
Subject: Re: c64/How to predict when garbage collection occurs..

Posted by [Dombo](#) on Sun, 15 Sep 2013 09:34:37 GMT

[View Forum Message](#) <> [Reply to Message](#)

Op 15-Sep-13 4:43, Anton Treuenfels schreef:

>
> Anton:
> Do you think you could.. somehow.. come-up with a (Holier than Gawd)
> code snippet, just to see, exactly, how you would approach this scenario??
>
> -----
>
> Uh, I'm not sure I'd try approaching the problem from the direction of
> an IRQ in any case. I'm not quite sure if you're trying to monitor the
> situation from BASIC via a flag set by an IRQ, or trying to do a
> complete garbage collection within an IRQ handler, or both.
>
> In the first case you'd have to poll the flag from within BASIC,
> presumably using PEEK. If you can do that regularly enough to matter
> then you could also get effectively the same result by PEEKing the
> locations of interest and subtracting their values directly without
> needing to fiddle with an IRQ. As a numerical operation doing so would
> not trigger a garbage collection unless you decided to perform one as a
> result.
>
> In the second case, no matter how much you managed to speed up the
> process it would still be too long for a single interrupt.
>
> In the third case (ha! you didn't know about that one, did you?!) it's
> all moot unless you have a faster garbage collector to use instead of
> the built-in one. Have you?
>
> I did write a replacement once. It worked, but perhaps the only elegant
> thing about it was that I tied it to the FRE() function so it was fairly
> transparent to the system. Other than that it was quite an ugly hack.

Another issue with doing the garbage collection from an IRQ handler is reentrancy.

I doubt the BASIC interpreter, and the garbage collection routine in particular, is written with reentrancy in mind. Imagine the BASIC interpreter loading some pointers to string data, gets interrupted by the IRQ handler which shuffles the location string data. Once the IRQ handler has done its thing the BASIC interpreter continues where it was interrupted, at which point the pointers it loaded before the interrupt occurred are no longer valid. The effect would be that at seemingly random times your BASIC program would act weird or locks up the computer.
