13.

How the Computer Works

Andrea Laue

Computing humanists engage digital technologies in their studies of humanistic artifacts. In this chapter, we will engage the computer as artifact, seeking a better understanding of how histories of culture and of technology converge in our personal computers (PCs). First, we will explore how computers "work", how their largely electrical mechanisms process data. Second, we will consider how computers "function", how in theory and in practice humans figure their working relationships with these machines. The computer was built to be a sophisticated device for the manipulation of symbols, and, at several levels, it is just that. But this description does not acknowledge that computers are also symbols, nor does it reveal the extent to which computers employ technologies symbolic of larger historical and cultural trends. In order to understand how computers function, we must understand how humans use computers to manipulate symbols but also how the mechanisms within the machine manipulate human users. I will consider here "Wintel" machines, desktop PCs running on technology largely developed by Microsoft and Intel. These systems draw extensively on John von Neumann's stored-program architecture (1945) and incorporate primarily Douglas Engelbart's input devices. While popular rhetoric encourages users to imagine digital technologies outside of various histories of technology, of culture and of labor, I argue here that those histories are essential to a proper understanding of how a computer functions.

How the Computer Works

In 1945 John von Neumann described a radical new architecture for computers, and although he was not imagining personal computers, his design largely informs the machines we use today. I will adopt his concept of the five logical parts of a computer – control, processing, storage, output, and input – and discuss them in the context of a more mundane narrative – just what happens when one types "Hi!" as the first line of an e-mail. There is not a one-to-one correspondence between von Neumann's logical parts and the hardware components present in modern machines, so I will demarcate a section for each logical part, providing a basic explanation of the role of that part, and explain in greater detail how one or two hardware components perform the operations characteristic of that logical part. My choice of components and sequence of discussion will be guided by the framing narrative, the story of how one types and sends an e-mail that says "Hi!" but I will occasionally offer asides rather than omit an essential component.

Input

Contemporary computers offer a range of input devices, including keyboards, the mouse, floppy disk drives, modems, and USB (universal serial bus) ports. Such devices are often called peripherals, a term that describes any component, internal or external, which adds hardware capabilities to the basic design of the system. Each of these peripherals offers a means of transferring data to the machine, an operation that may seem mundane now but was key for von Neumann as previous computers used only punchcards and paper tape. Modern input devices utilize several different standards and media, but the principles behind the transfer of data share fundamental characteristics. Since the user is likely to use a keyboard to input the characters "H", "I", and "!", we will start there.

Although most keyboards look quite similar from the outside, there are at least five different "insides" common today. These differ in the mechanisms engaged by the striking of a key, and the most common setup, the rubber dome keyboard, will be described here. Looking inside this keyboard we find three plastic sheets: one with conductive tracks, one that acts as a separator, and a final conductive layer. Two of these together form the key matrix, a circuit board in two layers – a bottom layer containing gaps that are completed when the top layer is pressed down

on to it. The third layer is shaped like the keyboard, with pliable rubber domes corresponding to each rigid plastic key. These domes contain a hard carbon center that, when depressed, completes a circuit. When released, the rubber dome regains its original shape, breaking the circuit.

Keyboards have simple microprocessors that constantly scan the circuit board looking for completed circuits. These detect both an increase and a decrease in current, the pressing and the releasing of a key. When a completed circuit is found, the microprocessor compares the completed circuit with its programmed character map, a sort of seating chart of characters. If more than one circuit is completed, the microprocessor checks to see if that combination of keys is recorded in the character map. The character map points to a specific scan code for the completed circuit, and it is this scan code that is sent to the computer for translation into the appropriate ASCII bit string. The scan codes are standard, but the character map is specific to a particular keyboard. The real significance of the character map lies in the fact that it separates the function of the keyboard from the letters printed on the plastic keys. If one reprograms the character map, striking the key labeled "H" might actually signal the microprocessor to send the code for the letter "o" to the machine. This means that computer keyboards can be reprogrammed for almost any arrangement of keys; although the peripheral is stamped with a QWERTY keyboard, nothing prevents one from actually using a Dvorak arrangement.

The keyboard stores the scan code in its memory while it sends an interrupt to the computer's BIOS (basic input/output system). This interrupt tells the computer to stop what it is doing and to receive the queued signals from the keyboard. The keyboard transmits the scan code to the keyboard controller, a circuit in the BIOS that forwards keyboard data to the operating system (OS). It is the keyboard controller that decides if the signal represents a system-level command or data intended for the application currently in use. In our case, the OS will send our input to an e-mail client.

But what exactly is sent between the keyboard and the BIOS, between the BIOS and the OS? While the characters appearing on the screen as you type match the characters printed on the keyboard, the machine never knows them as letters and punctuation marks. All data pass through the computer as nothing more than pulses of electricity; the machine recognizes two states: with charge and without charge. We often represent these two states with two digits, "1" and "0" respectively. (This actually reverses the historical representation: logical models of computing devices used binary logic previous to the actual construction of electromechanical or purely electrical machines.) This is also the origin of the term "bit" and by extension "byte." One bit of information is one switch, one 1 or 0, one "place"; eight bits is one byte. So our keyboards and our computers communicate via electricity, and mathematicians and logicians have developed methods for translating language (symbols) and many thought processes into binary code.

