Crossing the Divide: Architectural Issues and the Emergence of the Stored Program Computer, 1935–1955

PAUL CERUZZI

The rapid advance of computing technology since the 1940s has created an impression that all that happened in computing before then was somehow mere prologue to the real history. According to this popular notion, the computer age began with the invention of machines that computed at electronic speeds, that were capable of automatic sequence control with conditional branching, and that stored their programs internally. The classification of computing into "generations" with the "first" generation being those with vacuum tubes further reinforces this notion. This paper looks at some examples of machines built in the 1930s and 1940s that straddle both ages: machines that had some sort of sequence control, partially electronic counting circuits, or primitive branching capabilities. In particular, I examine a few systems that reveal especially well the nature of this transition: the ensembles of punched card equipment used by L.J. Comrie and Wallace Eckert for scientific instead of business use; the "Aberdeen Relay Calculator" that IBM built for the U.S. Army; and the "Card Programmed Calculator" that Northrop Aircraft invented for engineering applications that IBM later marketed.

Introduction

Since 1950, the technology of calculation and data processing has advanced at a pace that must be considered rapid by any measure. Faced with that, historians of computing often believe they can do little more than chronicle each dramatic advance, each milestone that marks the passing of a certain threshold of computing "power," usually defined as processing speed or memory capacity. Such listings are valuable but do little to aid one's understanding of the subject. Historians attempting to chronicle this era "stand before the daunting complexity of a subject that has grown exponentially in size and variety, which looks not so much like an uncharted ocean as like a trackless jungle. We pace on the edge, pondering where to cut in."[1]

The rapid pace of innovation after 1950 has also made it difficult to connect the history of modern computing with what happened before World War II. Interest in Charles Babbage, who tried but failed to build a "modern" computer (defined as having the design of post-1950 machines), is the exception that proves the rule.[2] The familiar division of computing history into "generations," with the first one beginning with the UNIVAC in 1951, reinforces the notion that everything that happened before then was only prologue to the story. (The generational metaphor had been familiar among engineers before the invention of the computer. Informally, the classification of computer generations according to tubes, transistors, and integrated circuits seems to have been established by 1960. It entered much wider use with IBM's 1964 introduction of System/360. IBM heralded the 360 as the first of the "third-generation" machines, but it did not use integrated circuits.[3]) Surely that view must be wrong. But where and how does one make the connection?[4]

James Cortada has shown that nearly all of the major players in electronic computing's early years—IBM, NCR, Burroughs, and Remington Rand—had deep roots in the "office appliance industry," supplying mechanical or electromechanical equipment to businesses since the late 19th century.[5] Arthur Norberg has shown that the ways that businesses used punched card equipment before World War II persisted into the postwar era, as those customers shifted to the radically different technology of electronic computers.[6]

Despite the connections Cortada and Norberg make between prewar and post-1950 computing, they also imply that there was a fundamental change in the nature of computation after 1945. The Computer Museum of Boston calls that change the crossing of the "Newton–Maxwell Gap": the difference in speeds of mechanical devices governed by Newton's laws of motion versus electronics devices governed by Maxwell's laws of electromagnetic radiation.[7] Proponents of electromechanical calculators, who frequently boasted that their machines were so reliable they could run unattended overnight, overlooked the fact that a machine like the ENIAC, if it could be kept running for as little as an hour, could do more work than an electromechanical machine like the Harvard Mark I running continuously, day and night, for three

months. (I define "work" simply by comparing the times each machine took to multiply two decimal numbers together: six seconds for the Mark I, .003 second for the ENIAC. Obviously, this is an oversimplified picture, but in general the comparison is valid.) Even if nothing else had changed, this speedup would have forced some sort of change in the nature of processing.

But something else did change. There was also a change at what we now call the architectural level of computing. The increase in speed helped propel this change, but it was occurring anyway, and in places it proceeded independently of advances in the base technology of vacuum tubes or relays.

This shift was one from an architecture that processed data in parallel to one that processed data serially. This paper examines the nature of that shift. It begins by taking a careful look at how people perceived the nature of computation in the late 1930s and how definitions of sequential versus parallel processing changed during the transition. By looking closely at several machines from the 1935–1945 era, I hope to build an architectural bridge across the chasm from computing's "prehistoric" period to its modern era.[8]

The von Neumann Model

A good place to start is with the architecture that emerged at the end of this period and that has remained remarkably stable over waves of successive advances in technology. It has come to be known as the "von Neumann" architecture. How it came to be called that will be discussed shortly; first, I will describe what that term has come to mean. As John Mauchly said, "Calculations can be performed at high speed only if instructions are performed at high speed."[9]

The modern definition of a computer is that it is a machine that handles data automatically under the direction of a set of instructions specified in advance. Among the ways it handles data are encoding or decoding, storing or retrieving from storage, sending data from one part of the machine to another (sometimes even over larger physical distances), and, of course, processing data (transforming it from one form to another by performing operations governed by laws of arithmetic or logic).

A computer with a von Neumann architecture, first of all, has separate functional units that process and store data. Typically, there is only a single channel between these two units through which all transfers of information must go (the so-called von Neumann bottleneck, about which more later). This feature arose mainly for engineering reasons: It was easier to design storage cells that did not also have to perform arithmetic on items stored therein.

Second, instructions and data are both stored in the same central memory device, which is designed so that any item can be retrieved as quickly as any other. This concept first arose after considering that the processing unit of a computer should not have to remain idle while awaiting delivery of the next instruction.[10] Also, engineers recognized that the ratio of instructions to data would vary for each problem, so it would not make sense to dedicate separate, and expensive, storage device units to each. This design implies that one may treat a coded instruction as a datum and perform an operation on it, thus changing it into another instruction.

