To solve our engineering problem, we will introduce what we'll call the "digital abstraction".

The key insight is to use the continuous world of voltages to represent some small, finite set of values, in our case, the two binary values, "0" and "1".

Keep in mind that the world is not inherently digital, we would simply like to engineer it to behave that way, using continuous physical phenomenon to implement digital designs.

As a quick aside, let me mention that there are physical phenomenon that are naturally digital, i.e., that are observed to have one of several quantized values, e.g., the spin of an electron.

This came as a surprise to classical physicists who thought measurements of physical values were continuous.

The development of quantum theory to describe the finite number of degrees of freedom experienced by subatomic particles completely changed the world of classical physics.

We're just now starting to research how to apply quantum physics to computation and there's interesting progress to report on building quantum computers.

But for this course, we'll focus on how to use classical continuous phenomenon to create digital systems.

The key idea in using voltages digitally is to have a signaling convention that encodes only one bit of information at a time, i.e., one of two values, "0" or "1".

We'll use the same uniform representation for every component and wire in our digital system.

It'll take us three attempts to arrive at a voltage representation that solves all the problems.

Our first cut is the obvious one: simply divide the range of voltages into two sub-ranges, one range to represent "0", the other to represent "1".

Pick some threshold voltage, V_TH, to divide the range in two.

When a voltage V is less than the threshold voltage, we'll take it to represent a bit value of 0.

When a voltage V is greater than or equal to the threshold voltage, it will represent a bit value of 1.

This representation assigns a digital value to all possible voltages.

The problematic part of this definition is the difficulty in interpreting voltages near the threshold.

Given the numeric value for a particular voltage, it’s easy to apply the rules and come up with the corresponding digital value.

But determining the correct numeric value accurately gets more time consuming and expensive as the voltage gets closer and closer to the threshold.

The circuits involved would have to be made of precision components and run in precisely-controlled physical environments

— hard to accomplish when we consider the multitude of environments and the modest cost expectations for the systems we want to build.

So although this definition has an appealing mathematical simplicity, it’s not workable on practical grounds.

This one gets a big red “X”.

In our second attempt, we’ll introduce two threshold voltages: V_L and V_H.

Voltages less than or equal to V_L will be interpreted as 0, and voltages greater than or equal to V_H will be interpreted as 1.

The range of voltages between V_L and V_H is called the “forbidden zone”, where we are forbidden to ask for any particular behavior of our digital system.

A particular system can interpret a voltage in the forbidden zone as either a 0 or a 1, and is not even required to be consistent in its interpretation.

In fact the system is not required to produce any interpretation at all for voltages in this range.

How does this help?

Now we can build a quick-and-dirty voltage-to-bit converter, say by using a high-gain op-amp and reference voltage somewhere in the forbidden zone to decide if a given voltage is above or below the threshold voltage.

This reference voltage doesn’t have to be super-accurate, so it could be generated with, say, a voltage divider built from low-cost 10%-accurate resistors.

The reference could change slightly as the operating temperature varied or the power supply voltage changed, and so on.

We only need to guarantee the correct behavior of our converter for voltages below V_L or above V_H.

This representation is pretty promising and we'll tentatively give it a green checkmark for now.

After a bit more discussion, we'll need to make one more small tweak before we get to where we want to go.