1. Pre-release history of redstone

Redstone was added during alpha as a simple system of wires, switches, and not gates that could control doors opening and closing, the direction of rails, wether or not a torch was lit, and the activation of explosives. This system had fully functional boolean logic and did a good job of modeling computer control in the game.

A modern player would probably think that redstone is… basically useless at this point. There are barely any components, and the ones that existed seemed simple and not very useful. There were *three* different ways of sending a redstone signal conditionally. The Switch, which turned an input on or off, the button that could send a pulse, and a pressure plate, which would send a signal as long as it was being depressed. In terms of outputs, redstone could open and close doors, turn a redstone torch on or off, and change minecart tracks.

Surely, this is an almost entirely ineffective system for getting anything done, and compared to modern redstone, it might seem that way. However that is *not* the case, because redstone's greatest strength wasn't in its own components and functions, it was in its interactions between itself and other systems.

2. Boats, minecarts, and pistons ("why pistons are different"

Let's look at three systems that interacted with redstone, Boats, Minecarts, and Pistons. Starting with boats and minecarts, since they were around first, and also do a good job of illustrating contrast between the alpha approach to redstone, and the beta approach to redstone.

Boats were once the most useful redstone component, because they are physics objects that are easily controlled using the water physics that were in the game from before redstone was added. With boats you could make devices that *could not* be easily implemented otherwise, like T-flip flops. The same mechanisms used to interface the physics system with redstone for boat control in circuits, also worked to let the player improve the capabilities of boats in general. The EATS road, and other similar projects created during this time would have been much simpler, and less useful if not for redstone.
Boats are, in my opinion, the most elegant example of the alpha redstone philosophy. Through simple changes to the game world, independently useful systems and objects become useful as a result of their *pre existing* functions and properties.
Later in this essay, when the "beta philosophy" around redstone design is defined, there will be a section calling back to this one that explains exactly why it matters that the system works the way described.

Minecarts were, when redstone was added, incomplete. The Furnace minecart system was broken and unfinished, and as a result, Players had to eventually resort to exploiting the scattered and unfinished parts of this system to create effective minecart travel.

Redstone changes only one element of minecarts. Where previously tracks would always be static, now they can switch direction when powered or unpowered by a redstone line. This interaction is *very important* to the history of redstone.
On a surface level, it is similar to redstone opening and closing doors, a simple action that could have been achieved by the player, but is now automatable through redstone. It also serves a similar function to redstone's interactions with boats, in its ability to control the speed, and position of a physics object, even if indirectly.
I think this read isn't particularly far from correct, and given the more limited ability to manage minecart speed (especially across reloading a location) when compared to boats, Redstone had a much stronger effect on minecarts than minecarts had on redstone.

This effect got even stronger in Beta 1.5, and was made overwhelming with a bug fix in beta 1.6. By the time the game entered full release, minecarts were a system entirely reliant on the redstone system to function effectively. The *only* way to use the minecarts for their intended purpose of high speed interaction was by crafting an expensive redstone component, and learning the basics of boolean logic. Minecarts *stopped* being an independent system based on physics interactions, and *started* being a client system to redstone, usable only with, and in many cases only for redstone circuits.

The very next update after minecarts were made entirely reliant on redstone, *Pistons* were added to the game, based on a popular mod at the time. The piston is, in my mind, the final blow to the alpha philosophy of redstone design. It is *entirely* a mechanism for redstone components to affect the world. Pistons cannot operate on their own, or based on some other external stimuli.
The biggest effect of pistons, wasn't actually their intended capabilities, of moving blocks around a small amount, but actually replacing boats in redstone devices. Almost every device that used a boat before now was easier to build, and more reliable to use, if built with a piston. Block Update Detectors, T Flip Flops, and more became systems entirely contained within redstone. Systems which external knowledge of the game mechanics were mostly unimportant to. In Beta 1.7, Redstone stopped being part of minecraft, one of many integrated systems that interacted in interested ways, and instead became a completely different system built in minecraft's engine, which occasionally brushed up against the rest of the game.

