HW#5 Cache Optimization

Chun-Jen Tsai
National Chiao Tung University
12/17/2020

Homework Goal

- Cache is crucial for the memory subsystem performance of a microprocessor:
  - Aquila has a pair of 4-way set associative caches
  - Dhrystone running on DRAM is only 0.64 DMIPS/MHz
- Your tasks:
  - Analyze the cache behavior and find out why the DMIPS is much lower than running Dhrystone on TCM
  - Improve the performance of the cache subsystem
- You should upload your report to E3 by 1/8, 17:00.

Some Simple Statistics

- Default size on Arty is 4KB each for I- and D-caches
  - 76% usage of LUT
  - 38% usage of FF
  - 68% usage of BRAM
- Using the linker script to put code/data and heap/stack in either TCM or DRAM, the DMIPS/MHz are as follows:
  - Everything on TCM: 0.90
  - Code/data on TCM, heap/stack on DDR: 0.84
  - Code/data on DDR, heap/stack on TCM: 0.67
  - Everything on DRAM: 0.64
  - With an 8KB I-cache, DMIPS increases by 0.04 ~ 0.05

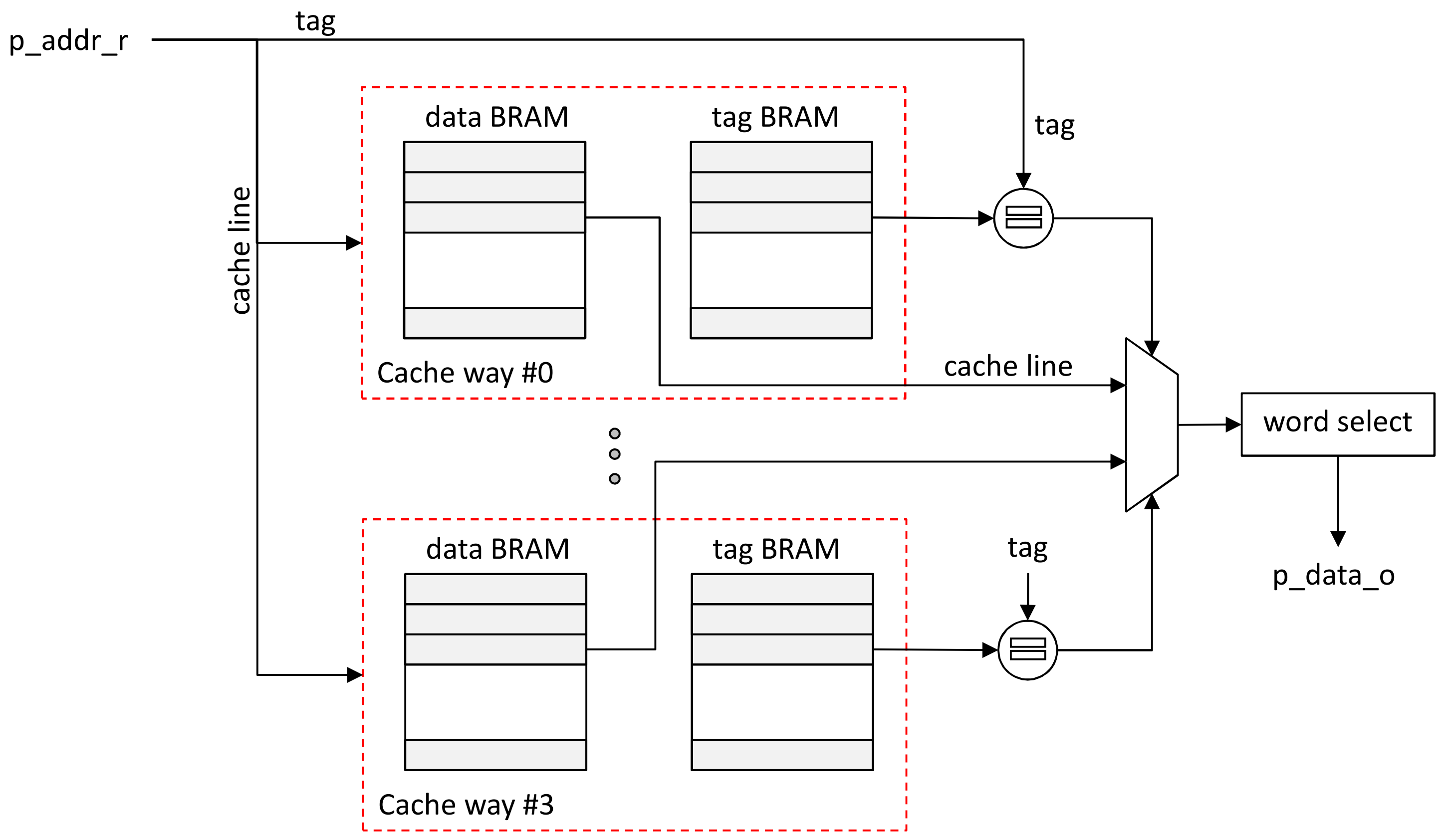
Handling Aquila Memory Requests

- Unlike TCM, a memory request to cache can take multiple cycles
  - Aquila uses strobing signals for memory requests (for both TCM and cache)
  - Therefore, `p_addr_i` must be registered as `p_addr_r` at the strobe cycle for cache memory