The most pertinent example for our purposes is the American Standard Code for Information Interchange (ASCII). The ASCII character set maps the 256 most common characters to binary equivalents, a unique string of eight digits. This is also the origin of the byte as a measurement of storage capacity, as one byte of information is sufficient for representing any character in the ASCII set. Thus, ASCII translates the Western alphabet into code that can be transmitted as pulses of electricity. While you type into your e-mail client, the characters are actually being stored as bit strings, as sequences of Is and Os. Let us return to our example, "Hi!"; in binary code: 101000 1101001 0100001. Standards-based e-mail clients and terminals understand all characters as bit strings, and it is bit strings that are transmitted from the keyboard to the computer.

Control

We will begin our discussion of the control unit at the motherboard, the physical component that houses the control and processing units. Most of the control unit is located in the chipset, a collection of microprocessors contained in the CPU (central processing unit). This chipset includes most essential components of the control unit and is usually visible on the motherboard. The motherboard is an intricate printed circuit board with miles of copper circuit paths called traces and numerous sockets for connecting peripherals. Together the traces constitute the bus, the metaphorical highway system that carries electricity between the processor, the memory, and the

various peripherals. Buses are the physical connections that transfer data as electronic pulses, and these connections may be a fraction of an inch in length within the CPU or several inches long when stretching from the processor to an expansion slot. As we will learn later, the bus is an important factor of a computer's overall processing speed.

Let us return to our depressed keys, our scan codes waiting in the BIOS, the component responsible for making sure that the various parts of the computer work together. Actually a bit of software code, the BIOS is stored in the chipset. Most computers employ three levels of software: the OS; the device drivers that include instructions for operating particular peripherals; and the BIOS that translates and transmits instructions and data between the OS and the drivers. The BIOS handles the transmission of the electrical signals sent between the various peripherals. So the BIOS receives our scan code, translates it to the appropriate ASCII code, and sends it to the OS.

Before we move on from the BIOS, let us review its role in the boot process, the routine followed each time the computer is powered on or restarted. Most likely, the boot process is the only time you'll have any direct experience with the BIOS, as you can watch its work on the screen as your computer starts up. During the boot process, the PC checks its components and prepares the machine to load the OS. The first step in the boot process is the power-on self-test (POST), which begins when the pressing of the power button sends the first electrical pulse to the microprocessor, telling it to reset its registers and to initiate the boot program stored in the BIOS. This boot program invokes a series of system checks, testing to see that components such as the system bus and clock are functioning properly. Next, the BIOS contacts the CMOS (complementary metal oxide semiconductor), a tiny bit of memory located on the motherboard containing basic information about the components of the computer. After that, the video adapter is checked and, if all is well, something appears on the monitor. Next is the test of the system RAM, during which the machine will seem to count each bit of memory. Then the POST continues through the various system components, checking functionality and, when appropriate, loading a device driver into the BIOS, ensuring proper and rapid functioning when that device is called by the OS. In most cases, the BIOS will then print to the screen a summary of the system configuration, which will flash by just before the OS starts. Then the POST contacts the CMOS again looking for storage devices tagged as boot devices, pieces of hardware that might contain an OS. Finally, the BIOS checks the boot sector of those devices for a master boot record that will take control of the PC for the remainder of the startup.

Another important element of the control unit is the system clock. The clock is really an oscillator, a tiny quartz crystal that vibrates when electricity is applied to it. Thus the fundamental mechanism is the same as that which drives most digital watches and electronic clocks. The vibrations of the clock cause pulses of electricity to flow through the motherboard, moving bits from place to place. Thus the rate of vibration is the shortest time in which an operation can be executed, and this rate is often called the clock speed. Many modern computers operate at speeds greater than 1 GHz (gigahertz), or more than 1 billion cycles per second. The clock is necessary because different parts of the computer, even different operations within the processor, operate at different rates. It is the job of the clock to act as a master control, to synchronize the flow of bits through the machine. The bits that compose our greeting will be transmitted according to the rhythm enforced by the oscillator, by the "ticking" of the clock. Our three bytes of information will take a minimum of twenty-four cycles – an unimaginably small fraction of a second – for each "step" between the keyboard and the modem.

Processor

The microprocessor is probably the most recognizable component of the computer. Intel's first microprocessor, the 8080 introduced in 1974, had 6,000 transistors; Intel's most recent microprocessor, the Pentium 4, includes 42 million transistors. The microprocessor, also called the central processing unit (CPU), has three basic capabilities: to perform basic mathematical operations, move data from one memory location to another, make decisions to jump from one instruction to another. At the heart of the processor is the arithmetic/logic unit (ALU), which performs very simple operations such as adding, subtracting, multiplying, dividing, and completing some basic logical operations. And each of these operations is performed on bit strings like the one that constitutes our e-mail greeting. Data and instructions are all translated

into Is and Os, pulses of electricity, and the processor is a mass of transistors, of switches, that can hold and manipulate these charges. Our e-mail greeting will flow through the CPU even though it will not be manipulated during its passing.

Storage

When we use the word "storage" we usually mean hard drives, floppy disks, or other relatively permanent but slow means of storing data. There are four major types of internal storage in modern computers, listed in order of decreasing speed and cost: registers, cache, random access memory (RAM), and hard disk drives (HDD). Registers and cache are located in the CPU or on the motherboard and come in relatively small amounts, say 512 MB (megabytes). In contrast, hard drives often have 80 GB (gigabytes) of storage for a fraction of the cost of the cache. We will learn a bit about RAM and hard drives, two locations where our greeting might be stored.

