

# 计算机组成原理

## 实验3 直接相联Cache设计



HITSZ 实验与创新实践教育中心  
Education Center of Experiments and Innovations, HITSZ

- 掌握直接相联Cache的基本结构及其设计方法
- 实现直接相连地址映射的机制
- 实现**读命中**和**读缺失**两种情况的处理



本实验为容量是8KB的主存，设计直接相联Cache，参数如下：

- Cache总容量512B、块大小4个字节 （128个Cache块）

实现在CPU发出读操作请求后的取数据操作，包括**读命中**和**读缺失**两种情况。

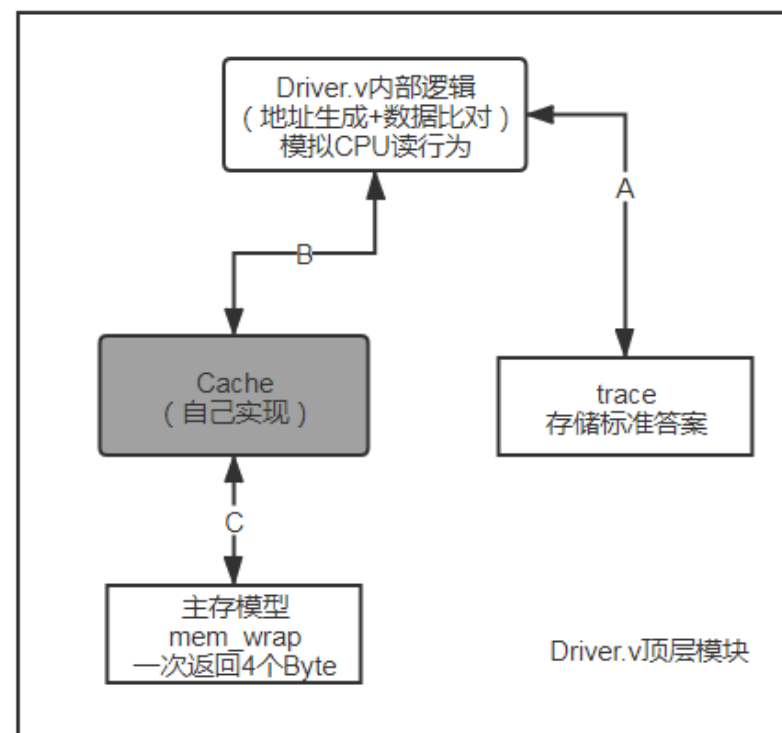
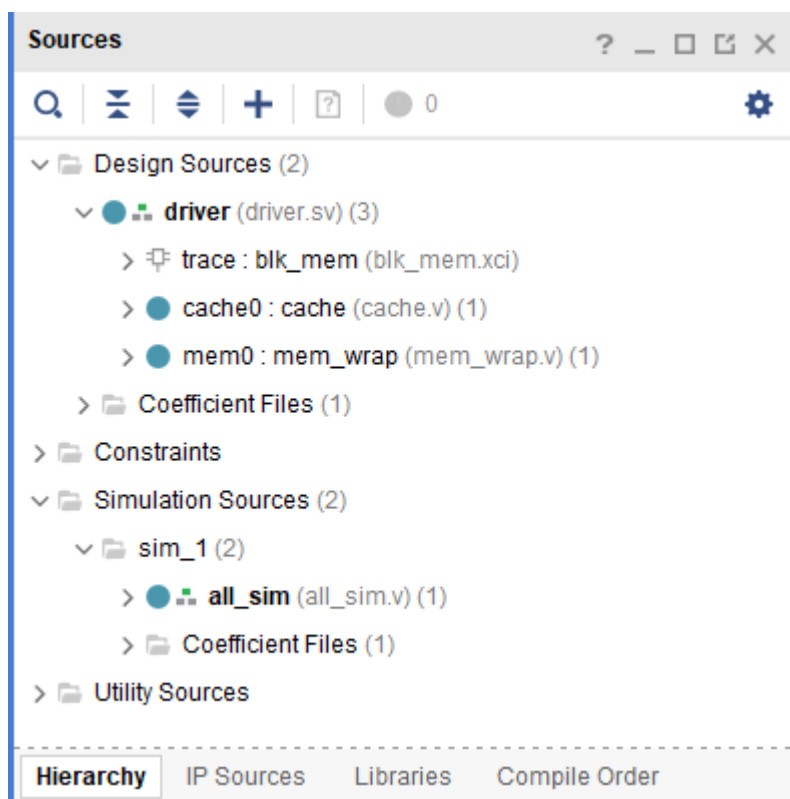
作业提交：