Third, typical von Neumann machines internally manipulate information in the binary number system, encoding not only numbers but also alphabetic characters, instructions, pictures, and sounds as strings of binary digits. Such machines manipulate information in blocks called words, whose length typically ranges from eight to 64 or more binary digits.

Finally, the basic cycle of a von Neumann computer is to transfer an instruction from the store to the processor, decode that instruction, and execute it on data retrieved from another part of the store. Once the processor executes an instruction, it fetches and executes another, typically from the very next position in memory unless directed elsewhere. Having a fast, random access memory means that the processor can branch to another stream of instructions quickly whenever needed. This cycle is basically sequential. The flow through the instructions stored in memory is likewise sequential and linear (Belzer et al. provide a typical modern description of the architecture.[10]), except when explicit branch instructions are encountered. This concept of fetching and then executing a linear stream of instructions is perhaps the most lasting of all; even computer designs that purport to be "non–von Neumann" typically retain the fetch–execute heartbeat of a single-processor machine.[11] As the late Alan Perlis once remarked, "Sometimes I think the only universal in the computing field is the fetch–execute cycle."[12]

Historians have sought to qualify the popular notions, first, that this architecture has not changed since 1945 and, second, that it was solely the invention of von Neumann. (One of the most balanced assessments of the various claims is found in William Aspray. Aspray notes those who worked with von Neumann prior to the appearance of the report and also argues that "credit for making stored programming a significant contribution should go to the computer manufacturers who implemented it in their products."[13]) These arguments, however relevant, should not obscure the fact that among the community of engineers and computer scientists between 1950 and 1985, that architecture was one they regarded as stable. (For example, in the entry on "Computer Science" in the 1976 *Encyclopedia of Computer Science*, S. Amarel states, "the stored program digital computer ... provides a methodologically adequate, as well as a realistic, basis for the exploration and study of a great variety of concepts, schemes, and techniques of information processing."[14])

There are many accounts of the relationship of von Neumann with Eckert and Mauchly and the relative contributions each made to the EDVAC report.[15] Mauchly's own account is told in *Datamation*.[16]

For this paper, it will suffice to note that the term *architecture* encompasses more than one simple concept; it is an aggregate of ideas interacting with each other in complex ways. Central to it is the notion of the stored program, but even that must be understood in the context of number base, word length, and processor design. Many computer designers recognized the advantages of the design described in the documents of the late 1940s. Each had to balance theoretical arguments favoring it with practical considerations of the limits of electronic components, especially memory devices. Social and economic forces were also involved. By the mid 1950s, the term *von Neumann architecture* came to represent the consensus that emerged, obscuring the messy details of these arguments in the process. Von Neumann's stature as a revered mathematician bolstered the claim of legitimacy to the design that emerged victorious at the end of this period.[17]

If the outline above describes what was accepted as the definition of a "computer" by the mid 1950s, what was an equivalent

architectural definition for computing equipment of the mid 1930s? This is a meaningless question: The term *computer architecture* is a modern one, dating from the late 1950s.[18] People who built or operated mechanical adding machines or ensembles of punched card equipment in the 1930s did not know that they were "computer architects," any more than they could have known that their work would lead to Windows-based workstations on the desks and in the homes of millions of people. But we can look at the way such equipment operated from the view of the overall flow of information.

Serial Versus Parallel Arithmetic

It now seems a relatively minor issue, but the question of whether a (binary or decimal) number was handled one digit at a time or as a whole was, in the late 1940s, the subject of some debate. In a mechanical adding machine or calculator having a 10-key keypad (also a telephone), one keyed in the successive digits of a number one at a time. On calculators having a set of keys for each decimal place—especially the Comptometer—one could simultaneously key in more than one digit, although in practice this was obviously limited by the number of "digits" on one's hands. The Comptometer carried out an addition as soon as a key was pressed; other calculators required that an operator key in all the digits, then pull a lever or turn a crank to initiate the operation. Punched card equipment likewise read or punched holes one digit at a time, but after reading they could operate on all the digits simultaneously.

The *First Draft of a Report on the EDVAC*, written in 1945, described a machine that economized on hardware by doing everything, including addition, one bit at a time (serially). In von Neumann's words:

> Thus, it seems worthwhile to consider the following viewpoint: The device should be as simple as possible, that is, contain as few elements as possible. This can be achieved by never performing two operations simultaneously, if this would cause a significant increase in the number of elements required.
>
> It is also worth emphasizing that up to now, all thinking about high-speed digital computing devices has tended in the opposite direction: toward acceleration by telescoping processes at the price of multiplying the number of elements required. It would therefore seem to be more instructive to try to think out as completely as possible the opposite viewpoint (i.e., doing only one thing at a time).[19]

A few years later, when von Neumann moved to the Institute for Advanced Study (IAS) at Princeton University and set up another computer project, he relaxed that constraint. The IAS computer would still execute instructions one at a time, but each instruction, say, addition, would operate on 40 bits at a time, which the IAS team called a "word." I believe this is the first use of the term *word* to describe the aggregate of binary digits that a computer would handle at a time.[20] That yielded greater processing speeds.[21] The IAS computer was completed in 1952. Through reports by Arthur Burks, Herman Goldstein, and von Neumann, as well as through interim engineering reports on the construction of the machine, the IAS design philosophy began to have a wide influence well before the IAS machine was running.[22]

