**Errata**

1) ARM Assembly Programming Example 2

* stmfd sp!, {r0-r4, lr} should be removed;
* ldmfd sp!, {r0-r4, pc} should be replaced with: **ldmfd sp!, {pc}** or **mov pc, lr**

2) Call Assembly Function from C:  
 (3). Compile-link t.c and ts.s into an executable file

* arm-none-eabi-ld –T t.ld –o t.elf t.o ts.o should be replaced with

arm-none-eabi-ld -T t.ld -o t.elf ts.o t.o

**REMARKS**

1) Function Call Convention in C:   
 (3). save LR, FP(r12) on stack, establish stack frame (FP point at saved LR)

* The frame pointer ($fp) points to the start of the stack frame and does not move for the duration of the subroutine call. This points to the base of the stack frame, and the parameters that are passed in to the subroutine remain at a constant spot relative to the frame pointer.

2) add fp, sp, #4 // FP point at saved LR

* Keep in mind that we are working with a full descending stack. This means that our address decreases when pushing variables on the stack. As mentioned the highest register is pushed to the highest address and the lowest register to the lowest address. In this case the link register (r14) comes before the frame pointer (r12). Because the exclamation mark was used during the stmfd call the stack pointer points now to the address of the frame pointer. Adding this address by 4 will result in the link register address.