Binary Prefixes for Quantities

Douglas A. Kerr

Issue 1
February 25, 2009

ABSTRACT

Early in the unfolding of the modern era of computer science, it was recognized that computer memory modules typically had location sizes that were integral powers of two. To allow sizes of commonly-encountered memory modules to be stated with modest-size integers, it became common to speak of the size of computer memory modules in terms of a multiple unit of 1024 (2^{10}) locations. Sadly, rather than adopting a distinct name and symbol for that multiple, the workers just hijacked the prefix term, "kilo", and symbol, "k", used in science and engineering (since 1795) for a unit multiple of 1000. This practice then escalated (in a not-always-consistent way) to larger multipliers such as *mega* (M) and *giga* (G). The result was ample opportunity for misunderstanding. Now, an international standard provides distinct, unambiguous prefix names and symbols for a series of "binary" multiples.

This article gives historical and technical background in this matter and describes the new (but sadly, rarely-used) scheme of "binary" multiples.

THE BINARY NUMBER SYSTEM

The onset of the "modern" computer era (perhaps in the late 1950's) brought many technical workers face-to-face with something that, if they had previously encountered it at all, was only as a curiosity in math class lectures on numbering system theory: the *binary numbering system*.

USE IN COMPUTERS

The use of the binary numbering system in computers was a natural consequence of the use of "two-state" electrical representation, where a given electrical lead would carry either a positive or negative voltage (or perhaps either a voltage or zero voltage). This context applied to the representation of codes for "operations" in the programming "machine language", to the representation of actual numeric quantities, and for the identification of address locations in the computer memory. Thus we came to deal regularly with binary numbers. We did, eventually, learn to write their values in terms of the corresponding *octal*, and later, *hexadecimal*, representations, for convenience.

If, for example, we had allocated a 12-bit value as a pointer to address locations, that could delineate 4096 distinct locations. Since we didn't want to waste anything, we would, for example, probably not organize our memory in modules of 4000 locations. Thus, the sizes of memory systems tended to be integral powers of 2.

A UNIT MULTIPLIER

It was soon noticed that, were we to designate substantial quantities (by the standard of the day) of memory locations with a multiple that was an integral power of two, the typical memory sizes we encountered could be described with modest-sized integers.

Recall that, in regular engineering use, a unit multiple of 1000 (with the prefix **kilo**, symbol **k**)[1] is the first multiple whose use is recommended. We don't, for example, give frequencies in decahertz (a multiplier of 10) or hectohertz (a multiplier of 100).

Following suit, many early computer workers decided that the "binary" multiple closest in size to 1000 should be the basic multiple of address space size that would be used: 1024. (1024 is 2^{10}.) For the moment, I'll call that unit multiple a "thousink" to avoid any ambiguity; applying ambiguity will be done by others.

Thus, for example, a module of memory with an 18-bit address pointer, and thus with a capacity of 2^{18} (262,144) words[2], could be succinctly described as having a capacity of 256 thousink words. (Without inventing the thousink, we would have to have said that the size of our 18-bit pointer memory was 262,144 words, or 262.144 kilowords, a rather bulkier notation.)

Of course, at this time, a new prefix, with a distinct symbol, should have been adopted for the "thousink". But in the rough-and-tumble world of early computer work, this seemed like a "prissy intellectual nicety", not worth wasting time on,[3] and so it became common in that context to refer to the multiplier 1024 as "kilo" and use the symbol "k" for it.

---

[1] This convention was established in 1795.

[2] Note that here I use "word" in its generic sense, a binary number some number of bits in length, not to mean specifically a "16-bit number", as later became common.

[3] Remember, this was the community that, in I/O for the CP/M operating system, assigned the ASCII character ETX (End of Transmission) to mean "cancel program execution", because it was generated on a standard ASCII keyboard by the combination Ctrl + C ("C" for "cancel").

Let me emphasize at this point that the only visible advantage of this convention was that the typical memory size we encountered could be described with modest-sized integers.

It doesn't really simplify any mathematical calculations that had to be made. It does mean that the size of a memory module with a pointer size of *n* could be reckoned in thousinks this way:

$$\text{size} = 2^{(n-10)} \text{ (thousinks)}$$

Thus, since computer folks generally know by heart powers of two up to perhaps 12, they could easily determine the size of memories (in thousinks) up to a pointer size of about 22 bits by inspection.