In their reports, Burks, Goldstine, and von Neumann referred to the IAS design as a "parallel machine." They were referring to the word-at-a-time—as contrasted with the EDVAC's bit-at-a-time—arithmetic. But the IAS computer still fetched and executed only one instruction from memory at a time—the infamous von Neumann bottleneck. The IAS's parallel transfer of a word's digits did not affect the sequential fetching and executing of operations from memory. (Note that the ENIAC, whose 18,000 vacuum tubes drove Wallace Eckert and Mauchly to seek a more elegant design, was a truly parallel machine: It could execute more than one instruction at a time. After the ENIAC was moved to the Ballistic Research Laboratory at Aberdeen, Maryland, it was modified so that it no longer could do that. That change was made to make it easier to program; it did not reduce the machine's complexity or number of tubes.) At least one ENIAC user was sorry to see the change.[23]

Note that the ENIAC, whose 18,000 vacuum tubes drove Wallace Eckert and Mauchly to seek a more elegant design, was a truly parallel machine: It could execute more than one instruction at a time.

The IAS model of parallel arithmetic (retaining sequential instruction fetching and executing) became the norm. Most designers chose a word length that would give a reasonable decimal precision to satisfy most scientists or engineers (the IAS's word length of 40 bits had a precision of about 12 decimal places). Aside from its use of binary arithmetic, this design resembles the functioning of a mechanical desk calculator of the era: a precision of about 10 decimal numbers and simultaneous addition of all the digits of a number with each turn of the crank.

In the following decade, many computer designers would choose a similar word length, but not all. The designers of the Whirlwind, for example, chose the much shorter word of 16 bits, partly because it was not primarily conceived as a machine that would do scientific calculations, but rather as a device suitable for controlling other machinery. Its designers later recalled that choice as the result of considering that 16 binary digits was sufficient for addressing the machine's memory. To have more would be wasteful of circuitry.[24] (The long word length of the IAS meant that essential extra bits had to be handled that were not being used for memory access. One remedy for this was to adopt multiple-address codes, whereby a single word held either one instruction and more than one address or else more than one instruction.[25])

Designers of computers for business often chose a different approach. Business calculations often did not require the decimal precision that scientific calculations did. Also, internal processing speeds were less critical than the speeds of other parts of a business computing system, so it could tolerate a processing unit that handled digits serially. The most successful of these designs was the IBM 1401 computer, announced in 1959. The 1401's word length could vary. To each coded character a bit was appended, signaling whether that character was the end of one word and the

start of another.[26] (Prior to the 1401, IBM had produced three other computers having this architecture.)

By the 1960s the simple notion of the processing of a word, whose length corresponded to a desired arithmetic precision, had become a complex design parameter. Indeed, one of the most important ways that the von Neumann design evolved over the years was the invention of different ways of allowing a short word length to handle numbers of high precision, as well as to address large blocks of memory. The mini and later the personal computer took advantage of these techniques to lower costs.

The single-bit-at-a-time design described by the 1945 EDVAC Report vanished by 1960. But it was resurrected a dozen years later. In the early 1970s, a number of manufacturers introduced pocket calculators; at least one of them, the Hewlett-Packard HP-65, returned to the EDVAC's way of handling numbers. Von Neumann initially favored serial arithmetic in the EDVAC to save on components. The HP-65's designers used serial arithmetic because it required fewer internal connections from one chip to another. That meant fewer wires, hence a more compact circuit board. Without serial arithmetic, the machine would not have fit in a shirt pocket. Serial arithmetic made the machine run slowly. But since calculator programs were relatively short, this was unnoticed by typical users.[27]

Sequential Versus Parallel Operations

More significant was the question of whether operations were performed on numbers sequentially or in parallel. Here an analysis of the precomputer era is more subtle, since by definition these machines relied on human intervention to do what later became known as executing a sequence of operations or a "program." The flow of instructions and data implied by the von Neumann model closely patterned the way human computers performed scientific calculations using mechanical calculators, books of tables, drafting tools, and pencil and paper[28] (see Fig. 1). (For example, in von Neumann's address to the Mathematical Computing Advisory Panel of the U.S. Navy in May 1946, he compared the electronic computers then under development to "what is at present still the major practical mode of computing, namely, human procedure with an electromechanical desk multiplier."[29]) But the number of people engaged in this activity was small compared to those engaged in accounting, inventory control, and other related business activities. (For an account of the early development of this activity, see James Beniger.[30]) For this latter group, the chief technical aid was the punched card machine; typically, an ensemble of three to six different punched card devices would comprise an installation.[6,31] The digital electronic computer had a profound impact because of its penetration into the world of business and commerce. What was the architecture of a punched card installation?

Punched card machines are often called "unit record equipment." With them, one encodes on a single card all relevant information about a particular entity (e.g., a sales transaction). Sets of columns on the card represent specific "fields" and record details such as sales commission, item sold, and salesman. Thus a single card can serve multiple uses by running it through appropriate equipment. A set of pluggable wires on each machine allowed one to count, sort, tabulate, and print on any desired set of columns.

Historical accounts of punched card machinery have described the functioning of the individual machines in great detail.[32,33] More relevant is what went on in the entire room comprising a punched card installation. That room—including the people in it—and not the individual machines is what the electronic computer eventually replaced.

