Study Questions (Chapter 04 – Part 1)

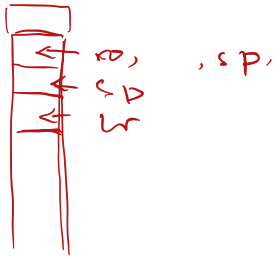
1. The program in Tutorial 14 utilizes an FD stack. Modify the program to make it utilize an ED stack. Graphically show the content of the stack and the location of the SP after executing each instruction in your program.
   Type, assemble, and run your modified program to make sure that it works properly.

2. The program in Tutorial 14 utilizes an FD stack. Modify the program to make it utilize an FA stack. Graphically show the content of the stack and the location of the SP after executing each instruction in your program.
   Type, assemble, and run your modified program to make sure that it works properly.

3. The program in Tutorial 14 utilizes an FD stack. Modify the program to make it utilize an EA stack. Graphically show the content of the stack and the location of the SP after executing each instruction in your program.
   Type, assemble, and run your modified program to make sure that it works properly.

```
Main    ADR sp,Stack ;set up r13 as the stack pointer
        MOV r0,#124 ;set up a dummy parameter in r0
        MOV fp,#123 ;set up dummy frame pointer
        STR r0,[sp,#-4]! ;push the parameter
        BL Sub ;call the subroutine
        LDR r1,[sp],#4 ;pop the parameter
        Loop B Loop ;wait here (endless loop)

Sub     STMFD sp!,{fp,lr} ;push frame-pointer and link-register
        MOV fp,sp ;frame pointer at the base of the frame
        SUB sp,sp,#4 ;create a local variable in the stack frame
        LDR r2,[fp,#8] ;get the pushed parameter
        ADD r2,r2,#120 ;do a dummy operation on the parameter
        STR r2,[fp,#-4] ;store it in the local variable
        ADD sp,sp,#4 ;remove the local variable
        LDMFD sp!,{fp,pc } ;restore frame pointer and return
        DCD 0x0000 ;clear memory
        DCD 0x0000
        DCD 0x0000
        DCD 0x0000
Stack   DCD 0x0000 ;start of the stack
```



```
Main    ADR sp,Stack ;set up r13 as the stack pointer
        MOV r0,#124 ;set up a dummy parameter in r0
        MOV fp,#123 ;set up dummy frame pointer
        STR r0,[sp,#-4]! ;push the parameter
        BL Sub ;call the subroutine
        LDR r1,[sp],#4 ;pop the parameter
        Loop B Loop ;wait here (endless loop)

Sub     STMFD sp!,{fp,lr} ;push frame-pointer and link-register
        MOV fp,sp ;frame pointer at the base of the frame
        SUB sp,sp,#4 ;create a local variable in the stack frame
        LDR r2,[fp,#8] ;get the pushed parameter
        ADD r2,r2,#120 ;do a dummy operation on the parameter
        STR r2,[fp,#-4] ;store it in the local variable
        ADD sp,sp,#4 ;remove the local variable
        LDMFD sp!,{fp,pc } ;restore frame pointer and return
        DCD 0x0000 ;clear memory
        DCD 0x0000
        DCD 0x0000
        DCD 0x0000
Stack   DCD 0x0000 ;start of the stack
```