???????

适用于 `design/` 下所有微架构 `.md` 文档。参考实现: [[LLC]] + `LLC.pipeline.html`。

0. ??

- 微架构文档是 人面向 **RTL / [[logix]]** 模型实现的合同 ：写完 `.md` 就能据此写 RTL，不再回头补
- 每个单元一份主 `.md`（如 `L1.md`、`DMA.md`、`LLC.md`），描述功能与流水线
- 拍级流水线 强制 作为独立章节，不和功能描述混写
- 可选的交互式可视化 HTML 文件挂在同目录（`<unit>.pipeline.html`），不写入 git LFS、不依赖外部资源

1. ?????

主 `.md` 按以下顺序组织章节（编号用 `#### N. 标题`）：

编号	章节	内容
§1	顶层参数	容量 / 宽度 / 深度等常量的表
§2	顶层框图	ASCII 框图或 mermaid，标出对外端口与内部主要模块
§3	顶层端口	控制 / 数据 / 出对方向各一节，沿用本仓库已有端口约定
§4	地址映射	若是存储类单元；否则按需
§5	控制总线协议子集	opcode 表、异步操作约定、不做合法性检查的边界
§6 - §M	各功能模块详述	每个功能模块独立小节（阵列、命中 / 缺失、出口、维护等）
§M+1	流水线	拍级时序契约，见 **§2**
§M+2 -	Pin / Lock / Bypass 等可选语义、与其他单元关系、复位、调试、待规格化参数、DGMP 关联	

数字章节用 H4 `#### N.`、子节用 H5 `##### N.M`、子子节用 H6 `###### N.M.K`。三级到顶。

2. ?????

每个微架构文档必须有 "流水线 "章节。结构如下，编号 N 取主文档下一个未用数字（LLC.md 取 §11）：

2.1 ??

- 一段话说明：本章把哪些功能章节（§a-§b）在全局时钟上展开到拍级
- 工艺前提（如 SRAM 端口形态 1RW vs 1R1W）必须显式
- 配套交互式视图：`<unit>.pipeline.html` 的位置和用途
- 渲染说明：Mermaid / WaveDrom / YAML 在哪些环境能渲染

2.2 §N.1 ????

按拍管理的资源表。一个三列表足够：

资源	端口形态	来源
`<name>`	`<arb / sram_port / ff_rw / cam / table / comb>`	§x.y

资源是硬件实体——一个仲裁器、一组 SRAM 端口、一个 FF 寄存器组、一个表。不要把功能动作写进资源清单。

2.3 §N.2 ????

mermaid `flowchart LR` 显示所有路径的分叉、合流、异步链。每条路径用一个 stage 块（不展开内部）。

2.4 §N.3 ~ §N.X ????

每条路径一个子节，内部固定四件套（用 H6 ###### 编号 §N.X.Y）：

1. 拓扑：mermaid `flowchart LR`，stage 节点链 + 分支条件
2. 拍级展开：WaveDrom JSON 代码块，横轴 cycle，纵轴 stage / latch / 资源占用，跨信号依赖用 `edge` 字段
3. **latch** 字段表：每个 stage→stage latch 携带的字段（含字段名 + 简短解释）
4. 资源占用表：拍 × 资源的矩阵

跨路径关系（与主路径的冲突、回填唤醒等）单独成一个子子节。

2.5 §N.{?? 2} ???????????

- 资源 × 占用方表：每个资源列出主要占用方 / 次要占用方 / 仲裁规则
- **stall** 触发表：什么条件下哪条路径会 stall、解除条件
- 全 **bank** 吞吐分析（如果适用）

2.6 §N.{?? 1} ??? schema

YAML 代码块作为机读真相源，给工具消费（未来生成 logix / RTL 骨架、跑静态检查）。schema 由四部分构成：

```
unit: <name>
clock: <main_clk>


resources:
```

```
  <name>: { kind: ..., ports/slots/width/banks: ... }

paths:
  <path_name>:
    stages:
      - { id: <S0/S1/...>, in: [...], uses: [...], out: [...] }

arbitration:
  - { resource: <res>, contenders: [...], rule: "<text>" }

stall_conditions:
  - { src: <res.cond>, effect: "<text>" }
```

写出哪几条路径作为样本即可，完整版可旁挂 `<unit>.pipeline.yaml` 文件。

3. ??????

模式	拍号	例
主路径	S0 / S1 / S2 / S3 / S4	读命中
主路径分支	S3' / S4'（带撇号）	读缺失从 S2 分叉到 mshr
异步流水线	F0 / F1 / F2	回填（fill）
独立 FSM	W0 / W1 / W2 / W3	维护扫描 walker
特殊短路径	B0 / B1	旁路 bypass
重发	S2'（带撇号，复用主路径名 + 撇号）	回填后 master 重发到 S2

不同路径的拍号空间不冲突，方便资源占用全景表交叉引用。

4. ????????