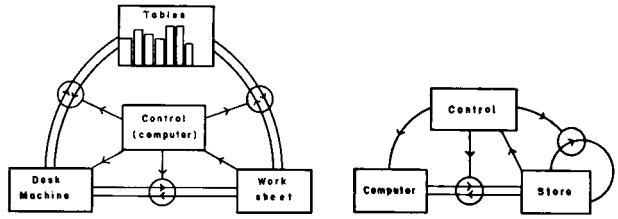

Fig. 1. Block Diagram of the human (left) and machine (right) process of computing.
(From D.R. Hartree, *Calculating Instruments and Machines* (Urbana, 1949), p. 57)

Consider the flow of data in a typical transaction using this equipment: Using cards punched with sales information, how would one obtain a report listing the total cash received for each specific item sold? The first step would be to run the cards through a sorter wired to sort on the columns that encoded the product number. Next, this deck of sorted cards would be run through a multiplying punch wired to compute the product of the quantity sold and the price per item. This product would be punched into a blank column of each card or onto new cards if necessary. Finally the cards would be run through a tabulator, which would sum the totals just computed.

The algebraic representation of this calculation would be as follows:

```
Amount received =
Number sold × Cost (Product a) +
Number sold × Cost (Product b) +
Number sold × Cost (Product c) +
etc.
```

All the multiplications would be done first, using one piece of equipment. (A Multiplying Punch was the obvious choice here, although none was available until 1931, when IBM offered its Model 601. But for many problems, an operation that combined the tabulator and sorter would multiply just as well, and this technique, called "progressive digiting," was preferred, even after multiplying punches became available.) Then, another piece of equipment would do all the additions, yielding the final result. In other words, the calculation would be performed on the equations by columns if they were arranged in a table. Transfer of control from one arithmetic operation to another was effected by a person who carried decks of cards from one machine to another.

In this scheme, operations are done on one card at a time as each goes through a machine. But they are not done sequentially as a digital computer described in the EDVAC report would do them. A modern computer could and may often be programmed to replicate the above process by storing data in an array and carrying out the operations in the same order as above. But in the late 1940s, great emphasis was placed on the fact that these new machines did things differently. The reason was that the kinds of problems envisioned for the new type of machines had a fundamental difference: The number to be multiplied or added was often not known until the immediately previous operation was

performed. This was especially true for obtaining the numerical solution to a differential equation—something for which nearly every one of the early machines was intended.

One could solve such a problem with punched card equipment by taking a single card and running it through each of the machines as necessary. But with only one known exception (an important one, described later), no one thought of using punched card equipment this way. Thus came the notion of "sequential" processing (e.g., the "Automatic Sequence Controlled Calculator" of Howard Aiken, a Harvard physicist) controlled by a punched tape or other programming mechanism, in which a succession of operations is performed on a single datum at a time. The other method of processing information was entrenched into business practices by the mid 1930s. It was reinforced by the deep penetration of the punched card equipment salesmen into the accounting offices of their customers.[32,34] For the stored program digital computer to prevail, it required more than just building computers that were reliable. It also required convincing both these customers and the salesmen who supplied them that the computer's way of doing calculations sequentially offered overwhelming advantages.

The first recognition that such methods might be preferable appeared when people began using punched card installations for scientific, rather than commercial, calculations. One of the first to do that was L.J. Comrie, superintendent of H. M. Nautical Almanac Office, later a founder of the Scientific Computing Service in London. In 1928 Comrie began a project to calculate future positions of the moon, based on calculations done for E.W. Brown's *Tables of the Motion of the Moon*, a set of 180 separate tables of 660 pages in length.[35,36] Comrie had data from Brown's tables punched onto 45-column Hollerith cards (ultimately half a million cards). Using a sorter, a tabulator, and a duplicating punch, Comrie computed the principal terms of the motion of the moon from 1935 to 2000. He estimated that "something like 100 million figures were added in groups and group totals printed in the course of seven months."[37] The system computed at about 20 additions per second.[38]

Comrie's work had influence across the Atlantic. Through Brown, who saw Comrie's system doing the same work that he had laboriously done by hand, the idea passed to Eckert, a student of Brown's at Yale. In 1934 Eckert (no relation to Presper Eckert of the Moore School) formed the Thomas J. Watson Computing Bureau in New York, where punched card machines supplied by IBM were applied to scientific problems. In 1940 Eckert summarized his work in a classic book, *Punched Card Methods in Scientific Computation*.

By the next decade, news of stored program, sequential notion of computing had eclipsed the methods described in Eckert's book. But punched card installations were still doing the bulk of computing. And punched card machines, Eckert said, "are all designed for computation where each operation is done on many cards before the next operation is begun." His book had emphasized how one could use existing equipment to do scientific work, but he stated that it was not worth the "expense and delay involved" in building specialized machines to solve scientific problems.[39]

Independent of Eckert, Aiken was working with IBM in the early 1940s to do just what Eckert felt was impractical: build an expensive machine designed from the start for scientific computations. Because of World War II, the expense was not a problem, and there was very little delay. With IBM's support, his "Automatic *Sequence* Controlled Calculator" was unveiled at Harvard University in 1944 after several years of design and development (emphasis the author's). In his initial proposal for that machine, written in 1937, Aiken described how he wanted something that would compute sequentially, not like the way punched card installations worked. Describing punched card installations of the kind Comrie and Eckert used, he said:

> This process, however, is the reverse of that required in many mathematical operations. Calculating machinery designed for application to the mathematical sciences should be capable of computing *lines* instead of *columns*, for very often, as in the numerical solution to a differential equation, the computation of the second value in the computed table of a function depends of the preceding value or values [emphasis added].[40]

Aiken's machine, later known as the Harvard Mark I, used IBM punched card registers that were modified to calculate sequentially.

More relevant is what went on in the entire room comprising a punched card installation. That room—including the people in it—and not the individual machines is what the electronic computer eventually replaced.

