5.13 Real stuff: The ARM Cortex-A53 and Intel Core i7 memory hierarchies

> This section has been set as optional by your instructor.

In this section, we will look at the memory hierarchy of the same two microprocessors described in COD Chapter 4 (The Processor): the ARM Cortex-A53 and Intel Core i7. This section is in part based on COD Section 2.6 (Putting it all together: Memory hierarchies in the ARM Cortex-A8 and Intel Core i7) of *Computer Architecture: A Quantitative Approach*, 5th edition.

The figure below summarizes the address sizes and TLBs of the two processors. Note that the Cortex-A53 has two 10-entry fully associative micro-TLBs backed by a shared 512-entry four-way set associative main TLB with a 48-bit virtual address space and a 40-bit physical address space. The Core i7 has three TLBs with a 48-bit virtual address and a 44-bit physical address. Although the 64-bit registers of these processors could hold a larger virtual address, there was no software need for such a large space, and 48-bit virtual addresses shrinks both the page table memory footprint and the TLB hardware.

Figure 5.13.1: Address translation and TLB hardware for the ARM Cortex-A53 and Intel Core i7 920 (COD Figure 5.42).

Both processors provide support for large pages, which are used for things like the operating system or mapping a frame buffer. The large-page scheme avoids using a large number of entries to map a single object that is always present.

Characteristic	ARM Cortex-A53	Intel Core i7
Virtual address	48 bits	48 bits
Physical address	40 bits	44 bits
Page size	Variable: 4, 16, 64 KiB, 1, 2 MiB, 1 GiB	Variable: 4 KiB, 2/4 MiB
TLB organization	1 TLB for instructions and 1 TLB for data per core Both micro TLBs are fully associative, with 10 entries, round robin replacement 64-entry, four-way set-associative TLBs TLB misses handled in hardware	1 TLB for instructions and 1 TLB for data per core Both L1 TLBs are four-way set associative, LRU replacement L1 I-TLB has 128 entries for small pages, seven per thread for large pages L1 D-TLB has 64 entries for small pages, 32 for large pages The L2 TLB is four-way set associative, LRU replacement The L2 TLB has 512 entries TLB misses handled in hardware

The figure below shows their caches. The Cortex-A53 has between one and four processors or cores while the Core i7 is fixed at four. Cortex-A53 has a 16 to 64 KiB, two-way L1 instruction cache (per core) and the Core i7 has a 32 KiB, four-way set associative, L1 instruction cache (per core). Both use 64 byte blocks. The Cortex-A53 increases the associativity to four-way for the data cache, other variables remain the same. Similarly, the Core i7 keeps everything the same except the associativity, which it increases to eight-way. The Core i7 provides a 256 KiB, eight-way set associative unified L2 cache (per core) with 64 byte blocks. In contrast, the Cortex-A53 provides an L2 cache that is shared between one and four cores. This cache is 16-way set associative with 64 byte blocks and between 128 KiB and 2 MiB in size. As the Core i7 is used for servers, it also offers an L3 cache shared by all the cores on the chip. Its size varies depending on the number of cores. With four cores, as in this case, the size is 8 MiB.

Figure 5.13.2: Caches in the ARM Cortex-A53 and Intel Core i7 920 (COD Figure 5.43).

Characteristic	ARM Cortex-A53	Intel Core i7
L1 cache organization	Split instruction and data caches	Split instruction and data caches
L1 cache size	Configurable 16 to 64 KiB each for instructions/data	32 KiB each for instructions/data per core
L1 cache associativity	Two-way (I), four-way (D) set associative	Four-way (I), eight-way (D) set associative
L1 replacement	Random	Approximated LRU
L1 block size	64 bytes	64 bytes
L1 write policy	Write-back, variable allocation policies (default is Write-allocate)	Write-back, No-write-allocate
L1 hit time (load-use)	Two clock cycles	Four clock cycles, pipelined
L2 cache organization	Unified (instruction and data)	Unified (instruction and data) per core
L2 cache size	128 KiB to 2 MiB	256 KiB (0.25 MiB)
L2 cache associativity	16-way set associative	8-way set associative
L2 replacement	Approximated LRU	Approximated LRU
L2 block size	64 bytes	64 bytes
L2 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L2 hit time	12 clock cycles	10 clock cycles
L3 cache organization	–	Unified (instruction and data)
L3 cache size	–	8 MiB, shared
L3 cache associativity	–	16-way set associative
L3 replacement	–	Approximated LRU
L3 block size	–	64 bytes
L3 write policy	–	Write-back, Write-allocate
L3 hit time	–	35 clock cycles

