Industry Advance

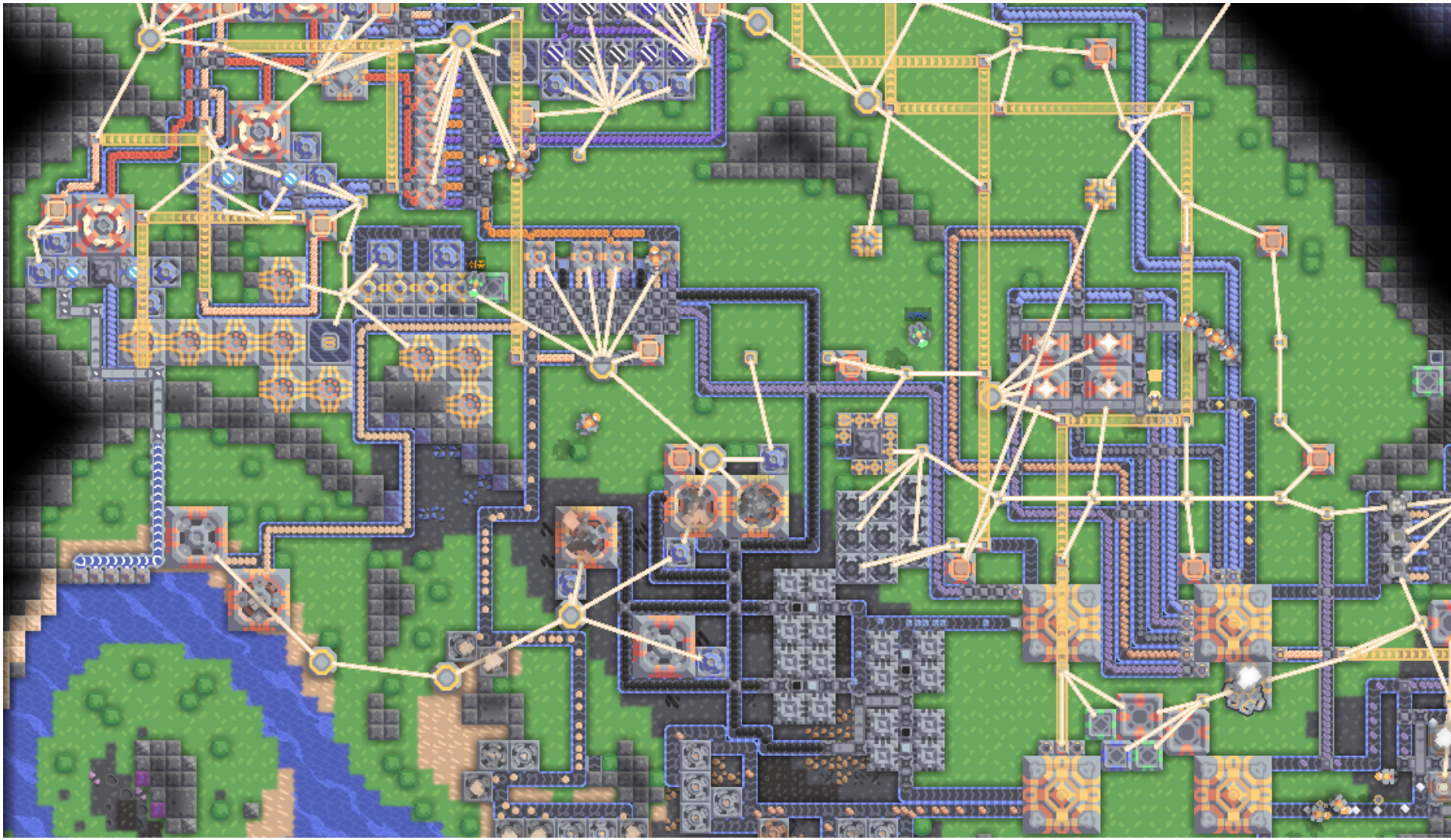
Davids Paskevics, Phillip Wellner, TODO

FU Berlin

2020

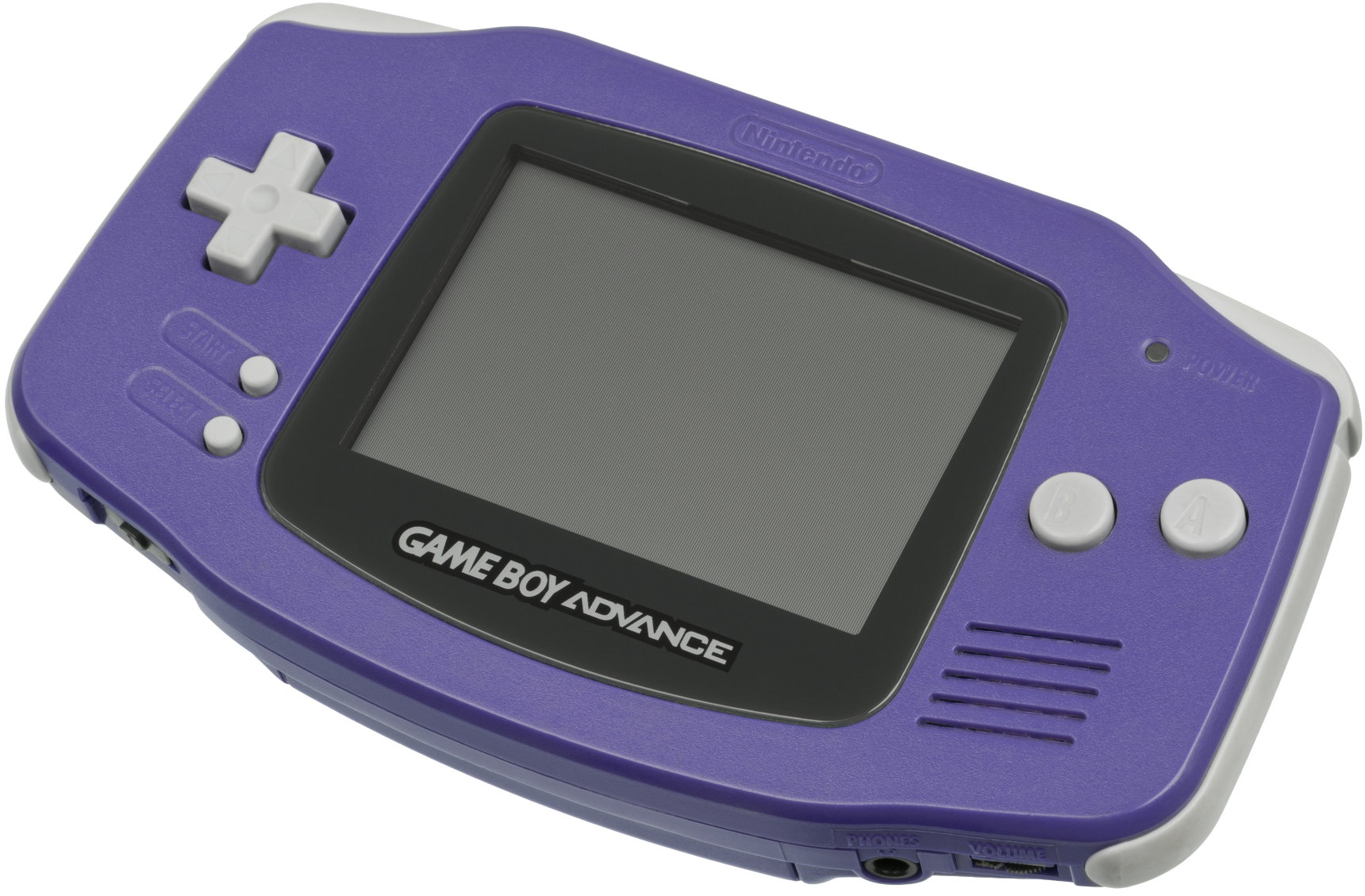
The project

- Game for the GBA console
  - Got idea @spline last year
- Port/clone of the libre Java game *Mindustry*
- Mashup of logistics sim/tower defense
- Wave-based gameplay good fit for portable gaming



Demo

The hardware



The hardware

- Handheld game console by Nintendo
- JP release in 2000
- Engineered for cost and battery life
  - No OS, just BIOS
  - 32 bit ARMv4t ISA processor
  - Clock @16MHz
  - No cache
  - 32KiB fast (on-die) RAM $\rightarrow$ We put stack here
  - 256Kib slow (off-die) RAM = narrow bus + wait cycles $\rightarrow$ We put heap here
  - Up to 32MiB cart ROM
  - Nintendo wanted to make the "ultimate 2D handheld" $\rightarrow$ no HW 3D support or FPU

Prior art

- ISA supported in *LLVM* (and therefore *rustc*)
- Cross-compiling *libcore* easy thanks to *build-std* cargo feature
- rust-console/gba crate provides basic HAL
- Snake game from around 2015, but no "complex" games written in Rust to our knowledge

Starting out

- "Hello world" is tricky
- Easier to draw something
- Shouldn't be hard, right?
- Until you want to draw something useful for a game, that is

Graphics on the GBA

- Modern consoles/PCs are powerful enough to allow uploading pixels to the display freely
- GBA has such "bitmap" modes as well (slow!)
- Also supports HW accelerated "tile modes" (fast!) TODO: Insert graphic on how tiles/sprites work
- Custom GFX format, needs a converter
  - *grit* most commonly-used homebrew tool, docs are meh
- All this makes custom engine a requirement:

We implemented

- Sprite allocator with refcounting
- Simple heap allocator