Aiken had some influence, but he did not contribute to the evolution of computer architecture as practiced at IBM, in part because of a poor relationship with IBM. That company continued to rely on Eckert for advice.[41] Eckert's modest vision was overwhelmed by the crush of scientific work placed on the machinery by the demands of World War II. With the entry of the United States into the war, urgent calculations were performed by ever-greater teams of people, by analog computers such as the Differential Analyzer, and by punched card installations of the type Eckert pioneered. The most legendary of the last was the punched card installation at Los Alamos, where it was used alongside teams of people operating mechanical calculators.[42,43] For a popular account of computing at Los Alamos, see Richard Feynmann.[44] But Eckert continued to believe that the punched card machinery, with its architecture of computing by columns not lines, was the proper direction computing ought to go.[45]

Ironically, Eckert was among the first to venture away from the architecture of punched card installations toward one more like stored program computing. He did not build a specialized machine at his lab, but he did have a device called a "Control Switch" placed between the multiplier, tabulator, and summary punch. Its function was to allow short sequences of operations (up to 12) to be performed on a single card before the next card was read.[46] Of the 12 steps, only the first six were performed automatically, the rest required some human intervention.

At the U.S. Army's Ballistic Research Laboratory at Aberdeen, Maryland, there was a similar multitude of approaches. With Eckert's advice, IBM built and installed two specially made devices that were essentially punched card machines but modified to do sequential processing as Eckert suggested in 1940. IBM called

these machines the Aberdeen Relay Calculators, later known as the Pluggable Sequence Relay Calculator (PSRC) These machines were the missing link between punched card equipment and the stored program digital computer[47] (see Fig. 2).

In late 1945, three more were built: one for the Naval Proving Ground at Dahlgren, Virginia, and the other two for the Watson lab in New York. Their architecture was based on punched card equipment, but during each machine cycle (i.e., during the reading of a card, which took about half a second), the machine could execute a sequence of up to 48 steps, controlled by a sequencing device called a "hub." It was also possible to execute a "double cycle," with the feeding of the next card held until all operations were finished. Control signals at each "sequence point" on the hub were transmitted to four other hubs, so the machine could carry out up to four parallel arithmetic operations (even more if these hubs were in turn connected to other hubs). Brian Randell notes that this method of programming demanded "the kind of detailed design of parallel subsequencing that one sees nowadays at the microprogramming level of some computers."[48]

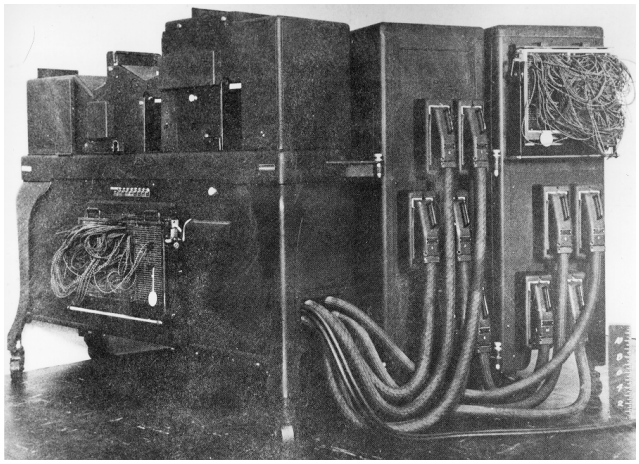

Fig. 2. Pluggable Sequence Relay Calculator, also known as the Aberdeen Relay Calculator.

(IBM Archives)

It is not known how the users at Aberdeen regarded these machines. When properly programmed, they were faster than any other machine that used mechanical or electromechanical computing elements (i.e., relays). One can infer that they were successful by the fact that additional copies were built in 1945. Later that year the machines were returned to IBM, given greater storage capacity and more flexible controls, and reinstalled at Aberdeen. That was after the ENIAC, the electronic computer intended for Aberdeen, was completed. Both were still in use in 1952, by which time the Ballistic Research Laboratory had not only the ENIAC but also the EDVAC, ORDVAC (both stored program computers), an IBM Card Programmed Calculator (described next), and the Bell Labs' Model V, a very large programmable relay calculator.[49] Again, I am following Basalla's reasoning from an analogy of the evolution of technology with biological evolution.

For work that required simple and infrequent multiplications, most punched card installations used combinations of tabulators and sorters. IBM first introduced a multiplying punch (the 601) in 1935, which soon became popular with those using punched card equipment for scientific or statistical work. In 1946 IBM introduced the 603, which was the first commercial IBM product to use vacuum tubes for calculating. Two years later, IBM replaced the 603 with the 604, which also used vacuum tubes. But more than that, the 604 incorporated the sequencing capability pioneered by the Aberdeen machines. Besides the usual plugboard control common to other punched card equipment, it could execute up to 60 steps for each reading of a card and setting of the plugboard.[50]

The 604, and its successor the IBM 605, became the mainstay of scientific computing at many installations for years, until reliable commercial computers became available in the mid 1950s. It was one of IBM's most successful products during that era: More than 5,000 were built between 1948 and 1958. It was in the context of setting up the 60 to 80 steps—all that these machines allowed—that many people learned the concepts of computer programming that von Neumann and his colleagues at the IAS described.

