CS152 Computer Architecture and Engineering
Computer Science Division
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

Sp97 D.K. Jeong

# Midterm #2 Solutions
April 23, 1997

| | |
|---|---|
| Name | Key |
| SID Number | |

You may bring one double-sided note. You have 170 minutes. Write your name on this cover and also at the top left of each page. The point value of each question is indicated in the bracket. Make sure to show your interim results to receive at least partial credit.

| Problem | Possible | Score |
|---|---|---|
| 1 | 15 | |
| 2 | 25 | |
| 3 | 20 | |
| 4 | 30 | |

| | | |
|---|---|---|
| Total | 90 | |

Name:

Question 1. [15 pts] (General)

(a) You are going to add a few new instructions to an existing ISA. How will you make old machines be able to run new codes?

Modify the exception handler in the old machine so that the handler, on unimplemented instruction exceptions on new instructions, emulates the behavior of the new instructions.

(b) Explain why a direct-mapped TLB is a bad idea.

Since one TLB entry is allocated for one page, unlike a cache where a cache entry is in blocks, locality of reference is expected to be a lot less. Furthermore, for small programs, the TLB index tends to be concentrated in all 0’s, differing only in address tags - MSB’s are used for segments such as data, heap, stack… Ping-pong effect will be significant.

(c) What are the particular characteristics of multimedia IOs compared to conventional IOs such as disk or mouse?

High bandwidth is required for video. Isochronous data transfer or guaranteed bandwidth is required for real-time applications. Synchronization among different IOs such as video and audio may be required.

Conventional IO’s are less bandwidth intensive and their delay tends to be dominated by the overhead (or setup time) of the devices.

Name:

Question 2. [25 pts] (Cache) The following table shows the address trace. Assume that address and data are 8-bit wide. Cache size is 16 bytes and block size is 2 bytes. Write policy is copy-back and LRU algorithm is used for replacement. Assume write allocate policy on write miss. Assume that all cache entries are initially invalid and its contents are Xs except Valid bits. (a) Check if each access is a hit or miss. (b) Draw the content of the cache when all the address trace is consumed. Assume that the content of the main memory is its address - e.g. address 0xC7 contains 0xC7. LRU is not shown but you have to keep it mentally or on a paper.

| Access Type | Address | Data | hit/miss? |
|---|---|---|---|
| Read | 00000000 | | Miss |
| Write | 00000001 | 0xFF | Hit |
| Read | 00001000 | | Miss |
| Write | 01000100 | 0xEE | Miss |
| Read | 00101100 | | Miss |
| Read | 10100011 | | Miss |
| Read | 00000000 | | Hit |

| Way 0 | | | | | Way 1 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| V | D | Add. tag | Data1 | Data0 | V | D | Add. tag | Data1 | Data0 |
| 1 | 1 | 000000 | 0xFF | 0x00 | 1 | 0 | 000010 | 0x09 | 0x08 |
| 1 | 0 | 101000 | 0xA3 | 0xA2 | 0 | x | x | x | x |

| Way 2 | | | | | Way 3 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| V | D | Add. tag | Data1 | Data0 | V | D | Add. tag | Data1 | Data0 |
| 1 | 1 | 010001 | 0x45 | 0xEE | 1 | 0 | 001011 | 0x2D | 0x2C |
| 0 | | x | x | x | 0 | x | x | x | x |

Address breaks down like this:

| Tag | Index | Block offset |
|---|---|---|
| 6 bits | 1 bit | 1 bit |

Name:

Question 3. [20 pts] (System Performance) A pipelined processor with a 100MHz clock is connected to a 50MHz bus through a 4Kbyte instruction and 8Kbyte data cache. Their block sizes are 16 bytes and 32 bytes, respectively. Do not assume any extra cycles for dirty block copy-back for a cache fill since a write buffer will handle it later. Main memory access time is 200ns for the first 32-bit data and the subsequent 32-bit data in a page mode access takes 50ns. Assume instruction cache and data cache responds in the same cycle when hit. Assume miss ratio is 6% for the instruction cache and 4% for the data cache.

(a) What is the system's MIPS?

CPI=CPI_base + miss_cycles/instruction
Assume CPI_base = 1.0
Miss_cycles/instr = Instr_MissRate*Instr_MissPen + %DataRef *Data_MissRate *Data_MissPen

Miss Penalty is 200 ns to fetch first 32 bits, and 50 ns to fetch each additional 32 bits. However, bus cycle = 1/50 MHz = 20 ns, so page mode access cannot take 2.5 bus cycles—it must take 3 cycles or 60 ns.
Instr_MissPen = (200 ns + 3*60 ns )* 1 cycle / 10 ns = 38 cycles
Data_MissPen = (200 ns + 7*60 ns) * 1 cycle / 10 ns = 62 cycles
Miss_cycles/instr = (.06)*38 ns + (.25)*(.04)*62 ns = 2.90
CPI = 1.0 + 2.90 = 3.90
MIPS = clk rate / CPI = 100 MHz / 3.90 = 25.6410 MIPS

(b) Assume that the main memory spends 5 bus cycles every 16ms on refreshing. During that time, memory access is delayed until the refresh cycle finishes. What is the new MIPS with refreshing?