3. The philosophies of redstone

I would like to propose that redstone's game design can generally be split into two different approaches, which I will be referring to as "the alpha philosophy" and "the beta philosophy" because I think that those two words' meanings in terms of game design are relevant to the ways the philosophies arose.
I'd like to clear up that I don't… Really attribute this change in philosophy to some cleverness on Notch's part that was somehow overwhelmed by the influence of Jeb and the other developers during Beta. I think the alpha philosophy (like a large portion of minecraft's best design) was something that arose accidentally as a result of the game's playful experimentation during the early phases of development. I think if Notch had continued to be a solo dev, the beta design

likely would have arisen no matter what, as Mojang went back and tried to “clean up” the game for its release.

I’m going to define the “alpha philosophy” of redstone design as follows:

Redstone should act as a boolean logic system, which can be used to control or alter the state of elements in other self-sufficient systems.

And the “Beta Philosophy” as follows:

Redstone should be a system which uses boolean logic and can use specialized single purpose components to directly interact with large portions of the game world directly.

A good example of the difference between the two philosophies is actually a pair of modern components. The Comparator, and the Sculk Sensor
The comparator is a device which is entirely contained within, and provides no utility without the redstone system. It can detect the strength of redstone signals, and has multiple modes that allow it to output different redstone values based on its input strengths.
The Sculk Sensor is an object found naturally in the game world, which emits a signal to other sculk sensors and shriekers nearby based on detected sounds. It *also* emits a redstone signal while doing so.

The comparator is the example of Beta Philosophy redstone design. It is a self contained component which reads the strength of redstone signals. It is *entirely* reliant on arbitrary and intentional implementation of devices emitting varying redstone signals to interact with almost any element of the game world that is not already part of the redstone system (i.e. weighted pressure plates, lecterns, chiseled bookshelves)

The Sculk Sensor is a piece of Alpha Philosophy redstone design. It serves an independent purpose, unrelated to redstone, but a clever player who is willing to experiment, is able to use it as a component in redstone circuits (noise activation), and to manage its activity using redstone circuits (noise machines). It is fully a part of an independent system, unrelated to redstone, but is able to interact with redstone.

I think the sculk sensor would have been a “stronger” example of Alpha Philosophy design if they didn’t output a signal, and a Block Update Detector was required to make use of them as inputs… but that is a nitpick, and also irrelevant now that Block Update Detectors themselves have been replaced by integrated circuits.

4. Integrated Circuits

An integrated circuit, in this context, is a craftable device which replaces a function which could previously have been created with a handmade circuit of pre-existing blocks. The simplest example of an integrated circuit is the repeater. Before beta 1.3, a repeater was a device that

was created by clever players using redstone torches. But, the repeater as a block directly replaced that kind of device for all players.

If you are a fan of modern redstone, you may be astonished, and perhaps even angry at me claiming something like “integrated circuits are bad design” when comparators, observers, repeaters, and copper bulbs are seen as beloved and important. But, I want you to not think of the convenience provided to an already experienced player, but the way that redstone interacts with the rest of the game, and how it is presented to an inexperienced player.

Integrated circuits are essentially, replacing gameplay with a menu interaction. Where before an IC’s introduction, the process of making Block Update Detector, or repeater, or t flip flop was one of gameplay, where players would interact with the game system in a meaningful and interesting way to solve a problem. After the introduction of those devices, the player thoughtlessly crafts a simple block to create the same effect, the same way, every time. The gameplay of problem solving that once existed, has been replaced for the sake of making certain obscure and niche devices more convenient to build.

When Mojang replaces gameplay with menus for item acquisition (such as with villagers) it is generally seen as weak and annoying design, but there is not similar backlash around these decisions in redstone. Why?

