Instructions

Format: This exam consists of:

4 PROBLEMS for a total of 70 points. Use the space provided for your answers. You may use additional pages as a scratchpad but keep your final answers only on the exam.

15 Multiple choice questions for a total of 30 points. Circle the correct answer.

The total is 100 points:
Fail: 54 or below
Grade 3: 55-70 points
Grade 4: 71-85 points
Grade 5: 86-100 points
Participation bonus (based on workshop attendance): 10 points.

YOU MUST RETURN ALL PAGES.

Good luck!

PROBLEMS

1. [20 points total] Write a MIPS assembly function to copy a string of characters to a new location in memory. Use the space provided for your code.

Function arguments:

$a0: address of the original string
$a1: address to copy the string

Return values:

$v0: address of the new string
$v1: length of the string

Note:

- String is null-terminated (i.e. ends with the NULL character with ASCII code 0)
- Your code should adhere to MIPS register convention (saved/non-saved registers.)
- Use the space provided for your code.

REGISTER NAME, NUMBER, USE, CALL CONVENTION

NAME	NUMBER	USE	PRESERVED ACROSS A CALL?
$zero	0	The Constant Value 0	N.A.
$at	1	Assembler Temporary	No
$v0-$v1	2-3	Values for Function Results and Expression Evaluation	No
$a0-$a3	4-7	Arguments	No
$t0-$t7	8-15	Temporaries	No
$s0-$s7	16-23	Saved Temporaries	Yes
$t8-$t9	24-25	Temporaries	No
$k0-$k1	26-27	Reserved for OS Kernel	No
$gp	28	Global Pointer	Yes
$sp	29	Stack Pointer	Yes
$fp	30	Frame Pointer	Yes
$ra	31	Return Address	Yes

MIPS Reference Data

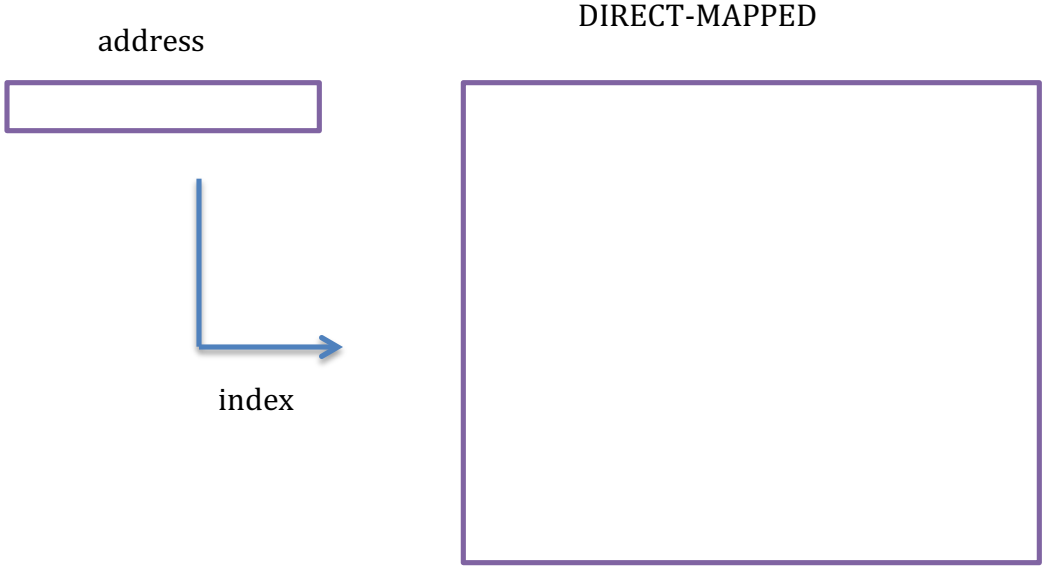
Type	Instruction	Example	Operation
Arithmetic	Add	add $s3, $s2, $s1	$s3 = $s2 + $s1
	Add Immediate	addi $s2, $s1, -1	$s2 = $s1 + SignExtImm[1]
	Sub	add $s3, $s2, $s1	$s3 = $s2 - $s1
	Sub Immediate	subi $s2, $s1, 1	$s2 = $s1 – SignExtImm
Data Movement	Load Word	lw $s1, 0($s2)	$s1 = memory[$s2 + SignExtImm]
	Store Word	sw $s1, 0($s2)	memory[$s2 + SignExtImm] = $s1
Logical	And	and $s1, $s2, $s3	$s1 = $s2 & $s3
	And Immediate	andi $s1, $s2, 1	$s1 = $s2 & ZeroExtImm[2]
	Or	or $s1, $s2, $s3	$s1 = $s2 | $s3
	Or Immediate	ori $s1, $s2, 1	$s1 = $s2 | ZeroExtImm
	Shift Left Logical	sll $s1, $s2, 1	$s1 = $s2 << shamt[3]
	Shift Right Logical	srl $s1, $s2, 1	$s1 = $s2 >> shamt
Unconditional Jump	Jump	j label	pc = {pc[31:28],address[25:0],00}[4]
	Jump Register	jr $s1	pc = $s1
	Jump and Link	jal label	$ra = pc; pc = {pc[31:28],address[25:0],00}
Conditional Branch	Branch Equal	beq $s1, $s2, label	if ($s1 == $s2) pc = pc + 4 + SignExt{Imm,00}
	Branch Not Equal	bne $s1, $s2, label	if ($s1 != $s2) pc = pc + 4 + SignExt{Imm,00}
	Set Less Than	slt $s1, $s2, $s3	if ($s2 < $s3) $s1 = 1
	Set Less Than Immediate	slt $s1, $s2, 100	if ($s2 < SignExtImm) $s1 = 1
Pseudo-Instructions	Jump and Link	jalr $s1	$ra = pc; pc = $1
	Branch Less Than	blt $s1, $s2, label	if ($s1 < $s2) pc = pc + 4 + SignExt{Imm,00}
	Branch Greater Than	bgt $s1, $s2, label	if ($s1 > $s2) pc = pc + 4 + SignExt{Imm,00}
	Branch Less Than or Equal	ble $s1, $s2, label	if ($s1 <= $s2) pc = pc + 4 + SignExt{Imm,00}
	Branch Greater Than or Equal	bge $s1, $s2, label	if ($s1 >= $s2) pc = pc + 4 + SignExt{Imm,00}
	Load Immediate	li $s1, 1	$s1 = 1

