Name:_______________________________________ Box:___________

Test 1 EC331 Embedded Systems (100 Point Maximum) Fall 2009 (KEH)

Closed notes, open Huang Textbook Only - 100 points max. 60 minutes

"Fill in the Blank"/"Multiple Choice" Questions

This is an objective test. You must have exactly the correct answer to each question for credit. (No partial credit given) All questions on this test apply to the 9S12C128 microcontroller.

1. (32 points – 1 point per blank) Fill in the chart below, indicating how many bytes must be READ from memory and how many bytes must be WRITTEN to memory by each instruction AFTER THE INSTRUCTION HAS BEEN FETCHED.

	Assembly Code	# Bytes Read From Memory	# Bytes Written to Memory
	LDX #$2A	0	0
	LDX $2A	2	0
	ADDA $4000	1	0
	STD $12,X	0	2
	RTI	9	0
a.	INC 50,X	_____	_____
b.	ADDA #$84	_____	_____
c.	JSR $4060,X	_____	_____
d.	JSR [$4060,X]	_____	_____
e.	MOVW A,X, 2,-Y	_____	_____
f.	MOVW #1234, 2,Y-	_____	_____
g.	INC [50,X]	_____	_____
h.	MUL	_____	_____
i.	PULX	_____	_____
j.	LSR [6,SP]	_____	_____
k.	LDY $1234, X	_____	_____
l.	LEAY $1234, X	_____	_____
m.	TARG: BRSET A,X,$20,TARG	_____	_____
n.	BSET $0400,$F0	_____	_____
o.	BCLR $0400,Y,$F0	_____	_____
p.	SWI	_____	_____

2. (20 points – 0.5 point per blank) Assuming the instructions below are executed in sequence, fill in the blanks below:

(A) LDAA #$79
ADDA #$89 After this ADDA instruction executes, the condition code (CCR) flags are:

H = ___ N = ____ Z = ____ V = ____ C = _____

Register A contains $__________

(B) DAA

After this DAA instruction executes, Register A contains $_______

and now the Carry condition code flag must be C = _____

(C) LDAA #$D5
ADDA #$B7 After this ADDA instruction executes, the condition code (CCR) flags are:

H = ___ N = ____ Z = ____ V = ____ C = _____

Register A contains $__________

(D) LDAA #$92
SUBA #$6B After this SUBA instruction executes, the condition code (CCR) flags are:

N = ____ Z = ____ V = ____ C = _____

Register A contains $__________

(E) LDAA #$3E
SUBA #$ED After this SUBA instruction executes, the condition code (CCR) flags are:

N = ____ Z = ____ V = ____ C = _____

Register A contains $__________

(F) LDD #$DEAD
SUBD #$BEEF After the SUBD instruction executes, the condition code (CCR) flags are:

N = ____ Z = ____ V = ____ C = _____

Register D contains $__________

(G) LDAA #$AD
CMPA #$35 After the CMPA instruction executes, the condition code (CCR) flags are:

N = ____ Z = ____ V = ____ C = _____

Register A contains $__________

(H) LDX #$1234
LEAX $4321,X
TFR X, D
ADDD #%0010100100000101 After the ADDD instruction executes, the condition code (CCR) flags are:
N = ____ Z = ____ V = ____ C = _____
Register D contains $__________ Register X contains $___________

3. (30 Points – 30/17 pts per blank) Given the following address map in an 9S12C128-based system, fill in the blanks:

Address	Contents
$0020	$DE
$0021	$02
$0022	$34
$0023	$02
$0024	$02
$0025	$35
$0041	$12
$0042	$34
$0043	$20
$0044	$00
$0045	$12
$0205	$10
$0206	$24
$0234	$00
$0235	$23
$0236	$00
$0237	$21
$0238	$05
$0239	$39
$02DE	$35
$02E0	$01
$02E1	$A5
$02E2	$36
$02E3	$FE
$1004	$89
$1005	$FE
$1024	$45
$1025	$67
$3437	$20
$3438	$00
$3439	$20
$343A	$02
$343B	$78
$3734	$37
$3735	$02

