

FullAuto parallel fluid assline v0.4
with a single ME interface

disclaimer: I'm not an expert just really interested in the pack. The testing and results based on personal experiences while reaching ZPM. Feel free to correct my mistakes or improve the design.

Szajkop#0522

The design is inspired by "Alexdoru" - <https://youtu.be/xXyrfBmeTBI>

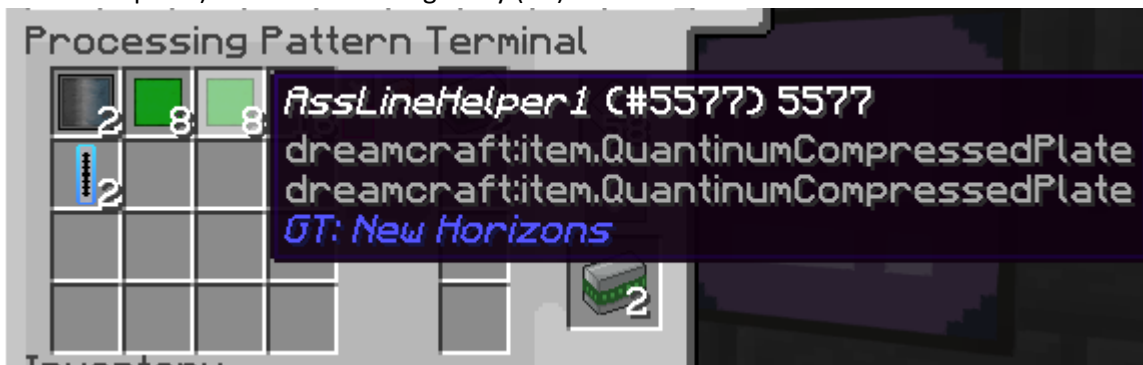
Working demo: https://youtu.be/8q9AXvagr_8

Advantages

- Using low tier components, to build it the biggest gate the assline itself
- Parallel processing up to "infinite" amount of asslines (*some more testing needed :D*)
- Can run from a single AE2 interface
- Full support for fluids and split itemstack recipes

Pre-requirements

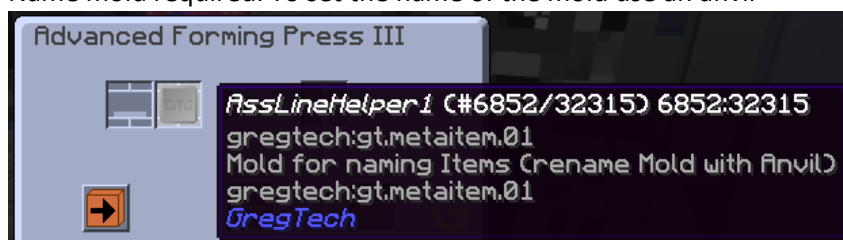
- **Forming press** with Mold(Name) to prevent items to stack. (needed for some crafts eg.: T5 rocket alloy or fusion computer). Later a Processing Array (PA) is recommended to mass rename items.



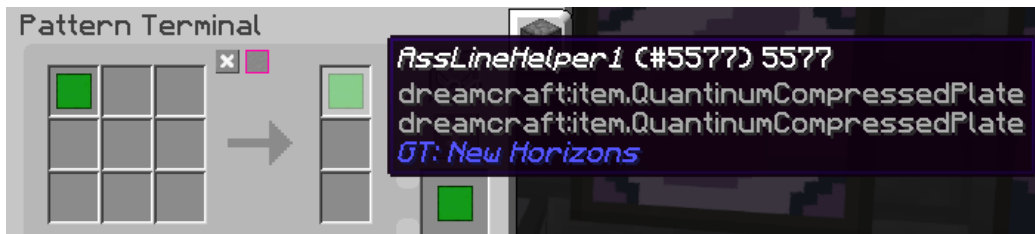
- **Volumetric flasks:** to insert fluids. Multiple mB values are needed: assemblers to "program" then fluid canners to fill. Note: stack size is 16, recommended to use the largest possible option. Only one stack of a single fluid type can be used.
- **Processing pattern terminal:** when the regular 9slot terminal is not enough

Renaming items

- preventing materials to merge in a recipe
- Forming press automation
 - Name mold required. To set the name of the mold use an anvil



- Write an AE2 pattern to make a renamed item from a regular item.



