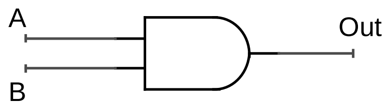
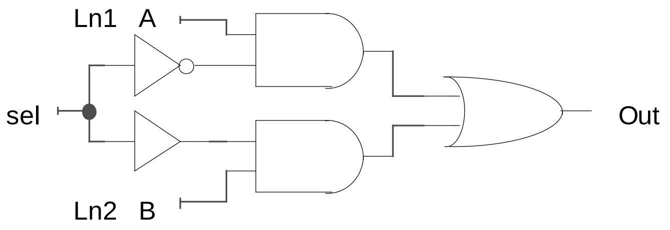
Lecture 33
July 28

Synthesizing VHDL

- The real goal in writing VHDL is to create a (physical) circuit
    - To do this, a VHDL description must be translated into a circuit diagram
    - “synthesis
- synthesis tools very between VHDL implementations
    - none can synthesis all VHDL
    - some can do more than others
- the result of synthesis depends on the target
    - ASIC: application specified integrated circuit
        - A custom chip—target is transistors on silicon
    - FPGA: field programmable gate array
        - A pre-made chip that can be programmed to do anything.
        - Target: a program for the specific FPGA
- Some VHDL has obvious implementations:
    - Out<=a and b;



    - Out<= (In0 and (not sel))
      or (In1 and sel)
        - A 2 to 1 mux



- Some is less obvious

```
If sel = ‘1’ then
        Out <=In1;
Else
        Out <=In0;
End if;

If rising_edge (clock)
        G <= d;
End if;
```

- Most implementation should be able to deal with these
- Structural implantations are (obviously) easy to turn into circuits
    - Good for pre-existing components or getting around limitations in synthe tools
    - Also for hand-optimizing
- Only these std_logic values have meaning in synthesis: ‘0’ , ‘1’ , ‘Z’ ,
    - E.g.

```
if sel = ‘0’ then
    out <= In0,
elsif sel = ‘1’ then
    out <=In1;
else
    out <= ‘-‘;
end if;
```

- The “after” quantifier can’t be synthesized
    - The delay comes from the implementation.
    - It’s there for the simulation
        - To make it look more like the real circuit
        - Assuming delays are guessed well.