Chapter 8 problems 8.2, 8.5, 8.11, 8.14. Matt Kreger HW #10

8.2 [M] Consider a main memory built with SDRAM chips. Data are transferred in bursts as shown in Figure 8.9, except that the burst length is 8. Assume that 32 bits of data are transferred in parallel. If a 400-MHz clock is used, how much time does it take to transfer:

(*a*) 32 bytes of data

(*b*) 64 bytes of data

c) What is the latency in each case?

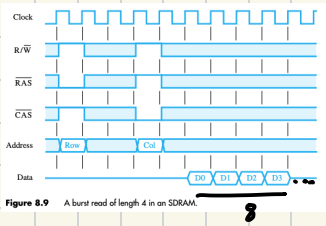


Figure 8.9 A burst read of length 4 in an SDRAM.

- 32 bits → 4 bytes transferred.
- Burst length: 8
- clk: 400 MHz → 4 × 8 = 32 bytes of data per burst

a) 32 bytes of data

$\frac{32 \text{ byte}}{4 \text{ byte}} = 8$ burst

$T = \frac{1}{f} = \frac{1}{400 \cdot 10^6} = 2.5$ ns per 'tick'

8 × 2.5 ns : 20 ns

b) 64 bytes

$\frac{64}{4} = 16$

16 × 2.5 = 40 ns

c) Latency for both cases is the initial burst of 2.5 ns

Chapter 8 problems 8.2, 8.5, 8.11, 8.14.

Matt Kreager HW #10

8.5 **[M]** The memory of a computer is byte-addressable, and the word length is 32 bits. A program consists of two nested loops—a small inner loop and a much larger outer loop. The general structure of the program is given in Figure P8.1. The decimal memory addresses shown delineate the location of the two loops and the beginning and end of the total program. All memory locations in the various sections of the program, 8-52, 56-136, 140-240, and so on, contain instructions to be executed in straight-line sequencing. The program is to be run on a computer that has an instruction cache organized in the direct-mapped manner (see Figure 8.16) with the following parameters:

Cache size	1K bytes
Block size	128 bytes

The miss penalty in the instruction cache is 80τ, where τ is the access time of the cache. Compute the total time needed for instruction fetching during execution of the program in Figure P8.1.

cache size : 1 kB → 1024 bytes

Block size : 128 Bytes → 1024/128 = 8 blocks

80τ miss penalty

τ hit time

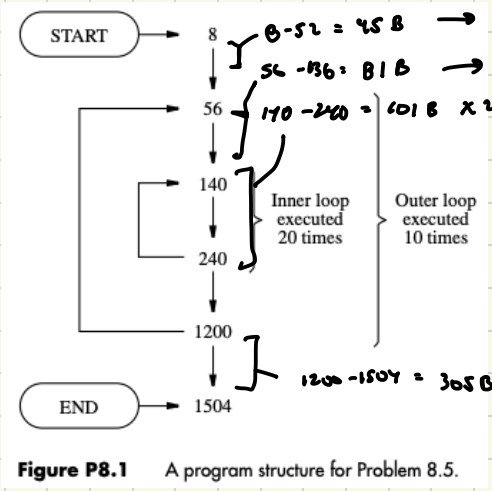


Figure P8.1 A program structure for Problem 8.5.

8-52 = 45 B → 1 block

56-136: 81 B → 2 blocks

140-240 = 101 B × 20 → 1 block

1200-1504 = 305 B → 3 blocks

cost

miss: 80τ

miss 80τ

→ exist: 80τ

inner loop: 20 iterations

exist block ×20

×10 : 20 · 80τ = 1600τ

miss 240τ

9 outer loops:

80τ + 80τ + 1600τ + 240τ

= 2000τ

Final

80τ + 80τ + 80τ + 1600τ misses

= 2080τ

×9 outer

18000τ

Total:

20,080 τ

Memory	Addr	Block #	Cache Index
8-52	0	0	0
56-136	0-1	0 & 1	0 & 1
140-240	1	1	1
1200-1504	9-11	9, 10, 11	1, 2, 3

Chapter 8 problems 8.2, 8.5, 8.11, 8.14.

