ASCII and Binary Representations of Keyboarding

HAVE YOU EVER WONDERED how the keys on your keyboard work with the computer, as you type? How does the computer know what each letter, number, or function on each key represents? How does the computer know that you have pressed the space bar, the delete key, the F12 key, the letter N, or the number 5? How does the computer know you are typing in a completely different language other than English? This lesson identifies what the standards are for key representations on a computer keyboard are and explains how each key press is interpreted by the computer.

Objectives:

1. Interpret the ASCII representations of computer keyboards.
2. Interpret the binary math representations of computer keyboards.
3. Explain how the new Unicode works for computer keyboards.

Key Terms:

abacus
Analytical Engine
ASCII
Babbage
character
Dvorak
glyph
keyboard layouts
Leibnitz
Lovelace
number crunching
QWERTY
Turing
unicode
UTF-8
UTF-16
UTF-32

ASCII and Binary Keyboard Representations

ASCII KEYBOARD REPRESENTATIONS

The ASCII code for keyboards has been around since the 1960s when teletypewriting was popular for keying information into large computer systems. There had to be a way in which each key press on the keyboard or teletypewriter was recognized as unique from the other keys. **ASCII**, which stands for the American Standard Code for Information Interchange, was developed to standardize how each **character**, or symbol, letter, number on the keys were represented in computer language, binary code. It was decided that seven bits (1's and 0's) would represent each character, thus the invention of the ASCII chart, with the first letter, capital A being represented as decimal number 65, converted to binary code 1000001.

0	0011 0000	I	0100 1001	b	0110 0010	v	0111 0110
1	0011 0001	J	0100 1010	c	0110 0011	w	0111 0111
2	0011 0010	K	0100 1011	d	0110 0100	x	0111 1000
3	0011 0011	L	0100 1100	e	0110 0101	y	0111 1001
4	0011 0100	M	0100 1101	f	0110 0110	z	0111 1010
5	0011 0101	N	0100 1110	g	0110 0110		
6	0011 0110	O	0100 1111	h	0110 1000	:	0011 1010
7	0011 0110	P	0101 0000	i	0110 1001	;	0011 1011
8	0011 1000	Q	0101 0001	j	0110 1010	?	0011 1111
9	0011 1001	R	0101 0010	k	0110 1011	.	0010 1110
		S	0101 0011	l	0110 1100	'	0010 1111
		T	0101 0100	m	0110 1101	!	0010 0001
A	0100 0001	U	0101 0101	n	0110 1110	,	0010 1100
B	0100 0010	V	0101 0110	o	0110 1111	"	0010 0010
C	0100 0011	W	0101 0111	p	0111 0000	(	0010 1000
D	0100 0100	X	0101 1000	q	0111 0001	)	0010 1001
E	0100 0101	Y	0101 1001	r	0111 0010	space	0010 0000
F	0100 0110	Z	0101 1010	s	0111 0011		
G	0100 0111			t	0111 0100		
H	0100 1000	a	0110 0001	u	0111 0101		

FIGURE 1. ASCII code — character to binary.

Early ASCII Charts

Earlier ASCII charts used seven-bit binary codes for each character on the keyboard, later adding a 8th bit leading zero to conform with today's 8-bits per character conversions for computer processing. Binary codes representing each character are necessary in order for the computer to be able to translate to 'hard electrical pulses:' a 1-bit is the presence of a pulse pattern, a 0-bit is the absence of a pulse pattern. This allows a string of bits to be processed by the keyboard controllers as well as the CPU microprocessors. Early ASCII charts showcased 0 to 127 possible characters, calling it the US-ASCII Chart. In looking at ASCII Conversion Charts, you will notice that there is a column for the character, a column for the decimal representation of the character abbreviation or **glyph** (a graphic character such as a letter, number or symbol on the key), a column for the hexadecimal and/or octal numeric representation of the character and a column for the binary code representation of the character. Why is that? It is so computer programmers who write code for computer applications can be able to use a two-digit hexadecimal number to represent a byte or eight binary digits. Hexadecimal numerals include 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, and also A, B, C, D, E, F, for sixteen numerals. The octal system numerals include 0, 1, 2, 3, 4, 5, 6, and 7 for eight numerals, and the binary system makes up 0, and 1, a byte representing 8 possible 0, 1 combination. It is much easier on the human eyes to code using hexadecimal values than binary code! However, chip technology and microprocessors don't care about pulsing a string of 0's and 1's! The converted key representations are hard-coded or pre-burned into a ROM (read only memory) chip character map or look-up table that contains all of the bit codes for each. Each word typed in the computer can be converted to a string of 1's and 0's using the US-ASCII chart. For example, the name Mike in binary code would be 1001101 1101001 1101011 1100101, not forgetting to use the codes for capital M and lowercase i, k, and e.

