Bitwise operation error with long long pic32

I am working wiht pic32mx controller and using the C32 toolsuite v2.02 compiler amd MPLAB-X IDE v 3.05. I am using the bit wise operation with the long long variable but getting erroneous result.

```
int main()

{

    long long data0, data1=0;

    data0 = 489631651402;
    data1 = data0 & 0x0FFFFFFFFF;

    printf("%llu\n%llu\n", data0, data1 );

    return 0;
}
```

OUTPUT:

```
    489631651402

    492260348528
```

According to calculation this result is erroneous. data0 and data1 must be equal.

Please suggest me any better way for this operation. if there's any error in above procedure please suggest the right path.

c microcontroller pic pic32

edited 35 mins ago
Joachim Pileborg
187k ● 14 ● 119 ● 216

asked 43 mins ago
ajay vachhani
21 ● 2

Did you have a look at the generated assembly code ? – Michael Walz 37 mins ago

3 So the number becomes *larger* when you bitwise-and it with a constant? Hm ... That's normally not very possible. I suggest you double-check this, and also why not print the numbers in hex, to make them slightly more understandable? Plus what Joachim said in his deleted answer, not sure why that's gone. – unwind 29 mins ago

The only problem i see with this code is using the `%llu` specifier with signed values. I doubt that would cause this error though. – interjay 21 mins ago

1 Please check this microchip.com/forums/m488183.aspx, I am not sure if this issue is related (you never know with C32) but it is worth of trying... – Darko Djuric 13 mins ago

It seems that the C32 compiler support for 64-bit types is flaky at best. First of all it seems to be an extension bolted onto a compiler basically supporting only C90 (as told in the forum thread linked to by @DarkoDjuric); Secondly the compiler have to generate all code to handle 64-bit types and 64-bit arithmetic since the CPU doesn't support it natively, so it is possible that you found a bug in their compiler for handling 64-bit integer types. – Joachim Pileborg 5 mins ago

1 Answer

Use ULLONG_MAX as the constant if you need a mask for all of the bits. Use 0x0FFFFFFFFF**LL** if you need to mask only those 48 bits

edited 17 mins ago

answered 40 mins ago
Michael
304 ● 2 ● 13

`ULLONG_MAX` is always more than `0x0FFFFFFFFF`. And its value is not guaranteed to be equal to anything (there's only a lower bound), so it wouldn't be a good idea to use it anyway. – interjay 32 mins ago

The object that was undergoing bitwise AND is of the long long type. – Michael 29 mins ago

I know it is, but that doesn't explain why you think `ULLONG_MAX` is a good replacement for `0x0FFFFFFFFF`. – interjay 23 mins ago

@interjay Are you sure? The text uses the word magnitude: *Their implementation-defined values shall be equal or greater in magnitude (absolute value) to those shown, with the same sign.* Doesn't that indicate power of 2 -1 ? – 2501 22 mins ago

Since we do not know the purpose of that bitwise AND, both you and I can be correct. I'll edit the answer – Michael 18 mins ago