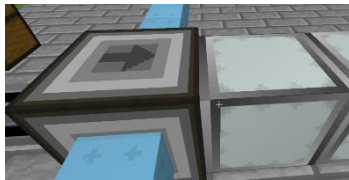
-
- Use ME interface on the forming press
- PA automation
 - after building the PA it can rename stacks of items in seconds
 - Controller needs to be set via a screwdriver to separate input buses. This will allow you to use many different rename molds.
 - input buses needs to be set via screwdriver to turn off input filter.

Writing patterns

- **Order of items** is important, pay attention to stack sizes (prevent stack merging).
- Fluids (flasks) can be placed anywhere in the recipe but the **order of fluids** is also important.
- When not possible to use a single flask (when fluid requirement is so low) set the pattern to craft multiple items at once with a single flask.
- Maximum recipe complexity: 15 items and 1 fluid. When more fluids required in the crafting, the “normal item” count is decreased.
- Its enough to make a single pattern.

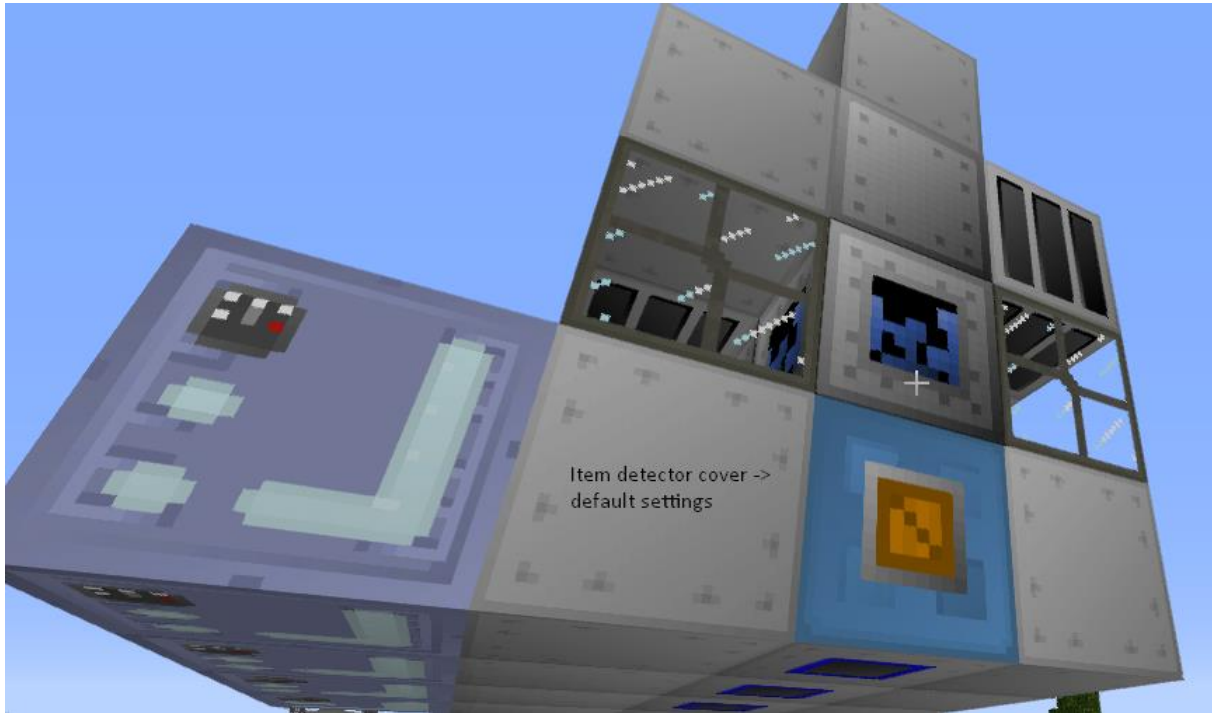
Working principle

1. Items from the pattern enters the ME interface with **blocking mode enabled** and the “output arrow” set to the main pipe.