A RAM chip is an integrated circuit made of millions of pairs of transistors and capacitors that are linked to create a memory cell. Each cell can hold one bit of data. A capacitor holds a charge, and the transistor acts as a switch, allowing the control circuitry to change the state of the capacitor. These capacitors and transistors are arranged as a grid, with the columns connected by bitlines and the rows connected by wordlines. In both cases, a "line" is a microscopic strand of electrically conductive material etched into the RAM chip. When a burst of electricity is transmitted through a bitline, every closed transistor, each switch that is "on", charges its capacitor. The combination of bitlines and wordlines defines an address for each switch and capacitor. This also accounts for why this type of memory is called random access: you can access any memory cell directly if you know its address. Most RAM is volatile, or dynamic, meaning that the capacitors are always leaking their charge and thus must be refreshed constantly. RAM is inserted directly into the motherboard, most often in the form of a dual in-line memory module (DIMM) that holds up to 128 MB of memory.

A hard disk drive is one of the few computer components that is mechanical as well as electronic. Hard drives contain a collection of platters and heads arranged so that the gaps between them are smaller than a human hair; thus hard drives are very sensitive to dust and jostling. The round platters are made of aluminum or glass coated with a magnetic material. The surface of the platter is divided two ways, as a series of concentric circles and a collection of wedges, and the resulting pieces are called sectors. Each sector contains a set number of bytes, usually 256 or 512. A single hard drive may contain as many as eight platters, and platters are stacked on a central motorized spindle that spins the platters so that the disk heads can read and write data. The heads write data by aligning the magnetic particles on the surface of the platter; the heads read data by detecting the magnetic charge of particles that have already been aligned. Again, all data is stored as a charge, as a 0 or a 1.

How does the drive manage those eight disks, keeping track of where files are stored and not inadvertently overwriting important data? Every Windows hard drive has a virtual file allocation table (VFAT), a sort of database of addresses for files. The VFAT records the sectors holding any saved file and offers empty sectors for writing. When you delete a file, the hard drive doesn't actually modify the data sectors. Rather, it edits the VFAT, removing the entry for that file. Thus the file is invisible but not erased. Depending on our e-mail client, our greeting has either been saved to RAM or to the hard drive while it waits to be sent.

Output

Output devices include monitors, printers, modems, Ethernet cards, and speakers. We will forgo a discussion of monitors in favor of a brief account of modems, the hardware device through which most of us were introduced to the Internet. The modem is particularly interesting because it is a point of union, of conversion between the digital and the analogue, between new and (relatively) old technologies of communication. The modem gets its name from its primary tasks: modulating and demodulating. As we continue to discover, the computer operates exclusively with Is and Os, with a digital code. By contrast, phone lines work with an analogue signal, an electronic current with variable frequency and strength. It is the job of the modem to demodulate data from a phone line, to convert the wave to a series of Is and Os, and to modulate the data generated by the computer, to convert the Is and Os to a continuous wave. This process accounts in part for

the limited speed of modems. The analogue signal in the phone line cannot change frequencies as quickly as a computer can send bits of information, yet the signal needs to change for every bit sent. Clever engineers devised techniques for avoiding this bottleneck, including defining four frequencies and four associated combinations of bits – 00, 11, 01, 10 – and experimenting with variations in amplitudes, but the physical medium of the telephone line still poses material constraints on the speed of transmissions. This means that our e-mail greeting, stored temporarily in RAM or perhaps saved to the hard drive, and displayed on the screen, will need to be translated to analogue before being sent to a friend.

Review

Let us review what just happened. We press a few keys on our keyboards, and each strike of a key completes a circuit mapped on the key matrix. Comparing the completed circuit to the key matrix, the keyboard's microprocessor selects a scan code for each character and transmits it to the computer's BIOS for decoding. After deciding that the keyboard data are bound for our e-mail client, the BIOS transmits the ASCII bit string for each character. These bit strings are stored temporarily in RAM or, if the e-mail has been saved, more permanently on our hard drive. When we send our e-mail, the ASCII bit strings will be sent to the modem, which will modulate the data before transferring it to the phone line.

Ending our discussion of how a computer works with the modem emphasizes the significance of the peripheral as a transitional device. Modems operate at the boundary between analogue and digital technologies, between nineteenth- and twentieth-century technologies of communication, between transmission and simulation. At multiple levels the machine is involved in sophisticated translations of information, as is evidenced by the fact that you really do not need to understand how a computer works to use a computer. A great accomplishment of software developers is the extent to which we can "talk" to computers using our own language, the extent to which the code, the Is and Os, the electric pulses are hidden. Many pioneers in computing technologies aspired to build machines capable of sophisticated symbol manipulation, as they understood that such skill – basically the ability to use language – is essential to human intelligence. Yet in simulating this capacity they added several layers of symbols, several additional levels of abstraction: printed letter to ASCII codes to a series of pulses.

How the Computer Functions

To begin, I will restate my guiding distinction: "work" describes the electronic processing performed by the machine; "function" refers to the theoretical and practical relations between the computer and its human operators. To explore this distinction, I focus on the system clock and the keyboard, but my evidence could have included several other components, most notably computer memory (Douwe 2000; Locke 2000; Williams 1997). I discuss the history of the adoption of and adaptation to these technologies, looking at how the computer as artifact reproduces metaphors of mind and incorporates bodies into mechanized routines. Relating the system clock and the computer keyboard to previous implementations of similar technologies, I explore motivations and rationales for the choices behind these particular mechanisms. Pioneers of digital technologies often thought their designs radical and their machines revolutionary, yet these inventions incorporate conventions and habits embedded in previous technologies. In this sense, digital technologies can be seen as extensions of numerous historical trends.