- 同文档内：`§N`、`§N.M`、`§N.M.K`
- 跨文档：`[[文件名]] §N`（Obsidian wiki link 风格），文件名不带 `.md` 后缀
- 大段引用别人的协议时，直接说"完全沿用 [[L1]] §3.2 的 NoC slave 信号集"，不要重复贴信号表

5. ???????

载体	表达	谁读
Mermaid `flowchart`	拓扑结构（节点 + 箭头）	人（GitHub / Obsidian / VS Code 原生渲染）
WaveDrom JSON	拍级时序（波形 + 跨拍依赖 + 资源冲突）	人（wavedrom.com/editor.html、Obsidian + WaveDrom 插件）

载体	表达	谁读
嵌入式 YAML	机读 schema（resources / paths / stages / arbitration）	工具

三者信息互补，不互相替代。同一份事实在多处出现时，YAML 是真相源——Mermaid / WaveDrom 是渲染产物（手写阶段允许双源，工具成熟后单源生成）。

不要使用：drawio（不好 diff、信息无法被工具消费）、PNG/SVG 截图（同前 + 改一处要重导）、Word/PPT（同前）。

6. ??? HTML???????

文件名约定：`<unit>.pipeline.html`，与主 `.md` 平级。

单文件、零外部依赖（CSS / JS 内嵌），打开即用。数据来源理想是同目录的 `<unit>.pipeline.yaml`；起步阶段允许把数据写在 `<script>` 的 JS 对象里，等 schema 稳定再切换。

四块信息（沿用 `LLC.pipeline.html` 模板）：

1. 流水线甘特图：横轴 cycle，纵轴路径，stage 色块按 cycle 落格
2. 资源占用网格：横轴 cycle，纵轴资源，单占用 / 多路径冲突分色
3. 按硬件级展开：横轴 cycle，每拍列出所有承载的并行子功能
4. 按路径展开：每条路径横排 stage 卡片，卡片内显示功能 / 资源 / 输入 / 输出 latch

第 3 / 第 4 互为转置视图。点击任一处的 stage / 子功能高亮另外两处对应项。

7. ????

1. 文档先行：先写 `.md`，再写 RTL。`.md` 锁定的拍数是 RTL 必须满足的契约。
2. 单源真相：拍级时序信息只在 "流水线" 章节一处描述。功能章节描述 "做什么"，不描述 "在哪一拍做"。
3. 工艺前提显式：1RW vs 1R1W、SRAM 端口数等工艺选择必须在引言或资源清单中写明。
4. 不重复：流水线章节不重述功能章节的逻辑；功能章节不写拍级时序。
5. 资源 **vs** 功能：资源清单只列硬件实体，不列功能动作。功能动作是 stage 里的 `comb` 字段。
6. 沿用 **LLC** 用过的词：cacheline / set / way / pinned way / bypass / fill / writeback / mshr / walker。术语在不同文档间保持一致。

8. ?????????

放在主文档末尾。两类内容：

1. 硬件参数：容量、深度、端口数、是否例化 ECC / prefetcher 等产品级决策
2. 流水线相关：仲裁规则细节、retire 拍号、FSM 状态转移等取决于 floorplan / 实现的部分

不要把 "功能未定义" 的内容塞进待规格化——功能必须在功能章节里明确。

9. ?????

角色	文件
主文档样板	`design/LLC.md`
流水线章节样板	`design/LLC.md` §11
交互式 HTML 样板	`design/LLC.pipeline.html`
上层抽象（不写流水线）	`design/Pipe.md`、`design/架构_微架构.md`

新增模块文档时：拷贝 `LLC.md` 结构、改章节内容、删掉用不到的小节。流水线章节直接套用 §11 的子节结构。

10. ????

提交前过一遍：

- □ §1 - §M 功能章节完整，每个硬件模块都有独立小节
- □ 流水线章节存在，包含资源清单 / 路径一览 / 每条路径四件套 / 资源占用全景 / 嵌入式 YAML
- □ 工艺前提显式写出
- □ 拍号命名遵循 §3
- □ 跨文档引用用 `[[...]] §N` 形式
- □ 资源清单只列硬件实体
- □ 待规格化清单只放参数和实现细节，不放未定义功能
- □ 配套交互式 HTML（如有）数据与文档一致
- □ 沿用本仓库已有的术语（参考 LLC.md / Pipe.md / GMP.md）

Revision #1
Created 2026-05-29 11:07:22 UTC by Colin
Updated 2026-05-29 11:07:22 UTC by Colin