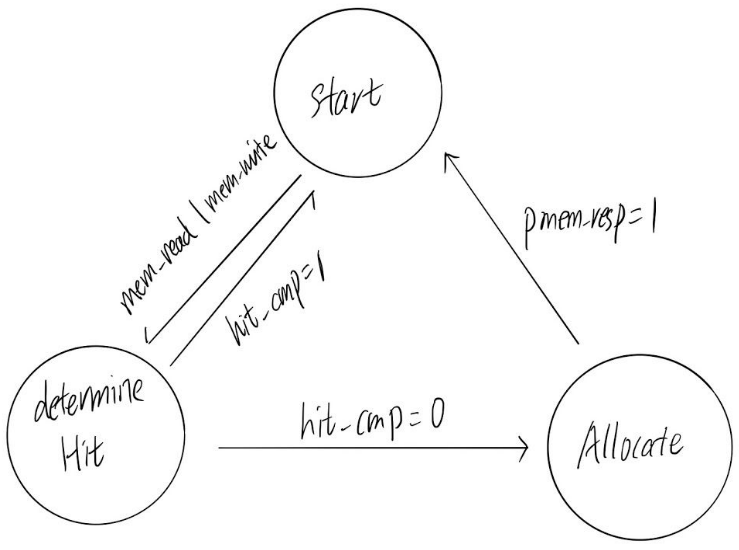
General strategy

1. make L1 cache almost unchanged from mp2
2. arbiter always prioritize instruction miss than data miss
3. make L2 cache 4-way, 256 bit/line, with a pseudo-LRU replacement policy

Interfaces

```
module arbiter(
        input i_read, i_address, d_read, d_address ,L2_resp, L2_rdata
        output i_resp, L2_read, L2_address, d_resp
);
module L1_icache(
        input i_address, arbiter_resp,
        output i_resp, i_read, i_rdata
);
module L1_dcache
        input d_address, arbiter_resp,
        output d_resp, d_read, d_rdata
);
module L2_cache(
        input L2_read, L2_address, pmem_resp, pmem_rdata,
        output L2_resp, L2_rdata. pmem_read, pmem_write
);
```

Cache state diagram



Start:
- read all data in parallel
- idle state if !mem_read/!mem_write

Determine_hit:
- Tag comparison
- mem_resp in always_comb

Allocate:
- read miss / write miss
- read data from pmem
- write dirty data to pmem