On 2/1, I installed the heatsinks, followed by installing etcher and flashing the OS onto the micro SD. On 2/5 I started with the setup of the Raspberry Pi, hooking it up with a monitor, keyboard, and mouse. Signing into my apartment’s wifi, I updated the OS and started working on the ARM programming assignment. I found a mistake in the instructions, it listed that the command Control-W saved a program, when it turns out Control-W actually opens input into a help screen. Control-O is the correct command to save a program, and Control-X asks if a save is needed before exiting. I created the first program saved as first.s, assembled it using ‘as -o first.o first.s’ and linked it using ‘ld -o first first.o’; Running this using ‘./first’ does not return an output, as asked by the promt. After using the -g command to add a flag and re-assemble and link the file, I looked at the lines of code using the GDB debugger. The list command shows 10 lines of the program at a time, entering will show the next 10. I added a breakpoint to line 11 using ‘b 11’ and ran it to see where it stopped. I utilized scrot to create screenshots in the Raspberry Pi during this project, the first of which (Screenshot1\_Break11) shows the changes in the registers from the initial program. A value of 8 in the register1, with a value of 1 in r7 from the exit command. The 8 results from 4 and 5 being added in register 1, with 1 being subtracted from it.

For part B, I read through the math for A B C and D, and created the following program from the required equation, A = (A+B) -(C\*D), where A = 10, B =11, C = 7, and D = 2.

@Second Program arithmetic1

.section .data

.section .text

.globl \_ start

\_start:

mov r1, #10 @ load r1 (A) with 10

add r1, r1, #11 @ Add 11 (B) to r1(A)

mov r2, #7 @ load r2 with 7(C)

mov r3, #2 @load r3 with 2(D)

mul r4, r2, r3 @ multiply r2 by 2(D)

sub r1, r4 @subtract r2(product of C\*D) from r1 (A+B).

mov r7, #1 @Program Termination: Exit Syscall

svc #0 @Program Termination: Wake Kernel

.end

The logic behind each line is listed; first A is loaded into register1 with a value of 10, and the value of B is added to it. In a second register r2, the value of C, 7, is loaded, with the value of D loaded into register3. Inside of register 4, these two values are multiplied and stored. Then, the value of that product is subtracted from the sum of A and B, stored in register1. The last two lines end the program the same way as the previous part. A second screenshot (Part2RegistersAndCode) was included to show the arithmetic1 code in the Raspberry Pi, as well as the listed registers at the end of the program. A break was added in on line 12 to check the registers before the program exited in the GDB debugger. Register1 shows the correct final total, 7, with the other three registers (r2/r3/r4) showing the value of C, D and C\*D respectively. Checking these totals, we find that the program did exactly what was expected when following A= (10+11)-(7\*2), a total of A = 7.