ECE 2030C 2:00pm	Computer Engineering	Fall 2010
4 problems, 6 pages	Exam Three	1 December 2010

Instructions: This is a closed book, closed note exam. Calculators are not permitted. If you have a question, raise your hand and I will come to you. Please work the exam in pencil and do not separate the pages of the exam. For maximum credit, show your work.
Good Luck!

Your Name (***please print***) _____

1	2	3	4	total
30	23	30	17	100

Problem 1 (3 parts, 30 points) Memory Systems

Part A (12 points) Consider a DRAM chip organized as **4 million addresses** of **16-bit words**. Assume both the DRAM cell and the DRAM chip are square. The column number and offset concatenate to form the memory address. Using the organization approach discussed in class, answer the following questions about the chip. *Express all answers in decimal (not powers of two)*.

total number of bits in address _____

number of columns _____

column decoder required (*n* to *m*) _____

number of words per column _____

type of mux required (*n* to *m*) _____

number of address lines in column offset _____

Part B (10 points) Consider a memory system with **256 million addresses** of **128-bit words** using **4 million** address by **16-bit word** memory DRAM chips.

word address lines for memory system _____

chips needed in one bank _____

banks for memory system _____

memory decoder required (*n* to *m*) _____

DRAM chips required _____

Part C (8 points) Design a **16 million address by 8 bit** memory system with **8 million x 4 bit** memory chips. *Label all busses and indicate bit width*. Assume R/W is connected and not shown here. Use a bank decoder if necessary. Be sure to include the address bus, data bus, and MSEL.

Problem 2 (3 parts, 23 points) Datapath Elements and State Machines

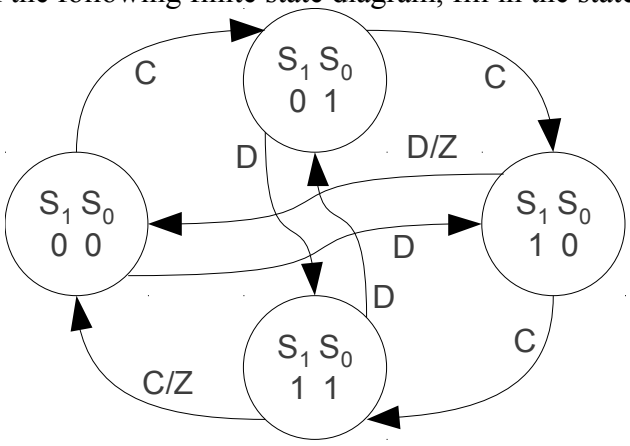
Part A (6 points) Suppose the following inputs (in `hexadecimal`) are applied to the 32-bit barrel shifter used in the datapath. Determine the output (in `hexadecimal`). Assume the shift amount is drawn from the 16-bit immediate value.

Shift Type	Shift Amount	Input Value	Output Value
logical	0x0010	BEAD2533	
arithmetic	0x0014	39317ACE	
rotate	0xFFF8	FADE3650	

Part B (9 points) Consider the following input and output values for a logical operation. Determine the *logical function and function code* (in hexadecimal) required for the operation.

X Input	Y Input	Output	Logical Function	Function Code
DCBA4321	FF00FF00	FFBAFF21		
DCBA4321	FF00FF00	23FFBCFF		
DCBA4321	FF00FF00	00FF00FF		

Part C (8 points) Given the following finite state diagram, fill in the state table below.



S_1	S_0	C/$\overline{D}$	NS_1	NS_0	Z	S_1	S_0	C/$\overline{D}$	NS_1	NS_0	Z
0	0	0				1	0	0			
0	0	1				1	0	1			
0	1	0				1	1	0			
0	1	1				1	1	1			

Give the simplified Boolean expression for computing **Z** in terms of the current state and the input.

Z = _____.

Problem 3 (5 parts, 30 points) Microcode

For Parts A-C, using the supplied datapath, write microcode fragments to accomplish the following procedures. Express all values, except memory addresses, in hexadecimal notation. Use 'X' when a value is don't cared. For maximum credit, complete the description field.

Part A (5 points) $6 ← mem[$4]. **Use only registers 4 and 6**.

#	X	Y	Z	rwe	im en	im va	au en	s/a	lu en	lf	su en	st	ld en	st en	r/w	msel	description
1																	

Part B (8 points) Multiply $9 by 33 and put the result in $9. **Use only registers 7 and 9**.

#	X	Y	Z	rwe	im en	im va	au en	s/a	lu en	lf	su en	st	ld en	st en	r/w	msel	description
2																	
3																	

Part C (8 points) mem[0x2020] ← $5. **Use only registers 1 and 5**.

#	X	Y	Z	rwe	im en	im va	au en	s/a	lu en	lf	su en	st	ld en	st en	r/w	msel	description
4																	
5																	

Part D (4 points) Write the MIPS instruction that is equivalent to the following microinstruction. (A summary of MIPS instructions is given on the next page.)

#	X	Y	Z	rwe	im en	im va	au en	s/a	lu en	lf	su en	st	ld en	st en	r/w	msel	description
6	2	8	7	1	0	x	0	x	1	6	0	x	0	0	x	0	

Equivalent MIPS Instruction: _____

Part E (5 points) Write the MIPS instruction that is equivalent to the following microinstruction. (A summary of MIPS instructions is given on the next page.)

#	X	Y	Z	rwe	im en	im va	au en	s/a	lu en	lf	su en	st	ld en	st en	r/w	msel	description
7	3	x	6	1	1	FFFD	0	x	0	x	1	0	0	0	x	0	

Equivalent MIPS Instruction: _____

Problem 4 (3 parts, 17 points) Assembly Programming

For maximum credit in each of the following problems, *use the fewest number of instructions*.