Despite this feeble advantage, and notwithstanding the smell of danger ahead, this convention spread in use.

THE IDEA GROWS

This usage was soon extended to designations of amounts of data to be placed in memory locations (for example, file sizes). At the time, the "byte" was not the common module of data, since typical computer systems used perhaps 6-bit words, but we might hear that the machine code for a particular program "occupies 4k words" (meaning 4096 words).

But there is no advantage to the notation in that case, except that it is consistent with the notation used for memory sizes.

Because of the rather confined context of this usage, it was thought that there would rarely be any ambiguity as to whether "k" mean a multiple of 1000 or 1024. If the quantity was voltage, or frequency, or clock rate, then it meant 1000; if it was memory space, or number of data words, it meant 1024.

COUNTING OTHER THINGS

But what if the item measured was not data words (bytes, perhaps, today) but bits? After all, in a memory array, a multiple of 1024 for bits was not inherent. Imagine a system using six-bit words. If we had, say, a 7-bit memory pointer, then the number of bits of memory it spanned would be 768; for an 8-bit pointer, 1536. So should we count bits in units of 1024?

And what if it was data transmission rate? Was 100 kb/sec 100,000 bits per second or 102,400 bits/second? Or data transfer rates: was 10 kB/sec 10,000 bytes/second or 10,240 bytes/second?

LARGER QUANTITIES

This scheme was eventually extended to larger quantities, but some further complications crept in. In general, following the lead in normal prefixes (which grow by multiples of 1000), it seemed as if the next one above the thousink should be 2^{20} in size (1,048,576). If we retained our obsession with address space size, that made no worse sense than the first one. This multiple prefix was called, in the "binary" context, "mega", and the symbol "M" was used for it (the indicators, in normal scientific notation, for a unit multiplier of 1,000,000).

As it became common in memory design to organize the memory in individual units of 8 bits (bytes), or else a binary multiple of 8 bits (16 bits, 32 bits, 64 bits), the byte become the basic denomination of memory size, or data size, and thus the designations counts in bytes came to be in so-called kilobytes (kB) or megabytes (MB).

But in some situations, address space wasn't the consideration. For example, in a disk storage unit, the overall capacity wasn't usually inherently an integral multiple of 2. Some workers felt that here, capacity should be denoted with the normal use of the unit prefixes, in which "M" indicated a multiplier of 1,000,000. Others, believing that computer workers somehow inherently thought in terms of multiples of 1024 for anything denominated in bytes, thought that the next unit, designated "M", should be 1000 x 1024, or 1,024,000. Others felt that the "binary mega" should obviously denote a multiplier of 2^{20} (1,048,576), as mentioned just above, by extensions of the doctrine for the thousink.

And so it unfolded. It is another case in which those that complained about ambiguity were told not to worry, that the context made the meaning clear. Except in cases where it didn't.

MOVING OUT OF THE COMPUTER

A few decades later, digital cameras came into existence, and there was interest in citing the number of pixels they could distinguish on their sensors. One camp felt that this should be done in modules of 2^{10} or 20^{20}. If we said that a camera has a pixel array size of 2 megapixels, that would mean about 2,097,152 pixels.

Now why should that be? Indeed, there is an issue in the internal structure of the camera of addressing the pixels, but that is of no importance to the user (and we really have no idea how that is actually organized). Still, the proponents of this notation argue "this is a digital thing, and we all know that in digital things, *mega* means a multiplier of 2^{20}." Except of course if we are speaking of a frequency, or maybe a data transfer rate—or maybe pixels.

A PARTIAL CURE

One technique adopted by a number of authors (including myself) to help minimize ambiguity here is the use of the symbol "K", rather than "k", to mean a multiple of 1024 rather than 1000, thus making clear which meaning is intended.

We are only able to do this because of a peculiarity in the scheme of unit multiple prefixes of the International System of Units. Normally, multiples greater than one are given upper-case letter symbols; those less than one (fractional multiples) are given lower-case letter symbols. But in the case of "kilo" there was concern that the symbol "K" for a prefix could have been confused with the symbol "K" for a unit (the Kelvin). Thus, lower-case "k" was adopted for kilo.

Of course this ploy of distinguishing the "binary" and "decimal" meanings of the multiplier prefix (using "K" *vs*. "k") could not be applied for larger multiples, such as "mega", where the standard symbol was already in upper case ("M").