One of IBM's biggest engineering customers, Northrop Aircraft of Hawthorne, California, connected a 603 multiplying punch to one of its tabulating machines instead of the usual arrangement of connecting it to a reproducing punch that could only read or punch cards. The result was a machine that could print the result of a calculation on paper instead of punching it on cards. With a slight further modification and the addition of a small box that stored up to 16 10-digit numbers plus a sign in banks of relays, the machine could use punched cards run through the tabulator to control the sequences carried out by the multiplier (instead of having control performed only by plug wires)[51]—up to three storage boxes could be connected to each machine.

In a sense, the Northrop arrangement was no different from an ordinary punched card installation, except that a set of cables and control boxes replaced the persons whose job it was to carry decks of cards from one machine to the next. One of the Northrop engineers recalled years later that they rigged up the arrangement because they were running a problem whose next step depended on the results of the previous step. What this meant was that the normal decks of cards that one might run through a machine were reduced to "a batch of one [card], which was awkward," said G.J. Toben.[52] In other words, with the cables connecting the machines, Northrop engineers turned the installation into one that executed instructions sequentially.

IBM later marketed a version of this arrangement as the Card Programmed Calculator (CPC) (see Fig. 3).[53] As a system, there were fewer installations of it than of the tabulators and multiplying punches that comprised it. Perhaps several hundred in all were rented between 1948 and the mid 1950s. But even that number was many times greater than the number of stored program installations worldwide until about 1954. For engineering-oriented companies like Northrop, the CPC filled a pressing need that could not wait for the problems associated with marketing stored program computers to be resolved.[54,55]

The Aberdeen calculators and the 604 were transitional machines between calculators, tabulators, and computers having the stored program architecture patterned after the EDVAC or IAS machines. The CPC carried the punched card approach too far to be of value to computer designers. By the time of its introduction in the late 1940s, the stored program architecture had already been publicized and was beginning to prevail. In that sense it was

less a transition than an example of carrying the momentum of the earlier design farther than it would have gone on technical merits alone. Its combination of program cards, plug boards, and interconnecting cables was like the epicycles of a late iteration of Ptolemaic cosmology while the Copernican system was already gaining acceptance. (This design did not die out entirely. In France, Compagnie des Machines Bull introduced in 1952 a machine having a very similar architecture. Called the "Gamma 3," it was very successful and was one of the first products produced in France to achieve a critical mass of sales.[56])

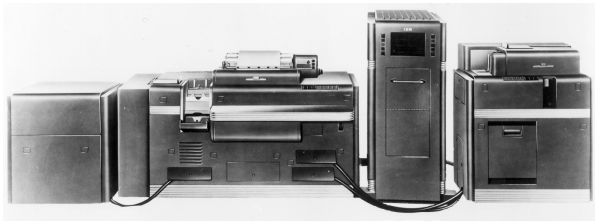

Fig. 3. IBM Card Programmed Calculator.

(IBM Archives)

Customers needing to solve difficult engineering problems, however, accepted the CPC. It offered the computational capability that only electronic computers approached, but at lower cost. And it was available. Customers such as the Southern California aerospace firms, typified by Northrop, carefully evaluated the CPC against the claims of vendors of the first stored program machines (especially the smaller drum-based ones that began to appear after 1951). Nearly all aerospace firms installed at least one CPC.

The above discussion has dwelled on aspects of hardware design. "Programming" would not be an appropriate word to describe the activity of a person working with a mechanical calculator or the process by which decks of cards were carried around a punched card installation. The CPC, however, was a bridge from that method to programming as we now define it. Because the CPC did have the ability to execute a number of sequential steps, in part controlled by punched cards, the act of preparing a program was a more distinct activity than merely "running the equipment." Also, the hardware limitations meant that CPC users went to great efforts to analyze a problem and try to make it solvable within those limits. Both those activities were carried over to stored program software. One early user remarked:

> Ingenuity could be expended on designing [plug] boards to optimize performance on particular problems that taxed the capacity of the computer, or it could be expended on the design of general-purpose boards for use in large classes of problems. A set of general-purpose boards represented a language for the CPC, and a number of languages were tried and used. Most scientific installations finally settled on a wiring that made the CPC appear to be a floating point machine with three-address logic and with a standard vocabulary of built-in routines like Square Root, Sine, and Exponential. Experience with the CPC systems had many influences on the programming systems that were designed for the stored program computers that followed.[57,58]

Conclusion

Since the mid 1980s, computer architecture has again become the focus of a debate of a similar intensity as experienced in the late 1940s. Inexpensive microprocessors have given new life to massively parallel architectures, which, if they prevail, will spell the end of the classic von Neumann structure. The emergence of RISC architecture has also been a central story of recent years. Many of the current debates echo what was debated decades ago. Computer designers at all levels seem to go through successive "wheels of reincarnation" as they rediscover old concepts long thought irrelevant or obsolete.[59] The examples cited above do not invalidate the general notion, however, that the nature of computing changed drastically after the invention of the stored program, electronic digital computer. But they do show that just as users and suppliers of business equipment crossed the Great Divide of the 1940s, so, too, did some of the internal, architectural features of the machines themselves.

References

[1] M.S. Mahoney, "The History of Computing in the History of Technology," *Annals of the History of Computing*, vol. 10, pp. 113-125, 1988.

[2] A. Hyman, *Charles Babbage, Pioneer of the Computer*. Princeton, N.J.: Princeton Univ. Press, 1982. I.B. Cohen, "Babbage and Aiken," *Annals of the History of Computing*, vol. 10, pp. 171-193, 1988.

[3] John Leech, private communication; also J.P. Eckert, "In the Beginning and to What End," *Computers and Their Future: Speeches Given at the World Computer Pioneer Conference*, Llandudno, Wales, 1970, section 3.