实验作业（包含状态转移图） （3分）

Cache.v文件和创建的IP核文件 (\*.xci) （4分）



- driver (driver.sv) ----- 顶层模块，模拟CPU的访存行为，并进行数据正确性检查，可以看作是CPU
- trace (IP核) ----- 存储标准答案
- cache (**cache.v**) ----- Cache模块 (**需要完成**)
- mem\_wrap (mem\_wrap.v) ----- 主存存储器模型



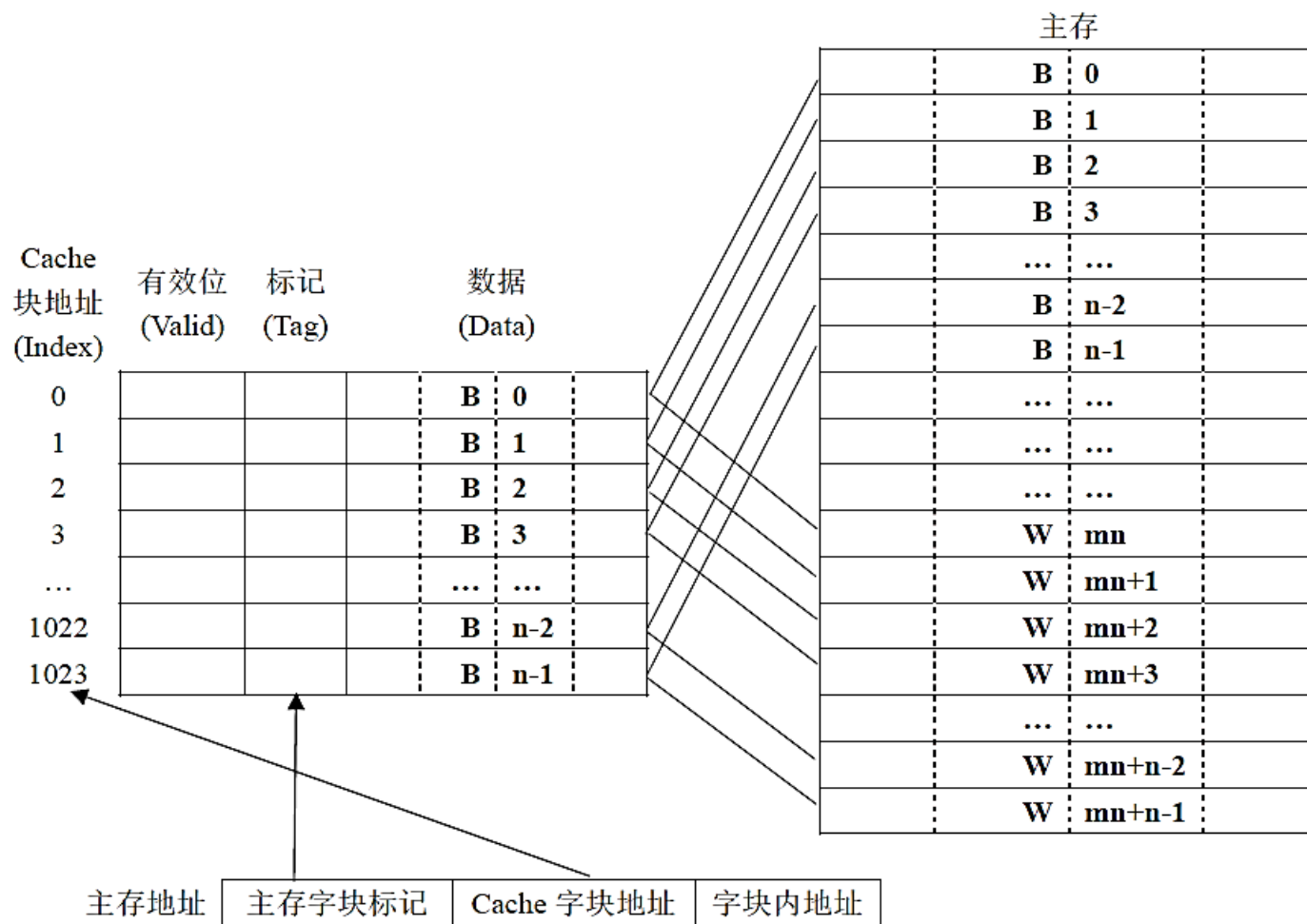
## • 接口规范

属性	名称	含义	位宽
全局信号			
输入	clk	时钟	1
输入	reset	复位（高电平有效）	1
与上层Driver模块的接口			
输入	raddr_from_cpu	CPU的读地址	13
输入	rreq_from_cpu	CPU的读请求	1
输出	rdata_to_cpu	Cache读出的数据	8
输出	hit_to_cpu	命中标记	1
与主存(mem_wrap)模块的接口			
输入	rdata_from_mem	主存模块读取的连续4字节	32
输入	rvalid_from_mem	主存读取完毕标记	1
输出	rreq_to_mem	读主存请求	1
