CSC 230 – Assignment #4, Part 5

Student Name: Jakob Valen
Student ID #: V00943160

Procedure	Test(s)	Total Operations	ALU	Jump	Branch	Memory	Other
Set_16x16	Tested with inserting 0x1 into row 5,col 7 and row 15,col 15	50	25(50%)	4(8%)	4(8%)	14(28%)	3(6%)
Get_16x16	Tested to get element at row 13,col 4 of original test array	26	12(46%)	2(8%)	3(12%)	7(27%)	2(8%)
Copy_16x16	Tested with original test case (original 16x16 byte array)	2484	1370(55%)	2(~0%)	305(12%)	518(21%)	289(12%)
Sum_neighbours	Tested with original byte array, sum of all elements around element at row 9, col 8	92	53(58%)	2(2%)	14(15%)	18(20%)	5(5%)
Bitmap_to_16x16	Original test file, a4-part-3.asm file was never modified	2682	1866(70%)	515(19%)	272(10%)	28(1%)	1(0%)
Draw_16x16	Original test file, a4-part-3.asm file was never modified	14931	9783(66%)	1027(7%)	1552(10%)	2312(15%)	257(2%)

For all programs we can see that most of the operations were ALU operations which are relatively fast instructions compared to memory instructions. Due to my implementation of parts 1&2 of this assignment the first 4 procedures all have very similar statistics since none of the procedures relies on another, and all deal with similar tasks. Bitmap_to_16x16 and

draw_16x16 are placing data within our 16x16 byte array and as a result are dealing with less memory which is reflected in the stats table above. As a result, Bitmap_to_16x16 and draw_16x16 are fairly fast programs compared to our other procedures as most operations are just ALU instructions.