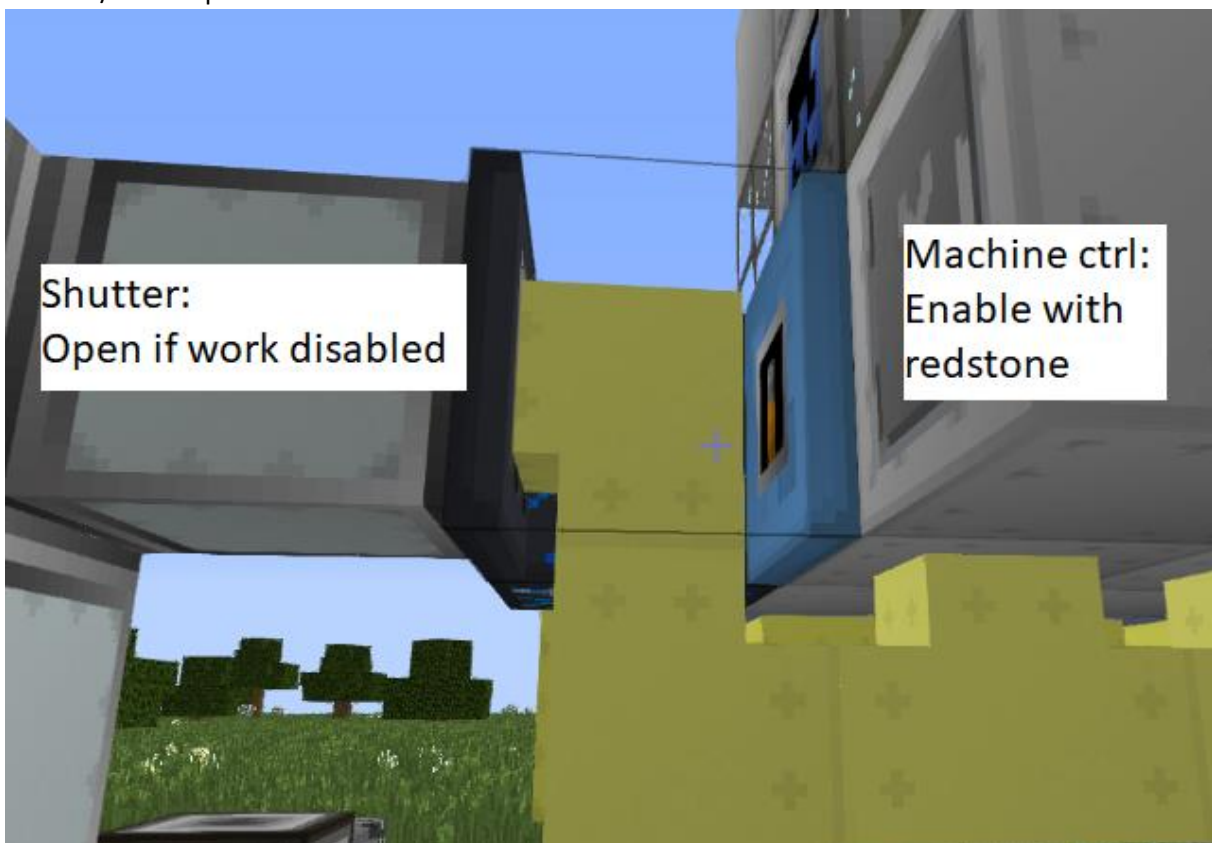


2. With the blocking mode enabled, only items for a single recipe will enter the item pipe system at a time. *When there is an order waiting for an empty assline the first pipe segment will hold the items and prevents the interface to insert in a new order. We need to move the entire assline recipe as a single burst into the assline bus-connected pipes so the pipes have to handle the 16 items all at once, thus I use huge PVC pipes here. (they move 16 stacks/sec). Higher capacity pipes can also be used. Once all of items found its destination the items for the next recipe gets inserted.*
3. The items “travel” inside the main pipe, which needs to have as high routing value as possible. I am using restrictive pipes to increase the routing value to 204800 on each segment. Each output from the main pipe needs to have a unique pipe segments before the exit. (eg.: to reach the first assline I use 6 huge PVC to reach the second assline I have to use anything else than 6 to prevent collision. 7 is fine)