A significant challenge facing cache designers is to support processors like the Cortex-A53 and the Core i7 that can execute more than one memory instruction per clock cycle. A popular technique is to break the cache into banks and allow multiple, independent, **parallel** accesses, provided the accesses are to different banks. The technique is similar to interleaved DRAM banks (see COD Section 5.2 (Memory technologies)).

The Cortex-A53 and the Core i7 have additional optimizations that allow them to reduce the miss penalty. The first of these is the return of the requested word first on a miss. They also continue to execute instructions that access the data cache during a cache miss. Designers who are attempting to hide the cache miss latency commonly use this technique, called a *nonblocking cache*. They implement two flavors of nonblocking. Hit under miss allows additional cache hits during a miss, while miss under miss allows multiple outstanding cache misses. The aim of the first of these two is hiding some miss latency with other work, while the aim of the second is overlapping the latency of two different misses.

Nonblocking cache: A cache that allows the processor to make references to the cache while the cache is handling an earlier miss.

Overlapping a large fraction of miss times for multiple outstanding misses requires a high-bandwidth memory system capable of handling multiple misses in parallel. In a personal mobile device, the memory system below it can often pipeline, merge, reorder, or prioritize requests appropriately. Large servers and multiprocessors typically have memory systems capable of handling several outstanding misses in parallel.

The Cortex-A53 and the Core i7 have prefetch mechanisms for data accesses. They look at a pattern of data misses and use this information to try to predict the next address to start fetching the data before the miss occurs. Such techniques generally work best when accessing arrays in loops.

The sophisticated memory hierarchies of these chips and the large fraction of the dies dedicated to caches and TLBs show the significant design effort expended to try to close the gap between processor cycle times and memory latency.

PARTICIPATION ACTIVITY 5.13.1: Memory hierarchies.

For each question, indicate whether the feature appears in the ARM Cortex-A53 only, the Intel Core i7 only, or both.

1) Two TLBs
   - ○ ARM Cortex-A53
   - ○ Intel Core i7
   - ○ Both

2) 44-bit physical address space
   - ○ ARM Cortex-A53
   - ○ Intel Core i7
   - ○ Both

3) LRU replacement scheme
   - ○ ARM Cortex-A53
   - ○ Intel Core i7
   - ○ Both

4) TLB misses handled in hardware

- ○ ARM Cortex-A53
- ○ Intel Core i7
- ○ Both

5) Fixed size of four processors (cores)

- ○ ARM Cortex-A53
- ○ Intel Core i7
- ○ Both

6) 4-way set associative L1 instruction cache

- ○ ARM Cortex-A53
- ○ Intel Core i7
- ○ Both

7) L3 cache

- ○ ARM Cortex-A53
- ○ Intel Core i7
- ○ Both

8) Nonblocking cache

- ○ ARM Cortex-A53
- ○ Intel Core i7
- ○ Both

Performance of the Cortex-A53 and Core i7 memory hierarchies

The memory hierarchy of the Cortex-A53 was measured using a 32 KiB two-way set associative L1 instruction cache, a 32 KiB four-way set associative L1 data cache, and a 1 MiB 16-way set associative L2 cache running the integer SPEC2006 benchmarks.

The Cortex-A53 instruction cache miss rates for these benchmarks are very small. The figure below shows the data cache results for the Cortex-A53, which have significant L1 and L2 miss rates. The L1 data cache miss rates go from 0.5% to 37.3%, with a mean of 6.4% and a median of 2.4%. The (global) L2 cache miss rates vary from 0.1% to 9.0%, with a mean of 1.3% and a median of 0.3%. The L1 miss penalty for a 1 GHz Cortex-A53 is 12 clock cycles, while the L2 miss penalty is 124 clock cycles. Using these miss penalties, COD Figure 5.45 (The average memory access penalty in clock cycles ...) shows the average miss penalty per data access. When these low miss rates are multiplied by their high miss penalties, you can see that they can represent a significant fraction of the CPI for 5 of the 12 SPEC2006 programs.