System clock

Let us begin by returning to John von Neumann, focusing this time on the biological metaphors that frame his understanding of his radical architecture. Trained in logic and mathematics, von Neumann's interest in information processing was strongly influenced by the work of Warren McCulloch and Walter Pitts, in particular the article "A Logical Calculus of the Ideas Immanent in Nervous Activity" (1943). Although he acknowledged that the McCulloch-Pitts model oversimplified neural activity, von Neumann was intrigued by the parallels proposed between man-made and organic systems (Aspray 1990: 180–1). In 1944 von Neumann joined the likes of McCulloch, Pitts, and Norbert Wiener to form the Teleological Society, a research group "devoted on the one hand to the study of how purpose is realized in human and animal conduct and on the

other hand how purpose can be imitated by mechanical and electrical means" (Aspray 1990: 315). Although the Teleological Society disbanded, many of its founding members initiated the Macy Conferences, probably the most famous early discussions of artificial intelligence.

Looking at von Neumann's *First Draft of a Report on the EDVAC* (1945), the first publication describing the stored-program architecture, one notices his analogies to organic systems. For instance, in this report the storage unit of a computer is first called "memory", and other logical parts are called "organs." Von Neumann understood organic systems to be highly complex, to employ parallel processing and to be adept at switching between analogue and digital modes of computation, and he planned to simulate the organic by implementing a few key principles – process hierarchies, serialization, regulation, and automation. Along with others in his field, von Neumann rationalized the organic and made discrete the continuous in an effort to mimic the intellect of man. Aware that current technology couldn't accomplish the sophistication of organic systems, von Neumann intended to design an architecture that could evolve sufficient complexity. He modeled the workings of his computer after the brain, hoping that the machine would function like – or at least with – man. In an attempt to replicate the functioning of the human mind, von Neumann built a model of the brain that worked in ways quite different than its organic precursor. The mechanics of the computer introduce several technologies foreign to the mind yet not so unfamiliar to the history of its acculturation and regulation.

Our brief investigation of these mechanics will focus on the system clock, a device central both to the work and to the function of the computer. Historians of technology often point to the clock as a mechanism central to the modernization of the West. In his history of technology *Technics and Civilization*, Lewis Mumford (1934: 14) writes: "The clock, not the steam-engine, is the key-machine of the modern industrial age. For every phase of its development the clock is both the outstanding fact and the typical symbol of the machine: even today no other machine is so ubiquitous." Mumford published his book a decade before von Neumann designed his stored-program architecture, the first computer architecture regulated by a control unit called a "clock." In his *Report*, von Neumann describes the electromechanical devices used to control computers previous to the EDVAC and then proposes an alternative for his new machine: "Alternatively, [computing instruments] may have their timing impressed by a fixed clock, which provides certain stimuli that are necessary for its functioning at definite periodically recurrent moments" (1945: 24). Even at this early stage von Neumann suggested that this "clock" might actually be an electronic oscillator controlled by a crystal. Two years later, von Neumann, Goldstine, and Burks described dynamic circuits, regulated by electric pulses emitted from a central source called a "clock": "Since the timing of the entire computer is governed by a single pulse source, the computer circuits will be said to operate as a synchronized system" (Burks et al. 1947: 131). Thus, the clock is essential to the proper working of the computer, and in this section I will argue that it is key to understanding the functioning of the computer as well.

The system clock lacks most mechanisms associated with clocks: it has neither gears nor hands, neither a face nor an audible signal to mark the minutes. Many advanced computer science books refer initially to a clock but later drop the metaphor and describe the oscillator, the tiny crystal whose vibrations regulate the flow of electricity through the motherboard. The etymology of "clock" relates our modern machines to medieval bells, in particular to bells that were struck to mark the hours *(OED 1989)*. Samuel Macey extends this relation, figuring the clock as a sort of automaton: "the earliest mechanical clocks may be considered automated versions of the keeper of the bell or *Glocke*, which gave us the term *clock* in the first place" (Macey 1989: 1 1). Gerhard Dohrn-van Rossum, historian and author *of History of the Hour* (1996), traces the history of man's relation to time-keeping devices, focusing less on the technology underlying the machines and more on man's evolving relation to the device. I draw on Dohrn-van Rossum's account, foregrounding his argument that time-keeping devices have functioned to synchronize, regulate, and abstract time.

Well before mechanical clocks were sufficiently precise to reliably mark the hours, bells divided the day and synchronized the activities of a populace. Mumford points to the monasteries as the origin of modern notions of time-keeping, arguing that the Benedictines "helped to give human enterprise the regular collective beat and rhythm of the machine; for the clock is not merely a means of keeping track of the hours, but of synchronizing the actions of men" (1934: 14). Church bells also marked the hours outside the monastery, alerting laypersons to the hours for prayers

and for mass, and by the fifteenth century many municipalities had acquired communal bells to signal secular meetings and events. Through variations in tone, duration, and number of rings, churches and municipalities developed elaborate codes to signal everything from mass to changing of the guards, from noontime break to opening of the courts. In both contexts – religious and secular – these bells announced hours that were irregular according to modern standards but nevertheless succeeded in improving the frequency with which monks and citizens were "on time" for various functions. In both contexts, the bells were functional well before devices for keeping time really worked.

