

"Martin Downing" <[martin.downing@ntlworld.com](mailto:martin.downing@ntlworld.com)> writes:

> Here's a thought for the day:  
> we have all had to get used to using the temporary function to enable memory  
> efficient code. Some of us less effectively than others!  
> ie: instead of  
>     $a = 2*a + b/a$   
> write  
>     $a = 2*a + b/\text{TEMPORARY}(a)$   
>  
> Personally although good practice I find it makes code hard to read. Who  
> agrees that this could and should be dealt with at the compilation stage,  
> obviously if A is being reassigned then the previous contents are lost so  
> the compiler could reuse A when processing the last copy of A on the right  
> hand side. Would that be so hard for RSI to implement?

I agree, Martin. A compiler writer would know exactly when a variable on the right hand side is being reassigned. However, your example points out at least one of the problems.

>     $a = 2*a + b/\text{TEMPORARY}(a)$

Since A appears twice on the right hand side, the compiler would need to be smart enough to not overwrite A after its first appearance. In fact, I am not sure that IDL makes any guarantees about order of evaluation and side effects. Isn't it possible that the TEMPORARY() gets called before the first A is evaluated? [ Not sure on this, but that's why I avoid the situation. ]

Second of all, if the compiler automatically TEMPORARY()'ed every variable that was reassigned, it makes debugging harder. What if your expression was:

>     $a = 2*a + F(B)$

If F(B) crashed after 2\*A was evaluated, then there may be no way to recover the original value of A. So, there would probably need to be a "debug" vs. "performance" compilation flag.

Craig

--  
-----

Craig B. Markwardt, Ph.D.      EMAIL: [craigmnet@cow.physics.wisc.edu](mailto:craigmnet@cow.physics.wisc.edu)  
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response  
-----

---