A brief history of the four industrial revolutions, and the tooling that facilitated them:

The advances that fuel each revolution can be characterized by notable refinements in precision. A key innovation in tooling that fostered the first industrial revolution was precision machine tooling, and the master plane gauge; precision (hand-scraped) gauge blocks accurate to within millionths of an inch. The master plane gauge provided a reference with which to manufacture and calibrate machine tools with high precision. While machine tools existed long before the 1820s, this advance in precision vastly improved the quality and feasibility of machine production.

The second industrial revolution can also be considered in terms of an advances in precision, but the precision was not necessarily related to the fabrication of parts, but to the design of the manufacturing process itself.

Definition of Tooling:

* “Tooling” refers to tools expressly developed and used for manufacturing or crafting.
* Tooling is an important subset of technical instruments.
* Technical instruments designed for broadly defined space of intentions
* The designers of tooling specify what the tools are capable of, but not what they are intended to produce.
* The scope of a given tool’s abilities are well-defined, but it is designed with a flexibility of application. Flexibility is a driving intention.
  + This could mean flexibility of function, and of configurability.

Industrial revolutions can be characterized by the production of technical artifacts that remap humankind’s relationship with nature. The principal component responsible for industrial revolutions is precision of tooling. Between periods of revolution, industrial progress realizes its technological potential within the boundaries defined by its tooling, in contrast, industrial revolutions occur in response to the expansion of these boundaries.

It’s important to note that innovation in tooling is not a discrete process, and while industrial revolutions can be described in terms of discrete states, the ideas and innovations that drive them are continuous, and precede the revolutions by decades or more.

*Talk about the lapping blocks and the transistor.*

Houkes & Vermass (2014) propose two classes of artifacts, “instruments” and “products.” Instruments are used to achieve an intended end, whereas products are the end result of a production process. (I don’t know if I have the ideas straight enough to situate tooling in terms of their argument on natural kinds, but might see about trying). While Houkes & Vermass make a valid argument for these two types, it is fruitful to attempt to identify subsets within the instrument and product classes. Tooling clearly is a subset of the instrument class, and like scientific instruments, I believe tooling holds a separate and identifiable position in the ontology of technical artifacts.

Relative to Krouse:

I will argue that tooling is a subset of the universal set of technical artifacts, identifiable because its function is the production of other artifacts. –Maybe check out Krouse’s argument that technology is a life form.

Scientific Instruments:

Use the argument on scientific instruments to develop the argument of why tooling is an identifiable subset. -This should provide a good framework in terms of phrasing and terminology.

Software as a Make Plan:

* Conventional computers are limited by the instruction sets of their processing units, but these are formal constraints rather than practical ones. A Turing-complete system can approximate any function, so a computer is only limited by its outputs and timescale.
* In the space of all possible software, we have an infinitely extensible make plan.
  + Read back to the MP paper to get some arguments going

Hardware & Agency

* Most matter is programmable, and all tools are (the human swinging the hammer is the software) but tooling has been fashioned expressly to be programmable.
* Computers store and sculpt information. The logic gate is the cutter, memory is the fixture, instructions are the machines, and software is the make plan.
* CPUs don’t require that many degrees of freedom to maintain flexibility. x86 ISA has vastly more operations than RISC-V, and we are seeing a rise in specialized ICs that are designed to fulfill more specific intentions (e.g. tensorrent’s RISC-V hardware).
* The push towards specialization is including a lot of inference-only hardware, interesting in that it has a strictly defined capability to run software (neural networks) that are defined by their flexibility to approximate any function, and have been shown to be Turing-complete without requiring access to external memory.