

Overview: The purpose of this is to correct inconsistencies between the bind BNF and the text. Specifically:

1. The text after the BNF states that “Possible target scopes include module, program, and interface declarations” but the BNF only shows modules and interfaces, and the text at the top only states modules and instances.
2. Clarify that what is allowed for a bind instantiation is restricted by what is allowed to be instantiated within the bind target scope.
3. Rewrite of the first section for clarity

Replace BNF in A.1.4:

```
bind_directive ::=  
    bind bind_target_scope [: bind_target_instance_list] bind_instantiation ;  
    | bind bind_target_instance bind_instantiation ;
```

With

```
bind_directive ::=  
    1  
    bind bind_target_scope [: bind_target_instance_list] bind_instantiation ;  
    | bind bind_target_instance bind_instantiation ;
```

Note 1: when the bind target scope or bind target instance is an interface identifier, the bind instantiation must be an interface instantiation.

Replace in 17.15:

To facilitate verification separate from design, it is possible to specify properties and bind them to specific modules or instances. The following are some goals of providing this feature:

- It allows verification engineers to verify with minimum changes to the design code and files.
- It allows a convenient mechanism to attach verification Internet Protocol (IP) to a module or an instance.
- No semantic changes to the assertions are introduced due to this feature. It is equivalent to writing properties external to a module, using hierarchical path names.

With this feature, a user can bind a module, interface, or program instance to a module or a module instance.

With

~~To facilitate verification separate from design, it is possible to specify properties and bind them to specific modules or instances. The following are some goals of providing this feature:~~

~~— It allows verification engineers to verify with minimum changes to the design code and files.~~

~~— It allows a convenient mechanism to attach verification Internet Protocol (IP) to a module or an instance.~~

~~— No semantic changes to the assertions are introduced due to this feature. It is equivalent to writing properties external to a module, using hierarchical path names.~~

~~With this feature, a user can bind a module, interface, or program instance to a module or a module instance.~~

It is often desired to keep verification code separate from the design code. SystemVerilog provides a *bind* construct that is used to specify one or more instantiations of a module, interface, or program block without modifying the code of the target. So for example, an assertion-based checker that is encapsulated in a module, interface, or program can be instantiated in a target module or a module instance in a non-intrusive manner. Similarly, an assertion-checker that is encapsulated in an interface can be bound to a target interface or interface instance.

Replace in Syntax 17-17

```
bind_directive ::=  
    bind bind_target_scope [: bind_target_instance_list] bind_instantiation ;  
    | bind bind_target_instance bind_instantiation ;
```

With

```
bind_directive ::=  
    1  
    bind bind_target_scope [: bind_target_instance_list] bind_instantiation ;  
    | bind bind_target_instance bind_instantiation ;
```

Note 1: when the bind target scope or bind target instance is an interface identifier, the bind instantiation must be an interface instantiation.

Replace in 17.15

Possible target scopes include module, program, and interface declarations.

With

Possible target scopes **include** are **modules**, **program**, interfaces and their instances.