

# **The New C Standard** (Excerpted material)

---

**An Economic and Cultural Commentary**

**Derek M. Jones**

derek@knosof.co.uk

### 3.5

bit

#### bit

unit of data storage in the execution environment large enough to hold an object that may have one of two values

#### Commentary

Although it is the most commonly used representation in silicon-based processors, two-valued logic is not the most efficient form of representation. The most efficient radix, in terms of representation space (number of digits times number of possible values of each digit), is,  $e^{[2]}$  (i.e., 2.718 . . .). The closest integral value to  $e$  is 3. While using a ternary representation maximizes the efficiency of representation, there are practical problems associated with its implementation.

The two states needed for binary representation can be implemented using a silicon transistor in its off (very low-voltage, high-current) and saturated (high-voltage, very low-current) states. Transistors in these two states consume very little power (voltage times current). Using transistors to implement a ternary representation would require the use of a third voltage (for instance, midway between low and high). At such a midpoint voltage, the current would also be mid-way between very low and high and the corresponding power consumption would be significantly higher than the off and saturated states. Power consumption, or rather the heat generated by it, is a significant limiting factor in processors built using transistors.

Vendors have chosen to trade-off efficiency of representation for the lower power consumption needed to prevent chips melting. Processors based on a binary representation are overwhelmingly used today. The definition of the C language reflects this fact.

#### C++

ISO 2382 The C++ Standard does not explicitly define this term. The definition given in ISO 2382 is “Either of the digits 0 or 1 when used in the binary numeration system.”

#### Other Languages

The concept of a bit is a low-level one, intimately connected to the processor architecture. Many languages do not get explicitly involved in this level of detail. The language of Carbon-based processors (at least on planet Earth), DNA, uses a unit of storage that has one of four possible values (a *quyte*<sup>[1]</sup>): *A*, *C*, *G*, or *T*.

#### Common Implementations

Processors rarely provide a mechanism for referencing a particular bit in storage. The smallest unit of addressable storage is usually the byte. There are a few applications where the data is not byte-aligned and processors supporting some form of bit-level addressing are available.<sup>[3]</sup> Automatic mapping of C source that performs its own manipulation of sequences of bits (using the bitwise operators) to use these processor instructions is not yet available. However, at least one research compiler<sup>[4]</sup> provides support via what, to the developer, looks like a function-call interface.

byte  
addressable unit

bit  
address of

NOTE It need not be possible to express the address of each individual bit of an object.

#### Commentary

The smallest object that is required to be addressable is one having character type. The number of bits in such an object is defined by the CHAR\_BIT macro, which must have a value greater than or equal to 8.

character types  
CHAR\_BIT  
macro

#### Common Implementations

Processors rarely support bit addressing, although a few of them have instructions for extracting a sequence of bits (that is not a multiple of a byte) from a storage location.

## References

1. Anon. How genes work: A quick guide to life's operating system. *The Economist*, June 29 2000.
2. B. Hayes. Third base. *American Scientist*, 89(6):490–494, 2001.
3. X. Nie, L. Gazsi, F. Engel, and G. Fettweis. A new network processor architecture for high-speed communications. In *IEEE Workshop on Signal Processing Systems (SiPS'99)*, 1999.
4. J. Wagner and R. Leupers. C compiler design for an industrial network processor. In *Proceedings of The Workshop on Languages, Compilers, and Tools for Embedded Systems (LCTES 2001)*, pages 155–164, 2001.