index Arithmetic Unit

The index arithmetic unit consists of registers for holding instructions to be modified and the index words that are used in the modification. Operations on index words consist of loading, storing, adding, and comparing; the index arithmetic unit includes gates for selecting the necessary fields in index and instruction words and an algebraic adder. The index words themselves are stored in locations 16-31, index core memory.

Instruction Controls

An instruction may be one word or one half word in length. Full and half length instructions can be intermixed.

Instructions are taken in succession under control of an instruction counter. Alteration of the succession of instructions is possible by branching operations, which can be controlled by a wide variety of conditions. Automatic interruption of the normal sequence can also be caused by many conditions. The conditions for interruption and controlling branching are represented by bits in an indicator register. The interruption system also includes a mask register for controlling interruption and a base address core register for selecting suitable alternate programs. When needed, the address of the input or output unit causing an interruption can be read from a unit address register which can only be set up by the Exchange.

The interpretation and execution of instructions is also controlled by an address monitoring system.

Instruction Look Ahead

Since the arithmetic unit operates very rapidly, it needs to receive modified instructions at high speed. This modification is independently performed in the indexing arithmetic unit. For best speed, index arithmetic for an instruction must be performed while the arithmetic and logical unit is executing a previous instruction.

The device which coordinates and controls instruction processing is known as the instruction look ahead.

In spite of the internal complexities, fetching, modification, and execution of instructions do not place any constraints on the programmer. The only external apparent difference in system operation due to the look ahead is higher speed.

Arithmetic Unit

The arithmetic unit consists of an apparatus for performing floating point arithmetic upon a data word in parallel and a closely associated mechanism for performing arithmetic and logical functions upon arbitrary fields of bits. In the latter case, the bits may be operated upon as individual entities or as numbers encoded in binary.

A special location is provided to serve as a rapid-access source of binary zeros. This is one word long, and always delivers zero bits. This special location has address 0. Storage of data in this location is ineffective, so that it serves as a convenient and rapid access data sink. Address 0 can thus be used whenever an operation calling for an address is needed.

Classes of Operations

The operations available may be divided into these broad categories:

Floating point arithmetic operations
Integer arithmetic operations
Connective operations
Index arithmetic operations
Branching operations
Transmission operations
Input-output operations
Miscellaneous operations

Within these categories the goal has been to provide a more generalized set of instructions.

Summary of Basic Arithmetic Operations

The arithmetic instruction set includes the elementary operations LOAD, ADD, STORE, MULTIPLY, and DIVIDE. Modifier bits are available to change the operand sign. Thus the operations 'Subtract' or Add Absolute' are obtained by use of sign modifiers to the ADD instruction. These same modifiers permit changing the sign of a number to be loaded, stored, multiplied or divided.

A convenient feature of the Multiply operation is that one of the factors is taken from the accumulator rather than a separate register, and it may be the result of previous computation. Similarly, Divide places the quotient in the accumulator, thus making it available directly for further arithmetic steps.

Extensions of the basic set of arithmetic operations permit adding and counting in memory, rounding, cumulative multiplication, comparison and further variations of the standard ADD operation.

Most arithmetic operations are available in the floating point mode as well as the fixed point or integer mode. The floating point set includes additional instructions to handle portions of a floating point number and double length numbers with ease. A floating point square root operation is provided.

Floating Point Arithmetic

Floating point arithmetic is done in binary, using the full 64-bit memory word. The emphasis in the design of the floating point arithmetic is on high speed computation of large mathematical problems. The binary radix makes available techniques which greatly speed up multiplication and division.

The 48-bit mantissa (equivalent to about 15 decimals), coupled with the storage efficiency of the binary radix, makes it possible to compute in the single precision mode for a large number of problems which have to be done in multiple precision on earlier computers. The floating point instruction set, however, contains provision for programming multiple precision work in a direct manner. The drop in speed, when going to multiple precision, is much less than with earlier machines. Thus, multiple precision is required less often, and when required it is simpler than before.

The floating point word includes a signed 11-bit exponent and a signed 48-bit mantissa. The exponent is a true signed binary integer and the mantissa is a true signed binary fraction.

Floating point arithmetic can be performed in either a normalized or an unnormalized mode.

Integer Arithmetic

The class of integer arithmetic encompasses all data arithmetic on other than the specialized floating point numbers. The emphasis here is on versatility and economy of storage. Individual numbers, or fields, may be of any length. Fields of different lengths may be assigned to adjacent locations in memory. Each field may be addressed directly by specifying its position and length in the instruction; the computer takes care of selecting the memory words required and altering only the desired information. Since the field length is explicitly stated in each instruction, rather than being implied by the data or by previous length-setting instructions, there is no restriction on the coding of variable field length data.

A significant feature of the integer DIVIDE operation is that it will produce meaningful results regardless of the magnitude of the dividend or the divisor (provided these fall within the bounds of numbers generally acceptable to the arithmetic unit). The only and obvious exception is a zero divisor. This greater freedom eliminates much of the scaling previously required before a Divide instruction could be accepted.

Alphabetic and other non-numeric fields of various lengths may be handled by integer arithmetic operations as if they were unsigned binary numbers, regardless of the character code or the number of bits used for each character.

The integer instruction set is also very similar to the floating point set; most of the operations in both sets have the same names and analogous meanings.

Connectives

Instructions which logically combine bits by And, Or, and Exclusive Or have been included in earlier computers. These and many other non-arithmetic data handling operations are here replaced in a simple and orderly fashion by connective operations which provide many logical facilities not previously available.

Each connective operation specifies a memory field of any length. Each bit in the memory field is logically combined with a corresponding bit in the accumulator.