- Support for maps of arbitrary size
- Text engine
- Filesystem
- Various debug facilities
- Test framework
- Map converter (WIP)
- Not much gameplay (just movement, placing blocks, inventory system)

We didn't finish

- Picked a project too ambitious in scope
- Underestimated engine dev effort
- Underestimated effort needed for asset pipeline
  - Undocumented Mindustry save format
  - Source of many bugs
  - Spent ¿25% of dev time there
- Project management failures (deadlines, working independently)
- Good ol' procrastination
- Nonetheless, Rust saved us some time:

Rust ecosystem

- Many easily integrated *no_std* crates
  - *serde* for map metadata loading from JSON
  - *tiny_ecs* for ECS (good pattern for structuring games in Rust)
  - *hashbrown* for hashmap
  - *fixed* for fixed-point maths (fast!)
  - Many more small ones used
- Crates reduced dev time by a lot
- Downside: Some assume stuff (atomics support)
- Downside: Others are needlessly *std*-dependent (patches)

Rust APIs/tooling

- Tile-based custom text output w/ formatting in ¡50 LOC thanks to *Write* trait
- Custom test framework w/ minimal boilerplate

Rust safety

- Only hit memory bugs in our *unsafe* allocator
- Saves us a lot of time (which we need to squash logic bugs instead)
- Could make more use of borrow checker for managing HW resources

Rust speed

- No CPU bottleneck yet, despite many abstractions
- Code isn't very optimized
- *const_fn* allowed for a low-cost FS implementation
- Easy mixing of ARM/Thumb code would be nice

Conclusion

- I believe Rust is a good fit (but I'm biased)
- Will develop project further
- Interested? Fork us on github: https://github.com/industry-advance/industry-advance (take a look under "Projects" for roadmap!)