[4] G. Basalla, *The Evolution of Technology*, Cambridge, England: Cambridge Univ. Press, 1988.

[5] J.W. Cortada, *Before the Computer: IBM, NCR, Burroughs, and the Industry They Created, 1865-1956*. Princeton, N.J.: Princeton Univ. Press, 1993.

[6] A. Norberg, "High Technology Calculation in the Early Twentieth Century: Punched Card Machinery in Business and Government," *Technology and Culture*, vol. 31, pp. 753-779, Oct. 1990.

[7] Computer Museum, Boston: slide 1.4: "Speed of Calculation in Generation," Set 1: Information Processing History Graphs and Charts, from its History of Computing Slide Series.

[8] R.E. Smith, "A Historical Overview of Computer Architecture," *Annals of the History of Computing*, vol. 10, no. 4, pp. 277-303, 1989.

[9] J. Mauchly, "Preparation of Problems for EDVAC-Type Machines," *Annals of the Harvard Computation Laboratory*, vol. 16, pp. 203-207, 1947; reprinted in B. Randell, ed., *The Origins of Digital Computers: Selected Papers*, 2nd ed. Berlin: Springer Verlag, 1975, pp. 365-369.

[10] "Digital Computer Architecture," J. Belzer, A. Holzman, and A. Kent, eds., *Encyclopedia of Computer Science and Technology*. New York: McGraw-Hill, 1970, vol. 7, pp. 289-326.

[11] For an example of how this subject is treated in introductory college-level textbooks, see H.G. Kershner, *Introduction to Computer Literacy*. Lexington, Mass.: D.C. Heath, 1990; and J.B. Rochester and J. Rochester, *Computers for People*. Homewood, Ill.: Richard D. Irwin, 1991.

[12] A. Perlis, "Epigrams on Programming," *ACM Sigplan Notices*, Oct. 1981, pp. 7-13.

[13] W. Aspray, "The Stored Program Concept," *IEEE Spectrum*, p. 51, Sept. 1990.

[14] "Computer Science," *Encyclopedia of Computer Science*. New York: Van Nostrand, 1976, pp. 314-318.

[15] J.P. Eckert, "Disclosure of a Magnetic Calculating Machine," Univ. of Pennsylvania, Moore School of Electrical Eng., memorandum of Jan. 29, 1944; in H. Lukoff, *From Dits to Bits: A Personal History of the Electronic Computer*. Portland, Oreg.: Robotics Press, 1979, pp. 207-209; also J.P. Eckert and J. Mauchly, "Automatic High Speed Computing: A Progress Report on the EDVAC," portions reprinted

