number symbols of a 7-bit word. Each letter, number, and control symbol is given a unique value. When the key on a computer keyboard is pushed, that 7-bit word is generated. The actual word contains 8 bits; the extra bit is often used for parity checking (even or odd), or sometimes fixed as a leading 1 or 0.

ASCII is another representation system for binary numbers. This further illustrates the reason why we cannot tell the value of a binary number without knowing the system. The number could be the symbol for a letter such as Q. We say that the value or meaning is a letter rather than a quantity.

15. OVERFLOW

When the result of an arithemtic operation of two legal numbers gives a number too large to be represented, the result is called an overflow error. The term overflow is a particular kind of error which cannot be avoided. In an 8-bit integer system, there is no legal result when 127 is added to 2 because the number 129 does not exist.

Overflow can apply to both fixed and floating point numbers.

16. UNDERFLOW

This is the reverse case of overflow: the result is too small. It is generally only applied to floating point numbers which have special problems when there is not enough range in the exponent. The fractional part is limited to the range of 1 to 0.5. Therefore, the smallest number might be 0.5 E-99. An underflow would result when we tried to divide this number by 2. The result would be the same as the initial number since there is nothing smaller.

17. CLIPPING

What audio engineers call clipping, digital engineers call limiting. A hardware of software trap is added to the arithmetic process such as that the overflow error case is treated explicitly. The result is artificial, limited (clipped) to the most positive or negative value. We call this an "error handling" algorithm. The overflow is an error, but we define the way in which it will be handled. The previous example, for adding 127 to 2, gives the result 127 when we have clipping. Similarly, 120 plus 12 gives 127 and -110 plus -30 gives -128.

18. TRUNCATION

This is the name for the process by which certain bits in the result are thrown away. The resulting error is called a truncation error. It typically results from multiplication and division since these processes double the size of the result. A 16 bit number multiplied by a 16 bit number result in 31 bits. If the result must be again represented in 16 bits, we throw away the lower 15. The act of discarding means that the result contains an unknown error. It is analogous to quantization error in an A/D converter.

Addition and subtraction do not result in truncation issues because the result has the same number of bits. The exception is the overflow case. There the upper bit is lost because the number is too small. If the addition were to have a *defined* divide by 2, then there could never be an overflow, but there would be a truncation error since the lowest bit is removed. The addition of two 16-bit numbers can result in a 17-bit answer. The extra bit is usually an issue of overflow, but a shift in the binary point can turn it into a truncation issue. ■