1. SignExt: sign extension, i.e. repeating the sign bit to create a 32-bit value
   Imm: immediate field of the instruction word; see instruction formats in question 3
2. ZeroExt: zero extension, i.e. filling the MSB part with zeros to create a 32-bit value
3. Shamt: shift amount; see instruction formats in question 3
4. Address: address field of the instruction word; see instruction formats in question 3

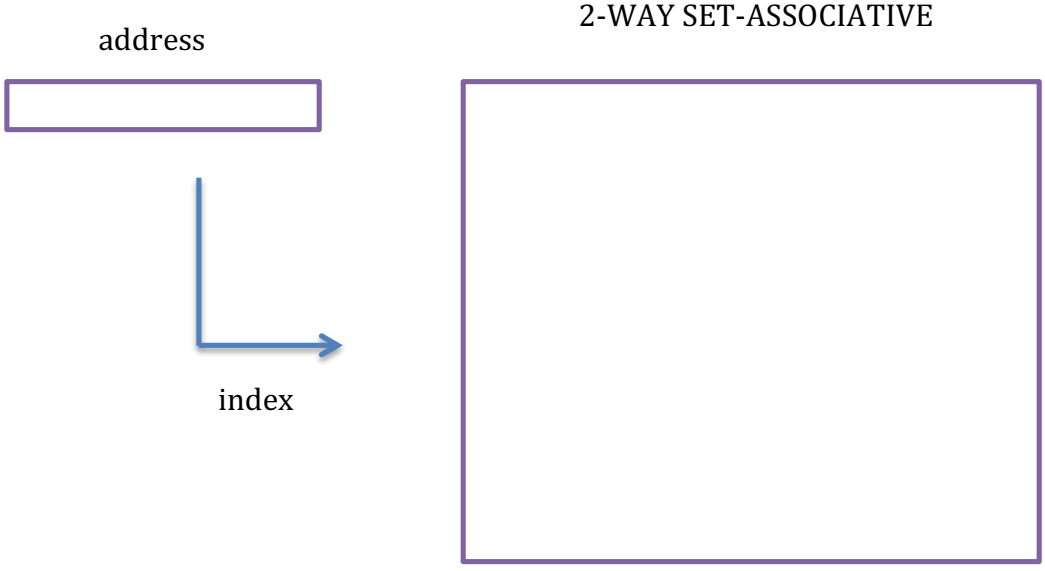
Label	Instruction	Comment
addarray:		Start your function here