输出	raddr_to_mem	读主存首地址	13

- 进行地址各个字段的分解
- 计算Cache各个参数，选择大小合理的Block RAM, 创建IP核并实例化
- 编写命中判断的组合逻辑
- 设计控制状态机
- 自己编写Testbench，验证相关功能正确性。
- 使用提供的all\_sim.v测试模块进行测试、调试，根据输出的调试信息，定位错误点



## • 地址分解



Cache的总容量为512B

每块4个字节

128个Cache块

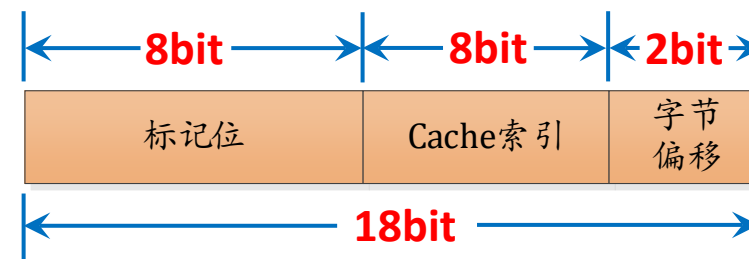
? 位

? 位

input [12:0] raddr\_from\_cpu, // CPU来的读地址

## • 举例

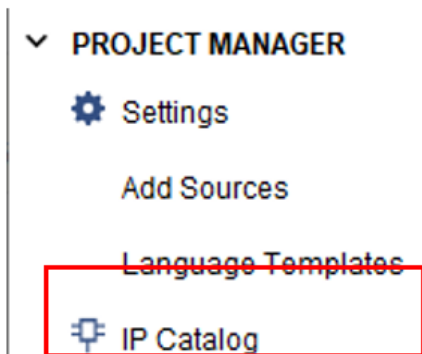
- 系统的主存容量为256KB (主存地址宽度为18位)
- 设计直接相联Cache
  - 大小为1024B, 块大小为4B
- Cache块大小是4B (块内偏移为2位)
- Cache块有256 ( $1024\text{B} / 4\text{B} = 256$ )
  - Cache块地址为8位
  - 剩余的 $18 - 8 - 2 = 8$ 位, 作为Tag使用



