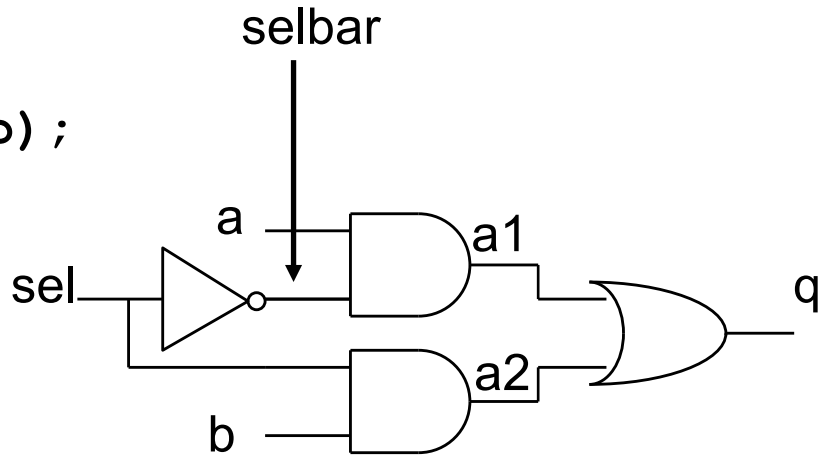
Structural SystemVerilog Design

Structural SystemVerilog Design

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```



Structural SystemVerilog Design

```
module mux21(
        output logic q,
        input wire logic sel, a, b);

        logic selbar, a1, a2;

        not(selbar, sel);
        and(a1, selbar, a);
        and(a2, sel, b);
        or(q, a1, a2);
endmodule
```

Each module definition starts with the keyword "module" and ends with the keyword "endmodule"

All details of the circuit fall somewhere in between.

Structural Verilog Design

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```

The name of the module is specified after the module keyword.

SystemVerilog is case-sensitive…

mux21 is a different module name than MUX21

Structural SystemVerilog Design

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```

The ports of the module are specified after the module name. Each port must be declared in this port list.

In this case, there are four ports to the module:

q, sel, a, b

Structural SystemVerilog Design

```
module mux21(
        output logic q,
        input wire logic sel, a, b);

        logic selbar, a1, a2;

        not(selbar, sel);
        and(a1, selbar, a);
        and(a2, sel, b);
        or(q, a1, a2);
endmodule
```

The direction of each port is declared using the *input* and *output* statements.

For this module, there are three input ports (sel, a, b) and one output port (q).

Structural SystemVerilog Design

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```

Internal signals that connect logic gates are declared.

There are three internal signals declared for this module:

selbar - sel inverted
a1 - output of first AND gate
a2 - output of second AND gate

Structural SystemVerilog Design

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```

Internal gates are instantiated using one of several predefined built-in gates.

The declaration for built-in gates is:

type(output, in1, in2, …);

Built-in gates can take any number of inputs (except not). Built-in gates include:

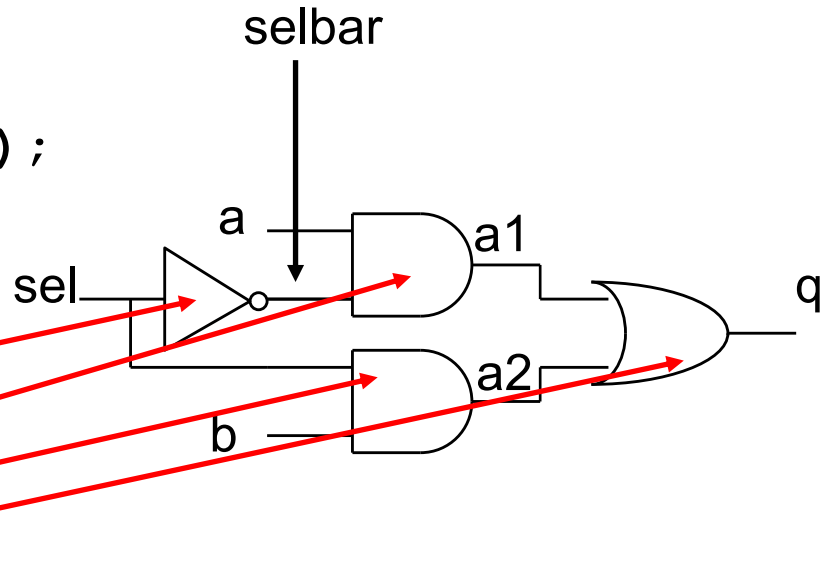
and(out, in1, in2, …);
or(out, in1, in2, …);
xor(out, in1, in2, …);
not(out, in);

Structural SystemVerilog Design

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```



Are these two the same circuits?

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       and(a1, selbar, a);
       and(a2, sel, b);
       or(q, a1, a2);
endmodule
```

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar, a1, a2;

       not(selbar, sel);
       or(q, a1, a2);
       and(a1, selbar, a);
       and(a2, sel, b);
endmodule
```

Other SystemVerilog Issues

- Each statement ends with a semicolon (;)
  - Except the *endmodule* statement
- Comments can be added using C comment style
  - // A pair of slashes for a single-line comment
  - /* Multiline comments also */
- SystemVerilog files are saved in a .sv file (i.e. mux21.sv)
- Save to the same file name as the module name.

Declaration of Wires

```
module mux21(
       output logic q,
       input wire logic sel, a, b);

       logic selbar;

       not(selbar, sel);
       and(a1, selbar, a);
       and(s2, sel, b);
       or(q, a1, a2);
endmodule
```

Input and output wires must be declared.

Declaring internal wires is optional.

WARNING: this means that mis-spelled wire names may not be caught by the SystemVerilog compiler !!!

The code to the left doesn't give a compile error. Finding the error is a bit challenging.

Can you find it?

Using a Macro

```
`default_nettype none
module mux21(
        output logic q,
        input wire logic sel, a, b
        );

        logic selbar, a1, a2;

        not(selbar, sel);
        and(a1, selbar, a);
        and(a2, sel, b);
        or(q, a1, a2);
endmodule
```

Will force compiler to emit errors about undeclared nets.

That is a back-quote, NOT a single quote! It is below the ~ key on most keyboards…

Instancing Modules

- Built-in logic functions
  - **`and`**, **`or`**, **`not`**, nand, nor, **`xor`**, **`xnor`**,
  - Independent of the technology used
  - Do NOT need an instance name

- Cells you have already designed
  - Like **`mux21`** in previous slides
  - Require an instance name

Module Instantiation Semantics

- A SystemVerilog design is *not* a program executed one line at a time

- It is a set of circuit modules that execute concurrently (demand driven)

```
not(selbar, sel);
and(a1, selbar, a);
and(a2, sel, b);
or(q, a1, a2);
```

Old Style

```
module mux21(q, sel, a, b);
      output logic q;
      input wire logic sel, a, b;

      logic selbar, a1, a2;

      not(selbar, sel);
      and(a1, selbar, a);
      and(a2, sel, b);
      or(q, a1, a2);
endmodule
```