in L.R. Johnson, *System Structure in Data, Programs, and Computers*. Englewood Cliffs, N.J.: Prentice Hall, 1970, pp. 184-187.
[16] "Amending the ENIAC Story," *Datamation*, pp. 217-220, Oct. 1979.
[17] H. Goldstine, *Computer From Pascal to von Neumann*. Princeton, N.J.: Princeton Univ. Press, 1972.
[18] W. Buchholz, ed., *Planning Computer System: Project Stretch*. New York: McGraw-Hill, 1962.
[19] J. von Neumann, "First Draft of a Report on the EDVAC," Univ. of Pennsylvania, Moore School of Electrical Eng., June 30, 1945, p. 20. (Reprinted in the *Annals of the History of Computing*, vol. 15, no. 4, pp. 27-75.)
[20] A.W. Burks, H.H. Goldstine, and J. von Neumann, *Preliminary Discussion of the Logical Design of an Electronic Computing Instrument*, part 1, vol. 1, 2nd ed. Princeton, N.J., Institute for Advanced Study, June 28, 1946, p. 4 and 5.
[21] M.V. Wilkes, *Memoirs of a Computer Pioneer*. Cambridge, Mass.: MIT Press, 1985, pp. 164-165.
[22] W. Aspray, *John von Neumann and the Origins of Modern Computing*. Cambridge, Mass.: MIT Press, 1991, pp. 86-91.
[23] D. H. Lehmer, "A History of the Sieve Process," N, Metropolis, J. Howlett, and G.-C. Rota, eds., *A History of Computing in the Twentieth Century*. New York: Academic Press, 1980, pp. 445-456.
[24] K.C. Redmond and T.M. Smith, *Project Whirlwind: The History of a Pioneer Computer*. Bedford, Mass.: Digital Press, 1980, pp. 52, 65.
[25] D. Knuth, "Von Neumann's First Computer Program," *Computing Surveys*, vol. 2, no. 4, p 251, 1970.
[26] C.J. Bashe, L.R. Johnson, J.H. Palmer, and E.W. Pugh, *IBM's Early Computers*. Cambridge, Mass.: MIT Press, 1986, p. 466-469.
[27] C.C. Tung, "The 'Personal Computer': A Fully Programmable Pocket Calculator," *Hewlett-Packard J.*, pp. 2-7, 1974.
[28] P. Ceruzzi, "When Computers Were Human," *Annals of the History of Computing*, vol. 13, no. 3, p. 237.
[29] *Annals of the History of Computing*, vol. 10, p. 248, 1989.
[30] J. Beniger, *The Control Revolution*, Cambridge, Mass.: Harvard Univ. Press, 1986.
[31] M. Campbell-Kelly, *ICL, a Business and Technical History*. Oxford, England: Oxford Univ. Press, 1989.
[32] M. Campbell-Kelly, "Punched-Card Machinery," W. Aspray, ed., *Computing Before Computers*. Ames: Univ. of Iowa Press, 1990, chapter 4.
[33] E.C. Berkeley, *Giant Brains*. New York: John Wiley, 1949, chapter 4.
[34] K. Flamm, "Review of *ICL, a Business and Technical History*," *Annals of the History of Computing*, vol. 13, no. 1, 1991.
[35] L.J. Comrie, "The Application of the Hollerith Tabulating Machine to Brown's Tables of the Moon," *Monthly Notices, Royal Astronomical Soc.*, vol. 92, no. 7, pp. 694-707, 1932.
[36] D. DeVorkin, "Interview of Paul Herget," *Amer. Inst. of Physics*, Apr. 1977.
[37] L.J. Comrie, "The Application of Commercial Calculating Machines to Scientific Computation," *Mathematical Tables and Other Aids to Calculation*, vol. 2, no. 16, pp. 149-159, 1946.
[38] L.J. Comrie, "Recent Progress in Scientific Computing," *J. of Scientific Instruments*, vol. 21, pp. 129-135, Aug. 1944.
[39] W. Eckert, *Punched Card Methods in Scientific Computation*. Cambridge, Mass.: MIT Press, 1984.
[40] H. Aiken, "Proposed Automatic Calculating Machine," memorandum written 1937, published in *IEEE Spectrum*, vol. 1, pp. 62-69, Aug. 1964.
[41] I.B. Cohen, *Howard Aiken, Computer Pioneer*. Cambridge, Mass.: MIT Press, 1991.
[42] N. Metropolis, "The Los Alamos Experience: 1943-1954," S. Nash, ed., *A History of Scientific Computing*. Reading, Mass.: Addison Wesley, 1990, pp. 237-250.
[43] N. Metropolis and E.C. Nelson, "Early Computing at Los Alamos," *Annals of the History of Computing*, vol. 4, pp. 348-357, Oct. 1982.
[44] R. Feynmann, *Surely You're Joking, Mr. Feynmann*. New York: W.W. Norton, 1985.
[45] J.P. Eckert, "Reply to George Stibitz," *Proc. Symp. Large-Scale Digital Calculating Machinery*, Cambridge, Mass., 1947; reprinted by MIT Press, 1985, p. 98.
[46] J. Lynch and C.E. Johnson, "Programming Principles for the IBM Relay Calculators," *Ballistic Research Laboratories, Report no. 705*, Oct. 1949, IBM Archives, Valhalla, New York.
[47] W. Eckert, "The IBM Pluggable Sequence Relay Calculator," *Mathematical Tables and Other Aids to Calculation*, vol. 3, pp. 149-161, 1948.
[48] B. Randell, ed., *The Origins of Digital Computers: Selected Papers*. Berlin: Springer-Verlag, 2nd ed., 1975, p. 188.
[49] Ballistic Research Laboratories, "Computing Laboratory," undated 15-page brochure, probably 1952, National Air and Space Museum, NBS Collection.
[50] M.R. Williams, *A History of Computing Technology*. Englewood Cliffs, N.J.: Prentice Hall, 1985, p. 256.
[51] W. Woodbury, "The 603-405 Computer," *Proc. Second Symp. Calculating Machinery*, pp. 316-320, Cambridge, Mass., Sept. 1949.
[52] Bashe et al., *IBM's Early Computers*, pp. 68-72.
[53] J.W. Sheldon and L. Tatum, "The IBM Card-Programmed Electronic Calculator," *Rev. Electronic Digital Computers*, Joint IRE-AIEE Conf., Feb. 1952, pp. 30-36.
[54] P. Ceruzzi, *Beyond the Limits: Flight Enters the Computer Age*. Cambridge, Mass.: MIT Press, 1989, chapter 2.
[55] Smithsonian Videohistory Program, Rand Corporation interview with Clifford Shaw, June 12, 1990, p. 13.
[56] B. LeClerc, "From Gamma 2 to Gamma E.T.: The Birth of Electronic Computing at Bull," *Annals of the History of Computing*, vol. 12, no. 1, pp. 5-22, 1990.
[57] S. Rosen, "Programming Systems and Languages—a Historical Survey," *Proc. 25th Spring Joint Computer Conf. 25*, pp. 1-15, 1964.
[58] D.B. MacMillan, "'Floating Decimal' Calculations on the IBM Card Programmed Electronic Calculator," *Mathematical Tables and Other Aids to Calculation*, vol. 5, pp. 86-92, 1951.
[59] T.H. Myer and I.E. Sutherland, "On the Design of Display Processors," *Comm. ACM*, vol. 11, no. 6, pp. 410-414, 1968.

Paul Ceruzzi is Curator of Aerospace Electronics and Computing at the Smithsonian's National Air and Space Museum in Washington, D.C. His duties include research, writing, exhibits planning, artifact collecting, and public outreach activities concerned with the subject of computing machinery as it applies to the practice of air and space flight.

Dr. Ceruzzi received a BA in 1970 from Yale University and PhD in American studies in 1981 from the University of Kansas. His graduate studies included a year as a Fulbright Scholar at the Institute for the History of Science in Hamburg, Germany, and he received a Charles Babbage Institute Research Fellowship in 1979. Before joining the staff of the National Air and Space Museum, he taught history of technology at Clemson University.

Dr. Ceruzzi is author or coauthor of several books on the history of computing and related issues: *Reckoners: The Prehistorv of the Digital Computer* (1983), *Computing Before Computers* (1990), *Smithsonian Landmarks in the History of Digital Computing* (1994), and *Beyond the Limits: Flight Enters the Computer Age* (1989).

The author can be reached at
National Air and Space Museum
Smithsonian Institution
Washington, DC 20560, U.S.A.