Exercises

Consider the assembly code which the compiler generates for a C function. Explain what each assembly instruction does and describe what data is in the register.

1. ;;;5 void fn(int8\_t \* a, int32\_t \* b, float \* c) {

000000 b5f0 PUSH {r4-r7,lr}

Saves registers r4-r7 and LR on stack to save registers which need to be restored to their original values.

1. 000002 b085 SUB sp,sp,#0x14

Allocate 0x14 (20) bytes of space on the stack for local variables.

1. 000004 4604 MOV r4,r0

Save first argument (a) in r4.

1. 000006 460d MOV r5,r1

Save second argument (b) in r5.

1. 000008 4616 MOV r6,r2

Save third argument (c) in r6.

1. ;;;6 volatile int8\_t a1, a2;

;;;7 volatile int32\_t b1, b2;

;;;8 volatile float c1, c2;

;;;9

;;;10 a1 = 15;

00000a 270f MOVS r7,#0xf

Load 15 (0xf) into register for a1.

1. ;;;11 a2 = -14;

00000c 200d MOVS r0,#0xd

Load 14 into register for a2.

1. 00000e 43c0 MVNS r0,r0

Complement 14 to -14 using Move Inverse instruction.

1. 000010 9004 STR r0,[sp,#0x10]

Store a2 on stack at offset 0x10.

1. ;;;12 \*a = a1\*a2;

000012 9804 LDR r0,[sp,#0x10]

Reload a2 from stack into r0

1. 000014 4378 MULS r0,r7,r0

Multiply a2 by a1, putting result in register for a1.

1. 000016 b240 SXTB r0,r0

Sign extend a1.

1. 000018 7020 STRB r0,[r4,#0]

Store **one-byte result** in memory location pointed to by r4, which is argument a.

1. ;;;13

;;;14 b1 = 15;

00001a 200f MOVS r0,#0xf

Load register r0 (for variable b1) with 15 (0xf).

1. 00001c 9003 STR r0,[sp,#0xc]

Save on stack at offset 0xc.

1. ;;;15 b2 = -14;

00001e 200d MOVS r0,#0xd

Load 14 into register for b2.

1. 000020 43c0 MVNS r0,r0

Complement 14 to -14 using Move Inverse instruction.

1. 000022 9002 STR r0,[sp,#8]

Store b2 on stack at offset 8.

1. ;;;16 \*b = b1\*b2;

000024 9902 LDR r1,[sp,#8]

Load r1 with b2 from stack.

1. 000026 9803 LDR r0,[sp,#0xc]

Load r0 with b1 from stack.

1. 000028 4348 MULS r0,r1,r0

Multiply b1 and b2.

1. 00002a 6028 STR r0,[r5,#0]

Store **longword result** in memory location pointed to by r5, which is argument b.

1. ;;;17

;;;18 c1 = 15;

00002c 4809 LDR r0,|L1.84|

Load r0 with floating point value of 15 from literal pool.

1. 00002e 9001 STR r0,[sp,#4]

Store in c1’s location on the stack.

1. ;;;19 c2 = -14;

000030 4809 LDR r0,|L1.88|

Load r0 with floating point value of 15 from literal pool.

1. 000032 9000 STR r0,[sp,#0]

Store in c2’s location on the stack.

1. ;;;20 \*c = c1\*c2;

000034 9900 LDR r1,[sp,#0]

Load r1 with c2.

1. 000036 9801 LDR r0,[sp,#4]

Load r0 with c1.

1. 000038 f7fffffe BL \_\_aeabi\_fmul

Call floating point multiply instruction to multiply c1 and c2.

1. 00003c 6030 STR r0,[r6,#0]

Store longword result (returned in register r0) to memory pointed to by r6, which is argument c.

1. ;;;21

;;;22 }

00003e b005 ADD sp,sp,#0x14

Deallocate the stack space for this function.

1. 000040 bdf0 POP {r4-r7,pc}

Restore registers r4 through r7 to original values by popping them off of the stack and then return from subroutine (return address also saved on stack).