There are sixteen possible ways to match bit for bit, or connect, two bits. Each of these connectives can be specified.

Index Arithmetic

Every instruction may have its address part modified by adding a number in a specified index register before using the address. Normally both the instruction and the index register remain unchanged. To alter the index registers is the function of the index arithmetic operations.

The set includes operations for loading, storing, incrementing, and comparing index values. The index value is a signed number and additions are algebraic. Provision is made for indirect addressing.

Branching

The branching operations either conditionally or unconditionally alter the instruction counter so as to change the course of a program. The number of operations is not large, but modifiers are available to provide a great deal of flexibility.

All machine state indicators such as sign, overflow, error, and input-output conditions, are collected in one 64-bit indicator register. The BRANCH ON INDICATOR instruction may specify any one of these 64 indicators as the condition to be tested. A modifier specifies whether branching is to occur when the indicator is on or off. Another modifier may cause the tested indicator to be reset to zero.

A second operation, BRANCH ON BIT, permits testing a single bit anywhere in memory with one instruction. The tested bit may also be modified. This instruction places a virtually unlimited number of indicators under the direct control of the program.

Transmission

The operation TRANSMIT provides the facilities to move a block of data from one set of addresses to another. One use of this operation is to preserve the contents of addressable registers, including index registers, in memory when it is necessary to bring in another program, and later to reload those registers to restore them to their earlier state before restarting the interrupted program.

Input-Output

There are basically two operations for controlling input-output and external storage units: READ and WRITE. Each instruction specifies the unit desired and a memory area for the data to be read or written.

The memory area is specified by giving the address of a Control Word which contains the first data address in memory and a count of the number of words to be transferred. The Control Word also contains an address which can specify the address of another Control Word. Control words can thus be chained together to define memory areas which are not adjacent.

All instructions for operating external units are issued by the computer program but are executed independently of the program. A number of data transfers can thus take place simultaneously, all sharing access to memory. Signalling functions inform the program when each external process is completed.

Program Interruption

A program interruption system is provided for two quite distinct purposes. The first of these is to provide a means by which a computer can respond rapidly to extra-program circumstances which occur at arbitrary times, performing useful work while waiting for such circumstances. These circumstances will most often be signals from the Exchange that some interrogation has been received or that an input-output operation is complete. For efficiency in real-time operation, the computer must respond to these forthwith. This demands a system by which such signals cause a transfer of control to a suitable special program.

The second purpose is to permit the computer to make rapid and facile selection of alternate instructions when program-activated indicators signal that special circumstances have occurred. For example, it is clearly desirable to have such a system for arithmetic overflow since the alternatives are tedious and wasteful programmed testing or a costly machine stop when the condition arises. As another example, it is desirable to have a special routine seize control and take corrective steps whenever the regular program attempts a division by zero.

These two purposes, response to asynchronously occurring external signals and monitoring of exceptional conditions generated by the program itself, are distinct, and it would be conceivable to have systems for handling each independently. However, a single system serves both purposes equally well, and provision of a single uniform system permits more powerful operating techniques.

Means are provided to select which indicators may cause interruption and when interruption will be permitted. Priorities can thus be established. If more than one interrupting condition should occur at one time, the system takes them in order. Special provisions are made to permit interruptions to any level to occur without causing program confusion.

Address Monitoring

Address monitoring facilities are provided for two reasons. One is to make it possible for a program supervising the debugging of another program to detect when reference is made to a memory location outside the area assigned to that program. Another is to protect one program from accidental destruction by another program being executed on a multiprogrammed basis while the first program is waiting for service.

Elapsed Time Clock

An Elapsed Time Clock is built in to measure elapsed time over relatively short intervals. It can be set to any value at any time, and an indicator shows when the time period has ended. This indicator will cause automatic program interruption.

RELIABILITY

A. Components

Within the system all circuits are designed for a maximum margin of reliability in relation to the effect of aging on the components.

These include transistors developed by IBM of an exclusive nature such as NPN and PNP Drift Transistors and power transistors. These transistors are at present in production within IBM. In addition, outside sources have already been established for quantity production to IBM specifications.

B. Circuitry

The development of the NPN Drift Transistor by IBM has made possible circuitry operation in the complementary mode. This increases reliability because of the positive signal indication for both binary states. Circuit design affords maximum reliability because all components are taken to end of life on both the driving circuit, the circuit under consideration, and the circuit being driven by the circuit under consideration.

C. Experience

IBM's experience on 15 currently installed Type 608 transistorized computers shows extremely high reliability.

Because Drift Transistors are manufactured using the same techniques as junction alloy type, it is fully expected that the same reliability will be obtained and in fact surpassed.

The same degree of reliability has been borne out by initial experience and testing of circuit design in the Stretch project.

September 30, 1958

APPLIED PROGRAMMING SUPPORT

A group of programs will be provided with the STRETCH system. As presently comprised, this group of programs will consist of:

1. A supervisory control program to handle the interrupt system, input-output units, and communication with the machine operator.

2. Programs to facilitate program check-out.

3. Certain programs of general utility such as loaders, conversion, editing programs and common mathematical functions.

4. An automatic programming system performing:

   A. Assembly (Translation of codes written 1-for-1).

   B. An algebraic language translation to machine language translation system.

September 30, 1958

SUMMARY

The STRETCH computer system represents a most significant advance in computer technology. It provides arithmetic speed and memory capacity for solving important scientific problems which are impractical on present computers but vital to our national progress and security.

In addition to the advanced solid state components used, STRETCH contains many new sophisticated logical concepts and instructions. Assembly programs and other library programs will be available with the STRETCH system.

The system will also bring with it the high level of service with respect to maintenance, education and other specialized staff service for which IBM is famous.