

## Re: how to detect the compile is 32 bits or 64 bits?

---

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2006-08/msg01498.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2006-08/msg01498.html)

---

- *From:* [roberson@xxxxxxxxxxxxxxxxxxxx](mailto:roberson@xxxxxxxxxxxxxxxxxxxx) (Walter Roberson)
  - *Date:* Sat, 5 Aug 2006 16:23:52 +0000 (UTC)
- 

In article <0f3Bg.2584\$9T3.1111@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>, Tim Prince <tprince@xxxxxxxxxxxxxxxxxxxx> wrote:

steve yee wrote:

i want to detect if the compile is 32 bits or 64 bits in the source code itself. so different code are compiled respectively. how to do this?

You invite bugs, as well as taking the question off the topic of standard C, if you write source code which has to change.

Not necessarily.

The usual question is about (sizeof)<various pointer types>. If your question is simply about (sizeof)int, you create problems by assuming it is determined by whether you have a 32- or 64-bit platform. If you insist on unions of pointers and ints, knowing whether it is 32 or 64 bits won't save you. More so, if you are one of those who writes code which depends on (sizeof)(size\_t x).

You appear to have read a fair bit into the poster's question that I don't think is justified by what the poster wrote.

Suppose I have an algorithm, such as a cryptography algorithm, that operates on chunks of bits at a time. The algorithm is the same (except perhaps for a few constants) whether I'm computing with 32 or 64 bits, but the chunk size differs for the two cases. In such a case, I -could- write the code using only the minimum guaranteed size, but on most platforms it would be noticeably more efficient to use the larger chunk size \*if the platform supports it\*. The constants for the algorithm could probably be computed at run-time and a generic algorithm used, but in the real world, having the constants

Re: how to detect the compile is 32 bits or 64 bits?

available at compile time increases compiler optimization opportunities leading to faster code.

In C, it is not an error to write `int x = 40000;` for use on platforms that have an `int` of at least 17 bits. It is not maximally portable, but it is not an error — and the OP was asking for a compile-time method of selecting such code for platforms that allow it, dropping back to smaller value assumptions when that is all the platform supports.

—

Okay, buzzwords only. Two syllables, tops. — Laurie Anderson

.