FA23

Interfaces

@nebu

What is an interface?

- "At its lowest level, an interface is a named bundle of nets or variables." – IEEE 1800-2012 25.2.
- Here's an example of what one looks like:

```
interface axi_stream;
  logic clk;
  logic rst;
  logic valid;
  logic ready;
  // ... and so on
endinterface
```

Why is it useful?

- Consider the port list of this DUT that uses the AXI Stream protocol for I/O:

```
module dut (
  input logic       clk,
  input logic       rst,
  // Input stream
  input logic       valid_i,
  input logic [7:0] data_i,
  output logic      ready_i,
  // Output stream
  output logic      valid_o,
  output logic      data_o,
  input logic       ready_o
);
```

- This still doesn't implement a majority of the AXI Stream protocol…

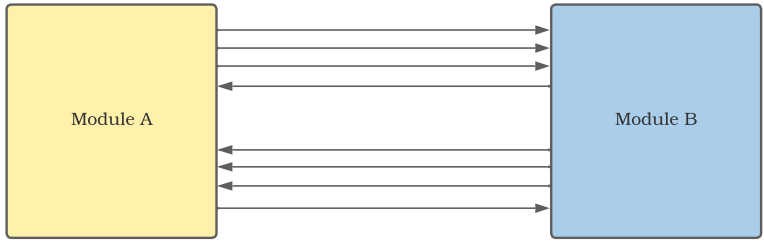
Now, consider hooking that DUT up

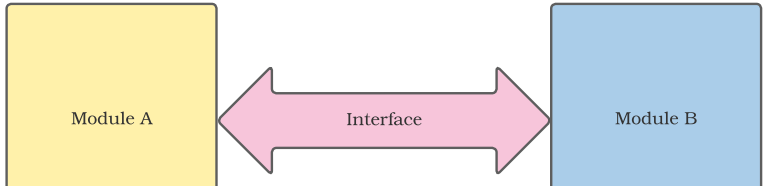
- There's a master of the DUT, and a slave of the DUT.
- These components have their own AXI Streams.
- Naming these signals is getting ambiguous – lots of potential for error.

```
dut dut_i (
  //...
  .valid_i,
  .data_i,
  //...
);
```

```
master master_i (
  //...
  .valid_o (valid_i),
  .data_o  (data_i),
  .ready_o (ready_i)
);
slave slave_i (
  //...
  .valid_i (valid_o),
  .data_i  (data_o),
  //...
);
```

Visually





With an interface...

- Encapsulates the signals associated with a logical “stream” into a bundle.
- Turning a set of names that are often used together into a single name makes life much easier, and RTL less error-prone.
- Verification benefits: interfaces can include tasks, functions, assertions.

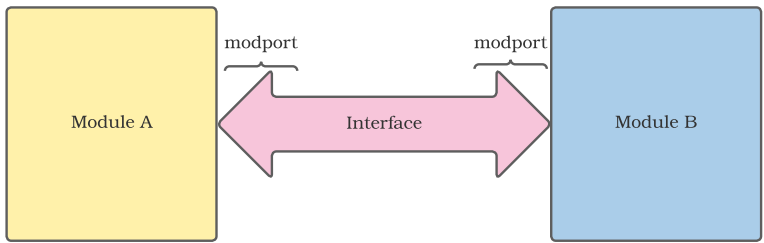
Specifying directions: `modport`

- Think back to the syntax at the beginning:

```
interface axi_stream;
  logic clk;
  logic rst;
  logic valid;
  logic ready;
  // ... and so on
endinterface
```

- This doesn't specify port directions!
- Port directions are different (typically opposite) depending on whether the module you're writing is the master or the slave.

Visually



For AXI Stream (Handshaking)

- Without defining and using `modport`'s, all interface signals are `inout` (bad).

```
interface axi_handshake;
  logic valid;
  logic ready;

  modport master (
    output valid,
    input  ready
  );

  modport slave (
    input  valid,
    output ready
  );
endinterface
```

Usage

```
module module_a (
  input logic       clk,
  input logic       rst,
  axi_stream.master data_out,
);

module module_b (
  input logic      clk,
  input logic      rst,
  axi_stream.slave data_in,
);
```

To instantiate:

```
axi_stream itf;
module_a module_a_i (.clk, .rst, .data_out(itf));
module_b module_b_i (.clk, .rst, .data_in(itf));
```

Parametrizable Interfaces

Interface Ports

Verification using interfaces