COMMERCIAL PRACTICE

For some years it has been the practice of hard disk drive manufacturers to specify the capacities of their products using quantity prefixes in the normal technical sense (1 MB = 1,000,000 bytes). Detractors say that this is merely a ploy to allow them to quote a bigger number than is "true". (That is, everybody should know that a drive that holds about 157,300,000 bytes should be rated at 150 MB, not 157 MB.) There have in fact been some infamous lawsuits over this matter.

The capacity of DVD disks is normally cited using the standard decimal meaning of the multiplier prefixes, as is the case for portable "flash" memory modules.. But floppy disk and CD disk capacities are normally stated using the "binary" meanings of the prefixes. Thus a 4.7 GB DVD has a capacity about 6.9 times that of a 650 MB CD (despite the fact that 4.7 gigarabbits is about 7.2 times as many rabbits as 650 megarabbits).

RECOGNITION IN STANDARDS

At one time, the use of "kilo", with the symbol "k", and "mega", with symbol "M". to mean 1024 and 2^{20}, respectively, as alternate meanings "in statements involving size of computer storage", were recognized by certain ANSI/IEEE standards. That has essentially been overturned by modern developments.

RELIEF

In 1999, the International Electrotechnical Commission (IEC), the international standards body responsible for the International System of Units (SI), issued an addendum to that standard that established a series of "binary-oriented" multiples with unambiguous names and symbols.

For example, the prefix for a multiple of 1024 (2^{10}) has the name "kibi" (from kilo binary) and the symbol "Ki". The prefix for the multiple of 1,048,576 (2^{20}) has the name "mebi" (from mega binary) and the symbol "Mi".

In all cases, including *kibi*, the initial letter of the symbol is upper-case (there being no concern with confusion with the symbol for any unit). The universal second syllable "bi" is to be pronounced "bee".

The table below gives the defined binary multiplier prefixes.

Name	Symbol	Value (binary)	Value (decimal)	Related decimal prefix		
				Name	Symbol	Value
kibi	Ki	2^{10}	1024	kilo	k	1000
mebi	Mi	2^{20}	~1.048 x 10^6	mega	M	10^6
gibi	Gi	2^{30}	~1.074 x 10^9	giga	G	10^9
tebi	Ti	2^{40}	~1.100 x 10^{12}	tera	T	10^{12}
pebi	Pi	2^{50}	~1.126 x 10^{15}	peta	P	10^{15}
exbi	Ei	2^{60}	~1.153 x 10^{18}	exa	E	10^{18}
zebi	Zi	2^{70}	~1.181 x 10^{21}	zetta	Z	10^{21}
yobi	Yi	2^{80}	~1.209 x 10^{24}	yotta	Y	10^{24}

Table of IEC standard binary multiple prefixes

Today, essentially all relevant domestic and international standards discourage the use of "kilo" and "k" (or "K") to mean a multiplier of 1024 and so forth.

Sadly, though, there has been little adoption of the new, unambiguous practice in the literature.

ENOUGH IS ENOUGH

In fact, the notion of using “binary” multiples for things such as memory array size or size of a data file has long (by about 40 years) overlasted any reason for it. Although, thankfully, we now have an unambiguous system for notation under that practice, it would really be best to just give it up. Nobody has any real interest in knowing the number of bytes in a document file to be downloaded in units of 1024 (except that they heard the size of the last file they downloaded in units of 1024). It is life imitating (bad) art.

We should just revert to denominating things with the standard multiples of the SI and (where there is a chance of misunderstanding owing to years of abuse) remind the reader that we really mean here “kilo” or whatever. The notion that we should avoid confusing people by consistency in the use of a confusing convention is self-defeating.

This article has an overall length of about 2.41 kilowords, and comprises about 11.7 kilocharacters. It is held in a Word file with a size of about 13.3 kB. Is it more useful, for a digital document, to say, using the unambiguous “binary” convention, that it is it is 2.34 kibiwords long and comprises 11.5 kibicharacters, held in a 13.0 KiB file? Hardly. Is it more meaningful if we say, using the ambiguous “binary” convention, that it has 2.34 kilowords and 11.5 kilocharacters, held in a 13.0 kB file? Hardly.

OUR PRACTICE

At present, in such places as the index of technical articles on my technical information site, The Pumpkin, I cite document file sizes in both decimal and binary multiples (using the proper, unambiguous notation for each).

However, I will soon be discontinuing the use of the binary multiples (while retaining the note that explains unambiguously what the notation I do use means).

#