US-ASCII Chart with 0–127 Character Representation

The US-ASCII Chart can be viewed at https://en.wikibooks.org/wiki/A-level_Computing/AQA/Paper_2/Fundamentals_of_data_representation/ASCII_and_unicode

BINARY MATH KEYBOARD CODES

Early reports of the binary language of 1's and 0's contributes Gotfried Leibnitz as having discovered this system. Leibnitz is quoted as saying "one is enough derived from nothing." His manifestation of the binary code was quoted by one mathematician Simon Marquis de LaPlace as saying Leibnitz "imagined that Unity represents God and zero the Void." The binary system went on to become one that is easy to derive math calculations from and report as pulse patterns representing what we type on the keyboard.

FIGURE 2. Gotfried Leibnitz — Father of Binary Math.

Number Crunching

We learn to do math calculations using the decimal system of 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. It was discovered that binary math can derive or provide answers in binary code from the decimal number or base 10 system. This is called division by 2. An example given is when dividing the number 125 by 2. A binary number representation of the number 125 is produced as the remainder after each division until there is nothing left to divide. So, 125 divided by 2 equals 62 with a remainder of 1; 62 divided by 2 is 31 with a reminder of 0; 31 divided by 2 is 15 with a remainder of 1; 15 divided by 2 is 7 with a remainder of 1; 7 divided by 2 is 3 with a remainder of 1; 3 divided by 2 is 1 with a remainder of 1; and 1 divided by 2 is 0 with a remainder of 1. This is how decimal numbers are converted to binary numbers in the computer's ALU (arithmetic logic units) and stored in memory for further calculations. The speed of computers has yielded this process of binary math calculations in the computer as fast computing or **number crunching**.

125	
62	**1**
31	**0**
15	**1**
7	**1**
3	**1**
1	**1**
0	**1**

The Analytical Engine

This is what early inventions of computers did all day long, but slowly, crunch numbers. One of the earliest computing devices to preclude today's computers was the Analytical Engine created by Charles Babbage, a scientist and mathematician. He designed the computing machine in 1800s to meet the growing need of the United States to be able to count the increase in population. Babbage envisioned his Analytical Engine as being able to use already existing loom card technology to input the instructions and data. Going back into the early history of computers, Ada Lovelace has been credited as providing the first machine language algorithm for Charles Babbage's early computing machine, using binary code. Babbage's computer was known as the Analytical Engine. This **Analytical Engine** was designed to consist of four parts: the mill or CPU which calculated units, the store or memory, and input and output.

Ada Lovelace

Ada Lovelace became close friends with Charles Babbage, especially interested in his Analytical Engine. Combined with her penchant for mathematical operations, Lovelace became especially involved with how the information would be coded into patterns for Babbage's computer engine. Input of information during this creation had been likened to using the Jacquard loom card technology that produced patterns on fabrics. These were the first thinking of pattern formation for inputting data to the computers. Ada became profusely involved with Charles Babbage's Analytical Engine, leaving behind copious notes on how his computer could be programmed with machine instructions. Though she died before her contributions would be realized amidst numerous overshadowing of first her father's illicit affairs, the poet Lord Byron, and then her own, her notes on how to program Babbage's computer would be discovered many years later by Alan Turing, now referred to as the Father of Computer Science.

Father of Computer Science

FIGURE 3. Alan Turing — The Father of Computer Science.

Alan Turing furthered the input ability of letters, numbers and symbols using Ada Lovelace's notes, going on to develop the Turning Machine. Turing is instrumental in furthering the development of theoretical concepts that drive the creation of any computer that has been created to date. He is especially instrumental in providing the express and detailed necessary mimicking of the human brain, though in a logical, straightforward way, based on asking yes and no answerable questions, making decisions that branch based on those responses.

These yes-no questions and making decisions based on the yes or no answer, drive the ALU (arithmetic-logic unit) of the CPU (central processing unit) of all computers. His theoretical projections even provide the basis for the current artificial intelligence era of computer technology. Turing's theoretical developments set the basis for binary coding of information: yes = 1; no = 0; on = 1; off = 0.

