Lab 5 Preparation

Lab 5 Components

- Part I: Create a counter
  - Use the synchronized counter circuit described in lectures.
- Part II: Slow down the clock clock50 50mHtz
  - Use the counter and the on-board clock to create a slower clock.
- Part III: Morse code decoder
  - Decode incoming Morse Code signals!
  - Uses code from Part II ☺

Design Guidelines

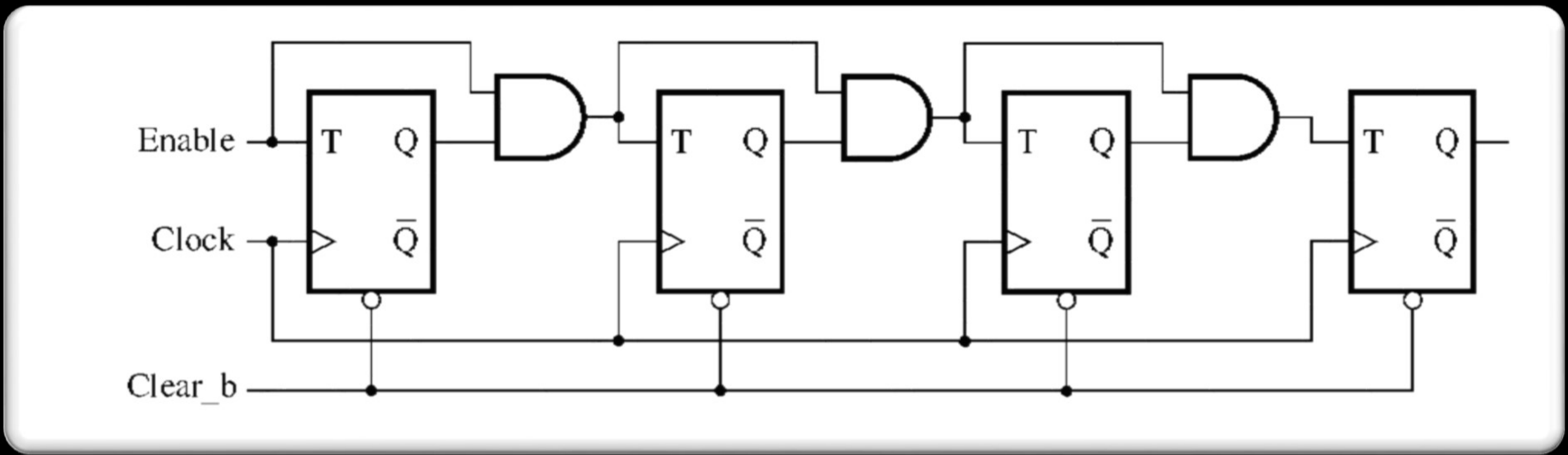
- A note on `always` blocks:
  - Combinational Circuits use blocking assignment statements (the = operator)
  - Sequential Circuits use non-blocking assignment statements (the <= operator)

```
always @(*) begin
  q = d;
end
```

```
always @(posedge clock) begin
  q <= d;
end
```

- Do NOT mix assignment types in the same always block!

Part I – 4-bit Counter



- Diagram shows a 4-bit synchronous counter, made with T-flip-flops
  - The T flip-flops here have an active-low asynchronous reset (`Clear_b`).
- Need to use hierarchical design to make an 8-bit counter.

Part I (continued)

- Prelab parts:
  - Draw and label 8-bit counter schematic.
  - Write Verilog code for flip-flop and counter
    - Don’t use “`tff`” as a module name.
  - Simulate your counter to confirm correctness.
  - Augment Verilog code with input/output layer.
  - Analyze your design!
    - Logic Utilization (in ALMs)
    - Maximum Frequency (F_{max})
    - Netlist Viewer

Tools in Quartus

Part II: More Counters

- Here is an alternate implementation of counters:

```
reg [3:0] q;

always @(posedge clock)
begin
   if (Clear_b == 1'b0)
     q <= 0;
  else
     q <= q + 1;
end
```

- Things to note:
  - Non-Blocking Assignment (sequential circuit)
  - Synchronous Active-Low Clear signal

Part II (continued)

- DE1-SoC has a 50MHz clock (pin `CLOCK_50`)
- Hertz (Hz) => number of cycles per second
  - 50MHz => How many clock cycles per second?
    - Would you be able to see that?

- Goal: Flush digits on a hex display with different frequencies (e.g., 1Hz, 0.5Hz, etc.)

Part II (continued)

- Use a counter w/ a parallel load input to produce an enable signal.