Part A (6 points) Write a MIPS code fragment that reads an integer value from memory location 1000 and puts it into register $6. **Use only registers $0 and $6.**

label	*instruction*	*comment*

Part B (7 points) Write a MIPS code fragment that branches to the label `Target` if register $3 ≥ $4. **Use only registers $3, $4, $2, and $0.**

label	*instruction*	*comment*

Part C (4 points) Write a MIPS code fragment that divides the integer in register $7 by 32 and put the result in register $7. **Use only register $7.**

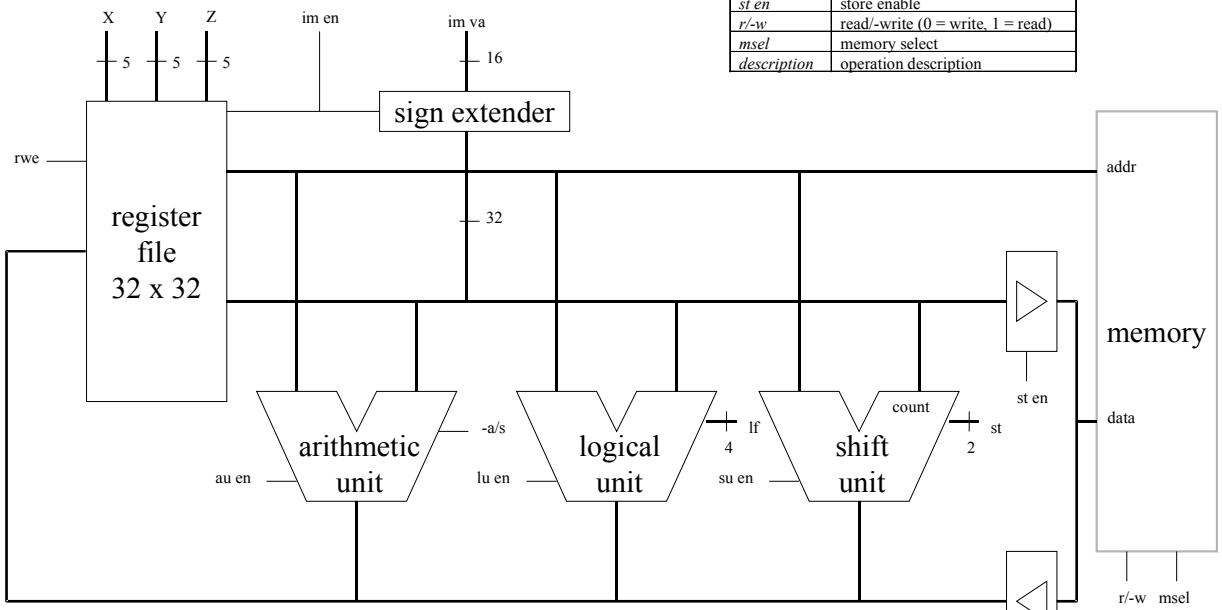
label	*instruction*	*comment*

MIPS Instruction Set

instruction	example	meaning
add	`add $1,$2,$3`	`$1 = $2 + $3`
subtract	`sub $1,$2,$3`	`$1 = $2 - $3`
add immediate	`addi $1,$2,100`	`$1 = $2 + 100`
multiply	`mul $1,$2,$3`	`$1 = $2 * $3`
divide	`div $1,$2,$3`	`$1 = $2 / $3`
and	`and $1,$2,$3`	`$1 = $2 & $3`
or	`or $1,$2,$3`	`$1 = $2 | $3`
xor	`xor $1,$2,$3`	`$1 = $2 xor $3`
and immediate	`andi $1,$2,100`	`$1 = $2 & 100`
or immediate	`ori $1,$2,100`	`$1 = $2 | 100`
xor immediate	`xori $1,$2,100`	`$1 = $2 xor 100`
shift left logical	`sll $1,$2,5`	`$1 = $2 << 5 (logical)`
shift right logical	`srl $1,$2,5`	`$1 = $2 >> 5 (logical)`
shift left arithmetic	`sla $1,$2,5`	`$1 = $2 << 5 (arithmetic)`
shift right arithmetic	`sra $1,$2,5`	`$1 = $2 >> 5 (arithmetic)`
load word	`lw $1, ($2)`	`$1 = memory [$2]`
store word	`sw $1, ($2)`	`memory [$2] = $1`
load upper immediate	`lui $1,100`	`$1 = 100 x` 2^{16}
branch if equal	`beq $1,$2,100`	`if ($1 = $2), PC = PC + 4 + (100*4)`
branch if not equal	`bne $1,$2,100`	`if ($1 ≠ $2), PC = PC + 4 + (100*4)`
set if less than	`slt $1, $2, $3`	`if ($2 < $3), $1 = 1 else $1 = 0`
set if less than immediate	`slti $1, $2, 100`	`if ($2 < 100), $1 = 1 else $1 = 0`
jump	`j 10000`	`PC = 10000`
jump register	`jr $31`	`PC = $31`
jump and link	`jal 10000`	`$31 = PC + 4; PC = 10000`

cycle	cycle number
X	register driven onto X bus
Y	register driven onto Y bus
Z	register written from Z bus
rwe	register write enable
im en	immediate enable on Y bus
im va	immediate value

au en	arithmetic unit enable
-a/s	-add / sub (0 = add, 1 = subtract)
lu en	logical unit enable
lf	logical function
su en	shift unit enable
st	shift type
ld en	load enable
st en	store enable
r/-w	read/-write (0 = write, 1 = read)
msel	memory select
description	operation description



logical functions

X	Y	out
0	0	lf_0
1	0	lf_1
0	1	lf_2
1	1	lf_3

shift types
0 = logical
1 = arithmetic
2 = rotate
+ count shifts right
- count shifts left