COMP253 — Lecture 3: Coding Schemes

(Consult textbook appendices on the ASCII Char Set & the link: http://www.dynamoo.com/technical/ascii-ebcdic.htm *)*

What is a Coding Scheme and why is it necessary?

Computers are able to process instructions and data in binary form, meaning that everything is represented as zeroes or ones ("0" or "1"); which actually means processor circuits being "open" or "closed." The representation of instructions and data is easily understood as numeric information, since there is a relationship between binary and decimal: *they are both number systems*.

But there is a difficulty in trying to represent data that does not describe a numeric quantity, data that is "symbolic" in nature that represents a *quality* not a *quantity*. The best example of such data is an alphabetic character set, such as English. To store, process, and transmit such data, it is necessary to establish an organised (and agreed upon) system that maps a symbol with a corresponding, and unchanging, numeric value.

This mapping is referred to as a coding scheme. And it is very important because only in this manner can the characters and symbols used by us (humans) be transposed to something that the computer can store; essentially, the computer is able to store a *quality* as a *quantity* (a *symbol* as a *number*).

Standardised Character Coding Schemes

Originally, each computer manufacturer designed a unique coding scheme for their systems, to represent all the characters on a keyboard, terminal, and printer. This was not an issue for users, since they were unconcerned with how the system processed their data; but for programmers it was extremely important, especially if the data was to be exchanged between systems, where each used a different coding scheme (consider two people writing with the Latin-character set, but one person is using English and the other French).

EBCDIC (Extended Binary Coded Decimal Interchange Code)

- developed by IBM, and used primarily on IBM systems and systems that interfaced with IBM computers
- each character is represented by a unique 8-bit binary number (a byte), with 256 (2^8) characters represented
- EBCDIC was created specifically to be easily represented on punch-cards
- there exist different variants of EBCDIC, with each being *slightly incompatible with the others* (the popular IBM AS/400 minicomputer/server uses the latest form of EBCDIC—the last system to do so)
- order of symbols: control characters, punctuation, lowercase characters, uppercase characters, then numbers

ASCII (American Standard Code for Information Interchange)

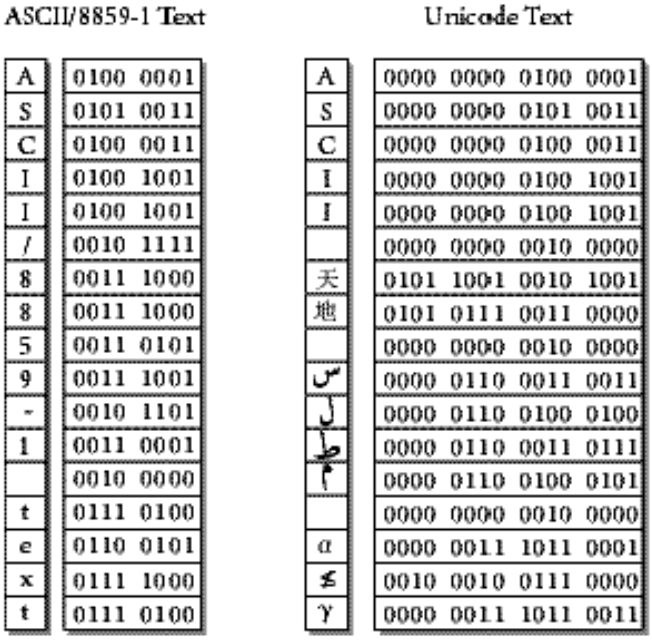
- unlike EBCDIC, ASCII was created for use by the entire computer industry and sanctioned by ANSI (American National Standards Institute), and has become the most popular coding scheme for all computers
- each character is represented by a single byte (7-bit binary number + msb bit for error checking), with 128 (2^7) characters represented; in modern ASCII, all 8-bits are used (no error checking bit) for 256 (2^8) characters *(note: 7-bit form is a subset of 8-bit; no incompatibility problems)*
- ASCII was created in 1960s, during the new era of terminals and electronic data transmissions
- order of symbols: control characters, punctuation, numbers, uppercase characters, lowercase characters, uppercase characters, extended (special) symbols (in 8-bit form)
- ASCII was universally accepted by language and network programmers because of the acceptance of ASCII by UNIX, system-to-system networks, and microcomputers

Examples:

binary ASCII	hex.	char.	binary EBCDIC	hex.	char.
0011 0000	30	0	1111 0000	F0	0
0011 0001	31	1	1111 0001	F1	1
0011 0010	32	2	1111 0010	F2	2
0011 0011	33	3	1111 0011	F3	3
0011 1001	39	9	1111 1001	F9	9
0100 0001	41	A	1100 0001	C1	A

Unicode (Universal Character Code; or Universal Coding Scheme) see: http://www.unicode.org

- created to accommodate all symbols from all languages and professions that require technological representation (and even some that do not); required because of the multi-lingual aspect of the Internet and World Wide Web (WWW)
- is compatible with existing ASCII representations; ASCII 8-bit is a subset of Unicode
- each symbol is represented by a single 16-bit binary number (a word), with 65536 (2^{16}) characters represented
- unlike ASCII that must create symbol pairs to represent *non-English* symbols (called "encoding"), Unicode is able to directly represent all language symbols with a single 16-bit word
- most modern operating systems, languages, and tools understand Unicode as a base coding scheme, and do not require further encoding: WinXP, Linux, MacOS X, MS .Net, Oracle, MS IE, Mozilla, and others.
- some of the languages represented: *Amharic, Arabic, Chinese (Traditional), Chinese (Simplified), Czech, Danish, Dutch, Esperanto, French, Georgian, German, Greek, Hebrew, Hindi, Icelandic, Interlingua, Italian, Japanese, Korean, Lithuanian, Macedonian, Maltese, Persian, Polish, Portuguese, Romanian, Russian, Spanish, Swedish, Thai, Tigrigna, Uyghur, Welsh*
  (note: many languages use the standard Latin/English alphabetic characters; for these Unicode holds the symbols unique to those languages, such as: ¿, æ, ï, ç, ž)
- order of symbols is difficult to describe because of the multi-lingual symbols that are represented, but suffice it to say, the ordering is logical and well-structured to maintain "alphabetic" order in all language forms
- an important aspect of Unicode is that it does not store the font (or "look") of the characters, but the actual character value and order, much like storing the value of "C" instead of its visual representation



The Future

Unicode is the example of what the next generation coding scheme will look like. By allowing for multi-language storage along with special "non-language" symbols, computers can be used to process data that is not confined to one language or industry; consider sorting a number of database records based on a Braille code rather than English characters.

But the purpose of such any coding scheme remains the same as EBCDIC and ASCII: to allow a representation of the symbols used by humans in a binary format for use in storage, processing, and transmission for a computer system.