Anyone who's ever worked with me knows that I have several weaknesses. One of them is extra sensitivity to things that reside under **sys** (global.sys) in Specman/e. If this is Chinese to you then you're probably a SystemVerilog guy: "sys" is the top most
e
struct (class) and the one struct people should strive NOT to use. The only thing that should be placed there is your own top unit, preferably customized to your project needs.

So the reason why people would want to use sys to place methods or other fields in the first place is because it's accessible from everywhere, which is extremely convenient. But methodology-wise it would not be the smartest thing to do. Let me tell you why:

1. If you were the only verifier in your project then I would let you slip away. But since this is not the situation in most of the cases - you might get contentions on struct member names.

2. Placing code under sys encourages the use of full e paths rather than pointers and ports - code integration becomes a big mess this way, trust me.

3. Additional instances of your env is impossible because you can't instantiate sys - no reuse.

4. You won't be able to use your environment in a larger context because it's not tightly encapsulated - no scalability.

5. There's no logical distinction between the various elements, everything under the same struct - very hard to maintain

With all that being said - sometimes we need a speedy action and using sys could be a quick

and dirty solution - that’s ok. But as a general guideline, unless you have a really good reason, don’t be **sys**sy, create your own top level struct/unit, and use self-contained components.