GIGA Embedded Startup Pattern - Best Practice

Key Principle

Always fully control when hardware drivers are constructed and initialized.

Why

Modern MCUs with external SDRAM, Display controllers, DMA, SPI/I2C buses often fail if drivers initialize too early.

Golden Rule

- NEVER use global static hardware objects that do bus activity or memory allocation in the constructor.
- ALWAYS use explicit pointers or call .begin() only after core rails are ready.

Correct Boot Order Example

1) Serial.begin()

2) pinMode() for early heartbeat LED

3) SDRAM.begin() (must be FIRST if LVGL or framebuffer needs SDRAM)

4) Display = new Arduino_H7_Video(...); Display->begin();

5) TouchDetector = new Arduino_GigaDisplayTouch(); TouchDetector->begin();

6) SPI.begin()

7) mfrc522 = new MFRC522(...); mfrc522->PCD_Init();

8) rtc = new RTC_DS1307(); rtc->begin();

Hidden Danger

Many libraries auto-call LVGL or buffer alloc in static constructors.

Safe Pattern

Use new and only construct AFTER SDRAM is online.

Example:

```
Arduino_H7_Video* Display = nullptr;

Display = new Arduino_H7_Video(...);

Display->begin();
```

Same for TouchDetector, RFID, RTC.

Result

- No random crashes
- No panic LED blinking
- Stable DMA, SDRAM, Display

This pattern guarantees deterministic startup on every boot.

Author

Alejandro Vazquez, 2025 - For Zonar Systems