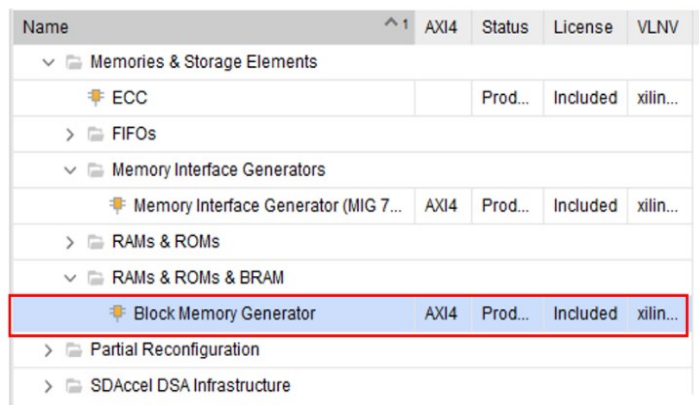


## • 创建IP核

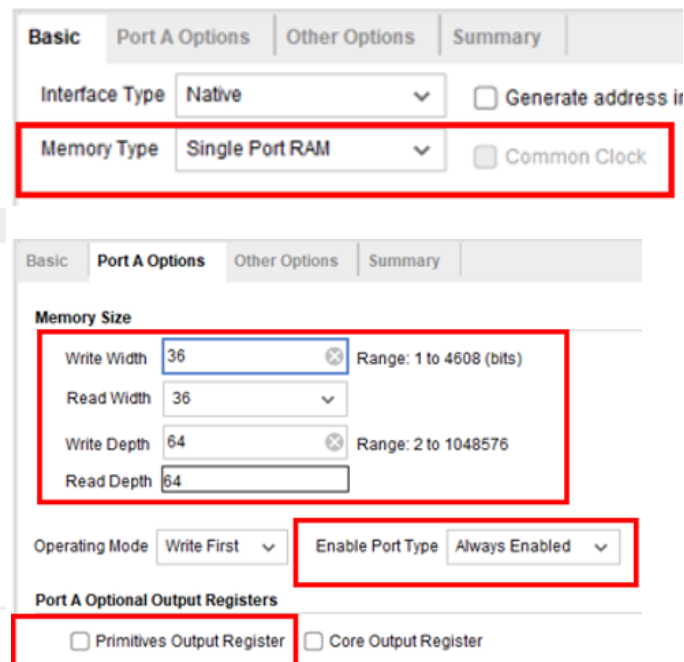
1) 点击Flow Navigator中的IP Catalog, 打开窗口添加IP核;



2) 双击Block Memory Generator;



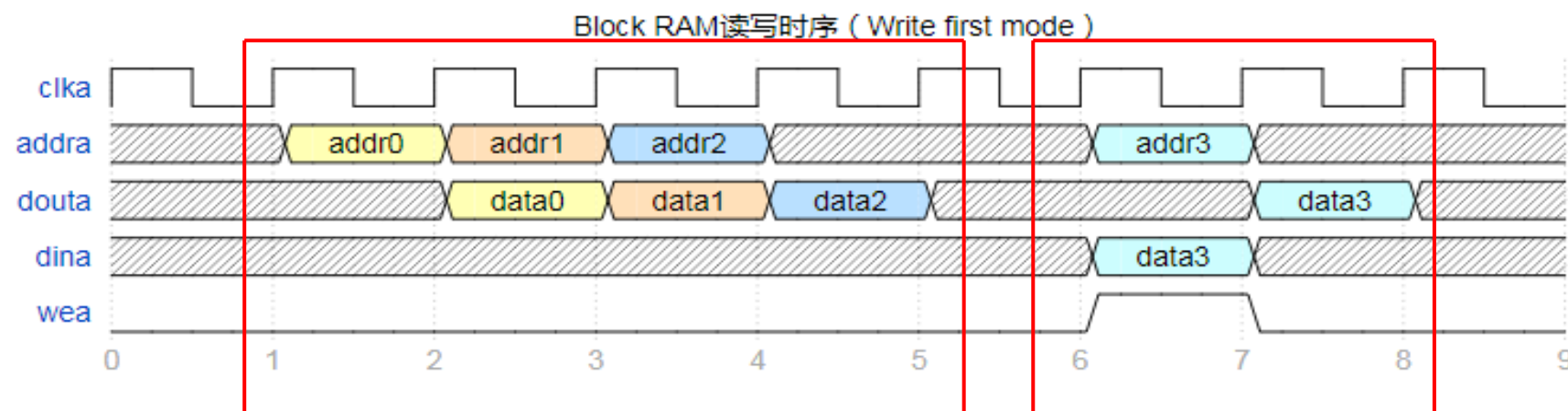
3) 配置RAM;



