CS 3280/ECE 3280 Exam 2 May 2, 2022

Name: ______________________________
Last, First

Pawprint: _________________________

<u>Show all work</u> on this exam paper. **No** notes, or calculators.

Tally Sheet:

Problem	Max Score	Your Score
1	25	
2	25	
3	20	
4	30	
Total	100	

1. (25 points) Short-answer questions

a) (5 points) Which **main problem** does *cache memory* **alleviate** in a high-speed processor system?

Difference in speed between processor and main memory

b) (4 points) What is the main advantage and the main disadvantage of a fully associative cache (as compared to a direct mapped cache design)?

Main advantage of a fully associative cache over a direct mapped cache:

Associative cache is more flexible resulting in higher hit-ratio.

Main disadvantage of a fully associative cache over a direct mapped cache:

More hardware needed to implement associative cache.

c) (4 points) Which **main problem** does *virtual memory* **address** in a high-speed processor system?

- Amount of Main Memory available in a machine varies; program/data might not fit into memory
- Program contains hard-coded addresses; program cannot be moved to a different memory location

d) (4 points) What is stored in the page table used in a virtual memory system?

Page table holds virtual-to-physical address translations

e) (4 points) What is stored in a translation lookaside buffer (TLB) (be specific)?

TLB holds recent virtual-to-physical address translations.

f) (4 points) A TLB lookup is much faster than a page table lookup. Why is that (be specific)?

TLB lookup much faster then page table lookup because TLB is fast cache while actual page table is stored in slower main memory.

2. (25 points) Consider the following processor system:

A 68HC11 is connected to 3 memory chips: 1 RAM, 1 ROM, and 1 EEPROM.

The RAM is mapped to the address space: $1000 - $1FFF
The ROM is mapped to the address space: $2000 - $3FFF
The EEPROM is mapped to the address space: $4000 - $7FFF

a) (6 points) Calculate the size (number of memory cells in KiB or in decimal) of each memory chip.

RAM: 4096 bytes = 4 KiB

ROM: 8192 bytes = 8 KiB

EEPROM: 16384 bytes = 16 KiB

b) (6 points) How many address lines does each memory chip have?

RAM: 12

ROM: 13

EEPROM: 14

c) (13 points) Come up with the Boolean equations for the CE (Chip Enable) lines for the RAM, ROM, and EEPORM chips:

$CE_{RAM} = E * \overline{A15} * \overline{A14} * \overline{A13} * A12$

$CE_{ROM} = E * \overline{A15} * \overline{A14} * A13$

$CE_{EEPROM} = E * \overline{A15} * A14$

3. (20 points) For each of the following questions, assume that the registers are initialized to the following values (i.e., each part is independent of the others before the execution of an instruction). Give the contents of the registers after each instruction is executed. All numbers shown are HEX-numbers. DO NOT leave any box empty (even if the contents don't change) - empty boxes will be counted wrong! Also, do not use a DASH "-" in a box and do not forget the $-sign for hex numbers.

(5 points) A. RTS

	X	SP	PC	ML($01F0)	ML($01F1)	ML($01F2)	ML($01F3)	ML($01F4)
Initial	$CDEF	$01F1	$6789	$11	$22	$33	$44	$55
After	$CDEF	$01F3	$3344	$11	$22	$33	$44	$55

(5 points) B. JSR $1000

	X	SP	PC	ML($01F0)	ML($01F1)	ML($01F2)	ML($01F3)	ML($01F4)
Initial	$CDEF	$01F1	$6789	$11	$22	$33	$44	$55
After	$CDEF	$01EF	$1000	$67	$8C	$33	$44	$55

(5 points) C. PSHX

	X	SP	PC	ML($01F0)	ML($01F1)	ML($01F2)	ML($01F3)	ML($01F4)
Initial	$CDEF	$01F1	$6789	$11	$22	$33	$44	$55
After	$CDEF	$01EF	$678A	$CD	$EF	$33	$44	$55

(5 points) D. Specify the contents of the registers after the following instructions are fetched and executed:

1. INS
2. TSX
3. STX 0,X

	X	SP	PC	ML($01F0)	ML($01F1)	ML($01F2)	ML($01F3)	ML($01F4)
Initial	$CDEF	$01F1	$6789	$11	$22	$33	$44	$55
After	$01F3	$01F2	$678D	$11	$22	$33	$01	$F3

4. (30 points) Consider the following main program variable declarations:

```
            ORG         $B000               (Main Program Data Section)
DATA1       FCB         5
DATA2       FCB         6
SUM         RMB         1
```

A) (15 points) Write the main program that will call the subroutine SUB and will pass variables DATA1 and DATA2 to that subroutine using **"call-by-reference over the stack"**. The subroutine (which you will implement on the next page) will pass back a 1-byte data item through **"call-by-value over the stack"**. The main program will then store the data item passed back from the subroutine into SUM (do not forget to initialize any crucial processor registers and make sure that the stack will not grow infinitely or will suffer from underflow in case the subroutine is called multiple times.) PUSH/PULL instructions are allowed. **Note**: You may not need to use all of the rows. **Note**: The problem continues on the next page.

LABEL FIELD	**OPCODE FIELD**	**OPERAND FIELD**
	ORG	$C000
MAIN	LDS	#$01FF
	LDX	#DATA1
	PSHX	
	LDX	#DATA2
	PSHX	
	JSR	SUB
	PULA	
	STAA	SUM
	STOP	

B) (15 points) Write the subroutine SUB that will retrieve the variable references passed by the main program (that you implemented on the previous page) and pass the **1-byte sum** of those variables through **"call-by-value over the stack"** back to the main program. The subroutine must not use the labels DATA1, DATA2, or SUM. Also, make sure that the stack will not grow infinitely or will suffer from underflow in case the subroutine is called multiple times. PUSH/PULL instructions are allowed. **Note**: You may not need to use all of the rows.

Note: this subroutine needs to be implemented without using any static or dynamic variables

LABEL FIELD	OPCODE FIELD	OPERAND FIELD
	ORG	$D000
SUB	PULX	
	PULY	
	LDAA	0,Y
	PULY	
	ADDA	0,Y
	PSHA	
	PSHX	
	RTS	