Tip forming - per-material tables (`[ace_tipform]`)

multiACE | configurable unload pull choreography for the Snapmaker U1

When a filament is unloaded, the last centimetres decide everything: the molten end must be **stretched into a slim, clean tip** as it leaves the melt zone. A blobby or stringy tip jams the next load or drags threads through the path. This is *tip forming*, and until now the U1 had exactly one choreography for every material (the firmware's `INNER_FILAMENT_UNLOAD` macro).

multiACE makes the **pull moves** of that choreography configurable per material. Everything around them stays stock: driving to the discard position, heating, the cooling fan, the ooze scrape-off against the discard edge (the U1 has **no cutter** - the tip is formed by the pull alone), the nozzle wipe and the debris shake-off. The **background swap** uses the same tables, so one tuned table serves both paths.

> **Where to edit:** Web UI → Config tab → *Tip forming (per material)* - tables are validated before saving, an invalid table can never halt Klipper. Each row has a **Stock** button (drops in the firmware's stock pull as a starting point) and a **wrench** button that opens a visual token builder (add/remove steps with a type dropdown; it writes the string for you). Or edit the `[ace_tipform]` section in `printer_data/config/extended/ace.cfg` by hand. Changes apply after a Klipper restart.

---

1. The token language

A table is one line of comma-separated tokens, executed in order:

Token	Meaning
`<mm>@<feedrate>`	Extruder move. Positive = push (purge/ram), negative = retract. Feedrate in **mm/min**.
`pause:<ms>`	Dwell. Waits for all moves to finish first.
`temp:<C>`	Set the nozzle target - **no wait** (like `M104`).
`waittemp:<C>`	Set the target and **wait until the nozzle is there, both directions** (100-350 °C). The SkinnyDip building block. **Moves after a waittemp below 175 °C are rejected** - Klipper refuses extruder moves below `min_extrude_temp` (170), so true cold pulls are not possible on the inline path.
`fan:<0-255>`	Part fan. **Inline only** - a parked head's fan is not addressable, the background swap skips this token.
`unloadtemp:<C>`	**The unload temperature for this material/vendor** (175-350 °C). Not a move step - it is extracted as a parameter, so the whole unload (heat-up, the pull, the probe pre-warm) runs at this temp instead of the load temp (default 250). A table may consist of **only** this token: then the stock pull choreography runs, just at your temperature.

Rules enforced at save/startup:

- A table must contain at least one move and must end with a **net retract** (sum of all moves < 0) - the tip has to leave the melt zone or the next load jams. *(Exception: a table that is only an `unloadtemp:` token needs no move - it keeps the stock pull, just at your temperature.)*
- Single moves are capped at 200 mm, feedrates at 6000 mm/min.
- `unloadtemp:` must be 175-350 °C (below `min_extrude_temp` the pull moves cannot run). The stuck-tip **retry** re-unload still heats to `max(load, unload)` - the freeing pull keeps full power even when you set the normal unload cooler.

Lookup order per unload: `<vendor>_<material>` → `<material>` → `soft` **(flexible filament)** → `default` → **stock**. The material key is the filament **type as declared on the slot**, lowercased (`pla`, `petg`, `abs`, …). Anything without a matching table behaves exactly as before - you can tune one material at a time, everything else stays stock.

Optional vendor axis. A table can be keyed to a specific vendor plus material by prefixing the vendor: `prusament_pla`, `prusa_research_petg` (the vendor is the slot's declared **brand**, lowercased, internal spaces become _). It wins over the plain `<material>` table, but **only when the slot actually carries that brand** (from RFID or a manually set identity) - a generic or brandless spool falls straight through to `<material>`, so adding a vendor table never disturbs your generic spools. In the web editor the vendor is a small optional field next to the material; leave it empty for the plain-material table.

V2 (ACE 2) coordination is automatic. An ACE 2 cannot freewheel, so multiACE arms forward-assist around pushes and dispatches a matching ACE unwind alongside every retract ≥ 3 mm. Small cooling wiggles (< 3 mm) run uncoordinated on purpose. You never encode ACE behaviour in the table - only the extruder side.

> **Temperature is the strongest single knob - start here.** Before tuning ram/pull/cooling moves, try lowering the unload temperature. Stock pulls PLA at 250 °C (the DB load temp); cooler PLA forms a **slimmer tip that bulges less**, so it re-loads through combiners and PTFE without jamming. A reporter went from "rarely >2 clean reloads" to ">5" just by dropping PLA to ~220 (their print temperature); Prusa's MMU shapes PLA tips at 200-220 for the same reason. The one-liner:
>
> ```
> pla: unloadtemp:220
> ```
>
> keeps the stock choreography and only lowers the temperature. Add a full table later if you also want ramming/cooling moves.
>
> *Related, load side:* `seat_overshoot_length` in the `[ace]` section (a hot press that seats the tip the last mm into the gears after a load) - not a tip-forming table, but the companion knob when re-loads are marginal.