Figure 5.13.3: Data cache miss rates for ARM Cortex-A53 when running SPEC2006int (COD Figure 5.44).

Applications with larger memory footprints tend to have higher miss rates in both L1 and L2. Note that the L2 rate is the global miss rate; that is, counting all references, including those that hit in L1. (See the Elaboration in COD Section 5.4 (Measuring and improving cache performance).) mcf is known as a cache buster. Note that this figure is for the same systems and benchmarks as COD Figure 4.74 (CPI on ARM Cortex A8 for the Minnespec benchmarks ...) in COD Chapter 4 (The Processor).

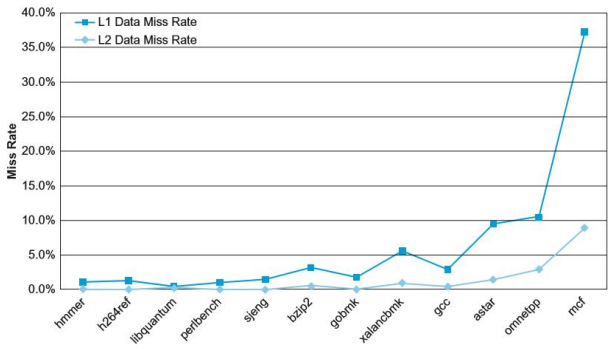
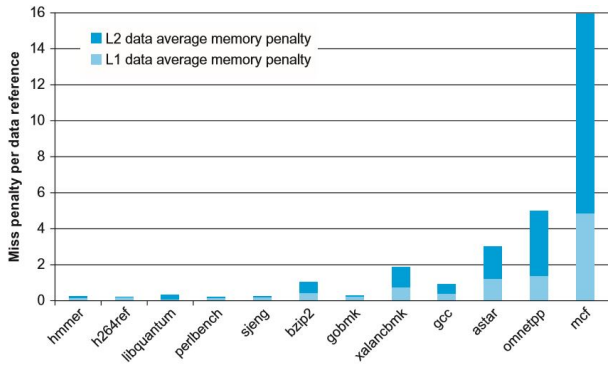


Figure 5.13.4: The average memory access penalty in clock cycles per data memory reference coming from L1 and L2 is shown for the ARM processor when running SPEC2006int (COD Figure 5.45).

Although the miss rates for L1 are significantly higher, the L2 miss penalty, which is more than five times higher, means that the L2 misses can contribute significantly.



The figure below shows the miss rates for the caches of the Core i7 using the SPEC2006 benchmarks. The L1 instruction cache miss rate varies from 0.1% to 1.8%, averaging just over 0.4%. This rate is in keeping with other studies of instruction cache behavior for the SPECCPU2006 benchmarks, which show low instruction cache miss rates. With L1 data cache miss rates running 5% to 10%, and sometimes higher, the importance of the L2 and L3 caches should be obvious. Since the cost for a miss to memory is over 100 cycles and the average data miss rate in L2 is 4%, L3 is obviously critical. Assuming about half the instructions are loads or stores, without L3 the L2 cache misses could add two cycles per instruction to the CPI! In comparison, the average L3 data miss rate of 1% is still significant but four times lower than the L2 miss rate and six times less than the L1 miss rate.

Figure 5.13.5: The L1, L2, and L3 data cache miss rates for the Intel Core i7 920 running the full integer SPECCPU2006 benchmarks (COD Figure 5.46).

Elaboration

Because speculation may sometimes be wrong (see COD Chapter 4 (The Processor)), there are references to the L1 data cache that do not correspond to loads or stores that eventually complete execution. The data in COD Figure 5.44 (Data cache miss rates for ARM Cortex-A53 …) are measured against all data requests, including some that are cancelled. The miss rate when measured against only completed data accesses is 1.6 times higher (an average of 9.5% versus 5.9% for L1 Dcache misses).

PARTICIPATION ACTIVITY 5.13.2: Performance of memory hierarchies.

1) The ____ benchmark is the most memory-intensive and causes high cache miss rates for both the ARM Cortex-A53's and Intel Core i7.

Check Show answer

2) In the Intel Core i7, the average ____ miss rate is four times lower than the L2 miss rate and six times less than the L1 miss rate.

Check Show answer

Provide feedback on this section