Subject: Re: Microcomputer History
Posted by Wayne Parham on Tue, 11 May 2021 22:19:49 GMT
View Forum Message <> Reply to Message

Well, yes and yes. On some of the most primitive computers, the code is entered one character (or half-character 'nibble') at a time and, yes, it's incredibly tedious. Which was one of the main early goals or programmers in the early days - to develop languages and operating systems that made programming less tedious.

When you enter code that way, it's called machine code, binary or object code. You are talking directly in the computer's language. You might tell the computer to get the contents of a memory location, load into a temporary place called a register, test the contents (like for equality, less than or greater than) and jump somewhere else in the program if the condition was met. Or maybe you'd load contents from memory, add to contents or another memory and store somewhere else.

Each instruction is one-byte code (in a machine like these), so for example, in the 6502 microprocessor, the load instruction is decimal 169 or hexadecimal A9. Microcomputer guys think in hexadecimal. Lots of minicomputer guys think in octal and that same load instruction is 251 in octal. So to program in machine code, you hand-enter a bunch of numbers in sequence. That's your program.

Next level up would be assembly language. In that case, you don't have to know the number of each opcode. You can call it by name, or actually by a "shorthand" abbreviation we call a mnemonic. That A9 load code I just described is called LDA in assembly language, the mnemonic for Load Accumulator. You can also label memory locations instead of calling them out by numbers. So it's one level of abstraction above machine code, not much above, just barely.

Then above that would be languages like BASIC or C. Those are more "human-readable" and require an interpreter or compiler to translate the program into machine code for the computer to be able to act upon.