After bells, civic officials developed several time-keeping technologies to regulate everything from the marketplace to the postal system. Beginning in the thirteenth century, officials regulated the market with clocks, setting times at which particular merchants could do business and hours during which locals could shop. Previous to the fifteenth century, universities varied the time and duration of lectures according to content. By the fifteenth century, particularly in secondary schools, days were divided into discrete blocks during which specific subjects were taught, and these hours were measured by bells and sandglasses. In addition to measuring and dividing the day, timekeeping devices increasingly aided the rationalizing of space. Between the fifteenth and eighteenth centuries, postal and courier services increasingly abstracted distance by measuring it in units of time. Delivery times once measured in days were soon measured in hours, and soon promptness became the measure of quality: the normative codes "should thus be understood as concepts for systematically organized transportation and communication links, which, if we trace them over a longer period, reveal the growing importance of the techniques of time measurement and time control" (Dohrn-van Rossum 1996: 343). Even before railroads, the technology most associated with the strict enforcement of standardized time, the markets, the schools, and the postal systems were regulated by increasingly abstract notions of time (and space) (Dohrn-van Rossum 1996).

It was not until the mid-seventeenth century that these functional time-keeping devices developed into working clocks. In 1656, Christiaan Huygens patented a pendulum clock that was accurate to within one minute a day, and by the late seventeenth century he had developed a balance wheel and spring assembly that could operate a pocket watch with an accuracy of 10 minutes per day. A substantial improvement over previous time-keeping mechanisms, the pendulum was subject to inaccuracies due to changes in temperature, air pressure, and gravity. Some 200 years later, quartz crystals replaced pendulums in the 1920s, further enhancing the accuracy of clocks. Using an electric circuit to cause a crystal to vibrate, quartz clocks shed the gears and escarpments – the tangible mechanisms – that often led to variations in time. Just thirty some years after the quartz clock, an atomic clock that was accurate to one second in 2,000 years appeared in 1955. The original definition of "second" was 1/86,400 of a mean solar day; in 1967 the natural frequency of the cesium atom – 9,192,631,770 oscillations of the atom's resonant frequency – was declared the international unit of time. Thus, by the late twentieth century measures of time had been abstracted to the point that they seemed to have little relation to the daily rhythms – solar or social – of man.

Modern system clocks utilize quartz crystals, the same technology that runs many digital watches, and von Neumann first suggested using these oscillators in his 1945 *Report.* Following a brief overview of his entire architecture, von Neumann began his section titled "Elements, Synchronism Neuron Analogy" with a description of the control unit (1945: 23). He argued that all digital computing devices must have control "elements" with discrete equilibrium and that a synchronous "element" – a system clock – is the preferable control mechanism. Going to the trouble to define "element" as a synchronous device with discrete equilibrium, he proceeded to a biological analogy: "It is worth mentioning, that the neurons of the higher animals are definitely elements in the above sense. They have all-or-none character, that is two states: Quiescent and excited" (1945: 24). Drawing on the work of MacCulloch and Pitts, von Neumann acknowledged that neurons were probably asynchronous but proposed that their behavior be simulated using synchronous systems (ibid.). Thus, von Neumann chose to implement a system clock because he was attempting to mimic the functioning of the human brain. Unable to build a machine that *worked* like a brain, he imagined constructing a computer that (apparently) functioned like a mind: "Von Neumann, especially, began to realize that what they were talking about was a *general-purpose machine*, one that was by its nature particularly well suited to function as an extension of the human mind" (Rheingold 2000: 86). Essential to this machine was a clock, a

metronome that made regular, synchronous, and rational this model of the mind. Attempting to extend the capacities of the human mind, von Neumann's design actually recalled technologies previously used to control it. Paradoxically, von Neumann's architecture is at once flexible and rigid, is both an extension of and a constraint on our conceptual, cultural, and practical understanding of our minds.

QWERTY keyboard

In this section we will investigate the QWERTY keyboard, using a distinction between "machine" and "tool" to frame our discussion. Karl Marx in *Capital* (1867) and later Lewis Mumford in *Technics and Civilization* (1934) define these terms in reference to the history of labor and production. Without referring directly to Marx or Mumford, J. C. R. Licklider and Douglas Engelbart, early designers of computer interface and input devices, later defined a plan for man-computer symbiosis that echoes this tradition. Marx writes that the origin or impetus of movement is the essential difference between a tool and a machine: with tools, movement originates in the laborer; with machines, movement issues from the mechanism itself (1867: 409). Working in an environment with machines requires education, that the body be trained to move along with the uniform and relentless motion of the mechanism (1867: 408). Conversely, the tool allows the laborer freedom of movement, an opportunity for motion independent of the mechanism. Working from Marx, Mumford associates tools with flexibility and machines with specialization. The essential difference, according to Mumford, is the degree of independence of operation. To summarize, a man works with a tool as an extension of his own body (and perhaps his own mind); in contrast, a machine employs a man as an extension of its own mechanism.