- Notes on I-cache:
  - An instruction can be returned in the same clock cycle of strobe upon cache hit
  - The returned instruction is intentionally delayed till next clock edge to match the behavior of TCM

Cache Organization on Aquila

- Data flow on cache hit:



Cache Memory Coding Issue

- Each cache line should have a pair of “valid” and “dirty” flags that stores the state of the cache line
  - Valid – the cache line contains valid data
  - Dirty – the data in the cache line have been modified
- These flags are synthesized using LUTs or BRAMs
  - For a small cache, using BRAM may be wasteful since BRMAs are allocated in 18-kbit unit
  - Aquila uses LUTs for tags and flags
- On Arty, 8KB I-cache + 4KB D-cache is possible, to enlarge them, you have to change the coding style

Register Array Implementation

- A register array can be implemented using LUTs, Flip-Flops, or BRAMs:

```
reg VALID_ [0 : N_LINES-1][0 : N_WAYS-1];
reg DIRTY_ [0 : N_LINES-1][0 : N_WAYS-1];
```

- How do you control the implementation methods?
  - Using pragma in your code (may not be honored)

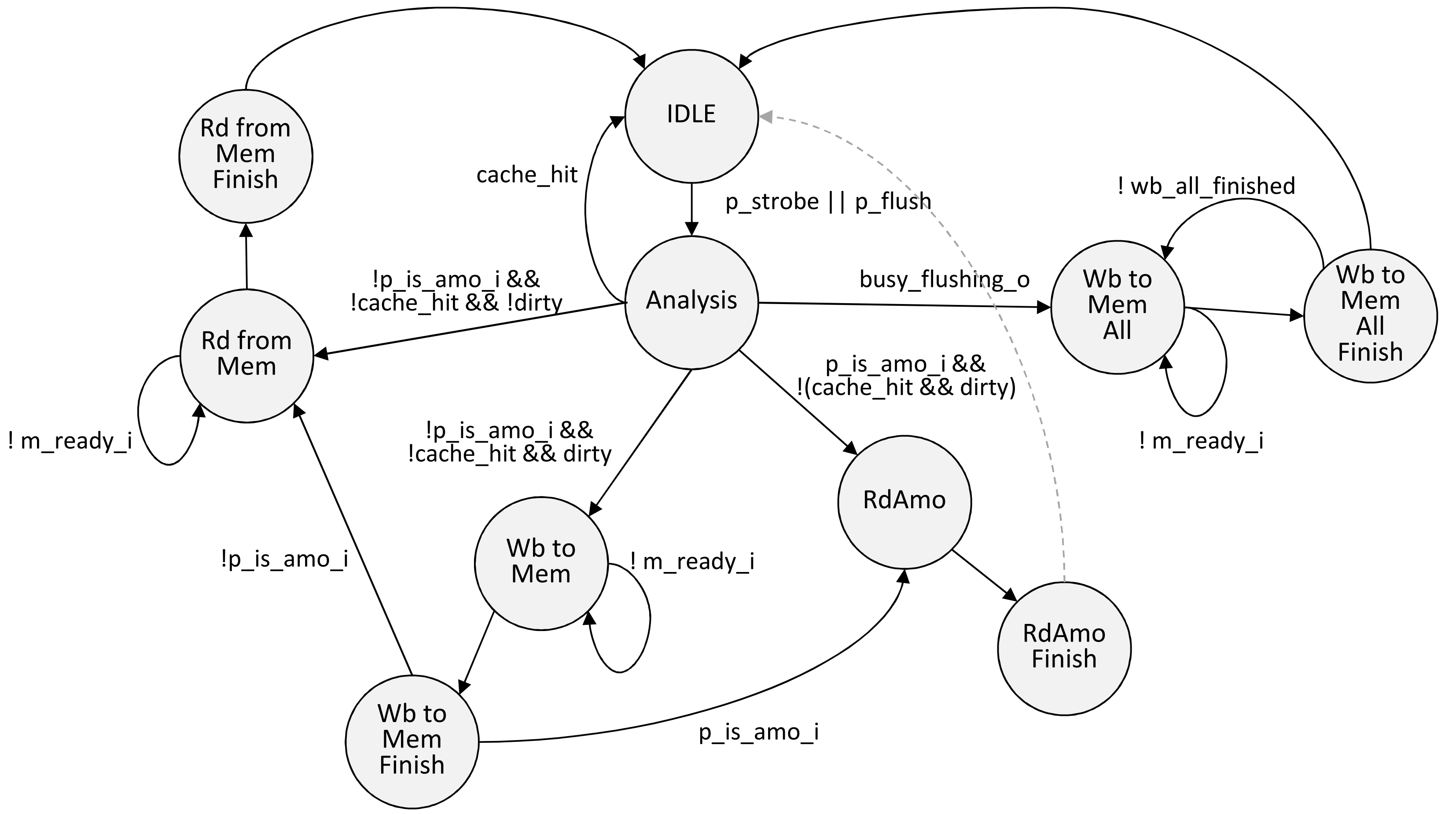
```
(* ram_style = “block" *) reg [0:31] my_ram [127:0];
```