2. [20 points total] Using 4 lines of 2-words each, consider a Direct-mapped cache and a 2-way Set-Associative cache (you can draw in the boxes provided below for help but it is not required). Assume that the address consists of 32-bits and each word is 4 bytes. Compare the two organizations in the following table.

DIRECT-MAPPED



2-WAY SET-ASSOCIATIVE

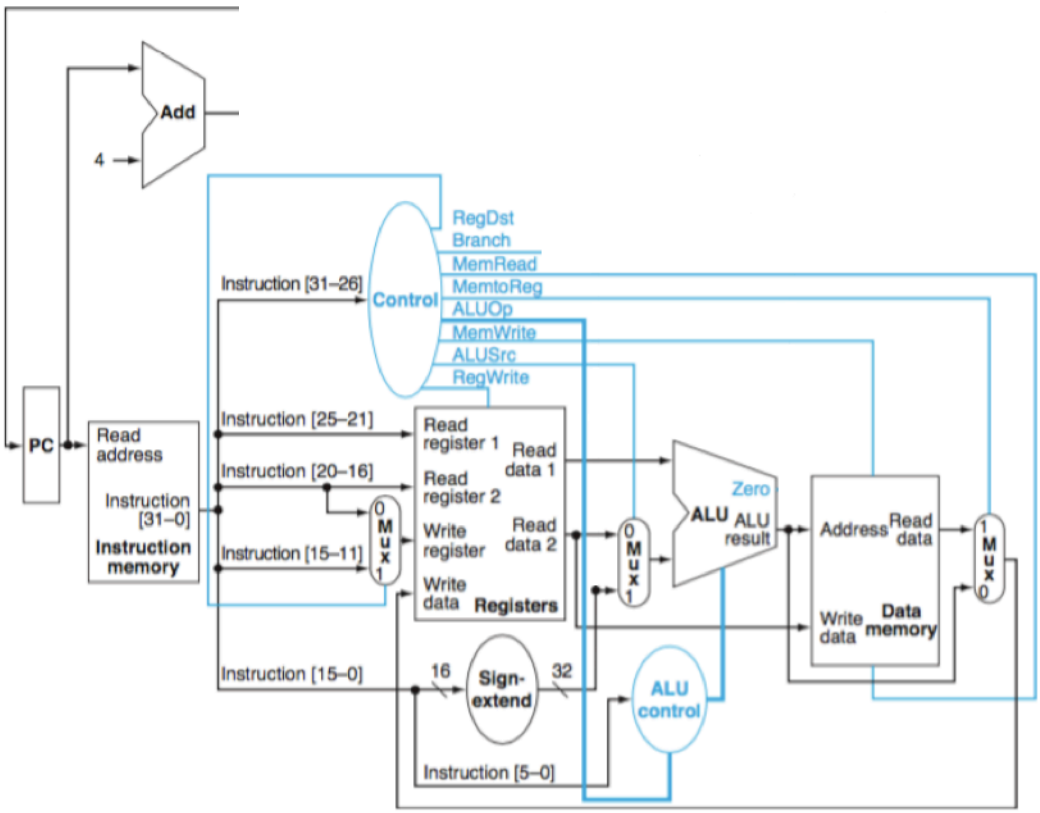


Compare (with short justification where needed) the two organizations with respect to:

a) Size of the address field that refers to the block	
b) Size of index (e.g., which organization has smaller index)	
c) Size of tag (e.g., which organization has smaller tag)	
d) Overall complexity (which one is simpler overall)	
e) Block placement flexibility	
f) Complexity to determine hit/miss	
g) Complexity of replacement algorithm	

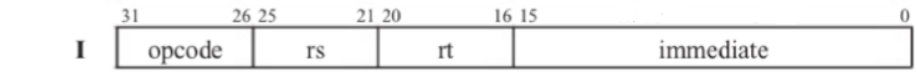
h) Speed (which organization you think is faster as a circuit)	
i) Miss rate for a typical address stream found in ordinary programs	
j) Miss rate for the following repeating address pattern (5 consecutive blocks): 0,4,8,12,16, 0,4,8,12,16, 0,4,8,12,16 **Assume LRU replacement for the Set-Associative Cache.**	

3. [20 Points] A basic MIPS processor is given as follows but a few circuits are missing. Implement "branch" instruction by adding the necessary components to the processor. Clearly explain your design using the space provided in the next page.