Matt Kreager HW #10

8.11 [M] A byte-addressable computer has a small data cache capable of holding eight 32-bit words. Each cache block consists of one 32-bit word. When a given program is executed, the processor reads data sequentially from the following hex addresses:

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4

This pattern is repeated four times.

(*a*) Assume that the cache is initially empty. Show the contents of the cache at the end of each pass through the loop if a direct-mapped cache is used, and compute the hit rate.

(*b*) Repeat part (*a*) for an associative-mapped cache that uses the LRU replacement algorithm.

(*c*) Repeat part (*a*) for a four-way set-associative cache.

12 bits binary

Split it

	val	tag (7 bits)	Block (3 bits)	location (2 bits)
	200	0010000	000	00
	204	0010000	001	00
	208	0010000	010	00
→	20C	0010000	011	00
	2F4	0010111	101	00
	2F0	0010111	100	00
	218	0010000	110	00
	21C	0010000	111	00
→	24C	0010010	011	00

block holds eight 32 bit words (word length 4 bytes)

8 x 32 = 256 blocks, 3 addr bits

$\frac{32}{8}$ = 4 bytes → 2 addr bits

0	0	0	0	200
1	0	0	1	204
2	0	1	0	208
3	0	1	1	20C, 24C
4	1	0	0	2F0
5	1	0	1	2F4
6	1	1	0	218
7	1	1	1	21C

Blocks

a) Direct mapped cache

Hit Rate

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 2/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 10/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 10/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 10/12

= 32/48 · 100 = 66.7%

b) Least Recently Used

200 204 208 20C 2F4 2F0 218 24C

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 3/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 6/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 9/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 9/12

= 27/48 = 56.25%

c) four-way set associative

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 3/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 10/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 12/12

200, 204, 208, 20C, 2F4, 2F0, 200, 204, 218, 21C, 24C, 2F4 — 12/12

= 37/48 = 77.1%

Chapter 8 problems 8.2, 8.5, 8.11, 8.14.

Matt Kreager HW #10

8.14 [M] A computer has two cache levels L1 and L2. Plot two graphs for the average memory access time (y-axis) versus hit rate h_1 (x-axis) for the two values $h_2 = 0.75$ and $h_2 = 0.85$. Use the values 0.90, 0.92, 0.94, and 0.96, for h_1. Assume that the miss penalties are 15τ and 100τ for the L1 and L2 caches, respectively, where τ is the access time of the L1 caches.

$AMAT = h_1 \cdot T_1 + (1-h_1) \cdot (h_2 \cdot T_2 + (1-h_2) \cdot T_m)$

h_1 h_2

CASE 1:

$h_2 = 0.75$

$h_1 = 0.90$

$= .90\tau + (1-.90) \cdot [0.75 \cdot 15\tau + 0.25 \cdot 100\tau]$

$= 4.525\tau$

$h_1 = 0.92$

$= 0.92\tau + 0.08(11.25\tau + 25\tau)$

$= 3.82\tau$

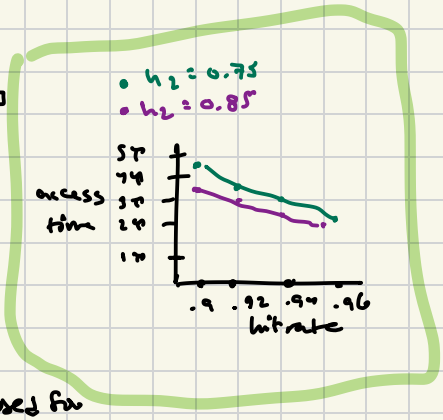
$h_1 = 0.94$

$= 3.115\tau$

$h_1 = 0.96$

$= 2.41\tau$

Same formula used for each changing value



CASE 2:

$h_2 = 0.85$

$h_1 = 0.90 \rightarrow 3.675\tau$

$h_1 = 0.92 \rightarrow 3.14\tau$

$h_1 = 0.94 \rightarrow 2.605\tau$

$h_1 = 0.96 \rightarrow 2.07\tau$