ENGR xD52: Midterm

Due before class on 11/17 to comparch14@gmail.com with subject “[Midterm] – Your Name”

Honor Code Policy

This midterm is to be done entirely on your own with no assistance from or collaboration with other humans. However, you do have free access to my slides, your notes, the internet, textbooks, ModelSim, etc. Document exactly which resources you use, including specific URLs.

You may talk to myself or the NINJAs. We will stop offering assistance starting 48 hours before the deadline.

Summary

The goal of this exercise is to faithfully reproduce part of a real product that is currently available for sale as faithfully and as cheaply as possible. For this, we are cloning the LED controller in my bike light.

Specification Document

Write a brief but informative specification document that clearly captures the design intent of the digital electronics portion of the product. This section should include:

- Inputs and Outputs. These are a single button and a single LED, but you need to state that clearly in your document.
- All operational modes of the system
  - Consider showing blink patterns graphically
- Measurements of relevant dimensions in appropriate units with actual numbers
  - E.g. 10Hz, rather than “quickly”

This section should NOT include:

- Any information about the mechanical aspect of the product
- Any implementation details. Explain what it does, not how it does it.

Done well, this should be roughly one page, including figures / charts / FSM diagrams. Another engineer should be able to take your spec document and recreate the device with no additional information.

Hint: I like http://wavedrom.com/editor.html and http://madebyevan.com/fsm/ for simple figures

Block Diagram

This is where you begin to answer “How”. Create a high-level block diagram view of a digital circuit that implements the system laid out in your specification document. Be sure to read the entire assignment before doing this aspect, as there are hints embedded in the scaffolding for the other deliverables. This document is written top-down for context, but can be done in the order that makes you happiest.

Schematic

Each of the components in your Block Diagram needs to be expanded hierarchically. At the bottom of the hierarchy are basic components: NAND, NOR, AND, OR, XOR, XNOR, NOT, Buffer, D Flip Flop, Multiplexer, Decoder. If you require additional components, they are to be built hierarchically.

For each new component you use, provide the following:

1) Specification. This is a 1-3 sentence description of what the component does.
2) Inputs
3) Outputs
4) Schematic
5) Size of the component in terms of the number of Gate Inputs it uses.

Cost Estimation

Estimate the cost of your design in terms of area of silicon consumed. For this, we will assume that cost linearly scales with the number of gate inputs for basic inverting gates. Non-inverting gates cost 1 extra for the implicit inverter.

Gate	Cost
Inverter	1
2 Input NAND gate	2
2 Input AND gate	2+1=3
N Input NOR gate	N
Edge Triggered D-Flip Flop	13

“Pre-Baked” Components

These components are included as scaffolding. You may use them as blackboxes without needing to copy them into your own schematic bundle. **Do not panic** if your design doesn’t use all of these components.

System Clock

Specification

The oscillator sub-circuit provides a 32,768 Hz square wave whenever power is applied. This is used to drive all clocked elements in the system. **DON'T GATE THE CLOCK, FOR SERIOUS Y'ALL**

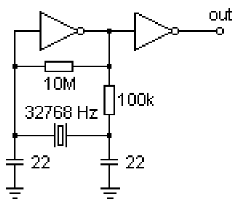
Inputs

No inputs are required for this component, as it free runs whenever power is applied.

Outputs

"clk" is a 32,768Hz square wave to drive the clocked logic in the rest of the system.

Schematic



Cost

This uses 2 inverters for a total of 2 GIE worth of space. The resistors, capacitors, and quartz crystal are all off-die and therefore not considered in this calculation.

Subcomponent	Cost per	# Used	Total
Inverter	1	2	2
			2

Positive Edge Triggered D-Flip Flop

Specification

D	Q	Clk	Q_{next}
0	X	Rising	0
1	X	Rising	1

Inputs

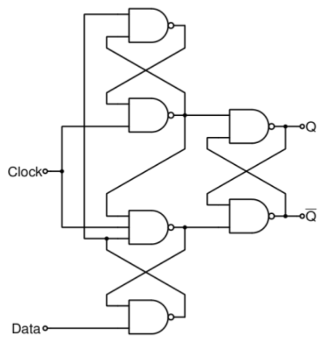
D is the data input. **clk** is the clock input.

Outputs

Q propagates the value of **D** on the positive edge of **clk**

~Q is the Boolean complement of **Q**.