Licklider (1960) essentially fears that computers are becoming machines rather than tools. Describing computers built previous to 1960 as "semi-automatic systems", he laments the trend toward automation at the expense of augmentation. He sees man incorporated into the system rather than being extended by it: "In some instances, particularly in large computer-centered information and control systems, the human operators are responsible mainly for functions that it proved infeasible to automate" (1960: 60). The human appears as a compromise required by the current state of technology, as a part that will be replaced once the memory is faster or the arithmetic unit more efficient. However, computers might act as extensions of humans, as devices for the augmentation of the intellect, as components in human-machine "symbiotic" systems. His proposed symbiosis is peculiar because he wants to introduce intuition and "trial-and-error procedures" into the system; as later expressed by Douglas Engelbart (1963), "we refer to a way of life in an integrated domain where hunches, cut-and-try, intangibles, and the human 'feel for a situation' usefully coexist with powerful concepts, streamlined terminology and notation, sophisticated methods, and high-powered electronic aids" (1963: 72). Licklider and Engelbart differ from von Neumann in that they do not want the computer to imitate the intellect but rather to enhance it. For both Licklider and Engelbart, organization is a component of intelligence, a function that could perhaps be better handled by a computer. But the essence of intelligence is a sort of synergy, a cooperative interplay between the human capabilities of organization and intuition, analysis and synthesis. While the language of progress and evolution employed by Licklider and Engelbart may be troubling, their visions of computers as extensions rather than imitations of man, as tools rather than machines, was revolutionary and remains relevant today.

The computer keyboard exemplifies this tug-of-war between machine and tool. Most know that the modern computer keyboard continues the tradition of the typewriter and its less-than-intuitive arrangement of keys. Often referred to as the "QWERTY" keyboard, this arrangement of keys was devised by Christopher Sholes, and the first Remington typewriters sold in 1873 implemented his design. For decades after the first Remington, manufacturers experimented with alternative arrangements, including a circular layout and an arrangement that featured keys in alphabetical order. Probably the most famous redesign was the Dvorak Simplified Keyboard, patented in 1936. August Dvorak claimed that his keyboard led to increased speed, reduced fatigue, and more rapid learning. There is much dispute about the data used to support these claims (Liebowitz and Margolis 1990), and more recent experimental evidence suggests that the Dvorak keyboard leads to only a 10 percent improvement in speed (Norman 1990: 147). Yet the QWERTY keyboard remains a standard even though there exists an empirically superior option. Don Norman argues that such a small improvement does not warrant the institutional costs of

conversion: "millions of people would have to learn a new style of typing. Millions of typewriters would have to be changed. The severe constraints of existing practice prevent change, even where the change would be an improvement" (1990: 150). Others add to this interpretation, citing the Dvorak keyboard as an example of the failure of the market economy (David 1985). Rather than entering this debate, I suggest that we accept parts of each argument – the Dvorak keyboard is not a marvelous improvement over the QWERTY arrangement, and the institutional cost of conversion is great – and then investigate the process through which QWERTY gained form and acceptance.

While the QWERTY arrangement might not be the ideal arrangement of keys independent of mechanism used to print those keys, the arrangement was the most functional of Sholes's designs. The design problem faced by Sholes was that of overlapping rods: if a typist struck in sequence two adjacent keys, their rods would intersect and jam. Then the operator would have to stop typing and dislodge the mechanism, losing valuable time. Sholes's keyboard minimized the likelihood of jamming by separating keys for letters that frequently appear adjacent to one another in common words. The arrangement aims to achieve maximum speed through maximum synchronization of operator and machine. It is often stated that Sholes designed the keyboard to slow down operators of the typewriter; in fact, Sholes designed the keyboard so as to better coordinate the actions of the machine and its operator, not just to slow the operator but to develop a sequence and a rhythm that was manageable by both human and machine. Bliven (1954) comments on the sound of a room full of typists, on the particular rhythm distinct to adept typists. Sholes's design of the machine was the engineering of this rhythm, a reorganization of the interface of the typewriter to facilitate the most efficient use of the mechanism.

Today we assume that good typists use all fingers, but this technique was not common until fifteen years after the typewriter went to market. In fact, there was much debate surrounding the feasibility of teaching this skill. Early typists used the "hunt and peck" method with remarkable success. In 1882, Mrs L. V. Longley issued a pamphlet proposing a method of typing that used all ten fingers. A typing instructor in Cincinnati, Mrs Longley met with great opposition in the trade press. Even five years after the publication of her pamphlet, trade magazines carried stories arguing the impracticality of her proposal, citing a lack of strength and dexterity in some fingers. Five years later, Frank McGurrin, a court stenographer from St Louis, again proposed a ten-finger method of typing. He added to his technique the skill of "touch typing", the ability to type without looking at the keyboard. He also met with resistance but silenced his adversaries when, in 1888, he won a typing contest using his new method. McGurrin used a Remington typewriter, and his victory led to the adoption of two standards: the QWERTY keyboard and the touch-typing method (Norman 1990: 147). Following McGurrin's victory, the same publication that berated Mrs Longley endorsed touch-typing. It was accepted that typewriters might be more than twice as fast as writing by hand, and typists were retrained to use all fingers.

In the early twentieth century, typewriter companies employed championship typists to compete in national competitions. These racers were given a racing typewriter and a salary for practicing their skills eight hours a day. Racing typists netted 120 error-free words per minute in front of crowds of businessmen at the various trade shows in Madison Square Garden. Once most speed typists mastered the art of touch-typing, other inefficiencies in the process were targeted. Charles Smith, the coach of the famous Underwood Speed Training Group, developed the "right-hand carriage draw", a new method of exchanging paper that saved as much as a full second. This method required the typist to grab the roller knob and the filled sheet with one hand while grabbing a fresh sheet from a prepared stack of papers and inserting it without ever looking away from the copy. All of this in one-third of a second (Bliven 1954: 125)! Techniques introduced in the racing circuit gradually infiltrated the business world, producing typists with elegant and precise routines to synchronize their movements with the machine. Rhythm was key: "Above all, the mark of good typing is a continuous, even rhythm. An expert keeps going" (Bliven 1954: 143). Thus the touch-typing system prescribed routines for striking the keys and for manipulating the impacted media, thereby rendering the typists extensions of their typewriters, parts of their machines.