A. The following two instructions are executed:

```
LDX $0024
LDD 1,-X
```

Now A = $________ B = $________ X = $__________________

B. The following two instructions are executed

```
LDAA $02E0
LDY   #$0236
LDX  A,Y
```

Now X = $______________ and Y = $______________

C. The following instructions are

```
LDX  #36
LDY  -2,X
```

Now X = $_______________ and Y = $_______________

```
LDY   $0024
LDAA -1,Y
LDAB [-1,Y]
LEAX -1,Y
```

Now D = $______________ and X = $________________

D. The following sequence of instructions are executed:

```
LDS  #$1000
LDY  $1024
PSHY
PULA
PULB
PSHY
PSHB
PULY
```

Now Y = $_____________ S = $_____________ D = $______________ ($0FFF) = $______ ($0FFE) = $_______

E. Assume the memory map above, and that he following program fragment is executed from location START:

```
START:      LDAA #4
            CLRB
            LDX #$0239
LOOP1:      ADDB 1,X-
            DBNE  A,LOOP1
            STAB $0400
LOOP2:      BRA LOOP2
```

After the STAB instruction is executed, what is in A and X, and what is stored at location $0400 ?

A = $_______ X = $_________________ ($0400) = $_________

4. (18 points --- 1.5 pts per missing program blank.) Subroutine "**String_Compare**"
Subroutine "String_Compare" compares the first N elements of two null-terminated ASCII strings, where N is the length of the shorter of the two strings. (A null-terminated string must end in the value $00.) The calling sequence follows:
(1) Push the ***starting address*** of "*null-terminated*" ASCII String1 on the stack.
(2) Push the ***starting address*** of "*null-terminated*" ASCII String2 on the stack.
(3) Push the ***address of a RAM word*** which, upon return from the subroutine, will hold the address of the element in String1 where the two strings disagree, or it will hold a value of 0 if the first N characters of the two strings are identical.
The input arguments *must be cleaned off* of the stack after returning to the main program. Subroutine **StringCompare** must NOT disturb the values in the registers D, X, and Y back in the calling program. Note: the stack map entries will not be graded, *but you will get no credit for the entire problem if the stack map is not filled in*!) ***Begin by filling in a map of the stack after the PSHY executes in subroutine*** **String_Compare*****. See the right side of the page below.*** Then fill in the twelve blanks in the calling program "**String_Compare_Test**" and the subroutine "**String_Compare**" that appear below.

```
                 XDEF String_Compare_Test
                 ABSENTRY String_Compare_Test
                 ORG $400
Mismatch_Address:    DS.W 1
                  ORG $4000
STRING1:         DC.B "This is a test to compare two strings", 0
STRING2:         DC.B "This is a test to compare 2 strings",0   ;***After running, Mismatch_Address contains $401A***
String_Compare_Test:                                                  ;     Put Your Stack Map here:
                 LDS #$1000                                           ;(Not all the blanks will be filled in.)
                  LDX   #STRING1                                      ;  Addr              Contents
                  PSHX                                                ;$1000                ---
                 LDX   #STRING2                                       ;$0FFF             _________
                 PSHX                                                 ;$0FFE             _________
                 LDX #Mismatch_Address                                ;$0FFD             _________
                 PSHX                                                 ;$0FFC             _________
                 BSR String_Compare                                   ;$0FFB             _________
                 LEAS ________              ;Blank 1                  ;$0FFA             _________
STOP_HERE:       BRA STOP_HERE                                        ;$0FF9             _________
                                                                      ;$0FF8             _________
String_Compare: PSHD                                                  ;$0FF7             _________
                 PSHX                                                 ;$0FF6             _________
                 PSHY                                                 ;$0FF5             _________
                 LDX     ________,SP        ;Blank 2                  ;$0FF4             _________
                 LDY     ________,SP        ;Blank 3                  ;$0FF3             _________
NextChar:        TST 0,X                                              ;$0FF2             _________
                 BEQ  _______________       ;Blank 4                  ;$0FF1             _________
                 ____________________       ;Blank 5                  ;$0FF0             _________
                 BEQ NoMismatchFound                                  ;$0FEF             _________
                 LDAA ______,X+             ;Blank 6
                 LDAB ______,Y+             ;Blank 7
                 CBA
                 BNE  _______________________  ;Blank 8
                  BRA NextChar
NoMismatchFound:  LDX #0
                 STX _______________________   ;Blank 9
                 BRA DONE
MismatchFound:  DEX
                  STX ______________________   ;Blank 10
DONE:
                 PULY
                 ____________________________  ;Blank 11
                 PULD
                ____________________________   ;Blank 12
                  ORG $FFFE
                  DC.W String_Compare_Test
```