Refresh will only hurt performance if the CPU needs to access memory at the same time a refresh operation is occurring. Since refresh takes 5 bus cycles = 100 ns, the probability of a conflict is 100 ns / 16 ms = 6.25 x $10^{-6}$.
However, we won't always have to wait 5 bus cycles on a conflict. The average number of bus cycles you need to wait is (1+2+3+4+5)/5 = 3.0 bus cycles = 6 processor cycles. So the average penalty per memory access increases by 6*6.25 x $10^{-6}$ = 3.75x$10^{-5}$.
CPI_refresh = 1.0 + 2.9*(1+3. 75x$10^{-5}$) = 3.90010875
MIPS_refresh = 100 MHz / CPI_refresh = 25.640311 MIPS

(c) An IO controller for a mouse is attached to the bus. Assuming that 200 instructions are executed for handling an IO interrupt with the same cache miss rate, what is the new MIPS (not counting instructions in the interrupt handler) when 30 interrupts are generated per second?

# of instructions / sec for I/O = 30 interrupts/sec * 200 Instr/interrupt = 6000 instr/sec
The processor is computing at the same speed as before, but 6000 instr/sec are not part of the program execution, and so need to be deducted from the MIPS.

MIPS_mouse = 25.6410 MIPS – .006 MIPS = <u>25.635 MIPS</u>

(d) The block sizes of the caches are doubled to reduce the miss rate. The resulting cache miss rate is 5% for I cache and 3% for D cache. Will the performance be increased compared to (a), why?

Although the miss rate is reduced, the miss penalty almost doubles, since the caches size has doubled.

Instr_MissPen = (200 ns + 7*60 ns) * 1 cycle / 10 ns = 62 cycles
Data_MissPen = (200 ns + 15*60 ns) * 1 cycle / 10 ns = 110 cycles

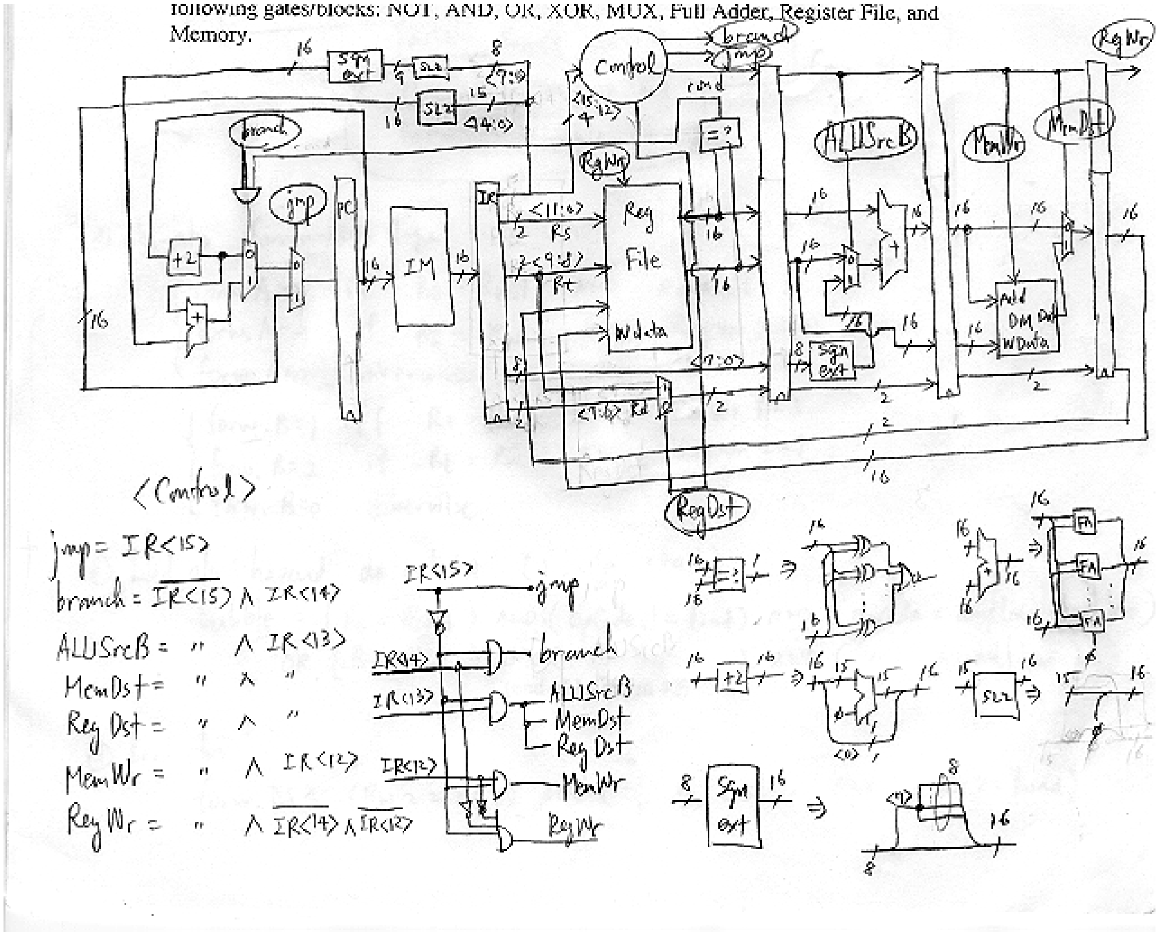
Miss_cycles / Instr = (.05)*62 cycles + (.25)*(.03)*110 cycles =3.925 miss_cycles/instr
CPI = 1.0 + 3.925 = 4.925
MIPS_double = 20.30 MIPS, which is almost 80% slower.

