Student Name: **HU ID #:**

Habib University

EE-371/CS-330/CE-321 Computer Architecture – Spring 2024
Instructors: Dr. Tariq Kamal, Dr. Farhan Khan, Dr. Waseem Hassan

Time = 50 minutes **Quiz 04** **Max Points: 20**

Instructions:

i. ***Smart watches, laptops, and similar electronics are strictly NOT allowed.***
ii. ***Answer sheets should contain all steps, working, explanations, and assumptions.***
iii. Attempt the quiz on clean papers with black/blue ink.
iv. Print your name and HU ID on all sheets.
v. Turn in your question paper along with your answer sheets.
vi. You are not allowed to ask/share your method or answer with your peers. The work submitted by you is solely your own work. Any violation of this will be the violation of HU Honor code and proper action will be taken as per university policy if found to be involved in such an activity.

CLO Assessment:

This assignment assesses students for the following course learning outcomes.

Course Learning Outcomes		CLO Assessed
CLO 1	***Explain*** the role of ISA in modern processors and instruction encodings and assembly language programming	
CLO 2	***Explain*** the architecture and working of a single cycle processor	
CLO 3	***Design*** the architecture to mitigate issues of a pipelined processor	
CLO 4	***Analyze the*** performance of cache operations	√

Student Name: **HU ID #:**

Q1: Assume we have 64-bit addressing and a 64 KiB direct mapped cache having 4 words per block.

a) Calculate total size for this cache. [4 points]

Sol:

$2^m = 4 \rightarrow m = 2$ (block offset – 2)

64 KiB → 16 KiWords → 4 KiBlocks = 4096 blocks → $2^n = 4096 \rightarrow n = 12$ (index bits)

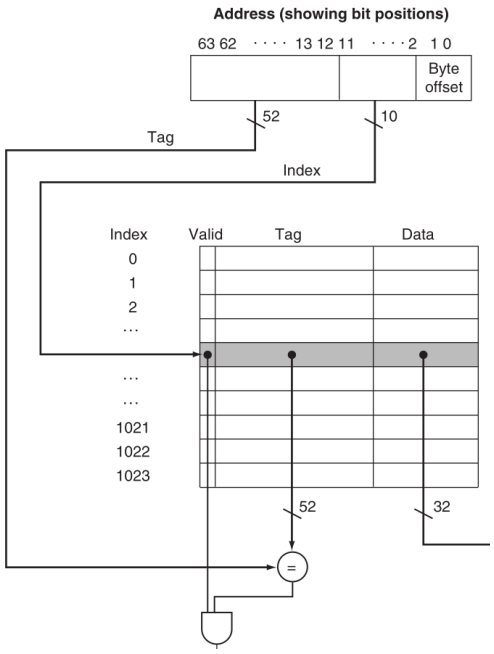
Tag bits = 64 – n – m – 2 = 64 – 12 – 2 – 2= 48

Size of a single cache line = tag bits + valid bit + block size = 48 + 1 + 4*32 = 177 bits

Size of cache = size of cache line * number of lines = 177 * 4096 = 724992 bits = 88.5 KiBytes

b) Draw hardware for the direct mapped cache showing division of memory address into tag, index and byte offset, along-with comparators and gates/multiplexers (if needed). Show bus-width(s) for each signal. [4 points]

Sol:



Notes:

1. Index will be 12 bits long, running from 0 to 4095
2. Tag will be 48 bits long
3. Data will be 4*32 bits long

Student Name: **HU ID #:**

c) If blocks per word is kept constant and we convert this cache into a 4-way set associative cache, find number of bits for the tag and index part. Also, redraw the hardware (part b) for this cache. [4 points]

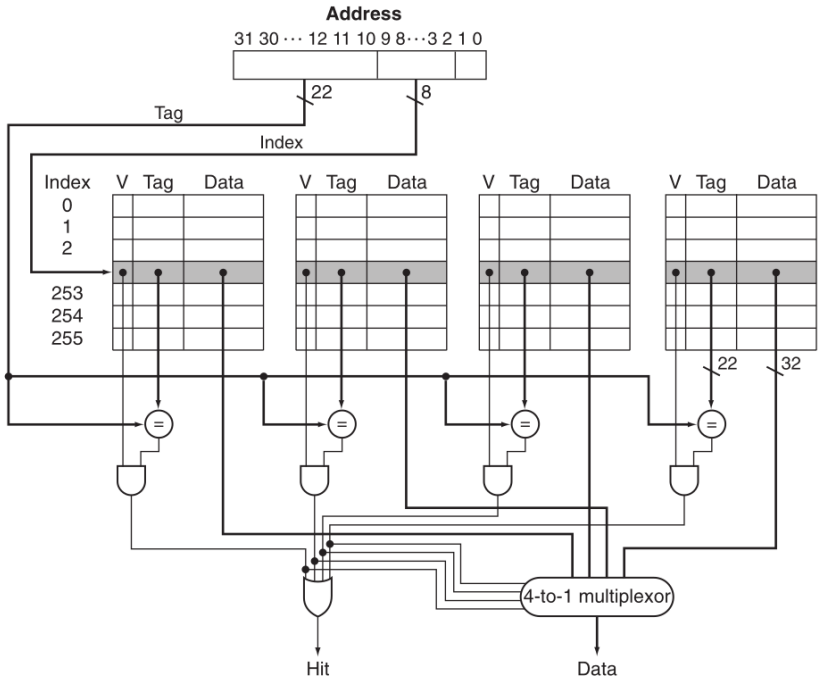
Sol:

Block offset i.e. words per block will remain same.