Schematic



Cost

Subcomponent	Cost per	# Used	Total
2NAND	2	5	10
3NAND	3	1	3
			13

Positive Edge Triggered D-Flip Flop with Enable

Specification

D	E	Q	Clk	Q_{next}
0	0	X	Rising	Q
0	1	X	Rising	0
1	0	X	Rising	Q
1	1	X	Rising	1

Inputs

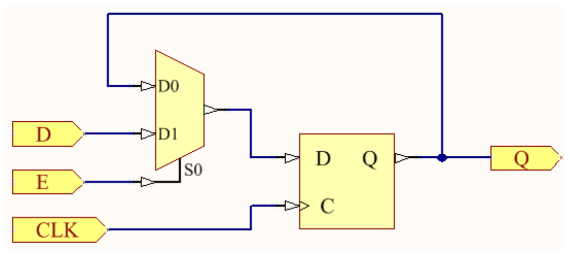
D is the data input. **clk** is the clock input. **E** is the enable input

Outputs

Q propagates the value of **D** on the positive edge of **clk** if and only if **E** is true

~Q is the Boolean complement of **Q**.

Schematic



Cost

Subcomponent	Cost per	# Used	Total
DFF	13	1	13
2Mux	7	1	7
			20

N stage Ring Counter

Specification

This parameterized component is a one-hot counter. When **E** is asserted, the hot bit moves one element down the ring each positive **clk** edge.

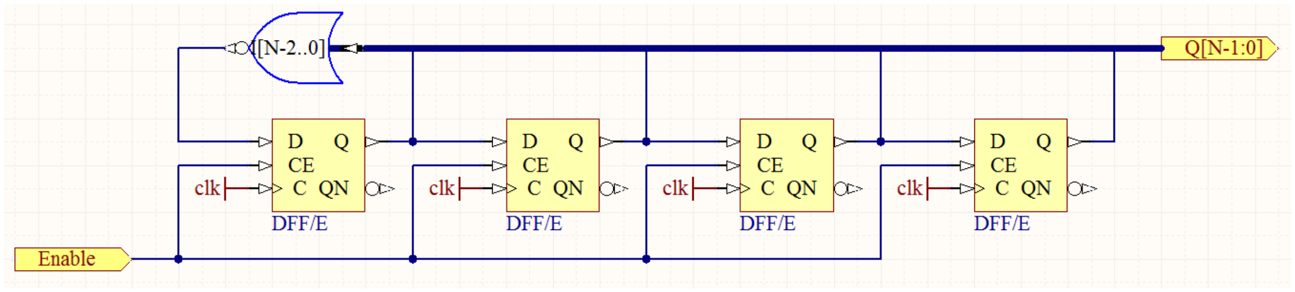
Inputs

clk is the clock input. **E** is the enable input

Outputs

Q[N-1:0] has exactly one bit high, all others are low. This high bit rotates through the group each positive **clk** edge when **E** is asserted high.

Schematic



Note: The NOR gate restarts the cycle when the hot bit is at the end of the ring. This structure obviates the need for a reset signal and is self-cleaning.

Cost

Subcomponent	Cost per	# Used	Total
DFF w/E	20	N	20N
(N-1)NOR	N-1	1	N-1
			21N-1

Partial Components / Hints

Up Counter

An up-counter is the binary encoded version of the one-hot ring counter. This means that it needs $\log_2 N$ D Flip Flops instead of N. Since your oscillator is several orders of magnitude faster than your Blink Patterns, I would suggest using an Up Counter as part of the subcircuit that generates that timing.

For extra fun here, take a look at half adders. They are the smaller cousin of full adders.

Input Conditioner

You will need to condition the button input. Assume that the button noise / bouncing decays within 1 millisecond. This will require careful selection of when to use a binary encoding and when to use a one hot encoding.

Tools

This Midterm does not require the use of ModelSim or the Xilinx ISE – there is no Verilog anywhere on this! Use them if they you'd like, they have tons of helpful features.

Creating the Spec

Our eyes are too slow to see some of the patterns that the light is actually generating. The spec document should capture the human feeling of the pattern, the implementation should capture how it is actually achieved.

You can see these subpatterns either by videotaping the light, or by shaking it back and forth while looking at it through one eye.

Extra Credit 1

Optimize your design for size. Get under 800 GIE for ☺ points.

Extra Credit 2

LED controllers often contain a "breathe" function: Think of a Mac in sleep mode. The LED starts off, its duty cycle is slowly increased to a peak value, and then slowly decreased back to zero. For extra credit, design a digital circuit that will show this behavior. Assume that the human vision flicker limitation is 128 Hz; keep your flicker rate faster than this.

Target a total cycle period of 4 seconds. Linear duty cycle ramping is easiest.