## **嵌入式性能剖析工具**

eProf/

├── core/ # 核心引擎

│ ├── profiler\_engine.c # 剖析主循环与调度

│ └── event\_scheduler.c # 多维度数据采集调度

├── data\_collectors/ # 数据采集层（核心）

│ ├── cpu\_perf\_collector.c # CPU硬件性能计数器

│ ├── memory\_access\_collector.c # 内存访问模式

│ ├── io\_latency\_collector.c # 块设备I/O延迟

│ └── irq\_collector.c # 中断与上下文切换

├── analyzers/ # 数据分析层

│ ├── hotspot\_analyzer.c # 热点函数分析

│ ├── bottleneck\_analyzer.c # 瓶颈识别

│ └── correlation\_analyzer.c # 多指标关联分析

├── output/ # 结果输出层

│ ├── flame\_graph\_generator.c # 火焰图生成

│ ├── text\_report\_generator.c # 文本分析报告

│ └── realtime\_dashboard.c # 实时仪表盘（可选）

└── utils/

├── ring\_buffer.c # 高性能环形缓冲区

└── symbol\_resolver.c # 地址到函数名的解析

### **1. 低开销CPU性能数据采集（核心亮点）**

使用Linux perf\_event\_open 系统调用，这是专业性能工具使用的接口。



### **2. 调用栈采样与火焰图生成**

这是项目的"杀手级"特性，能生成直观的火焰图。



### **内存访问模式分析**



### **4. I/O延迟监控**

// io\_latency\_collector.c

// 使用blktrace数据或/proc/diskstats

void analyze\_io\_latency() {

// 计算：await = (读写时间总和) / (读写完成次数)

// 高await值意味着I/O瓶颈

}

