

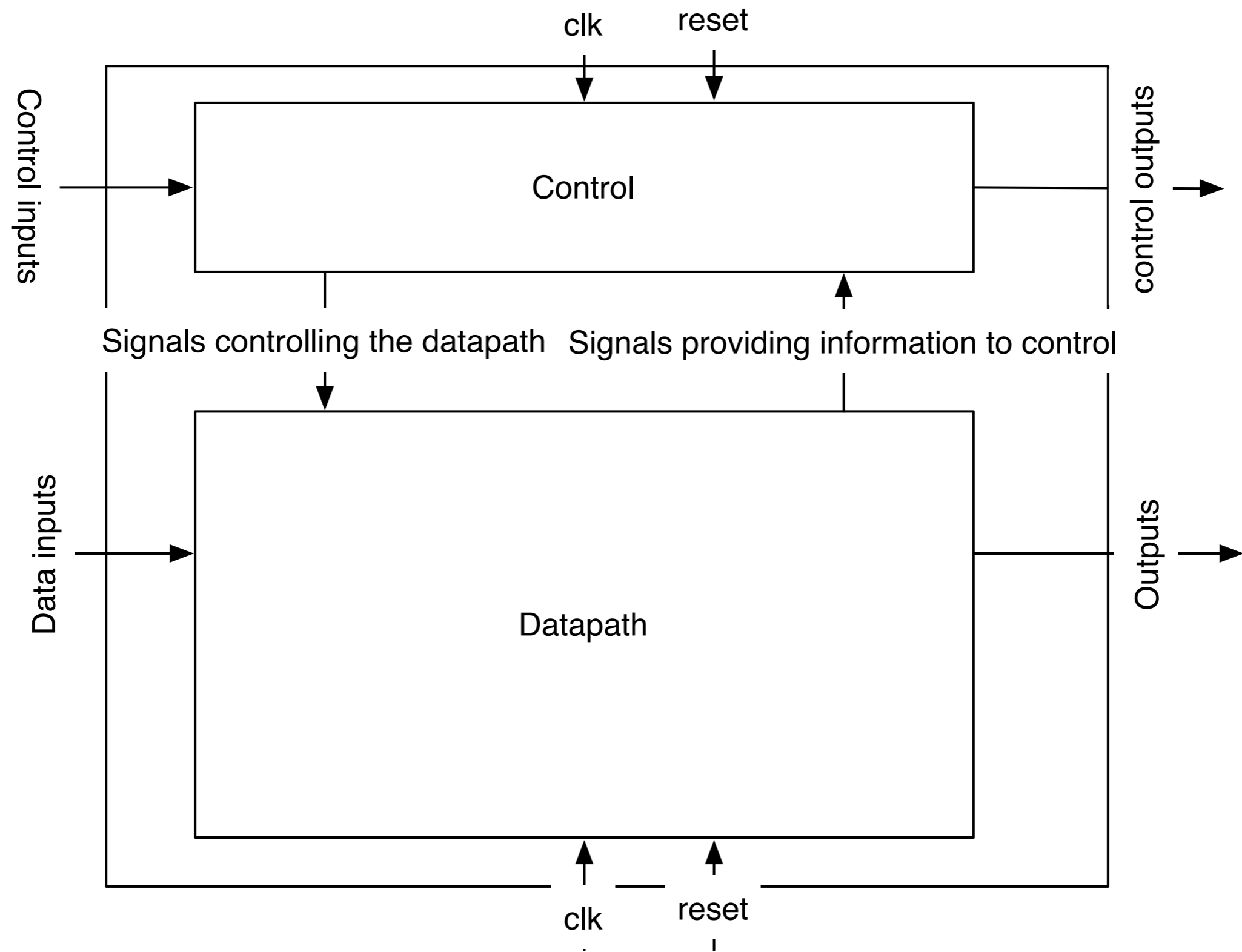
# Control Path Design and Lab 3

# Separating Control From Data

- The datapath is where data moves from place to place.
  - Computation happens in the datapath
  - No decisions are made here.
  - Things you should find in a datapath
    - Muxes
    - Registers
    - ALUs
    - Wide busses (34 bits for data. 17 bits for instructions)
    - These components are physically large
    - In a real machine, their spatial relationship are important.
  - Mostly about wiring things up.

# Separating Control From Data

- Control is where decisions are made
  - Things you will there
    - State machines
    - Random lots of complex logic
    - Little state (maybe just a single register)
  - Spatial relationships are harder to reason about or exploit.
- Because they are qualitatively so different, we will use different coding styles for each.
  - These are best practices from people who build real chips.
  - Following them will save you lots of pain
  - If you don't follow them, and you have problem, the TAs and I will tell you to go fix the coding style issues first.



# Designing the Control Path

- Identify control lines
  - Inputs from datapath.
  - Outputs to datapath.
- Draw out the state machine
  - Transitions are defined by inputs from the datapath.
- Figure out how the output lines should be set for each state.
- Implement!

# Identifying Outputs

- Any element of the control path that has a control input
  - muxes
  - alus
  - enabled registers
- Outputs to the outside world that are not data
  - Data valid lines
- If you designed your datapath correctly, this should be all of them.

# Identify the Inputs

- These are all the bits of information that the datapath generates to allow your design to make decisions.
- If you thought through your datapath carefully, you should already know what these are.

# Draw the State Machine

- The state machine implements the states that your datapath can be in.
- Any activity that takes multiple cycles needs its own state
  - Blocking on IO
  - Multi-cycle operations.
- Any time an output control line depends on anything but the input control lines from the same cycle, you will need one or more new states.
- State transitions are a function of input lines.
  - Write down the formula for each transition.

# Computing Outputs

- For each state, write down how to compute the outputs from the inputs.
- This can be different for different states.
- Make a table that describes how each control line will be computed in every state.

# Implement!

- Now that you have a complete design, you can implement.
- The control unit should be one module with three always blocks and one register.
  - One block computes the state transitions. (always@(\*))
  - One block computes the outputs. (always@(\*))
  - One block implements the register for the state. (always @ (posedge clk or reset))
- Use 'localparam' to define state names. -- No magic numbers! (0 and 1 are not magic)

# Computing State Transitions

```
always @(*)
begin
    // Default is to stay in the same state
    state_next = state;

    case ( state )
        STATE_1 :
            if ( something && something_else )
                state_next = ANOTHER_STATE;
        ANOTHER_STATE :
            if ( sky_falls )
                state_next = SCREAM;
        SCREAM :
            state_next = WAIT;
        ...
    endcase
end
```

# Computing Outputs

```
always @(*)
begin
    //Default control signals
    some_mux_sel_out    = SOME_MUX_SEL_X;
    reg_en_out         = 1'b0;
    case ( state )
        STATE_1:
        begin
            some_mux_sel_out = 1'b1;
        end
        ANOTHER_STATE:
        if ( sufficient_love_in ) begin
            reg_en_out = 1'b1;
        end
        else if ( is_monday_in ) begin
            reg_en_out = 1'b0;
        end
        ...
    endcase
end
```

# Implementing the State

```
always @(posedge clk or reset)
  if (reset)
    state <= WAIT;
  else
    state <= state_next;
```

# GCD Example

- See the 02\_Verilog2.pdf slides from last week.
- And the implementation I uploaded.