    - Type of RAM styles are: block, distributed, ultra

  - BRAMs can only be used to synthesize a memory array with at most two clocked ports, each port controlled by enable and read/write signals

Cache Controller

- The FSM of the D-Cache controller is as follows:



Measuring Performance Hotspot

- You should add some counters in the cache controller to find the hotspot by collecting the following statistics
  - Average cache latency for each memory request
    - Read/write latency should be separated
    - Miss/hit latency should be separated
  - Cache hit/miss rates

- By latency, we mean the #cycles between the `p_strobe_i` and `p_ready_o`.

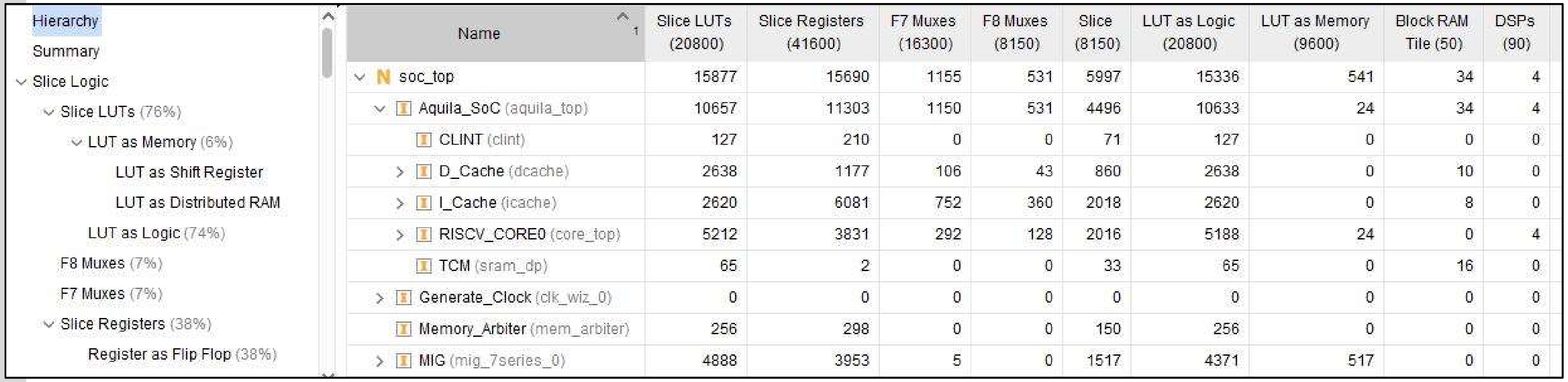
Things You Can Try

- There are a few things you can try to improve the performance of Aquila’s memory subsystem:
  - Change cache ways (0-way and 8-way are worth trying)
    - If direct-mapping is used, you should enlarge I-cache
  - Rewrite the 4-way cache controller to allow larger cache size
  - Redesign the cache controller based on the statistics you have collected
  - Applying good pre-fetching algorithm to I-cache

- Note that you can reduce the size of TCM to free more BRAMs for the caches

Current Resource Usage

- The FPGA resource utilization after PAR:



- Hierarchy
- Summary
- Slice Logic
  - Slice LUTs (76%)
    - LUT as Memory (6%)
      - LUT as Shift Register
      - LUT as Distributed RAM
    - LUT as Logic (74%)
  - F8 Muxes (7%)
  - F7 Muxes (7%)
  - Slice Registers (38%)
    - Register as Flip Flop (38%)

Name	Slice LUTs (20800)	Slice Registers (41600)	F7 Muxes (16300)	F8 Muxes (8150)	Slice (8150)	LUT as Logic (20800)	LUT as Memory (9600)	Block RAM Tile (50)	DSPs (90)
soc_top	15877	15690	1155	531	5997	15336	541	34	4
Aquila_SoC (aquila_top)	10657	11303	1150	531	4496	10633	24	34	4
CLINT (clint)	127	210	0	0	71	127	0	0	0
D_Cache (dcache)	2638	1177	106	43	860	2638	0	10	0
I_Cache (icache)	2620	6081	752	360	2018	2620	0	8	0
RISCV_CORE0 (core_top)	5212	3831	292	128	2016	5188	24	0	4
TCM (sram_dp)	65	2	0	0	33	65	0	16	0
Generate_Clock (clk_wiz_0)	0	0	0	0	0	0	0	0	0
Memory_Arbiter (mem_arbiter)	256	298	0	0	150	256	0	0	0
MIG (mig_7series_0)	4888	3953	5	0	1517	4371	517	0	0

Your Homework

- The goal of this homework is to improve the memory subsystem by modifying the I- and/or D-caches

- Write a report:
  - Describe how you collect the memory operation statistics and analyze the results
  - Describe your improvements to the memory subsystem