5. The Isolation of redstone

Have you ever seen that popular portrayal of minecraft in memes and the such that presents a multiplayer gaming community in a Minecraft server as having niches. Miners, adventurers, builders, redstoners, villager experts. There are nearly as many disparate player types as there are gameplay systems to interact with… now, I am going to talk in this video about something not every viewer might be familiar with, and I ask you to either go try this yourself, or if not, believe my extensive firsthand experience on this topic to be accurate.

If you play Minecraft alpha, there will not be the same experience. Players will not fall into niches after the first few days. Instead, what I see in Minecraft alpha, is that if players are engaged for long enough to keep playing for a while, that they’ll end up engaging with all of the game’s systems in similar proportion.

Every player will make *some* redstone, and *some* ambitious builds, and do *some* exploration, and *some* mining.

Now before I get into specifics, I want to mention that if you do try playing Minecraft alpha, especially multiplayer, you shouldn't play vanilla, but instead a continuation mod called NSSS that (in a very lightweight and sometimes unnoticeable way) expands on the design and content of Minecraft alpha. I would recommend playing it if you want to play a more stable, and more fleshed out experience for Minecraft alpha, I personally play it over minecraft alpha most of the

time, and this video was partially inspired by my hopes for where that mod goes, and how I think about minecraft is often directly inspired by my experiences playing NSSS.

Moving on.

Minecraft, in Alpha, was a game that encouraged interaction with, and understanding of, every system in the game. The building, engineering, and adventuring systems of the game locked into each other to create gameplay where a player interacts with all of them.

In the thesis video of this series I will go more in depth on how these loops as a whole interact, or don't, but for now I'm going to focus on the engineering aspect of the game.

I am being careful here to call the mechanics in alpha engineering, as opposed to a narrow reference to the redstone mechanics that are nowadays the dominant system of engineering gameplay. I made this choice because I feel like in alpha, engineering was a process that involved an understanding of multiple systems, of which redstone is often only a small section of the engineering work done.

For a good example of “engineering” separate of redstone, EATS is a device which is primarily made up of the boat mechanics and water physics, but uses redstone in a limited capacity as a control system to manage path choosing. Minecarts similarly functioned independently, while still interacting with redstone. And some kinds of engineering, like monster farms, very often involved no redstone at all.

Engineering is often the “late game activity” that players will spend the most time engaging with. as a player’s control of the world extends further, and their infrastructure becomes more extensive and unwieldy, automation and engineering become more important to players, and they are directly driven to automate and improve that infrastructure through those mechanics.

Walkways will become railways, stairs will become boat elevators, doors will be redstone operated, farms will be cleared with one click, instead of dozens.

This is not the fascination of a few devoted fanatics, or a game mechanic so overwhelming and self contained that many players can interact only with it to the exclusion of the rest of the game. This is a core part of the game’s progression and standard gameplay, which every player who plays the game for long enough is going to be pushed to interact with by the game design. And it being otherwise, is a *flaw* in the game, in my opinion, one of its most major.

6. Conclusion and summary

It is my belief, and the argument of this essay, that over the last thirteen or so years Mojang has made a set of decisions which exchanged the learnability, variety, wider gameplay focus, and player expression of minecraft’s engineering mechanics for the sake of making what was once one small component of those mechanics more reliable and compact.

This video is not a “fixing redstone” video, because I think that at this point minecraft has gone so far in another direction that “fixing” this mechanic to function in concert with the game's other systems is impossible, and would probably annoy so many more people than it pleases that it would never be done, and likely shouldn't be done.

This video is also not standalone, it's part of a series, and the full conclusion of this essay series might not be satisfyingly reached until the thesis video, so I apologize for the seeming lack of conclusion beyond the analysis itself in this video.
That said, thank you for watching, if the other videos in this series are out, make sure to check them out, and if not, you almost definitely already know how to make this website tell you when someone releases a video.