The Abacus

All of these men and women were instrumental in what now is the amazing computer translating our thoughts, our calculations, our decisions, our lives through binary math representations. In the words of Charles Babbage "at each increase of knowledge, as well as on the contrivance of every new tool, human labor becomes abridged." Less work for the human body, more work for the tool. Of course, we must go back even further than these men and women. Early, early computer tools were our fingers and toes. Ten fingers, ten toes. It is where we started counting everything we own, including people. The decimal system is base 10, ten things. The abacus is another precursor to computers. The **abacus** is a counting tool, usually in a frame, designed to go beyond counting pass ten. This device was used by early ancient cultures in counting and calculating in the marketplace or keeping track of one's possessions. Various cultures lay hold to the creation of the abacus: Asian, Arabian, African. Each isolated group of human beings, thinking of ways to make life's work less work on the body. Great human minds think alike, advancing and evolving to the practices of today's world.

UNICODE KEYBOARD REPRESENTATIONS

The development of the Unicode system in 1986 has allowed for the standardization of all keyboard systems internationally. Given that the ASCII character look up table was encoded with 0–127 different possible characters represented as Latin-based English letters in lower and uppercase, and decimal numbers 0–9, and representations of the space bar, delete key and other symbols, there came a need to offer an updated keyboard character look up table called the Unicode table.

Unicode Tables

Unicode allows for the representation of additional characters from other languages such as Greek, Arabic, Spanish, Chinese, Hebrew to be represented in the keyboard character look up table. Remember the ASCII character look up table allowed for 0–127 English characters, numerals and symbols. The Unicode table includes ASCII's originally assigned 0–127 characters represented as one-byte or 7 bits in the computer language. In addition to the ASCII's one-byte characters, the Unicode table has expanded to up to 16 possible bytes per character. Unicode embodies the web version of character representation known as UTF-8. **UTF-8**, Unicode transformation format, includes 128 ASCII characters represented as one byte for English; in addition, two-byte representation for Latin and Middle Eastern characters; and three-byte representation for Asian characters. **UTF-16**, supports additional characters in 4-byte representation up to one million. More recently, **UTF-32** also supports 4-byte representation. It is interesting to note that the more bytes assigned to a character may also decrease the efficiency of the use of those characters on the web, in favor of the English language's one-byte representation.

Keyboard Layouts

Windows operating systems do allow for keyboard language changes, although the research indicates that English is the most frequently used keyboard language. The most frequently used **keyboard layouts**, that is the way in which the keys are arranged on the keyboard are the QWERTY and the Dvorak layouts. The **QWERTY** keyboard has an English language layout with the Q, W, E, R, T, and Y keys on the upper left right below the number/symbol keys. On the QWERTY keyboard students learn to type fast without watching their fingers by placing their fingers on the **home row keys**, which are keys A, S, D, F and J, K, L, ; (semicolon), and reaching to other keys from that placement.

FIGURE 4. QWERTY keyboard keys.

The **Dvorak**, is a keyboard layout that uses the keys a, o, e, u, h, t, n and s as home row keys; all of the vowels and punctuation marks are on the left of the keyboard with consonants on the right.

Summary:

From the early days of computing and theoretical conjectures, computers have evolved to be one of the main tools of resource and power in the 21st century. Being able to communicate worldwide due to the open source architecture of unifying keyboard codes, so that different languages can speak to each other in a uniform manner is astounding. It certainly gives leverage to the earlier quote from Charles Babbage that at "at each increase of knowledge, as well as on the contrivance of every new tool, human labor becomes abridged." He was ahead of his time during his time, but he was on target for what the computer with keyboard input has evolved to today.

Checking Your Knowledge:

1. Explain the difference between ASCII v. Unicode character look up charts.
2. Explain how a key character can be converted from decimal to binary code.
3. Describe what an ASCII chart is in five words.
4. Decode the acronym for UTF-8.
5. How many bits are in an ASCII coded key character?

Expanding Your Knowledge:

Use the knowledge gained in this unit to write two paragraphs reflecting on Charles Babbage's quote "at each increase of knowledge, as well as on the contrivance of every new tool, human labor becomes abridged." How is his prediction reflected in today's use of smartphones?

Web Links:

US ASCII Character Set
http://www.columbia.edu/kermit/ascii.html

Babbage's Analytical Engine
https://www.youtube.com/watch?v=5rtKoKFGFSM

Computer Science's Wonder Woman: Ada Lovelace
https://www.youtube.com/watch?v=wnHHzBY1SPQ

The African Abacus
https://www.youtube.com/watch?v=cYbfUIfp2U4

Understanding ASCII and Unicode (GSCE)
https://www.youtube.com/watch?v=5aJKKgSEUnY