Mnemonic	Name	Opcode-funct	Type	Operation
beq	Branch Equal	000100- NA	I-Format	if (R[rs] == R[rt]) pc = pc + 4 + {SignExt{Imm},00}

BASIC INSTRUCTION FORMATS



4. [10 Points total] What is the speed-up for a processor with a 100ns clock frequency that is equally pipelined to 20 stages? Each pipeline register has the latency of 5ns. Justify your answer.

5. Multiple Choice [30 points total, each question 2 points]

1. How do you negate a two's complement number?
    a. Invert the most significant bit
    b. Invert
    c. Invert and add 1
    d. Invert and shift 1 step to the left
2. How many additions are needed for a serial multiplier to multiply 1010 times 1000?
    a. 1
    b. 2
    c. 4
    d. 16
3. The combinational logic does what in a sequential circuit?
    a. Store the current value
    b. Store the next value
    c. Calculate the current value
    d. Calculate the next value
4. What limits the speed of a sequential circuit?
    a. The slowest latch
    b. The slowest path through the combinational logic
    c. The slowest latch + the slowest path through the combinational logic
    d. The longest instruction
5. Which is faster, and why? DRAM or SRAM?
    a. DRAM: each bit is stored in a powered feedback loop so it can be read quickly
    b. DRAM: each bit is stored in a passive capacitor so it can be read quickly
    c. SRAM: each bit is stored in a powered feedback loop so it can be read quickly
    d. SRAM: each bit is stored in a passive capacitor so it can be read quickly
6. What happens if you activate two rows in a memory array at the same time?
    a. You corrupt the data you are reading
    b. You can read out twice as much data
    c. You can read and write at the same time
    d. You can read and write at the same time as long as it is to different rows
7. Double-pumping the register file fixed what kind of hazard?
    a. Data hazard
    b. Control hazard
    c. Pipeline Register hazard
    d. Structural hazard
8. When can we forward data in a pipeline?
    a. When the data is in the wrong place
    b. When we need the data from an instruction before us
    c. When multiple instructions need the same hardware at once
    d. All of the above
9. What is the minimum number of simultaneous reads and writes needed for a register file to work with the MIPS ISA?

a. 0 writes, 2 reads
b. 1 write, 2 reads
c. 1 write, 3 reads
d. 2 writes, 3 reads

10. What are the two key performance metrics for I/O?
    a. Throughput and latency
    b. Throughput and longevity
    c. Latency and longevity
    d. Density and longevity
11. Direct Memory Access (DMA) allows the I/O device to do what?
    a. Directly trigger interrupts when data is available
    b. Directly get access to the bus master
    c. Directly transfer data to the CPU's memory
    d. Directly modify the program's data
12. What is the benefit of polling?
    a. Simpler to program
    b. Uses more the processor for I/O
    c. Higher data throughput
    d. All of the above
13. If I have an Intel Xeon Phi processor with 62 cores, what is the best speedup I could get from a program that is 99% parallelizable?
    a. 38.5
    b. 58.4
    c. 61.4
    d. 62.0
14. Do page tables need a dirty bit? (hint: is Virtual Memory Write-Through or Write-back ?)
    a. No, they don't store data, just VA-to-PA mappings
    b. No, they are read only
    c. Yes, the bit indicates if the data is on disk
    d. Yes, the bit indicates if the page has been changed since it was loaded from disk
15. Writing to which address with the following page table would result in a page being loaded from disk?
    e. 3
    f. 4
    g. 8
    h. 12

VA	PA	On disk	Access bits
15	2	0	Read/Write
3	27	1	Read only
4	15	0	Read copy on write
12	7	0	Read only
8	14	1	Read/Write