为什么要先做 **PC** 离线仿真器？

1. 快速验证算法
    - 在 PC/GPU 上可以快速迭代网络结构、滤波器、HDR 融合、EIS、VSR 等算法。
    - 不需要受限于移动端 SoC 的内存、带宽和功耗，可以大胆尝试不同结构。
2. 端到端 **Pipeline** 搭建
    - 提前实现 RAW→AI-ISP→HDR→多摄融合→EIS/VSR→RGB 输出的完整流水线。
    - 可以在 Python/C++/CUDA 里拼接所有模块，方便调试和调参。
3. 前端 **+** 后端分工清晰
    - 前端 (Sensor 模拟器)：模拟 RAW 输出（不同曝光、噪声模型、多摄校准偏差）。
    - 后端 (Pipeline)：实现 AI/传统模块混合处理，并输出可视化结果和 IQA 指标。
4. 降低移植风险
    - 提前在 PC 侧定义好 模块接口、数据格式、张量 **shape**、调参 **API**。
    - 后续移植到 SoC/NPU/DPU 时，只需要替换算子实现，而不需要改整个 pipeline。
5. **IQA &** 主观测试
    - 可以批量跑测试集（白天/夜景/逆光/运动/多摄/视频），出 IQA 分数和 A/B 对比图。
    - 这些结果直接驱动后续硬件和模型优化。

仿真器建议架构

1) 前端（**Input Generator**）

- **RAW** 数据输入：从真实相机 dump 的 RAW + 自建噪声模型/曝光模拟器。
- 多摄同步模拟：时间戳、视差、畸变，模拟双/三摄输入。
- 运动合成器：加入虚拟抖动轨迹和运动物体（方便 EIS/VSR 验证）。

2) 后端（**Pipeline**）

- 模块化设计（每个模块可独立开关、替换 AI/传统实现）：
    - RAW Preproc：BPC、BLC、LSC（AI vs 传统）
    - Demosaic + RAW 去噪（AI）
    - HDR Alignment + Fusion + Tone Mapping
    - Multi-Cam Fusion / Seamless Zoom
    - AWB / EE（轻量传统）
    - EIS（运动估计 + 重采样）
    - VSR/RTSR（AI 超分）
- 统一接口：模块 I/O 格式统一（Tensor = H×W×C + metadata）。

3) 调试与可视化

- **GUI** 前端（PyQt/Streamlit/Flask Dashboard）：可以拖 RAW 输入、调整参数、看结果。
- 指标展示：IQA 分数曲线、时延估算、功耗推测。
- **A/B** 测试：并排对比“AI vs 传统”“版本 v1 vs v2”。

4) 底层实现建议

- 框架：Python (快速实验) + PyTorch/TensorFlow (AI 模块) + OpenCV/Numpy (传统 ISP)。
- 性能优化：CUDA kernel / TensorRT / ONNX Runtime，便于后续移植。

- 数据管理：把所有结果存数据库/文件夹，附带元数据（ISO、曝光、场景标签）。

路线图建议

1. **Step 1**：最小可运行 pipeline（RAW dump → AI demosaic → AWB/EE → 输出 RGB）。
2. **Step 2**：加 HDR（多帧对齐 + 融合 + TM），验证照片质量。
3. **Step 3**：加多摄（标定 + 校准 + 无感切换/融合）。
4. **Step 4**：加视频链路（EIS + VSR + 实时预览降采样路径）。
5. **Step 5**：接入 IQA（离线批量打分 + 在线实时预测）。
6. **Step 6**：整理 API/模块接口 → 定义 SoC 实现对接规范。

👉 总结：离线 **PC** 仿真器是必做的基线，相当于“算法 sandbox + 虚拟 ISP”。
它能让算法团队和硬件团队在 SoC tape-out 之前对齐接口、测试策略、模块边界，避免到硬件阶段才发现模块无法拼接。