Note: Item handling in GT pipes is done by calculating routing values based on the pipes routing value and the possible paths an item can take. Each item will exit at the smallest routing value (if it is not blocked)
4. The main pipe connects to a different pipe system via a shutter cover. In this item pipe system, the routing values has to be as low as possible. In the demo I use fluxed electrum item pipes which helped me in troubleshooting but you can use regular huge PVC pipes without the restrictive modifier (huge pipes block access by hand to the connected input buses)

5. The assline is ready to accept new items when its first ULV input bus is empty. This is checked by an item detector cover.



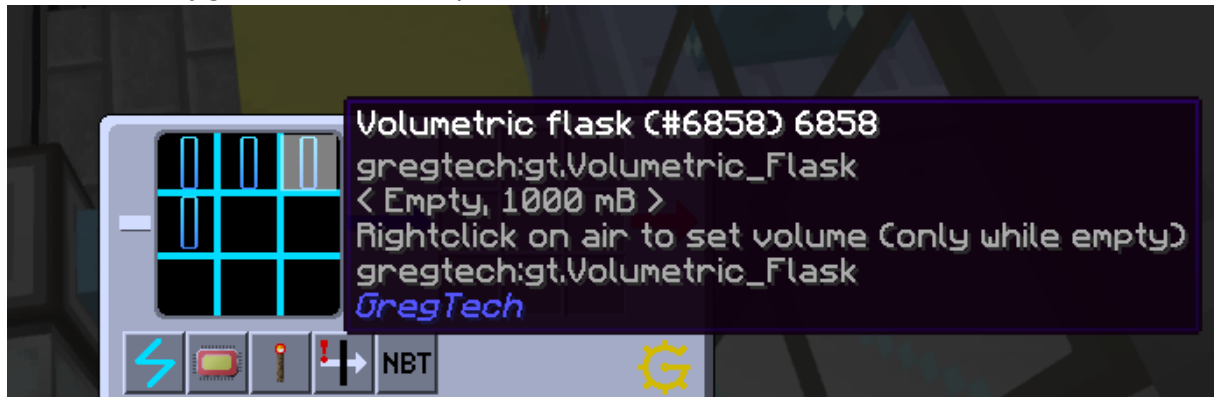
The signal from the cover gets picked up by a machine controller cover on the pipe and it controls a shutter to block/allow input



