# Exercise 6 Problems

Do NOT use a calculator on this assignment. You will not be allowed to use one on the test or quizzes.

1. For the following programs, generate a stack diagram as done in class (values in memory used for the stack and the SP register) for each line that affects the stack items. For example, a new diagram should be drawn for a PSHA operation, but not a LDAA #$AA operation. Also, list the final contents of the registers used once the programs have finished. Addresses are given in comments.

a. ORG $C000

1: LDAA #$AA

2: LDAB #$BB

3: LDX #$CCDD

4: LDS #$3600

5: PSHA

6: PSHB

7: PSHX

8: PULB

9: PULX

10: PULA

11: SWI

b. ORG $C000

1: LDS #$3600 ; C000

2: LDAA #$AA ; C003

3: LDAB #$BB ; C005

4: JSR SUBRA ; C007

5: SWI ; C00A

6: SUBRB PSHB ; C00B

7: PULB ; C00C

8: RTS ; C00D

9: SUBRA PSHA ; C00E

10: JSR SUBRB ; C00F

11: PULA ; C012

12: RTS ; C013

c. ORG $C000

1: LDS #$3600 ; C000

2: LDAA #3 ; C003

3: LDAB #4 ; C005

4: JSR SumN ; C007

5: SWI ; C00A

6: SumN TSTA ; C00B

7: BEQ Ret ; C00C

8: PSHB ; C00E

9: PSHA ; C00F

10: DECA ; C010

11: JSR SumN ; C011

12: PULB ; C014

13: ABA ; C015

14: PULB ; C017

15: Ret RTS ; C018

1. Writing the functional portion of a subroutine is no different from writing a main program. The difference is passing parameters to a subroutine, returning the result, and saving registers. This problem will focus on one specific, simple function of adding two 2-byte numbers (Num1, Num2) together and returning the 2-byte result (Result). Assume all three are defined using DS.W’s. Write assembly code for the subroutine, the portion of a main program that calls it using the parameters listed in each part, and a stack frame. Do not use known memory locations for temporary storage.
   1. Num1 passed by value in D

Num2 passed by value in X

Result returned by value in D

Caller preserves nothing

Callee preserves nothing

* 1. Num1 passed by reference in X

Num2 passed by reference in Y

Result returned by value in D

Caller preserves nothing

Callee preserves nothing

* 1. Num1 passed by value in D

Num2 passed by value in stack

Result returned by value in D

Caller preserves X and Y

Callee preserves nothing

* 1. Num1 passed by reference in stack

Num2 passed by reference in stack

Result returned by value in stack

Caller preserves D, X, and Y

Callee preserves nothing

* 1. Num1 passed by reference in stack

Num2 passed by reference in stack

Result returned by value in stack

Caller preserves nothing

Callee preserves D and X