---

2. The stock choreography, as a string

This is byte-for-byte what the firmware does today (nozzle ≥ 0.3 mm, non-flexible), expressed in table form:

```
default: 57@400, 3@1500, -27@2700, -5.5@40, -37.5@1500
```

Segment by segment:

Segment	What it does
`57@400`	Slow purge. Pushes fresh, clean material into the melt zone so the tip is formed from new filament, not printed-out residue.
`3@1500`	Short fast push - pressurises the melt right before the pull.
`-27@2700`	**Fast pull** out of the melt zone. Speed matters: the molten column must break away instead of drawing a long thread.
`-5.5@40`	**The actual tip stretch.** 5.5 mm at a crawl (40 mm/min ≈ 8 s): the still-soft end is slowly drawn thin and solidifies into the slim cone. This slow segment IS the tip shape.
`-37.5@1500`	Clears the formed tip up through the heatbreak.

The flexible-filament variant (`soft`) is the same without the big purge:

```
soft: 5@600, -27@2700, -5.5@40, -37.5@1500
```

Two practical dials on the stock shape:

- **Tip too long / too thin, or fine hairs at the very end** → shorten or speed up the slow-stretch segment (e.g. `-5.5@60`).
- **Tip blobby / mushroomed** → more purge (`60@400`) so the tip forms from fresh material, or a faster main pull (`-27@3000`).

---

3. Prusa-style, as a string

Prusa's MMU forms tips differently: no long slow stretch, instead **ramming** (a sharp pressure spike so the tip snaps off clean) followed by **cooling moves** (oscillating the filament in the cooling zone while the tip solidifies). Expressed in tokens:

```
petg: 30@400, 0.7@1600, 0.7@2200, 0.7@2800, -27@2700, pause:200, 2@600, -2@600, 2@600,
   -2@600, -35@1500
```

Part	Tokens	What it does
Purge	`30@400`	Fresh material for the tip (smaller than stock - ramming needs less).
Ramming	`0.7@1600, 0.7@2200, 0.7@2800`	Short pushes at *escalating* speed - the pressure spike that makes the molten end separate cleanly. Add steps / raise speeds for a harder snap.
Main pull	`-27@2700`	Out of the melt zone, same as stock.
Settle	`pause:200`	Lets the melt relax before shaping.
Cooling moves	`2@600, -2@600, …`	The Prusa signature: the filament end oscillates in the cooling zone while it solidifies - shapes the tip and keeps it from sticking. More pairs = more cooling = firmer tip.
Clear	`-35@1500`	Up through the heatbreak.

> This is a **starting point**, not a calibrated Prusa profile - Prusa's numbers are volumetric and tuned to their hotend geometry. Expect to iterate.

SkinnyDip (temperature-assisted, against fine hairs)

The community SkinnyDip technique adds temperature control: cool the nozzle for the toolchange (cooler = less stringing), and after the cooling moves **dip** the tip briefly back into the melt zone to burn off the remaining fine hairs, then extract fast:

```
pla: 30@400, 0.7@1600, 0.7@2400, -27@2700, fan:255, waittemp:180, 2@600, -2@600, 2@600,
   -2@600, 28@2000, pause:100, -28@2000, waittemp:220, fan:0, -35@1500
```

Part	Tokens
Purge + ram + pull	as above
Cool down	`fan:255, waittemp:180` - fan on, wait until the nozzle is at 180 °C
Cooling moves	`2@600, -2@600, …`

Part	Tokens
The dip	`28@2000, pause:100, -28@2000` - plunge the tip back to the melt zone, let the hairs burn off, extract fast. Depth is hardware-specific: too deep = blobs on the next print, too shallow = strings survive. Start ~25-30 mm on the U1 and iterate.
Restore	`waittemp:220, fan:0` - back to working temperature
Clear	`-35@1500`

Background-swap caveat: a parked head has no addressable fan, so `fan:` is skipped and `waittemp:` cools **passively** - a big drop can take minutes there. For tables that should run in background swaps, keep temperature drops small (or accept that the inline path benefits more).

For reference: Prusa's own testing found plain temperature-lowering during *ramming* does not improve tips - the wins reported by the community come from the **dip**, not from ramming colder. Treat `waittemp:` as the SkinnyDip enabler, not a general improver.

---

4. Tuning workflow

1. Web UI → Config → Tip forming: set `mode: custom`, add a table for ONE material (everything else stays stock - clean A/B).
2. **Save + restart Klipper.**
3. Load the head, then unload it (dashboard Unload button or `ACE_UNLOAD_HEAD HEAD=n`). The console logs `custom tip form (<material>, N tokens)` - if that line is missing, the custom table did not run (wrong material key or restart missing).
4. Inspect the tip on the pulled-back filament and compare against stock.

Symptom	Dial
Long thin hair at the very end	more cooling moves; or SkinnyDip
Blob / mushroom head	more purge; faster main pull; dip shallower
Tip too long, flag-shaped	shorten/speed up the slow stretch (stock style) or add cooling moves (Prusa style)
String bridges to the nozzle	cooler toolchange (`fan:255, waittemp:…`), faster extraction
Next load jams	tip too fat → see blob dials; check net retract clears the melt zone

A stuck/failed unload is always recoverable: the normal retry (hot re-unload) and pause semantics apply unchanged - a bad table costs you a retry, not a print.