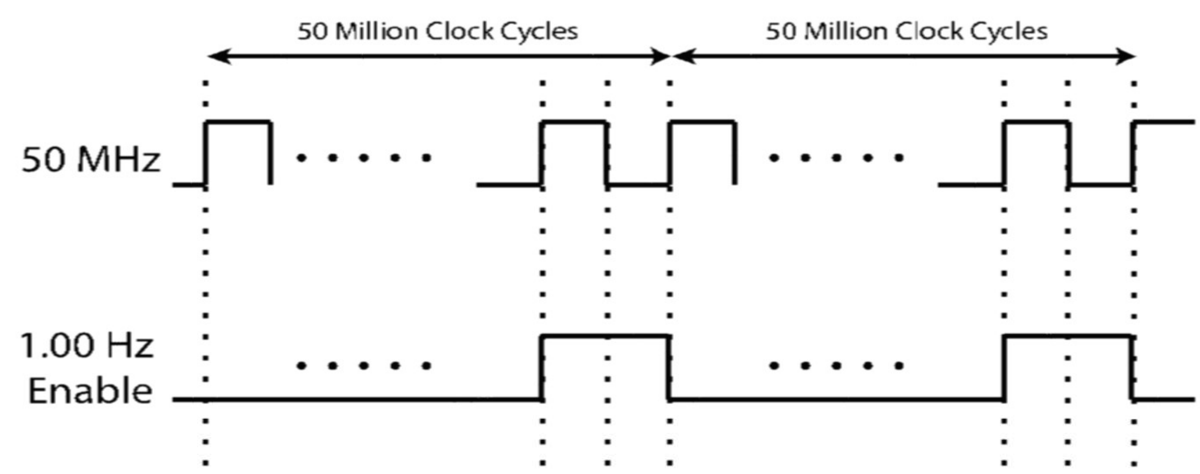


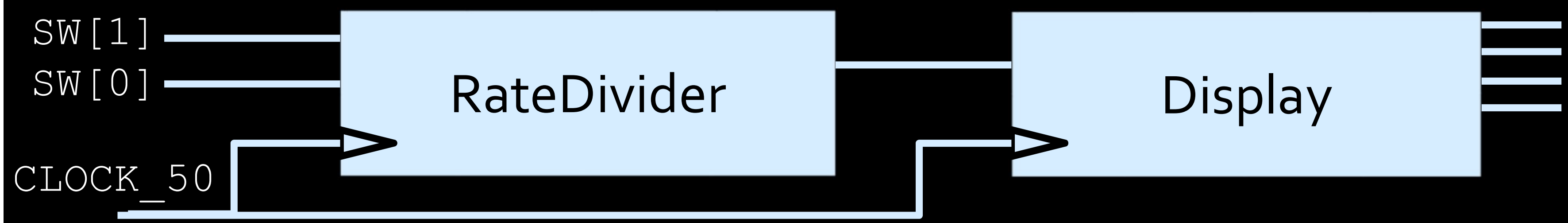
Figure 3: Timing diagram for a 1 Hz enable signal

- The enable signal will control whether the output of another module will change on a clock pulse or not.

by setting enable signal of the other module

Part II (cont'd)

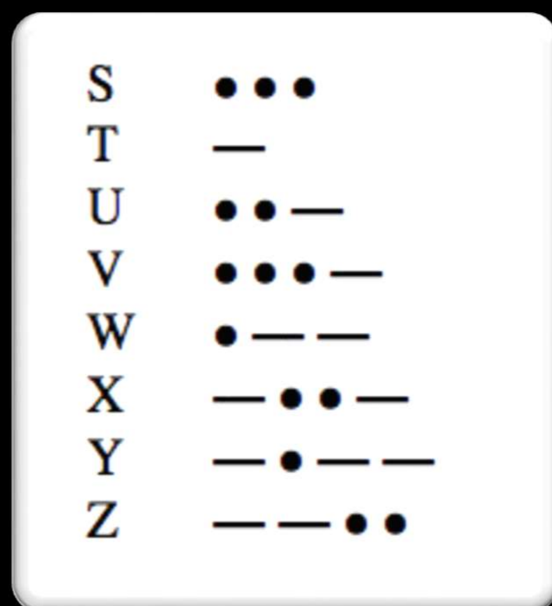
- You will need 2 counters:
  - A `RateDivider` counter and a `Display` counter
    - Both will be synchronized to the same 50MHz clock.



  - Recall what the purpose of an `Enable` signal is in a counter?
    - How often do you want the `Display` counter to increment?

Part III – Morse Code

- Morse Code: "A method of transmitting text information as a series of on-off tones, lights, or clicks that can be directly understood by a skilled listener or observer without special equipment. "



Part III (continued)

- You will be transmitting individual letters using a single red LED
  - Dot => 0.5 seconds LED on
  - Dash => 3 * 0.5 seconds LED on
  - Pause (between symbols or in the end of transmission) => 0.5 seconds LED off

- Switches will specify next letter to transmit
  - When the `KEY` input sends a signal, load a shift register with the appropriate bit values to send into the LED.
  - Use a lookup table to load this register, based on switch inputs.

Part III (continued)

- Deciding on the binary representation
  - Example: • – ("dot dash")
    - Could be represented as `101110` or `10001110` or `10111000` or `1001110000000`
  - Each bit corresponds to 0.5 seconds of light (`1`) or no light (`0`).