In the late nineteenth century, business required improvements in technologies of communication and record-keeping, and commercial culture embraced the typewriter and its promise of

increased speed and accuracy of transcription. But other cultures of use emerged as well. George Flanagan writes: "the idea of Art in typewriting has been sadly ignored" (1938: vi). Calling the machine "mechanically perfect" and typists "marvel-ous[ly] skilled", he writes a treatise in hopes of diverting attention from efficiency towards aesthetics. Flanagan recommends harmony and balance through the judicious arrangement of space and compactness (1938: 2). Utility should not be achieved at the expense of elegance; speed can be sacrificed for ornament. Inserting typewriting into the tradition of printing, Flanagan advises the typists to plan the layout of the page just as one plans the content of the paragraphs. He provides sample layouts – spacing, alignment, sizes for paragraphs – and includes formulas – combinations and sequences of keystrokes – for generating ornamental drop caps and borders. Although he recognizes the pressures of an office and the primacy of speed, Flanagan argues that typists should not ignore design. They might even profit from it: "Your work will become an endless romance, and we even suspect that your pay envelopes will grow fatter" (1938: vii). Seeing the typist as a craftsman rather than a laborer, Flanagan reshapes the machine as a tool. Although entertaining, the point isn't the aesthetics of ASCII art; rather, it is the revision of what had become a mechanistic relationship between a human and a machine. The idea of typing contests and of ornamental typing might seem equally trite today, but their historical value lies in the relationships they posit between typist and typewriter, between operator and tool or machine.

Although most computer keyboards follow Sholes's design, Engelbart experimented with many different input devices that drew on various traditions of symbol manipulation. Most interesting in this context is the chord keyset, a group of five piano-like keys that were operated by one hand. The keys were not printed with any characters, and operators used single keys and combinations of keys to produce up to 32 different codes, each of which represented a different character. Drawing on technologies developed by telegraphers, Engelbart expected that computer users would be trained in this code and would eventually surpass the speeds of typists using QWERTY keyboards. The chord keyset would also allow users to manipulate a keyset and a mouse at the same time without ever looking at their hands. Engelbart imagined a reincarnation of the Remington racer who could change paper without ever losing her place in the copy. His input devices would become extensions of the hands, and users would manipulate them without the aid of the other senses. Yet such facility would require learning another language, a new code, an additional symbol set that in its abstraction matched closely the binary set used by the computer itself. The boundary between man and machine seemed permeable, but man seemed to be the one crossing the line, the one adopting the characteristics of the other.

Engelbart argued in general terms that users would need to be trained to use computers, and the retraining necessary to use the chord keyset was just one step in that process. Of course, the typist's keyboard was eventually chosen over the telegrapher's, largely due to the anticipated institutional costs of retraining, and Engelbart's demo now seems rather quaint. But the choice to adopt the QWERTY keyboard demonstrates a few interesting points. First, we no longer recognize the extent to which typing – even the hunt-and-peck method – requires training; the typist's keyboard seems somehow natural. Second, while we resist new technologies that require abstract, mechanistic motion, we willfully embrace old ones that require equally abstract and mechanistic routines. If the QWERTY keyset casts the computer as machine, perhaps it's difficult to argue that the chord keyset does otherwise. The latter requires complicated and coordinated movements and an internalized understanding of an abstract code. Yet Engelbart, seeking augmentation rather than automation, argued that this degree of training is necessary before the computer may function as a tool. Engelbart's notions of training, and the boundaries between the training of man and the training of the computer, deserve further exploration. Returning to Marx and Mumford, we must work to cast the computer as tool rather than as machine.

Conclusion

Despite the apparent contradictions in Engelbart's efforts to build input devices that would augment the human intellect, he and Licklider provide inspiration for, if not demonstration of, what the computer might someday be. Engelbart in his 1961 report "Program on Human Effectiveness" describes his plan for the augmentation of the individual problem solver. His objective is to "develop means for making humans maximally effective as comprehending solvers of problems", and he plans to do this by inserting computers into various levels of our hierarchy

of problem-solving tools. Here again he returns to issues of training, proposing that humans be conditioned both to use his input devices and to understand the logical operations that make the computer work. Engelbart wants to build tools even if he must first build machines; he wants humans to train computers, but for now he must settle for computers that can train humans. Today, children use computers from such a young age that manipulating a mouse seems "intuitive" and programming languages seem "natural." Half of Engelbart's program has been accomplished: we have been trained to use the computer.

Even in 1960 Licklider was troubled by the extent to which scientists had been trained to use the computer. Disappointed that the computer was being used primarily to solve "preformulated problems" with "predetermined procedures" – "It is often said that programming for a computing machine forces one to think clearly, that it disciplines the thought process. If the user can think his problem through in advance, symbiotic association with a computing machine is not necessary" (1960: 61) – Licklider proposed that computers be employed to ask questions, not just to provide answers. Given the limits to technology, the first attempts at symbiotic systems will rely on the human operators to set the goals, formulate hypotheses, devise models, and so on (1960: 64). The operators will evaluate the output, judge the contribution of the machine, and fill in the gaps when the machine lacks the proper routine for handling some problem. Although Licklider imagines much more, his first iteration of the symbiotic system is an advanced information-handling machine. Consistent with his colleague, Engelbart also framed the problem of symbiosis as one of information management: a primary objective for his H-LAM/T project was "to find the factors that limit the effectiveness of the individual's basic information-handling capabilities in meeting the various needs of society for problem solving in the most general sense" (1963: 73). In practice, the symbiotic machine became a problem-solving rather than a problem-posing device. For the most part, that is how the computer continues to function. Licklider's dream remains largely unfulfilled. Perhaps transforming the computer from machine to tool, from a device that automates mundane mental tasks to one that augments critical and creative thought, is the task now facing computing humanists.

