FREQUENTLY ASKED QUESTIONS

April 18, 2017

Content Questions

In the one's catching latch, typically how long is the delay between D changing and the response?

The propagation delay through a latch or flip-flop will typically be a few nanoseconds to a few tens of nanoseconds, and will depend on the logic family and device (and can depend on temperature as well).

What does the behavior of the latch look like at a transition?

There will usually be some kind of voltage slope rather than a sharp transition between 0 and 1 or 1 and 0 (although we are idealizing it as an abrupt transition).

What is the application of latches?

The basic purpose of a latch is to save the state of some particular input. The latches from today's lecture are too simple to do much by themselves, but they can be used as part of more complicated circuits.

As an example from my own research: I might want to know which electronics channels fired when a particle went through my detector. I could design a circuit to latch the signals on each channel when a trigger condition was met, so that I could later read out the pattern of channels that fired.

There are lots of other applications for which one might want to save digital information. In general, latches and flip-flops are used to save the states of the bits that encode binary numbers (one wants one's computer to remember the numbers it computes or it isn't much use...)

What actually happens when the when the RS flip-flop input is 11?

In today's example with the NOR gate flip-flop, if the inputs go to 11, the output at $Q\bar{Q}$ will go to 00, which is allowed. However this is not a good situation, because we will not get stable, reproducible behavior if RS inputs go to 00 from 11— we could get $Q\bar{Q}$ either 10 or 01, possibly wobbling back

and forth before settling (unstably) into one of the two. This situation is to be avoided for practical circuit design.

(Note Eggleston has a NAND-based flip-flop, which has similar behavior but inverted state logic.)

Why are clocked flip-flops used?

Flip-flops in general are used to create circuits with memory, which encode bits or other logic states one wants to save and use for computation. An example would be to save the answers from the adders from last lecture. We'll see other examples next lecture.

Clocked flip-flops are usually used instead of asynchronous ones, because it's easier to make well-behaved circuits when logic transitions only happen at well-defined times, i.e., at the clock transitions. Asynchronous circuits are subject to logic races and glitches.

What exactly is the clock pulse? What happens after the clock pulse?

The clock pulse input to a flip-flop (or any digital logic circuit) is treated as an *independent input* to the device– it's kind of an external control signal. It's usually a square-wave periodic signal from an oscillator. The idea is to define when the transitions at the output should happen. The internal circuitry of a clocked flip-flop is such that the data (D or JK according to the type of flip-flop) only goes through to the output when the clock signal tells it to.

The specific behavior depends on the type of flip-flop. For "statically-clocked" or "level-sensitive" flip-flops, the output depends on whether the clock input is high or low. For "edge-triggered","dynamically-clocked" flip-flops, the output responds to the input only when the clock signal *changes* from 0 to 1 (for positive edge-triggered) or 1 to 0 (for negative edge-triggered) flip-flops.

Could you make the D flip-flop unstable or is it impossible?

Well, the D flip-flop (transparent latch) is set up such that the S and R inputs of an RS flip-flop have an inverter between them which guarantees that you will never get the 11 condition at the input. So the output will always be well defined (if the components are behaving as expected!)

How exactly does a flip-flop store bits?

The basic idea of a flip-flop is that the outputs (Q and $\bar{Q}$) "remember" what bits are on them, until something happens. For the simple RS flip-flop, the outputs are stable until one of the inputs goes positive. For the clocked varieties of flip-flop, outputs only change when the clock pulse comes in (or if an external set/reset bit turns on in the case that the flip-flop has those inputs). So the output stores a bit on it until something makes that bit change. A bunch of flip-flops lined up can store a bunch of bits— that's called a register.

How does the JK flip-flop store memory?

See above question. The JK flip-flop specifically stores a bit on its Q output (i.e., that output does not change– the bit stays put), until a clock pulse comes in at the same time as both J and K inputs are 1.

How does the bit shifting work in the shift register?

The simple shift register we covered has a bunch of D-type flip-flops (transparent latches) lined up, with the output of each feeding into the input of the next. When the clock pulse comes in, the input bit value gets transferred to the output (that's what a transparent latch does). So at each clock pulse, the bit values get shifted along the line.