4) 例化IP。

```
56 //----- Begin Cut here for INSTANTIATION Template
57 blk_mem_gen_0 your_instance_name (
58     .clka(clka),      // input wire clka
59     .wea(wea),        // input wire [0 : 0] wea
60     .addra(addra),    // input wire [5 : 0] addra
61     .dina(dina),      // input wire [35 : 0] dina
62     .douta(douta)     // output wire [35 : 0] douta
63 );
64 // INST_TAG_END ----- End INSTANTIATION Template
```

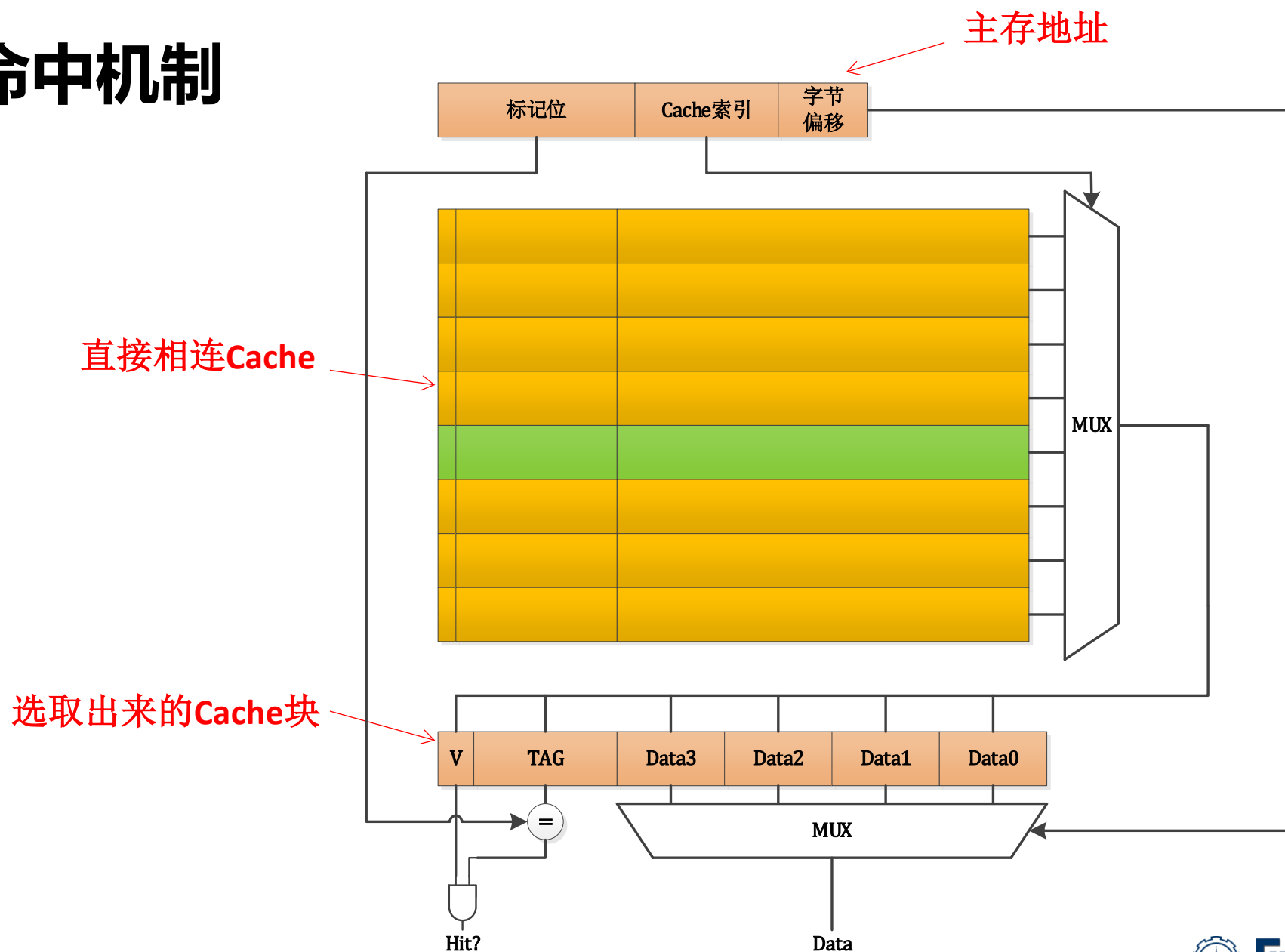
## • Block RAM的读写时序



clka 时钟线  
addra 地址线  
douta 输出端  
dina 输入端  
wea 写使能, 拉高写入

- 读时序：上一周期给出地址，下一周期输出数据，可连续读取
- 写时序：上一周期给数据、写地址，拉高wea信号，下一周期成功写入数据，刚刚写入的数据出现在douta口上