See also chapter 29: Speculative Computing.

References for Further Reading

Aspray, William (1990). *John von Neumann and the Origins of Modern Computing*. Cambridge, MA: MIT Press.

Bliven, Bruce (1954). *The Wonderful Writing Machine*. New York: Random House.

Burks, Arthur, Herman H. Goldstine, and John von Neumann (1947). *Preliminary Discussion of the Logical Design of an Electronic Computing Instrument*, 2nd edn. In William Aspray and Arthur Burks (eds.), *The Papers of John Von Neumann on Computing and Computer Theory (1986)*. Charles Babbage Institute Reprint Series for the History of Computing, vol. 12 (pp. 97–144). Cambridge, MA: MIT Press.

Campbell-Kelly, Martin, and William Aspray (1996). *Computer: A History of the Information Machine*. New York: Basic Books.

David, Paul A. (1985). *Clio and the Economics of QWERTY*. *American Economic Review* 75, 2: 332–7.

Dohrn-van Rossum, Gerhard (1996). *History of the Hour: Clocks and Modern Temporal Orders*, tr. Thomas Dunlap. Chicago: University of Chicago Press.

Douwe, Draaisma (2000). *Metaphors of Memory*, tr. Paul Vincent. Cambridge: Cambridge University Press.

Engelbart, Douglas C. (1961). *Proposal for Participation in the Program on Human Effectiveness*. Accessed October 1, 2002. At http://sloan.stanford.edu/mousesite/EngelbartPapers/B4_F12_HuEff3.html.

Engelbart, Douglas C. (1963). *A Conceptual Framework for the Augmentation of Man's Intellect*. In P. W. Howerton and D. C. Weeks (eds.), *The Augmentation of Man's Intellect by Machine*. In Paul A. Mayer (ed.), *Computer Media and Communication (1999)*. New York: Oxford University Press.

Engelbart, Douglas C. (1968). *Demo of the Augment Research Center*. Fall Joint Computer Conference. Accessed October 1, 2002. At http://sloan.stanford.edu/mousesite/1968Demo.html.

Englander, Irv (1996). *The Architecture of Computer Hardware and Systems Software*. New York: John Wiley.

Flanagan, George A. (1938). *A Treatise on Ornamental Typewriting*. New York: Gregg Publishing.

HowStuffWorks (2002). http://www.howstuffworks.com, by Marshall Brain. HowStuffWorks, Inc..

Kozierok, Charles M. (2001). *The PC Guide*. At www.pcguide.com.

Licklider, J. C. R. (1960). *Man-Computer Symbiosis: IRE Transactions on Human Factors in Electronics*. In Paul A. Mayer (ed.), *Computer Media and Communication (1999)*. New York: Oxford University Press.

Liebowitz, S. J. and Stephen E. Margolis (1990). *The Fable of the Keys*. *Journal of Law and Economics* 33, 1: 1–26.

Locke, Chris (2000). *Digital Memory and the Problem of Forgetting*. In Susannah Radstone (ed.), *Memory and Methodology* (pp. 25–36). New York: Berg.

McCullough, Warren and Walter Pitts (1943). *A Logical Calculus of the Ideas Immanent in Nervous Activity*. *Bulletin of Mathematical Biophysics*, 5: 115–33.

Macey, Samuel L. (1989). *The Dynamics of Progress: Time, Method, and Measure*. Athens, GA: University of Georgia Press.

Marx, Karl (1867). *Capital, Volume One*. In Robert C. Tucker (ed.), *The Marx-Engels Reader*, 2nd edn. (1978). New York: W W Norton.

Mumford, Lewis (1934). *Technics and Civilization*. New York: Harcourt, Brace and Company.

Norman, Don (1990). *The Design of Everyday Things*. New York: Doubleday.

OED (1989). *"clock, n." Oxford English Dictionary*, ed. J. A. Simpson and E. S. C. Weiner, 2nd edn. Oxford: Clarendon Press. Electronic Text Center, University of Virginia. Accessed October 1, 2002. At http://etext.virginia.edu/oed.html.

Petzold, Charles (2000). *Code*. Redmond, WA: Microsoft Press.

Rheingold, Howard (2000). *Tools for Thought*. Cambridge, MA: MIT Press.

von Neumann, John (1945). *First Draft of a Report on the EDVAC*. In William Aspray and Arthur Burks (eds.), *The Papers of John Von Neumann on Computing and Computer Theory (1986)*. Charles Babbage Institute Reprint Series for the History of Computing, vol. 12 (pp. 17–82). Cambridge, MA: MIT Press.

White, Ron (2001). *How Computers Work*, 6th edn. Indianapolis, IN: Que Publishers.

Williams, Michael R. (1997). *A History of Computing Technology*, 2nd edn. Los Alamitos, CA: IEEE Computer Society Press.