Excellent follow-up.
These mistakes are **very real**, very common, and they explain why many freshers stay "junior" for years despite working hard.

I'll structure this exactly like a mentor's **warning list**, with **what they do**, **why it's wrong**, **how it shows up on the job**, and **how I correct it**.

🚨 Common Mistakes Freshers Make in PHASE 0

(Engineering Mindset Failures)

This phase has **no equations, no code, no tools** — yet most failures happen here.

❌ 1. Treating Datasheets Like Textbooks

What they do

- Start from page 1
- Read sequentially
- Highlight everything
- Say: *"I read the manual"*

Why this is wrong

Datasheets are **reference documents**, not teaching material.
They are written assuming you already know *why* you're looking.

How it shows up on the job

- Cannot answer simple "why" questions
- Freezes during design reviews
- Misses critical notes hidden deep in the document
- Takes weeks to understand basic blocks

How I correct it

I stop them immediately and ask:

> "What problem are you trying to solve by reading this page?"

If they can't answer → they close the document.

❌ 2. Memorizing Without Understanding Purpose

What they do

- Memorize core counts, cache sizes, clock speeds
- Recite specs confidently
- Feel productive

Why this is dangerous

Specs **don't design boards** — decisions do.

Real-world failure

> Fresher knows DDR speed = 1600 MT/s
> But doesn't know:

- Why termination exists
- Why length matching matters
- Why timing margins collapse

Board comes back → DDR unstable.

How I correct it

I ask:

> "If this feature didn't exist, what would break?"

If they can't answer, the knowledge is rejected.

❌ 3. Ignoring Power, Clock, Reset Because "It's Boring"

What they do

- Rush to cores and peripherals
- Avoid power chapters
- Skip reset timing diagrams
- Assume reference design will handle it

Why this is catastrophic

> **90% of bring-up failures are power/clock/reset**

Not software. Not drivers.

How it shows up

- Board doesn't boot
- Random hangs
- Intermittent failures
- Blame placed on software team

How I correct it

I tell them bluntly:

> "If power, clock, and reset are wrong — nothing else matters."

Then I make them debug **non-booting boards on paper** before touching real hardware.

❌ 4. Not Connecting Spec → Pin → Schematic → Software

What they do

- Read processor manual separately
- Look at schematic separately
- Think software is "someone else's job"

Why this is a fatal mindset

Engineering is **end-to-end responsibility**.

Real example

- Boot mode decided by pins
- Pins strapped wrong
- Software team stuck for weeks
- Root cause → schematic oversight

How it shows up

- "It should work" statements
- Blame shifting
- No ownership

How I correct it

I force them to trace **one signal end-to-end**:

> Datasheet → Pin → Resistor → Net → Bootloader

They hate it at first.
They thank me later.

❌ 5. Being Afraid to Say "I Don't Know"

What they do

- Nod silently
- Avoid asking questions
- Google privately

- Hope confusion disappears

Why this is dangerous

Confusion compounds.
Silence costs weeks.

How it shows up

- Wrong assumptions
- Late discovery of errors
- Panic during bring-up

How I correct it

I explicitly say:

> "You are expected to NOT know.
> You are NOT allowed to pretend."

Then I reward good questions publicly.

❌ 6. Asking the Wrong Kind of Questions

What they ask

- "What does this register do?"
- "Which value should I set?"
- "Is this correct?"

Why this is weak engineering

These are **execution questions**, not **thinking questions**.

What strong engineers ask

- "Why does this register exist?"
- "What happens if this bit is wrong?"
- "When is this sampled?"

How I correct it

I don't answer shallow questions.
I redirect:

> "Tell me what you think first."

❌ 7. Over-Trusting Reference Designs

What they think

> “Reference design works, so ours will too.”

Reality

Reference designs:

- Assume ideal layout
- Assume exact components
- Ignore real constraints
- Are often incomplete

Real-world failure

- Slight power change
- Different DDR vendor
- Changed clock source
  → Reference design no longer valid

How I correct it

I ask:

> “Why did they choose this component and not another?”

If they can’t explain, they’re copying blindly.

❌ 8. Tool Worship (JTAG, IDE, Scripts)

What they do

- Focus on tools
- Believe tools will “fix” issues
- Panic when tools fail

Why this is wrong

Tools **observe**, they don’t **reason**.

How it shows up

- Endless re-flashing
- Random changes
- No hypothesis

How I correct it

I disable tools initially and ask:

> “Explain what you expect to happen — before running anything.”

❌ 9. Treating Debugging as Guessing

What they do

- Try random fixes
- Change multiple things at once
- “Let’s see if this works”

Why this fails

Debugging without reasoning = chaos.

How it shows up

- No reproducibility
- No learning
- Same mistakes repeated

How I correct it

I enforce:

- One change at a time
- Written hypothesis
- Expected outcome

No exception.

❌ 10. Thinking Phase 0 Is “Too Basic”

What they think

> “When will we do real engineering?”

Reality

Phase 0 **is** real engineering.

Skipping it leads to:

- Slow growth
- Dependency on seniors
- Fear of new processors

How I correct it

I show them senior engineers' notebooks and bug histories.

They realize:

> "The basics are what experts obsess over."

🧠 Summary Table

Mistake	Root Cause
Linear datasheet reading	Student mindset
Memorization	Exam conditioning
Ignoring power/reset	Over-focus on "cool" parts
No end-to-end thinking	Silo mentality
Fear of questions	Insecurity
Blind copying	Lack of ownership
Guess-based debugging	No mental model

🎯 Final Mentor Truth

> **Phase 0 is not about knowledge.**
> **It's about how you think when knowledge is incomplete.**

Engineers who master Phase 0:

- Learn faster
- Make fewer mistakes
- Become independent
- Are trusted with responsibility

If you want next, I can:

- Show **how to detect Phase-0 weakness in interviews**
- Create **Phase-0 evaluation tasks for freshers**
- Map **Phase-0 mastery to promotion-level skills**
- Show **how seniors silently test Phase-0 thinking**

Just say the word 👍