Name:

Question 4. [30 pts] (Pipelined CPU Design) You are to design a pipelined 16-bit microprocessor with only 5 instructions. Use a 5-stage pipeline and assume delayed branch with one delay slot. The register file contains 4 data words.

| Instruction | Format | | | | | Description |
|---|---|---|---|---|---|---|
| add | 0000 | Rs | Rt | Rd | XXXXXX | R[Rd] ← R[Rs] + R[Rt] |
| load | 0010 | Rs | Rt | Displacement | | R[Rt] ← M[R[Rs] + Displacement]] |
| store | 0011 | Rs | Rt | Displacement | | M[R[Rs] + Displacement]] ← R[Rt] |
| beq | 0100 | Rs | Rt | Displacement | | branch to PC+2+displacement*2<br>if R[Rs] = R[Rt] |
| jmp | 1 | target | | | | unconditional jump to target*2 |

(a) Draw a pipelined datapath, define control signals, and design a controller in the gate level. Make sure to indicate the width of buses. Minimize logic as much as possible. Assume that branch or jump instructions cannot be in the delay slot. Use only the following gates/blocks: NOT, AND, OR, XOR, MUX, Full Adder, Register File, and Memory.


following gates/blocks: NOT, AND, OR, XOR, MUX, Full Adder, Register File, and Memory.

<Control>

jmp = IR<15>

branch = $\overline{IR<15>}$ ∧ IR<14>

ALUSrcB = " ∧ IR<13>

MemDst = " ∧ "

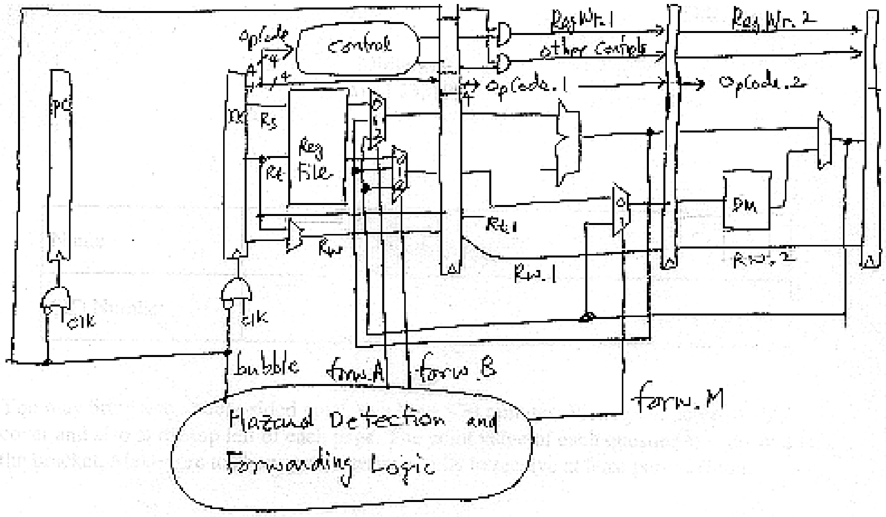
Reg Dst = " ∧ "

MemWr = " ∧ IR<12>

Reg Wr = " ∧ $\overline{IR<14>}$ ∧ $\overline{IR<12>}$

Name:

(b) Add a hazard detection logic and an internal forwarding mechanism to handle RAW hazards. (It must be able to handle alu-alu, load-alu, and load-store instruction sequences correctly.)



① alu-alu forwarding logic

forw.A = 1 if Rs = Rw.1 AND RegWr.1 = 1
forw.A = 2 if Rs = Rw.2 AND RegWr.2 = 1
forw.A = 0 otherwise

forw.B = 1 if Rt = Rw.1 AND RegWr.1 = 1
forw.B = 2 if Rt = Rw.2 AND RegWr.2 = 1
forw.B = 0 otherwise

② load-alu hazard detection (1 cycle stall)

bubble = (Rs = Rw.1) AND (opCode.1 = load) AND (opcode = add|load|store|bne)
OR (Rt = Rw.1) AND ( " ) AND ( " = add|bne)

(end of Midterm #2)

③ load-store

forw.M = (Rw.2 = Rt.1) AND (Opcode.1 = store) AND (opCode.2 = load)

(end of Midterm #2 solutions)