

Re: Version after Version

Source: <http://coding.derkeiler.com/Archive/Delphi/alt.comp.lang.borland-delphi/2005-10/msg00408.html>

- *From:* "Maarten Wiltink" <maarten@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 31 Oct 2005 10:00:42 +0100
-

"Frank de Groot" <franciad@xxxxxxxxx> wrote in message
[news:g1K8f.496\\$Ti5.16479@xxxxxxxxxxxxxxxxxxxxx](mailto:news:g1K8f.496$Ti5.16479@xxxxxxxxxxxxxxxxxxxxx)
> "Maarten Wiltink" <maarten@xxxxxxxxxxxxxxxxxxxxx> wrote in message
> [news:436360ab\\$0\\$11076\\$e4fe514c@xxxxxxxxxxxxxxxxxxxxx](mailto:news:436360ab$0$11076$e4fe514c@xxxxxxxxxxxxxxxxxxxxx)

>> I tend to do my operations one method call at a time, not MxN bits at a
>> time.
>
> Are you saing that when you need to, say, add 1000 numbers in an array,
> thay you do this:
>
> for i := 0 to High(anArray) do
> AddNextNumber(anArray, i);

Without a second thought. When you say "number", do you mean real numbers
or imaginary? Integer or floating point? Signed or unsigned? Any chance of
overflow? Adding up all numbers in an array is a fold() of the addition
operator over it. My first priority is maintainable, extensible, robust,
clear code. Not optimisation. When it takes more than ten milliseconds,
I'll start worrying about it. Not before.

> That would be insane.
> I am sure you are not **that** stupid.

No, strangely I'm not. I'd use Low(), too.

If AddNextNumber is a TCollectionItem.Create(), how exactly would you
put that in parallel? In my world, that happens a lot more often than
summing up a block of integers.

I'll allow you your niche, and keep reasonably up to speed on it. Kindly
return the courtesy.

> What you say is nonsensical. Every time you do more than on 32-bit
> operation on an array for example, you can make things much faster by
> loading 64 bits at a time from memory (as my dual-cpu, dual-core 275
> machine does), operating on those 64 bits and write them back in one

Re: Version after Version

> operation, to 64-bit wide memory.

What you say *depends*. I don't have many arrays. They rarely contain integers. That rather lessens the "much" in your argument.

> When you