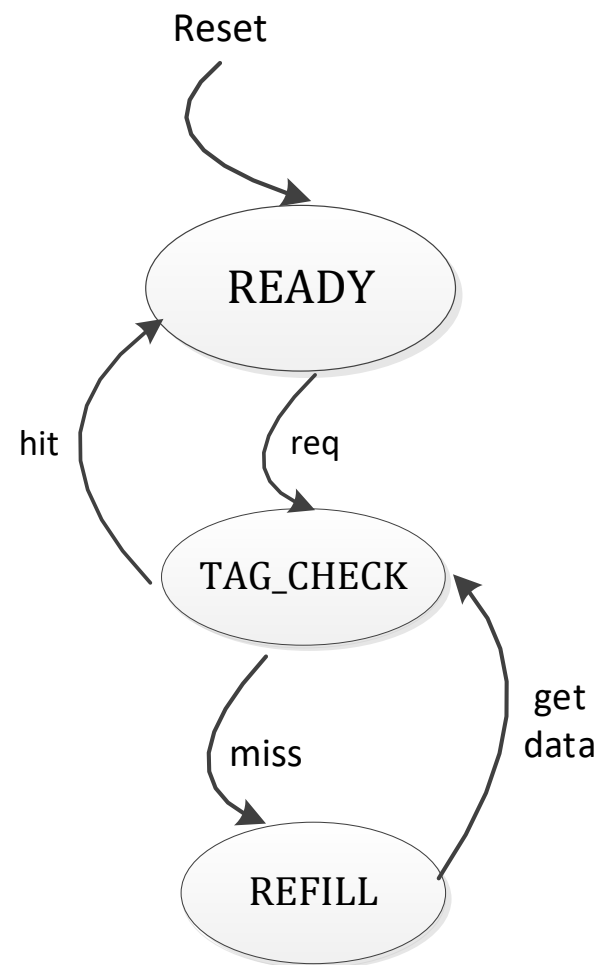
## • 判断命中机制



## • 实现状态机

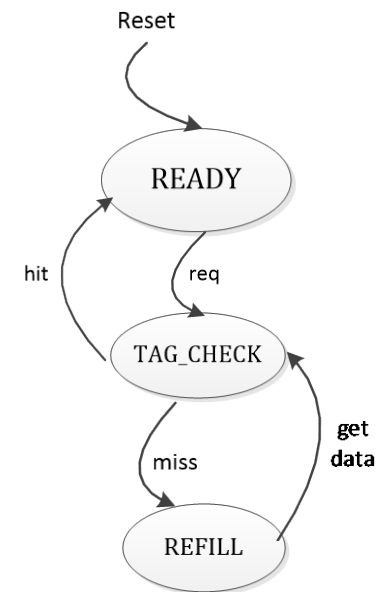
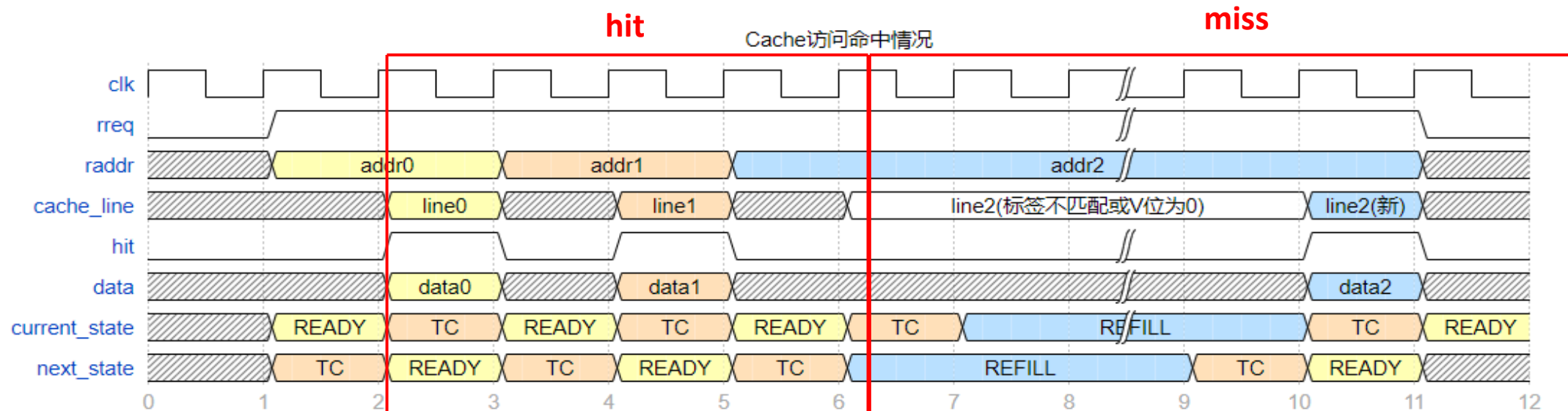
### Mealy状态机实现

- **READY**: Cache处在就绪状态
- **TAG\_CHECK**: 检查Cache是否命中
- **REFILL**: 将取回的字块, 加上相应的标签, 存进自己的存储体内。



## • Cache的读时序

- 读命中(hit)
- 读缺失(miss)



## • 测试

完成设计后，可以运行all\_sim.v文件中的仿真，运行仿真时，下方的Tcl Console也会打印相应的调试信息，帮助你定位出错点。



The screenshot shows a 'Tcl Console' window with a toolbar containing icons for search, zoom, run, pause, and other functions. The console displays the following text:

```
Cache已取回数据162
该地址测试正确，将测试下一个地址
-----
访问地址为 507
等待Cache响应，应得到数据 0
Cache访问命中!
Cache已取回数据 0
该地址测试正确，将测试下一个地址
-----
访问地址为 508
等待Cache响应，应得到数据 8
等待Cache响应，应得到数据 8
等待Cache响应，应得到数据 8
等待Cache响应，应得到数据 8
等待Cache响应，应得到数据 8
等待Cache响应，应得到数据 8
等待Cache响应，应得到数据 8
```

- 课堂上完成：作业第2部分的状态转移图
- 提交内容：
  - 实验作业（包含状态转移图）（3分）
  - Cache.v文件和创建的IP核文件 (\*.xci)（4分）
- 提交方式：  
[http://10.249.12.98:6620/auth/users/sign\\_in](http://10.249.12.98:6620/auth/users/sign_in)
- Deadline: 6月14日（周一） 24:00



# 开始实验



HITSZ 实验与创新实践教育中心  
Education Center of Experiments and Innovations, HITSZ