Original message
from Steven Sharp <sharp@cadence.com>
date Fri, Jan 28, 2011 at 7:04 PM
subject [sv-bc] nested interfaces as "interfaces to interfaces"

Taking Peter's first examples:

```
package P_3_2;

      // Modport suitable for connection to an RS-232 serial link
      modport rs_232 (
            input logic RXD, DSR, CTS, DCD,
            output logic TXD, RTS, DTR );

      // Modport representing a general-purpose test point of any type
      modport testpoint #(parameter type T = logic) (
            output T TP);

endpackage : P_3_2
```

and showing how it could be done with interfaces inside interfaces (possibly with mistakes, as I am doing this off the top of my head):

```
// Interface suitable for access to an RS-232 serial link

interface rs_232 (
                  input logic RXD, DSR, CTS, DCD,
                  output logic TXD, RTS, DTR );
endinterface


// Interface representing a general-purpose test point of any type

interface testpoint #(parameter type T = logic) (
                   output T TP);
endinterface



// Use of nested interfaces to give access to the internals of an
interface

interface my_rs232_interface;

      rs_232 link(.RXD(local_RXD), .DSR(local_DSR),
                  .CTS(local_CTS), .DCD(local_DCD),
                  .TXD(local_TXD), .RTS(local_RTS),
                  .DTR(local_DTR));

      testpoint tp(.TP(signal));

endinterface
```

```
interface my_other_rs232_interface;

      rs_232 link(.RXD(other_RXD), .DSR(other_DSR),
                  .CTS(other_CTS), .DCD(other_DCD),
                  .TXD(other_TXD), .RTS(other_RTS),
                  .DTR(other_DTR));

endinterface


module foo ( my_rs232_interface ifc,
             my_other_rs232_interface other);

      virtual rs_232 rs;
      virtual testpoint tp;

      initial
            begin
              if (want_other)
                rs = other.link;
              else
                rs = ifc.link;
              tp = ifc.tp;
              $display(rs.RXD, tp.TP);
              rs.TXD = 0;
            end

endmodule
```

We now have access to the common `rs-232` signals of two different (and incompatible) types of `rs-232` interface instances, using the same virtual interface. The declarations and instantiations of the nested interfaces are not significantly different from the proposed declarations and instantiations of the stand-alone modports. Are there some other usages of stand-alone modports that are likely to be common, that cannot already be handled by this existing capability?