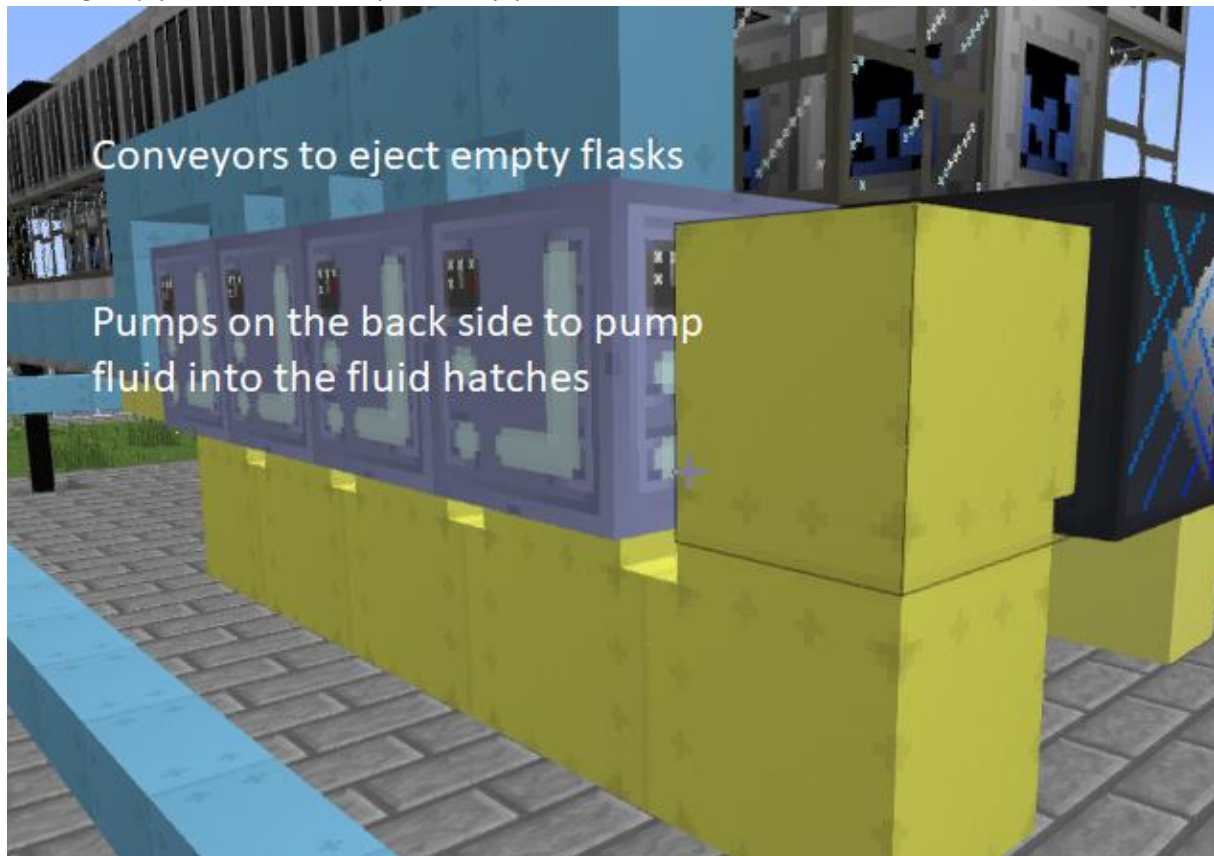
6. The first output needs to be connected to a GT item filter. The voltage tier is not relevant. This way all fluids will enter this path. Configure with 4 volumetric flasks and set the NBT to be ignored. This way this filter can accept the 4 flasks at once and can output them at once while preserving the order. Here any type of pipe that has at least 4stacks/sec throughput can be used.

Note: this limits the fluid input to max 1 stack of flasks per type. Also, to use higher capacity flasks they also

needs to be configured in a similar way.



7. The flasks then enter in the fluid tanks where the fluid gets pumped inside the assline hatches and the empty flasks gets piped out to the output (blue pipes).



8. The rest of the items will enter to the ULV input buses in order, the shutter will close and soon the assline starts to work. When it picks up the items the shutter will open and it will queue 1 recipe. While the shutter is closed the next shortest path is used (the 2. assline)
9. When the processing is completed the output enters the blue pipes and gets inserted to the ME system by the main interface.



Summary

The main bottleneck of this design will be the distributing pipe system itself. GT pipes move bursts of items with a delay and **more testing needed** to figure out how many asslines can it feed with constant processing. A possible workaround to this problem to get rid of the main pipe (in the picture the gray PVC ones) and use separate ME interfaces at the input of each assline. *(btw this is how I automated the CALs)*