果壳Cache验证报告

Leo Cheng · chengkelfan@qq.com · Lfan-ke@Github · heke1228@Gitee · heke1228@WeChat

功能梳理

参考资料：果壳文档，果壳官仓，2024验证报告
果壳缓存规格：三级流水，32k-L1，4路组相联，待测DCache，替换策略为随机替换，非一致性
果壳 Cache 的顶层接口及说明：

```
    """ 基本请求接口以及握手信号 """
    class BaseBundle(Bundle):
        clock, reset = Signals(2)
        class IO(Bundle):
            flush, empty = Signals(2)
        io = IO.from_prefix("io_")

    class HandShakeBundle(Bundle):
        ready, valid = Signals(2)

    """ MMIO/MEM/一致性的请求响应接口抽象 """
    class SRAMReqBundle(HandShakeBundle):
        class BitsSRAMReq(Bundle):
            addr, size, cmd, wmask, wdata = Signals(5)
        bits = BitsSRAMReq.from_prefix("bits_")

    class SRAMRespBundle(HandShakeBundle):
        class BitsSRAMResp(Bundle):
            cmd, rdata = Signals(2)
        bits = BitsSRAMResp.from_prefix("bits_")

    """ Cache的请求响应接口 """
    class RequestBundle(HandShakeBundle):
        class BitsRequest(Bundle):
            addr, size, cmd, wmask, wdata, user = Signals(6)
        bits = BitsRequest.from_prefix("bits_")

    class ResponseBundle(HandShakeBundle):
        class BitsResponse(Bundle):
            cmd, rdata, user = Signals(3)
        bits = BitsResponse.from_prefix("bits_")
```

行为：
CPU ↔ Cache：若命中则直接读写，否则重载后进行读写
Cache ↔ Mem：Cache未命中会触发突发读，若需要脏块写回则触发突发写，读写满一个缓存行
Cache ↔ MMIO：直接透传单个读写命令，地址空间：32'h3000_0000-32'h7FFF_FFFF
补充：突发读发起一次请求，期待8次响应，突发写的addr是首地址需要内置计数器计算地址偏移

测试点分解

基本读写

1. 测试对Cache进行读写，是否可以正确的读取到写入的字
2. 测试对Cache进行读写，检测是否会根据地址将命令转发到对应的MMIO/MEM
3. 测试对Cache进行读写，由于虚拟MEM为33k，略大于缓存大小，可覆盖缓存
4. 读写的范围需要将所提供的虚拟MMIO/MEM区域全覆盖，作为一个单独测试点

特殊测试

1. 测试读写MMIO对应的地址会阻塞流水线
2. 测试连续读写地址连续的三次及以上的MMIO不会触发突发读写
3. 测试读写Cache/MEM会在随机条件下触发脏块写回
4. 测试缓存命中时读写缓存需要的时钟周期大于缓存未命中时读写缓存需要的时钟周期
5. 测试缓存请求策略是关键字优先
6. 测试只有脏块会被写回
7. 测试干净块不会被写回

测试用例编写

基本读写

测试用例1：对整个MMIO进行读写，由于虚拟MMIO采用bytearray，所以读出的字应该等于写入的

> test_rw_all.py/test_basic_cache_seq_rw_mmio

测试用例2：对整个MEM进行读写，先写后读，检测所读取的被写入的字正确

> test_rw_all.py/test_basic_cache_seq_rw_mem

测试用例3：随机选择MEM区域进行读写，检测所读即所写

> test_rw_all.py/test_basic_cache_rw_mem

特殊测试

测试用例1：对MMIO区域进行连续读和连续写，检测不会触发突发读写的请求，且读写结果正确

> test_cache.py/test_mmio_no_burst

测试用例2：测试MMIO的读写会阻塞流水线进行请求直通，请求需要完成的时钟周期大于MEMHIT

> test_cache.py/test_mmio_block_pipeline

测试用例3：测试直接写入Cache-MEM区域不会触发MEM写，脏块被替换时才会触发MEM写

> test_cache.py/test_cache_write_back

测试用例4：测试Cache未命中时需要花费的响应周期大于缓存命中时

> test_cache.py/test_cache_miss_and_hit

测试用例5：测试缓存请求是关键字优先，请求一个CL偏移地址，会触发缓存的突发读首地址基地址

> test_cache.py/test_cache_keyword_priority

测试用例6：测试缓存脏块会被写回，且检测写回的首地址匹配

> test_cache.py/test_cache_dirty_wb

测试用例7：测试干净块不会触发写回

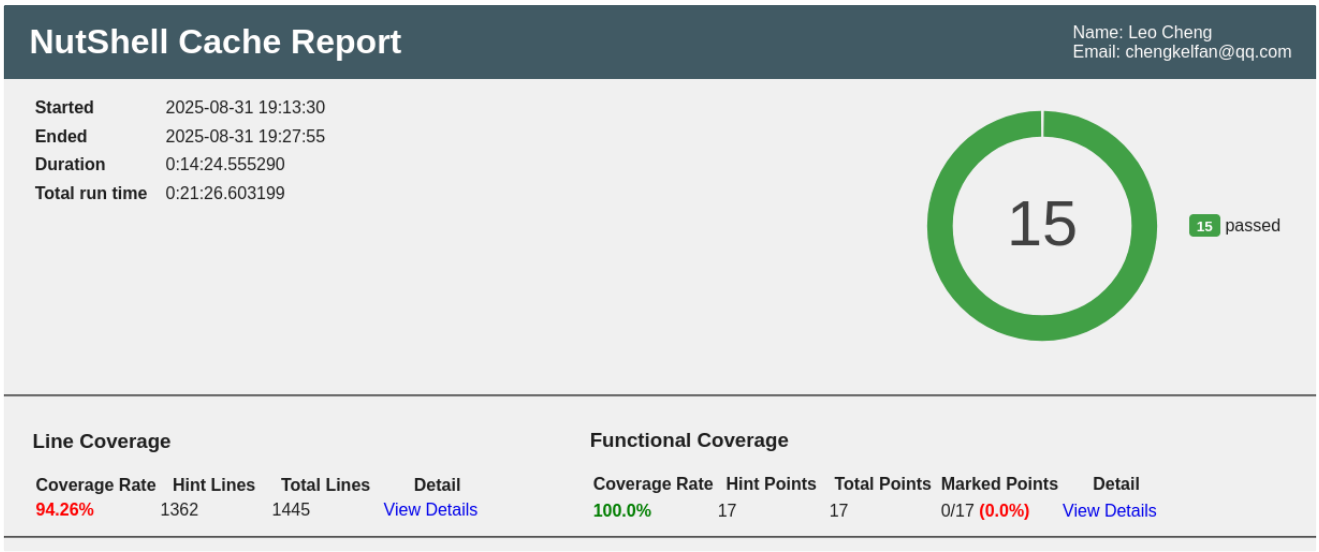
> test_cache.py/test_cache_clean_non_wb

验证结果分析

验证结果显示果壳 Cache 未存在功能性缺陷。但是缺少说明文档，只能通过STFSC以及盲测来收集端口状态机行为，对验证初期的进度影响较多。

验证结论

上述所有的测试点均覆盖，且验证成功。



toffee report