No. of sets = Total no. of blocks/Blocks per set = 4096/4 = 1024 sets = 2^{10}

Index → 10 bits from 0 till 1023

Tag_new = Tag_old + bits from index = 48 + 2 = 50 bits



Notes:

1. Index will be 10 bits long, running from 0 to 1023
2. Tag will be 50 bits long
3. Address will be 64 bits long
4. Data will be 4*32 bits long

Student Name: **HU ID #:**

Q2: Assume a processor is running at 2GHz. The main memory has an access time of 100 clock cycles. For this processor, a perfect cache (i.e. no misses) has a base CPI is 1.5.

a) Find the new CPI for this processor if it has a L1 instruction cache with a miss rate of 3% and L1 data cache with a miss rate of 5%. Assume that the data instructions constitute 33% of the total instructions. [4 points]

Sol:

CPI_new = CPI_base + CPI_instr + CPI_data

CPI_base = 1.5

CPI_inst = I*instr_miss_rate*miss_penalty = I*0.03*100 = 3I

CPI_data = I*percent_data_instr*data_miss_rate*miss_penalty = I*.33*0.05*100 = 1.65I

CPI_new = 1.5 + 3 + 1.65 = 6.15

Student Name: **HU ID #:**

b) If we add a common L2 cache which has an access time of 10 clock cycles and a miss rate of 1%, what will be the new CPI. Also find the relative speed up compared to part a. [4 points]

Sol:

CPI_new = CPI_base + CPI_instr_L1L2 + CPI_data_L1L2

CPI_instr_L1L2 = I*L1_instr_miss_rate*L1_miss_penalty + I*L2_instr_miss_rate*L2_miss_penalty

L1_instr_miss_rate = 3% - 1% = 2% L1_miss_penalty = 10 cc

L2_instr_miss_rate = 1% L2_miss_penalty = 10 + 100 cc

CPI_instr_L1L2 = I*(2/100)*10 + I*(1/100)*110 = 1.3I

CPI_data_L1L2 = I*percent_data_instr*L1_data_miss_rate*L1_miss_penalty + I*percent_data_instr *L2_data_miss_rate*L2_miss_penalty

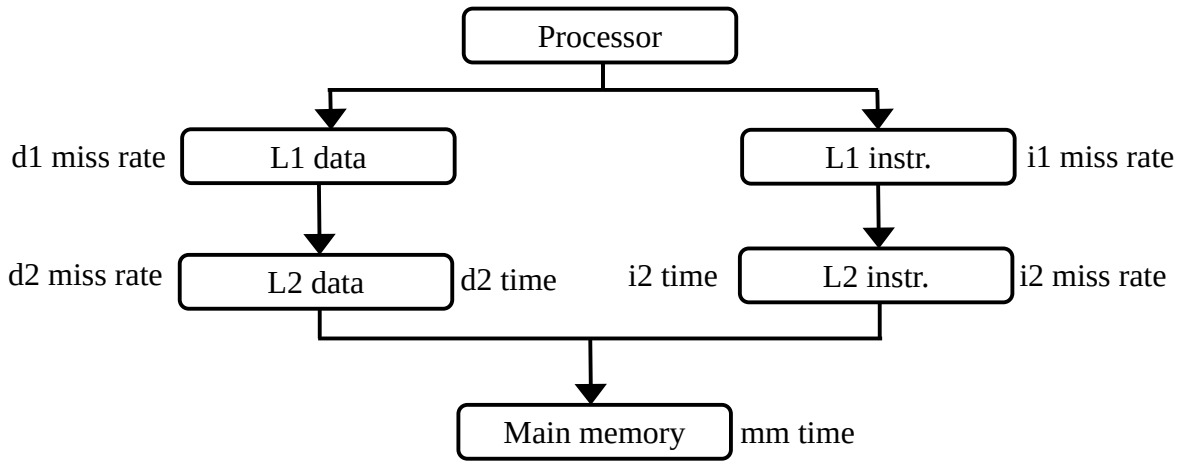
L1_data_miss_rate = 5% - 1% = 4% L2_data_miss_rate = 1%

CPI_data_L1L2 = I*.33*(4/100)*10 + I*.33*(1/100)*110 = 0.495I

CPI_new = 1.5 + 1.3 + 0.495 = 3.295

Speedup = 6.15/3.295 = 1.866

Generic method: (from book)



CPI_data = I*data_percent*(d1 miss rate*d2 time + d2 miss rate* mm time)
CPI_inst = I*(i1 miss rate*i2 time + i2 miss rate*mm time)
CPI_total = CPI_base + CPI_data + CPI_inst
Let's say L2 is missing i.e. we only have L1:
CPI_data = I*data_percent*(d1 miss rate* mm time)
CPI_inst = I*(i1 miss rate* mm time)
Access time means miss penalty for upper level.